Testing for Network Effects in Field Experiments: Examples from Legislative Studies

Sayali Phadke Bruce Desmarais

March 15, 2016

Abstract

Most social processes involve complex interaction and dependence among units in a network. The stable unit treatment value assumption (SUTVA)–the assumption that a unit's outcome is unaffected by other units' treatment statuses—is required in conventional approaches to causal inference. When SUTVA is violated, as in networked social interaction, treatment effects spread to control units through the network structure. We evaluate the evidence for spillover effects in data from three field experiments on US state legislatures. Randomized field experiments represent the gold standard in causal inference when studying political elites. It is not possible to bring political elites into the lab, and causal identification with observational data is fraught with problems. We propose new specifications for treatment spillover models, and construct networks through geographical or ideological proximity and co-sponsorship. Considering different combinations of spillover models and networks, we evaluate the robustness of recently developed non-parametric tests for interference. The approaches we illustrate can be applied to any experimental setting in which interference is suspected.

1 Introduction

Networks are integral parts of human interaction and hence social science research. If one unit in a network gets treated, the effect may trickle down throughout network. The currently established framework for causal inference relies on SUTVA (Stable Unit Treatment Value Assumprion). It assumes that whether or not one person/unit/node is treated, does not affect any other unit. However, SUTVA breaks down in a network setting.

It is therefore imperative to take the interference structure into account. Rather, in policy planning or designing marketing campaigns, a researcher may be interested in studying the propogation of treatment effect itself.

In field experiments on social groups, interference may be substantial. In this project we intend to study intererence models for randomized experiments coducted on social networks and causal inference basis this.

To understand, explain and predict social phenomena, social scientists typically look to individual actors' attributes to explain their behavior (e.g., an increase in an individual's wealth will result in a decrease in that individual's support for government spending on social welfare), or to attributes of the macro context (e.g., an increase in the unemployment rate will lead an individual's support for the party of the president to decrease). However, these two conceptual approaches to explaining individual behavior leave out a potentially powerful class of social dynamics – interpersonal influence. That is, the behavior of one individual may depend upon the behavior of one or more others (e.g., a person may decide to vote due to their friends claiming to have voted (**?**)). Inferences regarding influence involve the analysis of individual behaviors and the behaviors of those adjacent or nearby in some contact network. However, as in most settings, it is generally not possible to identify the causal effects that map onto the process of social influence in observational data (**?**). As such, we need experimental methods to identify causal influence effects.

1.1 tasks

- Points about why it is interesting to study propagation. (BD)
- Outline of the paper (SP)

2 Background

Review of relevant methodological work and substance.

2.1 tasks

- **PRIORITY:** Explain the Bowers et al method in our own words (SP). Describe the algorithm in enough detail that one could implement it based on our explanation.

  Below we see a description of the Fisherian inference algorithm as described in the Bowers et al paper. Here the aim is to test for a model $\mathscr{H}$ against the data we observe. Ths hypothesized model can include interference effect

  1. As discussed in the Method section, we assume the "sharp null hypothesis of no effects" i.e. we assume that the treatment assignment has no effect on any unit
  2. We begin by specifying the causal model which describes the change in potential outcomes when treatment assignment changes from $\mathbf{u}$ to $\mathbf{w}$; $\mathscr{H}(y_{i,\mathbf{u}}, \mathbf{w}, \theta)$. If spillover effects are theoretically motivated, we must also specify treatment assignment for u_j and w_j where $i \neq j$
  3. The potential outcomes from the causal model must be mapped to the observed outcomes $y_{\mathbf{z}}$. The treatment assignment in the experiment ($\mathbf{z}$) must be mapped to the uniformity trial which is based on a no-treatment assignment i.e. every unit is a control unit. In this condition, all z_is are zero and we refer to this as the baseline condition. Uniformity trial is specified as $\mathscr{H}(y_{\mathbf{z}}, \mathbf{0}, \theta) = \mathbf{y}_0$. These

mappings should give us the hypothesized value of θ and a data adjusted to the model

4. The test statistic we consider is $\mathscr{T}$ and the key characteristic is that it should be a small value when dsitribution of treated and control outcomes in the adjusted data (mentiond in point 3) are similar. $\mathscr{T}$ should be larger when distributions are dissimilar. Since we want the similarity to depend on not just the center but also higher-order moments of a distribution (spread, skewness etc.), we need a sensitive measure. Bowers et al recommend using the Kolmogorov-Smirnov (KS) test statistic.

   As noted in footnote 12 of the paper, KS statistic is the maximum difference between the empirical cumulative distribution functions (ECDFs) of treated (F_1) and control (F_0) units. So under the baseline condition,

$$\mathscr{T}_{\mathbf{y}_0,\mathbf{z}} = \max_{1\leq i\leq n} [F_1(y_i,0) - [F_0(y_i,0)]$$

   where $F(x) = \frac{1}{n}\sum_{i=1}^{n} I(x_i \leq x)$ is the proportion of x below x_i

5. We must form hypothesis for interference. Here we assume that treatment only spreads through edges and the spillover effect only depends on the number of neighbours treated. The model for interference is explained in the immediately next section. However, here we note that the spillover effect is modeled using a growth curve $\beta + (1-\beta)e^{-\tau^2 \mathbf{z}^T \mathbf{S}}$

6. Now we generate the distribution of test statistic under our hypothesis. The exact distribution is specified by computing $t_k = \mathscr{T}(\mathbf{y}_0, \mathbf{Z}_k)$ for each $\mathbf{Z}_k \in \Omega$.

Essentially, we are evaluating this for every possible treatment assignment. Alternatively, we can use sampling methods and limit theorems to estimate the distribution from data.

7. Finally, the p-value for our test can be calculates using the following formula:

$$\frac{\sum_{k=1}^{abs(\Omega)} I(x_i > t_k)}{abs(\Omega)}$$

- Paragraph on each category of papers that serve as relevant background (SP)
  - Interference models (diffusion, propagation) (SP–Review)
  - Experiments on networks (applications) (SP–Review)
  - Approaches to inference or estimation with propagation (SP–Review)
  - Potential outcomes framework (SP – find papers & Review)
  - Review of political networks (SP–Review)
  - Review of field experiments (SP–Review)
    - (**??????**)

3 Research Design

We plan to re-analzye data from past field experimental studies to understand how conclusions regarding direct effects and interference effects depend upon the network structure.

3.1 tasks

Develop a list of alternative propagation models to evaluate. (SP)

We vary the propagation models along three key features:

1. Distance from the nearest treated node (d_i)
2. Number/proportion of treated nodes at each d_i
3. Form

The key factors that we consider important in building propagation models are:

1. Grographical proximity
2. Ideological similarity
3. Co-sponsorship

Each of these factors is such, that a treatment such as the message sent through emails in New Hampshire, would possibly spread to untreates units as well. Legislators from adjoining districts may affect each other's opinions through geographic proximity as well as potentially via common issues faced by citizens in their constituencies.

Ideological similarity is a tricky variable because it could be hard to distinguish that from party affiliation. In the New Hampshire paper, there is a need to include this effect in the model since matched pairs are created based on party affiliation. If a Republican candidate receives the treatment, the chances that through various communication channels,

he/she will convey the message to the control group candidate from the same party and district, are very high.

Finally, serving on the same committee can also contribute to spreading the effect of a treatment. We must test for any dependence across these three fators before incorporating them into our model. Therefore it is important that we propose and test propogation models that consider the spread of treatment through our network.

**Notes:

1. Ideological similarity: This should get highest priority in modeling, since I believe, it would be easiest to affect an undecided legislator's vote through similarity in ideas and belief about citizen's issues and how to resolve them. I would propose that we model immediate neighbours to have a 50
2. Grographical proximity: An untreated legislator from adjoining district would be
3. Co-sponsorship: Serving on the same committee increases the chances of

4 Analysis

*Could we use the idea of communities to model spread of treatment across the network?

Replicating results from the Nickerson paper

The first table contains results of balance test for pre-treatment covariates in the analysis. The p-values are calculates using simple logit regressions. This table shows that there is covariate balance across treatment conditions

	Treatment	Control	p-value
Republican	40%	40%	1
Constituent support for spending	40.8%	41.3%	0.8
Constituent support for health care	54.2%	52.7%	0.34
Bush vote-share 04	51.3%	49.3%	0.59
Member vote-share 06	60.6%	61.2%	0.86
Running for re-election	91.4%	88.6%	0.69
Running unopposed	59.4%	58.1%	0.92
Supported prior health care bill	54.3%	51.4%	0.81

Table 1: Table 1: Randomization checks

The next two tables present regression results. We are modeling the likelihood of voting in favor of SB 24. In the first regression, we study whether the treatment effects differ substantively across districts where support for the governor's spending proposals was low and ones where it was not. This is our key independent variable. Each regression uses a Probit model with standard errors clustered on the 35 matched pairs on which the randomization was based. We use the Zelig program/package in R to incorporate clustered standard errors into the model. The original analysis was performed in STATA. We notice that our estimates are the same as those in the original paper and standard errors are very close as well.

	Coefficients	SEs
Constant	0.76	0.33
Treatment	0.16	0.49
Low support for spending	0.70	0.60
Low support*Treatment	-1.49	0.76

Table 2: Table 2A: Regression results without controls

In the second regression, we also control for whether the legislator was a Republican and the 2004 Presidential election results for the given district.

	Coefficients	SEs
Constant	-0.16	1.65
Treatment	-0.07	0.70
Low support for spending	1.50	0.83
Low support*Treatment	-1.86	1.04
Republican legislator	-1.61	0.68
2004 Democratic two-party presidential vote share	4.06	2.78

	Coefficients	SEs
Constant	-0.16	1.65
Treatment	-0.07	0.70
Low support for spending	1.50	0.83
Low support*Treatment	-1.86	1.04
Republican legislator	-1.61	0.68
2004 Democratic two-party presidential vote share	4.06	2.78

Table 3: Regression results with controls

Present original results from studies that we replicate: Coppock

The Coppock (2014) paper builds upon the New Mexico Legislator experiment conducted by Butler and Nickerson (2011). The next two figures replicate analysis from the original paper, as shown in the Coppock (2014) paper. Figure 1 below replicates the result under the assumption that indrect effecrs are exactly zero. X-axis represents the proposed values for direct treatment effect, Y-axis represents simulated p-values and the colouring is according to the p-values.

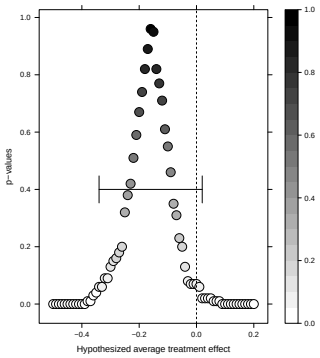

Figure 1: Figure 1

Figure 2 shows the heterogeneous effects of treatment. X-axis represents hypothesized effects in higher-support districts as against the hypothesized effects in lower-support districts on Y-axis. Once again, the colour scale indicates the p-value for each pair of hypotheses. Darker region indicates a higher p-value. We observe maximum p-value at effect values (0.05, -0.37) in higher-support and lower-support regions respectively.

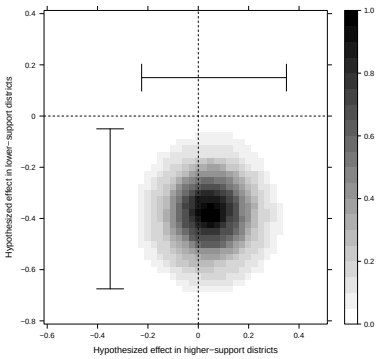

Figure 2: Figure 1

Present results when we assume some form of interference

Explore how alternative assumptions regarding interference change results

4.1 tasks

- Replicate Bergan. (SP)
- Find other network data for the New Mexico legislature. (BD)
- Geography and ideology data for New Hampshire (BD)
- Produce cosponsorship, ideology and geography estimates for both Bergan and Nickerson (SP & BD)
- For at least two spreading models (SP & BD)
- Replicate Nyhan (SP)

Records of standing committee membership in the 16 standing committees in place during the 2008 regular session was obtained from the New Mexico Legislative Council Service Librarian.

5 Appendix

5.1 Appendix 1

In this section, we will look at user-defined R-functions that replicate the Bowers et. al. methodology. There will be four steps in this:

- A function to transform the observed outcomes into potential outcomes for any treatment assignment w

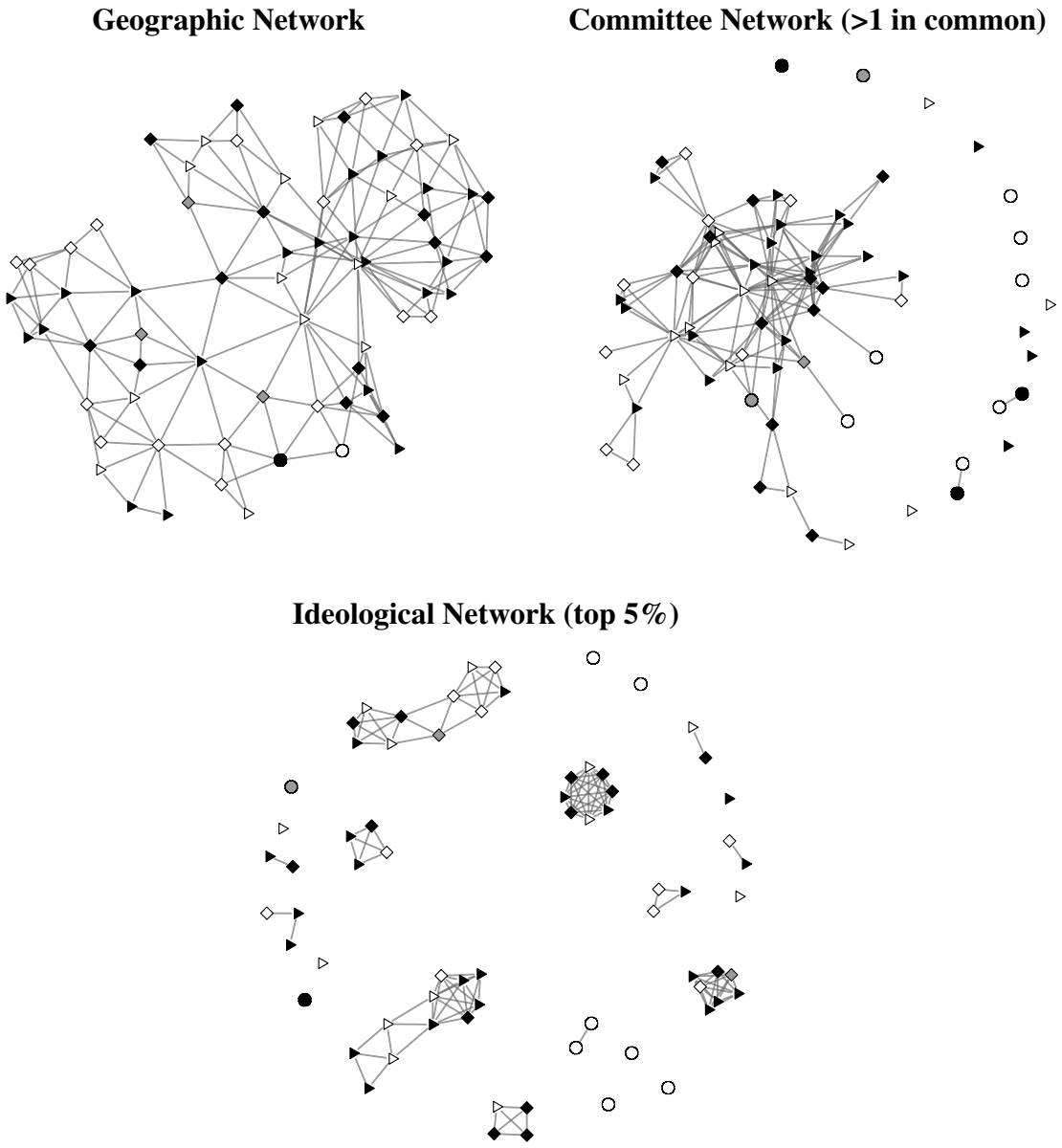

Figure 3: Different networks among New Mexico legislators. Colors denote outcome: black means voted with district, gray means abstained, white means voted against. Shape denotes treatment status. Triangles are treated. Squares are adjacent to treated. Circles are isolated from treatment

- A function to separate the hypothesized treatment effect
- A function to calculate test statistic
- A function to calculate the p-value.

The results from the ks.tes function in R for calculating Kolmogorov-Smirnoff test statistic are verified with that in Footnote 12 of the paper.

Function 1: calculating potential outcomes

```
set.seed(132)
library(doParallel)
library(foreach)
library(kSamples)
library(network)
library(permute)

#### Potential outcomes ####

#### Transform uniformity trial outcome into observed outcome
unif.to.z <- function(z, S, y.0, beta, tau){
  # z: observed treatment assignment
  # S: adjacency matrix
  # y.0: outcome vector for uniformity trial
  # beta: growth curve parameter
  # tau: rate of growth parameter

  scalar <- as.vector(t(z)%*%S)
  spillover <- rep(NA, n)
```

```
20
21    spillover <- beta + ((1-z) * (1-beta) * exp(-tau^2 * scalar))
22
23    # This is equation 4
24    h.y0.z <- spillover*y.0
25  }
26
27  #### Transform observed outcome into uniformity trial outcome
28  z.to.unif <- function(z, S, y.z, beta, tau){
29    # z: initial treatment assignment
30    # S: adjacency matrix
31    # y.z: observed outcome vector
32    # beta: growth curve parameter
33    # tau: rate of growth parameter
34
35    scalar <- as.vector(t(z)%*%S)
36    spillover <- rep(NA, n)
37
38    # Equation (3)
39    spillover <- beta + ((1-z) * (1-beta) * exp(-tau^2 * scalar))
40
41    # This is equation 5
42    h.yz.0 <- (1/spillover)*y.z
43  }
44
45  #### Transform observed outcome into outcome for ANY other assignment w
46  z.to.w <- function(z, S, w, y.z, beta, tau){
47    # z: initial treatment assignment
```

```
  # S: adjacency matrix
  # w: new treatment assignment
  # y.z: vector of outcomes for z
  # beta: growth curve parameter
  # tau: rate of growth parameter

  scalar.z <- as.vector(t(z)%*%S)
  scalar.w <- as.vector(t(w)%*%S)

  spillover.z <- rep(NA, n)
  spillover.z <- beta + ((1-z) * (1-beta) * exp(-tau^2 * scalar.z))


  spillover.w <- rep(NA, n)
  spillover.w <- beta + ((1-w) * (1-beta) * exp(-tau^2 * scalar.w))


  # Below is the actual function that transforms observed outcomes into
    potential outcomes
  # Equation (6)

  h.z.to.w <- (spillover.w / spillover.z) * y.z
}

#### Testing and p-value calculation ####

p.val <- function(z, y.z){

```

```
75  cl <- makeCluster(4) #Setup for parallel computing
76  registerDoParallel(cl)
77
78  # Calculate the outcome vector after taking away the effect of
      treatment
79  y.0 <- z.to.unif(z=z, S=S, y.z=y.z, beta=beta, tau=tau)
80
81  # Calculate test statistic
82  test.stat <- ks.test(y.0[z==1], y.0[z==0],
83                       alternative = "g")$statistic
84  sign <- noquote(strsplit(names(test.stat), NULL)[[1]])[3]
85  if(sign=="+"){
86    test.stat <- test.stat
87  }else{
88    test.stat <- test.stat*-1
89  }
90
91  # Calculate a vector of test statistic using permutations
92  results <- foreach (i = 1:perms) %dopar%{
93    require(permute)
94    perm.z <- z[sample(1:length(z),length(z),rep=F)]
95    perm.test.stat <- ks.test(y.0[perm.z==1], y.0[perm.z==0],
96                              alternative = "g")$statistic
97    sign <- noquote(strsplit(names(perm.test.stat), NULL)[[1]])[3]
98
99    if(sign=="+"){
100     return(perm.test.stat)
101   }else{
```

```
      return(perm.test.stat*-1)
    }
  }
  stopCluster(cl)

  # A vector of test statistics
  all.test.stat.vals <- unlist(results)

  # Calculating p-value
  pval <- sum(all.test.stat.vals > test.stat)/perms
  return(pval)
}
```

5.2 Appendix 2

Below code replicates the Coppock results using the framework setup in the Bowers replication code

```

##############################################
#### Replicating Coppock using our code ####
##############################################

##############################################
#### Replicating Coppock using our code ####
##############################################

rm(list=ls())
```

```r
set.seed(312)

library(doParallel)
library(foreach)
library(kSamples)
library(network)
library(permute)
library(wnominate)


#### Read the original Butler and Nicketson data
#### This is the New Mexico dataset

data <- read.table("nm.replication.tab", sep="\t", header=TRUE)

z <- data$treatment #observed treatment
y.z <- data$sb24 #observed outcome
n <- length(y.z) #number of observations
t <- length(z[z==1]) #number of treated units
perms <- 10000 #number of permutations to use in generating expected
    exposure
perms.test <- 1000 #number of permutations used in testing


#### Generate Similarity Scores (this code taken from CoppockJEPS_
    datapreparation.R)

nmhouse2008 <-read.csv("CoppockJEPS_rollcalldata.csv")
```

```r
bills <- data.frame(nmhouse2008[5:21])

## Nominate Scores

bills_nona <- bills
bills_nona[bills_nona==99] <- NA
rollcalls <- rollcall(bills_nona)
nominate_scores <- wnominate(rollcalls, polarity=c(1, 2), minvotes=10)
dwnom_scores <- nominate_scores$legislators$coord1D

get.similarity <- function(x, y){
  return((2-abs(x-y))/2)
}


## Create an adjacency/similarity matrix using ideology
S.ideo <- matrix(NA, ncol=70, nrow=70)
for (i in 1:70){
  for (j in 1:70){
    S.ideo[i,j] <- get.similarity(dwnom_scores[i], dwnom_scores[j])
  }
}
diag(S.ideo) <- 0
S.ideo[is.na(S.ideo)==T] <- 0


#### Generate expected exposure
perm <- replicate(perms, z[sample(1:length(z),length(z),rep=F)])
```

```
65
66 expected.exp <- rep(NA, n)
67 for (i in 1:n){
68   K <- length(which(perm[i,]==0))
69   L <- length(which(perm[i,]==1))
70
71   perm.k <- perm[,which(perm[i,]==0)]
72   perm.l <- perm[,which(perm[i,]==1)]
73
74   if (z[i] == 1){
75     sums <- rep(NA, K)
76     for (k in 1:K){
77       sums[k] <- sum(S.ideo[i,]*perm.k[,k])
78       }
79     expected.exp[i] <- sum(sums)/K
80     } else {
81       sums <- rep(NA, L)
82       for (l in 1:L){
83         sums[l] <- sum(S.ideo[i,]*perm.l[,l])
84         }
85       expected.exp[i] <- sum(sums)/L
86     }
87 }
88 rm(perm.k)
89 rm(perm.l)
90 rm(sums)
91
92
```

```r
#### Generate expected and net exposure
#### This is the spillover effect model

indirect.treatment <- function(permutation, adj.mat){ #any treatment assignment vector and adjacency matrix can be used
  # permutation: can be the initial treatment assignment or a permutation
  raw.exp <- rep(NA, n)
  for (i in 1:n){
    raw.exp[i] <- sum(adj.mat[i,]*permutation)
    }

  net.exp <- raw.exp - expected.exp
  standard.exp <- (net.exp - mean(net.exp))/sd(net.exp) #this is the spillover or indirect effect
  return(standard.exp)
}


#### We now model the uniformity trial transformation

z.to.unif <- function(z, outcome, beta1, beta2, permutation, adj.mat){
  # z: initial treatment assignment
  # outcome: vector of direct treatment outcomes
  # beta1: direct treatment effect parameter
  # beta2: indirect treatment effect parameter
  # permutation: matrix of permutations of z
  # adj.mat: adjacency matrix
```

```
118
119   exposure <- indirect.treatment(permutation, adj.mat)
120   # This is equation 5
121   h.yz.0 <- outcome - (beta1*z) - (beta2*exposure)
122   return(h.yz.0)
123 }
124
125
126 #### Testing and p-value calculation
127
128 beta1s <- seq(from=-.5, to=0.5, by=.1)
129 beta2s <- seq(from=-.5, to=0.5, by=.1)
130 pvals <- matrix(NA, length(beta1s), length(beta2s))
131
132 cl <- makeCluster(4) #Setup for parallel computing
133 registerDoParallel(cl)
134
135 pvalues.ideology <- foreach (i = 1:length(beta1s)) %do% {
136   abc <- foreach (j = 1:length(beta2s)) %do% {
137
138     # Calculate the outcome vector after taking away the effect of
      treatment
139     y.0 <- z.to.unif(z = z, outcome = y.z, beta1 = beta1s[i], beta2 =
      beta2s[j], permutation = perm, adj.mat = S.ideo)
140
141     # Calculate observed test statistic
142     exposure <- indirect.treatment(permutation = z, adj.mat = S.ideo)
```

```
    test.stat <- sum((lm(y.0 ~ z + exposure, na.action = na.omit)$resid)^2)

    # Calculate a vector of test statistic using permutations

    results <- foreach (k = 1:perms.test) %dopar% {
      require(permute)
      perm.z <- z[sample(1:length(z),length(z),rep=F)] #Each time we sample a permutation of z
      perm.y.0 <- z.to.unif(z = perm.z, outcome = y.z, beta1 = beta1s[i], beta2 = beta2s[j], permutation = perm, adj.mat = S.ideo)
      perm.exposure <- indirect.treatment(permutation = perm.z, adj.mat = S.ideo)

      perm.test.stat <- sum((lm(perm.y.0 ~ perm.z + perm.exposure, na.action = na.omit)$resid)^2)
      }


    # A vector of test statistics
    all.test.stat.vals <- as.numeric(unlist(results))

    # Calculating p-value
    pval <- sum(all.test.stat.vals > test.stat)/perms
  }
  as.numeric(unlist(abc))
}

```

```
stopCluster(cl)

for (i in 1:length(beta1s)){
  pvals[i,] <- unlist(pvalues.ideology[i])
}

summary(pvals)


#### Plotting p-values
library(lattice)

find_breaks <- function(x){
  breaks <- rep(NA, length(x)-1)
  for(i in 1:length(breaks)){
    breaks[i+1] <- x[i]!=x[i+1]
  }
  return(which(breaks))
}

direct_breaks <- find_breaks(apply(pvals, MARGIN=1, FUN=max) >.05)
indirect_breaks <- find_breaks(apply(pvals, MARGIN=2, FUN=max) >.05)

graph.frame <- expand.grid(x=beta1s, y=beta2s)
graph.frame$z <- as.vector(pvals)
col.l <- colorRampPalette(c('white', 'black'))(1000)
depth.breaks <- do.breaks(c(0,1), 20)
fig2 <- levelplot(z~x*y, graph.frame, cuts=20, col.regions=col.l,
```

```
                    colorkey=FALSE,
                    at=depth.breaks,
                    ylab = "Hypothesized indirect effect",
                    xlab = "Hypothesized direct effect",
                    scales=list(x=list(at=round(seq(-.5, .5, by=.1),
    digits=1), labels=round(seq(-.7, .2, by=.1), digits=1)),
                                y=list(at=round(seq(-.5, .5, by=.1),
    digits=1), labels=round(seq(-.7, .2, by=.1), digits=1))),
                    panel = function(...) {
                      panel.levelplot(...)
                      panel.abline(h = 0, lty=2)
                      panel.abline(v = 0, lty=2)
                      larrows(y0=-.5, y1= -.5, x0=beta1s[direct_breaks
    [1]], x1=beta1s[direct_breaks[2]], angle=90,code=3)
                      larrows(x0=-.6, x1= -.6, y0=beta2s[indirect_breaks
    [1]], y1=beta2s[indirect_breaks[2]], angle=90,code=3)
                    },
                    legend =
                      list(right =
                             list(fun = draw.colorkey,
                                  args = list(key = list(col = col.l, at
    = depth.breaks),
                                              draw = FALSE))),
)


pdf("CoppockJEPS_figure2_replication.pdf")
print(fig2)
```

```
217 dev.off()
218
219 rownames(pvals) <- beta1s
220 colnames(pvals) <- beta2s
221 write.csv(pvals, file="pvalues_rep.csv")
```

References