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

Sayali Phadke Bruce Desmarais

March 29, 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 cosponsorship. 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 For discussion on 03/23

- Table in the research design section
- Section 6.1: notes from lab meetings
- Issue with using Bowers framework for Coppock replication
- Guidance regarding literature review
- JSM and Political science conference

2 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.

2.1 tasks

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

3 Background

- 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)
  - (**??????**)

Table 1: The Results

	Distance from treated < 5		**Distance from treated > 5**	
	Linear	Non-linear	Linear	Non-linear
Number of treated neighbors	1	2	5	6
Proportion of treated neighbors	3	4	7	8

4 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.

4.1 tasks

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

1. Distance from the nearest treated node (d_i)
2. Number/proportion of treated nodes neighboring each d_i
3. Form of spread (linear or non-linear)

Tabulates the basic idea of the models we will look at. The linear/non-linear refers to the form taken by the spread of effect. Distance from the nearest treated node is vaguely separated as <5 or >5 for now. Finally, we want to look at the difference in spillovers effects caused due to number of treated neighbors as against proportion of treated neighbors. Potentially, the number/proportion at hops of different lengths can also be considered.

The types of networks we will consider in the analysis 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 untreated 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

5 Analysis

We begin this section by reviewing prior methodological work. We will look at two spillover specifications and testing frameworks given in Bowers et. al. and Coppock papers. We will extend their analysis by considering various other specifications for spillover effects. Additionally, we will look into tests other than the Kolmogorov-Smirnov (KS) test.

5.1 Review of existing methods

- **Bowers et. al. method:** This paper introduces a Fisherian inference algorithm to test for spillover of treatment effect. The model $\mathscr{H}$ is compared against the observed data. The hypothesized model of interference is specified by the researcher. The steps involved in conducting this test are as follows:

    1. 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:

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

- **Coppock method:** Coppock builds upon the New Mexico Legislator experiment conducted by Butler and Nickerson (2011). This paper also works with the idea of sharp null of no effects and uniformity trial. Using ideological similarity as

*Could we use the idea of communities to model spread of treatment across the network? *Explore how alternative assumptions regarding interference change results

5.2 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.

6 Summary

So far, we have replicated the two papers mentioned earlier; (**?**) and (**?**). The codes are available in the appendix. So far we have worked on the New Mexico legislators dataset from (**?**) and considered network based on ideological similarity. The plan ahead for this paper is:

- Consider different diffusion models by varying the distance from treated node, number/proportion of treated neighbors and form of spread of treatment
- Consider other legislator networks depending on geographical proximity and cosponsorship

- Consider additional test statistic such as the Anderson-Darling test and other tests mentioned in (**?**) (Mann-Whitney U test, Control Median test etc.)

6.1 Notes from lab meetings

- Would we like to give a motivating example for why people should care about spillover effects? Or should this paper itself become a motivation for people to understand the importance of it?
- In what format do we want to present the results? Potentially give visuals of p-values under 8 different models using each of the networks?
- Consider committee network
- Estimate a γ for weighted combination of networks under consideration
- What would we like to include in the paper about the possible attenuation bias? (Reference: email dated 11/11/15 with subject line 'attenuation')

7 Appendix

7.1 Appendix 1A

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:

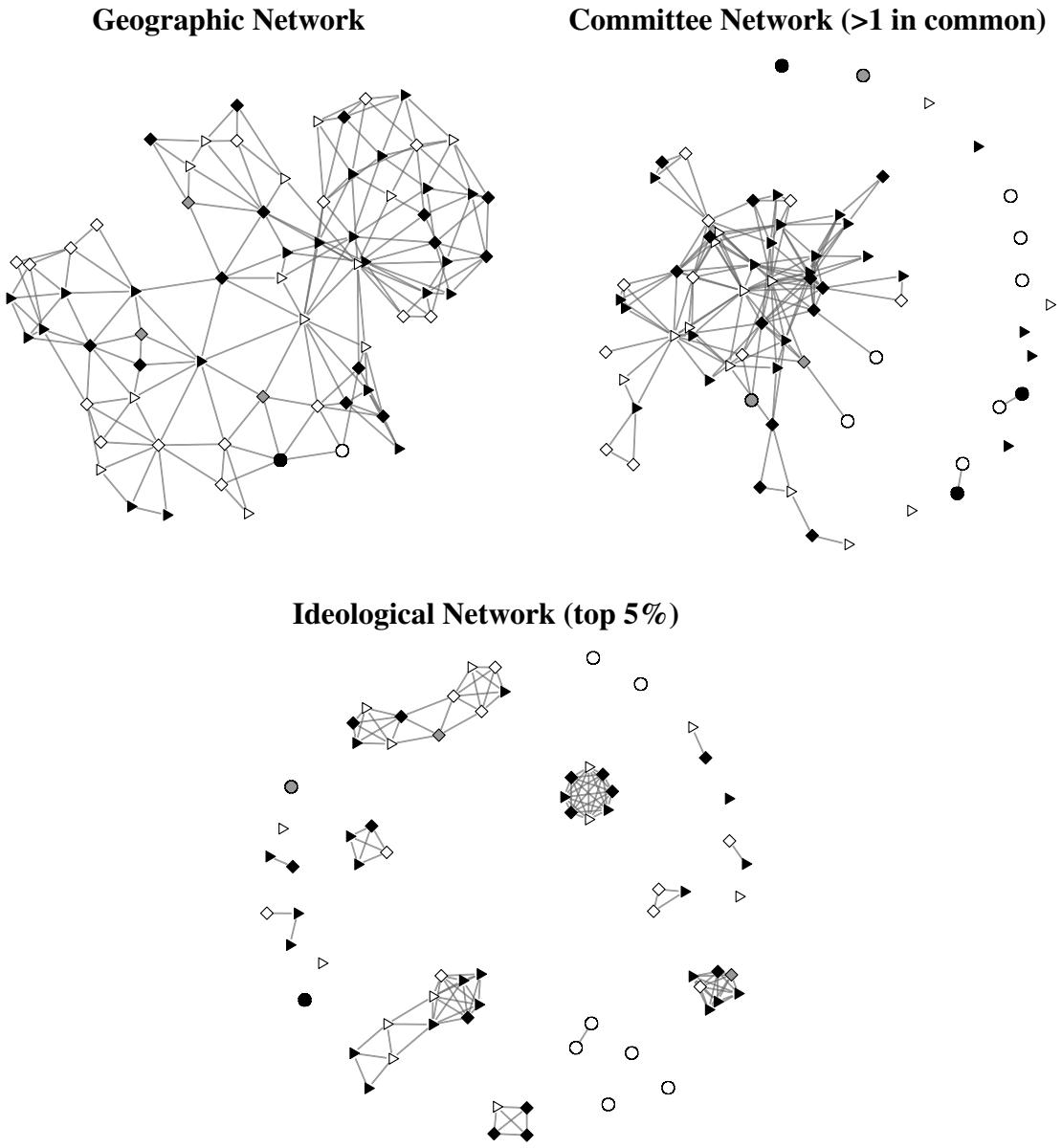

Figure 1: Different networks among New Mexico legislators. Colors denote outcome: black means voted with district, gray means abstained, white means voted against. Shape denotes treatment status. Triangles are treated. Squares are adjacent to treated. Circles are isolated from treatment

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

```r

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

  spillover <- beta + ((1-z) * (1-beta) * exp(-tau^2 * scalar))

  # This is equation 4
  h.y0.z <- spillover*y.0
}

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

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

  # Equation (3)
  spillover <- beta + ((1-z) * (1-beta) * exp(-tau^2 * scalar))

  # This is equation 5
  h.yz.0 <- (1/spillover)*y.z
}

```

```
#### Transform observed outcome into outcome for ANY other assignment w
z.to.w <- function(z, S, w, y.z, beta, tau){
  # 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 ####
```

```r

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

  cl <- makeCluster(4) #Setup for parallel computing
  registerDoParallel(cl)

  # Calculate the outcome vector after taking away the effect of
    treatment
  y.0 <- z.to.unif(z=z, S=S, y.z=y.z, beta=beta, tau=tau)

  # Calculate test statistic
  test.stat <- ks.test(y.0[z==1], y.0[z==0],
                       alternative = "g")$statistic
  sign <- noquote(strsplit(names(test.stat), NULL)[[1]])[3]
  if(sign=="+"){
    test.stat <- test.stat
  }else{
    test.stat <- test.stat*-1
  }

  # Calculate a vector of test statistic using permutations
  results <- foreach (i = 1:perms) %dopar%{
    require(permute)
    perm.z <- z[sample(1:length(z),length(z),rep=F)]
    perm.test.stat <- ks.test(y.0[perm.z==1], y.0[perm.z==0],
                              alternative = "g")$statistic
    sign <- noquote(strsplit(names(perm.test.stat), NULL)[[1]])[3]

```

```
    if(sign=="+"){
      return(perm.test.stat)
    }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)
}
```

7.2 Appendix 1B

Below code replicates the (**?**) results using the framework setup in the Bowers replication code (Version before final corrections by BD)

```

setwd("~/Dropbox/professional/Research/Active/causalityinnetworks-
    agenda/Interference_in_Field_Experiments/Analysis/coppock_
    replication_data/") # BD

rm(list=ls())
set.seed(312)
```

```
6
7  library(doParallel)
8  library(fields)
9  library(foreach)
10 library(kSamples)
11 library(network)
12 library(permute)
13 library(wnominate)
14
15
16 #### Read the original Butler and Nicketson data
17 #### This is the New Mexico dataset
18
19 data <- read.table("nm.replication.tab", sep="\t", header=TRUE)
20
21 z <- data$treatment #observed treatment
22 y.z <- data$sb24 #observed outcome
23 n <- length(y.z) #number of observations
24 t <- length(z[z==1]) #number of treated units
25 perms <- 10000 #number of permutations to use in generating expected
       exposure
26 perms.test <- 1000 #number of permutations used in testing
27
28
29 #### Generate Similarity Scores (this code taken from CoppockJEPS_
       datapreparation.R)
30
31 nmhouse2008 <-read.csv("CoppockJEPS_rollcalldata.csv")
```

```
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)])
```

```
60
61 expected.exp0 <- rep(0, n)
62 expected.exp1 <- rep(0, n)
63
64 for(p in 1:ncol(perm)){
65   zp <- perm[,p]
66   for(i in 1:n){
67     if (zp[i] == 1){
68         expected.exp1[i] <- expected.exp1[i] + sum(S.ideo[i,-i]*zp[-i])
69       }
70       else{
71         expected.exp0[i] <- expected.exp0[i] + sum(S.ideo[i,]*zp)
72       }
73   }
74 }
75 num_treat <- apply(perm,1,sum)
76 num_control <- apply(1-perm,1,sum)
77 expected.exp1 <- expected.exp1/num_treat
78 expected.exp0 <- expected.exp0/num_control
79
80
81 #### Generate expected and net exposure
82 #### This is the spillover effect model
83
84 indirect.treatment <- function(permutation, adj.mat){ #any treatment
     assignment vector and adjacency matrix can be used
85   # 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 - (permutation*expected.exp1 + (1-permutation)*
   expected.exp0)
  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(outcome, beta1, beta2, permutation, adj.mat){
  # outcome: vector of direct treatment outcomes
  # beta1: direct treatment effect parameter
  # beta2: indirect treatment effect parameter
  # permutation: vector of a permutation of z (can be z itself)
  # adj.mat: adjacency matrix

  exposure <- indirect.treatment(permutation, adj.mat)
  # This is equation 5
  h.yz.0 <- outcome - (beta1*permutation) - (beta2*exposure)
  return(h.yz.0)
}

```

```
112
113 #### Testing and p-value calculation
114
115 beta1s <- seq(from=-.7, to=0.2, by=.025)
116 beta2s <- seq(from=-.7, to=0.2, by=.025)
117
118 pvals <- matrix(NA, length(beta1s), length(beta2s))
119
120 cl <- makeCluster(4) #Setup for parallel computing
121 registerDoParallel(cl)
122
123 pvalues.ideology <- foreach (i = 1:length(beta1s)) %do% {
124   abc <- foreach (j = 1:length(beta2s)) %do% {
125
126     # Calculate the outcome vector after taking away the effect of
      treatment
127     y.0 <- z.to.unif(outcome = y.z, beta1 = beta1s[i], beta2 = beta2s[j
      ], permutation = z, adj.mat = S.ideo)
128
129     # Calculate observed test statistic
130     exposure <- indirect.treatment(permutation = z, adj.mat = S.ideo)
131     test.stat <- sum((lm(y.0 ~ z + exposure, na.action = na.omit)$resid
      )^2)
132
133     # Calculate a vector of test statistic using permutations
134
135     results <- foreach (k = 1:perms.test) %dopar% {
136       require(permute)
```

```
137       perm.z <- z[sample(1:length(z),length(z),rep=F)] #Each time we
      sample a permutation of z
138       perm.y.0 <- z.to.unif(outcome = y.z, beta1 = beta1s[i], beta2 =
      beta2s[j], permutation = perm.z, adj.mat = S.ideo)
139       perm.exposure <- indirect.treatment(permutation = perm.z, adj.mat
       = S.ideo)
140
141       perm.test.stat <- sum((lm(perm.y.0 ~ perm.z + perm.exposure, na.
      action = na.omit)$resid)^2)
142       }
143
144
145     # A vector of test statistics
146     all.test.stat.vals <- as.numeric(unlist(results))
147
148     # Calculating p-value
149     pval <- sum(all.test.stat.vals < test.stat)/perms.test
150   }
151   as.numeric(unlist(abc))
152 }
153
154 stopCluster(cl)
155
156 for (i in 1:length(beta1s)){
157   pvals[i,] <- unlist(pvalues.ideology[i])
158 }
159
160 pvals
```

```
161
162
163 ## Creating a plot
164 image.plot(beta1s, beta2s, pvals,
165            main = "Plot of p-values", xlab = "Direct effects", ylab = "
    Indirect effects")
166
167 pdf("pvalues_figure.pdf")
168 image.plot(beta1s, beta2s, pvals,
169            main = "Plot of p-values",
170            xlab = "Direct effects", ylab = "Indirect effects")
171 dev.off()
```

References