REDES Revista Hispana para el Análisis de Redes Sociales

Vol.33,#2, (2022), 203-209

http://revistes.uab.es/redes http://doi.org/10.5565/rev/redes.957

Modelos estocásticos orientados en el actor utilizando RSiena (IV). Formato de las variables.

Alvaro Uzaheta[1] y Alejandro Espinosa-Rada[1]

[1]ETH Zürich, Social Networks Lab

RESUMEN

Este texto es el cuarto guión de una serie de seis guiones escritos en el lenguaje y ambiente de programación R que conjuntamente constituyen la introducción al software RSiena para estimar modelos estocásticos orientados en el actor para redes dinámicas. Este cuarto guión amplía la información sobre la definición de datos como variables para RSiena y especificación de modelos introducida en el primer guión. El texto está acompañado por el código en R para reproducir y estimar los modelos.

Palabras clave: *Análisis de redes Sociales - Modelos estocásticos orientados en el actor – Análisis estadísticos de redes sociales – RSiena - R.*

ABSTRACT

This article is the fourth document of six scripts written in the language and environment R, which introduces the RSiena software to estimate the stochastic actor-oriented models for dynamic networks. This fourth script extends the information about how to specify data as variables in RSiena and to specify the model introduced in the first document. The text incorporates the R code into the document to reproduce and estimate the models.

Key words: *Social Network Analysis - Stochastic actor-oriented model – Statistical analysis for social networks – RSiena – R.*

[1] *Contacto con los autores: Alejandro Espinosa-Rada (alejandro.espinosa@gess.ethz.ch)[1]*

[1] El siguiente guión se encuentra basada en el guión oficial de la página web de SIENA (https://www.stats.ox.ac.uk/~snijders/siena/Rscript02SienaVariableFormat.R).

ISSN 1579-0185

INTRODUCCIÓN

Los materiales para esta serie incluyen 6 guiones (*scripts*) de ***RSiena*** escritos en el entorno y lenguage de programación **R** (2022) enfocado a la estadística computacional y gráficos. Los guiones se encuentran disponibles en la página oficial del programa para el análisis estadístico de datos de redes SIENA (acrónimo de *Simulation Investigation for Empirical Network Analysis*), con énfasis en redes sociales, implementado en el paquete *RSiena* y que cuenta con un manual de usuarios (Ripley et al., 2022). Para una revisión sobre aspectos teóricos y los usos contemporáneos de los modelos estocásticos orientados en el actor (SAOMs) para redes dinámicas en la comunidad hispana e internacional ver Espinosa-Rada (2022a).

Este guión es la cuarta contribución de una serie de seis guiones que introducen el software **RSiena**. Este guión brinda información adicional sobre la definición de los datos como variables en **RSiena** y de como especificar modelos para su posterior estimación que fue introducida en el primer guión de esta serie (Espinosa-Rada, 2022b). El primer guión de esta serie entrega un panorama general de cómo se estima un modelo SAOM (Espinosa-Rada, 2022b). El segundo guión, en cambio, introduce el ambiente **R** y los distintos tipos de estructuras de datos que pueden ser utilizados para importar o exportar los datos para los análisis (Espinosa-Rada, 2022c). El tercer guión, en cambio, presenta un ejemplo de análisis descriptivo antes de estimar un modelo SAOM (Espinosa-Rada y Uzaheta, 2022d).

A continuación, se preseta en el texto los códigos a ejecutar en el programa **R** antecedidos por una breve explicación. En color "(_)" se encuentran las líneas de códigos o comandos a ejecutar y/o que pertenecen a funciones propias del programa. De la misma forma, líneas de códigos antecedidos por # corresponden a comentarios y no comandos. Finalmente, líneas antecedidas por > corresponden a la salida que se imprime en la consola del código presentado en el mismo bloque.

Definiendo el conjunto de datos

La descripción y disponibilidad de los datos que se utilizarán en este tutorial, utilizados también en el primer guión introductorio de **RSiena** (Espinosa-Rada, 2022b), se encuentran disponibles en http://www.stats.ox.ac.uk/~snijders/siena/s50_data.htm

```
library(RSiena)

friend.data.w1 <- s501
friend.data.w2 <- s502
friend.data.w3 <- s503
drink <- s50a
smoke <- s50s
```

Variables dependientes en RSiena

El proceso de estimación de **RSiena** espera un objeto de tipo "`sienaDependent`" que defina la variable dependiente en el modelo. La función homónima sienaDependent se usa para definir este tipo de objetos el cual tiene dos escenarios típicos de uso, para definir una red como variable dependiente y para definir una variable de comportamiento como variable dependiente.

La definición de una red como variable dependiente se puede hacer a partir de objetos de tipo matriz (?matrix), arreglo multidimensional (?array) o una lista de matrices dispersas (?list ?Matrix del paquete homónimo). En este tutorial definiremos la red dependiente a partir de un arreglo multidimensional. En particular, la documentación de la función sienaDependent contiene ejemplos de cómo definir la red dependiente usando matrices dispersas. Para crear el arreglo se usa la función array usando como entrada la colección de redes observadas en los diferentes momentos de tiempo. Para nuestros datos el arreglo multidimensional es de dimensiones $50 * 50 * 3$. Esto quiere decir, tres matrices de $50\,x\,50$ que corresponden a las tres rondas de recolección de información de las redes de amistad disponibles. En general los números enteros usados para la definición de la dimensión hacen referencia a la cantidad de actores que pueden crear vínculos, cantidad de actores que pueden recibir vínculos y la cantidad de momentos de tiempo en los que se tiene información.

```
friendshipData <- array(c( friend.data.w1,
        friend.data.w2, friend.data.w3),
        dim = c(50, 50, 3))
```

Ahora que tenemos un objeto del tipo "*array*" podemos asignar a continuación el rol de variable dependiente para el análisis:

```
friendship <- sienaDependent(friendshipData)
```

¿Qué fue lo que construímos?

```
friendship
> Type          oneMode
> Observations 3
> Nodeset       Actors (50 elements)
```

Ahora podemos ver en la consola una breve descripción del objeto que fue creado y como **RSiena** lo interpreta. Es importante verificar que el tipo y las cantidades reportadas coinciden con los datos que se utilizan como entrada. En este caso por ser redes donde los mismos actores que crean vínculos pueden recibirlos el tipo de red identificada por RSiena es `oneMode`, una red de un modo. La cantidad de observaciones (`Observations`) corresponde a la cantidad de rondas analizadas. Para este caso, tres mediciones en el tiempo para estudiar redes de amistades. El conjunto de nodos que define a los actores (`Nodeset`) es reportado con su respectivo tamaño, en este caso no se definió un conjunto de entrada, por lo cual la función retorna el nombre genérico `Actors` que contendría 50 elementos.

En el caso que la red sea de dos modos es importante definir los dos objetos que contendrían la información de los conjuntos de quienes envían vínculos y quienes los reciben, en ese caso es conveniente revisar la documentación de la función sienaNodeSet. Un ejemplo de redes de dos modos es el reporte del tipo de actividades practicadas en el tiempo libre, o la pertenencia a un club cultural, correspondientes a focos de actividades de los actores (Feld, 1981). En ambos casos un modo corresponde a los actores que practican o pertenecen a clubes y el segundo modo corresponde a las actividades o clubes disponibles.

En el segundo escenario de uso de la función sienaDependent es para definir una variable de comportamiento como variable dependiente. En este caso los datos de entrada corresponden a una matriz con actores en filas y en columnas los valores reportados en cada momento de tiempo. La matriz drink reporta la frecuencia con la que los adolescentes consumen alcohol, sus dimensiones corresponden a 50 filas y 3 columnas.

```
drinking <- sienaDependent(drink,
                type = "behavior")
```

Ahora hemos creado un objeto que contiene la información del consumo de alcohol como variable dependiente. Es importante notar el uso del parámetro `type` para informar **RSiena** que en este caso la variable dependiente es de tipo comportamiento (`behavior`). El valor por defecto del parámetro `type` es "`oneMode`" que corresponde al caso más común de redes como la del ejemplo que estamos siguiendo. Otras valores que puede tomar este argumento son: "`bipartite`" para redes de dos modos y "`continuous`" para variables de comportamiento que contienen valores en escala continúa (por ejemplo, puntaje en una prueba de conocimiento).

Es importante hacer notar algo sobre la definición de variables dependientes en **RSiena**, y ello es que solo se debe usar una variable en un rol para un modelo. Por ejemplo, solo se puede ser variable dependiente de comportamiento o una covariante variable para un modelo, pero los mismos datos no se deben usar para definir la misma información usada como dependiente y covariante al mismo tiempo. El hecho que se use como variable dependiente de comportamiento no siginifica que esta información no se pueda usar como covariante para explicar la evolución de la red.

Variables covariantes en RSiena

A continuación, se asignarán las variables explicativas del modelo. Para ello se utilizará un conjunto de funciones de RSiena que se usarán dependiendo el tipo de información contenida por la variable explicativa.

La documentación de las diferentes funciones disponibles para definir variables explicativas se puede encontrar usando el comando:

```
?sienaDataCreate
```

Tipo de variable	Monádica	Diádica
Constante	coCovar	coDyadCovar
Variable	varCovar	varDyadCovar

Tabla 1: Funciones en RSiena para definir variables explicativas

La Tabla 1 presenta las principales funciones dependiendo el tipo de variable explicativa que se va a definir. Las columnas de la tabla distinguen variables definidas a nivel del actor (monádicas) como por ejemplo la edad o la frecuencia de consumo de alcohol; y las variables definidas a nivel de una pareja de actores (díadicas) como por ejemplo atender al mismo colegio en la primaria o si la pareja de actores tuvo una relación romántica en el pasado. Las filas de la tabla distinguen si la información de la variable explicativa es constante para el período de recolección de la información, como es el caso del sexo biológico de los actores, o si la información de la variable varía en el periodo de recolección de los datos,

como en el caso de la frecuencia de consumo de alcohol o tabaco.

Para fines ilustrativos en este tutorial consideremos que la variable fumar para la ola 1 es *una covariantes del actor que es constante.* Este tipo de variables usualmente se definen en R con una matriz con los actores en filas y una columna, para definirla en **RSiena** usamos el siguiente código:

```
smoke1 <- coCovar(smoke[, 1])
```

En el caso que la matriz que contiene los datos en R tiene la información de más de una covariante del actor que es constante en el tiempo, se pueden definir las covariantes usando varías invocaciones de la función de la siguiente manera:

```
attr1 <- coCovar(covariates[, 1])
…
attrp <- coCovar(covariates[, p])
```

La variable consumo de alcohol fue medida en las tres rondas de encuestas por lo tanto es una *covariante del actor que varía en el tiempo*. Este tipo de variables se definen en R en una matriz en donde las filas son actores y las columnas las diferentes mediciones, para definirla en **RSiena** usamos el siguiente código:

```
alcohol <- varCovar( drink )
```

De manera similar las variables díadicas se pueden definir en **RSiena**. Una *covariante díadica que es constante* es definida en R en una matriz. Una *covariante díadica que varía en el tiempo* es definida en un arreglo multidimensional siguiendo un patrón similar al usando para definir la variable dependiente de las redes de amistad. Las funciones coDyadCovar y varDyadCovar respectivamente se utilizan para definir este tipo de variables explicativas.

Combinando variables dependientes y explicativas en RSiena

A continuación, se combinan las variables dependientes y explicativas creadas en los pasos anteriores usando él siguiente código

```
mydata <- sienaDataCreate( friendship,
          smoke1, alcohol)
```

Como podrá apreciar, el nuevo objeto (mydata) es de clase siena.

```
mydata
>Dependent variables:  friendship
>Number of observations: 3
>
>Nodeset                  Actors
>Number of nodes              50
>
>Dependent variable friendship
>Type               oneMode
>Observations       3
>Nodeset            Actors
>Densities          0.046 0.047 0.05
>
>Constant covariates:  smoke1
>Changing covariates:  alcohol
```

El nuevo objeto contiene una breve descripción de los datos almacenados en él. Incluye datos sobre la densidad de las redes en cada ola, número de actores y el tipo de rol que tienen las variables que es predeterminado por las funciones usadas en su creación, tal como fue especificado en la sección anterior.

A este punto debe ser claro cuál es la diferencia entre los roles de las variables entre la anterior especificación y las siguientes

```
mydataAlt <- sienaDataCreate( friendship,
          smoke1, drinking)
```

Definiendo los efectos del modelo

Una vez se define la variable dependiente y las covariantes a usar en el modelo, es necesario definir el conjunto de efectos que se incluirán en el modelo. Los efectos son la pieza central del modelo ya que ellos son la herramienta utilizada por los investigadores para traducir hipótesis conceptuales a fórmulas matemáticas que se integran en el modelo. Cada uno de los efectos introducidos en el modelo tendrán asignado un parámetro que representa el peso de ese efecto en cada uno de los componentes del modelo.

A continuación, y de tal forma de especificar el modelo se crea primero un objeto "myeff" en donde es posible observar todos los efectos disponibles y actualmente implementados en **RSiena** que se ajustan a las características de las variables dependientes y explicativas en un objeto creado a través de la función sienaDataCreate.

```
myeff <- getEffects( mydata )
myeff
```

Como podrá observar, por defecto el programa incorporará el efecto de densidad (grados de salida, outdegree) y reciprocidad (reciprocity) dada su ubicuidad en redes sociales, y también incluye la velocidad de los cambios (rate) que

representa la cantidad esperada de oportunidades que tiene un actor para realizar cambios en sus vínculos. Un análisis a mayor profundidad de los factores que juegan un rol en el cambio de los vínculos de amistad requiere añadir efectos a esta especificación inicial. A continuación, veremos una lista detallada de todos los efectos disponibles para el conjunto de datos (se abrirá una ventana en html):

```
effectsDocumentation(myeff)
```

El archivo html muestra la colección de todos los efectos disponibles para la combinación de variable dependiente y covariantes presente en el conjunto de datos. En el manual de *RSiena* (Ripley et al., 2022, capítulo 12) es posible explorar con mayor detención el significado de cada uno de los efectos y su formulación matemática. El cuadro de efectos contenido en el archivo html presenta campos útiles para la identificación y uso de los efectos disponibles en RSiena. El campo `name` contine el nombre de la variable dependiente sobre la cual el efecto puede ser usado, en nuestro case friendship es la variable dependiente del modelo. Los campos `effectName` y `shortName` contiene la descripción y el nombre corto del efecto respectivamente. Este último es de gran utilidad para añadir efectos en el objeto myeff. El campo `type` describe a qué parte del modelo el efecto es aplicado. Los restantes campos describen argumentos adicionales del efecto, estos argumentos adicionales describen como usar las covariantes definidas de antemano en el conjunto de datos para definir efectos.

Existen tres funciones en RSiena para incluir o modificar efectos del cuadro html (o del manual) en el objeto myeff que usaremos para definir y estimar el modelo. La primera función nos permite incluir nuevos efectos en el modelo:

```
?includeEffects
```

Esta función usa el `shortName` incluido en el html o en la documentación de RSiena. Como ilustración, incluiremos algunos efectos:

```
# transitividad de tripletas (transTrip) y ciclo de 3
(cycle3)
myeff <- includeEffects(myeff, transTrip, cycle3)

# Efectos covariantes: covariante de ego (egoX),
covariante de alter (altX) y similaridad de ego y
alter
myeff <- includeEffects(myeff, egoX, altX,
          simX, interaction1 = "alcohol")
myeff <- includeEffects(myeff, simX,
          interaction1 = "smoke1")
```

La primera llamada de la función incluye efectos endógenos, es decir, efectos que se definen sobre la variable dependiente correspondiente a la red. Esta llamada incluye dos efectos al modelo (i.e., transTrip y cycle3). El segundo llamado de la función incluye tres efectos definidos usando la misma covariante, en este caso el consumo de alcohol (interaction1 = "alcohol"). De una manera similar, el último llamado incluye un nuevo efecto usando esta vez la información del consumo de cigarrillos (interaction1 = "smoke1"). Es importante notar que las nuevas llamadas de la función crean un nuevo objeto, esta es la razón por la cual el objeto myeff es reemplazado por la salida de las invocaciones de la función includeEffects.

Es posible remover efectos usando el argumento include de la función. Por ejemplo, para remover el efecto de ciclos de 3 del modelo:

```
myEff <- includeEffects(myEff, cycle3,
          include = FALSE)
```

La segunda función que nos permite incluir efectos en el modelo es:

```
? includeInteraction
```

Esta función permite incluir interacciones entre efectos al modelo. Las interacciones permiten estudiar cómo es que los pesos de un efecto están moderados por los valores que toma otro efecto o covariante. Es posible definir interacciones de hasta tres efectos si el tipo de la interaccion es permitido (campo `interactionType` en el html, más detalles en el manual de RSiena). La interacción del nivel de consumo de cigarrilo del ego y la reciprocidad se puede definir como

```
myeff <- includeInteraction( myeff, egoX, recip,
          interaction1 = c("smoke1", ""))
```

Es importante observar que el argumento interaction1 ahora usa un vector de carácteres de longitud dos. Este argumento cumple el mismo rol que en la función includeEffect, identifica la covariable que será utilizada. Aquí el efecto egoX opera sobre la covariante monadíca smoke1. En el caso que uno de los efectos no requiera la definición de una covariante se usa el carácter vació ("") como es el caso del efecto recip que es endógeno y no requiere la definición de una covariante. La longitud de este vector debe coincidir con la cantidad de efectos usados para la definición de la interacción.

La tercera función que nos permite modificar efectos en el modelo es:

```
? setEffect
```

Esta función permite modificar caracteristicas de los efectos definidos en el modelo. Por ejemplo, el efecto de las tripletas densas con al menos *xx* vínculos, donde *xx* es el parámetro del efecto, que puede tomar valores de 5 o 6 (cabe anotar que 6 es el máximo numero de vínculos en una tripleta). El valor por defecto de este parámetro en este efecto es 5. Podemos cambiarlo a 6 con el siguiente comando:

```
myeff <- setEffect(myeff, denseTriads,
                   parameter = 6)
```

El argumento parameter en este caso es descrito en el manual de **RSiena** (Ripley et al., 2022) sección 12.

Comentarios finales

En el siguiente guión se revisó cómo especificar los datos para ser analizado a través del software **RSiena**. Brindando especial atención a las particularidades de definir variables explicativas dependiendo del tipo de información y la temporalidad que contienen. Adicionalmente, se revisó la definición de las variables dependientes en **RSiena** dependiendo su tipo. Las definiciones de variables dependientes y explicativas convergen en la definición de un objeto de clase siena que es la base para definir el conjunto de posibles efectos que se pueden usar para la especificación de un modelo. A continuación, se abordaron las distintas formas de incluir y modificar efectos en la especificación de un modelo.

Los siguientes guiones de esta serie se concentrarán principalmente en la estimación del modelo. Esperamos que esta introducción permita dar una aproximación a los modelos estocásticos basados en el actor a través del uso práctico del programa, acorde al estado actual de su desarrollo y que sea accesible a la comunidad de habla hispana.

Agradecimientos

Agradecemos particularmente a Tom A.B. Snijders (University of Oxford y University of Groningen) por su amabilidad en permitirnos difundir, modificar y extender los siguientes guiones los cuales contaron también con contribuciones previas de Robin Gauthier, Ruth Ripley, Johan Koskinen, Paulina Preciado, Zsofia Boda y Christian Steglich. En un principio estos documentos iban ser una traducción textual a los guiones disponibles en la página de RSiena. No obstante, por sugerencia de Tom A. B. Snijders los guiones traducidos y modificados adquieren con esta contribución independencia de sus homólogos en inglés con el objeto de facilitar su difusión, mantensión y responsabilidad en su contenido en español del cual esperamos reflejar adecuadamente sus intensiones originarias.

FINANCIAMIENTO

Alejandro Espinosa-Rada contó con el apoyo del proyecto Fondecyt Regular N° 1220560 para realizar esta investigación, proyecto financiado por la Agencia Nacional de Investigación y Desarrollo (ANID, Chile).

REFERENCIAS

Espinosa-Rada, A. (2022a). "Usos contemporáneos de los modelos estocásticos orientados en el actor (SAOMs) para redes dinámicas". En: Francisca Ortiz y Alejandro Espinosa-Rada (Eds.). *El Análisis de Redes Sociales en América Latina*. *En prensa.*

Espinosa-Rada, A. (2022b). Modelos estocásticos orientados en el actor utilizando RSiena (I). Guión básico introductorio. *Revista Hispana para el Análisis de Redes Sociales*, 33(1), 100-111. DOI: https://doi.org/10.5565/rev/redes.936

Espinosa-Rada, A. (2022c). Modelos estocásticos orientados en el actor utilizando RSiena (II). Formato de los Datos. *Revista Hispana para el Análisis de Redes Sociales*, 33(1), 92-99. DOI: https://doi.org/10.5565/rev/redes.937

Espinosa-Rada, A. & Uzaheta, A. (2022d). Modelos estocásticos orientados en el actor utilizando RSiena (III). Análisis Descriptivo. *Revista Hispana para el Análisis de Redes Sociales*

Feld, S. L. (1981). The focused organization of social ties. *American Journal of Sociology*, 86(5), 1015-1035. DOI: https://doi.org/10.1086/227352

Ripley, R., Snijders, T. A.B., Boda, Z. Vörös, A. & Preciado, P. (2022). *Manual for SIENA version 4.0*. Oxford: University of Oxford, Department of Statistics, http://www.stats.ox.ac.uk/~snijders/siena/.

Snijders, T.A.B. (2001) The statistical evaluation of social network dynamics, *Sociological Methodology - 2001*, 40: 361-395.

Snijders, T.A.B. (2017). Stochastic Actor-Oriented Models for Network Dynamics. *Annual Review of Statistics and Its Application*, 4, 343-363.
DOI: http://dx.doi.org/10.1146/annurev-statistics-060116-054035
Here is the e-print access to this article.

R Core Team (2022). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. URL: https://www.R-project.org/

Remitido: 28-03-2022

Corregido: 30-03-2022

Aceptado: 04-04-2022