REDES Revista Hispana para el Análisis de Redes Sociales

Vol.33,#1, (2022), 92-99

http://revistes.uab.es/redes https://ddoi.org/10.5565/rev/redes.936

Modelos estocásticos orientados en el actor utilizando RSiena (I). Guión básico introductorio.

Alejandro Espinosa Rada[1]
ETH Zürich / The Mitchell Centre for Social Network Analysis, The University of Manchester

RESUMEN

Este texto es el primero 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. El texto está acompañado por los guiones para reproducir y estimar los modelos.

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

ABSTRACT

This article is the first 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 first script is an example of commands for estimating a network and behavioural dependent variables incorporating the main functions, instructions, and exercises. The text incorporates the scripts into the document to reproduce and estimate the models.

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

[1]*Contacto con los autores: Alejandro Espinosa Rada <anespinosa@uc.cl>*

INTRODUCCIÓN

Los materiales para esta serie incluyen 6 guiones (*scripts*) de ***RSiena*** escritos en el entorno y lenguage de programación **R** (2021) enfocado a la estadística computacional y de gráficos. Los guiones se encuentran disponibles en la página oficial[1] 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., 2021).

La principal aproximación utilizada en *SIENA* es centrada en el actor para el modelamiento de redes dinámicas (o de redes y comportamiento) (Snijders, 2001, 2017; Snijders, van de Bunt y Steglich, 2010) en donde se asume que los actores sociales, representados por nodos en la red, poseen un rol crucial en la decisión de realizar cambios sobre sus relaciones con otros actores (ejemplo: señalar a alguien como amigo(a) o colaborador). De la misma forma, cuando además se consideran los comportamientos o atributos de los actores asociados a las redes dinámicas, también es posible considerar cambios en los comportamientos (ejemplo: propensión de los actores a consumir drogas o alcoholes). Todos estos modelos consideran cadenas de Markov como parte de la estimación de sus parámetros, modelos que suelen ser apropiados para estudiar relaciones y variables de comportamiento las cuales son considerados como *estados* en comparación a *eventos* no duraderos. De la misma forma, los análisis estadísticos en SIENA se realizan basados en simulaciones computacionales de redes los cuales, debido a su demanda computacional, suele ser aplicados a redes de aproximadamente 10 a 1.000 nodos. Para múltiples replicaciones de redes pequeñas (de un mínimo de 4 a 5 nodos), hay opciones alternativas que utilizan modelos de multinivel (también conocidos como modelos jerárquicos o mixtos) a través de la función *sienaBayes()* (aún experimental).

Actualmente **RSiena** se encuentra alojada en GitHub[2] y dispone de un grupo de usuarios para intercambiar información y apoyo técnico (consultas y apoyo técnico únicamente en inglés)[3]. **RSiena** se encuentra disponible en el repositorio de **R** CRAN y en 2017 recibió el premio de la *International Network for Social Network Analysis (INSNA) William D. Richards Award* debido a que "pone a disposición pública un software de análisis de redes sociales sin el cual no sería posible estudiar redes sociales". El equipo investigador está liderado por Tom A.B. Snijders, Christian Steglich, Johan Koskinen, Nynke Nieznik, Viviana Amati, Christoph Stadtfeld y Robert Krause, con contribuciones previas de Ruth Ripley, Krists Boitmanis, Felix Schönenberger, Charlotte Greenan, Josh Lospinoso, Paulina Preciado, Michael Schweinberger, Mark Huisman y Marijtje van Duijn.

Para el caso de la comunidad luso-hispana, el número de tutoriales e investigaciones empíricas que utilizan esta perspectiva es escaso con notables excepciones (algunos ejemplos: de Federico de la Rüa, 2003, 2005; Maya-Jariego et al., 2006; Baerveldt et al., 2010; Díaz-José et al., 2013; Palacios y Berger, 2015; Maya-Jariego et al., 2018; Espinosa-Rada, En prensa). Principalmente debido a las barreras idiomáticas que limitan su uso en Iberoamérica y la rapidez en que el programa se ha ido desarrollando. La siguiente serie de guiones pretenden subsanar esta limitación.

A continuación, se presentan 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.

Directorio

Este guión tiene como requisito que el usuario tenga cierta familiaridad en el uso del lenguaje **R** de tal forma de ejecutar los códigos y disponder de un ejemplo sobre un modelo SIENA. No obstante, para aquellas o aquellos que requieran de una introducción a **R**, se sugiere buscar otros tutoriales o empezar por el siguiente guión de esta serie (Espinosa-Rada, 2022).

En lo que sigue, se presenta un ejemplo práctico del uso del software a través de una secuencia de comandos para estimar un modelo básico a través de la función *siena07()* del programa **RSiena**. Explicaciones substantivas y actualizadas de esta perspectiva en lengua castellana pueden encontrarse en

[1] Url de la página oficial de RSiena: https://www.stats.ox.ac.uk/~snijders/siena/
[2] Url de la página de Github: https://github.com/snlab-nl/rsiena/
[3] Url de grupo para apoyo técnico: https://groups.io/g/Rsiena

Espinosa-Rada (En prensa) y/o en inglés en Snijders, van de Bunt y Steglich (2010).

Para comenzar y antes de replicar el guión, se sugiere identificar en qué ruta se encuentra el directorio de trabajo en que el programa **R** guardará o buscará los archivos para el siguiente tutorial. Para ello, puede ejecutar en la línea de comando la siguiente función para identificar la carpeta en donde podrá acceder o interactuar con su computadora:

```
getwd()
```

Si prefiere que sea un directorio diferente, este puede ser modificado con un código similar. Por ejemplo:

```
# Microsoft:
setwd("C:\\Users\\nombre_usuario\\Documents\\Siena\\ ")

#Mac:
setwd("/Users/ nombre_usuario/Documents/Siena/")
```

Tome en consideración las dobles barras invertidas del código utilizadas en **R**.

Si en vez de trabajar con **R** trabaja con **RStudio,** el cual es uno de los posibles entornos de desarrollo integrado (intérprete) que facilita la usabilidad del programa, puede asignar el directorio de trabajo dirigíendose a "*Session*" para luego presionar "*Set Working Directory*" y finalmente "*Choose Directory*" donde podrá seleccionar la carpeta en donde se realizará la conexión entre el programa y su computadora.

Instalando RSiena y RSienaTest

Si *RSiena* no ha sido instalado, se puede utilizar el comando:

```
install.packages("RSiena",
repos="https://cloud.r-project.org")
```

Si el repositorio no fue especificado (repos=), puede seleccionar alguna de las opciones en la lista de repositorios disponibles que aparecerán por defecto. La instalación se realizará guardándose localmente en su computadora y no será necesario que vuelva a ser instalada con posterioridad a menos que se requiera actualizar la versión o en caso de que el paquete se haya desinstalado (por ejemplo, debido a cambio de versión de R).

La version de GitHub[4] se encuentra generalmente más actualizada que las versiones disponibles en **CRAN** (los cuales corresponden a los archivos generales de **R)**. Las versiones más actualizadas también se encuentran disponibles en la página web de SIENA, en donde es posible instalar los paquetes desde los archivos locales.

Definiendo el conjunto de datos

Es posible descargar los datos desde la página web de Siena ("Data sets" tab)[5] y descomprimir los archivos en el directorio de trabajo. Al abrir el sitio web se descargarán un conjunto de archivos que utilizaremos a continuación. Recuerde asignar adecuadamente el directorio de trabajo, dado que, en caso contrario, **R** no logrará encontrar adecuadamente los archivos contenidos en su computadora.

Una vez descargados los datos[6], los cuales deberán estar alojados en su carpeta local correctamente vinculado con su sesión (a través de la función *setwd()*), podrá proceder a leer los archivos de datos a través de los siguientes comandos:

```
friend.data.w1<-
as.matrix(read.table("s50-network1.dat"))

friend.data.w2<-
as.matrix(read.table("s50-network2.dat"))

friend.data.w3<-
as.matrix(read.table("s50-network3.dat"))

drink<-as.matrix(read.table("s50-alcohol.dat"))

smoke<-as.matrix(read.table("s50-smoke.dat"))
```

En caso de que no se encuentre conectado a internet, y suponiendo que ya ha instalado previamente el paquete **RSiena**, los datos se encontrarán alojados internamente en el programa. Para ello, solo tendrá que abrir el paquete en la consola de **R** utilizando la siguiente función:

```
library(RSiena)
```

Una vez que el programa se encuentre correctamente instalado, puede utilizar la función ?Rsiena para ver algunas de las

[4] Para instalar la versión más reciente desde GitHub: https://github.com/snlab-nl/rsiena/releases/

[5] Sitio web para descargar los archivos de este tutorial: http://www.stats.ox.ac.uk/~snijders/siena/s50_data.zip

[6] La descripción y disponibilidad de los datos que se utilizarán en este tutorial se encuentra en http://www.stats.ox.ac.uk/~snijders/siena/s50_data.htm

operaciones básicas de tal forma de identificar el flujo en que las funciones son ejecutadas (y que veremos con más detalle a continuación).

Ahora podrá utilizar las funciones y datos disponibles internamente en **RSiena.** Por ejemplo, es posible disponer del conjunto de datos *s50* en lugar de descargarlos directamente desde la página web del software. Al respecto, para observar su documentación simplemente presione en su consola `?s50` en donde, como podrá observar, los datos corresponderán a una red de amistad de 50 adolescentes medidos en 3 momentos en el tiempo.

Ahora puede observar el comienzo y el final de la matriz de la primera ola (6 primeros y 6 últimos casos respectivamente).

```
head(s501)
tail(s501)
```

Además, dado que el estudio buscó entender el consumo de drogas y alcohol por parte de los jóvenes, también se encuentra información tales como una variable de comportamiento (o atributos de los actores) correspondiente al consumo de alcohol (en donde 1="No bebe alcohol", 2="Una o dos veces al año", 3="Una vez al mes", 4="una vez a la semana", 5="más de una vez a la semana"):

```
s50a
```

A continuación, modificaremos el nombre de los objetos por conveniencia (este paso puede omitirse si los datos ya fueron leídos con anterioridad utilizando la conexión entre **R** y su carpeta local).

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

En los datos también se encuentra un conjunto de variables que corresponden al consumo de cigarrillos (en donde 1="no fuma", 2="fuma ocasionalmente", 3="Fuma regularmente, i.e. más de una vez por semana").

```
smoke <- s50s
```

Los datos deben ser especificados para cumplir un rol específico en el análisis, de tal forma que las variables sean compatibles con el análisis a realizar a través de **RSiena**.

Variable dependiente

```
?sienaDependent
```

Primero, hay que crear una matriz de $50 * 50 * 3$ compuesta por las tres matrices de adyacencia. Esto quiere decir, tres matrices de $50\ x\ 50$ (clase "array": class(friendshipData)).

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

Dado que ahora dispondremos de 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
```

Como se puede observar, la variable dependiente utilizará como información los vínculos observados entre los actores en tiempo discreto (las respectivas olas analizadas). Considérese que las relaciones aquí obedecen a vínculos que son considerados como "estables" (amistades para este caso). No obstante, si la granularidad del tiempo es más precisa (ejemplo: llamadas de teléfono, correos electrónicos, pases en un partido de fútbol), se sugiere considerar otros tipos de modelos que sean más acordes (ver Espinosa-Rada, En prensa).

A continuación, se prepararán los objetos que serán utilizados como variables explicativas.

Covariante de los actores

A continuación, se asignarán las variables explicativas del modelo. Para ello se utilizará en primer lugar la variable fumar para la ola 1 que, para fines ilustrativos en este tutorial, será considerada *como si fuese una covariante del actor que es constante* (pero que originariamente varía entre las olas):

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

Otra covariante del actor que será definida será en relación al consumo de alcohol (la selección de esta variable tiene propósitos únicamente ilustrativos):

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

Considérse que variables "constantes" son asignadas con la función coCovar y aquellas que "varían" en el tiempo son especificadas a través de la función varCovar. Otros formatos de variables pueden ser exploradas en la documentación:

```
¿sienaDataCreate
```

A continuación, se adjuntan las variabes creadas en un conjunto común de datos para el análisis, en donde la variable dependiente debe ser incorporada primero seguido por las variables explicativas o de control.

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

Como podrá apreciar, el nuevo objeto (mydata) contendrá información de interés (ejemplo: densidad en cada ola, número de actores o el tipo de variables que contiene el objeto).

A su vez, puede generar un informe con datos descriptivos de los datos (el cual será guardado en la carpeta asignada localmente en su computadora)

```
print01Report( mydata, modelname="s50")
```

De tal forma de especificar el modelo se crea primero un objeto "efecto" en donde es posible observar todos los efectos disponibles y actualmente implementados en **RSiena** que se ajustan a las características de la red explícitamente creadas a través de la función sienaDataCreate. Para este conjunto de datos (se abrirá una ventana en html):

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

En ciertas ocasiones, algunos comandos pueden generar errores o advertencias. Para ilustrar posibles errores, considérese la función effectsDocumentation la cual depende del paquete xtable. Si el paquete mencionado no se encuentra instalado en su computadora el programa realizará una advertencia indicando aquello. Por ejemplo: Error in loadNamespace(name): there is no package called ‘xtable’). En este caso, tendrá que instalar el paquete desde los repositories CRAN install.packages("stable") y, una vez instalado, podrá continuar ejecutando el resto de las funciones.

En el manual de *RSiena* (capítulo 12) es posible explorar con mayor detención el significado de cada uno de los efectos y su formulación matemática.

Para una descripción básica de algunos efectos estructurales:

```
?includeEffects
```

A continuación, asignaremos 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" )
```

Ahora podemos comprobar los efectos que han sido incorporados en el modelo.

```
myeff
```

Como podrá observar, por defecto el programa incorporará el efecto de densidad (grados de salida) y reciprocidad dada su ubicuidad en las redes sociales.

Modelo

Una vez especificada la "función objetiva" (efectos incorporados a través de la función includeEffects) podemos crear y modificar el algoritmo que realizará la estimación de los parámetros. Posteriormente, se podrán juntar los distintos ingredientes del modelo (algoritmo, datos y efectos) y estimar un primer modelo:

```
myalgorithm <- sienaAlgorithmCreate(
        projname = 's50' )
```

Esta función creará un archivo de registro (s50) que también se guardará en su computadora en la carpeta que fue asignada al principio del tutorial.

El algoritmo posee bastantes especificaciones (?sienaAlgorithmCreate), a menos que sepa lo que está haciendo, se recomienda seguir las recomendaciones sugeridas en el manual (Ripley et al., 2021) o en su defecto dejar las especificaciones por defecto.

```
# Estimar parámetros
?siena07
ans <- siena07( myalgorithm,
        data = mydata, effects = myeff)
ans
```

Esto entregará los resultados partiendo desde puntos aleatorios. Para asignar puntos fijos de partida (debido a que son simulaciones), utilice el parámetro *seed* en donde podrá asignar cualquier número el cual permitirá replicar los resultados con posterioridad. Por ejemplo, aquí se ha elejido arbitrariamente el número 21061918 (que coincide con la fecha de nacimiento invertida del antropólogo James Clyde Mitchell):

```
myalgorithm <- sienaAlgorithmCreate(
        projname = 's50',
        seed=21061918)
```

Una vez con los resultados, se pueden exportar los resultados del modelo en un archivo "html" (o "tex").

```
siena.table(ans, type="html", sig=TRUE, d=3,
vertLine=TRUE)
```

En algunos casos el modelo puede ser computacionalmente demandante y requerir de bastante tiempo antes de arrojar el resultado (ejemplo: esto se puede deber a que la matriz es muy grande, el modelo es muy complejo o se encuentra mal especificado), en dicho caso, puede ser aconsejable guardar el objeto en su computadora para futuras exploraciones y análisis.

```
save(ans, file="ans.Rda")
```

No obstante, para evaluar la convergencia es aconsejable mirar el "*overall maximum convergence ratio*" situado debajo de la estimación de los parámetros. El cual también puede ser analizado de forma separada:

```
ans$tconv.max
```

Si el total de la tasa de convergencia máxima es menor a 0.25, por convención se asume que la convergencia es adecuada. En caso contrario, podemos continuar la estimación utilizando el parámetro "previous answer" de la función siena07 en donde el modelo continuará con la estimación:

```
ans <- siena07( myalgorithm, data = mydata,
        effects = myeff, prevAns=ans)
ans
```

Si la convergencia es adecuada, puede observar la estimación de los resultados con más detalles:

```
summary(ans)
```

Un archivo que contiene aún más información se encuentra en el archivo *s50.txt* en la carpeta en donde fue asignado el directorio de trabajo.

Considere que cuando un comando de R está entre paréntesis *(….)*, los resultados serán impresos en la consola. A continuación, se incorporará la transitividad de tripletas recíprocas, el cual corresponde a una interacción entre transitividad de tripletas y reciprocidad para tener un modelo alternativo (ans1) al estimado hasta ahora (ans).

```
(myeff <- includeEffects( myeff, transRecTrip))
(ans1 <- siena07( myalgorithm,
        data = mydata,
        effects = myeff, prevAns=ans))

# Si es necesario, repetir la estimación con los nuevos resultados:
(ans1 <- siena07( myalgorithm,
        data = mydata,
        effects = myeff, prevAns=ans1))
# Si la tasa de convergencia máxima no es menor a 0.25, seguir estimando.
```

Una vez alcanzada una estimación adecuada, podrá ir al archivo *s50.txt* en el directorio de trabajo e interpretar el significado de su contenido.

Para realizar un análisis conjunto de los tres efectos de la variable alcohol:

```
?Multipar.RSiena
Multipar.RSiena(ans1, 7:9)
```

Como se podrá apreciar en la prueba de Ward y considerando la similaridad en el consumo de alcohol, el efecto es significativo. Sin embargo, diluyendo el efecto de alcohol de forma más específica entre los efectos de ego y alter, los tres efectos simultáneamente no son significativos.

Ejercicios

Ejercicio 1a:

- Excluir el efecto similaridad de fumadores (*smoke1*) y estimar el modelo de nuevo.
- Realizar esto a través de la función *setEffects()* utilizando el parámetro <<include>>.
- Se aconseja que se modifique el nombre de los objetos (ejemplo: *effects1* y *ans1*) para distinguir los modelos

Ejecicio 1b:

- Cambiar los tres efectos de alcohol y convertirlos en un solo efecto de similaridad de alcohol.
- Estime nuevamente el modelo.

Estudio de redes y compartamiento

Ahora vamos a redefinir el rol de la variable consumo de alcohol la cual será incorporada como una variable dependiente adicional (que en este caso es de comportamiento). Como podrá apreciar, este tipo de especificación permite identificar efectos de selección e influencia.

Una vez más, considere la opción de ayuda:

```
?sienaDependent
```

Mire con especial atención el parámetro <<type>> dentro de la función:

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

Junte las variables del conjunto de datos para el análisis

```
NBdata <- sienaDataCreate( friendship, smoke1,
        drinking )
NBdata
NBeff <- getEffects( NBdata )
effectsDocumentation(NBeff)
NBeff <- includeEffects( NBeff, transTrip,
        transRecTrip )
NBeff <- includeEffects( NBeff, egoX, egoSqX,
        altX, altSqX, diffSqX,
         interaction1 = "drinking" )
NBeff <- includeEffects( NBeff, egoX, altX,
        simX, interaction1 = "smoke1" )
NBeff
```

A continuación podrá incluir efectos correspondientes a la variable dependiente de comportamiento:

```
?includeEffects
NBeff <- includeEffects( NBeff, avAlt,
        name="drinking",
        interaction1 = "friendship" )
NBeff
```

Defina nuevamente el algoritmo con un nuevo nombre del projecto (en este caso s50_NB)

```
myalgorithm1 <- sienaAlgorithmCreate( projname =
's50_NB' )
```

Estime nuevamente, utilizando el algoritmo con el nombre del proyecto s50_NB, los datos que incorporan una variable dependiente adicional e incluyendo los efectos correspondientes a la especificación del modelo.

```
NBans <- siena07(myalgorithm1, data = NBdata,
effects = NBeff)
```

La convergencia puede ser mejorada (considerando el *overall maximum convergence ratio*) repitiendo la estimación de la misma forma que fue realizada con anterioridad. A continuación, mire los resultados

```
NBans
```

Realice una mejor representación de los resultados:

```
siena.table(NBans, type="html", sig=TRUE)
```

Esto producriá un archivo "html" que permitirá una mejor interoperabilidad entre distintos programas (otra alternativa es "tex").

Ejercicios

Ejercicio 2a:

- Reemplace el efecto del promedio de alter por el de similaridad promedio (*avSim*)

Ejercicio 2b:

- Incorpore el efecto de fumar en el modelo que considera la variable de comportamiento consumo de alcohol y estime nuevamente.

Ejercicio 3:

- Lea la sección 13.3 y 13.4 del manual de SIENA, descargue los guiones *SelectionTables.r* e *InfluenceTables.r* en la página web de SIENA, y realice visualizaciones a través de la tabla de selección y la tabla de influencia para la variable consumo de alcohol.

Comentarios finales

En el siguiente guión se revisó cómo a través de una base de datos es posible especificar su formato para ser analizado a través del software **RSiena**. De la misma forma, se estimó un primer modelo con una sola variable dependiente y que incorporó un conjunto de efectos para ser analizados. A continuación, se realizó una segunda especificación. Por último, se revisó brevemente otro modelo que incorporó una segunda variable dependiente permitiendo así investigar efectos de selección e influencia. Al final del análisis de las distintas ilustraciones, se ofrecen algunos ejercicios que permitan a los usuarios ganar mayor familiaridad con el programa.

Los siguientes guiones de esta serie reconstruirán el análisis realizado hasta ahora con mayor profundidad, de tal forma de incrementar la familiaridad con el programa **R** y las distintas opciones que ofrece **RSiena**. 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 que habla la lengua castellana.

Agradecimientos

Agradezco particularmente a Tom A.B. Snijders (University of Oxford y University of Groningen) por su amabilidad en permitirme 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 de 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, mantenimiento y responsabilidad en su contenido en lengua castellana del cual espero reflejar adecuadamente sus intenciones originarias.

REFERENCIAS

Baerveldt, Chris; van de Bunt, Gerhard G., y de Federico de la Rüa, Ainhoa (2010). Cómo y por qué los modelos de selección difieren entre estudiantes. La influencia potencial de los tamaños de la red, el nivel de información y la pertenencia a grupos. *REDES. Revista hispana para el análisis de redes sociales*, 19(2): 272-298.

de Federico de la Rüa, Ainhoa (2003). La dinamica de las redes de amistad. La eleccion de amigos en el programa Erasmus. *REDES. Revista Hispana para el Análisis de Redes Sociales*, 4(3): 1-44.

de Federico de la Rua, Ainhoa (2005). El análisis dinámico de redes sociales con SIENA. Método, Discusión y Aplicación. *EMPIRIA. Revista de Metodología de las Ciencias Sociales*, 10, 151-181.

Díaz-José, Julio; Rendón-Medel, Roberto; Aguilar-Ávila, Jorge, y Muñoz-Rodríguez, Manrrubio (2013). Análisis dinámico de redes en la difusión de innovaciones agrícolas. *Revista Mexicana de Ciencias Agrícolas*, 4(7): 1095-1102.

Espinosa-Rada, Alejandro (2021). 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, Alejandro (2022). Modelos estocásticos orientados en el actor utilizando RSiena (II). Formato de los Datos. *REDES. Revista Hispana para el Análisis de Redes Sociales*, 33(1), 100-111

Maya-Jariego, Isidro y de Federico de la Rüa, Ainhoa (2006). El análisis dinámico de redes sociales con SIENA. En José Luis Molina, Agueda Quiroga, Joel Marti, Isidro Maya Jariego, y Ainhoa de Federico (eds.), *Talleres de autoformación con progamas informáticos de análisis de redes sociales*. Bellaterra: Universitat Autonoma de Barcelona, Servei de Publicacions.

Maya Jariego, Isidro; Holgado Ramos, Daniel y Lubbers, Miranda J. (2018). Efectos de la estructura de las redes personales en la red sociocéntrica de una cohorte de estudiantes en transición de la enseñanza secundaria a la universidad. *Universitas Psychologica*, 17(1): 123-134.

Palacios, Diego y Berger, Christian (2015). Are good students desirable friends? Evidence for friendship selection among elementary students / ¿Son los buenos estudiantes amigos deseables? Evidencia para la selección de amistad entre estudiantes de educación primaria. *Estudios de Psicología: Studies in Psychology*, 36: 496-508.

Ripley, Ruth; Snijders, Tom A.B.; Boda, Zsófia; Vörös, András, y Preciado, Paulina (2021). *Manual for SIENA version 4.0*. Oxford: University of Oxford, Department of Statistics, http://www.stats.ox.ac.uk/~snijders/siena/

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

Snijders, Tom 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.

R Core Team (2021). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. URL: https://www.R-project.org/

Remitido: 04-04-2021

Aceptado: 11-11-2021