Internal Effect Parameters and Interaction Effects in RSiena

Tom A.B. Snijders

University of Oxford
University of Groningen

June 2025

Overview

1. Internal effect parameters
2. Change statistics
3. Interaction effects
4. Elementary effects
5. Contextual effects

Internal effect parameters

Some of the effects contain an *internal effect parameter*,
a number denoted in the manual by *p*.
(These are totally different from the statistical parameters
which are the weights of the effects in the evaluation function.)

The internal effect parameter is used to define several variations of effects;
they are fixed parameters which cannot be estimated.

Internal effect parameters are set by the function **`setEffect`**.

Examples of internal effect parameter

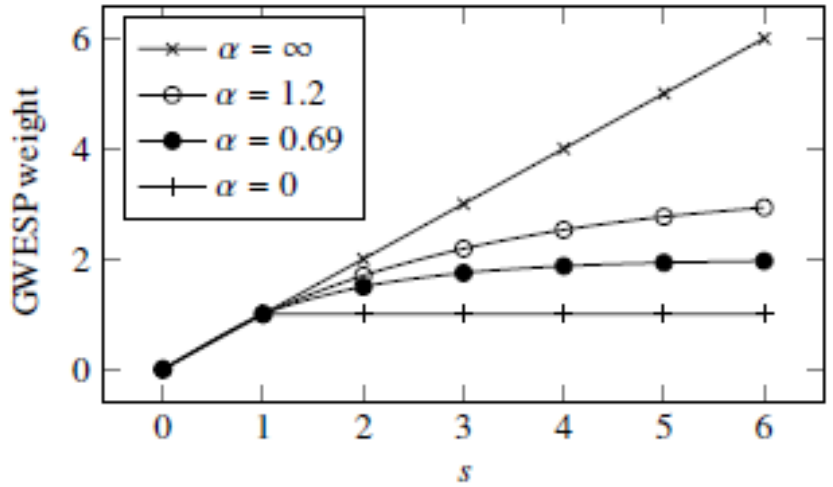
Internal effect parameters are set by the function **`setEffect`**.

E.g., if there is an effects object `effs`, the command

```
effs <- setEffect(effs, reciAct, parameter=2)
```

will specify the *[reciAct]* effect being defined by
the square root of the number of reciprocated ties of the actor.
For `parameter=2` the square root is not taken.

For the *[gwespFF]* effect,
the internal effect parameter is
$100 \times \alpha$ in the weight function.
Default 69; this refers to $\log(2) \approx 0.69$.



GWESP weight for a tie closing
s two-paths for $\alpha = \infty, 1.2, 0.69, 0$.

If nothing is specified for `parameter`, the default value will be used.
As for version 1.4.13 and later,
the value of the internal effect parameter will be given in the effect name
(earlier this was not always done, or possibly incorrectly).

This means that if there is no number present in the effect name,

(not to be confused with the shortName)

the effect has no internal effect parameter.

For effects with internal effect parameters,
please take into account their values — as described in the manual!!!

And report them (perhaps implicitly) in your presentations of results.

2. Interlude: Change statistics

Consider a SAOM with evaluation function, for network x,

$$f_i(\beta, x) = \sum_k \beta_k \, s_{ki}(x) \; .$$

The *change statistic* is defined by

$$\delta_{k,ij}(x) = s_{ki}\left(x^{(+ij)}\right) - s_{ki}\left(x^{(-ij)}\right) \; ,$$

where $x^{(+ij)}$ and $x^{(-ij)}$ are the networks x with and without tie $i \to j$, respectively.
(The same exists for ERGMs.)

These are what is used for the probabilities in a ministep.

In a ministep, the probability of toggling tie variable x_{ij},
if actor i has the opportunity to make a change, is

$$\pi_{ij}(\beta, x) \;=\; \frac{\exp\left(f_i(\beta, x^{(\pm ij)}) - f_i(\beta, x)\right)}{1 + \sum_{h \neq i} \exp\left(f_i(\beta, x^{(\pm ih)}) - f_i(\beta, x)\right)} \;,$$

where $x^{(\pm ij)}$ the network in which tie variable x_{ij}
is toggled into $1 - x_{ij}$ (and similarly for $x^{(\pm ih)}$).

Note that

$$f_i(\beta, x^{(\pm ij)}) - f_i(\beta, x) \;=\; \sum_k \beta_k \,(1 - 2\,x_{ij})\,\delta_{ij,\,k}(x) \;.$$

$(1 - 2\,x_{ij}) = 1$ if tie $i \to j$ does not exist so it can be deleted,
and –1 if it exists so it can be dropped.

This shows that the change statistics (with a + or a –) have the role
of the explanatory ~~(‘independent’)~~ variables.

For example:

$$\log\left(\frac{\mathrm{P}\{\text{add tie } i \to j \text{ to } x\}}{\mathrm{P}\{\text{leave network unchanged}\}}\right) = \sum_k \beta_k \, \delta_{ij,\,k}(x) \qquad \text{if } x_{ij} = 0 \ ;$$

$$\log\left(\frac{\mathrm{P}\{\text{drop tie } i \to j \text{ from } x\}}{\mathrm{P}\{\text{leave network unchanged}\}}\right) = -\sum_k \beta_k \, \delta_{ij,\,k}(x) \qquad \text{if } x_{ij} = 1 \ ;$$

and

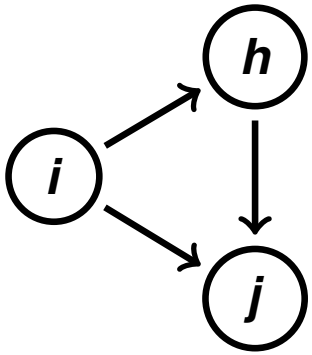
$$\log\left(\frac{\mathrm{P}\{\text{add tie } i \to j \text{ to } x\}}{\mathrm{P}\{\text{add tie } i \to h \text{ to } x\}}\right) = \sum_k \beta_k \left(\delta_{ij,\,k}(x) - \delta_{ih,\,k}(x)\right) \ ,$$

if x does not have the ties $i \to j$ or $i \to h$.

This was necessary to understand interaction effects.

3. Interaction effects

For the Stochastic Actor-Oriented Model and other network models,
the definition of interaction effects is not evident.
For example, what is the interaction of reciprocity and transitive triplets?
Which tie should be reciprocated in this figure?



Transitive triplet (i, h, j)

The *[transRecTrip]* effect
has the reciprocation between i and j.
This definition makes perfect sense;
nevertheless, the reciprocal tie
could have been put at a different place.

This leads to a distinction between
ad hoc interaction effects and *user-defined interaction effects*.
Ad hoc interaction effects have to be defined specifically in each case,
like *[transRecTrip]*.

User-defined interaction effects

User-defined interactions are defined by multiplying the change statistics.

They are created by the function **`includeInteraction`**.
Their `shortNames` are:

- ⇒ `unspInt` (for networks)
- ⇒ `unspBehInt` (for behavioral variables)
- ⇒ `contUnspInt` (for continuous behavioral variables).

The interaction effect must be 'included' to be part of the model,
but the underlying main effects need only be 'included'
if they are also required individually (which is usual but not necessary).

The number of possible user-defined interaction effects is limited, and
is set in the call of **`getEffects`** by keywords `nintn` and `behNintn`.

The difficulty with user-defined interactions in RSiena is that
the product of the change statistics does not necessarily correspond
to an evaluation effect with this product as change statistic.

Therefore a condition is required,
which uses the `interactionType` of effects.
This is described below.

There is some explanation in the manual (Section 5.11).
Further explanations are given in Sections 4.5, 4.6, and 4.7 of

https://www.stats.ox.ac.uk/~snijders/siena/Siena_algorithms.pdf

Function `includeInteraction`

This function works on the effects object.

It replaces the column entries for `effect1`, `effect2`,
and if the interaction involves three interacting effects `effect3`,
with the row numbers of the interacting effects in the effects object.

The *internal effect* parameters of the interaction effect
are 'borrowed' from the main effects.

Example: model with an interaction but without one of the main effects.

```
> themodel <- getEffects(thedata)
> themodel <- includeInteraction(themodel,recip, gwespFF)
```

```
                                                                  initial
   effectName                          shortName include fix   test  Value  parm effect1 effect2
 1 15 reciprocity                      recip     TRUE    FALSE FALSE     0     0     0        0
 2 54 GWESP I->K->J (69)               gwespFF   FALSE   FALSE FALSE     0    69     0        0
 3 419 reciprocity x GWESP I->K->J (69) unspInt   TRUE    FALSE FALSE     0     0    15       54

> themodel
   effectName                          include fix   test  initialValue parm effect1 effect2
 1 basic rate parameter friends        TRUE    FALSE FALSE    4.69604    0     0        0
 2 outdegree (density)                 TRUE    FALSE FALSE   -1.48852    0     0        0
 3 reciprocity                         TRUE    FALSE FALSE    0.00000    0     0        0
 4 reciprocity x GWESP I -> K -> J (69) TRUE   FALSE FALSE    0.00000    0    15       54
```

The default internal parameter value for **gwespFF** is $p = 69$, corresponding to an exponential weight of $\alpha = 0.01 \times p \approx \log(2)$.

Suppose you wish not to use the main effect of *[gwespFF]*,
but want to use a different internal effect parameter.

```
> themodel <- setEffect(themodel, gwespFF, param=41, include=FALSE)
```

```
   effectName              shortName include fix   test  initialValue parm
1 GWESP I -> K -> J (41) gwespFF   FALSE   FALSE FALSE           0   41
```

Since **`print.sienaEffects`** only prints `included` effects,
and **`setEffect`** does not change the interaction effect,
nothing changes now in the print of the effects object:

```
> themodel
                                                    initial
   effectName                          include    fix   test  Value     parm effect1 effect2
 1 basic rate parameter friends          TRUE    FALSE FALSE    4.69604   0    0       0
 2 outdegree (density)                   TRUE    FALSE FALSE   -1.48852   0    0       0
 3 reciprocity                           TRUE    FALSE FALSE    0.00000   0    0       0
 4 reciprocity x GWESP I -> K -> J (69) TRUE    FALSE FALSE    0.00000   0   15      54
```

However, **siena07** does know about the parameters in the main effects, even if they are not `included` in the model:

```
> (ans <- siena07(thealg, data=thedata, effects=themodel))
 Estimates, standard errors and convergence t-ratios
                                                    Estimate   Standard   Convergence
                                                               Error      t-ratio
 Rate parameters:
   0       Rate parameter                            6.4664   ( 1.2332   )
 Other parameters:
   1. eval outdegree (density)                      -2.2848   ( 0.1382   )   0.1003
   2. eval reciprocity                               1.6430   ( 0.3545   )   0.0563
   3. eval reciprocity x GWESP I -> K -> J (41)      2.6737   ( 0.6412   )   0.1047

 Overall maximum convergence ratio:    0.1691
 Total of 1766 iteration steps.
```

Network interaction effects

The `interactionType` of a network effect is defined to be `"ego"` if it can be written as

$$s_{ik}(x) = \sum_j x_{ij}\, c_{ki}(x) \ ,$$

where $c_{ki}(x)$ is independent of $(x_{i1}, \ldots, x_{in})$ and independent of j.
The change statistic then is $\Delta_{kij}(x) = c_{ki}(x)$.

The `interactionType` of a network effect is defined to be `"dyadic"` if it can be written as

$$s_{ik}(x) = \sum_j x_{ij}\, c_{kij}(x) \ ,$$

where $c_{kij}(x)$ is independent of $(x_{i1}, \ldots, x_{in})$.
The change statistic then is $\Delta_{kij}(x) = c_{kij}(x)$.

Network interaction effects (2)

Clearly, `"ego"` is a stronger condition for `interactionType` than `"dyadic"`.

For an effect with `interactionType="ego"`,
all interactions are allowed with arbitrary other effects;
for an effect with `interactionType="dyadic"`,
all interactions are allowed with other `"dyadic"` effects.

Behavior interaction effects

The `interactionType` of a behavior effect is defined to be `"OK"` if it can be written as

$$s_{ik}^{Z}(x, z) = z_i \, s_{ik}^{Z0}(x, z) \; ,$$

where $s_{ik}^{Z0}(x, z)$ is a function not depending on z_i.

(The argument x refers to a co-evolving variable, and might be null.)

This is the case for many effects.

Exceptions are, e.g., `quad` and `avSim`.

The change contribution then is

$$\Delta_{kij}^{Z}(x, z) = s_{ik}^{Z0}(x, z) \; .$$

For an effect with `interactionType="OK"`, all interactions are allowed with other `"OK"` effects.

4. Elementary effects

SAOM effects have been framed in the triple

1. evaluation
2. maintenance/endowment
3. creation

effects.

If the parameters for a creation and corresponding maintenance effect are the same, then it can be represented just as well by an evaluation effect.

These kinds of effects differ in how they contribute to the probability of a particular choice in the ministep.

The contributions to probabilities are based on
evaluation function f^{ev}
maintenance function f^{mt}
creation function f^{cr} .

Evaluation function plays a role for any step;
creation function only for upward change;
maintenance function only against downward change.

The definition is on the following page.

The probability that, given a current network x and actor i making the ministep, the network changes to $x^{\pm ij}$, is

$$\frac{\exp\left(u_i\left(x, x^{\pm ij}\right)\right)}{1 + \sum_{h \neq i} \exp\left(u_i\left(x, x^{\pm ih}\right)\right)}$$

where the objective function is

$$\begin{aligned} u_i\left(x, x^*\right) \;=\; & f_i^{\mathrm{ev}}(x^*) - f_i^{\mathrm{ev}}(x) \;+\; \Delta^+\left(x, x^*\right)\left(f_i^{\mathrm{cr}}(x^*) - f_i^{\mathrm{cr}}(x)\right) \\ & + \;\Delta^-\left(x, x^*\right)\left(f_i^{\mathrm{mt}}(x^*) - f_i^{\mathrm{mt}}(x)\right) \end{aligned}$$

and

$$\Delta^+\left(x, x^*\right) \;=\; \begin{cases} 1 & \text{if tie is created } (x^* = x^{+ij}) \\ 0 & \text{if tie is dropped, or no change} \end{cases}$$

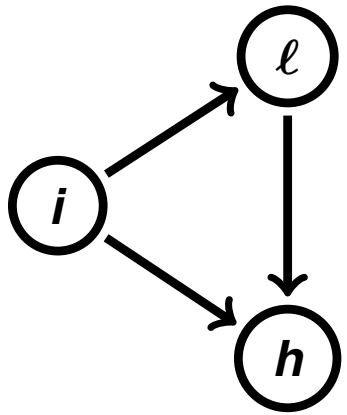
$$\Delta^-\left(x, x^*\right) \;=\; \begin{cases} 1 & \text{if tie is dropped } (x^* = x^{-ij}) \\ 0 & \text{if tie is created, or no change.} \end{cases}$$

However, not all probabilities of change can be based on changes in some (evaluation-type) function.

Example : transitive triplets

The transitive triplets effect is defined as

$$s_i(x) \;=\; \sum_{j,k} x_{ij}\, x_{ik}\, x_{kj}$$



with change statistic
(change when adding tie $i \to j$)

$$\delta_{ij}(x) \;=\; \sum_{k} x_{ik}\left(x_{kj} + x_{jk}\right)\;.$$

The first part refers to creating the tie $i \to j = h$,
the second part to creating the tie $i \to j = \ell$.

But one could be interested in only transitive closure,
as defined by closing of an open two-path ($i \to j = h$),

as distinct from creating ties to those with the same out-choices,
which is a kind of structural equivalence ($i \to j = \ell$).

This cannot be represented as a change in an evaluation function.

But one could be interested in only transitive closure,
as defined by closing of an open two-path ($i \to j = h$),

as distinct from creating ties to those with the same out-choices,
which is a kind of structural equivalence ($i \to j = \ell$).

This cannot be represented as a change in an evaluation function.

Therefore we need a different kind of effect:

But one could be interested in only transitive closure,
as defined by closing of an open two-path ($i \to j = h$),

as distinct from creating ties to those with the same out-choices,
which is a kind of structural equivalence ($i \to j = \ell$).

This cannot be represented as a change in an evaluation function.

Therefore we need a different kind of effect:

elementary effect

Elementary effect

An elementary effect is a term of the objective function $u_i\left(x, x^*\right)$
used to define change probabilities for ministeps,
referring to creation and/or maintenance of a tie $i \to j$,
without being necessarily a difference $f_i(x^{\pm ij}) - f_i(x)$ of some function f_i
(or similar with multiplication by Δ^+ or Δ^-).

If an elementary effect $e^{\mathrm{el}}_{kij}(x)$ has parameter β_k:
for adding the tie $i \to j$, $\beta_k \times e^{\mathrm{el}}_{kij}(x)$ is added to the objective function,
and for removing this tie it is subtracted.

Elementary effect

An elementary effect is a term of the objective function $u_i\left(x, x^*\right)$
used to define change probabilities for ministeps,
referring to creation and/or maintenance of a tie $i \to j$,
without being necessarily a difference $f_i(x^{\pm ij}) - f_i(x)$ of some function f_i
(or similar with multiplication by Δ^+ or Δ^-).

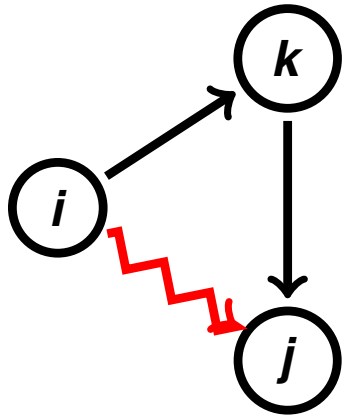
If an elementary effect $e^{\mathrm{el}}_{kij}(x)$ has parameter β_k:
for adding the tie $i \to j$, $\beta_k \times e^{\mathrm{el}}_{kij}(x)$ is added to the objective function,
and for removing this tie it is subtracted.

The evaluation function considers only the result of the ministep;
an elementary effect can express the detailed process / step
that leads to a given configuration.

Example : transTrip1 and transTrip2

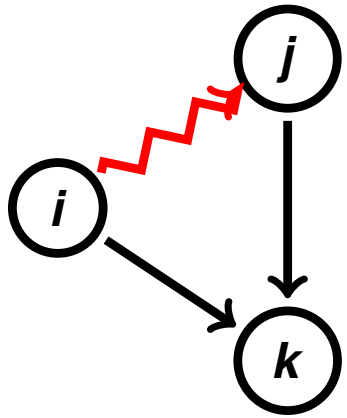
transTrip1 (transitive closure)

$$s_{ij}(x) = x_{ij} \sum_k x_{ik}\, x_{kj}$$



transTrip2
(structural equivalence outgoing ties)

$$s_{ij}(x) = x_{ij} \sum_k x_{ik}\, x_{jk}$$



Elementary effects can lead to the same configuration and therefore have the same target statistic (such as transTrip1 and transTrip2).

In such cases they cannot be distinguished empirically by estimation by the Method of Moments.

Elementary effects can lead to the same configuration and
therefore have the same target statistic
(such as transTrip1 and transTrip2).

In such cases they cannot be distinguished empirically
by estimation by the Method of Moments.

However, they can be be distinguished empirically
by estimation by the Generalized Method of Moments
(under development)
and by likelihood-based methods (Maximum Likelihood, Bayes).

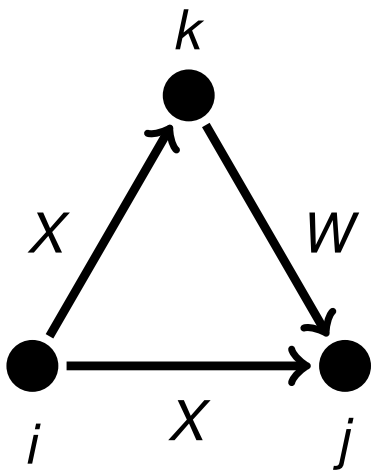
The use of elementary effects can give a more fine-grained representation of the process of network change;
but this may require more data;
like also distinction creation-maintenance requires more data.

Other example of elementary effects

XWX, ‘closure of covariate’.

E.g.: X = bullying, W = defending;
XWX = ’if *k* defends *j* and *i* bullies one of them,
then s/he will tend to bully both’.

1. XWX1: like XWX, dependent variable is only one of the XWX ties: $i \rightarrow j$
   ‘*i* bullies those who are defended by his victims’.
2. XWX2: dependent variable here is $i \rightarrow k$.
   ‘*i* bullies defenders of his victims’.



XWX1 and XWX2 are elementary effects.

Still other elementary effects

3. cl.XWX1: like XWX1 but for dependent network.
4. cl.XWX2: like XWX2 but for dependent network.

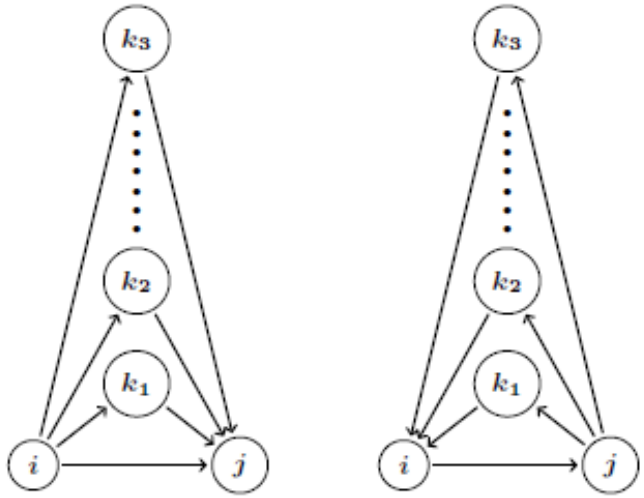
interactionType of elementary effects

Elementary effects always have `interactionType="dyadic"`,
and they can be used freely in interactions with other `"dyadic"` effects.

In Chapter 12 of the RSiena manual, it is mentioned for each effect
when it is an elementary effect.
When reporting RSiena results, this remains tacit.

E.g., the *[gwesp..]* effects are elementary,
because the focus is
on the tie $i \rightarrow j$ to be closed.

That is why there is no problem
to define the interaction `gwespFF×recip`.



5. Contextual effects

Some effects do not satisfy the conditions defining "ego" or "dyadic" effects;
e.g., `outAct` and `reciAct`,
but for these it would be nice anyway to use them in interactions.

Other versions of these are created which are "ego" or "dyadic",
meant to be used as contextual effects in interactions,
where the ‘context’ represents the conditions
under which ego makes the choice in the ministep.

They are also centered for their use in interactions.
They are represented as elementary effects,
with `interactionType="ego"` or `"dyadic"`,
and indicated by the suffix "_ego" or "_dya" in the **`shortName`**.

These are *[outAct_ego]* , *[inAct_ego]*, *[reciAct_ego]*, and *[inPop_dya]*.