What to do in RSiena if estimation is difficult

Tom A.B. Snijders

University of Oxford
University of Groningen

June, 2025

Overview

1. What to do if estimation is difficult

Estimation can be difficult for myriads of reasons.

Mostly, it is either because the observations depart too much
from what reasonably could be expected to be generated
in line with the assumptions of the Stochastic Actor-Oriented Model,
or because the model specification is too extensive
for the information contained in the data,
or because some essential elements are missing from the model specification.

Note that difficulties in obtaining convergence of the estimation procedure
may be a sign of model misspecification or overspecification.

(The converse is not true!!!)

Basic requirements

To repeat the basic requirements, with a bit of further detail:

1. Number of actors should be no too small (10 is too small) and not too large (1000 is very large).
   However, if you have many waves, small groups may be OK; if you have many groups, they could be combined in a multi-group or `sienaBayes` analysis.
2. Average degrees should be at least 2, because otherwise there is too little structure.
   However, if there are enough nodes with higher degrees and some isolates, it might be OK.
3. Jaccard similarities between successive waves should be 0.2 or higher.
   However, this rule does not apply if average degrees are strongly increasing/decreasing over the waves; also, it does not apply when there are co-evolving networks that do have high enough Jaccard coefficients; furthermore, it's just a rule of thumb and exceptions are possible.

Getting convergent estimation in normal cases

In normal cases, convergence can be achieved directly,
or with the use of `prevAns` (perhaps repeatedly):

```
ans1 <- siena07(....)
ans2 <- siena07(...., prevAns=ans1)
   ... perhaps repeat ...
```

As long as the overall maximum t-convergence ratio gets lower,
this can be continued,
perhaps with algorithms in which `nsub=1` or `nsub=2`.

However if (sooner or later) overall maximum t-convergence ratios
do not decrease any more, while being lower than (say) 0.5,
use other algorithm settings (see manual Section 6.4), e.g.:

```
algo3 <- sienaAlgorithmCreate(..., nsub=1, firstg=0.1,
                                   n2start=2000, n3=5000)
ans3 <- siena07(algo3, ..., prevAns=ans2)
```

This applies if `ans2` is rather close to convergence, but not quite.
The value of `n2start` should be considerably higher than
$100 \times (p + 7)$, where p is the number of estimated parameters;
the value of `n3` may be even higher.

If you use multiple processors (`useCluster=TRUE, nbrNodes=?`),
`n2start` and `n3` can be divided by whatever you use for `nbrNodes`.

For those who are interested, some intuitive explanation of why this works:

You can get some understanding of what happens
in the Robbins-Monro algorithm from running
`https://www.stats.ox.ac.uk/~snijders/siena/script_traceRM.R`.

The Robbins-Monro algorithm used in RSiena is explained in Section 5 of
Snijders, Tom A.B. (2001). The statistical evaluation of social network dynamics.
Sociological Methodology, 31, 361-395.

The reason for choosing `nsub=1` and a high `n2start` is that the precision of the estimate depends on the length of the last subphase in Phase 2 — provided this started with a parameter vector close enough the estimate.

The reason for choosing a high `n3` is that the estimation of the overall maximum t-convergence ratio (which is done in Phase 3) has a positive bias, and this bias will decrease when the number of simulations (`n3`) is larger.

More about `prevAns`

An earlier estimated model can also be used as `prevAns`
when building up estimated models from more simple to more complicated.

With the use of `prevAns`, the parameter values from the earlier model are copied; if the model specification has remained the same, Phase 1 is skipped.

If convergence is not very good even with repeated estimation with the `prevAns` option, perhaps it is useful to try and use function **`updateTheta`** to copy the results from the earlier estimation rather than using `prevAns`; this will use the same starting values but not skip Phase 1 of the estimation algorithm, and sometimes this turns out to lead to faster convergence.

What about the outdegree distribution

Since actors choose their outgoing ties, the outdegree distribution is crucial.

You should start data analysis by a good descriptive analysis.
The information produced by `print01Report` is a very limited description.
In any case, you should know the distribution of outdegrees and indegrees, and look at bivariate scatter plots of out/indegrees for successive waves, perhaps also reciprocated degrees.

This may indicate degree-related effects that should/might be included.
For example, *[outIso]* or perhaps *[antiInIso]*;
and/or *[outMore]* and *[outThreshold]*.

Note that *[outAct]* regulates the outdegree variance;
also *[outActSqrt]*, which downsizes larger degrees.

outdegree distribution ...

Outliers in the outdegrees may lead to problems; especially
if the many ties chosen by these actors change strongly from wave to wave.

The outliers might be real, e.g., for actors who have a special role.
Then this role could be indicated by a dummy variable (*[egoX]* and/or *[altX]*).
If there is little continuity in the choices made (perhaps because of role changes), the outgoing ties of these actors might be
represented by structural ones (manual Section 4.3.1).

This means that these ties are not themselves explained,
although they do play a role for explaining other ties.

Of course these methods should be used judiciously.

... outdegree distribution ...

Perhaps the outliers in the outdegrees indicate that the respondent
(or whatever other process leads to these observed outdegrees)
had a different understanding of what constitutes a network tie than others.

This may then also be shown by lower reciprocation rates for these actors.

One possibility to deal with this is to choose some threshold
for outdegrees to be considered as outliers, e.g., 15,
and for the actors with outdegrees higher than this threshold
set all non-reciprocated nominations to “missing” (`NA`),
while retaining their non-nominations and reciprocated nominations.
(This is similar to what Light et al., 2013, did for "extreme degree outliers".)

Light, John M., Greenan, Charlotte C., Rusby, Julie C., Nies, Kimberley M., and Snijders, Tom A. B. 2013.
Onset to First Alcohol Use in Early Adolescence: A Network Diffusion Model.
Journal of Research on Adolescence, 23, 487–499.

Multiple periods

When there are more than 2 waves, i.e., multiple periods, there is the possibility of time heterogeneity which may lead to difficulties in estimation.

Erratic average degrees across the waves are a sign of time heterogeneity.
A large number of waves (say, more than 5)
also entails a risk of time heterogeneity.

When you encounter estimation difficulties for a data set with multiple periods,
try to estimate the model for each consecutive pair of waves;
then you may see which parameter estimates are strongly different,
and give these interactions with time dummies
for the analysis with all waves.

Good model specification

Poor convergence should make you think again about the model specification.

Good models for a given data set will often converge better than poor models.

Think of covariates, interactions, non-linear transformations, ...

What was mentioned above about time heterogeneity is a case in point.

You may also think of alternative specifications of theoretically similar effects; e.g., closure may be represented by *[transTrip]*, *[transTies]*, *[gwespFF]* but also the other directions of *[gwesp..]*, and the Jaccard effects *[Jin]* and *[Jout]* (and more in the manual).

For some effects the internal effect parameter can be chosen differently; e.g., *[gwespFF]* and *[outOutAss]*.

Very large parameter values

Parameters estimated by **siena07** are scaled in such a way that
they are not very large in magnitude, mostly less than 20 (in absolute value).

Some effects associated with covariates (*[egoX]*, *[altX]*, etc.)
will have values that scale inversely with the covariates.
Actor covariates should be preferably have standard deviations between,
say, 0.1 and 10. This will normally lead to parameter values less than 10.
The same goes for dyadic covariates.

When the algorithm at some moment leads to provisional values larger than,
say, 50, this is a signal that the algorithm is diverging.

In versions of RSiena since 1.4.13, this leads to a stop of the algorithm. The default threshold value of 50 can be changed by the keyword `thetaBound`.

This will lead to a message such as

```
The update steps have led to a parameter
with maximum absolute value ...
which is larger than thetaBound = ...
***
If you wish to continue estimation in this session,
give a higher value for thetaBound.
```

where `***` gives the effects and their parameter values that were too high.

Note the “if”:
normally, you should not wish to continue estimation in this session.
(Of course, there will be exceptions.)

Usually you may regard this as a sign of divergence and treat it as such:
e.g., drop the offending effect, perhaps with `fix=TRUE, test=TRUE`.

Score-type tests

For an explanation of score-type tests, see the manual and slide set **ScoreTypeTests_s.pdf**.

They operate when effects are specified with `fix=TRUE, test=TRUE`.

When there is an effect that leads to converge difficulties,
you specify it with `fix=TRUE, test=TRUE` to gauge its importance.
Then follow the advice presented in **ScoreTypeTests_s.pdf**.

Initial values

Usually the ‘standard initial values’ that are put in automatically
into the effects object by **`getEffects`** are adequate for convergence.

These are overridden by using an earlier estimation as `prevAns` object.

You can specify other initial values for a given effect by using
keyword `initialValue` in the **`setEffect`** function.

Initial values for all effects (or all effects for some dependent variable)
can be changed by function **`updateTheta`**
into values in some earlier estimation object.

This may be utilized, e.g.,
for estimation of a co-evolution model of multiple dependent variables,
to insert values from separate estimations of each dependent variable.

2. What to do if the goodness of fit is inadequate

The fit can be checked, to some extent,
by using `sienaTimeTest` and `sienaGOF` for a variety of auxiliary functions.

1. If the fit for the degree distribution is poor,
   you may try additional degree-related effects (e.g., isolation!).
2. If the fit for triad census or geodesic distribution is poor,
   you may try additional triadic effects.
   The manual has a list of the triad codes (triad census, Section 15.1)
   which may give some help for finding how the triadic fit is deficient.

However, using additional degree-related effects
may be an improvement of the model specification by brute force.

In both cases, THINK about the model specification!!!

Are you missing additional covariate or structural effects?
Think of interactions, covariate transformations!

Time homogeneity should be checked first.

Auxiliary function **`dyadicCov`** for **`sienaGOF`** can be used
to check fit for ego-alter combinations
of monadic variables (not necessarily those used in the model);
see help page for **`sienaGOF-auxiliary`** (from version 1.2-25).

Auxiliary function **`egoAlterComb`** for **`sienaGOF`** can be used
to check fit for ego-alter combinations of dependent behavior variables
(from version 1.4-22).

3. Miscellaneous remarks

See the help pages for further information,
and Sections 5.11, 5.13, and 8.6 of the manual.

Also see the scripts on the Siena scripts webpage
and help page for **sienaGOF-auxiliary**.

First test time homogeneity, then goodness of fit.

Goodness of fit testing can be time consuming; construct the simulations with `returnDeps=TRUE` (in **siena07**) and `nsub=0` (in **sienaAlgorithmCreate**); you may explore it with a Phase 3 of reduced length (but `n3=1000` is best).

Testing of time homogeneity and goodness of fit is becoming more and more important; you will find less of it in older applications.

For larger networks, it may be difficult to get a good fit for the geodesic distribution.

The **summary** method of **sienaGOF** will give rough estimates of the improvement in fit when parameters specified with `fix=TRUE, test=TRUE` would be set free.