Maximum Likelihood Estimation in RSiena

Tom A.B. Snijders

University of Oxford
University of Groningen

2023

Principles of ML estimation in RSiena

Maximum Likelihood ('ML') estimation in RSiena uses so-called *data augmentation*: the observed networks at consecutive waves are 'augmented' by the unobserved *chain of microsteps* connecting them.

Estimation proceeds by simulating this connecting chain, estimating the parameters, and simulating the chain again given the current parameter values.

Simulation is according to a Metropolis-Hastings ('MH') algorithm.

This yields not only parameter estimates, but also a sample from the *conditional distribution of the chain of microsteps connecting the observations*.

Operation of ML estimation in RSiena

Section 6.11 of the RSiena manual contains information about ML estimation.

It is executed by function **`siena07`** when
`maxlike=TRUE` in **`sienaAlgorithmCreate`**.

ML estimation is more time-consuming than MoM estimation
but also more precise; especially for co-evolution models.

Differences between ML and MoM algorithms

The estimation uses the Robbins-Monro ('RM') algorithm with 3 phases
also used for Method of Moments ('MoM') estimation.
However, the simulation of the process in the RM steps is different.

MoM estimation in RSiena uses *forward simulation*:
starting from a given wave, the network dynamics is simulated;
this can lead, at the end of the period, to anything, and
the *average simulated estimation statistics* should be equal to the observed.

ML estimation in RSiena simulates the *connecting chain of microsteps*:
starting from a given wave, the network dynamics is simulated in such a way
that it leads to the observation at the next wave.

Thus, this is a system of *simulations-within-simulations* :
nested within each RM steps is a sequence of MH steps
in which the connecting chain is modified.

Consecutive RM steps are correlated (unlike in the MoM case).

Other differences between ML and MoM

Because the observations are to be connected by the simulated chains, and because of the principles of structurally determined values, data sequences for consecutive waves such as $x_{ij}(t) = 10,\ x_{ij}(t+1) = 1$ are logically impossible and will lead to error messages for ML estimation.

This can be solved by changing the data or by arranging the data as a multi-group data set (Manual Section 11.2) (cf. de la Haye, Embree, Punkay, Espelage, Tucker, and Green; *Social Networks*, 2017).

Also the method of joiners and leavers for changing network composition is not available for ML estimation.

Multiplication factor

The procedure for simulation of the connecting chain of microsteps
is determined by the parameters given to function **`sienaAlgorithmCreate`**.

The main parameter given to **`sienaAlgorithmCreate`** for ML estimation,
in addition to those whom we know from MoM estimation,
is the *multiplication factor* given as `mult` to **`sienaAlgorithmCreate`**.
This determines the number of MH steps per period,
which is given as `ans$nrunMH` for answer object `ans`.

Experience has shown that, to obtain good convergence,
the multiplication factor should be high enough that
autocorrelations of statistics between consecutive RM steps are less than 0.4.

These autocorrelations are given in the `projname.txt` file,
and since version 1.3.18 are given as `ans$ac3`.

(For older versions, `ans$ac` is a good approximation.)

Setting the multiplication factor

The default value is `mult=5`.

To get a lower autocorrelation for a given period,
the multiplication factor can be increased,
which then increases the number of MH steps.

The multiplication factor can be given as one number,
or a vector with length the number of periods.

Computation time is roughly proportional to the multiplication factor,
so it should not be set unnecessarily high.

Adequate values of the multiplication factor
do not depend strongly on the model specification,
and therefore can be determined from results for an empty model.

Setting the multiplication factor (continued 1)

The following procedure can be used for determining a good value for the multiplication factor. This is just one approach; after understanding it, you can use whatever way to get good values.

1. Use **`getEffects`** to specify the empty model (with or without the reciprocity effect) for your data set.
2. Specify algorithm settings by **`sienaAlgorithmCreate`** for a short estimation by maximum likelihood; e.g., with `mult=5, nsub=2, n3=500, maxlike=TRUE`, and a value for `seed` to make things replicable.
3. Estimate this model using **`siena07`**; the result will be denoted `mlans`. It does not matter that it has not yet converged.

Setting the multiplication factor (continued 2)

4. Inspect the autocorrelations `mlans$ac3` (or `mlans$ac`).
   If their maximum is less than 0.4, the multiplication factor is OK,
   and you are done;
   you may even try to reduce it, if it is much lower.
5. If the maximum is greater than 0.4, increase the multiplication factor for those periods for which `ac` for their rate parameters is greater than 0.4.
   Given that you started with `mult=5`, and if there is only one dependent variable, an example for doing this is

   ```
   mult.r <- round(20*mlans$ac[mlans$effects$basicRate], 1)
   ```

   This will set the multiplication factor to 5 for periods
   for which `ac` was exactly 0.4, and
   the others to a proportionally lower or higher value, with some rounding.
   Inspect the values `mult.r` to have an idea of their sizes.

Setting the multiplication factor (continued 3)

6. Create an algorithm object by **`sienaAlgorithmCreate`**,
   still using `nsub=2, n3=500, maxlike=TRUE`, but now with
   `mult=mult.r`,
   where `mult.r` is the new vector multiplication factor.
7. Now estimate the model again, using **`siena07`** and this algorithm object,
   and with `prevAns=mlans`. Again, convergence is not necessary.
   Give the result a new name, e.g., `mlans2`.
8. Continue as above from step 4; note that these are random simulations,
   so the autocorrelations also are somewhat variable.
9. The multiplication factor found in this way can be used also for more
   complicated models.

Standard errors

Standard errors, or (more precisely) the covariance matrix of the estimator, are calculated differently for MoM and for ML estimation.

(For statisticians: for MoM they are calculated using the delta method,

and for ML using the missing information principle.)

This requires for ML a longer Phase 3
(parameter `n3` in **`sienaAlgorithmCreate`**).
If standard errors are reported as `NA`,
this is a consequence of Phase 3 being too short.

Advisable values are `n3=3000` and higher.

Treatment of missing data

Missing data are treated in RSiena differently for MoM and ML estimation.
The treatment for ML estimation is as follows.

Simulations are for complete data sets;
if there are any missings at the initial or the end wave,
their values are included in the data augmentation,
and simulated, conditional on data and estimated parameters.

As a side-product, this leads to the estimated probability distribution
of the missing data given the observed data.

This is exploited in the imputation method for missing data in MoM estimation in Krause, Huisman, and Snijders (*Italian Journal of Applied Statistics*, 2018).

`https://www.stats.ox.ac.uk/~snijders/siena/AdSUMMissingDataMD.html`
`https://www.stats.ox.ac.uk/~snijders/siena/multipleImputation_for_RSiena.R`