Statistical Methods for Social Network Dynamics

B: Estimation and Testing

Tom A.B. Snijders

University of Groningen
University of Oxford

February 2023

Estimation

Suppose that at least 2 observations on $X(t)$ are available,
for observation moments $t_1, \ t_2$.
(Extension to more than 2 observations is straightforward.)

How to estimate θ?

Condition on $X(t_1)$:
the first observation is accepted as given,
contains in itself no observation about θ.

No assumption of a stationary network distribution.

Thus, simulations start with $X(t_1)$.

Estimation Principle: Method of Moments

Choose a suitable statistic $Z = (Z_1, \ldots, Z_K)$,
i.e., K variables which can be calculated from the network;
the statistic Z must be *sensitive* to the parameter θ
in the sense that higher values of θ_k
lead to higher values of the expected value $\mathrm{E}_\theta(Z_k)$;
denote its observed outcome by z;

determine value $\hat\theta$ of $\theta = (\rho, \beta)$ for which
observed and expected values of suitable Z statistic are equal:

$$E_{\hat\theta}\{Z\} \; = \; z \; .$$

This is a classical estimation principle in statistics:
the *Method of Moments ('MoM')*.

Questions:

- What is a suitable (K-dimensional) statistic?
  Corresponds to objective function.
- How to find this value of θ?
  By stochastic approximation (Robbins-Monro process)
  based on repeated simulations of the dynamic process,
  with parameter values
  getting closer and closer to the moment estimates.

Suitable statistics for method of moments

Assume first that $\lambda_i(x) = \rho = \theta_1$,
and 2 observation moments.

This parameter determines the expected “amount of change”.

A sensitive statistic for $\theta_1 = \rho$ is

$$C = \sum_{\substack{i,j=1 \\ i \neq j}}^{g} \mid X_{ij}(t_2) - X_{ij}(t_1) \mid ,$$

the “observed total amount of change”.

For the weights β_k in the evaluation function

$$f_i(\beta, x) = \sum_{k=1}^{L} \beta_k \, s_{ik}(x) \, ,$$

a higher value of β_k means that all actors
strive more strongly after a high value of $s_{ik}(x)$,
so $s_{ik}(x)$ will tend to be higher for all i, k.

This leads to the statistic

$$S_k = \sum_{i=1}^{n} s_{ik}(X(t_2)) \; .$$

This statistic will be sensitive to β_k :
a higher β_k will to lead to higher values of S_k.

Moment estimation will be based on the
vector of statistics

$$Z = (C, S_1, ..., S_{K-1}) \ .$$

Denote by z the observed value for Z.
The moment estimate $\hat{\theta}$ is defined as the parameter value
for which the expected value of the statistic
is equal to the observed value:

$$\boxed{\mathrm{E}_{\hat{\theta}} \{Z\} \ = \ z \ .}$$

Robbins-Monro algorithm

The moment equation $E_{\hat{\theta}}\{Z\} = z$ cannot be solved by analytical or the usual numerical procedures, because

$$\mathrm{E}_{\theta}\{Z\}$$

cannot be calculated explicitly.

However, the solution can be approximated by the Robbins-Monro (1951) method for stochastic approximation.

Iteration step:

$$\hat{\theta}_{N+1} = \hat{\theta}_N - a_N\, D^{-1}(z_N - z)\ , \tag{1}$$

where z_N is a simulation of Z with parameter $\hat{\theta}_N$,
D is a suitable matrix, and $a_N \to 0$.

Covariance matrix

The method of moments yields the covariance matrix

$$\mathrm{cov}(\hat{\theta}) \approx D_\theta^{-1} \, \Sigma_\theta \, D_\theta'^{\,-1}$$

where

$$\begin{aligned} \Sigma_\theta &= \mathrm{cov}\big\{ Z \,|\, X(t_1) = x(t_1) \big\} \\ D_\theta &= \frac{\partial}{\partial\theta} \mathrm{E}\big\{ Z \,|\, X(t_1) = x(t_1) \big\} \,. \end{aligned}$$

Matrices Σ_θ and D_θ can be estimated
from MC simulations with fixed θ.

After the presumed convergence of the algorithm
for approximately solving the moment equation,
extra simulations are carried out

(*a*) to check that indeed $\mathrm{E}_{\hat{\theta}}\{Z\} \approx z$,

(*b*) to estimate Σ_θ,

(*c*) and to estimate D_θ
using a score function algorithm
(earlier algorithm used
difference quotients and common random numbers).

Modified estimation method:

conditional estimation .

Condition on the observed numbers of differences between successive observations,

$$c_m = \sum_{i,j} \mid x_{ij}(t_{m+1}) - x_{ij}(t_m) \mid \ .$$

For continuing the simulations do not mind the values of
the time variable t,
but continue between t_m and t_{m+1} until
the observed number of differences

$$\sum_{i,j} \mid X_{ij}(t) - x_{ij}(t_m) \mid$$

is equal to the observed c_m .
This is defined as time moment t_{m+1} .

This procedure is a bit more stable; requires modified estimator of ρ_m .

In practice the differences are small.

Implementation: function `siena07`

Estimation by the Method of Moments according to the Robbins-Monro algorithm is implemented in RSiena by the function <siena07>.

This function has many other features to obtain good convergence; see the manual and `Siena_algorithms.pdf` at the SIENA website.

The options are set in a control object created by the function `sienaAlgorithmCreate`.

One of the options is the choice between MoM and ML;
but MoM is computationally much more efficient.

Computer algorithm in <siena07> has 3 phases:

1. brief phase for preliminary estimation of $\partial \mathrm{E}_\theta \{Z\} / \partial\theta$ for defining D;

Computer algorithm in <siena07> has 3 phases:

1. brief phase for preliminary estimation of $\partial \mathrm{E}_\theta \{Z\} / \partial\theta$ for defining D;
2. estimation phase with Robbins-Monro updates, where a_N remains constant in *subphases* and decreases between subphases;

   (Default nsub=4 subphases.)

Computer algorithm in <siena07> has 3 phases:

1. brief phase for preliminary estimation of $\partial \mathrm{E}_\theta \{Z\} / \partial\theta$ for defining D;
2. estimation phase with Robbins-Monro updates, where a_N remains constant in *subphases* and decreases between subphases;

   (Default nsub=4 subphases.)
3. final phase where θ remains constant at estimated value; this phase is for checking that

   $$\mathrm{E}_{\hat\theta} \{Z\} \approx z \ ,$$

   and for estimating D_θ and Σ_θ to calculate standard errors.

   (Default n3=1000 runs; for publications, more required – e.g., 5000.)

The three phases are made visible in the gui (graphical user interface) that is open during estimation (unless `batch=TRUE` in the options).

The current parameter values $\hat{\theta}_N$ are shown
(constant during Phases 1 and 3, changing during Phase 2)
as well as the 'quasi auto-correlations' of parameters during Phase 2;
negative values are a sign of being close to the final point.

If <siena07> is run with `returnTheta=TRUE`, then the parameter values obtained during Phase 3 are returned in the answer object.

This is used in script `script_traceRM.R` at the 'RSiena scripts' webpage to show plots of the updated parameters in Phase 2.

Convergence

After running the algorithm, the convergence must be checked before starting to interpret the results.

For each statistic Z_k used for estimation, we define

$$\begin{aligned} \bar{z}_{k\cdot} &= \text{average of simulated values in Phase 3;} \\ \text{sd}(z_{k\cdot}) &= \text{their standard deviation;} \\ z_k &= \text{the target value.} \end{aligned}$$

Ideally,

$$\bar{z}_{k\cdot} = z_k \text{ for all } k \;.$$

The requirement for convergence is

$$\text{tconv}_k = \frac{\left|\bar{z}_{k\cdot} - z_k\right|}{\text{sd}(z_{k\cdot})} \leq 0.1 \text{ for all } k \;.$$

tconv_k is called the *t-ratio for convergence*, and is given in the table with estimation results.

A further criterion is that

$$\text{tconv.max} \,=\, \max_w \frac{\sum_k w_k \left(\bar{z}_{k\cdot} \,-\, z_k\right)}{\text{sd}\Big(\sum_k w_k\, z_{k\cdot}\Big)} \,\leq\, 0.25 \;.$$

where the maximum is taken over all vectors w of weights of linear combinations of the Z_k.

tconv.max is called the *overall maximum convergence ratio*.

Obtaining convergent estimation

The default settings of the estimation algorithm are such, that for most data sets and models, convergence is achieved in one run.

If the model is complicated given the information available in the data, and also if some highly correlated parameters are being estimated, it can be necessary to run the estimation again, using the previous estimates as new starting values: the *prevAns* option.

If this still is not successful: see next page.

Algorithm settings for continued estimation

When the initial value is close to the final estimate,
the overall maximum convergence ratio depends on
the length of the last subphase in Phase 2.

Further, its estimation has an upward bias,
which becomes smaller for increasing length of Phase 3.

Therefore good Algorithm settings for continued estimation are:

1. `nsub = 1` (only one subphase)
2. `n2start =` ‘large’ (length of first subphase; e.g., 2000 or more)
3. `n3 =` ‘large’ (length of Phase 3; e.g., 3000 or more)
4. `firstg = 0.01` (small steps)

Also see the manual, Sections 6.2 and 6.3.

Extension: more periods

The estimation method can be extended
to more than 2 repeated observations:
observations $x(t)$ for $t = t_1, ..., t_M$.

Parameters remain the same in periods between observations
except for the basic rate of change ρ
which now is given by ρ_m for $t_m \leq t < t_{m+1}$.

For the simulations,
the simulated network $X(t)$ is reset to the observation $x(t_m)$
whenever the time parameter t passes the observation time t_m .

The statistics for the method of moments are defined as
sums of appropriate statistics calculated per period (t_m, t_{m+1}).

A special property of the SAOM is that
the interpretation of the parameters of the objective function
is not affected by the number of waves (2 or more);
for more periods (a period is the interval between two waves)
the only things added are the rate parameters per period.

However, for two or more periods,
it is necessary to check time homogeneity of the parameters
(function sienaTimeTest).
Note that, even with constant = time homogeneous parameters,
the network still may be systematically changing;
this depends on the combination of parameters with the initial network.

Maximum Likelihood estimation

For Maximum Likelihood estimation,
see the script *Maximum Likelihood Estimation in RSiena*

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

Tests

Parameters can be tested in the regular way, i.e., by the *t*-ratio

$$t_k = \frac{\hat{\beta}_k}{s.e.(\hat{\beta}_k)} ,$$

which can be referred to the standard normal null distribution.

If parameters are estimated by Maximum Likelihood, statisticians call this a Wald test[1]. Since usually in RSiena , parameters are estimated by the Method of Moments, we call this a Wald-type test.

[1] Abraham Wald, 1902-1950

Multiparameter tests

The Wald principle can also be used for multi-parameter tests
to test that several parameters together are 0
(or equal to a different hypothesized value).

This test is given by the function <multipar.RSiena>;
more generally, to test linear combinations, by <Wald.RSiena>.
The special case of testing equality of parameters is
<testSame.RSiena>.

Multiparameter tests are of practical importance because often there are high correlations between parameter estimates, and also because network concepts sometimes are represented by several effects taken together.

Score-type tests

The *score-type test* is a test that operates
without estimating the parameter.

This can be useful because estimation in RSiena is time-consuming,
and also because sometimes estimation of an effect
leads to instability of the estimation algorithm.

The score-type test for an effect is obtained by specifying

```
fix=TRUE, test=TRUE
```

in the specification by <setEffect> or <includeEffects>.

Score-type tests can be used in an exploration phase,
and also to check that parameters which are thought not to be important,
and expected to be non-significant, indeed are not significant.

For an answer object `ans` produced by <siena07> in this way,

```
score.Test(ans)
```

will show the score test for all tested parameters simultaneously.

See the help page for <score.Test> for ways to get one-parameter or multi-parameter score tests for selected sets of tested parameters.