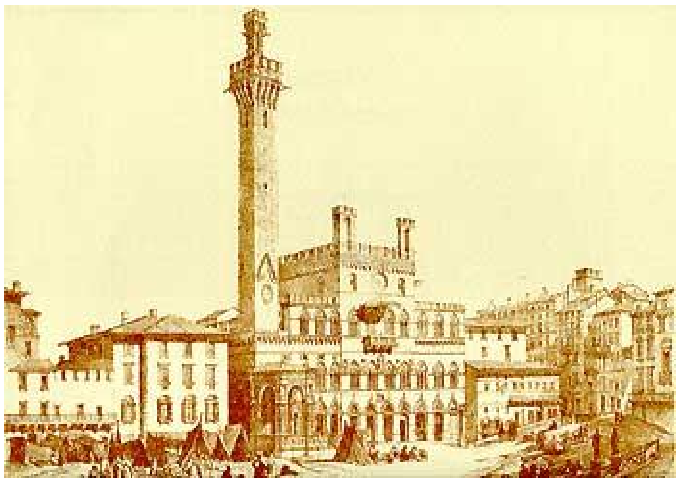
Manual for SIENA version 2.1

Tom A.B. Snijders
Christian Steglich
Michael Schweinberger
Mark Huisman

ICS, Department of Sociology
Grote Rozenstraat 31, 9712 TG Groningen, The Netherlands

February 14, 2005



Abstract

SIENA (for Simulation Investigation for Empirical Network Analysis) is a computer program that carries out the statistical estimation of models for the evolution of social networks according to the dynamic actor-oriented model of Snijders (2001, 2003) and Steglich, Snijders and Pearson (2004). It also carries out MCMC estimation for the exponential random graph model according to the procedures described in Snijders (2002) and Snijders, Pattison, Robins, and Handcock (2004). This manual gives information about SIENA version 2.1.

Contents

1 General information

SIENA[1], shorthand for Simulation Investigation for Empirical Network Analysis, is a computer program that carries out the statistical estimation of models for repeated measures of social networks according to the dynamic actor-oriented model of Snijders and van Duijn (1997), Snijders (2001), and Steglich, Snijders and Pearson (2004). The model is explained (for network evolution only) also in Snijders (2005). Some examples are presented, e.g., in van de Bunt (1999), van de Bunt, van Duijn, and Snijders (1999), and van Duijn, Zeggelink, Stokman,and Wasseur (2003). A website for SIENA is maintained at http://stat.gamma.rug.nl/snijders/siena.html .

The program also carries out MCMC estimation for the exponential random graph model ('*ERGM*'), also called p^* model, of Frank and Strauss (1986), Frank (1991), and Wasserman and Pattison (1996). This procedure is described in Snijders (2002). The model specification is discussed in Snijders, Pattison, Robins, and Handcock (2004).

This manual is about SIENA version 2.1. Further minor improvements of the version 2.1 manual are expected in the near future. The program and this manual can be downloaded from the web site, http://stat.gamma.rug.nl/stocnet/. One way to run SIENA is as part of the StOCNET program collection (Boer, Huisman, Snijders, & Zeggelink, 2003), which can be downloaded from the same website. For the operation of StOCNET, the reader is referred to the corresponding manual. If desired, SIENA can be operated also independently of StOCNET, as is explained in Section 15.

This manual consists of two parts: the user's manual and the programmer's manual. There are two parallel pdf versions: s_manx_s.pdf for screen viewing and s_manx_p.pdf for printing, where the 'x' stands for a version number. They were made with the LaTeX pdfscreen.sty package of C.V. Radhakrishnan which made it possible, e.g., to insert various hyperlinks within the manual. Both versions can be viewed and printed with the Adobe Acrobat reader. This is the print version. Section numbering may differ between the two versions.

The manual focuses on the use of SIENA for analysing the dynamics of directed networks. The case of non-directed networks is very similar, and at various points this case is described more in particular. Sections on data requirements, general operation, etc., apply as well to parameter estimation in the ERGM (non-longitudinal) model. Some specific sections are devoted to the specification of that particular model.

For getting started, one excellent option is to read the User's Manual from start to finish (leaving aside the Programmer's Manual). Another option is to focus on Sections 5 for the model specification, 6 to get a basic insight in what happens in the parameter estimation, 6.2 to understand the output file (which is meant to be as self-explanatory as possible), and 11 for the basis of getting started.

We are grateful to Peter Boer, Rob de Negro, and Evelien Zeggelink for their cooperation in the development of the StOCNET program and its alignment with SIENA. We also are grateful to NWO (Netherlands Organisation for Scientific Research) for their support to the integrated research program *The dynamics of networks and behavior* (project number 401-01-550), the project *Statistical methods for the joint development of individual behavior and peer networks* (project number 575-28-012), and the project *An open software system for the statistical analysis of social networks* (project number 405-20-20), which all contributed to the work on SIENA and StOCNET.

[1]This program was first presented at the International Conference for Computer Simulation and the Social Sciences, Cortona (Italy), September 1997, which originally was scheduled to be held in Siena. See Snijders & van Duijn (1997).

Part I
User's manual

The user's manual gives the information for using SIENA. It is advisable also to consult the user's manual of StOCNET because normally, the user will operate SIENA from within StOCNET.

2 Changes compared to earlier versions

The main changes in version 2.1 compared to version 1.98 are

1. extension by dependent actor variables, inclusion of effects related to group position and an update of the similarity effects, implementing methods in Steglich, Snijders and Pearson (2004), see Section 4.4;
2. the addition of Neyman-Rao goodness-of-fit tests according to Schweinberger (2005), see Section 7;
3. the possibility to analyse dynamics of non-directed networks, according to Snijders (2005b);
4. statistical Monte Carlo studies, see Section 19;
5. extension of the specifications of the exponential random graph ("p^*") model in line with Snijders et al. (2004), and slight modifications of the algorithm for this case to increase efficiency;
6. missing data handling is extended to covariates and dependent actor variables;
7. addition of the program SIENA08 for the multilevel analysis of multiple network evolution processes, implementing methods in Snijders and Baerveldt (2003);
8. analysing the dynamics of non-directed networks (*not yet documented*);
9. possibility to specify structural (i.e., non-random) zeros and structural ones in the adjacency matrices, see Section 4.1.1;
10. a new format for the project definition file *pname*.IN and the replacement of the internal project files *pname*.mo1 through *pname*.mo4 by files *pname*.MO for model definition and *pname*.SI for simulation directives (old project files still can be read);
11. correction of various errors.

The main changes in version 1.98 compared to version 1.95 are

1. the advanced option modeltype is added, implementing methods in Snijders (2003);
2. maximum number of actors increased to 500.

The main changes in version 1.95 compared to version 1.90 are

1. for the exponential random graph model some extra simulation options were added, and inversion steps were added to the algorithm;
2. some effects (3-star and 4-star counts) added to the exponential random graph model;

3. for changing covariates, the global rather than the periodwise mean is subtracted;
4. the program SIENA02 for data description was added.

The main changes in version 1.90 compared to version 1.70 are

1. possibility to use more than two observation moments;
2. inclusion of the exponential random graph ("p^*") model, corresponding to one observation moment;
3. possibility to have changes of composition of the network (actors leaving and/or entering);
4. changing actor covariates;
5. arbitrary codes allowed for missing data (instead of the automatic use of 6 and 9 as codes for missing data, the user now has to supply these codes explicitly);
6. small improvements in the user interface.

3 Parts of the program

The operation of the SIENA program is comprised of four main parts:

1. input of basic data description,
2. model specification,
3. estimation of parameter values using stochastic simulation,
4. simulation of the model with given and fixed parameter values.

The normal operation is to start with data input, then specify a model and estimate its parameters, and then continue with new model specifications followed by estimation or simulation. For the comparison of (nested) models, goodness-of-fit indicators can be consulted.

The program is organized in the form of *projects*. A project consists of data and the current model specification. All files internally used in a given project have the same root name, which is called the project name, and indicated here by *pname*.

The main output is written to the text file *pname*.out, auxiliary output is contained in the text file *pname*.log.

4 Input data

The main statistical method implemented in SIENA is for the analysis of repeated measures of social networks, and requires network data collected at two or more time points. It is possible to include changing actor variables (representing behavior, attitudes, outcomes, etc.) which also develop in a dynamic process, together with the social networks. As repeated measures data on social networks, at the very least, *two or more data files with digraphs* are required: the observed networks, one for each time point. The number of time points is denoted M.

The other statistical method implemented in SIENA if the parameter estimation for the exponential random graph model ('*ERGM*'). For this method, one observed network data set is required.

In addition, various kinds of variables are allowed:

1. *actor-bound* or *individual variables*, also called *actor attributes*, which can be symbolized as v_i for each actor i; these can be constant over time or changing;
   the changing individual variables can be dependent variables (changing dynamically in mutual dependence with the changing network) or independent variables (exogenously changing variables; then they are also called individual covariates).

2. *dyadic covariates*, which can be symbolized as w_{ij} for each ordered pair of actors (i, j); they are allowed only to have integer values ranging from 0 to 255. In the current version of SIENA, dyadic covariates are not allowed to change over time.

All variables must be available in ASCII data files, described in detail below. These files, the names of the corresponding variables, and the coding of missing data, must be made available to SIENA. In the StOCNET environment, files and variable names are entered in the Data dialog window, while missing data are identified in the Transformation dialog window. In the Model dialog window, network data and additional variables subsequently can be selected for SIENA analyses. This is done by first choosing SIENA from the list of available statistical methods, and then pushing the Data specification button.

Names of variables must be composed of at most 12 characters. This is because they are used as parts of the names of effects which can be included in the model, and the effect names should not be too long. The use of the default variable and file names proposed by StOCNET is not recommended.

4.1 Digraph data files

Each digraph must be contained in a separate input file in the form of an adjacency matrix, i.e., n lines each with n integer numbers, separated by blanks, each line ended by a hard return. The diagonal values are meaningless but must be present.

Although this section talks only about digraphs (directed graphs), it is also possible that all observed adjacency matrices are symmetric. This will be automatically detected by SIENA, and the program will then utilize methods for non-directed networks.

The data matrices for the two digraphs must be coded in the sense that their values are converted by the program to the 0 and 1 entries in the adjacency matrix. A set of code numbers is required for each digraph data matrix; these codes are regarded as the numbers representing a present arc in the digraph, i.e., a 1 entry in the adjacency matrix; all other numbers will be regarded as 0 entries in the adjacency matrix. Of course, there must be at least one such code number. All code numbers must be in the range from 0 to 9, except for structurally determined values (see below).

This implies that if the data are already in 0-1 format, the single code number 1 must be given. As another example, if the data matrix contains values 1 to 5 and only the values 4 and 5 are to be interpreted as present arcs, then the code numbers 4 and 5 must be given.

Code numbers for missing numbers also must be indicated. These codes must, of course, be different from the code numbers representing present arcs.

4.1.1 Structurally determined values

It is allowed that some of the values in the digraph are structurally determined, i.e., deterministic rather than random. This is analogous to the phenomenon of 'structural zeros' in contingency tables, but in SIENA not only structural zeros but also structural ones are allowed. A structural zero means that it is certain that there is no tie from actor i to actor j; a structural one means that it is certain that there is a tie. This can be, e.g., because the tie is impossible or formally imposed, respectively.

Structurally determined values are defined by reserved codes in the data matrix: the value 10 indicates a structural zero, the value 11 indicates a structural one. Structurally determined values can be different for the different time points. (The diagonal of the data matrix always is composed of structural zeros, but this does not have to be indicated in the data matrix by special codes.) The correct definition of the structurally determined values can be checked from the brief report of this in the output file, and by looking at the file *pname*.s01 (for the first time point), *pname*.s02 (second time point), etc. In these files, the structurally determined positions (structural zeros as well as structural ones) are indicated by the value 1, all others (i.e., the positions where ties are random) by the value 0.

Structural zeros offer the possibility of analyzing several networks simultaneously under the assumption that the parameters are identical. E.g., if there are three networks with 12, 20 and 15 actors, respectively, then these can be integrated into one network of $12 + 20 + 15 = 47$ actors, by specifying that ties between actors in different networks are structurally impossible. This means that the three adjacency matrices are combined in one 47×47 data file, with values 10 for all entries that refer to the tie from an actor in one network to an actor in a different network. In other words, the adjacency matrices will be composed of three diagonal blocks, and the off-diagonal blocks will have all entries equal to 10. In this example, the number of actors per network (12 to 20) is rather small to obtain good parameter estimates, but if the additional assumption of identical parameter values for the three networks is reasonable, then the combined analysis may give good estimates.

In such a case where K networks (in the preceding paragraph, the example had $K = 3$) are combined artificially into one bigger network, it will often be helpful to define $K - 1$ dummy variables at the actor level to distinguish between the K components. These dummy variables can be given effects in the rate function and in the utility function (for "ego"), which then will represent that the rate of change and the outdegree effect are different between the components, while all other parameters are the same.

4.2 Dyadic covariates

Each dyadic covariate also must be contained in a separate input file with a square data matrix, i.e., n lines each with n integer numbers, separated by blanks, each line ended by a hard return. The diagonal values are meaningless but must be present.

The reasons for restricting dyadic covariates to integer values from 0 to 255 has to do with how the data are stored internally. If the user wishes to use a dyadic covariate with a different range, this variable first must be transformed to integer values from 0 to 255. E.g., for a continuous variable ranging from 0 to 1, the most convenient way probably is to multiply by 100 (so the range becomes 0–100) and round to integer values. In the current implementation, this type of recoding cannot easily be carried out within StOCNET, but the user must do it in some other program.

4.3 Individual covariates

Individual (i.e., actor-bound) variables can be combined in one or more files. If there are k variables in one file, then this data file must contain n lines, with on each line k numbers which all are read as real numbers (i.e., a decimal point is allowed). The numbers in the file must be separated by blanks and each line must be ended by a hard return. There must not be blank lines after the last data line.

A distinction is made between constant and changing actor variables, where change refers to changes over time. Each constant actor covariate has one value per actor valid for all observation moments, and has the role of an independent variable.

Changing variables can change between observation moments. They can have the role of dependent variables (changing dynamically in mutual dependence with the changing network) or of independent variables; in the latter case, they are also called 'changing individual covariates'. Dependent variables are treated in the section below, this section is about individual variables in the role of independent variables – then they are also called individual covariates.

When changing individual variables have the role of independent variables, they are assumed to have constant values from one observation moment to the next. If observation moments for the network are $t_1, t_2, ..., t_M$, then the changing covariates should refer to the $M - 1$ moments t_1 through t_{M-1} , and the m-th value of the changing covariates is assumed to be valid for the period from moment t_m to moment t_{m+1} . The value at t_M, the last moment, does not play a role. Changing covariates, as independent variables, are meaningful only if there are 3 or more observation moments, because for 2 observation moments the distinction between constant and changing covariates is not meaningful.

Each changing individual covariate must be given in one file, containing $k = M - 1$ columns that correspond to the $M - 1$ periods between observations. It is not a problem if there is an M'th column in the file, but it will not be read.

The mean is always subtracted from the covariates. See the section on *Centering*.

4.4 Dependent action variables

From version 2 onward, SIENA also allows dependent action variables. These represent the actors' behavior, attitudes, beliefs, etc. The difference between dependent action variables and changing actor covariates is that the latter change exogenously, i.e., according to mechanisms not included in the model, while the dependent action variables change endogenously, depending on the changing network. Unlike the changing individual covariates, the values of dependent action variables are not assumed to be constant between observations.

Accordingly, each dependent action variable must be given in one file, containing $k = M$ columns, corresponding to the M observation moments.

4.5 Missing data

SIENA allows that there are some missing data on network variables, and (from version 2 onward) also on covariates and dependent action variables. These missing data must be indicated by missing data codes (to be specified in StOCNET, if SIENA is operated through StOCNET), *not* by blanks in the data set.

In the current implementation of SIENA, missing data are treated in a simple way, trying to minimize their influence on the estimation results. The simulations are carried out over all actors.

Missing data are treated separately for each period between two consecutive observations of the network. In the initial observation for each period, missing entries in the adjacency matrix are set to 0, i.e., it is assumed that there is *no* tie. Missing covariate data as well as missing entries on dependent action variables are replaced by the variable's average score at this observation moment. In the course of the simulations, however, the adjusted values of the dependent action variables and of the network variables are allowed to change.

In order to ensure a minimal impact of missing data treatment on the results of parameter estimation (method of moments estimation) and/or simulation runs, the calculation of the target statistics used for these procedures is restricted to non-missing data. When for an actor in a given period, any variable is missing that is required for calculating a contribution to such a statistic, this actor in this period does not contribute to the statistic in question. For network and dependent action variables, an actor must provide valid data both at the beginning and at the end of a period for being counted in the respective target statistics.

4.6 Composition change

SIENA can also be used to analyze networks of which the composition changes over time, because actors join or leave the network between the observations. This is described more extensively in Huisman and Snijders (2003). For this case, a data file is needed in which the *times of composition change* are given. For networks with constant composition (no entering or leaving actors), this file is omitted and the current subsection can be disregarded.

Network composition change, due to actors joining or leaving the network, is handled separately from the treatment of missing data. The digraph data files must contain all actors who are part of the network at any observation time (denoted by n) and each actor must be given a separate (and fixed) line in these files, even for observation times where the actor is not a part of the network (e.g., when the actor did not yet join or the actor already left the network). In other words, the adjacency matrix for each observation time has dimensions $n \times n$.

At these times, where the actor is not in the network, the entries of the adjacency matrix can be specified in two ways. First as missing values using missing value code(s). In the estimation procedure, these missing values of the joiners before they joined the network are regarded as 0 entries, and the missing entries of the leavers after they left the network are fixed at the last observed values. This is different from the regular missing data treatment. Note that in the initial data description the missing values of the joiners and leavers are treated as regular missing observations. This will increase the fractions of missing data and influence the initial values of the outdegree parameter.

A second way is by giving the entries a regular observed code, representing the absence or presence of an arc in the digraph (as if the actor was a part of the network). In this case, additional information on relations between joiners and other actors in the network before joining, or leavers and other actors after leaving can be used if available. Note that this second option of specifying entries always supersedes the first specification: if a valid code number is specified this will always be used.

For joiners and leavers, crucial information is contained in the times they join or leave the network (i.e., the times of composition change), which must be presented in a separate input file. This data file must contain n lines, each line representing the corresponding actor in the digraph files, with on each line four numbers. The first two concern joiners, the last two concern leavers: 1) the last observation moment at which the actor is not yet observed, 2) the time of joining (expressed as a fraction of the length of the period), 3) the last observation moment at which the actor is observed, 4) the time of leaving (also expressed as a fraction). Also actors who are part of the network at all observation moments must be given values in this file. In the following example, the number of observation moments is considered to be $M = 5$, which means there are four periods; period m starts at observation moment m and ends at $m + 1$ for $m = 1, 2, ..., 4 = M - 1$.

Example of file with times of composition change				
Present at all five observation times	0	1.0	5	0.0
Joining in period 2 at fraction 0.6 of length of period	2	0.6	5	0.0
Leaving in period 3 at fraction 0.4 of length of period	0	1.0	3	0.4
Joining in per. 1 (0.7) and leaving in per. 4 (0.2)	1	0.7	4	0.2
Leaving in per. 2 (0.6) and joining in per. 3 (0.8)	3	0.8	2	0.6

Note that for joining, the numbers 0 1.0 have a different meaning than the numbers 1 0.0. The former numbers indicate that an actor is observed at time 1 (he/she joined the network right before the first time point), the latter indicate that an actor is not observed at observation time 1 (he/she joined just after the first time point). The same holds for leavers: 5 0.0 indicates that an actor is observed at time point 5, whereas 4 1.0 indicates that an actor left right before he/she was observed at time point 5.

From the example it follows that an actor is only allowed to join, leave, join and then leave, or leave and then join the network. The time that the actor is part of the network must be an uninterrupted period. It is not allowed that an actor joins twice or leaves twice. When there is no extra information about the time at which an actor joins or leaves (in some known period), there are three options: set the fraction equal to 0.0, 0.5, or 1.0. The second option is thought to be least restrictive.

4.7 Centering

Individual as well as dyadic covariates are centered by the program in the following way.

For individual covariates, the mean value is subtracted immediately after reading the variables. For the changing covariates, this is the global mean (averaged over all periods). The values of these subtracted means are reported in the output.

For the dyadic covariates and the similarity variables derived from the individual covariates, the grand mean is calculated, stored, and subtracted during the program calculations. (Thus, dyadic covariates are treated by the program differently than individual covariates in the sense that the mean is subtracted at a different moment, but the effect is exactly the same.)

The formula for balance is a kind of dissimilarity between rows of the adjacency matrix. The mean dissimilarity is subtracted in this formula and also reported in the output. This mean dissimilarity is calculated by a formula given in Section 13.

The dependent network variable and the dependent action variables are not centered.

5 Model specification

After defining the data, the next step is to specify a model. In the StOCNET environment, this is done by clicking the Model specification button that is activated after a successful Data specification in StOCNET 's Model menu, provided that SIENA was selected from the list of available models.

The model specification consists of a selection of 'effects' for the evolution of each dependent variable (network or behavior). A list of all available effects for a given SIENA project is given in the secondary output file *pname*.log. Three types of effects are distinguished (see Snijders, 2001; Steglich, Snijders and Pearson, 2004):

- *rate function effects*
  The rate function models the speed by which the dependent variable changes; more precisely: the speed by which each network actor gets an opportunity for changing her score on the dependent variable.
  Advice: in most cases, start modeling with a constant rate function without additional rate function effects. Constant rate functions are selected by exclusively checking the 'basic rate parameter' (for network evolution) and the main rate effects (for behavioral evolution) on the model specification screen. (When there are important size or activity differences between actors, it is possible that different advice must be given, and it may be necessary to let the rate function depend on the individual covariate that indicates this size; or on the outdegree.)

- *utility function effects*
  The utility function (called *objective function* in Snijders, 2001) models the network actors' satisfaction with their local network neighborhood configuration. It is assumed that actors change their scores on the dependent variable such that they improve their total satisfaction – with a random element to represent the limited predictability of behavior. In contrast to the endowment function (described below), the utility function evaluates only the local network neighborhood configuration that results from the change under consideration (hence the reference to the utility concept). In most applications, the utility function will be the main focus of model selection.
  The network utility function normally should always contain the 'outdegree', or 'density', effect, to account for the observed network density. For directed networks, it mostly is also advisable to include the reciprocity effect, this being one of the most fundamental network effects. Likewise, behavior utility functions should normally always contain the tendency parameter, to account for the observed prevalence of the behavior.

- *endowment function effects*
  The endowment function (similar to the *gratification function* in Snijders, 2001) is an extension of the utility function that allows to distinguish between new and old network ties (when evaluating possible network changes) and between increasing or decreasing behavioral scores (when evaluating possible behavioral changes). The function models the loss of satisfaction incurred when existing network ties are dissolved or when behavioral scores are decreased to a lower value (hence the label 'endowment').
  Advice: start modeling without any endowment effects.

The estimation and simulation procedures of SIENA operate on the basis of the model specification which comprises the set of effects included in the model as described above, together with the current parameter values and the Model Type (see Section 5.2). After data input, the constant rate parameters and the outdegree effect in the network utility function have default initial values, depending on the data. All other parameter values initially are 0. The estimation process changes the current value of the parameters to the estimated values. Values of effects not included in the model are not changed by the estimation process. It is possible for the user to change parameter values and to request that some of the parameters are fixed in the estimation process at their current value.

5.1 Effects associated with covariates

For each individual covariate, there are several effects which can be included in a model specification, both in the network evolution part and in the behavioral evolution part (should there be dependent behavioral variables in the data). While the possible effects on behavioral evolution are still limited to a main effect of the covariate on the behavioral utility function, there is more choice concerning the possible effects on network evolution.

- *network rate function*
  1. the covariate's effect on the rate of network change of the actor;
- *network utility function*
  1. the covariate-similarity effect; a positive parameter implies that actors prefer ties to others with similar values on this variable – thus contributing to the network-autocorrelation of this variable, not by changing the variable, but by changing the network;
  2. the effect on the actor's activity (covariate-ego); a positive parameter will imply the tendency that actors with higher values on this covariate increase their outdegrees more rapidly;
  3. the effect on the actor's popularity to other actors (covariate-alter); a positive parameter will imply the tendency that the indegrees of actors with higher values on this covariate increase more rapidly;
  4. the interaction-effect of covariate-similarity with reciprocity;
  5. several other interaction effects of the covariate or covariate-similarity with endogenous network effects and other covariates or behavioral variables;
- *network endowment function*
  1. the covariate-similarity effect; a positive parameter indicates that being tied to similar network neighbors is preferred to both being tied to dissimilar network neighbors and creating new ties to similar others.

The usual order of importance of these covariate effects on network evolution is: utility effects are most important, followed by endowment and rate effects. Inside the group of utility effects, it is the covariate-similarity effect that is most important, follwed by the effects of covariate-ego and covariate-alter. Note that, while in principle, there is one endowment effect corresponding to each utility effect, in the current version of SIENA, the effects listed here are the only covariate-related network endowment effects implemented.

For each dyadic covariate, the following network utility effects can be included in the model for network evolution:

- *network utility function*
  1. main effect of the dyadic covariate;
  2. the interaction effect of the dyadic covariate with reciprocity;

- *network endowment function*
    1. main effect of the dyadic covariate.

The main utility effect is usually the most important. In the current version of SIENA, there are no effects of dyadic covariates on behavioral evolution.

5.2 Model Type

5.2.1 Model Type: directed networks

For directed networks, the Model Type distinguishes between the model of Snijders (2001) (Model Type 1) and that of Snijders (2003) (Model Type 2). Model Type 1 is the default model and is described in the basic publications on Stochastic Actor-Oriented Models for network dynamics.

In Model Type 2, the 'decisions' by the actors consist of two steps: first a step to increase or decrease their outdegree; when this step has been taken, the selection of the other actor towards whom a new tie is extended (if the outdegree rises) or from a an existing tie is withdrawn (if the outdegree drops). The decision by an actor to increase or decrease the number of outgoing ties is determined on the basis of only the current degree; the probabilities of increasing or decreasing the outdegree are expressed by the distributional tendency function ξ (indicated in the output as *xi*) and the volatility function ν (indicated as *nu*). Which new tie to create, or which existing tie to withdraw, depends in the usual way on the utility and endowment functions. Thus, the outdegree distribution is governed by parameters that are not connected to the parameters for the structural dynamics. The use of such an approach in statistical modeling minimizes the influence of the observed degrees on the conclusions about the structural aspects of the network dynamics. This is further explained in Snijders (2003).

For Model Type 2, in the rate function, effects connected to these functions ξ and ν are included. On the other hand, effects in the utility function that depend only on the outdegrees are canceled from the model specification, because they are not meaningful in Model Type 2. To evaluate whether Model Type 1 or Model Type 2 gives a better fit to the observed degree distribution, the output gives a comparison between the observed outdegrees and the fitted distribution of the outdegrees (as exhibited by the simulated outdegrees). For Model Type 2 this comparison is always given. For Model Type 1, this comparison is given by adding 10 to the Model Code in the advanced options. (For LaTeX users: the log file contains code that can be used to make a graph of the type given in Snijders, 2003).

For using Model Type 2, it is advised to first estimate some model according to Model Type 1 (this may be a simple model containing a reciprocity effect only, but it could also include more effects), and then – using the parameters estimated under Model Type 1 – change the specification to Model Type 2, and use the unconditional estimation method (see Section 6.3.3) (instead of the conditional method which is the default). It is likely that the very first model estimated under Model Type 2 will have results with poor convergence properties, but in such cases it is advised just to estimate the same model another time, now using the parameter values obtained under the previous Model Type 2 run as the initial values for the estimation.

To obtain a good model specification with respect to the rates of change in dependence of the outdegrees, three effects can be included:

1. the outdegrees effect
2. the factorial outdegree effect
3. the logarithmic outdegree effect.

These are the effects defined in formula (18) of Snijders (2003b) and indicated with the parameters α_1, α_2, and α_3, respectively. The user has to see from the estimation results which, or which two, out of these effects should be included to yield a good fit for the outdegrees.

The Model Type is specified in the model options as (part of) the Model Code.

5.2.2 Model Type: non-directed networks

For non-directed networks, the Model Type has five possible values, as described in Snijders (2005b).

1. Forcing model:
   one actor takes the initiative and unilaterally imposes that a tie is created or dissolved.

2. Unilateral initiative and reciprocal confirmation:
   one actor takes the initiative and proposes a new tie or dissolves an existing tie; if the actor proposes a new tie, the other has to confirm, otherwise the tie is not created.

3. Pairwise conjunctive model:
   a pair of actors is chosen and reconsider whether a tie will exist between them; a new tie is formed if both agree.

4. Pairwise disjunctive (forcing) model:
   a pair of actors is chosen and reconsider whether a tie will exist between them; a new tie is formed if at least one wishes this.

5. Pairwise compensatory (additive) model:
   a pair of actors is chosen and reconsider whether a tie will exist between them; this is based on the sum of their utilities for the existence of this tie.

In Models 1-2, where the initiative is one-sided, the rate function is comparable to the rate function in directed models. In Models 3-5, however, the pair of actors is chosen at a rate which is the *product* of the rate functions λ_i and λ_j for the two actors. This means that opportunities for change of the single tie variable x_{ij} occur at the rate $\lambda_i \times \lambda_j$. The numerical interpretation is different from that in Models 1-2.

6 Estimation

The model parameters are estimated under the specification given during the model specification part, using a stochastic approximation algorithm. In the following, the number of parameters is denoted by p. The algorithm is based on repeated (and repeated, and repeated...) simulation of the evolution process of the network. These repetitions are called 'runs' in the following. The estimation algorithm is based on comparing the observed network (obtained from the data files) to the hypothetical networks generated in the simulations.

Note that the estimation algorithm is of a stochastic nature, so the results can vary! This is of course not what you would like. For well-fitting combinations of data set and model, the estimation results obtained in different trials will be very similar. It is good to repeat the estimation process at least once for the models that are to be reported in papers or presentations, to confirm that what you report is a stable result of the algorithm.

The initial value of the parameters normally is the current value (that is, the value that the parameters have immediately before you start the estimation process); as an alternative, it is possible to start instead with a standard initial value. Usually, a sequence of models can be fitted without problems, each using the previously obtained estimate as the starting point for the new estimation procedure. Sometimes, however, problems may occur during the estimation process, which will be indicated by some kind of warning in the output file or by parameter estimates being outside a reasonably expected range. In such cases the current parameter estimates may be unsatisfactory, and using them as initial values for the new estimation process might again lead to difficulties in estimation. Therefore, when the current parameter values are unlikely and also when they were obtained after a divergent estimation algorithm, it is advisable to start the estimation algorithm with a *standard initial value*. The use of standard initial values is one of the model options.

6.1 Algorithm

During the estimation process, StOCNET transfers control to the SIENA program. The estimation algorithm has three phases:

1. In phase 1, the parameter vector is held constant at its initial value. This phase is for having a first rough estimate of the matrix of derivatives. In the case of longitudinal data, each run requires p simulations.

2. Phase 2 consists of several subphases. More subphases means a greater precision. The default number of subphases is 4. The parameter values change from run to run, reflecting the deviations between generated and observed values of the statistics. The changes in the parameter values are smaller in the later subphases.
   The program searches for parameter values where these deviations average out to 0. This is reflected by what is called the 'quasi-autocorrelations' in the output screen. These are averages of products of successively generated deviations between generated and observed statistics. It is a good sign for the convergence of the process when the quasi-autocorrelations are negative (or positive but close to 0), because this means the generated values are jumping around the observed values.

3. In phase 3, the parameter vector is held constant again, now at its final value. This phase is for estimating the covariance matrix and the matrix of derivatives used for the computation of standard errors. In the case of longitudinal data, each run again requires p simulations.
   The default number of runs in phase 3 is 500. This requires a lot of computing time, but when the number of phase 3 runs is too low, the standard errors computed are rather unreliable.

The number of subphases in phase 2, and the number of runs in phase 3, can be changed in the model options.

The user can break in and modify the estimation process in three ways:

1. it is possible to terminate the estimation;
2. in phase 2, it is possible to terminate phase 2 and continue with phase 3;
3. in addition, it is possible to change the current parameter values and restart the whole estimation process.

6.2 Output

There are three output files. All are ASCII files which can be read by any text editor. The main output is given in the *pname*.out file (recall that pname is the project name defined by the user). A brief history of what the program does is written to the file *pname*.log. The latter file also contains some supplementary output that usually is disregarded but sometimes is helpful. Some diagnostic output containing a history of the estimation algorithm which may be informative when there are convergence problems is written to the file *pname*.chk ('chk' for 'check'). This file is overwritten for each new estimation. Normally, you only need to look at *pname*.out.

The output is divided into sections indicated by a line @1, subsections indicated by a line @2, subsubsections indicated by @3, etc. For getting the main structure of the output, it is convenient to have a look at the @1 marks first.

The primary information in the output of the estimation process consists of the following three parts. Results are presented here which correspond to Table 2, column "t_1, t_3" of Snijders (2001). The results were obtained in an independent repetition of the estimation for this data set and this model specification; since the repetition was independent, the results are slightly different, illustrating the stochastic nature of the estimation algorithm.

1. Convergence check

In the first place, a convergence check is given, based on Phase 3 of the algorithm. This check considers the deviations between simulated values of the statistics and their observed values (the latter are called the 'targets'). Ideally, these deviations should be 0. Because of the stochastic nature of the algorithm, when the process has properly converged the deviations are small but not exactly equal to 0. The program calculates the averages and standard deviations of the deviations and combines these in a t-statistic (in this case, average divided by standard deviation). For longitudinal modeling, convergence is excellent when these t-values are less than 0.1 in absolute value, good when they are less than 0.2, and moderate when they are less than 0.3. The corresponding part of the output is the following.

```
Total of 1857 iterations.
Parameter estimates based on 1357
iterations, basic rate parameter as well as covariance and
derivative matrices based on 500 iterations.

Information for convergence diagnosis.
Averages, standard deviations, and t statistics for deviations from targets:
  1.     -0.236    7.006   -0.034
  2.      0.204    7.059    0.029
  3.     -1.592   22.242   -0.072

Good convergence is indicated by the t-statistics being close to zero.
```

In this case, the t-statistics are -0.034, -0.029, and -0.072, which is less than 0.1 in absolute value, so the convergence is excellent. In data exploration, if one or more of these t-statistics are larger in absolute value than 0.3, it is advisable to restart the estimation process. For results that are to be reported, it is advisable to carry out a new estimation when one or more of the t-statistics are larger in absolute value than 0.1. Large values of the averages and standard deviations are in themselves not at all a reason for concern.

For the exponential random graph (or p^*) model, the convergence of the algorithm is more problematic than for longitudinal modeling. A sharper value of the t-statistics must be found before the user may be convinced of good convergence. It is advisable to try and obtain t-values which are less than 0.15. If, even with repeated trials, the algorithm does not succeed in producing t-values less than 0.15, then the estimation results are of doubtful value.

2. Parameter values and standard errors

The next crucial part of the output is the list of estimates and standard errors. For this data set and model specification, the following result was obtained.

```
@3
Estimates and standard errors

 0. Rate parameter                                  5.4292  (   0.6920)
Other parameters:
 1. f: outdegree (density)                         -0.7648  (   0.2957)
 2. f: reciprocity                                  2.3071  (   0.5319)
 3. f: number of distances 2                       -0.5923  (   0.1407)
```

The rate parameter is the parameter called ρ in section 13.1.3 below. The value 5.4292 indicates that the estimated number of changes per actor (i.e., changes in the choices made by this actor, as reflected in the row for this actor in the adjacency matrix) between the two observations is 5.43 (rounded in view of the standard error 0.69). Note that this refers to unobserved changes, and that some of these changes may cancel (make a new choice and then withdraw it again), so the average observed number of differences per actor will be somewhat smaller than this estimated number of unobserved changes.

The other three parameters are the weights in the utility function. The terms in the utility function in this model specification are the outdegree effect defined as s_{i1} in Section 13.1.1, the reciprocity effect s_{i2}, and the number of distances 2 (indirect relations) effect, defined as s_{i5}. Therefore the estimated utility function here is

$$-0.76\, s_{i1}(x) \,+\, 2.31\, s_{i2}(x) \,-\, 0.59\, s_{i5}(x) \;.$$

The standard errors can be used to test the parameters. For the rate parameter, testing the hypothesis that it is 0 is meaningless because the fact that there are differences between the two observed networks implies that the rate of change must be positive. The weights in the utility function can be tested by t-statistics, defined as estimate divided by its standard error. (Do not confuse this t-test with the t-test for checking convergence; these are completely different although both are t ratios!) Here the t-values are, respectively, -0.7648/0.2957 = -2.59, 2.3071/0.5319 = 4.34, and -0.5923/0.1407 = -4.21. Since these are larger than 2 in absolute value, all are significant at the 0.05 significance level. It follows that there is evidence that the actors have a preference for reciprocal relations and for networks with a small number of other actors at a distance 2. The value of the outdegree parameter is not very important; it is important that this parameter is included to control for the density in the network, but as all other statistics are correlated with the density, the outdegree effect is difficult to interpret by itself.

When for some effects the parameter estimate as well as the standard error are quite large, say, when both are more than 2, and certainly when both are more than 5, then it is possible that this indicates poor convergence of the algorithm: in particular, it is possible that the effect in question does have to be included in the model to have a good fit, but the precise parameter value is poorly defined (hence the large standard error) and the significance of the effect cannot be tested with the t-ratio. This can be explored by estimating the model without this parameter, and also with this parameter fixed at some large value (see section 11.1) – whether the value is large positive or large negative depends on the direction of the effect. For the results of both model fits, it is advisable to check the fit by simulating the resulting model and considering the statistic corresponding to this particular parameter. (The indicative sizes of 2 and 5 result from experience with network effects and with effects of covariates on usual scales with standard deviations ranging between, say, 0.4 and 2. These numbers have to be modified for covariates with different standard errors.)

3. Collinearity check

After the parameter estimates, the covariance matrix of the estimates is presented. In this case it is

```
Covariance matrix of estimates (correlations below diagonal):
     0.087     -0.036      0.003
    -0.230      0.283     -0.033
     0.078     -0.440      0.020
```

The diagonal values are the variances, i.e., the squares of the standard errors (e.g., 0.087 is the square of 0.2957). Below the diagonal are the correlations. E.g., the correlation between the estimated outdegree effect and the estimated reciprocity effect is -0.230. These correlations can be used to see whether there is an important degree of collinearity between the effects. Collinearity means that several different combinations of parameter values could represent the same data pattern, in this case, the same values of the network statistics. When one or more of the correlations are very close to -1.0 or +1.0, this is a sign of collinearity. This will also lead to large standard errors of those parameters. It is then advisable to omit one of the corresponding effects from the model, because it may be redundant given the other (strongly correlated) effect. It is possible that the standard error of the retained effect becomes much smaller by omitting the other effect, which can also mean a change of the t-test from non-significance to significance.

6.3 Other remarks about the estimation algorithm

6.3.1 Changing initial parameter values for estimation

When you wish to change initial parameter values for running a new estimation procedure, this can be done in StOCNET as one of the model options. It can also be done by 'breaking in' into the SIENA program.

6.3.2 Automatic fixing of parameters

If the algorithm encounters computational problems, sometimes it tries to solve them automatically by fixing one (or more) of the parameters. This will be noticeable because a parameter is reported in the output as being fixed without your having requested this. This automatic fixing procedure is used, when in phase 1 one of the generated statistics seems to be insensitive to changes in the corresponding parameter.

This is a sign that there is little information in the data about the precise value of this parameter, when considering the neighborhood of the initial parameter values. However, it is possible that the problem is not in the parameter that is being fixed, but is caused by an incorrect starting value of this parameter or one of the other parameters.

When the warning is given that the program automatically fixed one of the parameter, try to find out what is wrong.

In the first place, check that your data were entered correctly and the coding was given correctly, and then re-specify the model or restart the estimation with other (e.g., 0) parameter values. Sometimes starting from different parameter values (e.g., the default values implied by the model option of "standard initial values") will lead to a good result. Sometimes, however, it works better to delete this effect altogether from the model.

It is also possible that the parameter does need to be included in the model but its precise value is not well-determined. Then it is best to give the parameter a large (or strongly negative) value and indeed require it to be fixed (see Section 11.1).

6.3.3 Conditional and unconditional estimation

SIENA has two methods for estimation and simulation: conditional and unconditional. They differ in the *stopping rule* for the simulations of the network evolution. In unconditional estimation, the simulations of the network evolution in each time period (and the co-evolution of the behavioral dimensions, if any are included) carry on until the predetermined time length (chosen as 1.0 for each time period between consecutive observation moments) has elapsed.

In conditional estimation, in each period the simulations run on until a stopping criterion is reached that is calculated from the observed data. Conditioning is possible for each of the dependent variables (network, or behavior), where 'conditional' means 'conditional on the observed number of changes on this dependent variable'.

Conditioning on the network variable means running simulations until the number of different entries between the initially observed network of this period and the simulated network is equal to the number of entries in the adjacency matrix that differ between the initially and the finally observed networks of this period.

Conditioning on a behavioral variable means running simulations until the sum of absolute score differences on the behavioral variable between the initially observed behavior of this period and the simulated behavior is equal to the sum of absolute score differences between the initially and the finally observed behavior of this period.

Conditional estimation is slightly more stable and efficient, because the corresponding rate parameters are not estimated by the Robbins Monro algorithm, so this method decreases the number of parameters estimated by this algorithm. Therefore, it is the default for models that do not include any dependent behavioral variables. For models including dependent behavioral variables, the default estimation type is unconditional (because in most applications, there will be no straightforward choice for the conditioning variable). The possibility to choose between unconditional and the different types of conditional estimation is one of the model options.

If there are changes in network composition (see Section 4.6), only the unconditional estimation procedure is available.

6.3.4 Automatic changes from conditional to unconditional estimation

Even though conditional estimation is slightly more efficient than unconditional estimation, there is one kind of problem that sometimes occurs with conditional estimation and which is not encountered by unconditional estimation.

It is possible (but luckily rare) that the initial parameter values were chosen in an unfortunate way such that the conditional simulation does not succeed in ever attaining the condition required by its stopping rule (see Section 6.3.3). This is detected by SIENA, which then switches automatically to unconditional estimation; after some time it switches back again to conditional estimation.

7 Goodness-of-fit

To compare competing models for network and behavior evolution, goodness-of-fit tests are indispensable. A generalized Neyman-Rao score test is implemented in SIENA which can be used for goodness-of-fit tests (see Schweinberger, 2005). In section 7.1, it is described how goodness-of-fit tests can be carried out, and in section 7.2 an example is given including interpretation. Section 7.3 considers an alternative application of the test, which may be of interest when facing convergence problems.

7.1 How-to-do

Most goodness-of-fit tests will have the following form: some model is specified and one or more parameters are restricted to some constant, in most cases 0. Such restrictions on parameters can be imposed in the StOCNET program collection by pressing the Model specifications button on the main SIENA interface, selecting the parameter of interest, pressing the Advanced button, checking the box in the column with label `t` corresponding to the parameter of interest, and specifying the value to which the parameter is restricted. Outside the StOCNET program collection, parameters can be restricted by opening the *pname*.MO file, going to the parameters of interest and setting the values in the fourth column equal to `1`. The goodness-of-fit test proceeds by simply estimating the restricted model (not the unrestricted model, with unrestricted parameters) by the standard SIENA estimation algorithm. No more information needs to be communicated. When the model is restricted, SIENA by default assumes that the restricted model is to be tested against the unrestricted model, and by default SIENA evaluates the generalized Neyman-Rao score test statistic.

7.2 Example

As an illustration, consider the following example taken from Schweinberger (2005). The data correspond to business firms (actors) holding shares of other firms, collected and studied by Pahor (2003). A simple baseline model was specified with some control parameters (taking into account, among other things, that ownership ties to other firms may depend on how large the other firm is) and in addition the outdegree parameter.

In general, the goodness-of-fit of some baseline model is tested with respect to some critical part of the model. Critical parts may be identified by substantive insight and/or explanatory analyses. It is important to realise that carrying out too many tests may result in an inflation of the type I error probability. In other words, goodness-of-fit tests are recommended only for the most critical parts of the model.

In the present example, Pahor (2003) expected on the basis of substantive reasons that firms tend to reciprocate ownership ties to ensure that the firms are ”on the same boat” (interest alignment). That is, it was desired to test whether the goodness-of-fit of the baseline model with respect to reciprocity was ”acceptable”. Hence the reciprocity parameter was added to the baseline model, but restricted to the value 0; that is, it is postulated that reciprocated ties do not have any added value given the other parameters in the model (control parameters and outdegree); in terms of estimating the model, it means that the other parameters are estimated, while reciprocity is kept constant at 0 throughout the estimation process (as can be seen on the SIENA interface), which is appealing in terms of computation time. Estimating the model and evaluating the generalized score test statistic, the following output is obtained:

```
@2
Generalized score test ⟨c⟩
-------------------------

Testing the goodness-of-fit of the model restricted by

 (1)   description:   f: reciprocity

_______________________________________________


   c = 22.6545   d.f. = 1   p-value < 0.0000

_______________________________________________
```

To grasp the meaning of the output, consider the case where the network is observed at two time points, and let R be the number of reciprocated ties at the second time point. Then it can be shown that the test statistic is some function of

Expected R under the restricted model $-$ observed R.

Thus, the test statistic has some appealing interpretation in terms of goodness-of-fit: when reciprocated ties do have added value for the firms—which means that the reciprocity parameter is not 0, other than the model assumes—then the deviation of the observed R from the R that is expected under the model will be large (large misfit), and so will be the value of the test statistic. Large values of the test statistic imply low p-values, which, in turn, suggests to abandon the model in favor of models incorporating reciprocity.

Under some conditions, the distribution of the test statistic tends, as the number of observations increases, to the chi-square distribution, where the number of degrees of freedom is equal to the number of restricted parameters. The corresponding p-value is given in the output file.

In the present case, one parameter is restricted (reciprocity), hence there is one degree of freedom `d.f. = 1`. The value of the test statistic `c = 22.65` at one degree of freedom gives `p-value < .0000`. The fact that 22.65 is remarkably large compared to the usual standard 3.84 (corresponding to significance level .05) indicates that the discrepancy between the degree of reciprocity expected by the model and the degree of reciprocity that indeed was observed must be huge. That is, it seems that the model which assumes that reciprocated ties have no added value is not tenable, and hence reciprocity should be included into the model and estimated as the other parameters.

7.2.1 Omnibus tests with more than one restriction

In the case where $K > 1$ model parameters are restricted, SIENA evaluates the test statistic with K degrees of freedom. A low p-value of the omnibus test would indicate that the goodness-of-fit of the model is intolerable. However, the omnibus test with K degrees of freedom gives no clue as to what parameters should be included into the model: the poor goodness-of-fit could be due to only one of the K restricted parameters, it could be due to two of the K restricted parameters, or due to all of them. Hence SIENA carries out, in addition to the omnibus test with K degrees of freedom, additional tests with one degree of freedom that test the single parameters one-by-one. The goodness-of-fit table looks as follows:

```
@2
Generalized score test <c>
--------------------------

Testing the goodness-of-fit of the model restricted by

 (1)   description:   f: reciprocity
 (2)   description:   f: transitivity
 (3)   description:   f: number of actors at distance 2
------------------------------------------------


   c = 32.1155   d.f. = 3   p-value < 0.0000


(1) apart:

   c =   0.1797   d.f. = 1   p-value = 0.6716

(2) apart:

   c =   7.0165   d.f. = 1   p-value = 0.0081

(3) apart:

   c =  13.3151   d.f. = 1   p-value = 0.0003
------------------------------------------------
```

In the example output, three parameters are restricted, reciprocity at value 1, transitivity and number of actors at distance 2 both at value 0. The p-value corresponding to the omnibus test indicates that the restricted model is not tenable. Looking at the separate tests, it seems that the misfit is in the first place due to transitivity and actors at distance 2. Since both parameters imply clustering in social networks, it seems that the assumptions of the baseline model with respect to the clustering of ties are incredible in the light of the observed data. Thus, it is sensible to improve the goodness-of-fit of the baseline model by including these clustering parameters, either one of them or both, and estimate them.

7.2.2 Testing homogeneity assumptions

SIENA by default assumes that the parameter values are constant across actors and periods. It may be the case that these assumptions are hardly credible in the light of substantive insight and empirical data.

Consider the assumption that all actors share the same value of some parameter, such as outdegree. Inspecting the number of ties that actors added between observation points will show variation across actors. Such variation is "natural", and does not necessarily imply that the actors do not share the same value of the outdegree parameter. However, some actors may deviate so strongly from the bulk of actors that the question may rise whether those actors have the same outdegree parameter value as the remaining actors. Therefore, SIENA admits to test homogeneity assumptions with respect to actors and periods. An outlier test can be carried out as described in Section 7.1 by specifying one actor as outlier candidate with respect to e.g. outdegree. When such outlier tests are desired, SIENA generates a goodness-of-fit table which is equivalent to the tables above. If the p-value that is found is low, it would seem that the specified actor indeed seems to deviate from the remaining actors with respect to the specified parameter, such as outdegree.

Consider now the assumption that the value of some parameter, such as outdegree, is constant across periods. Sometimes such assumptions are not defensible. For instance, due to exogenous changes in the "environment", actors may have uniformly changed strategies (corresponding to parameter values) as a response. As an example, in the above example with ownership ties between firms, governmental laws may have been introduced at some point in time which make it much less attractive to have lots of ownership ties. As a result, the outdegree parameter can be expected to be deviate from the value in previous periods. When it is tested whether the value of some parameter in some specified period is the same as in the other periods, the goodness-of-fit table that SIENA generates is once again equivalent to the tables shown above, and the interpretation of a low p-value would be that the value of the specified parameter in the specified period seems to deviate from the value of the parameter in other periods.

7.3 Alternative application: convergence problems

An alternative use of the test statistic is as follows. When convergence of the estimation algorithm is doubtful, it is sensible to restrict the model to be estimated. Either "problematic" or "non-problematic" parameters can be kept constant at preliminary estimates (estimated parameters values). Though such strategies may be doubtful in at least some cases, it may be, in other cases, the only viable option besides simply abandoning "problematic" models. The test statistic can be exploited as a guide in the process of restricting and estimating models, as small values of the test statistic indicate that the imposed restriction on the parameters is not problematic.

8 Simulation

The simulation option simulates the network evolution for fixed parameter values. This is meaningful mainly at the point that you have already estimated parameters, and then either want to check again whether the statistics used for estimation have expected values very close to their observed values, or want to compute expected values of other statistics. The statistics to be simulated can be specified in a special screen in StOCNET.

The number of runs is set at a default value of 1,000, and can be changed in the simulation options. The user can break in and terminate the simulations early.

The output file contains means, variances, covariances, and correlations of the selected statistics. The output file also contains t-statistics for the various statistics; these can be regarded as tests for the simple null hypothesis that the model specification with the current parameter values is correct.

The simulation feature can be used in the following way. Specify a model and estimate the parameters. After this estimation (supposing that it converged properly), add a number of potential effects. This number might be too large for the estimation algorithm. Therefore, do not Estimate but choose Simulate instead. The results will indicate which are the statistics for which the largest deviations (as measured by the t-statistics) occurred between simulated and observed values. Now go back to the model specification, and return to the specification for which the parameters were estimated earlier. The effects corresponding to the statistics with large t-values are candidates for now being added to the model. One should be aware, however, that such a data-driven approach leads to capitalization on chance. Since the selected effects were chosen on the basis of the large deviation between observed and expected values, the t-tests, based on the same data set, will tend to give significant results too easily.

The generated statistics for each run are also written to the file *pname*.sdt ('sdt' for 'simulation data'), so you can inspect them also more precisely. This file is overwritten each time you are simulating again. A brief history of what the program does is again written to the file *pname*.log.

8.1 Conditional and unconditional simulation

The distinction between conditional and unconditional simulation is the same for the simulation as for the estimation option of SIENA, described in Section 6.3.3.

If the conditional simulation option was chosen (which is the default) and the simulations do not succeed in achieving the condition required by its stopping rule (see Section 6.3.3), then the simulation is terminated with an error message, saying *This distance is not achieved for this parameter vector.* In this case, you are advised to change to unconditional simulation.

9 One observation: exponential random graph models

By choosing only one observation moment, the user specifies that not a model for network evolution is studied, but an exponential random graph model (‘*ERGM*’), also called a p^* model (Frank & Strauss, 1986; Frank, 1991; Wasserman & Pattison, 1996). SIENA carries out Markov chain Monte Carlo (MCMC) estimation for this model, as described in Snijders (2002). This algorithm computes a Monte Carlo approximation of the maximum likelihood estimate. However, if the model specification is not well in accordance with the data set, then the algorithm will not converge properly. This is discussed in Snijders (2002) and Handcock (2002). How to specify the model is discussed in Snijders, Pattison, Robins and Handcock (2005), focusing on how to specify transitivity. To model this concept more complicated effects are required than the traditional transitive triplet count. Also when the model specification is good, however, it may require repeated SIENA runs, each using the previously obtained estimate as the new starting value, to obtain satisfactory convergence of the algorithm. This means in practice that (1) great care is required for the model specification, (2) the user may have to tune two constants for the algorithm, called *rstep* and the *initial gain parameter*, which are discussed below. In any case, it is advisable always to chose the conditional estimation/simulation option, which means here that the total number of ties is kept fixed. For unconditional estimation, the total number of ties is a random variable. The choice between these two is made in the advanced options.

If there are structural zeros (see Section 4.1.1) and the elements of the adjacency matrix that are *not* structurally determined split the network into two or more components (which will happen in the case where several smaller networks are artificially combined into one network with structural zeros between the original smaller networks), then the conditional estimation option keeps the total number of ties constant within each component.

The program recognizes automatically if the data set is symmetric (an non-directed graph, with $x_{ij} = x_{ji}$ for all i, j) or anti-symmetric (a tournament, with $x_{ij} \neq x_{ji}$ for all $i \neq j$). In such cases, this is respected by the Metropolis-Hastings algorithm (number 5 in the list below) chosen by SIENA for the MCMC estimation, and the exponential random graph model is considered only on the set of all symmetric or all antisymmetric graphs, respectively.

The program has six possibilities for the definition of the steps in the MCMC procedure (cf. Snijders, 2002):

1. Gibbs steps for single relations x_{ij};
2. Gibbs steps for dyads (x_{ij}, x_{ji});
3. Gibbs steps for triplets (x_{ij}, x_{jh}, x_{ih}) and (x_{ij}, x_{jh}, x_{hi});
4. Metropolis Hastings steps for single relations x_{ij};
5. Metropolis Hastings steps for dyads (x_{ij}, x_{ji}) in which symmetric dyads remain symmetric and asymmetric dyads remain asymmetric. This is appropriate for symmetric and antisymmetric graphs.
6. Metropolis Hastings steps keeping the indegrees and outdegrees fixed; see Snijders and van Duijn (2002).

The choice between these types of steps is made in the model options. Some other options are available by modifying the *pname*.MO file as indicated in Section 16.2.1 below.

In the conditional option (where the number of arcs is fixed), options 1 and 4 exchange values of arcs x_{ij} and x_{hk} with $(i, j) \neq (h, k)$ with probabilities defined by the Gibbs and Metropolis-Hastings rules, respectively; option 2 changes values of dyads (x_{ij}, x_{ji}) and (x_{hk}, x_{kh}) with $(i, j) \neq (h, k)$, keeping $x_{ij} + x_{ji} + x_{hk} + x_{kh}$ constant; and option 3 changes the value of one triplet (x_{ij}, x_{jh}, x_{ih}) or (x_{ij}, x_{jh}, x_{hi}), keeping the sum $x_{ij} + x_{jh} + x_{ih}$ or $x_{ij} + x_{jh} + x_{hi}$ constant.

The number of steps for generating one exponential random graph is $r\,n^2/2d(1-d)$, where r is a constant which can be changed in the model options, and called *Run length* in the model and estimation options screen; n is the number of actors; and d is the density of the graph, truncated to lie between 0.05 and 0.95. The default value of r is 5. This value can be increased when it is doubted that the run length is sufficient to achieve convergence of the MCMC algorithm.

The algorithm uses by default a continuous chain to make successive draws from the ERGM, even for different parameter values. This is in order to improve convergence speed. For this purpose, the possibilities 1–6 mentioned above should be communicated in the model options (see p. 54) by the values 11–16.

When convergence (as evidenced by all t-statistics for convergence being less than 0.1 in absolute value) is not easy to obtain, then one can try to improve convergence in repeated runs of SIENA, with the following options, When some autocorrelations are markedly higher than 0.1, then it can help to increase the number of steps / runlength. When the provisional parameter estimate (used as initial value for the estimation algorithm) seems to be reasonably close to a satisfactory value, then decrease the initial gain parameter (see Section 10), e.g., to the value 0.001 or 0.0001.

10 Options for model type, estimation and simulation

There are several options available in SIENA. The main options concern the model type and the estimation procedure used.

Options concerning model type and estimation procedure can be accessed in the StOCNET environment via the Model specification screen's 'option' page. More detailed information is given starting at page 53.

1. There is a choice between conditional and unconditional estimation. If there are dependent action variables, the default for conditional estimation is to condition on the observed distance for the network variable; but it then is possible also to condition on the distances observed for the dependent action variables.

2. The Model Code.
   This defines the Model Type and an associated output option.
   In the longitudinal case, the meaning of this code is as follows.
   Model Codes 10 or more give extra output for evaluating the fit of the outdegree distribution and for the explained variation (Snijders, 2004);
   the integer Model Code in the unit position (i.e., Model Code itself if it is less than 10, and Model Code - 10 if the code is more than 10) defines the Model Type defined in Section 5.2.
   In the ERGM (non-longitudinal) case, the Model Code defines the kind of steps made in the MCMC algorithm. It is advised to use one of the values 11-16, because these generate a continuous chain which yields much better convergence.

3. The number of subphases in phase 2 of the estimation algorithm.
   This determines the precision of the estimate. Advice: 3 for quick preliminary investigations, 4 or 5 for serious estimations.

4. The number of runs in phase 3 of the estimation algorithm.
   This determines the precision of the estimated standard errors (and covariance matrix of the estimates), and of the t-values reported as diagnostics of the convergence. Advice: 200 for preliminary investigations when precise standard errors and t-values are not important, 500 for serious investigations, 1000 for estimations of which results are to be reported.

5. A constant used in other estimation procedures.
   In the ERGM (non-longitudinal) case, this is the multiplication factor r for the run length used in the MCMC algorithm.

6. The initial gain value, which is the step size in the starting steps of the Robbins-Monro procedure, indicated in Snijders (2001) by a_1 .

7. The choice between standard initial values (suitable estimates for the outdegree and reciprocity parameters and zero values for all other parameters) or the current parameter values as initial values for estimating new parameter values.

8. The selection of the period for which a goodness-of-fit on period homogeneity is to be carried out.

9. The selection of the effect for which a goodness-of-fit on actor homogeneity is to be carried out (1 for the outdegree effect, 2 for the reciprocity effect); if this is selected, a list of actors also has to be supplied.

10. A random number seed. If the value 0 is chosen, the program will randomly select a seed. This is advised to obtain truly random results. If results from an earlier run are to be exactly replicated, the random number seed from this earlier run can be used.

Options about the simulation runs can be accessed in the StOCNET environment via the simulation specification button on the SIENA model's main screen. This button only is activated when 'simulation' is chosen as the 'Run model'. There is one option for simulations that can be chosen here.

1. The number of runs in the straight simulations.
   Advice: the default of 1000 will usually be adequate.

Depending on the choice for conditional or unconditional estimation in the estimation options, also the simulations are run conditionally or unconditionally.

11 Getting started

For getting a first acquaintance with the model, one may use the data set collected by Gerhard van de Bunt, discussed extensively in van de Bunt (1999), van de Bunt, van Duijn, and Snijders (1999), and used as example also in Snijders (2001) and Snijders (2005). The data files are provided with the program. The digraph data files used are the two networks vrnd32t2.dat, vrnd32t4.dat. The networks are coded as 0 = unknown, 1 = best friend, 2 = friend, 3 = friendly relation, 4 = neutral, 5 = troubled relation, 6 = item non-response, 9 = actor non-response. In the Transformations screen of StOCNET, choose the values '1 2 3' as the values to be coded as 1 for the first as well as the second network. Choose '6 9' as missing data codes.

The actor attributes are in the file vars.dat. Variables are, respectively, gender ($1 = F$, $2 = M$), program, and smoking (1 = yes, 2 = no). See the references mentioned above for further information about this network and the actor attributes.

Specify the data in StOCNET by using subsequently the Data and Transformation menus (do not forget to click Apply when finishing each of these parts), then select SIENA in the Model menu and click the Data specification button. Select the network data, in temporal sequence, on the left side of the Data specification screen, then click on OK. Back on the SIENA main screen, now click on the Model specification button.

You will be requested to make some choices for the specification, the meaning of which should be clear given what is explained above. On the left hand side of the Model specification screen, you can specify utility and endowment effects, indicated by two columns of checkboxes marked 'u' and 'e', respectively. In the specification of the utility function, choose the outdegree effect, the reciprocity effect, and one other effect. In the specification of the endowment function, choose no effects at all. On the right hand side of the screen, you can specify rate function effects. At first, leave the specification of the rate function as it is (see Section 5, in which it was advised to start modeling with a constant rate function).

Then let the program estimate the parameters. You will see a screen with intermediate results: current parameter values, the differences ('deviation values') between simulated and observed statistics (these should average out to 0 if the current parameters are close to the correct estimated value), and the quasi-autocorrelations discussed in Section 6.

It is possible to intervene in the algorithm by clicking on the appropriate buttons: the current parameter values may be altered or the algorithm may be restarted or terminated. In most cases this is not necessary.

Some patience is needed to let the machine complete its three phases. How this depends on the data set and the number of parameters in the model is indicated in Section 14. After having obtained the outcomes of the estimation process, the model can be respecified: non-significant effects may be excluded (but it is advised always to retain the outdegree and the reciprocity effects) and other effects may be included.

11.1 Model choice

For the selection of an appropriate model for a given data set it is best to start with a simple model (including, e.g., 2 or 3 effects), delete non-significant effects, and add further effects in groups of 1 to 3 effects. Like in regression analysis, it is possible that an effect that is non-significant in a given model may become significant when other effects are added or deleted!

When you start working with a new data set, it is advisable first to investigate the main endogenous network effects (reciprocity, transitivity, etc.) to get an impression of what the network dynamics looks like, and later add effects of covariates.

11.1.1 Fixing parameters

Sometimes an effect must be present in the model, but its precise numerical value is not well-determined. E.g., if the network at time t_2 would contain only reciprocated choices, then the model should contain a large positive reciprocity effect but whether it has the value 3 or 5 or 10 does not make a difference. This will be reflected in the estimation process by a large estimated value and a large standard error, a derivative which is close to 0, and sometimes also by lack of convergence of the algorithm. (This type of problem also occurs in maximum likelihood estimation for logistic regression and certain other generalized linear models; see Geyer and Thompson (1992, Section 1.6) and Albert and Anderson (1984).) In such cases this effect should be fixed to some large value and not left free to be estimated. This can be specified in the model specification under the Advanced button. As another example, when the network observations are such that ties are formed but not dissolved (some entries of the adjacency matrix change from 0 to 1, but none or hardly any change from 1 to 0), then it is possible that the outdegree parameter must be fixed at some high positive value.

11.1.2 Exploring which effects to include

The present section describes an exploratory approach to model specification. A more advanced approach to testing model specifications is described in Section 7.

For an exploration of further effects to be included, the following steps may be followed:

1. Estimate a model which includes a number of basic effects;
2. Simulate the model for these parameter values but also include some other relevant statistics among the simulated statistics;
3. Look at the t-values for these other statistics; effects with large t-values are candidates for inclusion in a next model.

It should be kept in mind, however, that this exploratory approach may lead to capitalization on chance, and also that the t-value obtained as a result of the straight simulations is conditional on the fixed parameter values used, without taking into account the fact that these parameter values are estimated themselves.

It is possible that for some model specifications the data set will lead to divergence, e.g., because the data contains too little information about this effect, or because some effects are 'collinear' with each other. In such cases one must find out which are the effects causing problems, and leave these out of the model. Simulation can be helpful to distinguish between the effects which should be fixed at a high positive or negative value and the effects which should be left out because they are superfluous.

When the distribution of the outdegrees is fitted poorly (which can be investigated by the extra output requested by selecting Model Code larger than 10 in the model options), an improvement usually is possible either by including non-linear effects of the outdegrees in the utility function, or by changing to Model Type 2 (see Section 5.2).

11.2 Convergence problems

If there are convergence problems, this may have several reasons.

- The data specification was incorrect (e.g., because the coding was not given properly).
- The starting values were poor. Try restarting from the standard initial values (a certain non-zero value for the outdegree parameter, and zero values for the other parameters); or from values obtained as the estimates for a simpler model that gave no problems. The initial default parameter values can be obtained by choosing the model option "standard initial values".
  When starting estimations with Model Type 2 (see Section 5.2), there may be some problems to find suitable starting values. For Model Type 2, it is advised to start with unconditional estimation (see the model options) and a simple model, and to turn back to conditional estimation, using the current parameter values as initial estimates for new estimation runs, only when satisfactory estimates for a simple model have been found.
- Too many weak effects are included. Use a smaller number of effects, delete non-significant ones, and increase complexity step by step. Retain parameter estimates from the last (simpler) model as the initial values for the new estimation procedure, provided for this model the algorithm converged without difficulties.
- Two or more effects are included that are almost collinear in the sense that they can both explain the same observed structures. This will be seen in high absolute values of correlations between parameter estimates. In this case it may be better to exclude one of these effects from the model.
- An effect is included that is large but of which the precise value is not well-determined (see above: section on fixing parameters). This will be seen in estimates and standard errors both being large and often in divergence of the algorithm. Fix this parameter to some large value. (Note: large here means, e.g., more than 5 or less than -5; depending on the effect, of course.)

If there are problems you don't understand, but you do know something about the operation of SIENA , you could take a look at the file *pname*.log; and, if the problems occur in the estimation algorithm, at the file *pname*.chk. These files give information about what the program did, which may be helpful in diagnosing the problem. E.g., you may look in the *pname*.chk file to see if some of the parameters are associated with positive values for the so-called quasi-autocorrelations. If this happens from subphase 2.2 onward for some parameters, these may be the parameters that led to problems in the estimation algorithm (e.g., because the corresponding effect is collinear with other effects; or because they started from unfortunate starting values; or because the data set contains too little information about their value).

11.3 Composition change

Example data files for a network of changing composition are also provided with the program. These files are called vtest2.dat, vtest3.dat, and vtest4.dat. They contain the same network data as the friendship data files of van de Bunt (for these three observation times and with the same coding), except that in these data some joiners and leavers were artificially created. These actors were given the code '9' for the observation moment at which they were not part of the network. The attribute file vtestexo.dat contains the times at which the network composition changes (see also the example in Section 4.6). This file is necessary for the program to correctly include the times at which actors join or leave the network. For example, the first line of the file contains the values

```
1 0.7 3 0.0
```

which indicates that the first actor joins the network at fraction 0.7 of period 1 (the period between the first and second observation moments) and leaves the network right after the beginning of the third period, i.e., he/she does not leave the network before the last observation at the third time point. Thus, the first actor joins the network and then stays in during the whole period being analyzed.

12 Multilevel network analysis

The program Siena08.exe is a relatively simple multilevel extension to SIENA. This program must be run independently, i.e., not through StOCNET. It runs several SIENA estimations subsequently and combines the estimates in a meta-analysis or multilevel analysis according to the methods of Snijders and Baerveldt (2003).

All Siena projects to be used must already exist, i.e., each project (= data set) must have gone through SIENA01 or through StOCNET /SIENA at least once. This is for making the .mo files and transformed data files which are essentially used by Siena08.

An easy way to operate Siena08 is to make a shortcut in Windows, right-click on the shortcut and open the "properties" tab, and in the "Target" – which already contains the path and filename of the Siena08.exe file – add the projectname after the filename (separated by a space). E.g., suppose the projectname is ABC. Then there must be a project file with the name ABC.mli (the root name "ABC" can be chosen by the user, the extension name "mli" is prescribed.) If the number of network evolution projects combined in this Siena08 run is given by K, e.g. the K=3 projects with names A, B and C, then the file ABC.mli must consist of K lines each with the project name, e.g.,

```
A
B
C
```

Siena08 considers the model definition in the .mo file of the first project in the list (here project A, so in file A.mo), and applies this model definition to all projects.

To get started, try this out with a small data set, e.g. the data set included in the zip file with project CB and the subprojects CB1 CB3 CB4 . Then these three projects must be initialised first; that can be done by using StOCNET but another option is a batch file with the commands

```
start /w siena01 cb1
start /w siena01 cb3
start /w siena01 cb4
```

The batchfile must have extension .bat; it must be in a directory containing the Siena01.exe file as well as the data files; and you can run it as usual by double clicking on it.

13 Mathematical definition of effects

Here, the mathematical formulae for the definition of the effects are given. In Snijders (2001) and Steglich, Snijders and Pearson, (2004), further background to these formulae can be found. The effects are grouped into effects for modelling network evolution and effects for modelling behavioral evolution (i.e., the dynamics of dependent actor variables). Within each group of effects, the effects are listed in the order in which they appear in SIENA.

13.1 Network evolution

The model of network evolution consists of the model of actors' decisions to establish new ties or dissolve existing ties (according to *utility* and *endowment functions*) and the model of the timing of these decisions (according to the *rate function*).

13.1.1 Network utility function

The potential effects in the network utility function u^{net} (called *objective function* and denoted f in Snijders, 2001), are the following. Note that in all effects where a constants c occurs, this constant can be chosen and changed by the user. Also note that the utility effects which are a function only of the outdegree of actor i are excluded for Model Type 2. For non-directed networks, the same formulae are used, unless a different formula is given explicitly.

1. *outdegree effect* or *density effect*, defined by the outdegree
   $s^{\text{net}}_{i1}(x) = x_{i+} = \sum_j x_{ij}$,
   where $x_{ij} = 1$ indicates presence of a tie from i to j while $x_{ij} = 0$ indicates absence of this tie;

2. *reciprocity effect*, defined by the number of reciprocated ties
   $s^{\text{net}}_{i2}(x) = \sum_j x_{ij}\, x_{ji}$;

3. *transitivity effect*, defined by the number of transitive patterns in i's relations (ordered pairs of actors (j, h) to both of whom i is tied, while also j is tied to h),
   for directed networks, $s^{\text{net}}_{i3}(x) = \sum_{j,h} x_{ij}\, x_{ih}\, x_{jh}$;
   and for non-directed networks, $s^{\text{net}}_{i3}(x) = \sum_{j<h} x_{ij}\, x_{ih}\, x_{jh}$;

4. *balance*, defined by the similarity between the outgoing ties of actor i and the outgoing ties of the other actors j to whom i is tied,

   $$s^{\text{net}}_{i4}(x) = \sum_{j=1}^{n} x_{ij} \sum_{\substack{h=1\\ h\neq i,j}}^{n} (b_0 - \mid x_{ih} - x_{jh} \mid) ,$$

   where b_0 is a constant included to reduce the correlation between this effect and the outdegree effect, defined by

   $$b_0 = \frac{1}{(M-1)n(n-1)(n-2)} \sum_{m=1}^{M-1} \sum_{i,j=1}^{n} \sum_{\substack{h=1\\ h\neq i,j}}^{n} \mid x_{ih}(t_m) - x_{jh}(t_m) \mid .$$

5. *number of distances two effect*, or indirect relations effect, defined by the number of actors to whom i is indirectly tied (through one intermediary, i.e., at sociometric distance 2),
   $s^{\text{net}}_{i5}(x) = \#\{j \mid x_{ij} = 0,\, \max_h(x_{ih}\, x_{hj}) > 0\}$;

6. *popularity alter effect*, defined by $1/n$ times the sum of the indegrees of the others to whom i is tied,
$s^{\rm net}_{i6}(x) = \frac{1}{n} \sum_j x_{ij}\, x_{+j} = \frac{1}{n} \sum_j x_{ij} \sum_h x_{hj}$

7. *activity alter effect*, defined by $1/n$ times the sum of the outdegrees of the others to whom i is tied,
$s^{\rm net}_{i7}(x) = \frac{1}{n} \sum_j x_{ij}\, x_{j+} = \frac{1}{n} \sum_j x_{ij} \sum_h x_{jh}$

8. *outdegree up to c*, where c is some constant, defined by
$s^{\rm net}_{i8}(x) = \max(x_{i+}\, , \, c)$;

9. *square root outdegree - c$\times$ o.d.*, where c is some constant, defined by
$s^{\rm net}_{i9}(x) = \sqrt{x_{i+}} - c x_{i+}$,
where c is chosen to diminish the collinearity between this and the outdegree effect;

10. *squared (outdegree - c)*, where c is some constant, defined by
$s^{\rm net}_{i10}(x) = (x_{i+} - c)^2$,
where c is chosen to diminish the collinearity between this and the outdegree effect.

11. *sum of (1/(outdegree + c)*, where c is some constant, defined by
$s^{\rm net}_{i11}(x) = 1/(x_{i+} + c)$;

12. *sum of (1/(outdegree + c(outdegree + c + 1))*, where c is some constant, defined by
$s^{\rm net}_{i12}(x) = 1/(x_{i+} + c)(x_{i+} + c + 1)$;

13. *number of 3-cycles*,
$s^{\rm net}_{i13}(x) = \sum_{j,h} x_{ij}\, x_{jh}\, x_{hi}\,$;

14. *betweenness count*,
$s^{\rm net}_{i14}(x) = \sum_{j,h} x_{hi}\, x_{ij}\, (1 - x_{hj})\,$;

15. *number of dense triads*, defined as triads containing at least c ties,
$s^{\rm net}_{i15}(x) = \sum_{j,h} I\{(x_{ij} + x_{ji} + x_{ih} + x_{hi} + x_{jh} + x_{hj}\,) \geq c\}\,$,
where the 'indicator function' $I\{A\}$ is 1 if the condition A is fulfilled and 0 otherwise, and where c is either 5 or 6 (this effect is superfluous and undefined for symmetric networks);

16. *number of (unilateral) peripheral relations to dense triads*,
$s^{\rm net}_{i16}(x) = \sum_{j,h,k} x_{ij}(1 - x_{ji})(1 - x_{hi})(1 - x_{ki}) I\{(x_{jh} + x_{hj} + x_{jk} + x_{kj} + x_{hk} + x_{kh}\,) \geq c\}\,$,
where c is the same constant as in the *dense triads* effect; for symmetric networks, the 'unilateral' condition is dropped, and the definition is
$s^{\rm net}_{i16}(x) = \sum_{j,h,k} x_{ij}(1 - x_{hi})(1 - x_{ki}) I\{(x_{jh} + x_{hj} + x_{jk} + x_{kj} + x_{hk} + x_{kh}\,) \geq c\}\,$.

The effects for a dyadic covariate w_{ij} are

17. *covariate (centered) main effect*,
$s^{\rm net}_{i17}(x) = \sum_j x_{ij}\, (w_{ij} - \bar{w})$
where $\bar{w}$ is the mean value of $w_{ij}\,$;

18. *covariate (centered) $\times$ reciprocity*,
$s^{\rm net}_{i18}(x) = \sum_j x_{ij} x_{ji}\, (w_{ij} - \bar{w})$.

For actor-dependent covariates v_j (recall that these are centered internally by SIENA) as well as for dependent behavior variables (for notational simplicity here also denoted v_j), there are 10 possible effects:

19. 4 effects of *covariate-related similarity*, defined by the sum of centered similarity scores sim^v_{ij} between i and the other actors j to whom he is tied. The main effect of this type is this one:
$s^{\mathrm{net}}_{i19}(x) = \sum_j x_{ij}(\mathrm{sim}^v_{ij} - \widehat{\mathrm{sim}^v})$,
where $\widehat{\mathrm{sim}^v}$ is the mean of all similarity scores, which are defined as $\mathrm{sim}^v_{ij} = \frac{\Delta - |v_i - v_j|}{\Delta}$ with $\Delta = \max_{ij} |v_i - v_j|$ being the observed range of the covariate v;

    The first two of these effects are interactions of covariate-related similarity with network effects; the desired network effect's number needs to be given in the *parameter* column on the *advanced model specification* screen. E.g., the main effect above figures as an interaction of covariate-related similarity with the outdegree effect, the corresponding effect parameter would be 1 (for indicating interaction with the outdegree effect, which has number 1; see above). Possible parameters here are 1 through 7 (outdegree through activity-alter) and 16 (peripheral status).

    The next two effects are interactions of covariate-related similarity with other covariates or behavioral variables. The desired covariate's number needs to be given in the *parameter* column on the *advanced model specification* screen. The numbering is as follows: first come the constant actor covariates, followed by the dependent action variables and the changing actor covariates, and at the end come the dyadic covariates. If number 0 (zero) is given as parameter, then the main effect of covariate-related similarity is included.

20. 4 effects of *covariate-ego.* The main effect of this type would be the interaction of covariate-ego with the outdegree effect *covariate-related activity*, defined by i's outdegree weighted by his covariate value,
$s^{\mathrm{net}}_{i20}(x) = v_i\, x_{i+}$;

    Also here, the first two effects are interactions of covariate-ego with network effects, while the next two effects are interactions of covariate-ego with covariates. *Note that interaction of covariate-ego with other actor covariates or dependent actor variables is interpreted as an interaction of covariate-ego with the score of alter on this variable.* The selection of effects with which covariate-ego is to interact works the same way as sketched above for the interactions of covariate-based similarity.

21. *covariate-alter* or *covariate-related popularity*, defined by the sum of the covariate over all actors to whom i has a tie,
$s^{\mathrm{net}}_{i21}(x) = \sum_j x_{ij}\, v_j$;

22. *between covariate-dissimilars*, defined by the sum of dissimilarity of all pairs of actors that are connected by a shortest 2-path via actor i,
$s^{\mathrm{net}}_{i22}(x) = \sum_{j,h} x_{hi} x_{ij}(1 - x_{hj})(\widehat{\mathrm{sim}^v} - \mathrm{sim}^v_{hj})$;

Additional possible effects are documented in Steglich, Snijders and Pearson (2004) and for Model Type 2 in Snijders (2003).

13.1.2 Network endowment function

The network endowment function e^{net}, (called *gratification function* and denoted g in Snijders, 2001), is the way of modeling effects which operate in different strengths for the creation and the dissolution of relations. The potential effects in this function, and their formulae, are the same as in the utility function. However, the endowment function contributes to the probabilities of making changes only for dissolution of ties, and not for formation of ties. For further explication, consult Snijders (2001, 2005) and Steglich, Snijders and Pearson (2004).

13.1.3 Network rate function

The network rate function $\lambda^{\rm net}$ (lambda) is defined for Model Type 1 (which is the default Model Type) as a product

$$\lambda_i^{\rm net}(\rho, \alpha, x, m) = \lambda_{i1}^{\rm net}\lambda_{i2}^{\rm net}\lambda_{i3}^{\rm net}$$

of factors depending, respectively, on period m, actor covariates, and actor position (see Snijders, 2001, p. 383). The corresponding factors in the rate function are the following:

1. The dependence on the period can be represented by a simple factor

   $$\lambda_{i1}^{\rm net} = \rho_m^{\rm net}$$

   for $m = 1, ..., M-1$. If there are only $M = 2$ observations, the basic rate parameter is called $\rho^{\rm net}$.

2. The effect of actor covariates with values v_{hi} can be represented by the factor

   $$\lambda_{i2}^{\rm net} = \exp(\sum_h \alpha_h\, v_{hi})\,.$$

3. The dependence on the position of the actor can be modeled as a function of the actor's outdegree, indegree, and number of reciprocated relations. Define these by

   $$x_{i+} = \sum_j x_{ij},\ x_{+i} = \sum_j x_{ji},\ x_{i(r)} = \sum_j x_{ij}x_{ji}$$

   (recalling that $x_{ii} = 0$ for all i).
   Denoting the corresponding parameter by α_1, the dependence on the outdegree is represented by

   $$\lambda_{i3}^{\rm net} = \frac{x_{i+}}{n-1}\exp(\alpha_1) + \left(1 - \frac{x_{i+}}{n-1}\right)\exp(-\alpha_1).$$

   This formula is motivated in Snijders and Van Duijn (1997). This defines a linear function of the outdegree, parametrized in such a way that it is necessarily positive.
   For a general dependence on the outdegree, indegree, and number of reciprocated relations, one can use an average of such terms, the second and third one depending in the analogous way on x_{+i} and $x_{i(r)}$, respectively.
   Also a reciprocal dependence on outdegrees can be specified; this can be meaningful because the rate effect of the outdegree becoming a value 1 higher might become smaller and smaller as the outdegree increases. Denoting the corresponding parameter by α_2, this effect multiplies the factor λ_{i3} by

   $$1 + \frac{\alpha_2}{1 + x_{i+}}\,.$$

   This function was chosen so that for a parameter $\alpha_2 = 0$, there is no effect (multiplication by a factor 1); and no problems (division by 0) occur when $x_{i+} = 0$.

13.1.4 Network rate function for Model Type 2

For Model Type 2 (see Section 5.2), the network rate function is defined according to Snijders (2003) by

$$\begin{aligned}\rho_m \, \lambda_{i+}(s) &= \rho_m \, \frac{\nu(s)\,\xi(s)}{1 + \xi(s)} \,, \\ \rho_m \, \lambda_{i-}(s) &= \rho_m \, \frac{\nu(s-1)}{1 + \xi(s-1)} \,,\end{aligned}$$

where $\rho_m \, \lambda_{i+}(s)$ and $\rho_m \, \lambda_{i-}(s)$ represent, respectively, the rate at which an actor of current outdegree s increases, or decreases, his outdegree by 1. The parameter ρ_m is a multiplicative effect of the observation period.

Function ξ (*xi*) is called the distributional tendency function and is represented according to Snijders (2003, formula (17)) by

$$\xi(s) = \exp\left(\alpha_1 - \alpha_2 \log(s+1) - \frac{\alpha_3}{s+1}\right) \,.$$

where the names given in SIENA are

- α_1 : outdegrees effect;
- α_2 : logarithmic outdegree effect;
- α_3 : factorial outdegree effect.

The reasons for these names and interpretation of the effects can be found in Snijders (2003). To the exponent also effects of actor covariates can be added.

The so-called volatility function ν (*nu*) is defined as

$$\nu(s) = \left(1 + \alpha_4 \frac{1}{s+1}\right) \,.$$

Also to this exponent effects of actor covariates can be added.

13.2 Behavioral evolution

The model of the dynamics of a dependent actor variable consists of a model of actors' decisions (according to *utility* and *endowment functions*) and a model of the timing of these decisions (according to a *rate function*), just like the model for the network dynamics. The decisions now do not concern the creation or dissolution of network ties, but whether an actor increases or decreases his score on the dependent actor variable by one, or keeps it as it is.

13.2.1 Behavioral utility function

Effects for the behavioral utility function u^{beh} can be selected from the following:

1. *behavioral tendency effect*,
   $s^{\text{beh}}_{i1}(x) = z_i \,,$
   where z_i denotes the value of the dependent behavioral variable of actor i;
2. *similarity effect*, defined by the sum of centered similarity scores sim^z_{ij} between i and the other actors j to whom he is tied,
   $s^{\text{beh}}_{i2}(x) = \sum_j x_{ij}(\text{sim}^z_{ij} - \widehat{\text{sim}^z})$;

3. *indegree effect*,
   $s^{\text{beh}}_{i3}(x) = z_i \sum_j x_{ji}$;

4. *outdegree effect*,
   $s^{\text{beh}}_{i4}(x) = z_i \sum_j x_{ij}$;

5. *indegree up to c effect*, where c is a constant between 1 and $n-1$,
   $s^{\text{beh}}_{i5}(x) = z_i I x_{+i} \le c$,
   where again IA denotes the indicator function of the condition A;

6. *similarity $\times$ reciprocity effect*, defined by the sum of centered similarity scores sim^z_{ij} between i and the other actors j to whom he is reciprocally tied,
   $s^{\text{beh}}_{i6}(x) = \sum_j x_{ij} x_{ji} (\text{sim}^z_{ij} - \widehat{\text{sim}^z})$;

7. *dense triads effect*, defined by the number of dense triads in which actor i is located,
   $s^{\text{beh}}_{i7}(x) = z_i s^{\text{net}}_{i15}(x)$;

8. *peripheral effect*, defined by the number of dense triads to which actor i stands in a unilateral-peripheral relation,
   $s^{\text{beh}}_{i8}(x) = z_i s^{\text{net}}_{i16}(x)$.

For each actor-dependent covariate v_j (recall that these are centered internally by SIENA) as well as for each of the other dependent behavior variables (for notational simplicity here also denoted v_j), there is one main effect:

9. *covariate effect*,
   $s^{\text{beh}}_{i9}(x) = z_i v_i$.

Additional possible effects are documented in Steglich, Snijders and Pearson (2004).

13.2.2 Behavioral endowment function

Also the behavioral model knows the distinction between utility and endowment effects. The formulae of the effects that can be included in the behavioral endowment function e^{beh} are the same as those given for the behavioral utility function. However, they enter calculation of the endowment function only when the actor considers decreasing his behavioral score by one unit (downward steps), not when upward steps (or no change) are considered. For more details, consult Steglich, Snijders and Pearson (2004).

13.2.3 Behavioral rate function

The behavioral rate function λ^{beh} consists of a constant term per period,

$$\lambda^{\text{beh}}_i = \rho^{\text{beh}}_m$$

for $m = 1, ..., M-1$.

13.3 Exponential random graph distribution

The exponential random graph distribution, which is used if there is only one observation, is defined by the probability function

$$\text{P}_\theta\{X = x\} = \exp\left(\theta' u(x) - \psi(\theta)\right) \ ,$$

where $u(x)$ is a vector of statistics. The following statistics are available. The selection of statistics is discussed extensively in Snijders et al. (2004), with attention especially to statistics 16–19.

Note that SIENA will note whether the observed graph (x_{ij}) is symmetric or not, and choose accordingly between the statistics for undirected and directed graphs.

1. For undirected graphs, the number of edges $\sum_{i<j} x_{ij}$;
   for directed graphs, the number of arcs $\sum_{i\neq j} x_{ij}$.
2. The number of reciprocated relations $\sum_{i<j} x_{ij}\, x_{ji}$.
3. The number of out-twostars $\sum_i \sum_{h<k} x_{ih}\, x_{ik}$.
4. The number of in-twostars $\sum_i \sum_{h<k} x_{hi}\, x_{ki}$.
5. The number of two-paths (mixed twostars) given for undirected graphs by $\frac{1}{2}\sum_i \sum_{h\neq k} x_{hi}\, x_{ik}$ and for directed graphs by $\sum_i \sum_{h\neq k} x_{hi}\, x_{ik}$.
6. For undirected graphs, the number of transitive triads $\frac{1}{6}\sum_{i,j,h} x_{ij}\, x_{ih}\, x_{jh}$;
   for directed graphs, the number of transitive triplets $\sum_{i,j,h} x_{ij}\, x_{ih}\, x_{jh}$.
7. The number of three-cycles given for undirected graphs by $\frac{1}{6}\sum_{i,j,h} x_{ij}\, x_{jh}\, x_{hi}$ and for directed graphs by $\frac{1}{3}\sum_{i,j,h} x_{ij}\, x_{jh}\, x_{hi}$.
8. The number of out-threestars $\sum_i \binom{x_{i+}}{3}$.
9. The number of in-threestars $\sum_i \binom{x_{+i}}{3}$.
10. The number of out-fourstars $\sum_i \binom{x_{i+}}{4}$.
11. The number of in-fourstars $\sum_i \binom{x_{+i}}{4}$.
12. The sum of reciprocal outdegrees $\sum_i 1/(x_{i+} + c)$ for some constant c.
13. The sum of transformed outdegrees $\sum_i 1/[(x_{i+} + c)(x_{i+} + c + 1)]$ for some constant c.
14. The number of pairs directly and indirectly connected,
    i.e., tied pairs (i, j) for which there exists at least one h such that $x_{ih} = x_{hj} = 1$,
    i.e., tied pairs for which there is at least one twopath from i to j;
    for undirected graphs this is $\sum_{i<j} x_{ij}\ \max_{h\neq i,j}\{x_{ih}\, x_{hj}\}$, for directed graphs it is $\sum_{i\neq j} x_{ij}\ \max_{h\neq i,j}\{x_{ih}\, x_{hj}\}$.
15. The number of indirectly connected pairs, i.e., pairs (i, j) for which there is at least one twopath from i to j;
    for undirected graphs this is $\sum_{i<j} \max_{h\neq i,j}\{x_{ih}\, x_{hj}\}$, for directed graphs it is $\sum_{i\neq j} \max_{h\neq i,j}\{x_{ih}\, x_{hj}\}$.
16. The alternating k-out-stars combination
    $$c^2 \sum_{i=1}^{n} \left\{ \left(1 - \frac{1}{c}\right)^{x_{i+}} + \frac{x_{i+}}{c} - 1 \right\} ,$$
    for some value c.

17. The alternating k-in-stars combination

$$c^2 \sum_{i=1}^{n} \left\{ \left(1 - \frac{1}{c}\right)^{x_{+i}} + \frac{x_{+i}}{c} - 1 \right\}$$

for some value c.

18. For undirected graphs, the related k-triangles statistic

$$c \sum_{i<j} x_{ij} \left\{ 1 - \left(1 - \frac{1}{c}\right)^{L_{2ij}} \right\}$$

for some value c, where L_{2ij} is the number of two-paths from i to j, $L_{2ij} = \sum_h x_{ih} x_{hj}$; for directed graphs, the formula is

$$c \sum_{i,j} x_{ij} \left\{ 1 - \left(1 - \frac{1}{c}\right)^{L_{2ij}} \right\} ,$$

with the same definition of L_{2ij}.

19. For undirected graphs, the k-parallel two-paths statistic

$$c \sum_{i<j} \left\{ 1 - \left(1 - \frac{1}{\lambda}\right)^{L_{2ij}} \right\}$$

which formula is replaced for directed graphs by

$$c \sum_{i,j} \left\{ 1 - \left(1 - \frac{1}{\lambda}\right)^{L_{2ij}} \right\} .$$

20. For each dyadic covariate w_{ij}, the sum $\sum_{i,j} x_{ij}\, w_{ij}$.

21. For each dyadic covariate w_{ij}, the associated reciprocity effect defined by $\sum_{i,j} x_{ij}\, x_{ji}\, w_{ij}$.

22. For each individual covariate v_i (changing or constant; recall that all covariates are centered), three effects are included.
The first is the v_i-related popularity effect $\sum_i x_{+i}\, v_i$;

23. next is the v_i-related activity effect $\sum_i x_{i+}\, v_i$;

24. finally the v_i-related dissimilarity effect $\sum_{i,j} x_{ij} \left(|\, v_i - v_j \,| - \bar{d} \right)$
where $\bar{d}$ is the mean of all $|\, v_i - v_j \,|$ values.

14 Limitations and time use

The estimation algorithm, being based on iterative simulation, is time consuming. The time needed is approximately proportional to $p^2 n^a C$ where p is the number of parameters, n is the number of actors, the power a is some number between 1 and 2, and C is the number of relations that changed between time m and time $m+1$, summed over $m = 1$ to $M-1$. For data sets with 30 to 40 actors and something like 5 parameters, the estimation process takes a few minutes on a fast PC. The number of actors n should not give a problem up to, say, 200. For large data sets and models, the estimation process may take more minutes up to several hours.

Section 20 indicates the constants in the program that define limitations for the data sets used.

Part II
Programmer's manual

The programmer's manual will not be important for most users. It is intended for those who wish to run SIENA outside of the StOCNET environment, for those who want to know what all the *pname.** files are all about, and for those who want to have a look inside the source code.

The SIENA program consists of a basic computation part programmed by the authors of this manual in Turbo Pascal and Delphi; associated with the StOCNET windows shell, programmed by Peter Boer and Rob de Negro in Delphi, with first Evelien Zeggelink, then Mark Huisman, and later Christian Steglich as the project leader. The computational part can be used both directly and from the windows shell. The StOCNET windows shell is much easier for data specification and model definition.

15 Running SIENA outside of StOCNET

The present computational part is composed of six executable programs. These programs are:

1. SIENA01.EXE for the basic data input, using an existing basic information file (see below);
2. SIENA02.EXE for data description;
3. SIENA04.EXE for confirmation of the model specification;
4. SIENA05.EXE for simulations with fixed parameter values;
5. SIENA07.EXE for parameter estimation;
6. SIENA08.EXE for multilevel analysis.

These programs can be used also independently of StOCNET. To run them, the project name *must* be given in a command line, e.g.

```
SIENA01 bunt
```

if `bunt` is the name of the project, and there exists a `bunt.in` file. This `bunt` is called a *command line parameter*. There are the following four ways to specify a command with a command line parameter in Windows:

1. The command line can be given at the DOS prompt (in a Windows environment);
2. it can be given in the Windows "Run" command (for Windows 98 and higher);
3. a batch file with extension BAT can be created, e.g., with filename SIE1.BAT, containing the single line
   ```
   SIENA01 bunt
   ```
   so that the program SIENA01 will be executed when the batch file is called (it may be necessary to close the DOS window after the program is ready);
4. or a shortcut to the executable file can be made, where the project name is indicated in the "target" in the "properties" of the shortcut.

The third and fourth ways are the most straightforward in Windows.

Note that a shortcut is made by opening Windows Explorer, giving a right mouse-click on the executable file, and then giving a left mouse-click on "Create Shortcut". When the shortcut has been made, the project name of the SIENA project must be added to the shortcut as follows: give a right mouse-click on the shortcut, then give a left mouse-click on "properties", and in the "Target" field add a space and the project name after the path-plus-filename of the executable file.

To run SIENA outside of StOCNET, the steps taken are the following.

1. Write the basic information file *pname*.IN which describes the data files and variable names, according to Section 16.1. This file must be in ASCII (text) format.

2. Make shortcuts or batch files as indicated above for each of the programs SIENA01 and SIENA07; and, if desired, also for SIENA02, SIENA04, and SIENA05.

3. Give the session name (indicated here as *pname*) as the command line parameter in the shortcuts or batch files.

4. Click on the shortcut or batch file for SIENA01. (This should be done only once to create the project, because calling SIENA01 for an existing project will overwrite the output file!)

5. Open the file *pname*.MO in a text editor, edit it to obtain the desired model specification (see section 16.2.1), and save it as a text (ASCII) file.

6. Click on the shortcut or batch file for SIENA07.

The last two steps – modifying the *pname*.MO file and running SIENA07 – can be repeated as much as one likes.

Some of the model-defining statistics in the SIENA model contain numbers, or fixed parameters, that can be set at other values by the user. Examples are the value 3 in "outdegree up to 3" for the longitudinal version of SIENA, and the value 2 in the linear combination of k-out-stars in the non-longitudinal ("p^*") version of SIENA. (These numbers should not be confused with the statistical parameters that are estimated by SIENA07.) These values are represented in the *pname*.MO file as the last of the 6 values in the line for this effect. When these values are modified, the change must be put into effect by calling SIENA04.

Other possibilities are:

7. Get some basic descriptive statistics by running SIENA02.

8. Simulate the model for fixed parameter values by running SIENA05. The statistics to be simulated are indicated in the file *pname*.si, which can be modified to add to or delete from the list of simulated statistics.

16 SIENA files

Internally the following files are used. Recall that *pname* is the name of the project, which the user can choose at will.

16.1 Basic information file

The basic information file is called *pname*.IN, and contains the definition of the numbers of cases and variables, the names of the files in which data are initially stored and their codes (including missing data identification), and the names of the variables. This file is written by StOCNET when the data are defined, and can also be written by any text editor that can produce ASCII files. It is read by SIENA01.EXE. The basic information file contains up to eight sections, each starting with a row containing the section number (@1 through @8). These sections must have the following contents:

1. Section @1 contains basic information about type and amount of data. This section is required for all SIENA projects. It must contain nine rows, each starting with an integer number:
   - number of observations of the network (1 for exponential random graph modeling, 2 or more for modeling network evolution over time; denoted by M);
   - number of actors (denoted further by n);
   - number of dependent network variables (must be equal to one in the current version of SIENA). The network data are further specified in Section @2;
   - number of dependent actor variables. Possible dependent actor variables are further specified in Section @3;
   - number of files with constant actor covariates (further specified in Section @4);
   - number of files with changing actor variables (further specified in Section @5);
   - number of constant dyadic covariates (further specified in Section @6);
   - number of exogenous changing dyadic covariates (must be equal to zero in the current version of SIENA; Section @7 is reserved for the specification of this type of covariate data);
   - indicator of file with times of composition change (0 means no change of network composition, 1 means composition changes). Possible composition change data are further specified in Section @8.
2. Section @2 contains information about the network data, as follows:
   - for each of the M network observations, the following three lines:
     - a line with the name of the data file;
     - a line with the codes that are regarded as a present arc in the digraph;
     - a line with the codes that are regarded as missing data;
   - a line with the name of the network variable.

   All codes should be in the range from 0 to 9.
3. Section @3 contains information about the dependent actor variables, in this format:
   - for each dependent actor variable, the following three lines:

    - a line with the name of the data file;
    - a line with the code that is regarded as missing data;
    - a line with the name of the variable.

4. Section @4 contains information about constant actor covariates, as follows:
   - For each file containing such covariates, there must be the following lines:
     - a line with the name of the data file;
     - a line with the number of variables in this file;
     - for each variable:
       * a line with the code for missing data;
       * a line with the name of the variable.

   Note that this format differs from the one used in Sections @3 and @5 through @7 because here, data files can potentially contain more than one covariate, while in these other sections, only one variable is given per file.
5. Section @5 contains information about changing actor covariates, in the same format as the dependent actor variables are given:
   - for each changing actor covariate, the following three lines:
     - a line with the name of the data file;
     - a line with the code that is regarded as missing data.
     - a line with the name of the covariate.
6. Section @6 contains information about constant dyadic covariates, in the same format:
   - for each constant dyadic covariate, the following three lines:
     - a line with the name of the data file;
     - a line with the code that is regarded as missing data.
     - a line with the name of the covariate.
7. Section @7 refers to changing dyadic covariates, which cannot be handled by SIENA yet.
8. Section @8 contains information about network composition change, namely:
   - a line with the name of the file containing times of network composition change.

Whenever a certain type of data is not present, leave out the entire section in the file *pname*.IN corresponding to this type. For example, if you do not have any files containing changing actor covariates, leave out Section @5. If there are problems in reading the basic input file, try deleting superfluous blanks and/or empty rows. See to it that the basic input file is an ASCII text file, with numbers separated by blanks, lines separated by hard returns.

The variable names given in the input file will be used in the output files. If no names are provided and SIENA is run in the StOCNET environment, SIENA uses the default variable names generated by StOCNET in the StOCNET Data menu. This is not recommendable because it can lead to identical names for different variables. If no names are provided and SIENA is run outside of the StOCNET environment, SIENA uses its own default variable names, and this problem does not occur.

An example for the basic input file is the following file bunt.IN. This refers to data files that are included with the program, collected by Gerhard van de Bunt. This example, which contains a file with three covariates, is used in van de Bunt (1999) and in van de Bunt, van Duijn, and Snijders (1999).

```
@1[general information about SIENA project ⟨bunt⟩:]
2 [number of waves]
32 [number of actors]
1 [number of dependent network variables]
0 [number of dependent actor variables]
1 [number of files with constant actor covariates]
0 [number of exogenous changing actor covariates]
0 [number of constant dyadic covariates]
0 [number of exogenous changing dyadic covariates]
0 [indicator for file with composition change directives]

@2[network files in temporal order; names follow:]
vrnd32t2.dat
1 2 3 [code for tie]
6 9 [code for missing]
vrnd32t4.dat
1 2 3 [code for tie]
6 9 [code for missing]
friendship

@4[files with constant actor covariates:]
vars.dat
3 [number of covariates in this file; names follow:]
99 [code for missing]
gender
99 [code for missing]
program
99 [code for missing]
smoke
```

The basic data input is carried out by executing SIENA01.EXE. This programs reads the basic information file. Some preliminary output is given in the files *pname*.out and *pname*.log.

16.2 Definition files

The program writes and reads for internal use the following two definition files:

- *pname*.MO   defines model specification and options for model estimation,
- *pname*.SI   defines statistics and number of runs for simulation.

These definition files are read in a format where certain lines are skipped entirely and other lines are skipped after reading a certain number. These skipped parts are between square brackets [...]. Their purpose is to give information to the human reader about the meaning of the lines. Note, however, that SIENA does not check for the brackets, but skips information on the basis of line numbers and reading numerical information.

The three files *pname*.IN, *pname*.MO and *pname*.SI must be compatible (as they contain some overlapping information) for successfully running SIENA.

16.2.1 Model specification through the MO file

To change the model specification outside the StOCNET shell, you can change the *pname*.MO file by an ASCII text editor. In this way you can used advanced SIENA options which are not yet available through StOCNET; whether such options exist, will depend on the versions of SIENA and StOCNET.

When looking at the *pname*.MO file, you can see that this file by and large contains the same information as the screen opening up in StOCNET when clicking the Model specification button. More precisely, the *pname*.MO file consists of several sections, marked by @-symbols, as follows:

- Section @1 contains general information about the project, such as data format, numbers and names of variables. The information given in this section must be compatible to the information provided in the file *pname*.IN.
- Sections starting @2.x contain the model specification for dependent network variables (indexed by x). There are two subsections:
  - Subsections @2.x.1 contain the specification of the network rate function, which looks like this:

    ```
    @2.1.1 [rate function effects for dependent network variable ⟨#1⟩:]
    11 [number of these effects :]
    [each effect is given in two rows]
    [first row contains label of the effect,]
    [second row contains flags for inclusion, fixing and testing,]
    [the starting value and potential extra parameters for effect calculation.]
    basic network rate parameter
    0 0 0   5.851107 0
    outdegrees effect on rate
    0 0 0   0.000000 0
    [...]
    ```

    The section must contain first a row with the number of effects, further down for each of these effects two rows, one containing the effect name, the other containing a sequence of numbers. These have the following meaning:

    * a 0/1 entry denoting whether the effect is included (1) or excluded (0);
    * a 0/1 entry denoting whether the effect is fixed (1) or not fixed (0) during the estimation process;
    * a 0/1 entry indicating whether a fixed effect shall be included (1) or excluded (0) in goodness-of-fit calculations (see Section 7);
    * the starting value of the parameter for the estimation procedure;
    * an effect-specific parameter (for modeling the constants c in the mathematical definition of the effects, see Section 13.1.1.
  - Subsections @2.x.2 contain the specification of the network decision rule (including the utility endowment functions), and which has the following shape:

    ```
    @2.1.2 [objective function effects for dependent network variable ⟨#1⟩:]
    46 [number of such effects :]
    [first row contains label of the effect,]
    [next three rows: functions in which effect can be included:]
    [row 1: utility function, row 2: endowment function, row 3: reinforcement function,]
    [columns correspond to: (a) inclusion, (b) random effects, (c) fixing, (d) testing,]
    [(e) starting value, (f) potential extra parameters for effect calculation.]
    outdegree (density)
    1 0 0 0  -1.143698 0
    0 0 0 0   0.000000 0
    0 0 0 0   0.000000 0
    ```

```
reciprocity
1 0 0 0   1.467572 0
0 0 0 0   0.000000 0
0 0 0 0   0.000000 0
[...]
```

The section must contain first a row with the number of effects, further down for each of these effects four rows, one containing the effect name, the other containing sequences of numbers. These have the following meaning:

* a 0/1 entry denoting whether the effect is included;
* a 0/1 entry denoting whether a corresponding random effect effect is included;
* a 0/1 entry denoting whether the effect is fixed;
* a 0/1 entry indicating whether a fixed effect is included in goodness-of-fit calculations;
* the starting value of the parameter for the estimation procedure;
* an effect-specific parameter.

Of the three rows that follow this pattern, the first row corresponds to the utility function, the second row corresponds to the endowment function, and the third row is reserved for future model extensions that allow the modeling of adaptive learning behavior.

In the current version of SIENA, but one dependent network variable can be analyzed at a time, so there will only be sections @2.1.y.

- Sections starting @3.x contain the model specification for dependent action variables (again indexed by x). as for the network variables, there are two subsections:
  - Subsections @3.x.1 contain the specification of the behavioral rate function,
  - Subsections @3.x.2 contain the specification of the behavioral decision rule (including the utility endowment functions).

  These subsections have the same format as the corresponding subsections for the network evolution model.

- The final Section @4 contains specifications of various estimation options. Most of these are accessible in StOCNET, e.g., through the model options mentioned in Section 10. The consecutive options are the following:

  1. Estimation method:
     - code 0 for unconditional estimation;
     - code 1 or code 21 for estimation conditional on observed changes in the network;
     - codes codes $21+k$ for estimation conditional on observed changes on the dependent action variable k.

     For exponential random graphs, another available option is to include incidental vertex parameters:
     - code 10 for unconditional estimation with incidental vertex parameters,
     - code 11 for conditional estimation with incidental vertex parameters.

2. A code for the type of model.
   For longitudinal data this is the Model Code described in the section on model options. For exponential random graph models, this code defines the steps used in the one-observation case for simulating a random (di)graph (see also the description above):
   - code 1: Gibbs steps for single relations;
   - code 2: Gibbs steps for dyads;
   - code 3: Gibbs steps for triplets;
   - code 4: Metropolis Hastings steps for single relations;
   - code 5: Metropolis Hastings steps for dyads in which symmetric dyads remain symmetric and asymmetric dyads remain asymmetric (appropriate for undirected graphs and for tournaments, i.e., antisymmetric graphs);
   - code 6: Metropolis Hastings steps conditional on indegrees and outdegrees.

   To each of these values, the number 10 may be added (so the values become 11–16). In that case a *continuous* chain is used: i.e., the last generated graph is used as the intial value in the MCMC sequence for simulating the next graph. Otherwise (i.e., for the values 1–6), the initial value is an independently generated random graph. For practical purposes, one should always use a continuous chain.
3. The number of subphases in phase 2 of the estimation algorithm (advice: 4).
4. The number of phase 3 iterations for the estimation algorithm (advice: 500 for longitudinal data, 1000 for modeling one-observation data by an exponential random graph).
5. A number r proportional to the number of steps used for generating one graph in the one-observation case. The number of steps is $r\,n^2/2d(1-d)$ where n is the number of actors and d is the observed density of the graph; if the observed density is less than .05 or more than .95, the value $d = .05$ is used.
6. The initial value of the gain parameter in the estimation algorithm (advice: 0.1).
7. An indicator for the initial value used in the estimation algorithm:
   - code 0: current value as specified in the preceding sections,
   - code 1: standardized starting value, meaning that a good starting value is chosen for the outdegree effect and in the one-observation (exponential random graph) case also for the reciprocity effect. For still other configurations, also starting values for the rate parameters and the tendency effect for dependent action variables are internally determined. All other effects then have a 0.0 starting value.
8. An indicator allowing for tests of temporal (between-period) homogeneity of parameters:
   - code 0: no test,
   - code $m > 0$: test for period m
9. An indicator allowing for tests of actor homogeneity of parameters:
   - code 0: no test,
   - code 1: test of the outdegree (density) effect,
   - code 2: test of the reciprocity effect.
10. A space-separated list giving the row numbers of the actors for such an actor homogeneity test.
11. A value determining the use of random numbers during the estimation process:
    - code 0: randomize seed,
    - code $c \neq 0$: c is used as random seed.

If you change anything in the *pname*.MO file, you must run SIENA04.EXE to let SIENA check (and if necessary: censor or re-define) the model specification.

16.2.2 Specification of simulations through the SI file

For running simulations, SIENA needs to be told which statistics it should simulate. This is done by manipulating the *pname*.SI file by an ASCII text editor before running the simulations. This file consists of two sections, again marked by @-symbols, as follows:

- Section @1 contains the list of possible statistics that can be simulated. It looks as follows:

```
@1 [statistics that can be generated]
60 [number of these statistics :]
[each statistic is given in two rows]
[first row contains label of the statistic,]
[second row contains flag for inclusion]
Amount of change
1
Number of ties
1
Number of reciprocated ties
0
[...]
```

  As can be seen, for each effect, there are two rows:

  - a row containing the statistic's name;
  - a row contining an indicator whether the statistic shall be simulated.

- Section @2 contains options for simulation. Currently, there is but one option:

  - the default number of simulations for straight simulation (advice: 1000).

The file *pname*.SI by and large contains the same information as the screen opening up in StOCNET when clicking the Statistics specification button (which is possible as soon as Simulation is selected as the Run model).

16.3 Data files

After the initial project definition the original data files are not used any more, but the project data files are used. These are the following.

• *pname*.d01	Network data file time 1.
• *pname*.d02, etc.	Network data file time 2, etc.
• *pname*.m01, etc.	Network missings file time 1, etc.
• *pname*.s01, etc.	Fixed part of network structure period 1, etc.
• *pname*.dav	Data file constant actor-dependent variables (centered).
• *pname*.mav	Missings file constant actor-dependent variables (centered).
• *pname*.dac	Data file with changing actor-dependent variables.
• *pname*.mac	Missings file changing actor-dependent variables (centered).
• *pname*.z01, etc.	Data files dyadic covariates.
• *pname*.n01, etc.	Missings files dyadic covariates.
• *pname*.dex	Data file times of composition change.

The user does not need to care about these data files (but should not delete them either).

16.4 Output files

The output for the user goes to *pname*.out. Extra output is written to *pname*.log, which in the first place is a log file of what the program did. The estimation procedure also writes a file *pname*.chk, containing a more detailed report of the estimation algorithm. The latter two files are for diagnostic purposes only. The *pname*.chk file is overwritten with each new estimation procedure.

17 Units and executable files

The basic computational parts of SIENA are contained in the following units.
First, there are four basic units.

1. S_DAT contains the basic data structures.
2. S_BASE contains the basic model definition.
3. S_SIM contains the procedure for straight simulation.
4. S_EST contains the procedure for estimation.
5. S_TEST contains procedures for goodness-of-fit tests.
6. S_DESC contains procedures for data description.

Then there is an intermediate unit.

7. S_START contains the procedure ReadWriteData to start a project by reading the *pname*.IN file and the initial data files, and writing the internally used files. It uses only S_DAT. Procedure ReadWriteData from S_START must be followed always by procedure BeforeFirstModelDefinition from S_BASE.

Further there are three units containing specific kinds of utilities.

8. SLIB is a library of various computational and input/output utilities.
9. RANGEN is a library for generation of random variables. It uses the URNS suite for random numbers generation.
10. EIGHT is a unit for storing the data. Its name was chosen for historical reasons (a byte was used to store eight booleans).

17.1 Executable files

The basic data input is carried out by executing SIENA01.EXE. This program executes ReadWriteData and BeforeFirstModelDefinition, thereby reading the basic information file.

Data description is carried out by SIENA02.EXE which executes Describe.

In the StOCNET operation the model specification is carried out by StOCNET changing the .MO file and then running SIENA04.EXE.

SIENA04.EXE is only used for checking admissibility of a model specification, and for keeping consistency between the different session files. It reads and updates the .MO file, and also writes the corresponding .SI file. If you change *pname*.MO by an text editor outside of StOCNET, it is advisable to run SIENA04.EXE before proceeding.

The simulation is carried out by SIENA05.EXE which executes Simulate.

The estimation is carried out by executing SIENA07.EXE which executes Estimate.
The multilevel combination of estimation runs for several data sets according to the same basic model specification is done by SIENA08.EXE which also executes the procedure Estimate (although taking it from S_EST8). The essential difference between SIENA08.EXE and SIENA07.EXE is contained in the difference between units Siena_8 and Siena_7.

18 Parameters and effects

In the source code there are two kinds of parameters: alpha and theta. The alpha parameters are used in the stochastic model, and each alpha parameter corresponds to one effect, independently of whether this effect is included in the current model specification. Their values are stored in the *pname*.MO file, which also indicates (by 0-1 codes) whether these variables are included in the model, and whether they are fixed at their current value in the estimation process. The theta parameters are the statistical parameters that correspond to the effects in the current model specification.

The distinction between these two types of parameters in principle also allows linear (or other) restrictions between the alphas. In the present version, the possibility of such restrictions is not implemented, but the distinction between alpha and theta allows to elaborate this possibility in a later version.

The effects for the evolution model are distinguished between effects for the network dynamics and effects for the behavior dynamics. The effects are defined in several procedures, which of course must correspond. To each effect corresponds one statistic used for estimating the parameter for this effect. This is spelled out in Section 18.2.
In unit S_DAT, the numbers of the various types of effects are defined:

1. NetworkEffects_f, the number of effects in the utility function for the network dynamics;
2. NetworkEffects_g, the number of effects in the endowment function for the network dynamics;
3. NetworkEffects_l, the number of effects in the rate function for the network dynamics;
4. ActionEffects_f, the number of effects in the utility function for the behavior dynamics;
5. ActionEffects_g, the number of effects in the endowment function for the behavior dynamics;
6. ActionEffects_l, the number of effects in the rate function for the behavior dynamics;
7. NetworkFunctions, the number of statistics that can be calculated and which can be used if there are no dependent behavior variables (this number must be at least NetworkEffects_f + NetworkEffects_g + NetworkEffects_l; it is allowed to be larger – then there are some statistics that are superfluous for parameter estimation by the method of moments).
8. ActionFunctions, the number of statistics that can be used additionally to the statistics mentioned before, in case that there are dependent behavior variables.

In unit S_BASE, the definitions of the effects and statistics are given.

- A. For the rate function:
  1. Procedure DefineModel_lnames defines the names of the effects and the index numbers of the corresponding statistics;
  2. function NetworkLambda defines the network change rate function;
  3. procedure Transform_l calculates some variables for more efficient calculations in NetworkLambda;
  4. for Model Type 2, the rate function for network change also depends on the functions ξ and ν;
  5. function ActionLambda defines the behavior change rate function.
- B. For the utility function:

1. Procedure DefineModel_fnames defines the names of the effects and the index numbers of the corresponding statistics;
2. function Contr_fn gives for each effect for the network dynamics the contribution to the difference between the value of the utility function after a change will be made, and its current value.
3. function Contr_fa gives for each effect for the behavior dynamics the contribution to the difference between the value of the utility function after a change will be made, and its current value.

- C. For the endowment function:
  1. Procedure DefineModel_gnames defines the names of the effects and the index numbers of the corresponding statistics;
  2. function Contr_gn gives the contribution of each effect to the endowment function for the network dynamics;
  3. function Contr_ga gives the contribution of each effect to the endowment function for the behavior dynamics.
- D. For the statistics:
  1. Procedure DefineFunctionnames defines the names of the statistics;
  2. procedure Statistics calculates the statistics.

18.1 Starting to look at the source code

If you wish to start with understanding the structure of the source code, it may be helpful to take the following tour of some essential ingredients.

1. Unit EIGHT contains the fundamental data structures.

   (i) Functions starting with y represent dependent variables.

   (a) yn is the adjacency matrix of the dependent network variable (current value):
   yn(i,j) is the tie variable from i to j;

   (b) ya is the vector of dependent individual variables (current value):
   ya(i,h) is the h'th individual variable for actor i;

   (c) ynm gives the adjacency matrices of the dependent network variable (all values):
   ynm(i,j,m) is the tie variable from i to j at observation moment m.

   (d) mis is the indicator matrix for missing values:
   mis(i,j,m) = 0 if ynm(i,j,m) is an observed value and 1 if it is missing.

   Note that yn and ya change dynamically during the simulation process; the variables containing the data from which ynm and mis are calculated, are read in the input phase and then are left unaltered.
   For the sake of flexibility, the variables in the list above are implemented as functions, not as arrays.

   (ii) Variables starting with z represent individual variables.

   (a) Array z contains the constant actor variables; the number of such variables is nz.

   (b) Array zz contains the dyadic variables; their number is nzz.

   (c) Array zc contains the changing actor variables; their number is nzc.
   (d) The number of dependent actor variables is nza; these variables are the first nza among the nzc changing actor variables.

2. Unit S_DAT contains the defining ingredients for the network models. The intended demarcation between S_DAT and S_BASE, is that the former contains the model definition and the latter the operation of the model dynamics. This is not realized completely because of various conflicting constraints.

3. Unit S_BASE contains the simulation procedures.
   The main procedures in this unit are the following.
   (a) Function FRAN generates the required statistics and is called by procedure Simulate in S_SIM and by Estimate in S_EST.
   The procedure FRAN first simulates the network and behavior by the procedure Runepoch and then calculates statistics by the procedure Statistics; the values of these statistics are output arguments of FRAN.
   (b) Procedure Statistics calculates the statistics from the generated network (or adjacency matrix) and behavior variables, and is called by FRAN.
   (c) Procedure Runepoch generates a random network and action variables for given parameter values and a given initial situation by simulating the actor-oriented evolution model for one period between two observations. This procedure is called by procedure FRAN. If there are M observations ($M \geq 2$), Runepoch is called $M - 1$ times.
   (d) Procedure Runstep makes one stochastic step according to the actor-oriented evolution model, i.e., it either changes one entry (i, j) of the adjacency matrix to be changed, or it changes the value of one action variable for one . The time variable time is also increased by an amount depending stochastically on the rate functions. This procedure is called by procedure Runepoch.
   (e) Procedure ChangeTie is called at the end of procedure RunStep if a change in the network is made, and carries out the required change of the adjacency matrix and the associated updates of various statistics.
   (f) Procedure ChangeBehavior is called at the end of procedure RunStep if a change in behavior is made, and carries out the required change of the dependent action variable and the associated updates of various statistics.
   (g) Function NetworkLambda, which is the rate function for the network changes made by each actor, and is used in procedure Runstep.
   For Model Type 2, it uses functions ξ and ν.
   (h) Function ActionLambda, which is the rate function for the behavior changes made by each actor, and is used in procedure Runstep.
   (i) Function contr_fn, which defines the contribution $s^X_{ih}(x)$ (see Section 13.1.1) of each given effect h to the utility function for the network, and is used in procedure Runstep.
   (j) Function contr_gn, which defines the contribution of each given effect to the endowment function for the network, and is also used in procedure Runstep.
   (k) Function contr_fa, which defines the contribution $s^Z_{ih}(x)$ of each given effect h to the utility function for the behavior, and is used in procedure Runstep.
   (l) Function contr_gn, which defines the contribution of each given effect to the endowment function for the behavior, and is used in procedure Runstep.

4. In unit S_EST, the Robbins-Monro algorithm is contained in the procedure POLRUP (for Polyak-Ruppert, see Snijders, 2001). When parameters are to be tested by Neyman-Rao tests, S_EST calls the procedure TestStatistic in the unit S_TEST.
5. Unit S_SIM is used for simulations, and calls the function FRAN in S_BASE.
6. The unit S_ERGM_EST and the file S_ERGM.PAS which is an include file used in S_BASE contain procedures used only for the ERGM (1 observation) case.

18.2 Procedures containing definitions of the effects

If new effects are added to the model, these additions must be made to each of the following procedures (in a coherent way, of course).

- For each kind of effect:
  1. Procedure DefineFunctionNames in unit S_BASE, which contains the names of all the statistics calculated from each simulation;
  2. Function NetworkFunctions or ActionFunctions in unit S_DAT, giving the total number of statistics calculated; NetworkFunctions is the number of statistics calculated if there are no dependent behavior variables, and ActionFunctions is the number of statistics that can be used additionally to these, in case that there are one or more dependent behavior variables.
  3. Procedure Statistics in unit S_BASE, which calculates these statistics.
- For an effect in the rate function for network change:
  1. Function NetworkEffects_l in unit S_DAT, the total number of possible effects in the network change rate function;
  2. Procedure DefineModel_lnames in unit S_BASE, which contains the names of all effects in the network change rate function, and the numbers of the corresponding statistics (used for fitting the parameters);
  3. Function NetworkLambda in unit S_BASE, the network change rate function itself.
- For an effect in the utility function for the network:
  1. Function NetworkEffects_f in unit S_DAT, the total number of possible effects in the network utility function;
  2. Procedure DefineModel_fnames in unit S_BASE, which contains the names of all effects in the network utility function, and the numbers of the corresponding statistics;
  3. Function Contr_fn in unit S_BASE, the contribution of a given effect to the difference between the value of the utility function after the change will be made, and its current value.
- For an effect in the endowment function for the network:
  1. Function NetworkEffects_g in unit S_DAT, the total number of possible effects in the network endowment function;
  2. Procedure DefineModel_gnames in unit S_BASE, which contains the names of all effects in the network endowment function, and the numbers of the corresponding statistics;

3. Function Contr_gn in unit S_BASE, the contribution of a given effect to the endowment function.

- For effects in the rate function for behavior change, the analogous procedures have to be changed as those for the rate function for network change:
  function ActionEffects_l in unit S_DAT, procedure DefineModel_lnames in unit S_BASE, and function ActionLambda in unit S_BASE.
- For effects in the utility function for behavior change, the analogous procedures have to be changed as those for the utility function for network change:
  function ActionEffects_f in unit S_DAT, procedure DefineModel_fnames in unit S_BASE, and function Contr_fa in unit S_BASE.
- For effects in the endowment function for behavior change, the analogous procedures have to be changed as those for the endowment function for network change:
  function ActionEffects_g in unit S_DAT, procedure DefineModel_gnames in unit S_BASE, and function Contr_ga in unit S_BASE.

The functions Contr_fn, Contr_gna, Contr_fa, Contr_ga, NetworkLambda, and ActionLambda are evaluated very frequently by the algorithm. Therefore these have been written so that very few calculations are needed to evaluate them. Such calculations for a large part are replaced by updating and storing the basic numerical information needed to compute them. These updates are contained in the procedure ChangeTie in unit S_BASE, and the initialisation is contained in the procedure Initialise_Running_Statistics.

19 Statistical Monte Carlo Studies

According to Sir Ronald A. Fisher, there are three main statistical problems, model specification, model estimation, and problems of distribution. The last one concerns the distribution of statistics, such as the distribution of parameter estimates around the true (data-generating) parameter value or the distribution of test statistics, and can be studied by SIENA as follows.

Open the unit Siena_7 and go to the procedure TEstForm.FormActivate. The first statement in the procedure is `simulate := false`. Set the global variable `simulate` to true. SIENA will then simulate data sets according to the probability model specified in the *pname*.MO file.

Then, manipulate the global constant `sequences` declared in unit Siena_7 by setting it to some positive integer value k (the default is 1). The constant `sequences` gives the number of runs (sequences).

The result is that running `Siena` will generate k data sets according to the probability model specified *pname*.MO file. From each data set, the parameters are estimated and test statistics are evaluated.

Some Matlab source files are (by default) generated by SIENA. The source code, when interpreted by Matlab, produces histograms of some statistics, in particular histograms of the parameter estimates and the test statistics.

It should be noted that SIENA generates networks with desired properties, but (by default) no covariates. If covariates are desired, suitable code must be added at the beginning of the procedure SimulateData in the unit S_EST. Please note that both internal and external storage (see Section 16.3) of generated covariates is required. Internal storage is difficult unless one knows SIENA -it is advisable to contact the authors in such cases.

20 Constants

The program contains the following constants. Trying to use a basic information file that implies a data set going beyond these constants leads to an error message in the output file and stops the further operation of SIENA.

name	meaning	in unit
nmax	maximum number n of actors	EIGHT
nrg	maximum array size for random number generation	RANGEN
pmax	maximum number p of included effects	SLIB
ccmax	maximum number of possible statistics	SLIB
nzmax	maximum number nz of individual variables	EIGHT
nzzmax	maximum number nzz of dyadic covariates	EIGHT

The constant individual covariates and the changing individual variables both have *nzmax* as their maximum.

Reasonable values for these constants are the following:
$nmax = 500$;
$nrg = nmax$;
$pmax = 60$;
$ccmax = 500$; the maximum number of statistics is given by MaxFunctions which can be looked up in unit S_Dat (in version 2.1 of SIENA, e.g., for two dependent action variables and M observations, the number of statistics is $53 + 3 \times M + 17 \times (nz + nzc) + 5 \times nzz$);
$nzmax = 10$;
$nzzmax = 10$.

The number M of observations may not be higher than 99. Since the number of observations is dealt with by a dynamic array, this is not reflected by some constant. The only reason for the upper bound of 99 is that the index number of the observation is used in the internal data file extension names and may not have more than two digits. But 99 seems quite a high upper bound for practical data sets.

21 References

Albert, A., and J.A. Anderson. 1984. On the existence of the maximum likelihood estimates in logistic regression models. *Biometrika*, **71**, 1 – 10.

Boer, P., Huisman, M., Snijders, T.A.B., and E.P.H. Zeggelink. 2003. *StOCNET: An open software system for the advanced statistical analysis of social networks.* Version 1.4. Groningen: *Pro*GAMMA/ICS. http://stat.gamma.rug.nl/stocnet/.

Frank, O. 1991. Statistical analysis of change in networks. *Statistica Neerlandica*, **45**, 283–293.

Frank, O., and D. Strauss. 1986. Markov graphs. *Journal of the American Statistical Association*, **81**, 832 – 842.

Geyer, C.J., and E.A. Thompson. 1992. Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society*, ser. B, **54**, 657 – 699.

Handcock, Mark S. 2002. "Statistical Models for Social Networks: Inference and Degeneracy." Pp. 229 – 240 in *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, edited by Ronald Breiger, Kathleen Carley, and Philippa E. Pattison. National Research Council of the National Academies. Washington, DC: The National Academies Press.

Huisman, M.E., and T.A.B. Snijders. 2003. Statistical analysis of longitudinal network data with changing composition. *Sociological Methods & Research*, **32**, 253 – 287.

Pahor, M. 2003. *Causes and Consequences of Companies' Activity in Ownership Network.* Ph.D. Thesis, University of Ljubljana, Slovenia.

Schweinberger, M. 2005. *Statistical Modeling of Network Dynamics Given Panel Data: Goodness-of-fit Tests.* Submitted for publication.

Snijders, T.A.B. 2001. The statistical evaluation of social network dynamics. Pp. 361-395 in *Sociological Methodology – 2001*, edited by M.E. Sobel and M.P. Becker. Boston and London: Basil Blackwell.

Snijders, T.A.B. 2002. Markov Chain Monte Carlo Estimation of Exponential Random Graph Models. *Journal of Social Structure*, Vol. 3 (2002), No. 2.
Available from http://www2.heinz.cmu.edu/project/INSNA/joss/index1.html.

Snijders, T.A.B. 2003. Accounting for Degree Distributions in Empirical Analysis of Network Dynamics. *Proceedings of the National Academy of Sciences USA*, to be published.
Available from http://stat.gamma.rug.nl/snijders/siena.html.

Snijders, T.A.B. 2004. Explained Variation in Dynamic Network Models. *Mathématiques, Informatique et Sciences Humaines / Mathematics and Social Sciences*, 168(4).

Snijders, T.A.B. 2005. Models for Longitudinal Network Data. Chapter 11 in P. Carrington, J. Scott, and S. Wasserman (Eds.), *Models and methods in social network analysis.* New York: Cambridge University Press.

Snijders, T.A.B., P.E. Pattison, G.L. Robins, and M.S. Handcock. 2004. *New specifications for exponential random graph models.* Submitted for publication.

Snijders, T.A.B., and M.A.J. Van Duijn. 1997. Simulation for statistical inference in dynamic network models. Pp. 493 – 512 in *Simulating Social Phenomena*, edited by R. Conte, R. Hegselmann, and P. Terna. Berlin: Springer.

Snijders, T.A.B., and M.A.J. Van Duijn. 2002. Conditional maximum likelihood estimation under various specifications of exponential random graph models.
Pp. 117–134 in Jan Hagberg (ed.), *Contributions to Social Network Analysis, Information Theory, and Other Topics in Statistics; A Festschrift in honour of Ove Frank.* University of Stockholm: Department of Statistics.

Steglich, Ch., T.A.B. Snijders, and M. Pearson, M. 2004. *Dynamic Networks and Behavior: Separating Selection from Influence.* (Submitted.)

Van de Bunt, G.G. 1999. *Friends by choice. An actor-oriented statistical network model for friendship networks through time.* Amsterdam: Thesis Publishers.

Van de Bunt, G.G., M.A.J. van Duijn, and T.A.B. Snijders. 1999. Friendship networks through time: An actor-oriented statistical network model. *Computational and Mathematical Organization Theory*, **5**, 167 – 192.

van Duijn, M.A.J., E.P.H. Zeggelink, M. Huisman, F.N. Stokman, and F.W. Wasseur. 2003. Evolution of Sociology Freshmen into a Friendship Network. *Journal of Mathematical Sociology* 27, 153–191.

Wasserman, S., and P. Pattison. 1996. Logit models and logistic regression for social networks: I. An introduction to Markov graphs and p^*. *Psychometrika*, **61**, 401 – 425.