Simstats0c/FRAN

Ruth Ripley

2013

February 18, 2021

1. Introduction

The R function *simstats0c* (so named during development when I had other versions which did not call C!) is the function which controls the simulations. It is used as an argument to the function *sienaModelCreate* and stored as the element named FRAN of the model object. It has three (or 3+) modes: initial, to set up the C++ data structure, terminal (to tidy up) and ordinary (to do one complete simulation). (There is an extra mode for using multiple processors, which does some of the work of an initial call.) It uses C++ functions for most of its work.

2. Outstanding work

1. When setting up the effects for behaviour variables, it is assumed that the one-mode network is non-symmetric. Should not!
   Note by Tom (February 2021): I'm not sure that this still is the case. It should be checked.
2. Allow data values indicating structurally fixed values to be a parameter.
3. Validate things

3. Model specification

3.1 The data as input to *simstats0c*

A *siena* (or *sienaGroup*) object and a *sienaEffects* (or *sienaGroupEffects*) object.

3.2 The model as it reaches C++

Virtual data A set of dependent variables, actor sets representing the nodes for the variables, and covariates of various types. Never actually exist, but the R routines must check that each data object is consistent with the virtual data. The virtual data contains details of inter-network constraints: *higher, atLeastOne, disjoint*

Data Each data object consists of several waves for each of the dependent variables, each of which uses some subset of each corresponding actor set, together with covariate objects of appropriate sizes.

Edgelists Networks and dyadic covariates are passed to C++ as valued edgelists.

Networks Networks have three edge lists: data, missing, structural values. Missing and structural values have been replaced in the data.

Behaviour variables Passed in as a pair of matrices, data and missing entries (as 1's). Missing values have been replaced in the data matrix, as have structural values(?)

Covariates Missing values have been replaced. Node set name(s) are attached to the object as attributes.

Joiners and leavers This is for actors joining or leaving the network. Data is reformatted to be, for each actor set, a list of two items. The first is a data frame with columns event type, period, actor and time. The second is a matrix of booleans, indicating for each actor whether active at the start of each period.

Uponly, downonly Uponly and downonly flags are attached to all dependent variables as attributes, one value for each pair of observations. They indicate that the dependent variable must not decrease, or must not increase, respectively.

symmetric flag for each one-Mode network dependent variable. Only true if true for all observations of the virtual network.

maxdegree one integer per dependent network, same value used for each instance of the virtual network. For valued graphs (not yet!), maxdegree will refer to the number of non-null outgoing ties per actor.

derivs required flag to suppress calculating the scores

from finite differences flag to indicate that this run is a 'repeat' which needs to reuse the random numbers of a previous run. Implemented by obtaining from R and storing the random number objects (`.Random.seed`) at the start of the simulation for each period in each non finite difference run and restoring the random number object into R in each finite difference run. The values are stored in the FRANstore function in between!

Effects A data frame containing identification information for each effect. In the setup phase the effects objects are created and pointers to objective function effects are stored on this data frame. In simulation calls, the pointers are used to access these effects to update the parameters. Rate effects are accessed differently.

A single effects object contains all the effects for a multi-group project: some of the basic rate effects need to be selected out when reading the data frame for particular data object.

The data frame also has columns which could be used for pairs of R functions (for user-defined effects) (one each for effect and statistic calculation, to be passed back to R, a simple way to implement user-defined effects where speed is not critical). This interface for user-defined effects is not currently implemented.

balmean A calculation for each network, at virtual data level .

similaritymean A calculation for each behavior variable or covariate, at virtual data level.

range For each behavior variable and covariate, at virtual data level

mean For each dyadic covariate, at virtual data level.

4. Initial call to simstats

- Convert data and pass to C++
- Select requested effects and pass to C++
- Pass Model options to C++
- Check consistency of networks with each other and with covariates. Need at least one dependent variable. (a check that data has been interpreted correctly!) (Not really done yet!)

5. The Loop Processing

```
Update the parameters for the effects
for each group do
  Create new EpochSimulation object
  for each period (omitting the final one) do
    if from Finite Differences then
      restore the random number seed
    else
      store the random number seed
    end if
    repeat
      Get tau (section 5.1)
      if the time is greater than the time to the next joiner or leaver in any
      network then
        change the composition of the networks (section 5.7)
        set the time so far to the time of the composition change event
        next iteration
      end if
```

```
            Choose dependent variable to alter with probabilities proportional to
            the total lambda in each dependent variable.
            For symmetric networks totlambda is defined diferently
            Choose actor proportional to lambda
            (For symmetric networks use lambda and sum of lambdas as usual,
            even if totlambda was calculated differently)
            **if** *deriv* **then**
              Accumulate rate scores Section 5.4.
            **end if**Choose change 5.2
            Update state
            If required, add scores to accumulators
          **until** time or change is reached, or decide it will not be reached
          Calculate statistics (section 5.6)
      **end for**
    **end for**
    **return** statistics, scores (if requested), total time (if conditional), simulated
    networks (if requested, only 1 at the moment!)
```

5.1 Get *tau*

```
for each dependent variable do {calculate totlambda}
  for each actor do {calculate lambda}
    if inactive $\equiv$ all links structurally fixed in this period or not yet joined the
    network then
      lambda = 0
    else
      lambda = product of basic rate for the period with other selected rate
      effects multiplied by relevant theta value
    end if
    Add to totlambda
  end for
  Add to sum of totlambda
  if symmetric and model type BFORCE, BAGREE, BJOINT then
    Replace totlambda by square of totlambda - sum of squared lambdas
    (This is $\sum_{i<>j} \lambda_i \lambda_j$ for active $i, j$.)
  end if
end for
Get a random exponential with sum of totlambda as parameter.
```

5.2 Choose Change

```
Define alters =$\{1 \ldots$ dim2$\}$
if not symmetric or model type is A then
  if bipartite then
    add $\{0\}$ to alters
  end if
  for $j \in$ alters do {calculate deltas and effects; this condition changed from
  = to ¿=, ts, 6-5-13}
```

```
    if out-degree of actor $>=$ maxdegree then
      change is only permissible if there is a link to this alter, to change its
      value
    else if link from actor to $j$ is structurally determined then
      delta $= \{\}$
    else {changes are permissible}
      if behaviour variable then
        delta $= \{+1, -1\}$ {may allow larger changes}
        if uponly  then
          delta $= \{+1\}$
          if equal to maximum then
            delta $= \{\}$
          end if
        else if downonly then
          delta $= \{-1\}$
          if value = minimum then
            delta $= \{\}$
          end if
        else if value = maximum then
          delta $= \{-1\}$
        else if value = minimum then
          delta $= \{+1\}$
        end if
      else {not behaviour variable}
        if j = 0 or j = actor then
          delta $= 0$
        else
          delta $= \{1 - 2\text{y}\}$ {for valued networks, this would alter}
        end if
      end if
    end if
    calculate p(j, delta) effect of each delta on the objective function (sum of
    effects multiplied by corresponding parameters)
  end for
  Choose change proportional to $\exp($p(j,delta)$)$
  if symmetric and model type is A and subtype is 2 (alias M1 in PA paper)
  and alter is not equal to actor then
    Check if alter agrees Section 5.3.2
    if alter does not agree then
      abort the change
    end if
  end if
else {symmetric type B (alias M2, D2, C in PA paper) }
  Choose alter with same probabilities as actor (repeat until not equal to actor)
  Calculate probabilities of acceptance. Section5.3.3
  if not accepted change then
    abort the change
  end if
end if
if deriv then
  Calculate score functions. Section 5.4.
end if
```

5.3 Symmetric network models

There are various types in two groups: Type A or type B. In type A the change is selected in the normal way, but for subtype 2 the alter gets a chance to say no. In type B we select the alter using the rate probabilities and then simply consider that one link.
In the PA paper, Type A models have new names:
A subtype 1 = forcing = D1,
A subtype 2 = unilateral & reciprocal confirmation = M1.
Type B models are called:
BAGREE = B subtype 1 = pairwise conjunctive = M2;
BFORCE = B subtype 2 = pairwise disjunctive = D2;
BJOINT = B subtype 3 = pairwise compensatory = C (same as tie-based).

Section 5.2 referred to the following additional steps in the case of a symmetric network.

5.3.1 Decision probabilities

Depending on options and outcomes, we may in the sequel need probabilities p_{ij} or p as defined now. These probabilities are defined for each relevant link from the perspective of the appropriate actor, by

```
  Calculate the change contributions for each effect for this link from $i$ to $j$,
  from $i$'s perspective.
  Calculate $y_{i,j}$, the unnormalised probability by multiplying by the parameters
  and summing.
  if link between $i$ and $j$ exists then
    change the sign of the unnormalised probability $y_{i,j}$
  end if
  if subtype not 3 then
    Calculate
      $$p_{i,j} = \frac{\exp(y_{i,j})}{(1+\exp(y_{i,j}))}$$
  else
    Calculate
      $$p = \frac{\exp(y_{i,j}+y_{j,i})}{(1+\exp(y_{i,j}+y_{j,i}))}$$
  end if
```

5.3.2 Model Type A subtype 2

```
**if** the proposal implies the creation of a tie **then**
  Calculate the probability $p_{alter,actor}$ as defined above.
  **if** a random number is less than $p_{alter,actor}$ **then**
    alter agrees
  **else**
    alter does not agree
  **end if**
**end if**
```

5.3.3 Model Type B

```
if BFORCE then
  calculate $p_{actor,alter}$ as described above
else if BAGREE then
  calculate $p_{actor,alter}$ and $p_{alter,actor}$ as described above
else if BJOINT then
  calculate $p$ as described above
end if
if BFORCE then {actor forces}
  accept if $p_{actor,alter}$ is greater than a random number
else if BAGREE then {both must agree for existence}
  if link between $i$ and $j$ exists then
    accept if $p_{actor,alter} + p_{alter,actor} - p_{actor,alter} * p_{alter,actor}$ is greater than a
    random number.
  else
    accept if product of probabilities is greater than a random number
  end if
else {BJOINT, joint decision}
  accept if p is greater than a random number
end if
```

5.4 Calculation of scores

For symmetric networks other than Model Type A, see section 5.5. For symmetric networks with Model type A and subtype 2, the rate scores are the same as in this section, but the non-rate scores are defined in section 5.5.

In Siena3, when calculating scores, only steps which are not aborted because of composition change are included in the scores. (This is a bug in Siena3. We include them unless we are parallel running.)

Let z_{rk} be the observed value of the kth statistic in the rth simulation
r_m the total rate for this dependent variable at step m of a simulation
r_M the total rate for this dependent variable at the generation of the final time interval in the unconditional case
t_m the *tau* of the mth step (maybe a composition change)
s_{ikm} be the value of the kth statistic for the ith actor at the mth step

r_{im} be the total rate for actor i at the mth step
$s_{i,j\delta k}$ be the change in the kth statistic corresponding to choice of i as actor and j as alter with change *delta*
$p_{i,j\delta}$ the probability of selecting the change *delta* with actor i and alter j
c_{rk} the score corresponding to θ_k in the rth simulation
i the actor, j the alter at the mth step of the total M. t_c is the time without an event due to composition change (Does not matter whether z_{rk} is statistic or deviation from observed, provided the same in each term!).

for a basic rate parameter, (λ), for a dependent variable,

$$c_{rk} = \sum_{m=1}^{M} \left(\frac{\delta_{pq}}{\lambda_m} - r_m t_m/\lambda_m \right) - \delta_{bc} r_M \left(1 - \sum_{m=1}^{M} t_m - \sum_c t_c \right) /\lambda_m$$

for other rate parameter

$$c_{rk} = \sum_{m=1}^{M} \left[\delta_{pq} s_{ikm} - \sum_{d=1}^{n} s_{dkm} r_{dm} t_m \right] - \delta_{bc} \sum_{d=1}^{n} s_{dkM} r_{dM} \left(1 - \sum_{m=1}^{M} t_m \right)$$

for other parameters

$$c_{rk} = \sum_{m=1}^{M} \left[s_{i,j\delta k} - \sum_{a,\textit{delta}} s_{i,a\delta k} p_{i,a\delta} \right]$$

with derivative matrix

$$D_{jk} = \frac{1}{R} \sum_{r=1}^{R} z_{rj} c_{rk} - \frac{1}{R^2} \sum_r z_{rj} \sum_r c_{rk}$$

where δ_{pq} is 1 if the rate parameter is selected and 0 otherwise (0 for all dependent variables if there is composition change), δ_{bc} is 1 if the simulation is unconditional and 0 otherwise, and R is the number of simulations performed (considering each period as a separate simulation).

Addendum: for maximum likelihood we need derivatives of scores too. Using similar notation, with $v_{rk_1k_2}$ the derivative of the score corresponding to$(\theta_{k_1}, \theta_{k_2})$ in the r-th chain, this is

for a basic rate parameter, (λ), for a dependent variable, not the total here, just the per actor rate

$$v_{rkk} = \sum_{m=1}^{M} \text{nbr of non-structurally determined links in the chain}/\lambda_m^2$$

for other parameters

$$v_{rk_1k_2} = \sum_{m=1}^{M} \left[\left(\sum_{a,\textsf{delta}_a} s_{i,a\delta k_a} p_{i,a\delta} \right) \left(\sum_{b,\textsf{delta}_b} s_{i,b\delta k_b} p_{i,a\delta} \right) - \sum_{a,\textsf{delta}} s_{i,a\delta k_1} p_{i,a\delta} s_{i,a\delta k_2} \right]$$

with derivative matrix

$$D_{jk} = \frac{1}{R} \sum_{r=1}^{R} v_{rjk}$$

Note that non-basic rate effects are not allowed in the maximum likelihood case and basic rate effects are uncorrelated with all other effects.

5.5 Symmetric network scores

5.5.1 Model type A, subtype 2 (AAGREE)

Here we have added a term to the log likelihood, accept or reject with probability p_{ji}. The scores for the non rate effects will be

$$c_{rk} = \sum_{m=1}^{M} \left[s_{i,j\delta k} - \left(\sum_{a,\textsf{delta}} s_{i,a\delta k} p_{i,a\delta} \right) + p_{ji}^{1-t}(1-p_{ji})^t(-1)^{1-t} s_{j,i\delta k} \right]$$

Where t is 1 if agree to change, 0 otherwise.

5.5.2 Model type B

Here we do not do the normal choice. My calculations suggest that for the basic rates it should be twice the normal scores using total rate and basic rate as usual. (Within variable).

The total rate for each variable is

$$r_m = \sum_{i<>j} r_{im} r_{jm} \text{ for active } i, j$$

So the rate scores will be,

for a basic rate parameter, (λ), for a dependent variable,

$$c_{rk} = \sum_{m=1}^{M} \left(\frac{2\delta_{pq}}{\lambda_m} - 2r_m t_m/\lambda_m \right) - 2\delta_{bc} r_M \left(1 - \sum_{m=1}^{M} t_m - \sum_{c} t_c \right) /\lambda_m$$

for other rate parameter, with j the alter,

$$c_{rk} = \sum_{m=1}^{M} \left[\delta_{pq}(s_{ikm} + s_{jkm}) - \sum_{i<>j} (s_{ikm} + s_{jkm}) r_{im} r_{jm} t_m \right]$$
$$- \delta_{bc} \sum_{i<>j} (s_{ikM} + s_{jkM}) r_{iM} r_{jM} \left(1 - \sum_{m=1}^{M} t_m \right)$$

While calculating rates we store

$$2 \left[r_m \sum_i s_{ikm} r_{im} - \sum_i s_{ikm} r_{im}^2 \right]$$

because

$$\sum_{i<>j} (s_{ikm} + s_{jkm}) r_{im} r_{jm} = 2 \left[\sum_i r_i \sum_i s_{ikm} r_{im} - \sum_i s_{ikm} r_{im}^2 \right]$$

The scores for the non rate effects will be:

For subtype 1 (BFORCE):

2 different cases: the probability incorporates whether there is a link from i to j, so the scores will differ depending whether the change is accepted or not.

$$c_{rk} = \sum_{m=1}^{M} \begin{cases} (1 - p_{ij}) s_{i,j\delta k} & \text{accept} \\ -p_{ij} s_{i,j\delta k} & \text{reject} \end{cases}$$

For subtype 2 (BAGREE), 4 different cases as we combine the probabilities differently depending on whether there is a tie or not:

$$c_{rk} = \sum_{m=1}^{M} \begin{cases} (1 - p_{ij}) s_{i,j\delta k} + (1 - p_{ji}) s_{j,i\delta k} & \nexists \text{ link } i \to j, \text{ accept} \\ \dfrac{-p_{ji} p_{ij}}{(1 - p_{ij} p_{ji})} \left[(1 - p_{ij}) s_{i,j\delta k} + (1 - p_{ji}) s_{j,i\delta k} \right] & \nexists \text{ link } i \to j, \text{ reject} \\ \dfrac{(1 - p_{ij})(1 - p_{ji}) \left[s_{i,j\delta k} p_{ij} + s_{j,i\delta k} p_{ji} \right]}{p_{ij} + p_{ji} - p_{ij} p_{ji}} & \exists \text{ link } i \to j, \text{ accept} \\ -s_{i,j\delta k} p_{ij} - s_{j,i\delta k} p_{ji} & \exists \text{ link } i \to j, \text{ reject} \end{cases}$$

For subtype 3 (BJOINT):

$$c_{rk} = \sum_{m=1}^{M} \begin{cases} (1 - p)(s_{i,j\delta k} + s_{j,i\delta k}) & \text{accept} \\ -p(s_{i,j\delta k} + s_{j,i\delta k}) & \text{reject} \end{cases}$$

5.6 Calculate statistics

- set leavers and structurally determined values of network links back to the ones in the stored data as at start of current period (so don't count in change etc.)
- temporarily set the behaviour variables back to the beginning of the simulation while calculate the network statistics and vice versa
- when calculating the statistics for any period, ignore any actor with missing information at either end of the period. Except for endowment statistics where the end of period values are not relevant. (In any network, any statistic, or just those which are used in the current calculation? Only looks at current network at the moment. Not done by missing flags for the actor.)

5.7 Change network composition

```
if leaver then
    set active flag to false
    remove any links current
    Reset Structactive flags for all other active actors. (only structactive if have
    a link which is not structurally determined to an active actor, and you've just
    set an actor inactive.)
else if joiner then
    Set active flag to true
    activate ties from the stored networks into the current simulated one.
    reset structactive flags
end if
update current average mean of the behaviour variables, which excludes
structurally inactive actors. Also update the time so far to the time at which the
composition changes.
update the scores if necessary
```

6. Prior processing in R

starting values Note that structural values are excluded from the distance but not from the total cell count (and missing cell count is done by subtraction). Need care to get these sorted!

effects user requests - validate as appropriate. If there are structural zeros will need to check whether some effects should be removed. (not done yet!)

missing data Create a separate edge list, and replace in the original one by 0 or any value in a preceding network. Similarly for behavior variables: impute by last previous observation or next following or mode for the wave.

structural zeros and ones Create a separate edge list, and replace by (value-10) in the original one. To do: Make the value 10 a parameter, with any lower values non-structural and those higher structural.

if conditional remove the relevant rate parameter from the working copies of theta
Remove corresponding targets, and associated flags
Divide the other basic rate parameters by the one removed
Set up array to store the times used in simulations in phase 3 from which to estimate the parameter conditioned on
Extract relevant targets for use as distance.

joiners and leavers Set the values in the networks in accordance with the option requested in the input flag, and the pattern, as below.

- Can only do unconditional estimation with joiners or leavers.
- Input format is a list of vectors, one for each actor, containing the start and end of (inclusive) intervals during which the actor was present.
- Attributes are attached to the object, (created in `sienaDataCreate`) containing:

  action matrix with row for each actor, column for each observation. Values are
  1. Data to be imputed or zeroed is not preceded or followed by real data
  2. Data to be imputed or zeroed is preceded but not followed by real data
  3. Data to be imputed or zeroed is both preceded and followed by real data

  event data frame with details of the events in the format required for C++ (see above!) (Actually exclude events at less then time 1e-10 here).

  activeStart Matrix of booleans with row for each actor, column for each observation. TRUE if the actor was active at the start of the period. passed unchanged to C++.
- alter the network data, using the `action` attribute.

```
for each network using this actor set (not behavior variables) do
  for each period do
    for each actor with an action value of 1 do
      if the composition change option is 1 or 2 then
        zero the data in the value edgelist
        remove the entries from the missing data edgelist
        remove from the data to be used to calculate distances for
        the report
      else {option is 3}
        add to missing data edgelist
        mark as missing in data to be used to calculate distances
        for the report
      end if
    end for
```

```
      for each actor with an *action* value of 3 do
        if the composition change option is 1 or 2 then
          carry forward the data in the preceding edgelist
          remove the entries from the missing data edgelist
          set to 0, not missing in the data to be used to calculate
          distances for the report
        else {option is 3}
          add to missing data edgelist
          mark as missing in data to be used to calculate distances
          for the report
        end if
      end for
      for each actor with an *action* value of 2 do
        if the composition change option is 1 then
          carry forward the data in the preceding edgelist
          remove the entries from the missing data edgelist
          set to 0, not missing in the data to be used to calculate
          distances for the report
        else {option is 2 or 3}
          add to missing data edgelist
          mark as missing in data to be used to calculate distances
          for the report
        end if
      end for
    end for
  end for
```

- Check for empty sets of active actors:
  - for each pair of consecutive periods, count number active at start of both.
  - If this is less than 2,
    - count how many active at start
    - go through the events to find if at any stage there is only one or none left
    - If so, quit with an error message that one or none is left.