nwrem

Relational event models in Stata
(a package that builds on `nwcommands`)

User manual and reference

Thomas U. Grund
RWTH Aachen University

This manual describes `nwrem` 1.7.x
(examples produced with `nwrem` 1.7.0, 21 November 2026)

How to cite. This manual describes the series 1.7 of `nwrem` (it holds for all versions 1.7.x); the listings were produced with version 1.7.0 (21 November 2026). It is the citable reference for the package. Please cite it as:

Grund, T. U. (2026). *nwrem: Relational event models in Stata. Manual.* Zenodo. https://doi.org/10.5281/zenodo.24398012

Please also report the exact version of the program that produced your results, as printed by `nwrem version` and stored in `e(version)`.

Acknowledgments

`nwrem` implements the relational event framework of Carter T. Butts [1] and exists because of his work. The R package `relevent` [2], also by Butts, is the reference implementation against which the statistics, the likelihoods and the estimates of `nwrem` were checked (version 1.2-1); the R code and manuals of `relevent` were the specification against which the Stata code was written and tested. `nwrem` is not affiliated with or endorsed by the author of `relevent`, and its errors are the author's. It is meant for researchers who work in Stata or prefer it to R. It is not a replacement for `relevent`, which remains the reference and the more complete program.

Use of an AI coding assistant. The software was developed with Claude Code (Anthropic), an AI coding assistant, working under the author's direction. It helped port algorithms from `relevent` into Stata, Mata and C and helped write the tests. The author specified the scope, reviewed the code, and is responsible for the package. Its correctness rests on validation, not on the assistant: the results are compared numerically with `relevent`, and the test suite reruns these comparisons. This manual was drafted with the same assistant; every listing in it is the output of a real run of the program.

Contents

Acknowledgments **2**

1 Introduction **5**
1.1 What `nwrem` is . . . 5
1.2 Relation to `relevent` . . . 5
1.3 What is covered . . . 6
1.4 Installation and requirements . . . 6
1.5 License . . . 7

2 The model **8**
2.1 Events, history and risk sets . . . 8
2.2 The rate of a pair . . . 8
2.3 The ordinal likelihood . . . 8
2.4 The waiting-time (interval) likelihood . . . 9
2.5 There is no intercept . . . 9
2.6 Risk sets . . . 9
2.7 Estimation . . . 9

3 Getting started **11**
3.1 Declaring the events . . . 11
3.2 The data of the examples . . . 12
3.3 A first model . . . 13
3.4 Coefficient names and order . . . 14

4 Syntax and options **15**
4.1 Syntax . . . 15
4.2 Effects . . . 15
4.3 Likelihood and conditioning . . . 16
4.3.1 interval, starttime() and endtime() . . . 16
4.3.2 condition() . . . 16
4.4 Risk set . . . 16
4.5 Posterior mode . . . 17
4.6 Other options . . . 17
4.7 Covariates of the actors . . . 17
4.8 Covariates of the pair . . . 18
4.8.1 covmatch(): matching the networks to the actors . . . 18
4.9 Effects that cannot be estimated . . . 19

5 The effects **20**
5.1 Notation and conventions . . . 20
5.2 Degree effects . . . 20
5.3 Inertia and reciprocity of receipt . . . 21

5.4 Recency effects . . . 21
5.5 Covariates . . . 22
5.6 Triadic and shared-partner effects . . . 22
5.7 Fixed effects . . . 22
5.8 Participation shifts . . . 22

6 Stored results and comparing models **24**
6.1 Stored results . . . 24
6.2 Information criteria and likelihood-ratio tests . . . 25
6.3 Replaying, the confidence level and reproducibility . . . 26

7 Diagnostics and graphs after the fit **28**
7.1 What the diagnostics need . . . 28
7.2 Graph options . . . 28
7.3 The rank of the observed dyad: `estat rank` . . . 29
7.4 Activity of the actors: `estat actors` . . . 30
7.5 The course over time: `estat time` . . . 32

8 Worked examples **35**
8.1 History and activity . . . 35
8.2 Covariates of actors and of pairs . . . 36
8.3 The waiting-time likelihood . . . 39
8.4 Risk sets, entry and exit . . . 42
8.5 Posterior mode . . . 45
8.6 Triadic, recency, fixed and participation effects . . . 46

9 Validation against `relevent` **50**
9.1 Estimates against `relevent` . . . 50
9.2 The shared out-partner statistic . . . 52
9.3 The likelihood, gradient and Hessian of the plugin . . . 52
9.4 Effects that are not estimable . . . 52
9.5 Other checks . . . 52

10 Limits, known issues and messages **53**
10.1 What is not implemented . . . 53
10.2 Known issues and cautions . . . 53
10.3 Messages and what to do . . . 54

11 Technical notes **57**
11.1 The structure of a fit . . . 57
11.2 What the plugin computes . . . 57
11.3 Speed . . . 58

A From R's `relevent` to `nwrem` **60**

Chapter 1

Introduction

1.1 What `nwrem` is

`nwrem` fits relational event models (REMs) in Stata. A relational event is one act from a sender to a receiver at a known moment: an e-mail, a phone call, a message in a chat, a handshake. The data are a list of such events with their times. A REM looks at the events one after the other and asks, at each moment, why the next event was between *this* sender and *this* receiver and not between any other pair of actors who could have interacted instead. Statistics of the history, such as how active the sender has been so far, whether the sender has written to this receiver before, or whether the receiver has just written to the sender, raise or lower the rate of each pair, and `nwrem` estimates by how much. The model is that of Butts [1].

`nwrem` is a Stata package of its own that builds on the `nwcommands` package. It works on event data declared with `nwset`'s `eventtime()` option, reads the attributes of the actors from the ordinary Stata dataset, and returns its results in `e()`, so that `estimates`, `lrtest` and the other postestimation tools of Stata work. The likelihood, its gradient and its exact Hessian are evaluated by a compiled plugin; the estimates are the maximum of the likelihood (or the posterior mode) to the accuracy of the floating-point arithmetic.

1.2 Relation to `relevent`

The reference implementation of the model is the R package `relevent` [2] by Carter T. Butts. `nwrem` follows it in its model, in the names and definitions of its statistics and in its likelihoods, and its estimates were checked against it (Chapter 9). `nwrem` is nevertheless an *independent reimplementation*, written in Stata, Mata and C, using the published description of the model and the documented behavior of `relevent` (version 1.2-1) as the specification.

Where the two programs differ in a way that matters to a user, this manual says so. The most important differences are these.

- The statistics are chosen by Stata options (`nodsnd`, `frpsndsnd`, `covsnd(varlist)`), not by a character vector of effect names. The option names are those of `relevent` in lower case. The translation table is in Appendix A.
- The event list is declared with `nwset` (Chapter 3) instead of being passed as a matrix; actors that never occur in the list are added with `nwaddnodes`; the attributes of the actors are variables of the current dataset.
- The statistic `ospsnd` is the symmetric statistic that its definition describes, not the asymmetric one that `relevent` 1.2-1 computes (Section 5.6).
- `nwrem` always computes the covariance matrix (the exact observed information); `relevent` does so only with `hessian=TRUE`.

1.3 What is covered

- **Likelihoods.** The ordinal partial likelihood and the waiting-time (interval) likelihood, each with maximum likelihood or with the posterior mode under independent Student t priors, and the conditioning on the first events (Chapter 2).
- **Effects.** All statistics of `relevent` that are not identically zero: the degree and inertia effects, the recency effects, the covariate effects (of actors and of pairs, with several covariates per option), the triadic and shared-partner effects, the fixed effects of the actors and six participation shifts (Chapter 5).
- **Risk sets.** The complete risk set of all ordered pairs, a restriction to given pairs, and actors that enter and leave the observation.
- **Postestimation.** The results of every fit are stored for `estimates` (including `estimates stats`), `lrtest` and the other standard tools (Chapter 6), and three diagnostics with graphs, `estat rank`, `estat actors` and `estat time`, which compare the fitted model with the observed events (Chapter 7).

What `relevent` has that `nwrem` does not. Bayesian simulation (MCMC and sampling importance resampling), the sampling of the risk set (`sample.size`), the simulation of event sequences, covariates that change over time are not implemented (Chapter 10). There is no intercept, for a reason that lies in the model (Section 2.5).

1.4 Installation and requirements

`nwrem` needs Stata 17 or later and the core package `nwcommands`, version 3.29.0 or later. Install the core first, then `nwrem`:

```
. net install nwcommands, from(https://nwcommands.org/net/) replace
. nwinstall, modelpkg(nwrem)
```

`nwinstall, modelpkg(nwrem)` installs the latest release; `nwinstall, everything` installs the core and all four model packages in one step. Without `nwinstall` the package can be installed directly from its repository, and a particular version from the release tag:

```
. net install nwrem, from(https://raw.githubusercontent.com/thomas-u-grund/nwrem/main/) replace
. net install nwrem, from(https://raw.githubusercontent.com/thomas-u-grund/nwrem/v1.7.0/)
    replace
```

`net describe nwrem` shows the version and the distribution date of the installed package, `nwrem version` the version of the package, of the core, the plugin in use and the version of `relevent` that the results were compared with, and `e(version)` holds the version in the stored results. Quote it when you report an analysis.

The native plugin. Every fit runs in a compiled plugin (Chapter 11). The package ships the plugin for macOS (universal), Linux and Windows; there is no slower fallback in Mata. If the file for your platform is missing, `nwrem` stops with return code 601 and names the file; if the installed plugin does not fit the `.ado` files or cannot be loaded, it stops with return code 3495. Reinstall the package in both cases. If the plugin does not load on your computer, you can compile it there:

```
. nwinstall, buildplugin(nwrem)
```

This needs a C compiler (on macOS the command line tools of Xcode, on Linux `gcc` and `make`, on Windows the Microsoft Build Tools); restart Stata afterwards. The C source is part of the package (`net get nwrem, from(https://raw.githubusercontent.com/thomas-u-grund/nwrem/main)`

copies it into the current directory) and of the repository `https://github.com/thomas-u-grund/nwrem`.

The Mata library. The Mata library of `nwrem` (`lnwrem.mlib`) is built with Stata 17, the oldest supported version, because a library built by a newer Stata is ignored by an older one (return code 3499). If the library does not load in your Stata, build it with your own Stata: `nwinstall, buildmata(nwrem)`. The command clears the data and the Mata objects in memory, so save first, and restart Stata afterwards.

Platforms and versions. Stata 17 or later is required; Stata 16 and older have not been tested and are not expected to work. The tests of the fitting command were run on macOS with Stata 17, Stata 18 and StataNow 19.5. The tests of the diagnostics of Chapter 7 were run on macOS with Stata 17 and on Linux (Ubuntu 22.04) and Windows 11 with StataNow 19.5 MP; Stata 18 was not run for them. The matching of covariate networks (`covmatch()`, Section 4.8.1) was tested on macOS with Stata 17 only. The timings in Chapter 11 were measured on Linux (Ubuntu 22.04, StataNow 19.5 MP).

The examples of this manual. All listings are the output of Stata 17 runs with `nwrem` 1.7.0 and the core package 3.33.0; the do-files that produce them are in the folder `manual/ex` of the repository. The data are the e-mail log of a manufacturing company (Section 3.2); they are distributed with the core package and read by `nwwebuse`.

1.5 License

`nwrem` is free software under the GNU General Public License, version 3, and Thomas U. Grund is its sole author. Because the package follows the structure of `relevent`, whose license is GPL (≥ 2), it is distributed under the GPL-3, which is compatible with it. The example data are not covered by the GPL-3 of the code; their source and license are given in Section 3.2.

Chapter 2

The model

This chapter states the model that `nwrem` fits and the quantities that it reports. The notation follows Butts [1]. The statistics themselves are defined in Chapter 5; how to run the command is the subject of Chapters 3 and 4.

2.1 Events, history and risk sets

The data are m events $e_1, \ldots, e_m$ among n actors. Event $e_i = (s_i, r_i, t_i)$ is an act of the sender s_i towards the receiver $r_i \neq s_i$ at time t_i, in the order of the times. Events with the same time keep the order in which they stand in the network; because the ordinal likelihood depends only on the order, this order is part of the data. Every event is read as directed from the first variable (the sender) to the second (the receiver).

Let h_i be the history before event i: the events $e_1, \ldots, e_{i-1}$. At event i every ordered pair (j, k) with $j \neq k$ that is in the *risk set* $\mathcal{R}_i$ could have been the event. By default $\mathcal{R}_i$ holds all $n(n-1)$ ordered pairs; Section 2.6 describes the restrictions. Each pair has a vector of *statistics* $x(j, k; h_i)$, computed from the history only: the share of the earlier events in which j sent, whether j has written to k before, how recently, and so on. All statistics depend on the earlier events, never on the coefficients and never on the event itself.

2.2 The rate of a pair

The model gives each pair a rate (an intensity) that depends on the statistics through a log-linear predictor,

$$\lambda_i(j, k) = \exp\{\theta^\top x(j, k; h_i)\}, \qquad (j, k) \in \mathcal{R}_i, \tag{2.1}$$

with the coefficient vector θ to be estimated. A positive coefficient means that pairs with a larger value of the statistic have a higher rate. The coefficients are log rate ratios: a difference of one unit in a statistic multiplies the rate of a pair by $\exp(\theta_p)$.

2.3 The ordinal likelihood

The default likelihood uses only the order of the events. Given that an event occurs at all, the probability that it is the pair (j, k) is the rate of that pair relative to the rates of all pairs in the risk set. The likelihood is the product of these probabilities over the events,

$$L(\theta) = \prod_{i > c} \frac{\exp\{\theta^\top x(s_i, r_i; h_i)\}}{\sum_{(j,k) \in \mathcal{R}_i} \exp\{\theta^\top x(j, k; h_i)\}}. \tag{2.2}$$

It is a partial likelihood in the sense of Cox [3]: the rates of all pairs may share an unspecified baseline that changes over time, and it cancels. The first c events, set with `condition(c)`, are used to build the history of the statistics but do not enter the product (Section 4.3.2).

2.4 The waiting-time (interval) likelihood

With the option `interval` the times between the events enter the likelihood as well, so that the model describes the *rate of events* and not only their order. Let $\Delta_i = t_i - t_{i-1}$, with t_0 the start of the observation (`starttime()`, default 0), and let $\Lambda_i = \sum_{(j,k)\in\mathcal{R}_i} \lambda_i(j,k)$ be the total rate over the risk set. The contribution of event i is the density of the waiting time and of the pair, $\lambda_i(s_i, r_i)\exp(-\Delta_i\Lambda_i)$. The time from the last event to the end of the observation T (`endtime()`) contributes the probability of no event in it, $\exp\{-(T - t_m)\Lambda_{m+1}\}$. The log likelihood is

$$\ell(\theta) = \sum_{i>c}\Big[\theta^\top x(s_i, r_i; h_i) - \Delta_i\Lambda_i\Big] - (T - t_m)\Lambda_{m+1}. \tag{2.3}$$

The waiting times must be strictly increasing: two events with the same time are refused (return code 459), as `relevent` refuses them. With `condition(c)` the first interval that counts is the one after event c.

2.5 There is no intercept

`nwrem` has no intercept option. In the ordinal likelihood a term that is constant over the pairs of a risk set cancels in the ratio of equation (2.2); this is the same reason for which a Cox model has no identifiable baseline. The waiting-time model has no intercept either: a pair with all statistics equal to zero has the rate 1 per unit of time. The scale of the time variable therefore fixes the overall level of the rates, and the coefficients of statistics that are shares (the degree effects, the inertia effects) have to make up for a baseline that does not fit. If the times are measured in a unit in which a typical pair has a rate far below 1, these coefficients become large and negative. Choose the unit of time so that the rate of a typical pair is of the order of one event per unit; Section 8.3 shows the effect and a simple rule.

2.6 Risk sets

The risk set can be restricted in two ways, and the two can be combined. A network (`riskset()`) lists the ordered pairs that can ever be an event; all other pairs are removed from every $\mathcal{R}_i$. Per-actor times of entry and exit (`entry()`, `exit()`) remove an actor, as sender and as receiver, from the risk sets of the events that lie outside its window. An actor is present at an event whose time lies between its entry and its exit time, both ends included. The risk set of an interval is the risk set at the time of the event that ends it. In the last interval, from the last event to `endtime()`, an actor is present if its entry time is not later than the end of the observation and its exit time is not earlier than it. The statistics are always computed from all events; only the pairs over which the likelihood normalizes are restricted. An event of a pair that is not in the risk set is an error.

2.7 Estimation

The statistics do not depend on θ, so the log likelihood is a sum of log-sum-exp functions of a linear predictor and is concave in θ. It has a unique maximum unless the effects are collinear or the data separate (an effect that never occurs among the events, for example, has a coefficient that tends to minus infinity). `nwrem` starts at $\theta = 0$ and takes damped Newton steps with the exact gradient and the exact Hessian of the likelihood (a full step, halved until the log likelihood does not fall), until the Newton decrement is at the level of round-off. The covariance matrix of the estimates is the inverse of minus the Hessian at the maximum, the observed information; the table shows the standard errors, z statistics, p-values and confidence intervals that follow from the normal approximation.

The effects that are exact functions of the effects before them in the model are found before the optimizer runs and left out, with the coefficient reported as 0 (Section 4.9).

Posterior mode. With `postmode` the function that is maximized is the log likelihood plus the log density of independent Student t priors, one for each coefficient, with location `priormean( )`, scale `priorscale()` and `priordf()` degrees of freedom: the posterior mode, as `relevent`'s `fit.method="BPM"`. A normal prior is obtained with `priordf(.)`. The defaults are the location 0, the scale 100 and four degrees of freedom, a very weak prior. The standard errors are those of the curvature of the log posterior at the mode, and `e(ll)` is the log likelihood at the mode without the prior. A posterior mode is useful where the maximum does not exist, because a coefficient tends to infinity, and as a shrinkage estimate.

The null model. `e(ll_0)` is the log likelihood of the model without effects. For the ordinal likelihood it is $-\sum_{i>c} \ln |\mathcal{R}_i|$, the same probability for every pair of the risk set. For the waiting-time likelihood it is the log likelihood of a constant rate ρ for every ordered pair, with $\rho = m/\{(T - t_0)\, n(n-1)\}$ estimated from all events; it is defined only for the complete risk set.

Chapter 3

Getting started

This chapter takes you from an event list to a first fitted model, shows how to read the output and explains how the coefficients are named. All commands were run with `nwrem` 1.7.0.

3.1 Declaring the events

`nwrem` does not read the events from variables. It takes the *name* of a network that exists in the current `nwcommands` session and that was declared as an *event network* with the option `eventtime()` of `nwset`:

```
. nwset sender receiver, directed eventtime(t) name(mail)
```

Each row of the data is one event: the variables `sender` and `receiver` identify the actors (numbers or strings) and `t` is the time. The rules for the event data are these.

- Every event needs a time, a sender and a receiver that are actors of the network. Events without a time or with a missing actor are refused when the network is declared, and by `nwrem` (return code 416) if they get there in another way.
- A sender must differ from the receiver. A self-loop is refused with return code 3498; drop such rows before `nwset`.
- Two-mode event networks are refused (return code 198).
- An event can carry a value (a third variable in `nwset`). `nwrem` reads sender, receiver and time only and ignores the value.
- Events with the same time keep the order of the rows (Section 2.3); they are allowed with the ordinal likelihood and refused with `interval`.
- The times need not be sorted; `nwrem` sorts the events.

Direction. Every event is read as directed from the first variable (the sender) to the second (the receiver), and the risk set at each event holds the $n(n-1)$ ordered pairs, whatever the declaration says; this follows `rem.dyad()` of `relevent`, which has no undirected variant. A network declared with `nwset`'s option `undirected` is fitted in the same way and gives the same results as the directed declaration; the option changes only the network summary that `nwset` builds from the events. Because the sender and receiver effects depend on the order of the two variables, put the actor who starts the event first. This manual declares event networks with the option `directed`.

Actors that do not occur in the events. The actors of the network are the actors that occur in the list. An employee who sends and receives nothing in the period is nevertheless a possible sender and receiver at every moment, and belongs to the set of pairs that the model compares. Add the missing actors as isolates with `nwaddnodes`, as in the example below; this changes n, and with it every normalizer of the likelihood.

Attributes of the actors. The per-actor variables for `covsnd()`, `covrec()`, `covint()`, `entry()` and `exit()` are read from the *current dataset*, not from the event data. `nwload` *netname*`, xvars` loads a dataset with one row per actor and the label of the actor in the variable `_nwnode`. When the dataset has this variable, each actor of the network is matched to its row by label, whatever the sort order of the data (an actor without a row, or with more than one, is refused with return code 198); without `_nwnode` the dataset must have exactly one row per actor, in the actor order of the network. A missing value of a covariate is refused (return code 416). Merge further variables into this dataset (Section 8.2) and name them in the options.

Pair covariates and risk sets are networks. A covariate of the pair (`covevent()`) and the pairs of a restricted risk set (`riskset()`) are other networks in memory with the same actors as the event network; By default row and column i of such a network are read as actor i of the event network (matching by position); the option `covmatch(label)` matches them by node label instead, and `nwrem` prints a note on how each network was matched (Section 4.8). Section 4.8 gives the rules.

3.2 The data of the examples

The examples use the e-mail metadata of a manufacturing company: 50,233 e-mails among 144 employees between January and September 2010, each with the sender, the receiver and the time in days since 1 January 2010; there is no content. The source is Michalski [4], released under CC0 1.0; see also Nurek and Michalski [5]. The files distributed with the core package are a derived subset prepared by the author of this package: only e-mails with exactly one recipient and only the 154 employees with a known supervisor were kept, e-mails to oneself were removed, the employee identifiers are prefixed with `E`, and the time is in days. They are read by `nwwebuse manufacturing` (the events) and `nwwebuse manufacturing_nodes` (the reporting hierarchy: the direct supervisor, the number of steps to the chief executive, and the division). The data are not covered by the GPL-3 of the code. To keep the fits quick, most examples use the first five days of the log (`t < 5`): 631 e-mails, which name 102 of the 144 employees; the other 42 are added as isolates.

```
. nwwebuse manufacturing, nwclear
(plain Stata dataset loaded - not a saved network file; build a network from it
directly, e.g. with nwfromedge or nwgenerate)

. list in 1/5

     +------------------------------+
     | sender   receiver          t |
     |------------------------------|
  1. |    E39        E81   3.235255 |
  2. |    E39        E81   3.235822 |
  3. |   E112       E136    3.25566 |
  4. |   E112        E60    3.25728 |
  5. |   E112        E57   3.257963 |
     +------------------------------+

. count
  50,233

. quietly levelsof sender, local(a) clean

. quietly levelsof receiver, local(b) clean

. local all : list a | b

. display "actors in the log: " `: word count `all''
```

```
actors in the log: 144

. keep if t < 5
(49,602 observations deleted)

. count
  631

. display "span in days: " %5.2f (t[_N] - t[1])
span in days:  1.75

. quietly levelsof sender, local(a) clean

. quietly levelsof receiver, local(b) clean

. local present : list a | b

. local miss : list all - present

. display "actors in the window: " `: word count `present'' ", added as isolates:
> " `: word count `miss''
actors in the window: 102, added as isolates: 42

. nwset sender receiver, directed eventtime(t) name(mail)

. local miss : subinstr local miss " " ",", all

. nwaddnodes mail, nodenames(`miss')

. nwset
(1 networks)
--------------------
      mail
```

3.3 A first model

The simplest model has two effects. `nodsnd` is the normalized out-degree of the sender, the share of all earlier events in which this actor was the sender. `nidrec` is the normalized in-degree of the receiver, the share of the earlier events in which this actor was the receiver. Chapter 5 defines every effect.

```
. nwrem mail, nodsnd nidrec
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                          Actors: 144
Events: 631                            Log likelihood: -5.99e+03
------------------------------------------------------------
-----------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   15.55723   1.605499     9.69   0.000
      nidrec |   25.03383   1.069949    23.40   0.000
-----------------------------------------------------
```

The header names the likelihood and the estimator, the network, the number of actors and of events and the log likelihood; the log likelihood is printed in a rounded form in the header, and `e(ll)` holds the exact value. The table lists each coefficient with its standard error, z statistic and p-value. Coefficient names are printed in full (up to 32 characters). The confidence intervals are left out when `linesize` is smaller than 100, because a long name then does not fit next to them; the option `ci` brings them back (names that do not fit are then cut), and so does an explicit

`level(#)`. The option `noci` leaves them out whatever the line size, and `ci` together with `noci` is refused (return code 198). The numbers do not depend on the display.

How to read the coefficients. A coefficient is the log rate ratio of equation (2.1). The degree effects are shares between 0 and 1, so their coefficients are large: a sender whose share of the earlier e-mails is 0.1 higher has $\exp(1.556) \approx 4.7$ times the rate of sending the next e-mail, and a receiver whose share of earlier receipts is 0.1 higher is $\exp(2.503) \approx 12$ times as likely to be the one written to. People who have been busy stay busy, and people who have received a lot keep receiving.

3.4 Coefficient names and order

The coefficients are named after the effect: `nodsnd`, `frpsndsnd`. The covariate effects add the name of the variable or of the network (`covsnd_level`, `covevent_samesup`); a name longer than 32 characters is cut to 32. The fixed effects add the number of the actor, counted in the actor order of the network from the second actor on (`fesnd_2`, `fesnd_3`, ...), because the first actor is the reference. The order of the coefficients is the fixed order of Table 5.1 in Chapter 5, not the order in which you typed the options. Several variables in one option give one coefficient each, in the order of the variables.

Replaying and storing. `nwrem` typed without arguments, or with `level()` only, redisplays the results of the last fit; `estimates replay` works as well. Results can be stored with `estimates store` and compared (Chapter 6).

Chapter 4

Syntax and options

This chapter lists every option of `nwrem`. The effects are defined in Chapter 5; the likelihoods and the estimators in Chapter 2; the names of the corresponding arguments of `relevent` are in Appendix A.

4.1 Syntax

```
nwrem [netname], effects [likelihood options] [risk set options] [prior options] [seed(#) level
    (#) ci noci]
nwrem [, level(#) ci noci]                          (replays the last fit)
nwrem version
```

netname is the event network to fit; the default is the current network. It must be an event network declared with `nwset ..., eventtime()`; any other network is refused with return code 198. At least one effect is required (return code 198); there is no default effect. The options can be given in any order.

4.2 Effects

The options that select the statistics of the model. Chapter 5 gives the definition of each effect.

Option	Description
`nodsnd`, `nidrec`, `nidsnd`, `nodrec`	Normalized degrees: out-degree and in-degree of the sender and of the receiver.
`ntdegsnd`, `ntdegrec`	Normalized total degree of the sender and of the receiver.
`frpsndsnd`	Inertia: the fraction of the sender's past sends that went to this receiver.
`frrecsnd`	Reciprocity of receipt: the fraction of the sender's past receipts that came from this receiver.
`rsndsnd`, `rrecsnd`	Recency of sending and recency of receipt (reciprocal rank).
`otpsnd`, `itpsnd`	Outbound and inbound two-paths.
`ospsnd`, `ispsnd`	Outbound and inbound shared partners.
`covsnd(varlist)`, `covrec(varlist)`, `covint(varlist)`	Numeric covariates of the actors: of the sender, of the receiver, of both together. One coefficient per variable.
`covevent(netlist)`	Covariates of the pair, read from networks with the same actors. One coefficient per network.
`covmatch(position|label)`	How the networks of `covevent()` and `riskset()` are matched to the actors of the event network: by `position` (default) or by node `label`. Section 4.8.1.
`fesnd`, `ferec`, `feint`	Fixed effects of the actors as sender, as receiver, as both; one coefficient per actor but the first.
`psabba`, `psabby`, `psabxa`, `psabxb`, `psabxy`, `psabay`	Participation shifts, which depend on the previous event.

4.3 Likelihood and conditioning

Option	Description
`interval`	The waiting-time (interval) likelihood, equation (2.3), instead of the ordinal likelihood. Requires strictly increasing event times.
`starttime(#)`	Start of the observation, with `interval`. Default 0. Without `interval` the option is refused (return code 198).
`endtime(#)`	End of the observation, with `interval`. Default: the time of the last event, so that there is no last interval. Without `interval` the option is refused.
`condition(#)`	The first # events only build the history of the statistics and do not enter the likelihood. Both likelihoods.

4.3.1 interval, starttime() and endtime()

With `interval` the likelihood of the waiting times, equation (2.3), replaces the ordinal likelihood. The first interval begins at `starttime()`; a start later than the first event is an error (return code 459). `endtime()` marks the end of the observation: the time from the last event to it contributes the probability of no event in that interval. An end earlier than the last event is an error (459). If `endtime()` is not given the observation ends with the last event. Two or more events with the same time are refused (459). Because the rate of a pair with all statistics zero is one per unit of time, choose the unit of the time variable accordingly (Section 2.5). The stored results `e(starttime)` and `e(endtime)` hold the window (`e(endtime)` only when `endtime()` was given).

4.3.2 condition()

`condition(c)` lets the first c events (in the order of the times) build the history of the statistics but removes them from the likelihood; c must be a non-negative integer smaller than the number of events. This is useful where the statistics of the first events rest on too little history: the degree shares after one event are 0 or 1. The statistics of all later events still count the conditioned events. `e(N)` is the number of all events, `e(N_cond)` the number of conditioned events. With `interval` the first interval that counts is the one after event c.

4.4 Risk set

Option	Description
`riskset(netname)`	A network with the same actors. A tie value of 0 means that the ordered pair can never be an event, any other value that it can.
`entry(varname)`	Per-actor time of entry. Before it the actor is not in the risk set, as sender and as receiver. A missing value: present from the start.
`exit(varname)`	Per-actor time of exit. After it the actor leaves the risk set. A missing value: present until the end.

`riskset()` reads the ties of the network: the cell from the sender to the receiver decides, the diagonal is ignored. A network with missing tie values is refused (return code 416), a two-mode network is refused (return code 6088), and the network must have the same number of actors as the event network (return code 198). `entry()` and `exit()` name numeric variables of the current dataset and are matched to the actors exactly as the covariates of Section 3.1; each may be given

alone. The times are on the scale of the event times. An actor is present at an event whose time lies between its entry and its exit time, both ends included. The three options can be combined. If an event involves a pair that is not in the risk set, or an actor outside its window, `nwrem` stops with return code 459 and names the event (its number in the sorted sequence); remove the event or widen the risk set.

4.5 Posterior mode

Option	Description
`postmode`	Posterior mode instead of the maximum likelihood estimate: independent Student t priors on all coefficients.
`priormean(#)`	Location of the prior. Default 0.
`priorscale(#)`	Scale of the prior; must be positive. Default 100.
`priordf(#)`	Degrees of freedom of the prior; must be positive. Default 4. `priordf(.)` is a normal prior.

The three prior options are refused without `postmode` (return code 198). The prior is the same for every coefficient, including the coefficients of the fixed effects and the covariates. The posterior mode corresponds to `fit.method="BPM"` of `relevent` with `prior.mean`, `prior.scale` and `prior.nu`. The prior is not applied to `e(ll)`; the stored results `e(priormean)`, `e(priorscale)` and `e(priordf)` hold the prior of the fit.

4.6 Other options

Option	Description
`seed(#)`	Sets the random-number seed before the estimation; an integer from 0 to 2147483647 (anything else: return code 198). The fit starts from zero and uses random starting values only when the Newton iteration does not reach a maximum (no maximum exists, or effects are not identified), so the seed makes such fits reproducible. Different seeds give the same estimate up to the convergence tolerance.
`level(#)`	Confidence level of the displayed intervals. Default 95. Given explicitly, it also shows the intervals (Section 3.3). Also accepted by the replay form of the command.
`ci`	Shows the confidence intervals even when `linesize` is smaller than 100; names that do not fit next to them are cut.
`noci`	Leaves the intervals out. `ci` together with `noci`: return code 198.
The options `nopvalues`, `cformat()`, `pformat()` and `sformat()` of Stata's coefficient tables are passed through.	

4.7 Covariates of the actors

`covsnd(varlist)`, `covrec(varlist)` and `covint(varlist)` take numeric variables of the current dataset (a string variable is refused with return code 109), with one value per actor as described

in Section 3.1. Every variable gets its own coefficient. The same variable may appear in several of the three options. The coefficient of a variable `x` is named `covsnd_x`, `covrec_x` or `covint_x`. `nwrem` has no effect that compares categories, so a categorical attribute is entered as indicator variables.

4.8 Covariates of the pair

`covevent(netlist)` takes one or several networks. A pair covariate belongs to the ordered pair (sender, receiver), not to one actor, so it is read from the tie values of another network, as the dyadic covariates of `nwergm` are. Declare the covariate as its own network first and name it in the option; the coefficient is named `covevent_`*netname*. The rules:

- The network must have the same actors as the event network (return code 198).
- The value for the ordered pair (sender, receiver) enters the rate of that pair directly; there is no broadcasting of an actor value, as with `covsnd()`.
- A two-mode network is refused (return code 6088), and so is a network with missing tie values (return code 416). The diagonal is not part of the risk set; a missing or non-zero diagonal is read as zero.
- Declaring a covariate network from an edge list with `nwset` does *not* symmetrize it: an ordered pair without a row gets the value 0 even if the reverse pair was given. For a symmetric covariate supply both ordered pairs, or build it from a matrix (Section 8.2). For a directional covariate ("i reports to j") the asymmetry is exactly what `covevent()` is for. If you give the covariate as an edge list, add the option `edgelist` to `nwset`, because `nwset` otherwise reads three variables as a wide affiliation matrix.

4.8.1 covmatch(): matching the networks to the actors

The networks of `covevent()` and `riskset()` must have the same number of actors as the event network, but `nwrem` has to decide which row and column of such a network belongs to which actor of the event network. `covmatch(position)`, the default, reads row and column i as actor i of the event network, whatever the node labels; a covariate network whose nodes are in another order is then misaligned without an error. `covmatch(label)` reorders the covariate network by node label: the label sets of the two networks must be identical and every label must occur once, otherwise `nwrem` stops with return code 459 and a message. The option applies to every network of `covevent()` and to `riskset()` and is an error without one of them; its value is stored in `e(covmatch)`.

Every fit with `covevent()` or `riskset()` prints one note line per network that says how the network was matched, for example

```
covevent(samesup) matched to the event network by position (labels agree)
covevent(samesup) matched to the event network by label
```

With position matching the note says whether the labels of the two networks agree. If they do not, the line is a warning (in red on the screen): *by position, but the node labels are in a DIFFERENT ORDER* when the same labels occur in another order, and *by position (labels DIFFER: position matching may be wrong ...)* when the labels are different. Then check the networks, or fit again with `covmatch(label)`.

The per-actor variables of `covsnd()`, `covrec()`, `covint()`, `entry()` and `exit()` are matched in the same spirit but without this option: when the current dataset has the variable `_nwnode`, each actor is matched to its row by label, whatever the order of the rows (an actor without a row, or with two rows, is refused with return code 198); without `_nwnode` the dataset must have exactly one row per actor, and row i is actor i of the event network.

4.9 Effects that cannot be estimated

Exact collinearity. An effect that is an exact function of the effects before it in the model, apart from a term that is constant over the risk set at every event, cannot change the likelihood. Examples are `ntdegsnd` together with `nodsnd` and `nidsnd` (it is half their sum), `covint(x)` together with `covsnd(x)` and `covrec(x)`, or any of `frpsndsnd`, `rsndsnd` and the fixed effects in a very small network. `nwrem` finds such effects before the optimization, using the order of Table 5.1: the effect that comes later is left out. Its coefficient is reported as 0 with the label `(omitted)`, its variance is 0, it is named in `e(unidentified)`, and a note says so under the table. The remaining coefficients and the log likelihood are those of the model without it. All effects are examined, including the participation shifts, the triadic effects and the fixed effects. An effect is omitted when what is left of its sum of squares, centered within each risk set, after the effects before it is below 10^{-12} of its uncentered sum of squares. The check is exact for statistics that are exact functions of others; it does not detect a near-collinearity (see below).

Near-collinearity and separation. A set of effects that is nearly, but not exactly, collinear gives a likelihood with a flat ridge: different starting values would reach visibly different individual coefficients with almost the same log likelihood. Because the fit starts at zero and converges to the exact maximum, the estimates are reproducible, but a single coefficient on a ridge is imprecise, and its standard error is large. The degree effects are running shares that converge towards $1/n$ for every actor as the history grows, so their cross-actor spread narrows; estimates of these effects can stay noisy even with thousands of events. This is a property of the effects, not of the program. A coefficient that tends to infinity because the data separate (an actor that never sends, with `fesnd`) is reported with a huge negative value and a standard error far above 100; such an effect is listed in `e(unidentified)` as well, with the same note, and is *not* set to zero. Use `postmode` to obtain a finite estimate.

Fixed effects. The fixed effects add one coefficient per actor but the first for each of `fesnd`, `ferec` and `feint`. A model with more than 600 coefficients of this kind (the number of actors times the number of fixed-effect options above 600) is refused (return code 198). They are evaluated for every pair at every event, so they are slower for large networks than the other effects.

Other stops. A model that cannot be fitted (for example, fewer than two events: return code 2001; the optimization found no maximum: return code 430) stops with a message. The messages that the examples produced are listed in Section 10.3.

Chapter 5

The effects

This chapter defines every effect precisely. An effect is a statistic $x_p(j, k; h_i)$ of the ordered pair (j, k), sender j and receiver k, computed from the history h_i before event i (equation (2.1)). All 14 effects of the original framework, the triadic effects, the fixed effects and the participation shifts are evaluated in the plugin. The coefficients appear in the output in the order of Table 5.1.

5.1 Notation and conventions

At event i let $q = i - 1$ be the number of earlier events. Let $a(j)$ be the number of earlier events with sender j (the out-count), $b(j)$ the number with receiver j (the in-count), and $c(j, k)$ the number of earlier events from j to k. All counts include the conditioned events (`condition()`), and they do not depend on the risk set, on the likelihood and on the coefficients. Repeated events between the same pair count again. Events with the same time are in the order of the network, so an event sees the earlier-listed events of its own time. The number n is the number of actors of the network, including the isolates added with `nwaddnodes`.

The first event. Before any event has happened there is no history. At the first event every degree share is defined as $1/(n-1)$ for all actors, as in `relevent`; the same value for all actors is irrelevant for the ordinal likelihood, and for the waiting-time likelihood it only enters the rate of the first interval. The inertia statistics follow the same convention for each actor separately (below): a sender who has sent nothing yet has $1/(n-1)$ for every receiver in `frpsndsnd`, not 0, and an actor who has received nothing yet has $1/(n-1)$ for every partner in `frrecsnd`. The other history statistics have no such convention: `rsndsnd`, `rrecsnd`, the triadic and shared-partner effects and the participation shifts are 0 when nothing has happened yet. The values are part of the fitted model; reading the fractions literally as 0 would give a different log likelihood.

5.2 Degree effects

The six degree effects are *shares* of the activity so far. `nodsnd` is the share of the earlier events in which the sender was the sender, `nidrec` the share in which the receiver was the receiver. `nidsnd` and `nodrec` cross the roles: the share of earlier events in which the *sender* was a receiver, and in which the *receiver* was a sender. `ntdegsnd` and `ntdegrec` are the total degree: half the sum of the out-share and the in-share (the average of the two shares, as `NTDegSnd` and `NTDegRec` of `relevent`). Because the shares sum to one over the actors, they move towards $1/n$ for everyone as the history grows; at the first event, when there is no past, each share is $1/(n-1)$. The effects describe the *persistent* activity of an actor: a positive `nodsnd` means that actors who have been the senders of a larger share of the history tend to be the senders of the next event.

Table 5.1: The effects in the order of the coefficients.

#	Option	Coefficients	Statistic $x(j,k)$
1	`nodsnd`	1	$a(j)/q$
2	`nidrec`	1	$b(k)/q$
3	`nidsnd`	1	$b(j)/q$
4	`nodrec`	1	$a(k)/q$
5	`ntdegsnd`	1	$\{a(j)+b(j)\}/(2q)$
6	`ntdegrec`	1	$\{a(k)+b(k)\}/(2q)$
7	`frpsndsnd`	1	$c(j,k)/a(j)$
8	`frrecsnd`	1	$c(k,j)/b(j)$
9	`covsnd(varlist)`	one per variable	z_j
10	`covrec(varlist)`	one per variable	z_k
11	`covint(varlist)`	one per variable	$z_j + z_k$
12	`covevent(netlist)`	one per network	w_{jk}
13	`rsndsnd`	1	1/rank of k among the receivers of j
14	`rrecsnd`	1	1/rank of k among the senders to j
15	`otpsnd`	1	$\sum_l \min\{c(j,l), c(l,k)\}$
16	`itpsnd`	1	$\sum_l \min\{c(k,l), c(l,j)\}$
17	`ospsnd`	1	$\sum_l \min\{c(j,l), c(k,l)\}$
18	`ispsnd`	1	$\sum_l \min\{c(l,j), c(l,k)\}$
19	`fesnd`	$n-1$	$1\{j=u\}$ for $u=2,\ldots,n$
20	`ferec`	$n-1$	$1\{k=u\}$ for $u=2,\ldots,n$
21	`feint`	$n-1$	$1\{j=u \text{ or } k=u\}$ for $u=2,\ldots,n$
22	`psabba`	1	previous event $A \to B$: pair is (B,A)
23	`psabby`	1	pair is (B,y), $y \neq A$
24	`psabxa`	1	pair is (x,A), $x \neq B$
25	`psabxb`	1	pair is (x,B), $x \neq A$
26	`psabxy`	1	pair is (x,y) with $x,y \notin \{A,B\}$
27	`psabay`	1	pair is (A,y), $y \neq B$

The sums run over the actors l other than the two of the pair. z is a covariate of the actors, w the tie value of a covariate network. The shares divide by q and, at the first event, are $1/(n-1)$ (Section 5.1); a ratio with a zero denominator is $1/(n-1)$ for $a(j)=0$ or $b(j)=0$.

5.3 Inertia and reciprocity of receipt

`frpsndsnd` is the fraction of the sender's own past sends that went to this receiver, $c(j,k)/a(j)$: inertia, the tendency to keep writing to the same people. `frrecsnd` is the fraction of the sender's past receipts that came from this receiver, $c(k,j)/b(j)$: reciprocity, the tendency to answer the people who wrote to you. As long as the sender has sent nothing (has received nothing), the denominator is zero and the statistic is not 0 but $1/(n-1)$ for every receiver; once the sender has sent (received) at least once it is the fraction, 0 for receivers never sent to (never received from). This is the convention of `relevent`.

5.4 Recency effects

The recency effects measure how *recently*, not how often, a partner was met. For `rsndsnd` the actors to whom the sender j has sent are ordered by the time of the last such event, the most recent first. The statistic of the pair (j,k) is the reciprocal of the rank of k in this list: 1 for the receiver of the sender's last event, 1/2 for the receiver before, and so on; it is 0 if j has never

sent to k. A repeated send moves the receiver to the front. `rrecsnd` is the same for the actors who have sent to j: the reciprocal rank of k among the senders to j, the most recent first; it is a recency-based reciprocity effect. A sender who contacted a partner once, very recently, scores as high on `rsndsnd` for that partner as one who contacted them constantly and most recently: only the order matters, not the count. This is the difference from `frpsndsnd` and `frrecsnd`, which measure how often.

5.5 Covariates

`covsnd(varlist)` gives a pair the value z_j of a variable of the sender, `covrec(varlist)` the value z_k of the receiver, `covint(varlist)` the sum $z_j + z_k$; each variable has its own coefficient. `covint` is therefore an exact function of `covsnd` and `covrec` of the same variable, and these three cannot be fitted together (Section 4.9). The coefficient of `covsnd(x)` is the log rate ratio for a sender whose x is one unit larger; it is an *ego* effect. `covrec` is an *alter* effect. The values do not change in time. `covevent(netlist)` gives a pair the tie value w_{jk} of a network, one coefficient per network; the cell from sender to receiver is used, so the covariate can be directed.

5.6 Triadic and shared-partner effects

These effects count the paths through a third actor in the history, weighted by the number of events. For the pair (j, k) and an actor l other than j and k:

- `otpsnd`, outbound two-paths: j has sent to l and l has sent to k; the statistic is $\sum_l \min\{c(j,l), c(l,k)\}$, the smaller of the two counts, summed over l.
- `itpsnd`, inbound two-paths: the same count for the paths from the receiver through some l to the sender, $\sum_l \min\{c(k,l), c(l,j)\}$.
- `ospsnd`, outbound shared partners: j and k have both sent to l, $\sum_l \min\{c(j,l), c(k,l)\}$.
- `ispsnd`, inbound shared partners: both have received from l, $\sum_l \min\{c(l,j), c(l,k)\}$.

`ospsnd` and `relevent`. `otpsnd`, `itpsnd` and `ispsnd` are the statistics of `relevent`. `ospsnd` is the intended *symmetric* statistic, as above: the value for (j, k) equals the value for (k, j). The statistic computed by `relevent` 1.2-1 is not symmetric: for a pair of actors it counts the shared out-partners twice for one direction of the pair and not at all for the other, depending on the order in which the actors first appear as senders. Estimates of `nwrem` for this effect therefore differ from those of `relevent`; the deviation is intended. Chapter 9 states how the effect was checked.

5.7 Fixed effects

`fesnd`, `ferec` and `feint` give each actor, except the first one, a coefficient of its own: an indicator that the actor is the sender (`fesnd`), the receiver (`ferec`), or either of the two (`feint`, one coefficient for the two roles). The first actor, in the actor order of the network, is the reference; the coefficients are named `fesnd_2`, ..., `fesnd_`n by the position of the actor. An actor that never sends (never receives) has a coefficient that tends to minus infinity, and the note of Section 4.9 appears.

5.8 Participation shifts

Participation shifts describe how the next event follows the previous one. Let the previous event be $A \to B$ (sender A, receiver B). The six statistics are indicators: `psabba` is 1 for the reply $B \to A$; `psabby` for B sending to someone other than A; `psabxa` for someone other than A and B sending to A; `psabxb` for someone other than A and B sending to B; `psabxy` for an event in which

neither actor is A or B; and `psabay` for A sending again, to someone other than B. They are zero at the first event. Together with the repetition of the previous event, $A \to B$ again, which is the reference category and has no shift of its own, the six sets of pairs partition the ordered pairs at every event. `relevent` defines seven further shifts (those with undefined targets, for example `PSAB-BO`) whose coefficients are always zero; `nwrem` does not offer them.

Chapter 6

Stored results and comparing models

6.1 Stored results

`nwrem` is an estimation command. It stores the following in `e()`.

Result	Contents
Scalars	
`e(N)`	Number of events, including the conditioned events.
`e(nodes)`	Number of actors.
`e(ll)`	Log likelihood at the estimates; with `postmode` without the prior.
`e(ll_0)`	Log likelihood of the model without effects (Section 2.7). Not stored with `interval` when a risk set, `entry()` or `exit()` is used.
`e(df_m)`	Number of estimated coefficients (the omitted ones are not counted).
`e(deviance)`	-2 `e(ll)`.
`e(aic)`, `e(aicc)`, `e(bic)`	Information criteria: AIC = deviance $+ 2k$, AICc = AIC $+ 2k(k+1)/(\nu - k - 1)$ (stored only if $\nu - k - 1 > 0$), BIC = deviance $+ k \ln \nu$, with k the number of coefficients and ν the number of events that enter the likelihood ($\nu =$`e(N)`$-$`e(N_cond)`, and one less with `interval`).
`e(N_cond)`	Number of events that only build the history (`condition()`).
`e(starttime)`, `e(endtime)`	Observation window with `interval`; `e(endtime)` only if `endtime()` was given.
`e(priormean)`, `e(priorscale)`, `e(priordf)`	The prior of `postmode`.
Macros	
`e(cmd)`	`nwrem`
`e(version)`	Release of `nwrem` that produced the results.
`e(title)`	"Relational event model (" likelihood and estimator ")".
`e(lik)`	`ordinal` or `interval`.
`e(depvar)`	Name of the event network.
`e(cmdline)`	The command line of the fit; `estat` (Chapter 7) sets the model up again from it.
`e(covmatch)`	`position` or `label`; stored only with `covevent()` or `riskset()`.
`e(estat_cmd)`	`nwrem_estat`, the program that handles `estat`.
`e(unidentified)`	Effects without a usable standard error: left out because they are exactly collinear with others (coefficient 0), or with a standard error above 100; empty when all are identified.
Matrices	
`e(b)`	Coefficient vector.
`e(V)`	Covariance matrix, the inverse of minus the Hessian (the observed information); with `postmode`, of the log posterior.

`nwrem version` stores `r(version)` and `r(date)` (the release), `r(core)` (the core version), `r(plugin)` (the plugin file in use), `r(reference)` (the version of `relevent` that the results were compared with) and `r(license)`.

```
. nwrem version
nwrem version 1.7.0, 21nov2026
  core package nwcommands : installed (development copy, no distribution date)
```

```
  native plugin           : ~/Library/Application Support/Stata/ado/plus/r/rem_sim_macos.plugin
> (ships with the package, no separate version number)
  compared with           : R relevent 1.2-1
  license                 : GNU GPL-3
```

A listing of the stored results. The model of Section 3.3 with four effects gives the following; `e(ll)` holds the value that the header prints in a rounded form.

```
. ereturn list

scalars:
                  e(N) =  631
              e(nodes) =  144
                 e(ll) =  -5531.783319413064
           e(deviance) =  11063.56663882613
               e(ll_0) =  -6267.507153726425
               e(df_m) =  4
             e(N_cond) =  0
                e(aic) =  11071.56663882613
               e(aicc) =  11071.63053658971
                e(bic) =  11089.3558622763

macros:
             e(depvar) : "mail"
                e(lik) : "ordinal"
              e(title) : "Relational event model (ordinal partial likelihood, .."
            e(version) : "1.7.0"
          e(estat_cmd) : "nwrem_estat"
            e(cmdline) : "nwrem mail, nodsnd nidrec frpsndsnd frrecsnd"
                e(cmd) : "nwrem"
         e(properties) : "b V"

matrices:
                  e(b) :  1 x 4
                  e(V) :  4 x 4
```

6.2 Information criteria and likelihood-ratio tests

Because `e(ll)`, `e(df_m)` and `e(N)` are stored, `estimates stats` and `lrtest` work. The log likelihood is a proper log likelihood of the ordinal model (or of the waiting-time model), so models for the *same* events with the same likelihood and the same conditioning can be compared by their information criteria, and nested models by a likelihood-ratio test. Models with the ordinal and with the waiting-time likelihood, or with different values of `condition()`, `riskset()` or a different set of actors, are not comparable, because they describe different data. `estimates stats` computes the BIC with `e(N)`, which counts conditioned events; without `condition()` it agrees with `e(bic)`.

```
. quietly nwrem mail, nodsnd nidrec

. estimates store m1

. quietly nwrem mail, frpsndsnd frrecsnd

. estimates store m2

. quietly nwrem mail, nodsnd nidrec frpsndsnd frrecsnd

. estimates store m3
```

```
. estimates stats m1 m2 m3

Akaike's information criterion and Bayesian information criterion

------------------------------------------------------------------------------
       Model |          N   ll(null)  ll(model)      df        AIC        BIC
-------------+----------------------------------------------------------------
          m1 |        631  -6267.507  -5992.517       2   11989.03   11997.93
          m2 |        631  -6267.507  -5733.442       2   11470.88   11479.78
          m3 |        631  -6267.507  -5531.783       4   11071.57   11089.36
------------------------------------------------------------------------------
Note: BIC uses N = number of observations. See [R] BIC note.

. lrtest m3 m1

Likelihood-ratio test
Assumption: m1 nested within m3

 LR chi2(2) = 921.47
Prob > chi2 = 0.0000

. lrtest m3 m2

Likelihood-ratio test
Assumption: m2 nested within m3

 LR chi2(2) = 403.32
Prob > chi2 = 0.0000
```

In this example the model with both the activity effects and the history effects is much better than either of the two submodels: the log likelihood rises from −5,992.5 (activity) and −5,733.4 (history) to −5,531.8.

The likelihood-ratio test against the null model. `e(ll_0)` is the log likelihood without effects, so $2\{$`e(ll)` − `e(ll_0)`$\}$ is the statistic of the test of all coefficients together, on `e(df_m)` degrees of freedom:

```
. estimates stats

Akaike's information criterion and Bayesian information criterion

------------------------------------------------------------------------------
       Model |          N   ll(null)  ll(model)      df        AIC        BIC
-------------+----------------------------------------------------------------
           . |        631  -6267.507  -5531.783       4   11071.57   11089.36
------------------------------------------------------------------------------
Note: BIC uses N = number of observations. See [R] BIC note.

. display "likelihood ratio against the model without effects: " %7.2f 2*(e(ll) -
> e(ll_0)) " on " e(df_m) " d.f."
likelihood ratio against the model without effects: 1471.45 on 4 d.f.
```

6.3 Replaying, the confidence level and reproducibility

`nwrem` without arguments redisplays the last fit; `level()` changes the confidence level of the displayed intervals without refitting (and shows them); `ci` and `noci` work in the replay as in the fit.

```
. nwrem, level(90)
```

```
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                          Actors: 144
Events: 631                            Log likelihood: -5.53e+03
------------------------------------------------------------
------------------------------------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|     [90% conf. interval]
-------------+----------------------------------------------------------------
      nodsnd |   14.36435   1.616692     8.89   0.000     11.70512    17.02357
      nidrec |   22.01803   1.182936    18.61   0.000     20.07228    23.96379
   frpsndsnd |   2.296269   .1856076    12.37   0.000     1.990972    2.601566
    frrecsnd |   3.821268   .1618693    23.61   0.000     3.555017    4.087519
------------------------------------------------------------------------------
```

The fit is deterministic. It starts at zero and takes Newton steps; the random numbers are used only when the iteration fails to reach a maximum. Two fits with different seeds therefore give identical results:

```
-------------+----------------------------------------
      nodsnd |   14.36435   1.616692     8.89   0.000
      nidrec |   22.01803   1.182936    18.61   0.000
   frpsndsnd |   2.296269   .1856076    12.37   0.000
    frrecsnd |   3.821268   .1618693    23.61   0.000
------------------------------------------------------

. matrix b2 = e(b)

. display "largest difference between the two fits: " %9.2e mreldif(b1, b2)
largest difference between the two fits:  0.00e+00
```

Chapter 7

Diagnostics and graphs after the fit

After a fit, three `estat` subcommands check how well the model reproduces the data: `estat rank`, `estat actors` and `estat time`. They are extensions of `nwrem`: `relevent` has no such methods. Each one evaluates the fitted linear predictor of *every* dyad of the risk set at *every* event, at the coefficients in `e(b)`, and summarises the result in a table and a graph. The estimation results and the data in memory are not changed. All examples of this chapter use the model of Section 8.1 with the activity and the history effects, fitted to the e-mails of the first five days (631 events, 144 actors, 20,592 ordered pairs); the do-file is `ex10_estat.do`.

```
. nwrem mail, nodsnd nidrec frpsndsnd frrecsnd
```

7.1 What the diagnostics need

The data as at the fit. `estat` sets the model up again exactly as for the fit: it runs the data preparation of the command line that `nwrem` stores in `e(cmdline)` (`e(estat_cmd)` names the program that handles `estat`). The event network and the covariate variables or networks used in the fit must therefore still be in memory, unchanged, when `estat` is called. Nothing needs to be refitted. Events that only build the history (`condition()`) are left out of all quantities. Both likelihoods, `riskset()`, `entry()`, `exit()` and the fixed effects are supported.

The effort is one evaluation of the likelihood over all dyads of every risk set, so it grows with the number of events times the square of the number of actors. The evaluation runs in the plugin that ships with the package version; an older plugin stops with return code 3495 and a message, and the Linux and Windows plugins of the package are built for this purpose together with the macOS plugin.

The graphs need the core package `nwcommands` in version 3.29.0 or later; with an older core the graph stops with return code 199 and a message, before anything is computed. The tables (`noplot`) work with any core.

7.2 Graph options

The three subcommands draw their graph by default; `noplot` shows the table only. The options for the graph are the same for all three.

Option	Effect
`noplot`	Table only. `name()`, `saving()`, `export()` and graph options are errors with `noplot` (return code 198).
`name(`*name*`)`	Name of the graph in memory. The defaults are `rank`, `actors` and `time`; an existing graph of that name is replaced. A trailing `, replace` is accepted and ignored.
`saving(`*file*`)`	Saves the graph as a `.gph` file (the extension is added; an existing file is replaced).
`export(`*file*`[, `*opts*`])`	Exports the graph with `graph export`: the extension sets the format (`.png`, `.pdf`, `.svg`, ...), an existing file is replaced, and the options are those of `graph export`, for example `width()`.
all other options	Passed to `twoway` (to `histogram` for `plot(hist)`, to `graph combine` for the two panels of `estat actors`) after the defaults, so they override them: `scheme()`, `title()`, `xlabel()`, `xsize()`, and so on.

The default titles, subtitles and axis titles are black; `title("")` removes the title, which is what the figures of this manual do, because the caption says it. A directory that does not exist in the path of `saving()` or `export()` stops the command with return code 603. One call can name, save and export a graph, and take a graph option:

```
. estat rank, name(rk) saving(rank) export(rank.png, width(1200)) scheme(s2color)
```

The options that are given without the graph are refused:

```
. capture noisily estat rank, noplot name(x)
estat rank: name(), saving(), export() and graph options need plot.

. display "return code " _rc
return code 198

. capture noisily estat rank, noplot title("x")
estat rank: name(), saving(), export() and graph options need plot.

. display "return code " _rc
return code 198
```

7.3 The rank of the observed dyad: `estat rank`

For every event, `estat rank` ranks all dyads of the risk set by the fitted linear predictor (rank 1 is the pair that the model finds most likely) and records the rank of the dyad that was observed. Dyads with exactly the same predictor share the mid-rank. This is the standard check of the predictive accuracy of a relational event model.

```
. estat rank, noplot

Rank diagnostic of the fitted model (an extension of nwrem; relevent has no such
method).
Rank of the observed dyad among all dyads of the risk set at the time of the
event, by the fitted linear predictor (1 = most likely; tied dyads share the
mid-rank).
------------------------------------------------------------------------
Events evaluated                                          631
Events left out (condition())                               0
Mean size of the risk set                             20592.00
------------------------------------------------------------------------
                                  observed fitted model no effects
Mean rank                          2287.27               10296.50
Median rank                         546.00
Mean percentile rank                0.1111                 0.5000
Share of events in the top 10       0.0349       0.1437    0.0005
```

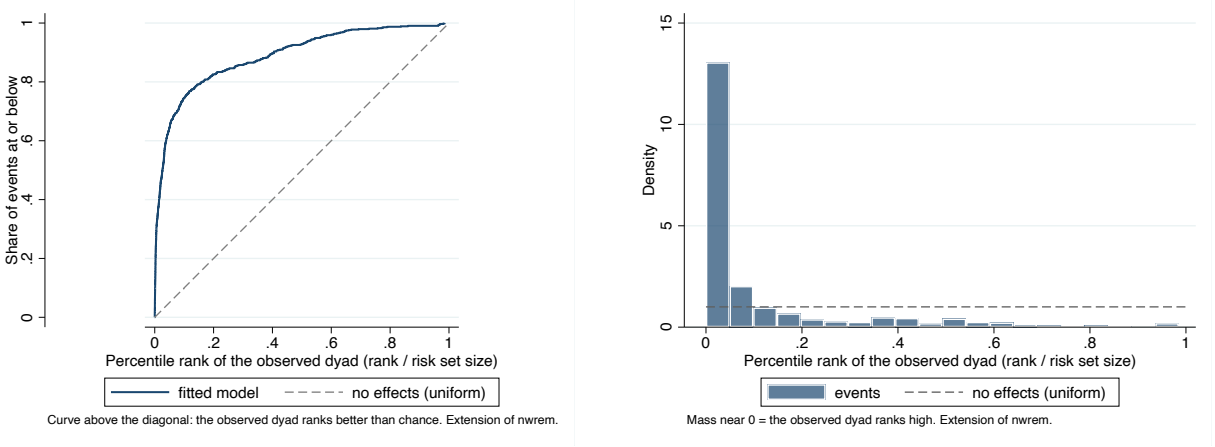


Figure 7.1: `estat rank` (left) and `estat rank, plot(hist) bins(20)` (right) for the model of this chapter. Left: the cumulative distribution of the percentile ranks of the observed dyads; a curve above the diagonal means that the observed dyads rank better than chance. Right: their histogram against the uniform density, which is what a model without effects gives. The titles were removed with `title("")`.

```
Mean log prob. of observed dyad       -8.7667                 -9.9327
-------------------------------------------------------------------------
Percentile rank = rank / size of the risk set. Fitted model: top-10 share
expected under the model (the summed probability of the 10 most likely dyads).
No effects: every dyad equally likely.
```

The table gives the mean and the median rank, the mean percentile rank (the rank divided by the size of the risk set), the share of events whose dyad is in the top k (by default 10; `topk()` changes k), and the mean log probability of the observed dyad. Each is shown next to its value for a model without effects, in which every dyad is equally likely (mean percentile rank 0.5). The share of top-k hits is also given as the fitted model expects it, the summed probability of its k most likely dyads; ties at the border of the top k count fractionally.

Here the observed dyad has a mean rank of 2,287 among 20,592 pairs, a mean percentile rank of 0.111 against 0.5 for the model without effects, and a median rank of 546. In 3.5 percent of the events the observed pair is among the ten most likely pairs (22 of 631), against 0.05 percent for a model without effects. The model itself expects 14.4 percent: it ranks the observed pair at the very top less often than its own probabilities say it should. The mean log probability of the observed dyad rises from −9.93 to −8.77. With `topk(5)` the shares are 2.5, 10.7 and 0.02 percent.

The default graph is the cumulative distribution of the percentile ranks (Figure 7.1, left). With `plot(hist)` it is the histogram of the percentile ranks, with `bins()` bars (default 10), drawn against the uniform density (right). Here the mass is at the left end: for about two events in three the observed pair is within the best 5 percent of the pairs (the first bar). The results are stored in `r()`: `r(mean_rank)`, `r(median_rank)`, `r(mean_pct_rank)`, `r(hit)`, `r(hit_model)`, `r(hit_null)`, `r(mean_logp)`, `r(mean_logp_null)`, `r(mean_risksize)`, `r(N)` and `r(topk)`.

7.4 Activity of the actors: `estat actors`

`estat actors` compares, for every actor, the number of events observed as sender and as receiver with the number that the model expects: the sum over the events of the model probability that the actor is the sender (receiver) of the event; under the waiting-time likelihood, the summed rate integrals.

```
. estat actors, role(sender) noplot
```

```
Observed and expected events by actor (an extension of nwrem; relevent has no
such method).
Expected = sum over the events of the model probability that the actor is the
sender (receiver); under the waiting-time likelihood, the summed rate integrals.
------------------------------------------------------------------
Actors                                      144  Events         631
------------------------------------------------------------------
                                Sender                Receiver
Correlation observed, expected    0.4068                0.8620
Sum of squared Pearson residuals    702.26                471.32
------------------------------------------------------------------
Pearson residual = (observed - expected) / sqrt(expected); the sums describe the
fit, they are not a test (events are not independent draws given the effects).
```

The table reports, for each role, the correlation of the observed and the expected counts and the sum of the squared Pearson residuals, $(\text{observed} - \text{expected})/\sqrt{\text{expected}}$. The sums describe the fit; they are not a test, because the events are not independent draws given the effects. Here the receivers are reproduced well (correlation 0.86) and the senders poorly (0.41): with these effects the model reproduces who is written to much better than who writes. The counts of all actors are in the matrix `r(actors)`, one row per actor in the order of the network, with the columns `obs_sender`, `exp_sender`, `obs_receiver` and `exp_receiver`; `list` lists them. The first rows:

```
. estat actors, role(sender) noplot

Observed and expected events by actor (an extension of nwrem; relevent has no
such method).
Expected = sum over the events of the model probability that the actor is the
sender (receiver); under the waiting-time likelihood, the summed rate integrals.
------------------------------------------------------------------
Actors                                      144  Events         631
------------------------------------------------------------------
                                Sender                Receiver
Correlation observed, expected    0.4068                0.8620
Sum of squared Pearson residuals    702.26                471.32
------------------------------------------------------------------
Pearson residual = (observed - expected) / sqrt(expected); the sums describe the
fit, they are not a test (events are not independent draws given the effects).

. matrix A = r(actors)

. matrix B = A[1..6,1..4]

. matrix list B, format(%9.2f)

B[6,4]
      obs_sender    exp_sender  obs_receiver  exp_receiver
r1         10.00          3.83         11.00          4.97
r2         14.00          4.90          6.00          4.92
r3          5.00          3.19          0.00          2.60
r4          4.00          7.34          2.00          3.07
r5          7.00          3.71          5.00          2.94
r6          5.00         13.34          4.00          3.29
```

After a fit with `interval` the expected counts are summed rate integrals of a model without a baseline rate, so their total need not equal the observed number of events, and a level far off says that the effects cannot reproduce the overall event rate; the table says so in a note under it. Judge such a fit by the correlations, not by the levels. For the waiting-time fit of Section 7.5:

```
. estat actors, role(sender) noplot

Observed and expected events by actor (an extension of nwrem; relevent has no
```

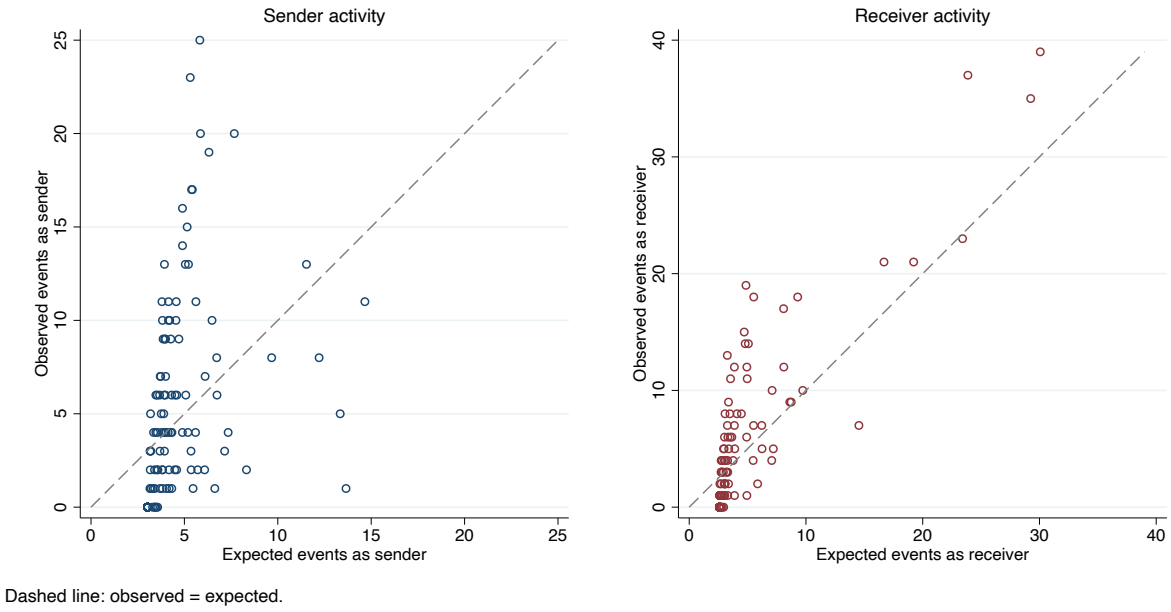


Figure 7.2: `estat actors` for the model of this chapter: expected (horizontal) and observed (vertical) number of events per actor, as sender (left) and as receiver (right). The dashed line is the 45 degree line, on which observed and expected counts agree. Both axes of a panel start at 0 and have the same range, and the panels are square.

```
such method).
Expected = sum over the events of the model probability that the actor is the
sender (receiver); under the waiting-time likelihood, the summed rate integrals.
-------------------------------------------------------------------------
Actors                                        144  Events         628
-------------------------------------------------------------------------
                                  Sender                Receiver
Correlation observed, expected    0.0052                0.7303
Sum of squared Pearson residuals    842.54              678.05
-------------------------------------------------------------------------
Pearson residual = (observed - expected) / sqrt(expected); the sums describe the
fit, they are not a test (events are not independent draws given the effects).
Note: waiting-time likelihood. The expected totals are summed rate integrals;
      the model has no baseline rate, so they need not equal the observed
      totals, and a level far off says the effects cannot reproduce the overall
      event rate (compare the correlations, not the levels).
```

The graph is a scatter plot with the 45 degree line (Figure 7.2). By default there are two panels, one for the senders and one for the receivers; `role(sender)` or `role(receiver)` draws one panel only in the graph (the table and `r()` always hold both roles), and `labels` labels the points with the names of the actors. Both axes of a panel run from 0 to the same maximum and the panel is square, so that the 45 degree line is the diagonal; `noequal` lets the axes have different ranges. The footnote of the graph is one short line, so that the graph fits a width of 3.6 inches. The results in `r()` are `r(n_actors)`, `r(corr_sender)`, `r(corr_receiver)`, `r(chi2_sender)`, `r(chi2_receiver)` and the matrix `r(actors)`.

7.5 The course over time: `estat time`

`estat time` compares observed and expected quantities over the sequence of events, in `bins()` bins of equally many consecutive events (deciles by default; at most the number of events). What

is compared depends on the likelihood.

Ordinal likelihood. The ordinal likelihood has no time scale. For each bin the table shows the number of events whose observed dyad is among the k most likely dyads of the risk set (`topk()`, default 10; ties are split) and the number expected under the model.

```
. estat time, bins(8) noplot

Observed versus expected over time (an extension of nwrem; relevent has no such
method).
Ordinal likelihood (it has no time scale): observed = number of events whose
observed dyad is among the 10 most likely dyads of the risk set (ties split);
expected = sum of the probabilities of the 10 most likely dyads.
------------------------------------------------------------------
   bin   events     time from    time to     observed    expected
     1        78         3.235       3.331       4.000      18.799
     2        79         3.331       3.386       4.000      10.775
     3        79         3.386       3.488       4.000      10.810
     4        79         3.489       3.568       3.000      11.036
     5        79         3.568       4.337       4.000       9.272
     6        79         4.341       4.463       0.000       9.882
     7        79         4.463        4.54       2.000       9.848
     8        79          4.54       4.989       1.000      10.267
------------------------------------------------------------------
total                                            22.000      90.689
```

The bins are of 78 or 79 events. The observed number of top-10 hits is small and falls over the period (4, 4, 4, 3, 4, 0, 2, 1), the expected one is about ten per bin after the first, where it is 18.8. The model finds the observed pair among its best ten less often than it expects, in every bin and increasingly so toward the end. The bin boundaries and the counts are in the matrix `r(bins)`, one row per bin, with the columns `events`, `time_from`, `time_to`, `observed` and `expected`; `r(n_bins)`, `r(topk)`, `r(obs_total)` and `r(exp_total)` hold the number of bins, k and the totals.

```
. matrix list r(bins), format(%9.2f)

r(bins)[8,5]
       events  time_from    time_to   observed   expected
r1      78.00       3.24       3.33       4.00      18.80
r2      79.00       3.33       3.39       4.00      10.77
r3      79.00       3.39       3.49       4.00      10.81
r4      79.00       3.49       3.57       3.00      11.04
r5      79.00       3.57       4.34       4.00       9.27
r6      79.00       4.34       4.46       0.00       9.88
r7      79.00       4.46       4.54       2.00       9.85
r8      79.00       4.54       4.99       1.00      10.27
```

Waiting-time likelihood. With `interval` the bins show the number of events and the integrated rate of the model over the intervals that end with those events; the last bin also carries the final interval up to `endtime()`. The model has no baseline rate, so a level of the expected curve that is far from the observed one says that the effects cannot reproduce the overall rate of events. The example uses the fit of Section 8.3 (the ties of the times removed, the time in units in which the null model has the rate one):

```
. nwrem mail2, nodsnd nidrec frpsndsnd interval starttime(0) endtime(0.0305)
. estat time, bins(8) noplot
```

```
. estat time, bins(8) noplot
```

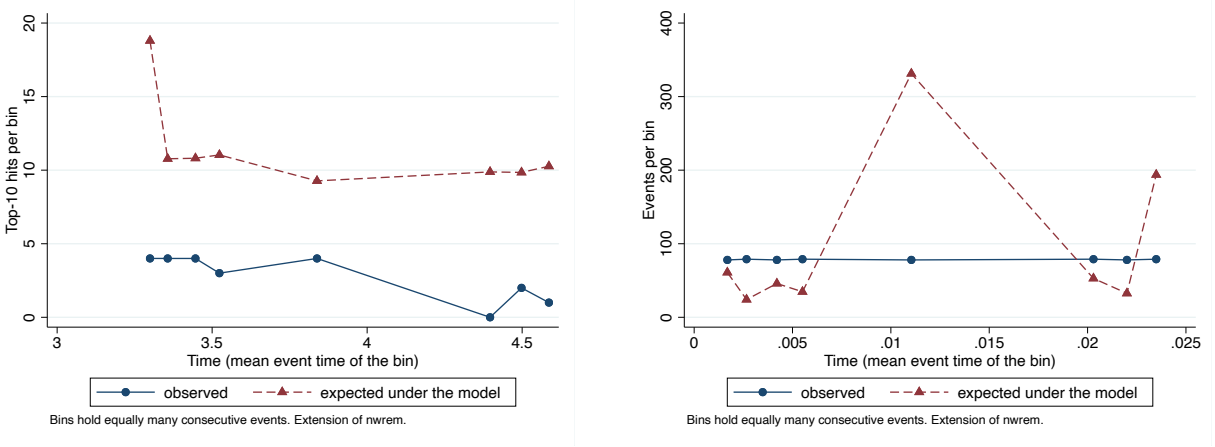


Figure 7.3: `estat time, bins(8)`. Left: after the ordinal fit, the number of top-10 hits per bin, observed and expected under the model (horizontal axis: mean event time of the bin, in days). Right: after the waiting-time fit, the number of events per bin, observed and expected (the time is in the unit of the fit).

```
Observed versus expected over time (an extension of nwrem; relevent has no such
method).
Waiting-time likelihood: observed = number of events in the bin of consecutive
events; expected = sum of the integrated rates of the fitted model over the
intervals that end with those events (the last bin also carries the final
interval up to endtime()). The model has no baseline rate, so a level that is
far off says the effects cannot reproduce the overall event rate.
------------------------------------------------------------------------
   bin    events     time from     time to     observed     expected
     1        78     .0005973      .00222       78.000       60.981
     2        79      .002222     .003158       79.000       24.134
     3        78      .003166     .004903       78.000       45.947
     4        79      .004933     .006242       79.000       34.705
     5        78      .006246      .01933       78.000      331.261
     6        79       .01933       .0214       79.000       52.888
     7        78       .02141       .0227       78.000       32.556
     8        79        .0227      .03031       79.000      193.788
------------------------------------------------------------------------
total                                          628.000      776.261

. quietly estat time, bins(8) name(tmi) export(../fig/es_timeint.pdf) title("") xs
> ize(4.2) ysize(3.2)
```

With 628 events observed and 776 expected, the level is about right overall, but the expected counts differ strongly from bin to bin: the fifth bin, which contains a long interval without e-mails (about 0.8 days, from day 3.57 to day 4.34 in the ordinal table above), carries 331 expected events against 78 observed, because the effects have no way to make the rate drop in a quiet period. The graph (Figure 7.3) plots the observed and the expected quantity against the mean event time of the bin, with the bins in order.

Chapter 8

Worked examples

All examples use the e-mail log of Section 3.2. The listings are the output of the do-files in `manual/ex` (named in each section). The examples show how the options work and how to read the output; they are not a substantive analysis of the company.

8.1 History and activity

ex02_mail.do. The first model of Section 3.3 used the activity of the actors. The second asks about the history of the pair: do people write to those they wrote to before (`frpsndsnd`), and to those who wrote to them (`frrecsnd`)?

```
. nwrem mail, frpsndsnd frrecsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                           Actors: 144
Events: 631                             Log likelihood: -5.73e+03
------------------------------------------------------------
-----------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
   frpsndsnd |   3.166245   .2084019    15.19   0.000
    frrecsnd |    3.48016    .197031    17.66   0.000
-----------------------------------------------------

. display "events: " e(N) ", actors: " e(nodes) ", log likelihood: " %9.2f e(ll)
events: 631, actors: 144, log likelihood:  -5733.44
```

If another 10 percentage points of a sender's e-mails had gone to a given person, the rate of the next e-mail going to that person again is multiplied by $\exp(0.317) \approx 1.4$. Replying also raises the rate: for each 10 percentage points of the sender's earlier receipts that came from a given person, the rate of writing to that person is multiplied by $\exp(0.348) \approx 1.4$ as well. The model fits far better than the activity model: the log likelihood is $-5{,}733$ against $-5{,}993$ for the same number of coefficients. Both sets of effects together:

```
. nwrem mail, nodsnd nidrec frpsndsnd frrecsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                           Actors: 144
Events: 631                             Log likelihood: -5.53e+03
------------------------------------------------------------
-----------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   14.36435   1.616692     8.89   0.000
      nidrec |   22.01803   1.182936    18.61   0.000
```

```
   frpsndsnd |   2.296269   .1856076    12.37   0.000
    frrecsnd |   3.821268   .1618693    23.61   0.000
------------------------------------------------------
```

Once the activity of the actors is in the model, the inertia effect shrinks (3.17 to 2.30) and the activity effects shrink a little (`nodsnd` from 15.6 to 14.4, `nidrec` from 25.0 to 22.0), while the reciprocity of receipt grows (3.48 to 3.82). All four effects remain clearly different from zero, and the log likelihood rises to −5,531.8. Part of what looked like "who writes to whom" in the second model was activity, and part of what looked like "busy people stay busy" in the first was people writing to those they always write to. This is the usual reason for fitting several effects together.

8.2 Covariates of actors and of pairs

ex03_cov.do. The attributes of the employees come from the reporting hierarchy. `nwload mail, xvars` loads a dataset with one row per actor and the label in `_nwnode`; the variables of the second data set are merged to it by label, and the original order is restored because the order of the rows after a merge is not guaranteed.

```
. gen str4 label = _nwnode

. gen long row = _n

. merge m:1 label using 'staff', keep(master match) nogenerate

    Result                      Number of obs
    -----------------------------------------
    Not matched                             0
    Matched                               144
    -----------------------------------------

. sort row

. gen byte junior = (level >= 4)

. list _nwnode level junior supervisor division in 1/4

     +-------------------------------------------------+
     | _nwnode   level   junior   superv~r   division |
     |-------------------------------------------------|
  1. |    E102       2        0        E36        E36 |
  2. |    E103       2        0        E85        E85 |
  3. |    E104       1        0        E86       E104 |
  4. |    E105       2        0       E104       E104 |
     +-------------------------------------------------+

. tabulate level

   Steps to |
 the CEO in |
        the |
  reporting |
  hierarchy |
  (0 = CEO) |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |          1        0.69        0.69
          1 |         11        7.64        8.33
          2 |         59       40.97       49.31
          3 |         29       20.14       69.44
          4 |         43       29.86       99.31
          5 |          1        0.69      100.00
------------+-----------------------------------
```

```
      Total |        144      100.00
```

`level` is the number of steps to the chief executive (0 is the CEO); `junior` is an indicator that the level is 4 or more. The two options `covsnd(level)` and `covrec(level)` ask whether people lower in the hierarchy send and receive less, over and above the history of the pair:

```
. nwrem mail, frpsndsnd frrecsnd covsnd(level) covrec(level)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                           Actors: 144
Events: 631                             Log likelihood: -5.71e+03
------------------------------------------------------------
-----------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
   frpsndsnd |   3.114259   .2134847    14.59   0.000
    frrecsnd |   3.462585   .2013303    17.20   0.000
covsnd_level |  -.2328724   .0401241    -5.80   0.000
covrec_level |  -.1209538   .0397192    -3.05   0.002
-----------------------------------------------------
```

Each step down the hierarchy multiplies the rate of sending by $\exp(-0.233) \approx 0.79$ and the rate of being written to by $\exp(-0.121) \approx 0.89$. With `covint(level)` one coefficient is shared by the sender and the receiver ($z_j + z_k$):

```
. nwrem mail, frpsndsnd covint(level)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                           Actors: 144
Events: 631                             Log likelihood: -5.85e+03
------------------------------------------------------------
-----------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
   frpsndsnd |   5.131241   .1148863    44.66   0.000
covint_level |  -.2045331   .0277353    -7.37   0.000
-----------------------------------------------------
```

Several covariates in one option. Every variable gets a coefficient. With the continuous level and the indicator together, the junior employees send less (coefficient -0.417), and the level, over and above, adds little for the receiver:

```
. nwrem mail, frpsndsnd covsnd(level junior) covrec(level junior)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                           Actors: 144
Events: 631                             Log likelihood: -5.84e+03
------------------------------------------------------------
-----------------------------------------------------
         mail | Coefficient  Std. err.      z    P>|z|
--------------+--------------------------------------
    frpsndsnd |   5.161763    .116121    44.45   0.000
 covsnd_level |  -.1370062   .0670139    -2.04   0.041
covsnd_junior |  -.4172811   .1631169    -2.56   0.011
 covrec_level |  -.0648092   .0708551    -0.91   0.360
covrec_junior |  -.2031686    .165776    -1.23   0.220
-----------------------------------------------------
```

Pair covariates. A covariate of the pair is a network with the same actors. Here the network `samesup` has a tie from i to j when the two have the same direct supervisor, and `samediv` when they belong to the same division. Both are symmetric, and are built from a matrix (see Section 4.8). The first fit has one covariate network; the second has both, with one coefficient each.

```
. nwrem mail, frpsndsnd covevent(samesup)
Note: covevent(samesup) matched to the event network by position (labels agree).
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                          Actors: 144
Events: 631                            Log likelihood: -5.86e+03
------------------------------------------------------------
------------------------------------------------------------
            mail | Coefficient  Std. err.      z    P>|z|
-----------------+------------------------------------------
       frpsndsnd |    5.04925   .1169899    43.16   0.000
covevent_samesup |   .8608735   .1256691     6.85   0.000
------------------------------------------------------------

. estimates store one

. nwrem mail, frpsndsnd covevent(samesup samediv)
Note: covevent(samesup) matched to the event network by position (labels agree).
Note: covevent(samediv) matched to the event network by position (labels agree).
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                          Actors: 144
Events: 631                            Log likelihood: -5.85e+03
------------------------------------------------------------
------------------------------------------------------------
            mail | Coefficient  Std. err.      z    P>|z|
-----------------+------------------------------------------
       frpsndsnd |    5.03107   .1172784    42.90   0.000
covevent_samesup |   .6976388   .1343369     5.19   0.000
covevent_samediv |    .286555   .0862983     3.32   0.001
------------------------------------------------------------

. estimates store two

. estimates table one two, b(%9.3f) stats(ll N)

--------------------------------------
    Variable |    one         two
-------------+------------------------
   frpsndsnd |     5.049       5.031
covevent_s~p |     0.861       0.698
covevent_s~v |                 0.287
-------------+------------------------
```

Pairs that share the supervisor have $\exp(0.861) \approx 2.4$ times the rate of other pairs at the same history; adding the division reduces this to $\exp(0.698) \approx 2.0$ and gives pairs of the same division a further factor of $\exp(0.287) \approx 1.3$. (The names in the table are shortened by Stata's display; the stored names are `covevent_samesup` and `covevent_samediv`.)

Collinear covariates. `covint` is the sum of `covsnd` and `covrec` of the same variable, so the three cannot be estimated together. The last one in the order of Table 5.1 is left out:

```
. nwrem mail, covsnd(level) covrec(level) covint(level)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail                          Actors: 144
Events: 631                            Log likelihood: -6.23e+03
```

```
------------------------------------------------------------
------------------------------------------------------
        mail | Coefficient  Std. err.      z    P>|z|
-------------+----------------------------------------
covsnd_level |  -.2516265    .039796    -6.32   0.000
covrec_level |  -.1969078   .0395808    -4.97   0.000
covint_level |          0  (omitted)
------------------------------------------------------
Note: no usable standard error for covint_level.
      These effects are not identified: typically an effect is an exact function
      of others in the model (the total-degree share ntdegsnd is half the sum of
      nodsnd and nidsnd). Its estimate is arbitrary; drop one of the effects.
```

The note under the table is the general note of the command (Section 4.9); the coefficient of the omitted effect is 0. A missing value of a covariate stops the fit:

```
. replace level = . in 1
(1 real change made, 1 to missing)

. capture noisily nwrem mail, frpsndsnd covsnd(level)
covsnd(level): 1 actor(s) of mail have a missing value; the covariate must be
defined for every actor.

. display "return code: " _rc
return code: 416
```

8.3 The waiting-time likelihood

ex04_interval.do. Some e-mails of the log carry the same time: three pairs of e-mails in the first five days. The ordinal likelihood accepts such ties, and keeps the order of the rows; the waiting-time likelihood needs strictly increasing times and refuses them:

```
. keep if t < 5
(49,602 observations deleted)

. duplicates report t

Duplicates in terms of t

--------------------------------------
   Copies | Observations       Surplus
----------+---------------------------
        1 |          625             0
        2 |            6             3
--------------------------------------

. tempfile ev

. quietly save `ev'

. nwbuild mail

. capture noisily nwrem mail, nodsnd nidrec frpsndsnd interval
nwrem: the interval likelihood needs strictly increasing event times; events at
the same time have equal times.

. display "return code: " _rc
return code: 459
```

For the comparison we drop the later e-mail of each tie (628 events remain) and fit the ordinal

model:

```
. use ‘ev’, clear
(Manufacturing company emails, 2010 (Nurek and Michalski 2020; CC0))

. duplicates drop t, force

Duplicates in terms of t

(3 observations deleted)

. nwbuild mail1

. nwrem mail1, nodsnd nidrec frpsndsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail1                          Actors: 144
Events: 628                             Log likelihood: -5.72e+03
------------------------------------------------------------
-----------------------------------------------------
       mail1 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   15.08328    1.62332     9.29   0.000
      nidrec |   16.88381   1.278413    13.21   0.000
   frpsndsnd |   4.222017   .1436294    29.40   0.000
-----------------------------------------------------

. estimates store ordinal
```

The time unit matters. With `interval` the time enters the likelihood. The model has no baseline rate: a pair with all statistics zero has the rate 1 per unit of time. With the time in days, a typical pair has a rate of about $628/(1.8 \cdot 144 \cdot 143) \approx 0.017$ per day, nowhere near 1, and the coefficients of the share effects have to make up for it:

```
. nwrem mail1, nodsnd nidrec frpsndsnd interval starttime(3.2) endtime(5)
------------------------------------------------------------
Relational event model (waiting-time likelihood, MLE)
Network: mail1                          Actors: 144
Events: 628                             Log likelihood: -1.95e+04
------------------------------------------------------------
-----------------------------------------------------
       mail1 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |  -228.9112   3.391084   -67.50   0.000
      nidrec |  -131.0217   2.498987   -52.43   0.000
   frpsndsnd |    1.75211    .247156     7.09   0.000
-----------------------------------------------------

. display "log likelihood " %9.1f e(ll) ", of the null model " %9.1f e(ll_0)
log likelihood  -19515.4, of the null model   -3188.9
```

The huge negative coefficients are not an error of the program; they are the answer to a model in which the time is in a very unsuitable unit. A better unit is the one in which the null model (the same rate for all ordered pairs) has a rate of one: the length of the window times $n(n-1)$, divided by the number of events, here 59 days. With the time divided by this number:

```
. * the time unit in which the null model has a rate of 1 per ordered pair: the le
> ngth of the window times n(n-1) / N
. scalar unit = (5 - 3.2) * 144 * 143 / 628

. display "one time unit = " %5.1f unit " days"
```

```
one time unit =  59.0 days

. quietly use ‘ev’, clear

. quietly duplicates drop t, force

. quietly replace t = (t - 3.2) / unit

. quietly nwbuild mail2

. nwrem mail2, nodsnd nidrec frpsndsnd interval starttime(0) endtime(‘=(5 - 3.2) /
>  unit’)
------------------------------------------------------------
Relational event model (waiting-time likelihood, MLE)
Network: mail2                          Actors: 144
Events: 628                             Log likelihood: -214.4741
------------------------------------------------------------
-----------------------------------------------------
       mail2 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   1.384774   .6537213     2.12   0.034
      nidrec |     3.8404    .581565     6.60   0.000
   frpsndsnd |   4.727286   .1137118    41.57   0.000
-----------------------------------------------------

. estimates store interval

. display "likelihood: " e(lik) "   window: " e(starttime) " to " %6.4f e(endtime)
>  "   log likelihood of the null model: " %9.2f e(ll_0)
likelihood: interval   window: 0 to 0.0305   log likelihood of the null model:   -
> 628.00
```

Now the coefficients are of the same order as those of the ordinal model; the log likelihood of the null model is -628 exactly, as it should be for a unit rate. The waiting-time model uses more information than the ordinal one (the log likelihoods are not comparable between the two likelihoods). Its estimates differ from those of the ordinal model (the inertia effect is 4.73 against 4.22, the activity effects are smaller), because the times carry information of their own:

```
. estimates table ordinal interval, b(%9.3f) se(%9.3f) stats(ll N)

--------------------------------------
    Variable |  ordinal    interval
-------------+------------------------
      nodsnd |    15.083       1.385
             |     1.623       0.654
      nidrec |    16.884       3.840
             |     1.278       0.582
   frpsndsnd |     4.222       4.727
             |     0.144       0.114
-------------+------------------------
          ll | -5716.341    -214.474
           N |       628         628
--------------------------------------
                         Legend: b/se
```

Without `endtime()` the observation ends with the last event, and `e(endtime)` is not stored:

```
. nwrem mail2, nodsnd nidrec frpsndsnd interval
------------------------------------------------------------
Relational event model (waiting-time likelihood, MLE)
Network: mail2                          Actors: 144
Events: 628                             Log likelihood: -209.8953
------------------------------------------------------------
```

```
------------------------------------------------------
       mail2 | Coefficient  Std. err.      z    P>|z|
-------------+----------------------------------------
      nodsnd |   1.384178    .653735     2.12   0.034
      nidrec |   3.838955   .5819593     6.60   0.000
   frpsndsnd |   4.733082   .1137111    41.62   0.000
------------------------------------------------------

. display "window: " e(starttime) " to " %6.4f e(endtime)
window: 0 to      .
```

Conditioning. `condition(100)` lets the first 100 events build the history and removes them from the likelihood, for both likelihoods. `e(N)` still counts all 628 events:

```
. nwrem mail2, nodsnd nidrec frpsndsnd interval starttime(0) endtime(‘=(5 - 3.2) /
>  unit’) condition(100)
------------------------------------------------------------
Relational event model (waiting-time likelihood, MLE)
Network: mail2                          Actors: 144
Events: 628                             Log likelihood: -174.2445
------------------------------------------------------------
------------------------------------------------------
       mail2 | Coefficient  Std. err.      z    P>|z|
-------------+----------------------------------------
      nodsnd |    23.8392   2.508857     9.50   0.000
      nidrec |    14.5457    1.42869    10.18   0.000
   frpsndsnd |   3.950331   .1456929    27.11   0.000
------------------------------------------------------

. display "events: " e(N) "   conditioned: " e(N_cond)
events: 628   conditioned: 100

. nwrem mail2, nodsnd nidrec frpsndsnd condition(100)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mail2                          Actors: 144
Events: 628                             Log likelihood: -4.68e+03
------------------------------------------------------------
------------------------------------------------------
       mail2 | Coefficient  Std. err.      z    P>|z|
-------------+----------------------------------------
      nodsnd |   60.64005   3.140132    19.31   0.000
      nidrec |   20.46394   1.501178    13.63   0.000
   frpsndsnd |   4.346982   .1675796    25.94   0.000
------------------------------------------------------
```

8.4 Risk sets, entry and exit

ex05_riskset.do. Suppose that the question concerns e-mails within the divisions of the company: pairs of employees from different divisions are not part of the risk set. We keep the 277 e-mails of the window that stay within a division (of 631), build the network `samediv` with a tie for every ordered pair of employees of the same division, and name it in `riskset()`:

```
. count
  631

. count if divs == divr
  277
```

```
. nwbuild mail
```

```
. mata:
------------------------------------------------- mata (type end to exit) --------
: D = st_sdata(., "division")

: n = rows(D)

: M = J(n, n, 0)

: for (i = 1; i <= n; i++) {
>     for (j = 1; j <= n; j++) M[i, j] = (i != j) & (D[i] == D[j])
> }

: end
--------------------------------------------------------------------------------

. nwset, mat(M) name(samediv) directed

. mata: printf("ordered pairs that can be events: %g of %g\n", sum(M), n * (n - 1)
> )
ordered pairs that can be events: 6664 of 20592
```

```
. nwrem within, nodsnd nidrec frpsndsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: within                       Actors: 144
Events: 277                           Log likelihood: -2.42e+03
------------------------------------------------------------
-----------------------------------------------------
      within | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   8.442042   2.414806     3.50   0.000
      nidrec |   10.95608   1.274955     8.59   0.000
   frpsndsnd |   4.788696    .180856    26.48   0.000
-----------------------------------------------------

. nwrem within, nodsnd nidrec frpsndsnd riskset(samediv)
Note: riskset(samediv) matched to the event network by position (labels agree).
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: within                       Actors: 144
Events: 277                           Log likelihood: -2.20e+03
------------------------------------------------------------
-----------------------------------------------------
      within | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |    5.36297   1.954483     2.74   0.006
      nidrec |   9.036033   1.196073     7.55   0.000
   frpsndsnd |   3.847225   .1725891    22.29   0.000
-----------------------------------------------------
```

The first fit uses all $144 \cdot 143 = 20{,}592$ pairs as the risk set, the second only the 6,664 within-division pairs. The log likelihoods are not comparable, because the sets of alternatives differ. The coefficients differ as well, because every event is now compared with a different set of alternatives. If an event is not in the risk set, `nwrem` stops:

```
. capture noisily nwrem mail, nodsnd nidrec riskset(samediv)
Warning: riskset(samediv) matched to the event network by position, but the node
      labels are in a DIFFERENT ORDER: position matching is probably wrong; use
      covmatch(label).
```

```
nwrem: event 1 of the sorted event sequence is a dyad that is not in the risk
set given by riskset().

. display "return code: " _rc
return code: 459
```

The warning above the error is the note that every fit with `riskset()` or `covevent()` prints about the matching of the network to the actors (Section 4.8.1): here the network `samediv` was built from the events of the window, so its nodes are in another order than those of `mail`, and with the default position matching the risk set would be read for the wrong actors. In the fits above, where the labels agree, the note says so.

Entry and exit. Per-actor windows are variables of the current dataset. We let the employees of division E85 leave the observation after day 4.0 and those of division E70 join it on day 3.6. An event after the exit time of one of its actors is an error:

```
. quietly use ‘ev’, clear

. count if t > 4 & (divs == "E85" | divr == "E85")
  28

. count if t < 3.6 & (divs == "E70" | divr == "E70")
  34

. quietly nwload mail, xvars

. quietly gen actor = _nwnode

. quietly gen long row = _n

. quietly merge m:1 actor using ‘staff’, keep(master match) nogenerate

. quietly sort row

. quietly gen double leave = 4 if division == "E85"

. quietly gen double join = 3.6 if division == "E70"

. capture noisily nwrem mail, nodsnd nidrec exit(leave)
nwrem: event 423 of the sorted event sequence involves an actor outside its
entry() / exit() window.

. display "return code: " _rc
return code: 459
```

We drop the events that violate the windows (569 events remain), and fit the model first with the complete risk set and then with the windows:

```
. quietly use ‘ev’, clear

. quietly drop if t > 4 & (divs == "E85" | divr == "E85")

. quietly drop if t < 3.6 & (divs == "E70" | divr == "E70")

. count
  569

. nwbuild mailw

. quietly nwload mailw, xvars

. quietly gen actor = _nwnode
```

```
. quietly gen long row = _n

. quietly merge m:1 actor using ‘staff’, keep(master match) nogenerate

. quietly sort row

. quietly gen double leave = 4 if division == "E85"

. quietly gen double join = 3.6 if division == "E70"

. nwrem mailw, nodsnd nidrec frpsndsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mailw                          Actors: 144
Events: 569                             Log likelihood: -5.17e+03
------------------------------------------------------------
-----------------------------------------------------
       mailw | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   13.30402   1.565793     8.50   0.000
      nidrec |    15.8121   1.187604    13.31   0.000
   frpsndsnd |   4.151412   .1507897    27.53   0.000
-----------------------------------------------------

. nwrem mailw, nodsnd nidrec frpsndsnd entry(join) exit(leave)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: mailw                          Actors: 144
Events: 569                             Log likelihood: -5.11e+03
------------------------------------------------------------
-----------------------------------------------------
       mailw | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   12.99724   1.560241     8.33   0.000
      nidrec |   15.35007   1.185841    12.94   0.000
   frpsndsnd |   4.113891   .1505241    27.33   0.000
-----------------------------------------------------
```

8.5 Posterior mode

ex06_postmode.do. With the default prior (a Student t with four degrees of freedom and scale 100) the posterior mode is almost the maximum likelihood estimate. A normal prior of scale 2 pulls the large share coefficients towards zero, a normal prior of scale 0.5 around 3 pulls them towards 3.

```
. estimates table mle weak normal2 normal05, b(%9.3f) se(%9.3f) stats(ll N)

-------------------------------------------------------------
    Variable |    mle        weak       normal2    normal05
-------------+-----------------------------------------------
      nodsnd |    14.364      14.360       9.308       4.403
             |     1.617       1.616       1.155       0.466
      nidrec |    22.018      22.014      15.881       5.337
             |     1.183       1.183       1.117       0.469
   frpsndsnd |     2.296       2.296       2.530       3.012
             |     0.186       0.186       0.191       0.180
    frrecsnd |     3.821       3.821       3.723       3.498
             |     0.162       0.162       0.170       0.169
-------------+-----------------------------------------------
          ll | -5531.783   -5531.783   -5550.102   -5639.199
           N |       631         631         631         631
```

```
------------------------------------------------------------
                                                Legend: b/se
```

The log likelihoods are those of the likelihood at the mode, without the prior, so they fall as the prior pulls the estimates away from the maximum. The four fits are made in `ex06_postmode.do`; the table shows the estimates and standard errors of each.

8.6 Triadic, recency, fixed and participation effects

ex07_effects.do. These effects need an event history in which the same pairs recur, so we use the 67 e-mails among the twelve most active employees of the window. Two of the twelve never send and one never receives in these data.

```
. count
  67

. display "actors: $allact"
actors: E128 E136 E137 E152 E166 E17 E47 E60 E62 E65 E81 E90

. quietly levelsof sender, local(sn) clean

. quietly levelsof receiver, local(rn) clean

. local nsend : list global(allact) - sn

. local nrec : list global(allact) - rn

. display "never send: `nsend'   never receive: `nrec'"
never send: E137 E60   never receive: E17

. nwbuild core12

. nwsummarize core12
-------------------------------------------------
   Network name:  core12
   Network id:  1
   Directed: true
   Valued: false
   Two-mode: false
   Nodes: 12
   Selfloop: false
   Arcs: 27
   Minimum value:  0
   Maximum value:  1
   Density:  .205
   Mean outdegree:  2.25
   Temporal: true
      Temporal type: event
```

The shared partners and two-paths:

```
. nwrem core12, nodsnd otpsnd itpsnd ospsnd ispsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -310.4355
------------------------------------------------------------
-------------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+-----------------------------------------
      nodsnd |   2.856477   1.122909     2.54   0.011
      otpsnd |   .5424904   .2931342     1.85   0.064
```

```
      itpsnd |  -.1985253   .3566604    -0.56   0.578
      ospsnd |   .4053784    .123499     3.28   0.001
      ispsnd |  -.9729397   .4110358    -2.37   0.018
-------------------------------------------------------
```

The coefficient of `ospsnd`, the symmetric number of shared out-partners (the number of colleagues to whom both have written), is positive: pairs with more shared recipients have a higher rate. Recency, in the presence of frequency:

```
. nwrem core12, nodsnd nidrec frpsndsnd rsndsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -266.4054
------------------------------------------------------------
-------------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+-----------------------------------------
      nodsnd |   .4161406   1.253919     0.33   0.740
      nidrec |   1.528693   .8577036     1.78   0.075
   frpsndsnd |  -2.919509   .7881275    -3.70   0.000
     rsndsnd |   4.376669   .4775756     9.16   0.000
-------------------------------------------------------
```

In these 67 e-mails, once the recency of the last contact is in the model, the frequency of contact (`frpsndsnd`) has a negative coefficient: the person to whom a sender has written most often is not the one to whom the next message tends to go; the most recent partner is. The fixed effects of the senders:

```
. nwrem core12, fesnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -298.8818
------------------------------------------------------------
-------------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+-----------------------------------------
     fesnd_2 |  -.3364722   .4140393    -0.81   0.416
     fesnd_3 |  -28.32337   377813.7    -0.00   1.000
     fesnd_4 |  -1.029619   .5209881    -1.98   0.048
     fesnd_5 |  -1.029619   .5209881    -1.98   0.048
     fesnd_6 |   -1.94591   .7559289    -2.57   0.010
     fesnd_7 |  -.5596158   .4432026    -1.26   0.207
     fesnd_8 |  -28.32337   377813.7    -0.00   1.000
     fesnd_9 |  -2.639057   1.035098    -2.55   0.011
    fesnd_10 |  -.2411621   .4029115    -0.60   0.549
    fesnd_11 |  -2.639057   1.035098    -2.55   0.011
    fesnd_12 |  -.3364722   .4140393    -0.81   0.416
-------------------------------------------------------
Note: no usable standard error for fesnd_3 fesnd_8.
      These effects are not identified: typically an effect is an exact function
      of others in the model (the total-degree share ntdegsnd is half the sum of
      nodsnd and nidsnd). Its estimate is arbitrary; drop one of the effects.
```

Two actors that never send have a coefficient that would tend to minus infinity; the program stops at a large negative value, the standard error is huge, and the note appears. With the sender effects alone, the estimate of an actor is the logarithm of the ratio of the number of e-mails it sent to the number sent by the first actor, so actors with the same number of e-mails have identical coefficients. A normal prior of scale 2 (`postmode priordf(.) priorscale(2)`) gives finite, estimable values:

```
. nwrem core12, fesnd postmode priorscale(2) priordf(.)
------------------------------------------------------------
Relational event model (ordinal partial likelihood, posterior mode)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -300.9190
------------------------------------------------------------
-----------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
     fesnd_2 |  -.0738654   .4113092    -0.18   0.857
     fesnd_3 |  -2.752192   1.051959    -2.62   0.009
     fesnd_4 |  -.7328695   .5026824    -1.46   0.145
     fesnd_5 |  -.7328695   .5026824    -1.46   0.145
     fesnd_6 |  -1.512032   .6650319    -2.27   0.023
     fesnd_7 |  -.2898373   .4371627    -0.66   0.507
     fesnd_8 |  -2.752192   1.051959    -2.62   0.009
     fesnd_9 |  -1.976718    .792895    -2.49   0.013
    fesnd_10 |   .0191642    .401283     0.05   0.962
    fesnd_11 |  -1.976718    .792895    -2.49   0.013
    fesnd_12 |  -.0738654   .4113092    -0.18   0.857
-----------------------------------------------------
```

The fixed effects of the receivers, with a degree effect of the sender:

```
. nwrem core12, ferec nodsnd
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -306.0910
------------------------------------------------------------
-----------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
      nodsnd |   3.671309   1.106011     3.32   0.001
     ferec_2 |   .0235195   .7072077     0.03   0.973
     ferec_3 |   1.272116   .5629661     2.26   0.024
     ferec_4 |   .1934604   .6709338     0.29   0.773
     ferec_5 |   .8916515   .5917241     1.51   0.132
     ferec_6 |  -26.04711   221236.3    -0.00   1.000
     ferec_7 |   .0109052   .7072671     0.02   0.988
     ferec_8 |  -.7427872   .8661779    -0.86   0.391
     ferec_9 |  -.0489967   .7072907    -0.07   0.945
    ferec_10 |   .6822698   .6123992     1.11   0.265
    ferec_11 |   .1830564   .6709797     0.27   0.785
    ferec_12 |   .4104313   .6455216     0.64   0.525
-----------------------------------------------------
Note: no usable standard error for ferec_6.
      These effects are not identified: typically an effect is an exact function
      of others in the model (the total-degree share ntdegsnd is half the sum of
      nodsnd and nidsnd). Its estimate is arbitrary; drop one of the effects.
```

The participation shifts describe how the next event follows the previous one. The pair that repeats the previous event is the reference category:

```
. nwrem core12, psabba psabby psabxa psabxb psabxy psabay
------------------------------------------------------------
Relational event model (ordinal partial likelihood, MLE)
Network: core12                         Actors: 12
Events: 67                              Log likelihood: -262.8356
------------------------------------------------------------
-----------------------------------------------------
      core12 | Coefficient  Std. err.      z    P>|z|
-------------+---------------------------------------
```

```
      psabba |   .4418328   .4272466     1.03   0.301
      psabby |   -4.49981   1.054093    -4.27   0.000
      psabxa |  -29.47336   264795.5    -0.00   1.000
      psabxb |  -2.420368   .4859127    -4.98   0.000
      psabxy |  -3.438938   .3867462    -8.89   0.000
      psabay |  -2.420368   .4859127    -4.98   0.000
-------------------------------------------------------
Note: no usable standard error for psabxa.
      These effects are not identified: typically an effect is an exact function
      of others in the model (the total-degree share ntdegsnd is half the sum of
      nodsnd and nidsnd). Its estimate is arbitrary; drop one of the effects.
```

The reply $B \to A$ (`psabba`) has a coefficient not different from zero, and all the other shifts are negative: relative to a repetition of the previous event, every other kind of event is less likely. The coefficient of `psabxa` (someone else writes to the previous sender) is not identified because it does not occur in these data.

Chapter 9

Validation against `relevent`

`nwrem` is checked against the R package `relevent`, version 1.2-1, in three ways: its estimates are compared with those of `relevent` on simulated event logs; its likelihood, gradient and Hessian are compared with independent evaluators; and its handling of effects that cannot be estimated is compared with the behavior of the statistics of `relevent`. This chapter states what was compared and with which tolerances. The test suite of the package (the folder `cscripts` of the repository) reruns all of these comparisons, with the reference values stored in the repository; the scripts that made the reference values in R are in the folder `dev`.

9.1 Estimates against `relevent`

Design. There are 60 cases: event logs simulated with 6 to 8 actors and 150 to 300 events (with repeated pairs, replies and, where the model needs them, actor covariates, covariate networks, risk sets, entry and exit times), each with a model that uses the options of this version: all original effects, the triadic effects, the fixed effects, the participation shifts, several covariates per option, the waiting-time likelihood (with and without conditioning and with an `endtime()` after the last event), the posterior mode with several priors, and restricted risk sets. For each case `relevent` was run in R and the coefficients, standard errors and log likelihood stored. Two reference routes were used.

Exact route. The statistics of the effects were taken from `relevent`'s own routines and fitted with `rem()` of `relevent`, which has an analytic gradient and Hessian and a trust-region optimizer. This gives the exact maximum of the likelihood of `relevent`. `nwrem` must agree with it to 10^{-5} in the coefficients and in the standard errors and to 10^{-9} in the log likelihood (differences measured as $|x - x_0|/(1 + |x_0|)$).

`rem.dyad()`. The function `rem.dyad()` of `relevent` itself, where the model is one that it can fit (48 of the 60 cases). It uses BFGS with a numerical Hessian and stops at the fifth to sixth digit of the coefficients; the tolerance is $2 \cdot 10^{-4}$.

`rem.dyad()` cannot fit a restricted risk set or an entry and exit window, and it computes the asymmetric statistic of `ospsnd`. For the 8 cases with a risk set or windows and the 4 cases with `ospsnd`, only the exact route exists: the likelihood is that of `rem()` of `relevent` with a support matrix (for the risk sets) or with its own statistics (all others); the statistics of the restricted risk sets and of the symmetric `ospsnd` were built in R by a short function that implements the definition of Chapter 5 and are not those of `relevent`. That check shows that `nwrem` computes the symmetric statistic, the risk sets and the windows as defined, and that the likelihood built on them is that of `relevent`; it is not a check of those statistics against code of `relevent`.

Result. The test passes in all cases. The largest deviations per group are: The deviations from the exact maximum are of the size of the round-off of the optimization (below $3 \cdot 10^{-6}$ in every case; most are far smaller), and the log likelihoods agree to about $3 \cdot 10^{-11}$ or better. The deviations

Table 9.1: Largest deviation from `relevent`, by group of cases (Stata 17, `nwrem` 1.7.0).

		Exact route			`rem.dyad()`
	Cases	coef.	s.e.	log lik.	cases: dev.
Original effects (degree, inertia, recency)	3	$2.3 \cdot 10^{-6}$	$1.1 \cdot 10^{-7}$	$3.4 \cdot 10^{-14}$	3: $2.3 \cdot 10^{-6}$
Waiting-time likelihood, conditioning	11	$2.0 \cdot 10^{-7}$	$7.3 \cdot 10^{-8}$	$3.5 \cdot 10^{-14}$	11: $2.9 \cdot 10^{-7}$
Triadic and shared-partner effects (without `ospsnd`)	12	$2.0 \cdot 10^{-7}$	$2.5 \cdot 10^{-8}$	$2.3 \cdot 10^{-14}$	12: $2.1 \cdot 10^{-5}$
Symmetric `ospsnd`	4	$7.4 \cdot 10^{-8}$	$1.9 \cdot 10^{-8}$	$9.7 \cdot 10^{-15}$	–
Fixed effects	6	$2.4 \cdot 10^{-8}$	$2.1 \cdot 10^{-9}$	$5.9 \cdot 10^{-15}$	6: $2.1 \cdot 10^{-6}$
Participation shifts	9	$9.6 \cdot 10^{-7}$	$8.0 \cdot 10^{-8}$	$4.0 \cdot 10^{-14}$	9: $3.3 \cdot 10^{-6}$
Several covariates and covariate networks	3	$5.5 \cdot 10^{-9}$	$1.1 \cdot 10^{-9}$	$8.3 \cdot 10^{-16}$	3: $4.9 \cdot 10^{-7}$
Posterior mode	4	$5.7 \cdot 10^{-7}$	$1.7 \cdot 10^{-8}$	$2.4 \cdot 10^{-11}$	4: $8.2 \cdot 10^{-7}$
Risk sets, entry and exit	8	$1.0 \cdot 10^{-6}$	$4.1 \cdot 10^{-8}$	$2.0 \cdot 10^{-14}$	–

from `rem.dyad()` are those of the BFGS iteration of `relevent` (about 10^{-5} at most here). The package therefore returns the exact maximum of the likelihood, to at least five digits; `relevent` with its default control settings stops earlier. The first lines of the test output and its summary:

```
. test_nwrem_relevent14   (the first 10 of the 60 case lines, in columns)
case   exact route: coef, se, log lik.  |  rem.dyad(): coef, se
a1      2.22e-09  1.14e-10  1.16e-15  |   3.40e-08  2.92e-07
a2      2.35e-06  1.06e-07  3.37e-14  |   2.32e-06  1.88e-07
a3      1.04e-07  7.21e-09  3.96e-16  |   5.57e-08  6.36e-08
b1      4.17e-08  9.91e-09  8.41e-16  |   1.12e-07  2.00e-08
b2      1.80e-08  4.25e-09  3.04e-15  |   1.86e-08  7.01e-08
b3      4.72e-08  9.51e-09  2.64e-15  |   1.02e-07  2.08e-08
b4      6.80e-11  1.79e-11  1.52e-15  |   1.37e-07  7.86e-08
b5      3.47e-09  1.22e-09  1.79e-15  |   1.18e-07  8.32e-08
b6      3.09e-13  3.59e-14  5.14e-15  |   6.08e-08  3.61e-08
b7      2.03e-07  7.30e-08  3.54e-14  |   2.93e-07  3.61e-08

PASS: nwrem agrees with relevent in all 60 cases
test_nwrem_relevent14: ALL OK

. test_nwrem_collinear14  (the first 5 of the 39 case lines)
col1     parameters   8  omitted   6   same
col2     parameters   8  omitted   6   same
col3     parameters   8  omitted   6   same
col4     parameters   8  omitted   6   same
col5     parameters   8  omitted   5   same

PASS: the omitted coefficients agree with the reference in all 39 cases (76 omitted coefficients
     in all)
test_nwrem_collinear14: ALL OK
```

Timing runs on larger simulations (Chapter 11) agree with `rem.dyad()` to about three significant digits (the largest relative difference of a coefficient was $4.2 \cdot 10^{-3}$, in a cell where the optimizer of `relevent` stops early).

Earlier checks of the original effects. The original effects (degree, inertia, recency, covariates of the actors and of the pair) were checked in separate runs with `rem.dyad()` on fixed event logs of 8 to 10 actors and 400 to 500 events, one effect family at a time and with the degree and inertia effects together; the scripts and the exported data are in the folder `dev`. The degree-accumulator semantics were checked against the output of the function `acl.deg()` of `relevent`.

9.2 The shared out-partner statistic

`ospsnd` is the one statistic that is deliberately not that of `relevent` 1.2-1 (Section 5.6). The symmetric statistic was checked as described above: 60 cases include four with `ospsnd` (alone with `nodsnd`, and together with the other triadic effects), and `nwrem` agrees with the exact route of `relevent`'s likelihood fed with the symmetric statistic (largest deviation in the coefficients $7.4 \cdot 10^{-8}$). If you need the estimate of `relevent` for `OSPSnd`, `nwrem` cannot reproduce it.

9.3 The likelihood, gradient and Hessian of the plugin

The plugin is compared with independent evaluators written in Mata (the reference evaluators of the test suite, which agree with `relevent`) on random small event sequences with ties between event times, repeated pairs and valued covariate networks (negative, zero and large values, with missing or non-zero diagonals). The comparisons cover every effect alone, every pair of effects, random subsets and the full set; every one of the 2^{14} subsets of the 14 original effects on a small system; the fast and the dense evaluator of the plugin against each other; extreme coefficients, for which the fast evaluator gives up and the dense one takes over; and the exact Hessian against the numerical derivative of the gradient. The log likelihood and the gradient agree to 10^{-9} (relative) and the Hessian with the numerical derivative to 10^{-5}.

9.4 Effects that are not estimable

For 39 very small networks (2 and 3 actors, in which many effects become exactly dependent) the set of coefficients that `nwrem` leaves out was compared with the set found from the statistics of `relevent`: in all 39 cases the sets are identical (76 omitted coefficients in all).

9.5 Other checks

The test suite also contains: tests of every error path with its return code and of the stored results and identities between them (the log likelihood of the null model, the information criteria); a test that a network declared directed and the same network declared undirected give the same estimates; tests of the stops that follow from a missing or outdated plugin; and tests of the messages at a screen width of 80 characters. The complete test suite of version 1.7.0 was run on macOS with Stata 17, Stata 18 and StataNow 19.5. It has not yet been run on Linux or Windows for this version.

Chapter 10

Limits, known issues and messages

This chapter says what `nwrem` cannot do and which problems are known, and explains the messages that a user is likely to meet.

10.1 What is not implemented

- **Bayesian simulation.** The simulation-based estimation of `relevent` (`fit.method` MCMC and BSIR) is not implemented: its output is random and cannot be validated against the random numbers of R. The posterior mode (`postmode`) is available.
- **Sampling of the risk set** (`sample.size` of `relevent`) is not implemented; the likelihood always uses the complete risk set, or the restriction of `riskset()`, `entry()` and `exit()`.
- **Simulation of event sequences** and **predictions** for new data are not implemented; the diagnostics of Chapter 7 compare the fitted model with the data at hand.
- **Covariates that change over time.** All covariates (`covsnd()`, `covrec()`, `covint()`, `covevent()`) are fixed in time. Covariates that change with the events would need an event-wise data structure. Statistics of the history (degrees, inertia, recency, triads, participation shifts) change with the events by definition.
- **The participation shifts that are always zero.** `relevent` defines seven participation shifts with undefined targets (for example `PSAB-BO`); its coefficients for them are always zero. `nwrem` offers the six shifts that are not (Section 5.8).
- **No intercept** (Section 2.5).
- **Two-mode event networks** are refused.
- **No dialog boxes.** The package has no dialog.

10.2 Known issues and cautions

- **`ospsnd` differs from `relevent`.** By design (Section 5.6).
- **The waiting-time likelihood and the unit of time.** The rate of a pair with all statistics zero is one per unit of time. If the unit is unsuitable, the coefficients of the share effects become very large (Section 8.3).
- **Noisy degree effects.** Degree-type effects are running shares that converge to $1/n$; estimates can stay noisy with thousands of events, and several of them in one model can lie on a flat ridge of the likelihood (Section 4.9).

- **Separation.** A coefficient that tends to minus (or plus) infinity because an actor never sends or receives, or an effect never occurs, is reported with a huge value and standard error and is listed in `e(unidentified)`; `postmode` gives finite values.
- **Effects with a heavy cost.** The fixed effects are evaluated densely for every pair at every event, and a model with a large number of actors and fixed effects is slow; the effects that depend on the sparse history (inertia, recency, triadic effects, covariate networks) cost time in proportion to the number of pairs that have been in contact. Chapter 11 gives measured times.
- **Ties in time.** The ordinal likelihood keeps tied events in the order of the rows, so the estimates depend on this order if there are many ties. The waiting-time likelihood refuses ties.
- **Platforms.** The plugin is built for macOS, Linux and Windows; the test suite of this version has been run on macOS only (Section 1.4). Windows has not been tested.
- **Results of other versions.** Results of fits made with other versions may differ; report the version printed by `nwrem version`.

10.3 Messages and what to do

The messages that the examples of this manual produced and those that the checks of the command produce, and the usual remedy.

Message (return code)	Meaning and remedy
nwrem requires at least one effect (198)	No effect was given. Name at least one.
nwrem requires a network declared via nwset's eventtime() option (198)	The network is not an event network. Declare it with `nwset ..., eventtime()`.
nwrem does not support two-mode networks (198)	Event networks must be one-mode.
starttime() and endtime() belong to the waiting-time likelihood (198)	Add the option `interval`.
priormean(), priorscale() and priordf() belong to posterior-mode estimation (198)	Add the option `postmode`.
condition() must be a non-negative integer (198); *condition() must be smaller than the number of events* (198)	Give an integer from 0 to the number of events minus one.
seed() must be an integer between 0 and 2147483647 (198)	Use a valid seed.
N actor(s) of net have no row in the current dataset / have more than one row (198)	The covariates and the entry and exit times need exactly one row per actor with the label in `_nwnode`; load the dataset with `nwload net, xvars`.
the current dataset has N observations but net has M actors (198)	Without `_nwnode` the data need one row per actor, in the actor order of the network.
covsnd(x): N actor(s) have a missing value (416)	The covariate must be defined for every actor.
covmatch(label): the node labels of the networks differ / a label occurs twice (459)	With `covmatch(label)` the label sets of the covariate network and of the event network must be identical, each label once.
Warning: ... matched to the event network by position, but the node labels are in a DIFFERENT ORDER (note, no error)	With the default position matching the networks are read row by row; fit again with `covmatch(label)` or reorder the network.
covevent() network has N actors (198), *has missing tie values* (416), two-mode (6088)	A covariate network must have the same actors, no missing ties and one mode.
N event(s) have a sender or receiver that is not an actor, a missing event time (416)	Drop such rows before `nwset`.
event k has sender == receiver (3498)	A self-loop; drop such rows.
the interval likelihood needs strictly increasing event times (459)	Events with equal times: drop or separate them, or use the ordinal likelihood.

Message (return code)	Meaning and remedy
starttime(#) is later than the first event; *endtime(#) is earlier than the last event* (459)	Correct the window.
event k ... is a dyad that is not in the risk set (459)	The pair of the event is not allowed by `riskset()`. Widen the risk set or drop the event.
event k ... involves an actor outside its entry() / exit() window (459)	Correct the times or drop the event.
the fixed effects add N parameters per effect ... too many (198)	At most 600 fixed-effect coefficients. Use fewer actors or fewer fixed-effect options.
nwrem needs at least 2 events (2001)	Too few events.
the optimization did not find a maximum of the likelihood (430)	No maximum was reached; check for separation and collinearity, use `postmode`.
no usable standard error for ... (a note)	Effects that are collinear (coefficient 0, `(omitted)`) or not identified; drop an effect or use `postmode` (Section 4.9).
the compiled plugin ... was not found (601)	The plugin of your platform is missing. Reinstall the package; build it with `nwinstall, buildplugin(nwrem)`.
the installed plugin ... is too old (3495)	The plugin does not fit the `.ado` files or cannot be loaded. Reinstall; restart Stata; build it on your machine if it does not load.
... too new to be run by this version of Stata (3499)	The Mata library was built by a newer Stata; build it with `nwinstall, buildmata(nwrem)`.

The following listing shows several of these messages as the program prints them.

```
. capture noisily nwrem mail, nodsnd starttime(3)
starttime() and endtime() belong to the waiting-time likelihood: add the option
interval.
invalid syntax

. display "return code: " _rc
return code: 198

. capture noisily nwrem mail, nodsnd priorscale(2)
priormean(), priorscale() and priordf() belong to posterior-mode estimation: add
the option postmode.
invalid syntax

. display "return code: " _rc
return code: 198

. capture noisily nwrem mail, nodsnd condition(-1)
condition() must be a non-negative integer (the number of leading events that only
>  build the history)
invalid syntax

. display "return code: " _rc
return code: 198

. capture noisily nwrem mail, nodsnd seed(-1)
seed() must be an integer between 0 and 2147483647
invalid syntax

. display "return code: " _rc
return code: 198
```

```
. clear

. input sender receiver t

        sender   receiver          t
  1. 1 2 1
```

```
  2. 2 2 2
  3. 1 3 3
  4. 3 1 4
  5. end

. nwset sender receiver, directed eventtime(t) name(loop)

. capture noisily nwrem loop, nodsnd
nwrem: event 2 has sender == receiver (self-loop); relational event models are
defined over i != j dyads only.

. display "return code: " _rc
return code: 3498
```

Chapter 11

Technical notes

This chapter describes how `nwrem` computes its estimates and how fast it is.

11.1 The structure of a fit

A fit has four steps. (1) The events are read from the network, checked and sorted by time (events with the same time keep the order of the rows). (2) The model, that is the events, the covariates, the risk set and the windows, is written once into a data frame and handed to the plugin; the frame is dropped at the end, also after an error. (3) The plugin checks for effects that are exact functions of others, by one evaluation at $\theta = 0$ (Section 4.9); the evaluation at $\theta = 0$ serves at the same time as the log likelihood of the null model and as the first Newton step. (4) A damped Newton iteration maximizes the log likelihood, calling the plugin once per evaluation with the current coefficients; each call returns the log likelihood, the gradient and the exact Hessian.

Newton iteration. The log likelihood is concave, so the Newton direction $d = (-H)^{-1}g$ is an ascent direction. A full step is taken and halved (up to 30 times) until the log likelihood does not fall. The iteration stops when the Newton decrement $g^\top(-H)^{-1}g$ is below $\max(10^{-12}, 10^{-13}|\ell|)$, at most 100 iterations; if the decrement is still above round-off level a few extra steps are taken while they reduce the norm of the gradient. The covariance matrix is $(-H)^{-1}$ at the end. When Newton does not reach a maximum (no maximum exists, the Hessian is singular), `nwrem` falls back to Mata's `optimize()`: the BFGS method from up to eight starting values (the first fixed, the others random), followed by Newton steps from the result, and, if BFGS fails, the Nelder–Mead method; the seed (`seed()`) matters only there. If that fails as well, the command stops with return code 430.

Mata and the plugin. Mata does the bookkeeping: it reads and sorts the events, writes the model into the frame, calls the plugin and runs the Newton loop. The plugin does the work that costs time. There is no slower fallback for the likelihood in Mata: a platform without a plugin cannot run `nwrem` (return code 601).

11.2 What the plugin computes

The log rate of the pair (j, k) at an event is a sum over the effects of $\theta_p x_p(j, k)$ (equation (2.1)). Every statistic is the sum of a part of the sender, a part of the receiver and a sparse dyadic part that is nonzero only for the pairs that have been in contact (the inertia, recency and triadic effects), for the nonzero cells of a covariate network, or for the pairs of the previous event (participation shifts). The normalizer of the ordinal likelihood, $\sum_{(j,k)} \exp(\theta^\top x)$ over the risk set, is then the product of the sum over senders and the sum over receivers (without the diagonal), plus a correction over the sparse dyads only. The cost per event is therefore of the order n plus the number of

pairs in contact so far, and not n^2. The counts, the recency lists, the weighted two-path and shared-partner counts and the list of pairs in contact are updated after every event and do not depend on θ. The sums that leave out the diagonal are formed from prefix and suffix sums, so that no cancellation arises.

When $\exp(\cdot)$ would overflow, a normalizer would underflow, or the sparse correction would cancel more than about six digits, the plugin evaluates that likelihood with a dense $O(n^2)$ evaluator instead, which is slower and equally exact. The fixed effects are always evaluated densely. The plugin carries no state between calls: the statistics do not depend on θ, and everything that depends on the model is in the frame. The C source (`rem_sim.c`, with StataCorp's plugin interface, which is not part of `nwrem`'s own code) is part of the package (Section 1.4); the plugin does not depend on the version or the flavor of Stata (BE, SE, MP), only on the operating system and the processor. The plugin uses one processor core.

Precision. The Hessian is exact, not a numerical derivative; the standard errors are those of the exact observed information. The estimates are the maximum of the likelihood to the round-off of the arithmetic, which is why they differ in the fifth digit or earlier from those that an optimizer with default tolerances reports.

11.3 Speed

We timed `nwrem` 1.4.0 against `rem.dyad()` of `relevent` 1.2-1 (R 4.6.0), with the ordinal likelihood and maximum likelihood, on simulated event streams of 200 to 20,000 events among 20 to 100 actors and on five sets of effects: S1, two degree effects (`nodsnd nidrec`); S2, four degree effects; S3, four degree effects and two inertia effects (`frpsndsnd frrecsnd`); S4, two degree effects and two recency effects (`rsndsnd rrecsnd`); S5, two degree effects and three covariate effects (`covsnd covrec covint`). `nwrem` always computes the covariance matrix from the exact Hessian, so the like-for-like setting of `relevent` is `hessian=TRUE`; its default is `hessian=FALSE`, and both were timed.

Machine and method. Intel Core i7-12700K (20 logical processors), 125 GB memory, Ubuntu 22.04.4, StataNow 19.5 MP (the fits use one core), R 4.6.0; one server without other load. The times of `nwrem` 1.7.0 were measured on 16 November 2026. The times of `relevent` 1.2-1 were measured on 15 November 2026 on the same machine with an earlier version of `nwrem` (1.2.0), and not repeated, so the two programs were not timed in turn. Each fit ran in its own Stata or R process after an untimed warm-up fit; the timer covers the estimation command (for R, the estimation call including its preprocessing), and loading the data is not timed. The table shows medians of up to 7 repeats with a different seed in each (5, 3 or 1 in the largest cells of `relevent`). Both programs received the same data and the same specification with their default control settings. Each fit had a limit of 600 seconds. The data are simulated by generator scripts in the repository of the core package (folder `bench`). Timings depend on the machine, the data and the effects. The first call in a Stata session takes a short time longer, because the Mata library and the plugin are loaded; this is not in the times.

Set	Events, actors	`nwrem` (s)	`relevent`, Hessian (s)	`relevent`, no Hessian (s)	Ratio with / without	
S1	200, 20	0.006	0.133	0.107	22	18
	1 000, 30	0.009	1.104	0.774	123	86
	5 000, 20	0.016	3.240	2.478	202	155
	5 000, 50	0.024	16.3	11.8	679	491
	5 000, 100	0.041	58.5	40.7	1 428	992
	20 000, 100	0.213	396.1	320.3	1 859	1 504
S2	200, 20	0.010	0.292	0.155	29	16
	1 000, 30	0.013	3.213	1.636	247	126

Set	Events, actors	`nwrem` (s)	`relevent`, Hessian (s)	`relevent`, no Hessian (s)	Ratio with / without	
	5 000, 20	0.039	8.795	5.306	226	136
	5 000, 50	0.048	49.5	29.3	1 032	611
	5 000, 100	0.068	151.4	73.4	2 227	1 079
	20 000, 100	0.261	–	576.4	n.r.	2 208
S3	200, 20	0.013	7.012	3.428	539	264
	1 000, 30	0.051	106.3	49.2	2 085	965
	5 000, 20	0.176	312.2	174.6	1 774	992
	5 000, 50	0.667	–	–	n.r.	n.r.
	5 000, 100	1.334	–	–	n.r.	n.r.
	20 000, 100	9.534	–	–	n.r.	n.r.
S4	200, 20	0.014	0.402	0.249	29	18
	1 000, 30	0.042	4.333	2.694	103	64
	5 000, 20	0.137	12.8	8.103	93	59
	5 000, 50	0.599	84.1	62.0	140	104
	5 000, 100	0.792	203.8	125.4	257	158
	20 000, 100	6.568	–	–	n.r.	n.r.
S5	200, 20	0.012	0.558	0.326	46	27
	1 000, 30	0.015	5.587	3.187	372	212
	5 000, 20	0.040	15.2	9.390	379	235
	5 000, 50	0.053	81.6	48.5	1 540	916
	5 000, 100	0.083	295.4	169.2	3 560	2 038
	20 000, 100	0.352	–	–	n.r.	n.r.

A dash means that `relevent` did not finish within 600 seconds, or was not run because it had timed out in a smaller cell of the same effect set (*n.r.* in the last two columns).

Reading the table. `nwrem` was faster than `relevent` in every cell in which both programs finished: the time of `relevent` divided by the time of `nwrem` ranged from 22 to 3,560 with the Hessian and from 16 to 2,208 without. The smallest ratios are with the recency effects (S4: 29 to 257 with the Hessian). Fits with several hundred events take a few milliseconds. `relevent` did not finish most of the fits with 20,000 events, while `nwrem` finished all of them in 0.2 to 9.5 seconds; the largest fit, S3 with 20,000 events and 100 actors, took 9.5 seconds. The estimates agree: the largest relative difference of a coefficient between the programs was $4.2 \cdot 10^{-3}$, in a cell where the optimizer of `relevent` stops early; in the other cells they agree to 10^{-3} or better.

Where the time goes. The degree and actor covariate effects (S1, S2, S5) cost little because their statistics are sums over senders and receivers; the effects that depend on pairs in contact (S3, S4, the triadic effects, covariate networks with many nonzero cells) cost time in proportion to the number of such pairs, and the fixed effects are evaluated for every pair at every event. For a large network with many pairs in contact, the time grows with the number of pairs that have occurred.

Appendix A

From R's `relevent` to `nwrem`

The table translates the arguments and effects of `rem.dyad()` of `relevent` to `nwrem`. The right column is the Stata command or option; `net` stands for a network declared as an event network.

R (`relevent`)	`nwrem`
The event list passed to `rem.dyad(edgelist, n, ...)`	`nwset sender receiver, directed eventtime(t) name(net)`; actors that are absent from the log: `nwaddnodes`
`rem.dyad(..., ordinal=TRUE, fit.method="MLE")`	`nwrem net,` *effects*
`NODSnd`, `NIDRec`, `NIDSnd`, `NODRec`	`nodsnd`, `nidrec`, `nidsnd`, `nodrec`
`NTDegSnd`, `NTDegRec`	`ntdegsnd`, `ntdegrec`
`FrPSndSnd`	`frpsndsnd` (inertia)
`FrRecSnd`	`frrecsnd` (reciprocity of receipt)
`RSndSnd`, `RRecSnd`	`rsndsnd`, `rrecsnd`
`OTPSnd`, `ITPSnd`, `ISPSnd`	`otpsnd`, `itpsnd`, `ispsnd`
`OSPSnd`	`ospsnd`: the symmetric statistic; the estimate differs from that of `relevent` 1.2-1 (Section 5.6)
`FESnd`, `FERec`, `FEInt`	`fesnd`, `ferec`, `feint` (one coefficient per actor but the first)
`PSAB-BA`, `PSAB-BY`, `PSAB-XA`, `PSAB-XB`, `PSAB-XY`, `PSAB-AY`	`psabba`, `psabby`, `psabxa`, `psabxb`, `psabxy`, `psabay`
The other participation shifts (`PSAB-B0`, `PSA0-X0`, `PSA0-XA`, `PSA0-XY`, `PSAB-X0`, `PSA0-AY`, `PSAB-A0`)	not implemented (always zero in `relevent`)
`CovSnd`, `CovRec`, `CovInt` with `covar=list(...)`	`covsnd(varlist)`, `covrec(varlist)`, `covint(varlist)`: numeric variables of the current dataset, one value per actor
`CovEvent`	`covevent(netlist)`: networks with the same actors that hold the values of the pairs
`ordinal=FALSE` (waiting times)	`interval` with `starttime()` and `endtime()`
`conditioned.obs`	`condition(k)`
`fit.method="BPM"` with `prior.mean`, `prior.scale`, `prior.nu`	`postmode` with `priormean()`, `priorscale()`, `priordf()`
`rem()` with a support matrix	`riskset(net)` (a pair with the value 0 cannot occur), `entry(var)` and `exit(var)` for actors that enter or leave
`hessian=TRUE/FALSE`	the covariance matrix is always computed
`fit.method="MCMC"`, `"BSIR"`; `sample.size`; `gof`	not implemented
`summary()` of the fit	the table printed by `nwrem`; `e(b)`, `e(V)`, `e(ll)`, `e(N)`, `e(nodes)`
`logLik()`, `AIC()`, `BIC()`	`e(ll)`, `e(aic)`, `e(bic)`; `estimates stats`

Bibliography

[1] Butts, C.T. (2008). A relational event framework for social action. *Sociological Methodology* 38(1), 155–200. https://doi.org/10.1111/j.1467-9531.2008.00203.x

[2] Butts, C.T. *relevent: Relational Event Models*. R package (compared with version 1.2-1). https://CRAN.R-project.org/package=relevent

[3] Cox, D.R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society, Series B* 34(2), 187–220.

[4] Michalski, R. (2020). Manufacturing company email metadata and corporate hierarchy. Harvard Dataverse, V2. https://doi.org/10.7910/DVN/6Z3CGX (CC0 1.0).

[5] Nurek, M., Michalski, R. (2020). Combining machine learning and social network analysis to reveal the organizational structures. *Applied Sciences* 10(5), 1699.