Università degli Studi di Padova

Dipartimento di Scienze Statistiche

Corso di Laurea Magistrale in

Scienze Statistiche

Balanced and Imbalanced triads effects on the network structure

Relatore: Prof. Alessandra R. Brazzale
Dipartimento di Scienze Statistiche

Correlatore: Prof. Ernst C. Wit
Unit of Statistics and Probability
University of Groningen

Laureando: Francesco Contin
Matricola N. 1034523

Anno Accademico 2014/2015

Contents

1 Introduction

A social network is a structure in which social agents, such as individuals or organisations, interact with each other. This interaction can be the creation, maintenance or withdrawal of ties. The actors must belong to a meaningfully delineated social group. This delimitation will imply that actors are only studied in this limited pool which on one hand will bound the generality of the study but, on the other hand, yields consistency to the data. The structure can be decomposed into a set of dyadic links between the generic units observed.

The aim of the analysis of a social network is to understand the dynamics that led the network to develop as a whole; the changes in its structure can be due to the characteristics of the network itself and to the actors characteristics. Traditionally, social network analysis had a focus on the rich description of network data, but the recent development of methods of inference for network data has the potential for a wider use [1].

The fields in which the network analysis can be applied are nowadays surprisingly wide, they can be, e.g.:

- Communication studies, which are related to fields such as sociology, psychology, anthropology, political science and economics, as the information transfer can be studied as a network.
- Diffusion of innovation and ideas, in which the point of interest could be to explain why someone becomes an "early adopter" and someone does not.
- Economic sociology, which considers behavioural interactions of individuals and groups through social capitals and social markets.
- Biological network analysis, which is closely related to the social network analysis, but often focusing on local patterns in the network.
- Social media, since computer networks combined with social networking software produce a new medium for social interaction. This can also include the transfer of informations, goods or services in the real world.

This list includes only few of the most important fields in which the study of social networks can be applied. Hence, individuals observed can belong to many different environments as, e.g., they can be the pupils of a school whose friendship relations are studied; they can be firms, for which the commercial relationships are of interest, or authors of papers for whom the citation frequency is studied.

In this study it is assumed that the ties are directed from an actor to another, and not that the relationship is both ways. Considering the pupils example, it means that if one pupil names another as his/her friend, this does not mean that the second pupil will also name the first as his/her friend; this second actor, in fact, could even dislike the first.

This structure, form a mathematical point of view, is a directed graph, or digraph, where the actors are represented by nodes and the ties between the nodes indicate the presence or absence of a social linkage. The nature of the relationship between the agents leads to structural changes along time which, in turn, affect the actors; this can be interpreted as a mutually dependent feedback process. Among the various aspects that can lead a social network to change, we can consider the individual's characteristics, the network characteristics, the pairs of actors' characteristics and a residual random influence.

The individual's characteristics can lead the network to change because similar actors are more likely to be interested in each other's friendship than two actors that have less in common; this effect is usually called *"selection"*. The network characteristics can have an effect as well, e.g. friends of friends becomes friends [2]; this means that the network itself can lead the actors to create or withdraw ties: this effect is usually called *"influence"*. Hence, in this perspective, it is possible to state that the actors will be either conscious or unconscious in the creation or withdrawal of a tie; if the actor's characteristics lead to the action that modifies the network, this choice will be conscious; on the contrary, if the network's characteristics will lead it, the choice will be unconscious for the actor. Moreover, not only individual's characteristics, but also the pairs of actors' characteristics can be informative about the creation or withdrawal of a tie. The random influences can be seen as aspects that cannot be observed about the actors or random events that lead two actors to create or withdraw a tie.

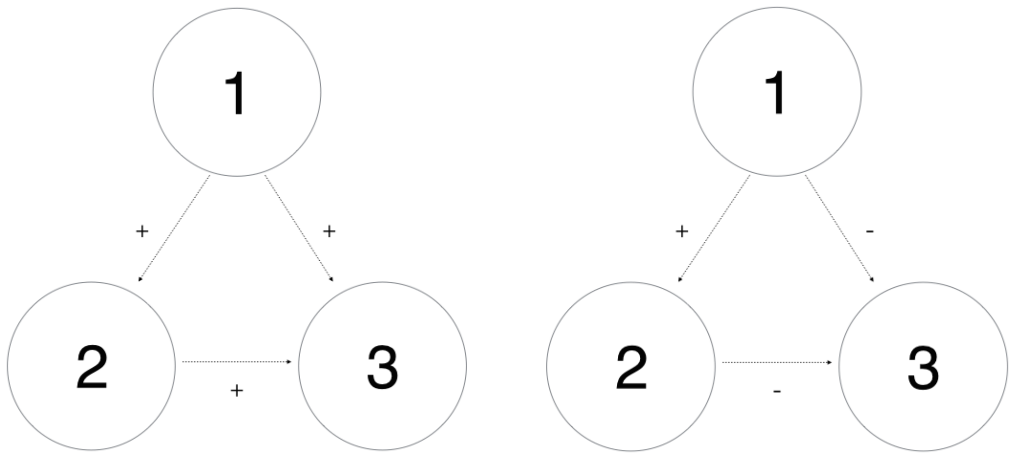
The relationship between selection and influence is one of the most interesting aspects in the study of the evolution of a social network, and will be exhaustively covered in the next sections. «In order to distinguish between this two effects, longitudinal data needs to be collected, which at least conceptually opens up the opportunity to study both the processes in parallel.»(Veenstra et all. 2013,[2]). Longitudinal data consists in repeated observations of the network at different time points, including the collection of the behavioural covariates of the actors. This means that, in order to be able to study the network in this manner, at least two observations of the whole network will be needed. The covariates could be, e.g., the age of the pupils or the family environment in which they live, the smoking attitude for older individuals, or the number of competitors for a firm.

This work focuses on the triads, i.e. the interaction within groups of three individuals. The following steps will be the introduction of the question of interest on which this work is oriented and the analysis without the use of network-specific models. Afterward the attention will be focused on the evolution of the network and on a network-specific model. At last, the network-specific model introduced will be employed to propose an answer to the question of interest.

2 Cognitive Dissonance and Balance Theory

The starting point is Fritz Heider's balance theory on cognitive dissonance. This study will focus on two individuals which are socially related, i.e. like or dislike the each other, and in in this condition form a positive or negative attitude towards a third individual.

The balance theory distinguishes between balanced triads and imbalanced triads [3]. A triad is said to be balanced when two individuals like each other and agree in their attitude towards the third individual, or when they dislike the each other and disagree in their attitude towards the third individual. Instead a triad is said to be imbalanced when they like the each other but they disagree in attitude, or when they dislike the each other and agree in attitude. «According to Heider, humans prefer the cognitive balance. They experience psychological distress when they are in an imbalanced situation and try to create a balanced one by either changing their attitude to the object, or their relationship to the other individual. »(Steglich and Niezink,[3])



(a) Balanced triad named $Happy_1$, in which two individuals share the same positive attitude they have with a third individual

(b) Balanced triad named $Happy_2$, in which two individuals in a positive relation share a negative attitude towards the third individual

Figure 1: Balanced conditions $Happy_1$ and $Happy_2$

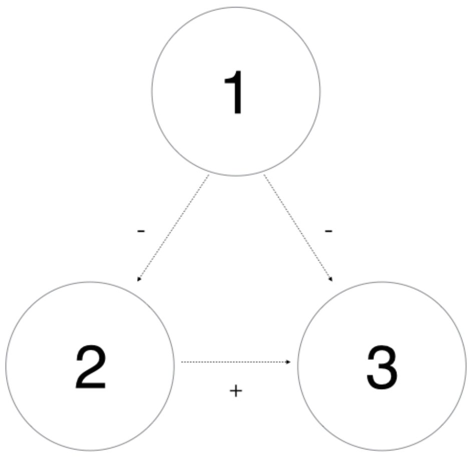


Figure 2: Balanced triad named $Happy_3$, in which two individuals in a negative relation disagree in opinion towards a third individual

In this study three kinds of balanced triads have been identified, and named $Happy_1$, $Happy_2$ and $Happy_3$. In Figure 1a it is possible to observe the first kind of balanced condition, $Happy_1$, looks like in a directed network representation where the arrows represent the relations and the nodes represent the actors. In this triad the individual marked as "1" is said to be *affected*: he is the one enjoying the status of balance, as the arrows show in the figure. In Figure 1b it is possible to see the second kind of balanced conditions, in which individual "1" has a positive relation with individual "2" and they share the same, negative, opinion toward individual "3"; in this triad, again, individual "1" is *affected.*

Figure 2 shows the last possible kind of Heider's balanced conditions in triads, i.e. the condition in which the individual "1" is affected by the disagreement with individual "2" about the attitude towards individual "3", where with the individual "2" there is a negative attitude.

As for the balanced triads, the imbalanced ones have been divided into three different kinds: $Dissonant_1$, $Dissonant_2$ and $Dissonant_3$. In Figure 3 it is possible to observe the first example of imbalanced conditions: the "affected" individual, marked as "1", is suffering the fact that he/she is not sharing the same attitude towards individual "3", with individual "2" which is considered as a friend.

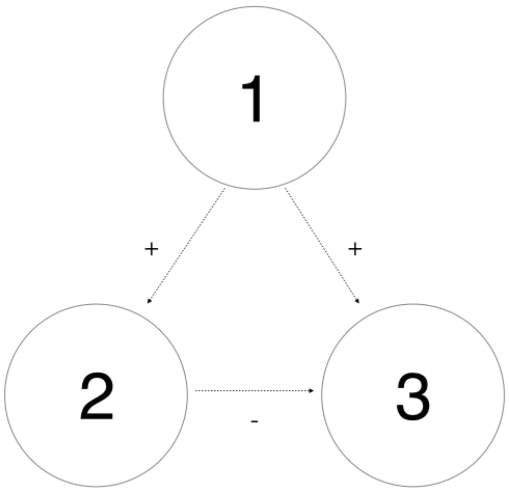
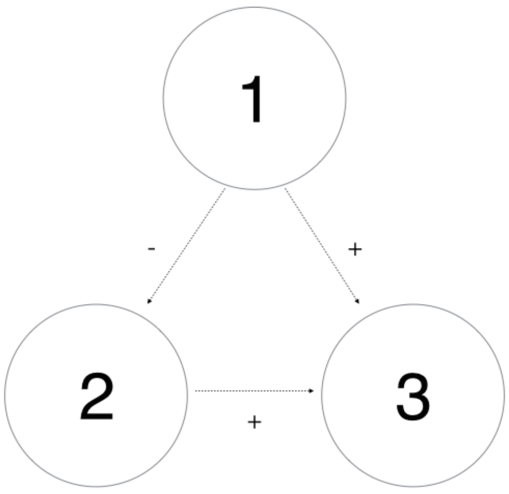
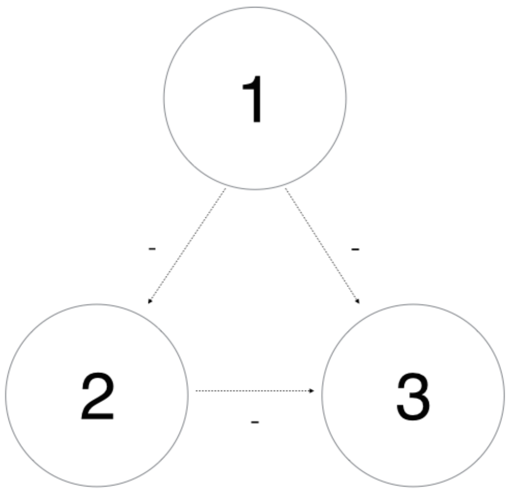


Figure 3: Imbalanced triad named $Dissonant_1$, in which the affected individual disagrees in opinion with an individual towards whom there is a positive attitude



(a) Imbalanced triad named $Dissonant_2$, in which there is agreement in positive attitude toward a third individual with someone "disliked"



(b) Imbalanced triad named $Dissonant_3$, in which there is agreement in negative attitude toward a third individual with someone "disliked"

Figure 4: Imbalanced triads in which the affected individual agrees in opinion with an individual towards whom there is a negative attitude

The other two kinds of imbalanced conditions are shown in Figure 4, and they share the commonality that both are based on an agreement attitude with a "disliked" individual. For consistency reasons, here too the *affected* individual is "1"; thESE two kinds of imbalanced triads were kept distinguished because, even though they are both based on agreement with someone "disliked", the $Dissonant_2$ condition could be considered better than the one in $Dissonant_3$. This is due to the fact that in the latter all the relations are negative, whereas in the other, two of the arrows are positive. Since the aim is to understand if these different conditions affect the psychological distress, it is reasonable to think that different amounts of positive relations, although in an imbalanced condition, can differently affect the individuals.

Starting from Heider's assumption, the aim of this study is to verify, on network data, the effect of the increase or decrease of the quantity of such triads on the individuals' stress level. The statistical tools used in the first analysis will not be network-specific, but simple as the linear regression model, with mixed effects in order to take into account all the individual's differences, and generalised additive models.

2.1 The Data

The data hereby introduced will be used along all the study: first with simple models in order to understand what to expect from the balanced and imbalanced triads, later for describing the network as a stochastic process and finally to thoroughly study the network behaviour.

The data is a panel in three waves, collected in a secondary school in Northern Netherlands during the school year 2013/2014. All the 125 pupils belong to the third year, therefore they are between 13 and 16 years old, with an average of 14.3; 64 of them are male and 61 female. The questionnaire's administration was repeated 3 times (November 2013, February 2014 and May 2014). In each questionnaire, in addition to the network status, it was collected information about the stress level of the pupils, their family status, their hobbies and about their feelings in the past two weeks. Of the 125 pupils participating to the study, 106 were present in all three questionnaires, and of these 106 only 86 answered to both the network and stress level questions; when, on the stress level measurement section, the missing data were negligible they were replaced with the mean of the non missing answers.

The networks collected in each questionnaire are formally two, one with the friendship relations and one with the dislike relations; solely for the purpose of this section the two networks were overlapped in order to work with one simple structure summarising all the types of relations. Each pupil was asked to nominate in-between 1 and 20 other pupils as friends and the same was asked for the ones he/she dislikes. The network is structured in such a way to be a directed graph, i.e. the relationships are not both ways; if one pupil names the other as a friend it does not necessarily mean that this second pupil will name the first as a friend.

As previously mentioned, three time points are available, which can be seen as different observations in a panel study. This panel is not balanced due to the missing answers, with the consequence that it is not possible to follow the behaviour of all the participants through time; the incomplete paths will be used as much as possible. If the second time point is missing they are basically useless, but if, e.g., only the first or the last time point are missing the evolution can still be studied.

In this section of the study, the number of each balanced and imbalanced condition is compared to a happiness proxy variable in order to understand the relationship between the two. The happiness proxy variable was measured with Likert-scale questions on the agreement with a sentence, on a scale from 1 to 5; the more points a person has, the happier he/she is. The questions used for this variable are:

- I feel comfortable with my friends
- I feel I am part of this school
- Sometimes I feel I do not belong to this school
- I feel very different from most other students at my school.
- I can be myself at my school

- I am proud to belong to this school
- Other students at my school like me as I am

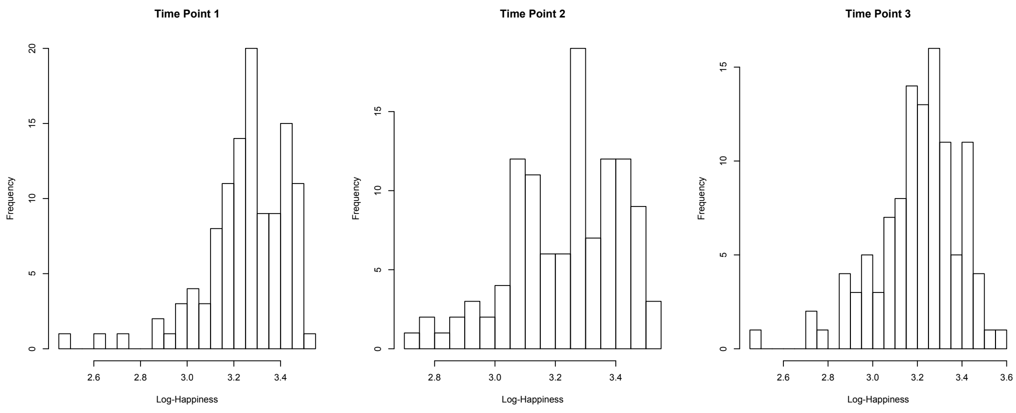


Figure 5: Histogram of the logarithm of the Happiness variable in three time points

The sentences with negative meanings are scaled back to allow a comparison with the positive ones, by subtracting to 6 the values of the answers. The Cronbach alpha was used in order to measure the reliability of the data; its value is between 0.77 and 0.78 for all the time points, which indicates a fair internal consistency.

The Happiness variable, since always greater or equal to zero, after a suitable rescaling, was used in a logarithmic transformation in order to smooth all the peaks and to treat them linearly. Even after the logarithm transformation, which should take care of unusual variance behaviours, the data is asymmetric and it is therefore not possible to state that it has a normal distribution as shown in Figure 5. Furthermore, the nature of the data suggests intra-individual autocorrelation, due to the fact that the happiness level on a time point is likely to depend on the previous happiness status. Moreover, the intra-individual correlation could be also caused by the fact that each person can react differently to changes in their friendship relations. This is shown in Figure 6, where the log-Happiness (from now on *Happiness*) is plotted against itself at different time points.

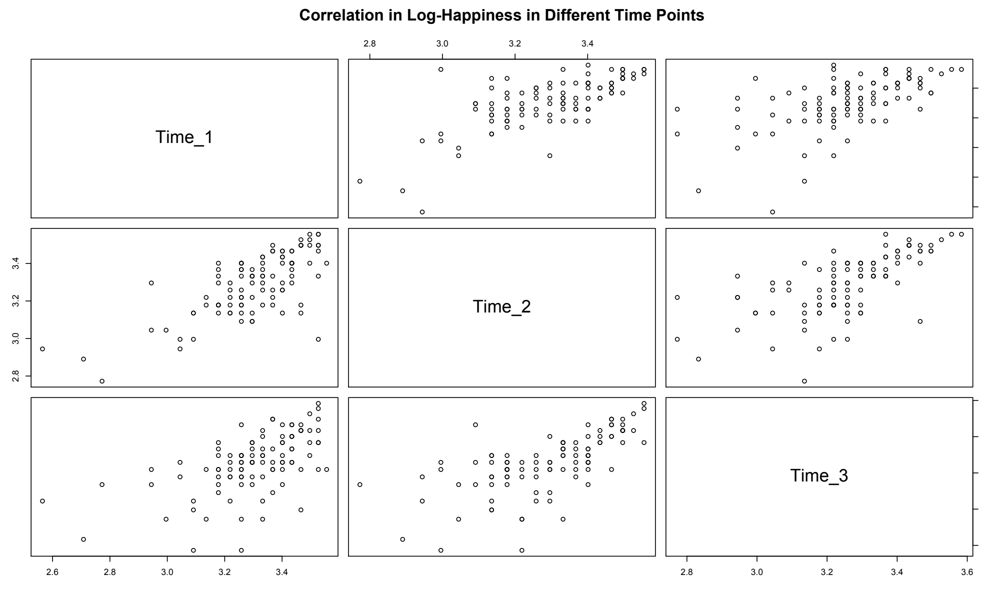


Figure 6: Autocorrelation within the interest variable at different time points: the positive correlation is clear between successive observation and it lightly scatters as time passes between the observations

The correlation between successive time points is high and positive, while, when the time difference is greater, it tends to be less obvious: this could mean that happiness is autocorrelated mostly to its first lag. These aspects will be taken into account in the models.

As previously mentioned, the triads considered are of two kinds, balanced and imbalanced, and both of them are split in three sub-categories; in the balanced condition they are:

- $Happy_1$ is a triad in which there is agreement in positive opinion toward the third person between two individuals with a positive attitude
- $Happy_2$ is a triad in which there is agreement in negative opinion towards the third person between two individuals with a positive attitude
- $Happy_3$ is a triad in which there is disagreement in the opinion towards a third person between two individuals with a negative attitude

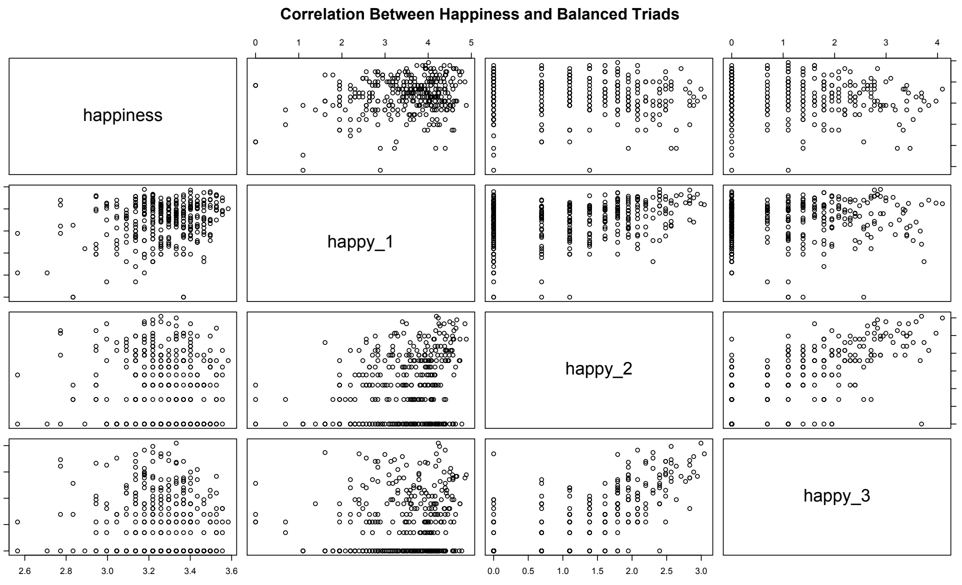


Figure 7: Correlations between all the balanced triads and the *Happiness* variable

In Figure 7 it is possible to observe that the variable of interest, *Happiness*, seems to be positively correlated with $Happy_1$ triads; the positive correlation is also present with the other two variables, but in a less obvious way. Furthermore, it is worth mentioning that $Happy_1$ and $Happy_2$ seem to be positively correlated to each other, as well as $Happy_2$ and $Happy_3$. This could be related to the fact that all the individuals in $Happy_1$ conditions belong to a same group of friends, and are therefore more likely to like each other. On the other hand, $Happy_2$ and $Happy_3$ conditions could be originated from the meeting different groups of friends and, therefore, are more likely to dislike each other.

The imbalanced condition sub-categories are the following:

- $Dissonant_1$ is a triad in which there is disagreement in the opinion towards a third individual between two individuals with a positive attitude

- $Dissonant_2$ is a triad in which there is agreement on positive attitude toward a third individual between two individuals with a negative attitude

- *$Dissonant_3$* is a triad in which there is agreement on negative attitude toward a third individual between two individuals with a negative attitude

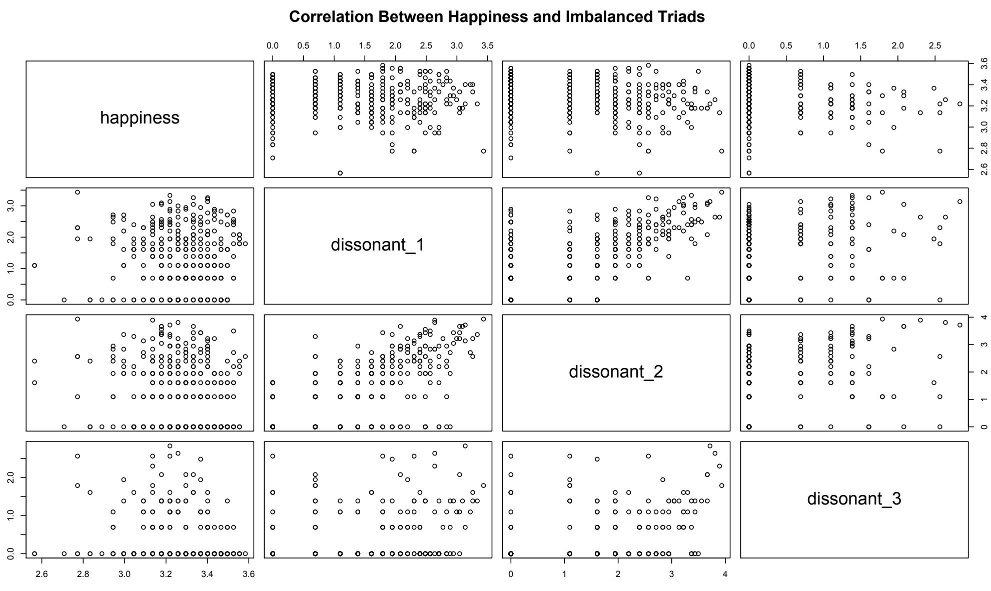


Figure 8: Correlations between all the imbalanced triads and the *Happiness* variable

Figure 8 shows the correlation between the interest variable, $Happiness$ and the imbalanced triads. The correlation is not obvious as in the previous plot, but, excluding the individuals with little amount of imbalanced condition, there seems to be negative correlation with the interest variable. In this case, the correlation between $Dissonant_1$, $Dissonant_2$ and $Dissonant_3$ can not be considered as a consequence of inclusion/exclusion from groups, but it probably derives from the "competition" for entering into a group or the process of exiting from one of those.

In order to get an insight about how triads change along time, it is possible to observe their evolution in Table 1 and Table 2, the highest values are on the diagonal, which means that the triads have not changed, or on the bottom line, which means that in the second observation it is not a triad anymore.

Wave 1 \ Wave 2	$Happy_1$	$Happy_2$	$Happy_3$	$Diss_1$	$Diss_2$	$Diss_3$	$Non\ Triads$
$Happy_1$	2111	3	2	29	37	0	2796
$Happy_2$	7	60	0	14	7	1	179
$Happy_3$	4	0	86	6	1	0	287
$Dissonant_1$	26	12	2	72	38	1	379
$Dissonant_2$	29	5	5	35	90	0	394
$Dissonant_3$	0	0	7	0	0	4	18
$Non\ Triads$	2310	368	576	440	523	114	

Table 1: Transition matrix comparing Time Point 1 vs Time Point 2 and all possible triads evolution; *Non-Triads* means that one or more ties were withdrew and the individuals do not belong to a triad anymore.

Wave 2 \ Wave 3	$Happy_1$	$Happy_2$	$Happy_3$	$Diss_1$	$Diss_2$	$Diss_3$	$Non\ Triads$
$Happy_1$	2237	4	3	29	58	0	2157
$Happy_2$	0	132	0	8	12	2	302
$Happy_3$	0	0	197	5	11	3	464
$Dissonant_1$	16	21	9	107	74	0	388
$Dissonant_2$	13	17	18	50	182	1	432
$Dissonant_3$	0	1	0	0	0	16	103
$Non\ Triads$	2387	329	423	420	656	100	

Table 2: Transition matrix comparing Time Point 2 vs Time Point 3 and all possible triads evolution; *Non-Triads* means that one or more ties were withdrew and the individuals do not belong to a triad anymore.

2.2 The Models

2.2.1 The mixed effect model

The mixed effect model is an extension of a linear model commonly used in the panel data analysis [5]. This kind of models allow to treat in a different way the effects due the to different individuals considered and the different time points. A general formulation of the model can be written as follows:

$$\begin{aligned} Y_{it} &= \beta_0 + BX_{it} + v_{it} \\ v_{it} &= \mu_i + \lambda_t + u_{it} \\ u_{it} &\sim IID(0, \sigma^2) \end{aligned} \tag{1}$$

Examining Equation 1, it is possible to notice that both the response variable and the explanatory ones are both time and individual dependent. The term of main interest now is:

$$v_{it} = \mu_i + \lambda_t + u_{it}$$

- μ_i are the "individual effects" which are the deviation from the mean due to the effect of each individual. If estimated, they consist in $N-1$ parameters, where N is the number of observed individuals in the panel.
- λ_t are "time effects" which are the deviation from the mean due to the fact that the panel has different time points. If estimated, they consist in $T-1$ parameters, where T is the number of time points.
- u_{it} is the "idiosyncratic component" and it is the generic error term, which is not due to the time effects or the individual effects.

Now that the model is introduced, before examining in detail the nature of μ_i and λ_t, some choices are to be made; they can be summarised in:

- Can the parameters be considered invariant along time?
- Can the parameters be considered invariant among individuals?

They are basically questions about the nature of the terms μ_i and λ_t: should they be considered as random variables or as parameters to be estimated? In order to understand if the parameters in B have to be treated differently for every time point, can be done a statistical test: a comparison between the complete model, the one with all the B for each time point, and the one with B kept constant for all the time points jointly. In this case, the F-test does not reject the null hypothesis, hence the reduced model can be considered as sufficient for the analysis.

On the individual side, considering the data (only 3 observations of 121 individuals), the second option is not acceptable since it would be inappropriate to estimate so many parameters in order toproperly treat this component. It is therefore more suitable to treat the individual component as a random effect, having hence:

$$\mu_i \sim IID(0, \sigma_\mu^2)$$

The final model will be:

$$\begin{aligned} Y_{it} &= \beta_0 + BX_{it} + v_{it} \\ v_{it} &= \mu_i + u_{it} \\ where\ u_{it} &\sim IID(0, \sigma^2)\ ;\ \mu_i \sim IID(0, \sigma_\mu^2) \\ with\ u_{it} &\perp \mu_i \end{aligned} \tag{2}$$

The model, estimated with R (package *plm*), is summarised in the following box. It can be observed that the individual component has a share of about 64% of the total variance and the idiosyncratic component has the remaining 36%. The model, computed on the logarithm transformation of all the variables, is statistically useful since the joint test of nullity for the parameter is widely refused. This model, moreover, explains the 55% of the total variance of the data, thanks to the treatment of the individual component.

```
Oneway (individual) effect Random Effect Model
   (Swamy-Arora's transformation)
Call:
plm(formula = form, data = panel, effect ="individual",
model = "random", random.method = "swar")

Unbalanced Panel: n=121, T=1-3, N=336

Effects:
                   var   std.dev share
idiosyncratic 0.009329 0.096588 0.361
individual    0.016511 0.128493 0.639
theta    :
   Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
 0.3991   0.6019   0.6019   0.5921   0.6019   0.6019
```

```
Residuals :
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.45800  -0.05010   0.01230   0.00086   0.07080   0.21400

Coefficients :
                Estimate  Std. Error  t-value   Pr(>|t|)
(Intercept)    3.1437358   0.0363632  86.4537  < 2.2e-16 ***
happy_1        0.0539704   0.0112296   4.8061   2.35e-06 ***
happy_2        0.0156878   0.0133551   1.1747    0.24098
happy_3       -0.0078647   0.0107661  -0.7305    0.46560
dissonant_1   -0.0243914   0.0122831  -1.9858    0.04789 *
dissonant_2   -0.0161602   0.0090664  -1.7824    0.07561 .
dissonant_3   -0.0087444   0.0145028  -0.6029    0.54696
time2         -0.0066872   0.0138907  -0.4814    0.63054
time3         -0.0353904   0.0143473  -2.4667    0.01415 *

Signif. codes:  0'***'0.001'**'0.01'*'0.05'. 0.1' '1
Total Sum of Squares:    6.7579
Residual Sum of Squares: 3.0672
R-Squared        :  0.55114
Adj. R-Squared  :  0.53638
F-statistic: 49.18 on 8 and 327 DF, p-value: < 2.22e-16
```

Model's Findings

Since the data is an unbalanced panel the use of a particular method ($SWAR$) is mandatory in order to estimate the variance; this solution is preferable to eliminating some observations to obtain a balanced panel. Considering that the variation range of the response variable is around 1, it is possible to proceed with the analysis of the coefficients estimation. $Happy_1$ has a high positive value, an $Happy_1$ triad brings 5% of happiness, and this parameter is statistically different from zero. $Happy_2$ and $Happy_3$, on the other hand, are not statistically different from zero, which allows to state that their effect is not relevant for the study. Moreover, it is possible to notice that two of the coefficients of the balanced triads are positive and that the last one ($Happy_3$, which represents disagreement with someone disliked) is not. These kinds of triads are difficult to analyse because, although the disagreement can balance the condition, it is not empirically proven whether it is the disagreement with someone "disliked" or the negative attitude itself that creates the distress condition.

The parameters of the imbalanced condition are all negative, and two out of three are statistically different from zero. $Dissonant_1$, disagreement with someone liked, can be considered as a stressing condition, since the parameter is negative and consequently the presence of this kind of triads negatively affects the response variable. $Dissonant_2$, agreement with someone disliked, can be interpreted as the previous parameter with the only distinction that its value is smaller. The last imbalanced condition found in the theory is $Dissonant_3$, where all the individuals dislike the each other. This parameter is highly non significant which allows to state that perhaps, in a condition where between two individuals there is a negative attitude, it does not significantly affect the overall happiness if the third individual is disliked.

The last parameters concern the time points and were introduced in order to evaluate if they can be considered as homogeneous. The first and the second time point can be considered homogeneous since the coefficient for the dummy variable is not statistically different from zero. The first time point and the third, however, are not homogeneous; this could be a consequence of the fact that longer time passed between the first and the last administration of the questionnaire. This result does not significantly affect the use of common parameters for all the time points since the "poolability" test carried out in the first part of the analysis allowed such a treatment.

Figure 9 shows how the sample residuals distribute with respect to the theoretical ones; the model does not estimate accurately the extreme values, which are distant from a normal distribution, but it fits accurately the central values. The bias of the model is that all the extreme values are underesti-

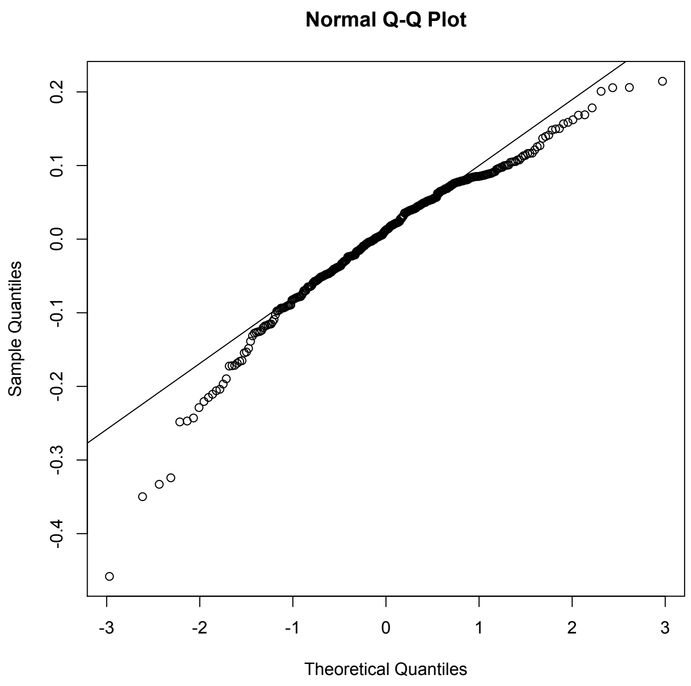


Figure 9: Quantile-Quantile plot of the mixed effects model residuals, where the normality is not completely fulfilled since all the extreme values are underestimated

mated as all of them are under the theoretical distribution line. In Figure 10 it is possible to observe how the residuals distribute with respect to every response variable, and it does not appear any systematic behaviour in it.

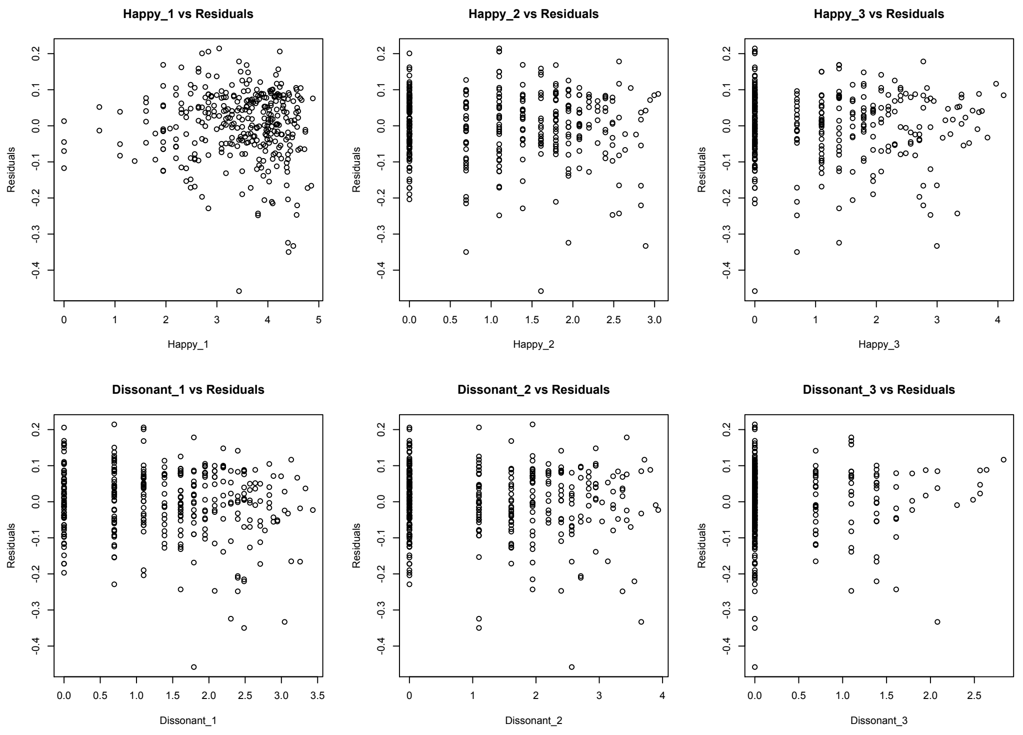


Figure 10: Residuals versus all the explanatory variables included in the model, exception made for the time dummy variables; does not seem to be systematic

2.2.2 The generalised additive models

The second family of models applied is the generalised additive models (GAM); this kind of models consist of fitting a predictor for each variable and collecting in an additive fashion all the outcomes [4]. This way of acting has less model-specific constraints than the linear model and will be able to collect more variable-specific behaviours. The model can be formally written as follows:

$$\begin{aligned} Happiness = \alpha_0 &+ f_1(Happy_1) + f_2(Happy_2) + f_3(Happy_3) + \\ &+ f_4(Dissonant_1) + f_5(Dissonant_2) + f_6(Dissonant_3) + \\ &+ Time + \varepsilon \end{aligned} \tag{3}$$

The $f(x)$ functions in Formula (3) are cubic regression splines. This kind of functions are piecewise third degree polynomials. One of the tuning parameters for this estimation procedure is the number of polynomials to use or, more precisely, the number of knots in which the basis functions will match to create a unique polynomial function. For the purpose of this study, the cubic splines will be fitted to the data through regressions spline. The dummy variables in this kind of models will affect the smooth polynomial fitting the data, as a baseline effect, rising or reducing their height. In this kind of models the number of parameters and, hence, of splines fitted is significantly important and it is not possible to use a tool allowing to differentiate all the individuals. Although this possibility is not available, it is still worth to consider this smooth fitting in order to understand the overall behaviour of the balanced and imbalanced triads with respect to the Happiness dependent variable chosen. The following box shows the summary of the model:

```
Family: gaussian
Link function: identity

Formula:
happiness~s(happy_1, bs="cr")+s(happy_2, bs ="cr")+
+ s(happy_3, bs="cr")+s(dissonant_1, bs="cr")+
+s(dissonant_2, bs="cr")+s(dissonant_3,bs="cr")+time

Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.2815225  0.0153764 213.412   <2e-16 ***
time2       -0.0004642  0.0218472  -0.021    0.983
time3       -0.0265384  0.0222177  -1.194    0.233
---
Signif. codes:  0'***'0.001'**'0.01'*'0.05'.'0.1' '1
```

```
Approximate significance of smooth terms:
                    edf Ref.df      F  p-value
s(happy_1)       1.7457  2.181 14.432 5.41e-07 ***
s(happy_2)       1.0087  1.017  0.489  0.48707
s(happy_3)       0.9998  1.000  0.086  0.76891
s(dissonant_1) 1.3230  1.568  0.673  0.45974
s(dissonant_2) 0.9998  1.000  6.885  0.00909 **
s(dissonant_3) 1.0000  1.000  1.344  0.24708
---
Signif. codes:  0'***'0.001'**'0.01'*'0.05'.'0.1' '1

R-sq.(adj) =  0.139   Deviance explained = 16.2%
GCV score = 0.026222  Scale est. = 0.025436  n = 336
```

Model's Findings

This model has an explanatory capacity of about 16%; the difference with the adjusted R-squared could be interpreted as an high consumption of degrees of freedom due to the number of knots used in the splines fitting (ten each). Nevertheless, using ten degrees of freedom seemed to be the best compromise between smoothness and precise fitting.

In this model, as in the one introduced before, the intercept has an important role; however the time dummy variables lose their meaning confirming the fact that the three time points can be considered as homogeneous. For what concerns the balanced triads variables, only $Happy_1$ has a statistically significative effect, and in the imbalanced condition the same is true for $Dissonant_2$.

In order to understand the behaviour of the regression splines used in the model, it is useful to refer to Figure 11; since this kind of models matches different polynomials, a convenient way to interpret it is to observe how the matched polynomial looks like. The outcome of the model will be the sum of these behaviours. It is possible to observe that $Happy_1$ splines have a

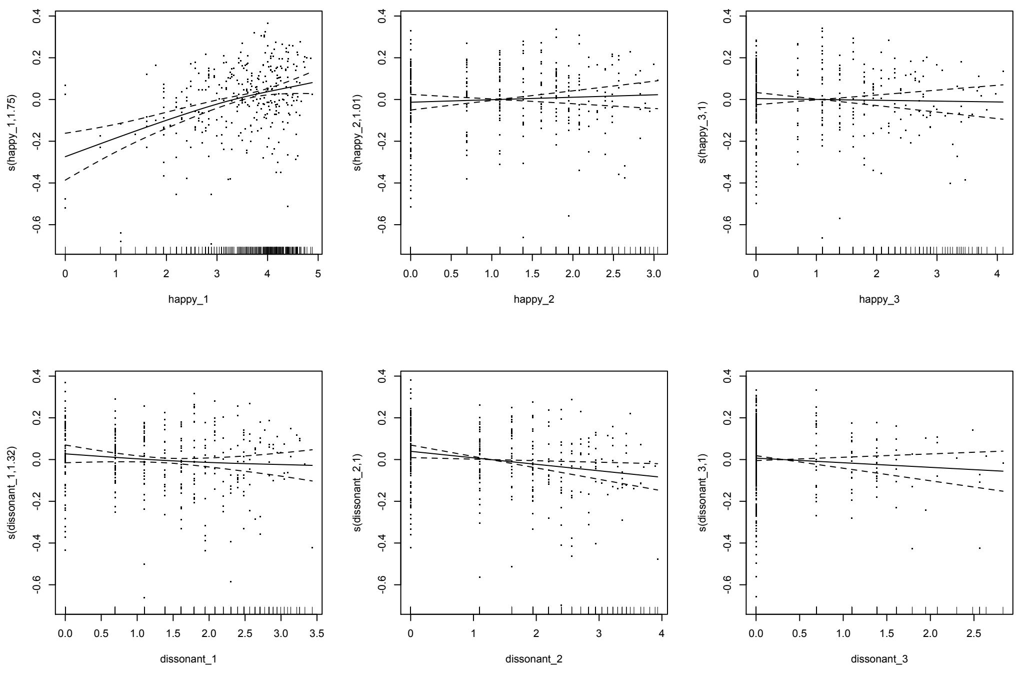


Figure 11: Gam behaviour spilt in each component (explanatory variable), the result of the model is the sum of all these behaviours

positive effect on the happiness level. $Happy_2$ and $Happy_3$ do not seem to have any relevant effect on the dependent variable in this model; this was noticed as well in the summary of the model, in which those splines were not significantly different from zero. The splines referring to the imbalanced condition seem all to have a negative effect, but only $Dissonant_2$ is actually significative.

In the Quantile-Quantile plot of Figure 12 it is possible to observe that the GAM model as well underestimates the extreme values of the dependent variable, but it seems to be more consistent with the high values compared to the mixed effect model; in any case, the distribution is not totally satisfying the normality assumption.

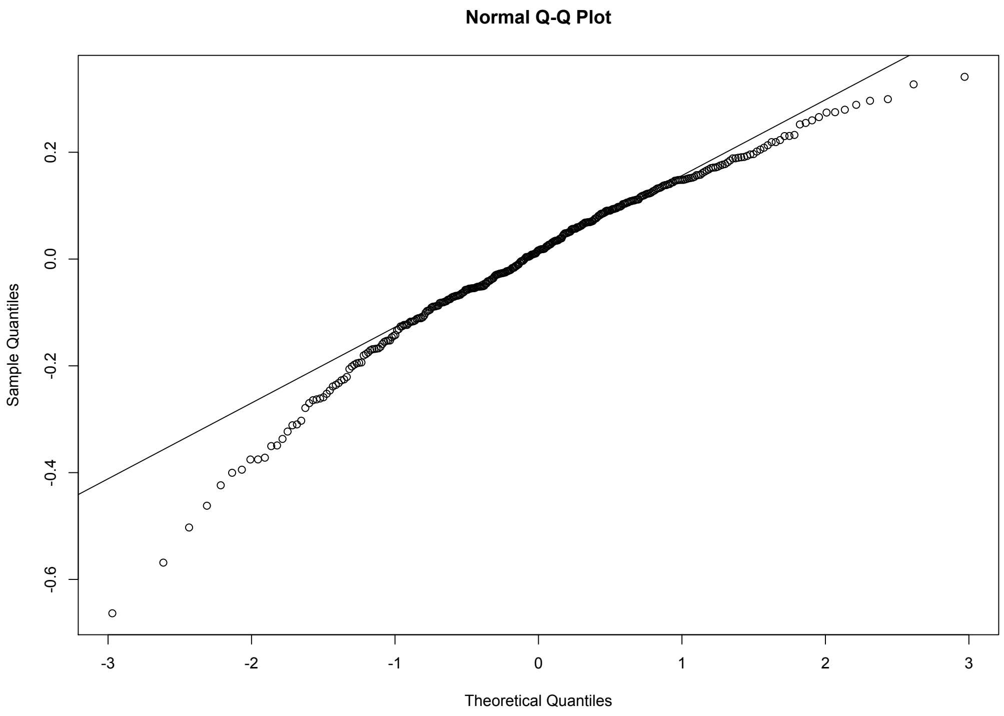


Figure 12: Quantile-Quanitle plot of the GAM model's residuals, there is an underestimation of the extreme values

2.3 Conclusions

The initial question was whether balanced and imbalanced conditions can increase the happiness level in the individuals. What the models have shown is that, on average, the presence of balanced triads can lead to an increase of happiness. Both models agree on the fact that only $Happy_1$ condition has a significant effect on the happiness level. Consequently it seems that only the agreement in opinion between two individuals that like each other has an overall effect, whereas the other balanced conditions, e.g. the disagreement with someone which is disliked, do not affect the happiness level.

The imbalanced condition, as expected, has a negative effect on the happiness of the average individual. $Dissonant_2$ was selected by both models to have negative and significant effect on the happiness level; $Dissonant_1$ was found significant only in the mixed effect model, although had a negative effect in both the models; it therefore seems adequate to consider it as an interesting variable. The $Dissonant_3$ was not significative in any models; it will therefore not be considered in the future. The variables that resulted significant will be introduced in the network study and the results will be compared with the ones of this first step.

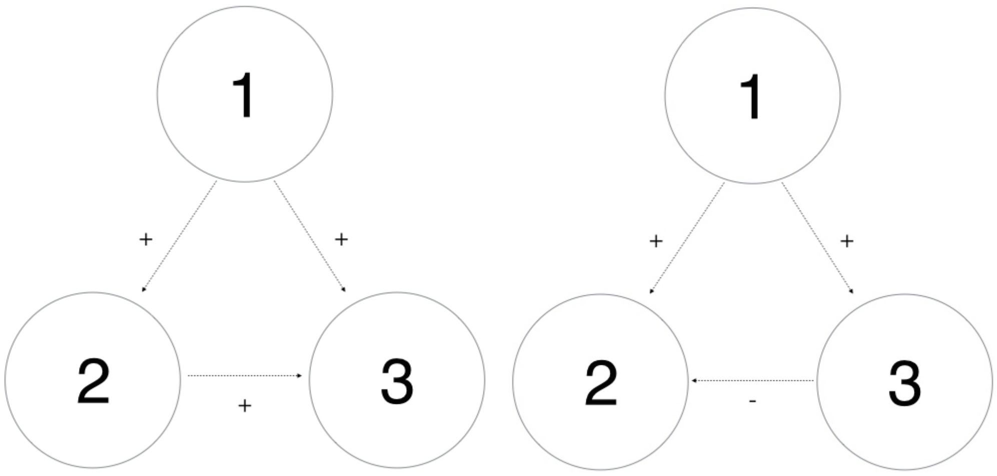
2.3.1 Discussion

Only three time points were available for this analysis. Although they seemed to be homogeneous, this does not necessarily mean that different time points would lead to the same results. It could be possible that, if more time points are available, two distant observations of the network, as well as the parameters of the independent variables, are not statistically homogeneous.

However, having more observations of the whole network could lead to a more precise estimation, and also, at some point, to estimate the μ_i effects in the mixed effects model. This estimation would lead to a model that can find the overall behaviour with regard to the independent variables and at the same time take into account the differences of the actors involved in the study.

The lack of more observations and the use of few explanatory variables of interest led the residuals not to have a normal distribution, as assumed from the models. In the linear model quantile-quantile plot it is possible to observe that some residuals in the margins of the plot are far from the theoretical distribution; this could be due to the heteroscedasticity of the residuals, introducing in the model an underestimation of the variance of the parameters. On the other hand, the GAM model was able to collect in a more precise way the variability of the data with lower values; however, it has an underestimation problem for higher values. The logarithmic transformation seems to have a good effect on the heteroscedasticity, but at some time it does not seem to be sufficient.

Moreover, as shown in Figure 13, also the structure of the triads deserves some remarks. In both triads individual "1" is affected, with the only difference of the relation between individual "2" and "3". Figure 13a shows



(a) $Happy_1$ triad for individual "1" (b) *Dissonant* triad for individual "1"

Figure 13: Two triads with "affected" individual element "1" and composed by the same individuals, triad 13a is counted as balanced and triad 13b is counted as imbalanced

the "$Happy_1$" triad, composed by the elements "1", "2" and "3", in which individual "1" is affected. Figure 13b shows the "$Dissonant_2$" triad, as well composed by the elements "1", "2" and "3", in which individual "1" is affected. This implies that the same triad is both balanced and imbalanced. It was not possible to understand if, and to what extent, this could be misleading. One possible solution is to change the definition of the triads and consider the both-way friendship; however, using this solution the number of triads in the network would decreased significantly, therefore losing important information.

3 Networks as a Stochastic Process

A the network is defined as a set of links between some actors, i.e. a set of arrows (representing the links) connecting some nodes (representing the actors). Since these sets can vary in their composition, they can be understood as a dynamic process, whose consequent evolution can be understood as a stochastic process, although for this second step some more assumptions are required. Ties can be established, gain strength or be dissolved, and this can happen in all kinds of relation such as friendship, sales agreement or cell interaction [6]. Every change in the network will be driven by multiple factors, such as the characteristics of the considered nodes or the characteristics of the network itself. This second aspect will be analysed in the next sections since it is one of the core aspects used in modelling.

The peculiar and statistically relevant aspect of the networks is that they are not composed of independent observations, indeed all the changes in the network will cause consequent effects: creation of new ties or withdraw of the existing. It is possible to state that every change in the network depends, given all the other observed variables as fixed, on its first shape, which leads to the definition of a Markov chain.

A Markov chain is a stochastic process in which the probability distribution of future states, in this case the shape of the network, depends only from the last status. This can be easily observed in a particular kind of networks in which only one change at the time is allowed: the new state that will be reached depends solely on the actual state which, in turn, depends only on the preceding state. This assumption cannot be said as realistic but, with the available data and the complexity of the real process, it is the better available [7]. In this first introduction of the network as a Markov chain stochastic process, a simple network structure will be used: the one with dichotomous variables indicating the presence or absence of a like or friendship relation. This will later be extended to the case in which other kinds of relation are allowed, as in the multilevel network, where both likes and dislikes appear.

3.1 Definition of the process

At this point, a rigorous notation is needed in order to exactly define the process and try to explain its evolution. The network can be represented by the set of every possible tie; hence, an adjacency matrix, $X(t)$ is defined, so that it will depend on the time t in which the network is observed, with generic element $x_{i,j}(t)$. This last element will point out whether there is a relation between the two actors or not.

$$X_{i,j} = \begin{cases} 1 & \text{if there is a like relation between actor } i \text{ and actor } j \\ 0 & \text{otherwise} \end{cases}$$

$X_{i,j}(t)$ is a time dependent random variable, and it may also depend on other explanatory variables which can be either time invariant or change with the time [7].

It is now worth to further explain the distinction between the actual process that allows the network to evolve and its possible observations. The evolution of the network, as stated before, can be considered as a dynamic process that, although split in mini-steps allowing only one change at the time, is not possible to be observed directly in most of the applications. In social studies, e.g., it is not possible to administrate a questionnaire every day in order to evaluate every single change and, perhaps, even so it would not be possible to gather all the changes. The questionnaire administrations along time are few (in order to be able to work with the data at least 2) and the changes collected between the two time points are important for the study. Hopefully they are neither radically different nor considerably similar.

Once two observations of the whole network are available, it is possible to compare them and suppose that there are K changes between the two time points. What the observer can state is that there were at least $K + 2S$ with $S \in [0, 1, 2, \dots)$ changes. Suppose now that the observer wants to estimate, through a simulation, the evolution of the network between the two time points: the most straightforward way to do it is to consider a continuous time Markov chain.

As beforehand mentioned, one of the assumption of the Markov chain process is that the future only depends on the present status of the network, which can be extended to the continuous time as follows: $\{X(t) \mid t \in \mathscr{T}\}$ is a stochastic process with a finite outcome space $\mathscr{X}$; $\{X(t) \mid t > t_a\}$ depends only on the present $X(t_a)$ and with $t_a < t_b$, $x \in \mathscr{X}$ it is possible to state that:

$$P(Y(t_b) = x \mid X(t) = x(t) \; \forall \; t \leq t_a) \; = \; P(Y(t_b) = x \mid X(t_a) = x(t_a)) \quad (4)$$

The process, in this case, is only a function of the elapsed time between t_a and t_b. If $\{X(t) \mid t \in \mathscr{T}\}$ is a continuos time Markov chain with stationary transition distribution, it is possible to find the intensity matrix of the infinitesimal generator as follows:

$$\begin{aligned} q(\mathbf{x},\mathbf{y}) &= \lim_{dt \to 0} \frac{P(X(t+dt) = \mathbf{y} \mid X(t) = \mathbf{x})}{dt} \quad \text{for } \mathbf{x} \neq \mathbf{y} \\ q(\mathbf{x},\mathbf{x}) &= \lim_{dt \to 0} \frac{1 - P(X(t+dt) = \mathbf{x} \mid X(t) = \mathbf{x})}{dt} \end{aligned} \quad (5)$$

This matrix expresses the probability of each change in the network as the elapsed time tends to zero. $\mathbf{y}$ is an adjacency matrix with only one differing element form $\mathbf{x}$, an adjacency matrix as well [8].

3.1.1 Who makes the move and when: the rate function

Until now, no assumptions on the decision process for creating or withdrawing a tie were made; it will be now explained how the actors change their ties. It is natural to assume that the actor has the complete control of the ties: he/she can decide whether to create or withdraw a tie no matter if the receiving actor agrees. The instant of time in which the change happens is defined by a *rate function* which indicates how frequently the actors make a move:

$$\lambda_i(x) = \lim_{dt \to 0} \frac{1}{dt} P\left\{X_{it}(t+dt) \neq X_{it}(t) \text{ f.s. j} \in \{1, \ldots, g\} | X(t) = x\right\} \quad (6)$$

where g is the number of actors in the network [8].

In the easiest condition, it can be assumed that the changing rate is constant for all the actors, said ρ: $\lambda_i(x) = \rho \ \ \forall \ i$. It is also assumed that there is independence between the actors in the elapsed time Δt in which they make the ministep. The waiting time between successive ministeps for each actor will have an exponential distribution, hence the expected number of changes will be $g\rho(t_b - t_a)$: it depends from the number of actors in the whole network (g), the changing rate (ρ) and the interval of time considered $(t_b - t_a)$.

The easiest condition is a good starting point; however, when some covariates on the behaviour of the actors are available, it is natural to assume that those have different behaviours in terms of waiting time before the next ministep. It is useful to introduce the covariates taking into account the fact that the final rate parameter must be positive; one efficient proposal is:

$$\rho_i(\alpha, x) = \rho_m exp\left(\sum_h \alpha_h v_{hi}\right), \tag{7}$$

where V_h are all the covariates available. This rate function can be further extended in order to include further aspects of the network such as the number of outgoing ties an actor has, or the number of incoming ties, etc.; all these network dependent variables will be used as the other covariates, keeping into account that this kind of data can vary from one ministep to the other. Now that the exponential distribution for each actor has been defined, it is straightforward to define the overall process distribution. Considering $\lambda_+(x) = \sum_{i=1}^{g} \lambda_i$, it is possible to use an exponential distribution with parameter $\lambda_+(x)$ for the random variable defining the time interval Δt between the ministeps (the definition of $\lambda_+(x)$ is derived from a property of the exponential distribution, explained more extensively in section 3.2.1).

Once the process timing for the next move has been defined, the actor making the move must be decided. In the simulation process it will be chosen randomly with probability $\frac{\lambda_i(x)}{\lambda_+(x)}$.

3.1.2 Which change to make

After deciding how much time has passed from the last change and which actor will create or withdraw a tie, the next step is to make the actual change. Suppose that the chosen actor at time $t + \Delta t$ is i; there are $g - 1$ possible other actors with whom the relation can change. More precisely, for the simple network so far considered, the change between the chosen actor i and another actor j will be: $X_{i,j}(t + \Delta t) = 1 - X_{i,j}(t)$. The probability distribution for the choice of the second actor involved in the change will depend on the characteristics of the two actors and on the network status; the latter involves the network as it is at time t and how it could be at time $t + \Delta t$. The tool used in this step is called *"objective function"* which will be properly introduced later (section 4.5); at this moment, it is enough to know that it is a function in which the aforementioned data is included and weighted with the estimated parameters. The function defining the change of the relation between actor i and actor j can be written as follows:

$$f_i \left(X(i \rightsquigarrow j) \right) \tag{8}$$

In order to compute the probability, this function will be divided by the sum of all the other possible outcomes as normalising constant; moreover, an exponential transformation will be used in order to maintain all the values positive. The result is as follows:

$$P_{ij} = \frac{exp\left\{ f_i \left(X(i \rightsquigarrow j) \right) \right\}}{\sum_{h=1, h \neq i}^{g} exp\left\{ f_h \left(X(i \rightsquigarrow j) \right) \right\}} \tag{9}$$

P_{ij} gives the mass probability function for all possible js, and from this mass function the actual j which will be included in the move will be sampled. The rate parameter introduced in the previous section is usually constant between two network observations, as well as in all the ministeps incurring between the two. The objective function strongly depends on the network structure and, although the parameters in it are constant, it changes its outcome in each ministep.

3.1.3 Example

The aim of this section is to apply the discussed evolution procedure to an empirical example. For this purpose a small set of the data introduced in section 2.1 is used. The whole dataset, as mentioned, collects the data from 125 pupils. Here only a class (the second) composed by 25 of them is considered. All the outgoing or incoming ties outside of the class were ignored. The next steps will be: the representation of the data in a graph structure, the adjacency matrix and some steps of the stochastic process.

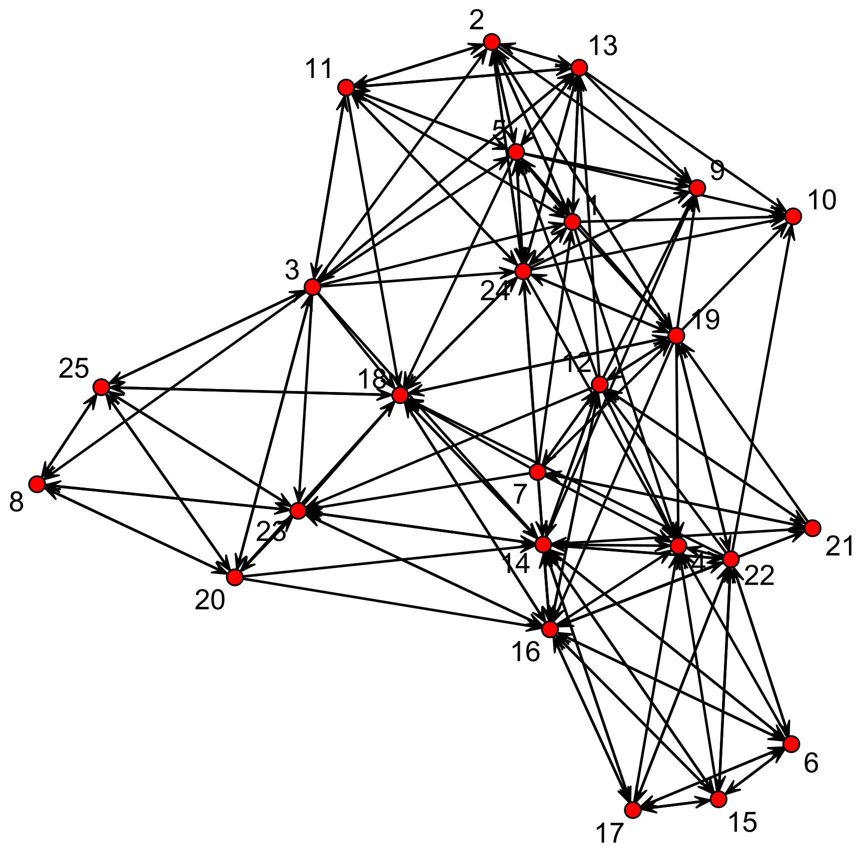
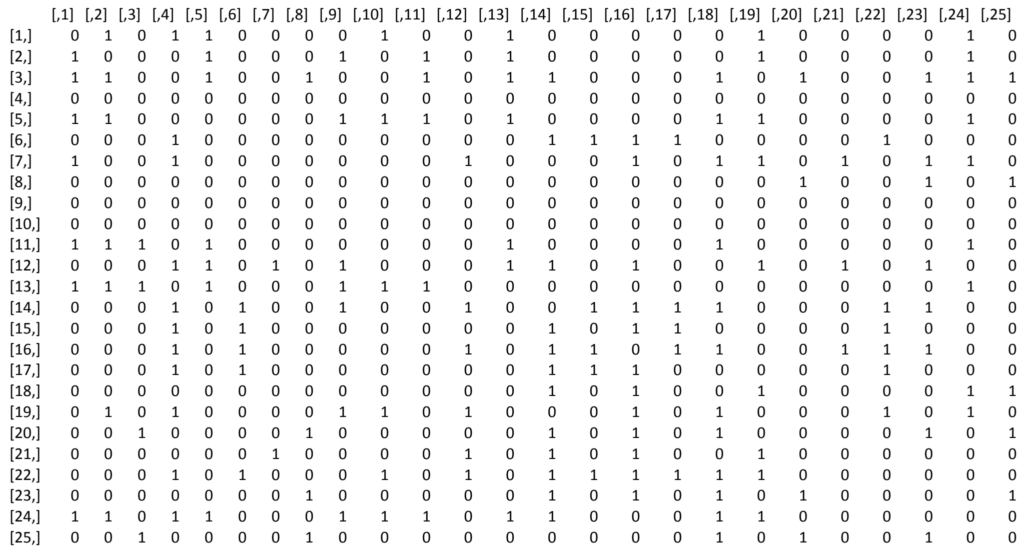


Figure 14: Network representation in graph form of pupils relations within one class

In Figure 14 it is possible to observe the representation of the simple network, with only one possible relation (friendship), where the arrows represent the relations and the nodes represent the individuals.



	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]	[,21]	[,22]	[,23]	[,24]	[,25]
[1,]	0	1	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0
[2,]	1	0	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0	1	0
[3,]	1	1	0	0	1	0	0	1	0	0	1	0	1	1	0	0	0	1	0	1	0	0	1	1	1
[4,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[5,]	1	1	0	0	0	0	0	0	1	1	1	0	1	0	0	0	0	1	1	0	0	0	0	1	0
[6,]	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	0	0	0
[7,]	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0
[8,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1
[9,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[10,]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
[11,]	1	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
[12,]	0	0	0	1	1	0	1	0	1	0	0	0	1	1	0	1	0	0	1	0	1	0	1	0	0
[13,]	1	1	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
[14,]	0	0	0	1	0	1	0	0	1	0	0	1	0	0	1	1	1	1	0	0	0	1	1	0	0
[15,]	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0
[16,]	0	0	0	1	0	1	0	0	0	0	0	1	0	1	1	0	1	1	0	0	1	1	1	0	0
[17,]	0	0	0	1	0	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0
[18,]	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	1
[19,]	0	1	0	1	0	0	0	0	1	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0
[20,]	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	0	1	0	1
[21,]	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1	0	0	1	0	0	0	0	0	0
[22,]	0	0	0	1	0	1	0	0	0	1	0	1	0	1	1	1	1	1	1	0	0	0	0	0	0
[23,]	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	1
[24,]	1	1	0	1	1	0	0	0	1	1	1	0	1	1	0	0	0	1	1	0	0	0	0	0	0
[25,]	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0

Figure 15: Network adjacency matrix

The matrix in Figure 15 is the adjacency matrix which creates the network and the process; the "1" between two individuals stands for the presence of a relation, whereas the "0" stands for absence of relation.

The next step is to simulate its evolution given some network-based information and some covariates. In order to have a reliable simulation of the process, the network was studied with the RSiena package in R, and the parameters obtained will be used for the simulation [9] [10]; the procedure for the estimation will be explained in detail in sections 4.8 - 4.10. The software estimated a rate parameter (ρ) of 4.18; consequently each individual, on average, has about 4 chances to change in the elapsed time between the two observations. Hence, the time before the next move in the network follows an exponential distribution with parameter ρ, which is constant for all the individuals. Once the *focal actor*, the actor creating or withdrawing a tie, is chosen, the function generating the discrete probability distribution for the move can be summarised as follows:

$$f_i = exp\left\{\beta_1 s_{i1}(x) + \beta_2 s_{i2}(x) + \beta_3 s_{i3}(x) + \beta_4 s_{i4}(x) + \beta_5 s_{i5}(x, w)\right\}. \quad (10)$$

The s_{ih} in Equation 10 are:

- s_{i1} is the out-degree density, which stands for the number of outgoing ties each individual has ($\sum_{h=1}^{g} x_{ih}$)
- s_{i2} is the reciprocity, which stands for the number of reciprocated relations ($\sum_{h=1}^{g} x_{ih}x_{hi}$)
- s_{i3} is the in-degree popularity, which stands for the number of ingoing ties to the other individual ($\sum_{h=1}^{g} x_{jh}$)
- s_{i4} is the out-degree popularity, which stands for the number of outgoing ties from the other individual ($\sum_{h=1}^{g} x_{hj}$)
- s_{i5} is the interaction between same gender and friendship, ($\sum_{h=1}^{g} x_{ih}w_{ih}$ where $w_{ih} = 1$ if i and h are both of the same gender)

obtaining the final formula:

$$f_i = exp\left\{\beta_1 \sum_{h=1}^{g} X_{ih} + \beta_2 \sum_{h=1}^{g} X_{ih}X_{hi} + \beta_3 \sum_{h=1}^{g} X_{jh} + \beta_4 \sum_{h=1}^{g} X_{hj} + \beta_5 \sum_{h=1}^{g} X_{ih}w_{ih}\right\}$$

and

$$P_{ij} = \frac{exp\left\{\beta_1 s_{i1} + \beta_2 s_{i2} + \beta_3 s_{i3} + \beta_4 s_{i4} + \beta_5 s_{i5}\right\}}{\sum_{z=1,z\neq i}^{g} exp\left\{\beta_1 s_{i1} + \beta_2 s_{i2} + \beta_3 s_{i3} + \beta_4 s_{i4} + \beta_5 s_{i5}\right\}} \tag{11}$$

As for the rate parameter, also $\beta_1, \beta_2, \beta_3, \beta_4$ and β_5 were estimated; they will have the following values:

Parameters	β_1	β_2	β_3	β_4	β_5
Values	-1.29	1.67	0.07	0.03	0.09

Table 3: Parameters estimation for the example network

Suppose now the the process is at time 0; the next event will happen at time Δt with $\Delta t \sim EXP(\rho)$. For the simulation purpose, a value from this distribution will be sampled. Once the time for the next event is defined, an actor will be sampled; since all the actors have the same rate function, the focal one will be extracted from a discrete uniform distribution where all the actors have the same probability. After the focal actor has been chosen, the mass probability function for the tie to change is obtained with Formula 11. Finally, the second actor in the evolution process will be sampled from this mass probability function. The process is now showed with a numerical example.

Example of one mini-step

A random time from an exponential distribution with $\rho = 4.18$ is sampled: 0.318. This is the time instant in which the first change happens, and the sampled actor is: 14. The values obtained from Formula 10 are:

```
[1]    8.70   8.70   6.11  14.06   8.20   7.82   5.83   8.72
[9]   10.71  10.00   7.59   7.43   8.44   0.00   7.82  11.18
[17]   7.82  12.97   9.39   7.75   7.03   7.43  10.47   8.85
[25]  10.08
```

Plugging in the results in Formula 11, the mass probability function obtained is:

```
[1]     0.039 0.039 0.027 0.063 0.037 0.035 0.026 0.039
[9]     0.048 0.045 0.034 0.033 0.038 0.000 0.035 0.050
[17]    0.035 0.058 0.042 0.035 0.031 0.033 0.047 0.039
[25]    0.045
```

Finally, the tie, or the missing tie, to be changed can be extracted from this discrete distribution. In this case, proceeding with the simulation, the second actor will be: 25. The change in a mathematical form is $X_{14,25}(0.318) = 1 - X_{14,25}(0)$.

The described procedure shows how the ministeps are the core of the network evolution. In order to be able to properly notice the differences between the networks, two simulations were conducted (Figure 17 - 18): the first including 10 ministeps and the second including 50 ministeps, both starting from the network in Figure 14.

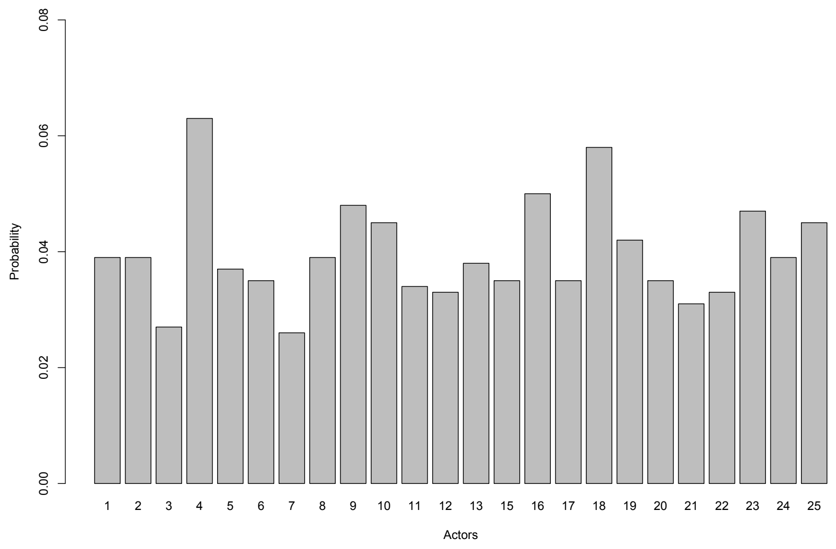


Figure 16: Mass probability function for the choice of the second individual in the network given the focal actor 14

This simulation does not take into consideration of all the aspects it should in order to show a complete representation of the network evolution. It

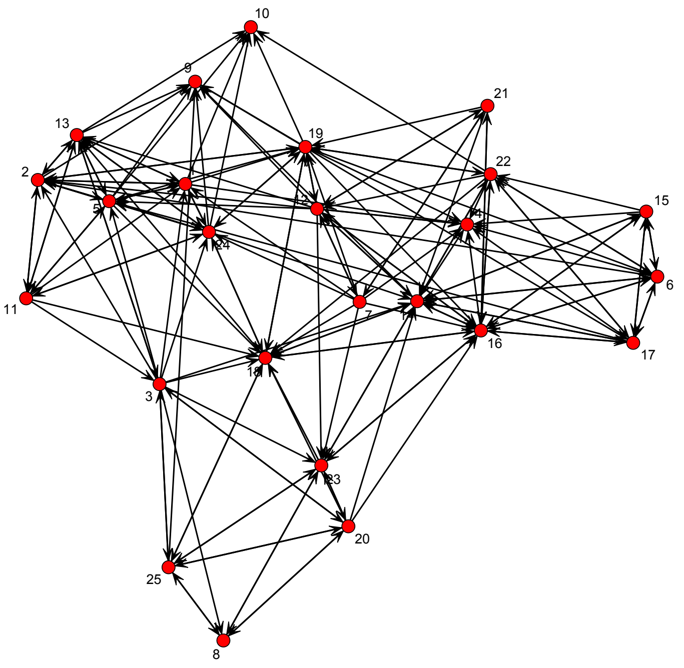


Figure 17: Graph representation of the network after 10 simulated ministeps

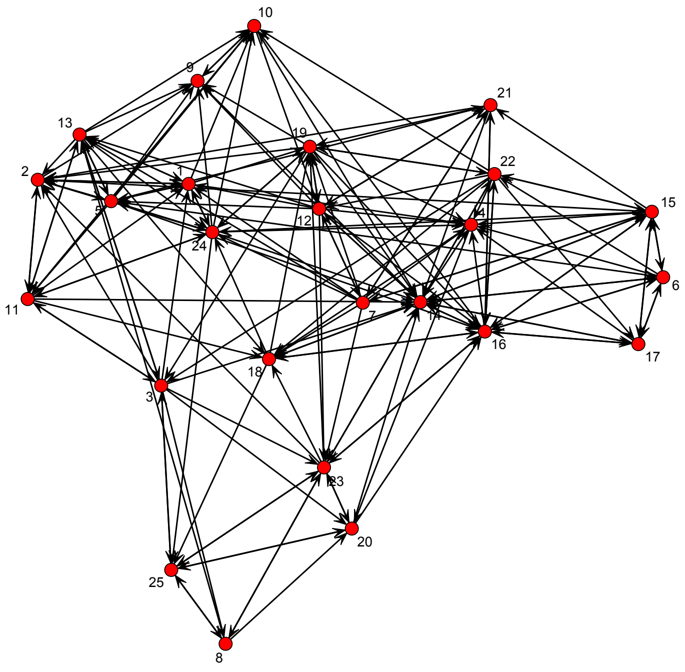


Figure 18: Graph representation of the network after 50 simulated ministeps

can be noticed that the network evolves in a compact way (no individuals are left apart), which in the social sciences field in not common. The behaviour of the individuals and further aspects of the network should be taken into account in order to exhaustively simulate the evolution. It is not worth, hence, any inference on the parameters computed so far.

3.2 Multilevel Process

Suppose now to have a more complex process to study, in which different kinds of relations are admitted within the same set of individuals. Like and dislike relations between pupils in a school compose the network hereby considered, but multilevel networks can be extended to plenty of fields. In order to make this process feasible, it is assumed that an individual can have only one kind of relation toward another individual in the same instant, e.g. either he/she likes or dislikes the other individual. This attitude can change along time, but in a single time observation only one of the two options is viable.

The data comes from two distinct networks including the same individuals; both levels are represented with the presence or absence of a specified relation. This leads to a multilevel network in which either there is a relation between the nodes or there is not and, when a relation is present, it can be of two different kinds: like or dislike.

3.2.1 Process description

In this section, the two networks will be overlapped and studied based on their characteristics. The difference with the facilitated process is that here the first decision to be made is on which level, between the two available, a change will be made. Once the level is selected, it is possible to refer to the facilitated single-level network process.

In order to chose the level of the network, a rate function which will define the time for the successive move will be used; this same function will also define the mass probability function for the level of the network. The rate function has to be defined in a more exhaustive way: it will be, as in the single-level network, an exponential distribution for both networks involved in the multilevel process. Therefore, there will be two rate parameters, one for each network: ρ_{friend} and ρ_{enemy}. The time of the next choice will have the following distribution:

$$\begin{aligned} \Delta t_{+} &= \min\{\Delta t_{friend}, \Delta t_{enemy}\} \\ \text{where } \Delta t_{friend} &\sim EXP(\rho_{friend}) \\ \Delta t_{enemy} &\sim EXP(\rho_{enemy}) \end{aligned} \tag{12}$$

The exponential distribution employed in Formula 12 has a suitable property which helps to easily manage the minima distribution:

$$\min\{\Delta t_{friend}, \Delta t_{enemy}\} \sim EXP(\rho_{friend} + \rho_{enemy})$$

Moreover it is possible to decompose the minimum as follows:

$$\text{so } Pr\left(\Delta t_{friend} = \min\{\Delta t_{friend}, \Delta t_{enemy}\}\right) = \frac{\rho_{friend}}{\rho_{friend} + \rho_{enemy}}$$

$$\text{and } Pr\left(\Delta t_{friend} = \min\{\Delta t_{friend}, \Delta t_{enemy}\}\right) = \frac{\rho_{enemy}}{\rho_{friend} + \rho_{enemy}}$$

The result is the probability of which level of the network will change in the successive ministep. The multilevel process implies two *objective functions*, both with different parameters, computed in order to better fit changes observed on each level. The two functions, at the same time, will depend on the status of both levels. Once the level is sampled from its discrete probability distribution, the linked function will be used.[11].

3.2.2 Example

Starting from the network used in the previous section, a second network with the dislike relations will be added.

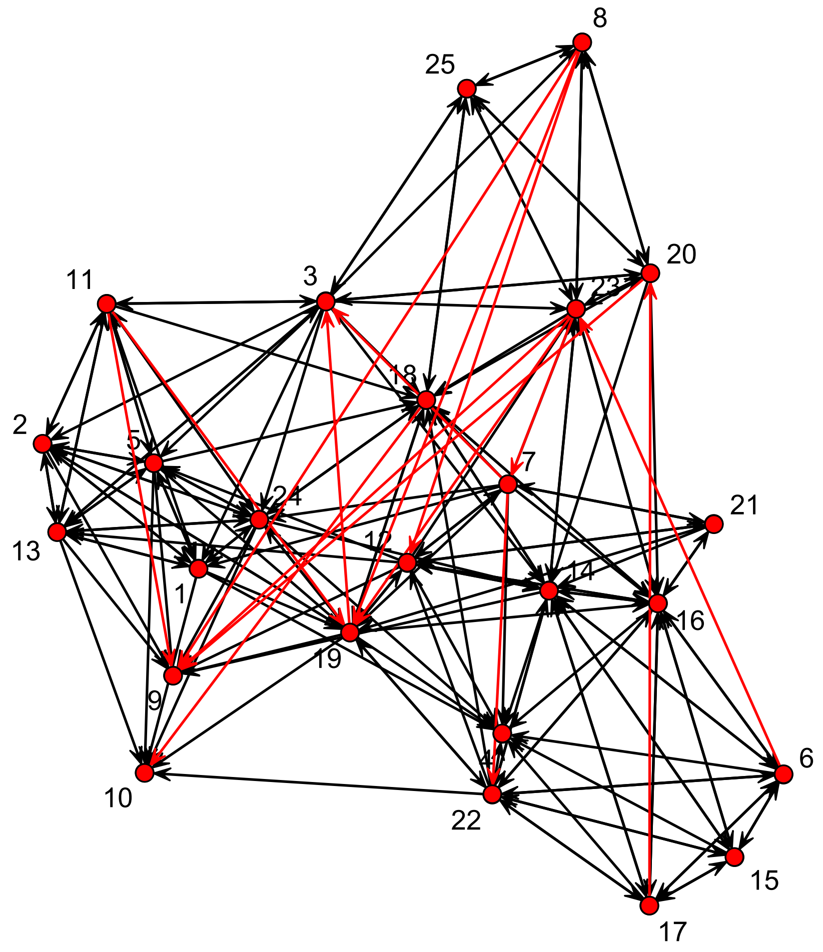


Figure 19: Two mode network; the black arrows belong to the friendship network and the red arrows belong to the "dislike" relation network

In Figure 19, the two overlapped networks can be observed; the black arrows are the ones also present in Figure 14, while the red arrows are the dislike relations expressed by the pupils. Again, all the relations going outside of the class were ignored by the study.

Since the aim of this section is to simulate the evolution of the network, some elements must be defined:

- Friendship network rate parameter
- Dislike network rate parameter
- Mass probability generating functions

The third element, the mass probability generating functions, will be distinguished between the two levels, with some terms belonging to both levels and others concerning only one. As in the previous example, the parameters were estimated using RSiena, used in order to have a reliable simulation of the network evolution [10]. The two rate parameters estimated are: $\rho_{friend} = 4.15$ and $\rho_{enemy} = 1.73$. The objective functions for both levels will be:

$$\begin{aligned} &P_{ij} = exp\{\beta_1 s_{i1}(\mathbf{x}_f) + \beta_2 s_{i2}(\mathbf{x}_f) + \beta_3 s_{i3}(\mathbf{x}_f) + \beta_4 s_{i4}(\mathbf{x}_f) + \beta_5 s_{i5}(\mathbf{x}_f, \mathbf{x}_e)\} \\ &\text{for the } \textit{friendship network}\text{, and} \\ &P_{ij} = exp\{\beta_1 s_{i1}(\mathbf{x}_e) + \beta_2 s_{i2}(\mathbf{x}_e) + \beta_3 s_{i3}(\mathbf{x}_e) + \beta_4 s_{i4}(\mathbf{x}_e) + \beta_5 s_{i5}(\mathbf{x}_e, \mathbf{x}_f)\} \\ &\text{for the } \textit{dislike network.} \end{aligned} \tag{13}$$

In Formula 13 $\mathbf{x}_f$ is the *friendship network* and $\mathbf{x}_e$ is the *dislike network*. Consequently the mass probability generating functions is:

$$P_{ij} = \frac{exp\{\beta_1 s_{i1}(\mathbf{x}) + \beta_2 s_{i2}(\mathbf{x}) + \beta_3 s_{i3}(\mathbf{x}) + \beta_4 s_{i4}(\mathbf{x}) + \beta_5 s_{i5}(\mathbf{x}, \mathbf{y})\}}{\sum_{z=1, z\neq i}^{g} exp\{\beta_1 s_{i1}(\mathbf{x}) + \beta_2 s_{i2}(\mathbf{x}) + \beta_3 s_{i3}(\mathbf{x}) + \beta_4 s_{i4}(\mathbf{x}) + \beta_5 s_{i5}(\mathbf{x}, \mathbf{y})\}}$$

where $\mathbf{x}$ and $\mathbf{y}$ must be adjusted in order to appropriately consider the *friendship* and the *dislike network*.

The $s_{i1}, \ldots, s_{i4}$ are the same as in section 3.1.3, which means out-degree, reciprocity, in-degree popularity and out-degree popularity; s_5 here is defined

$\beta\backslash$ Net	Friend	Enemy
β_1	-1.49	-3.61
β_2	1.63	-7.27
β_3	0.10	0.50
β_4	-0.02	-0.23
β_5	-2.17	-0.56
ρ	-4.15	1.73

Table 4: Parameters estimated for the two networks, on the column "Friend" all the parameter for the *friendship network*, on the column "Enemy" the one regarding the *dislike network*. They are all different because the same effects can affect in various manners the different levels.

as reciprocity with the other level, therefore this effect is included only if an individual dislikes another one which instead likes him/her; this effect represents one way in which one level can affect the other. In order to express this effect in mathematical notation, consider X_f as the *friendship network* and X_e the *dislike network*, then $s_{i5} = \sum_{j=1}^{g} x_{f,ij} x_{e,ji}$. Table 4 summarises all the estimated parameters which will be used in the network simulation.

The steps that will be followed in order to simulate the network are:

1. Sampling which level will make the next move using the probability given from the two rate parameters

2. Sampling the focal actor from a distribution with discrete uniform probability

3. Sampling the tie to change using the function linked to the level selected

4. Change of the tie and restart from step 1

As beforehand mentioned, the process for the simulation is similar to the single-level one, with the difference that first the level on which the change will happen must be chosen. Given the rate parameters value discussed in the previous chapter, the discrete probability distribution for the choice of the level can be observed in Figure 20. These two rates indicate how many changes are allowed for each actor in the observed period on average; they are

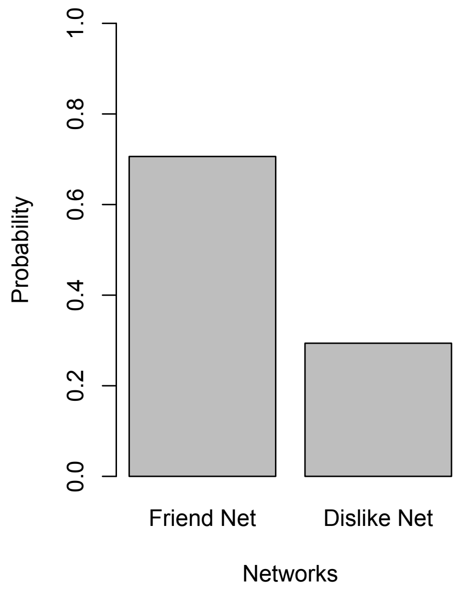


Figure 20: Discrete probability distribution for the sampling of the level

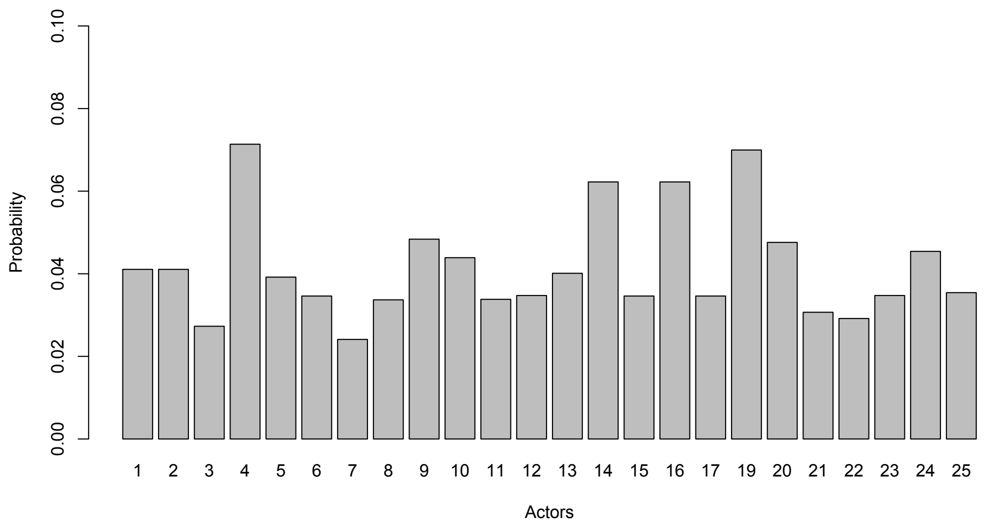


Figure 21: Discrete probability distribution for the second individual choice in the *friendship network* given the focal actor 18

constant during the simulation, as well as the discrete probability distribution for the levels.

Since these two parameters are considered to be constant also through all the individuals, the sample of the focal actor is independent from the level choice and it is uniformly distributed along all the individuals.

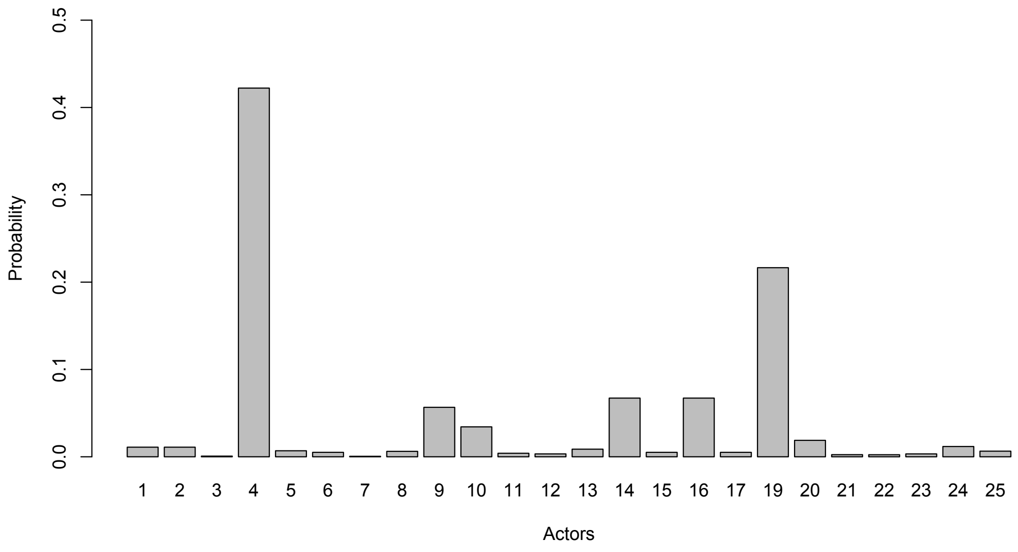


Figure 22: Discrete probability distribution for the second individual's choice in the *dislike network* given the focal actor 18

The sampling of the second individual towards which the tie is going to change depends on the level chosen and on the focal actor; given the first, the second actor always has a different mass probability function. In Figure 21, the discrete probability distribution on the *friendship network* is shown, given that the focal actor is individual 18; in Figure 22, the discrete probability distribution on the *dislike network* is shown, given the same focal actor. Two simulations of the network were implemented, following the proposed procedure.

The results are represented in a tabular form which summarises all the ministeps as well as which change has been done to the network and the representation of the network evolution. The data collected during the simulation is the time in which the change happened (starting from zero), on which level it is happening, the two actors involved and the kind of change (creation or withdrawal of a tie).

In Table 5 it is possible to observe that seven out of ten changes in the whole network were on the *friendship network* and only three were on the *dislike* one. This result was expected, given the mass probability distribution

Step	Time	Network	*ego*	*alter*	Prev. Value	Succ. Value
1	0.04	1	20	16	1	0
2	0.18	1	23	11	0	1
3	0.35	1	6	8	0	1
4	0.47	2	25	14	0	1
5	0.49	1	24	6	0	1
6	0.51	1	3	14	1	0
7	0.62	1	18	4	0	1
8	0.64	1	19	14	0	1
9	0.94	2	21	18	0	1
10	1.26	2	16	4	0	1

Table 5: Summary of the network simulation after ten ministeps. Network "1" is the *friendship network* and Network "2" is the *dislike network*; the last two columns show if the tie is created or withdrew

of the two networks. The difference in mass probability distribution depends on the number of changes made in the real network in the observed elapsed time, and in the *friendship network* there are more changes.

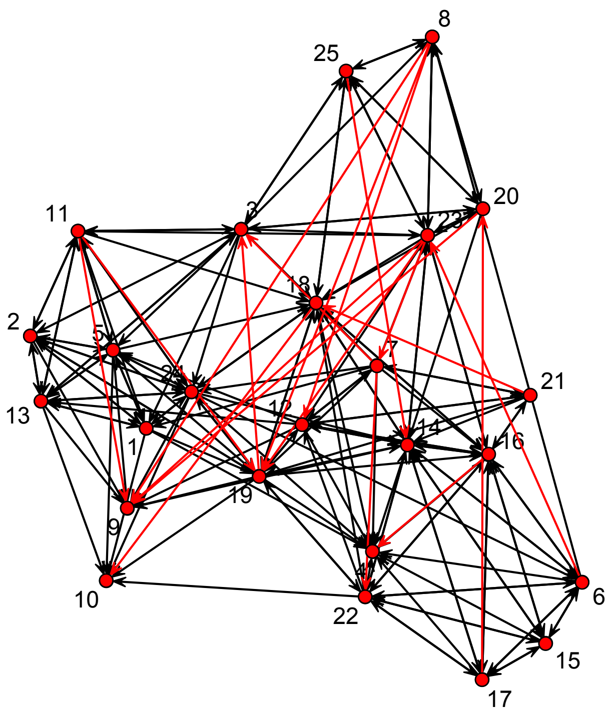


Figure 23: Representation of the network simulation after ten ministeps

Additionally, it is worth to notice that this simulation is mainly "constructive", i.e. the ties created are more than the ones withdrew, and this is due to the model chosen; the model is too simple for reliable future steps to be built on it, but its aim is to explain in a facilitated way the evolution of the network, not to allow any inference on it.

In Figure 23 and in Figure 24 it is possible to observe the two simulated networks. After ten ministeps, and more consistently after fifty ministeps, the network is more dense than the one in Figure 19, where the whole simulation started.

At this point, the linkage between the network and the stochastic process should be clear, as well as how can the evolution of a network be simulated. In the next chapter a more complex and reliable model to estimate the effects of the network will be explained.

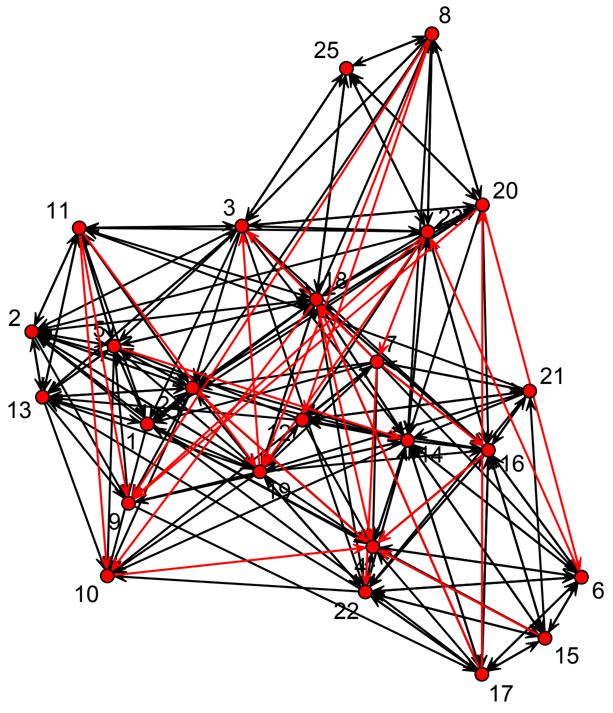


Figure 24: Representation of the network simulation after fifty ministeps

4 Actor-Based Stochastic Models

The previous sections serve as an introduction to a more complicated model that will now be explained. This model was first proposed by Tom Snijder in 1996, and it is in continuous evolution. Accordingly with all the previous sections, the nodes of the network are the actors and the ties represents the social relation between them. The network evolution is, as already stated, a dynamic process and it can be assumed that it evolves like a stochastic process driven by the actors; in fact, the actors will decide whether to construct a tie or withdraw it, consequently modifying the network structure. This assumption is one of the core assumptions of the Actor-Based Stochastic Model introduced by Snijders; «Distinguishing characteristics of the stochastic actor-based models are flexibility, allowing to incorporate a wide variety of actor-driven micro mechanism influencing tie formation; and the availability of procedure for estimating and testing parameters that also allow to assess the effect of a given mechanism while controlling for the possible simultaneous operation of other mechanism or tendencies. »(Introduction to Stochastic Actor-Based Models for Network Dynamics, Snjiders, 2009 [12])

4.1 Evolution and coevolution

The decisions made by the actors leading the evolution of the network will be driven by multiple endogenous or exogenous factors. The endogenous effects refer to the current status of the network itself, while the exogenous factors that can affect the evolution are, e.g. the actor covariates (characteristics of the actor itself) and dyadic covariates (interaction of the characteristics of the nodes connected by a tie, as mutual presence or absence).

On one hand, the network evolution is led by the network itself and from the characteristics of the actors but, on the other hand, also the characteristic of the actors can be modified by the network structure; this is what is called *coevolution*. Coevolution is driven by the *influence process*. An example of *influence process* is smoking in adolescence: it could be that smokers are more friendly to other smokers (e.g. they can meet the each other in the smoking area) or it could be that being in a group of friends in which almost everyone smokes can lead an individual to smoke as well. In this example, the first case is the influence of the characteristic on the network structure (*Homophily*) and the latter is the influence of the network structure on the actor's characteristics (*Assimilation*).

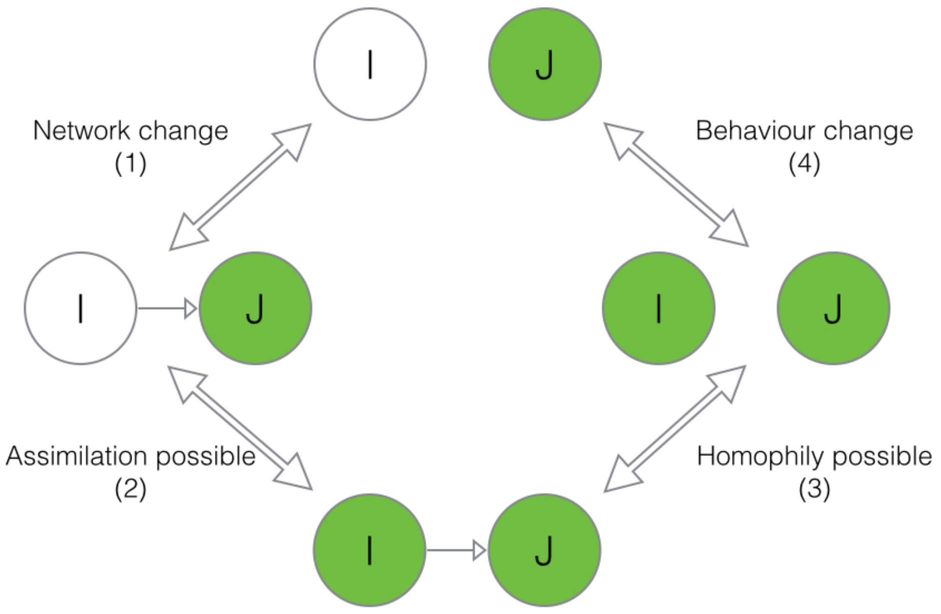


Figure 25: Possible effects of homophily and assimilation between two individuals in a network; the behaviour is represented by the color and the relation by the arrows

The diagram in Figure 25 represents the possible effects of homophily and assimilation on a dyadic relation. At the top of the diagram there are two unrelated individuals with different behaviours (individual I is white and individual J is green). If (1) happened, individual I would link to individual J; referring the smoking example, a smoker and a non-smoker would become friends. In (2) the assimilation affects the individuals and the non-smoker influenced by his/her friend would start smoking. The right side shows that if (4) happened, for external causes the non-smoker individual would start smoking. Now the homophily effect can influence the friendship relation between the two individuals (3). The behavioural component, smoking in the example, can assume other meanings as drinking attitude, stress, ect..

The only aspect required to this kind of variable is to be discrete with a small set of possible finite values (practically maximum 7); this is due to the fact that also this variable can be intended as a stochastic process evolving together with the network and both can influence each other. Thanks to the fact that panel data is used in this model, it will be also possible to understand a sort of causal relation between the network evolution and the change in the individuals' behaviour [13].

4.2 Notation and data structure

The set $\Re$ contains all the possible relations between the elements in $\mathcal{X}$, the set of social actors, hence it is a subset of the Cartesian product $\mathcal{X} \times \mathcal{X}$; if $(i, j) \in \Re$ there is a tie between the two actors. The relations are assumed to be non-reflexive: $(i, i) \notin \Re \; \forall \; i$ and can happen that $(i, j) \in \Re$ but $(j, i) \notin \Re$. The relations are summarised in the adjacency matrix $X = (X_{ij})$ which will be a $g \times g$ matrix, where g is the number of actors in the network. $X_{ij} = 1$ if there is a tie between i and j and $X_{ij} = 0$ if there is not. The actor attributes are assumed to be ordered, discrete and having a finite set of possible values; Z_{li} denotes the value for actor i of the lth attribute. The time dependence can be represented as: $X = X(t)$ and $Z_h = Z_h(t)$, where t denotes the instant in which the network and the attributes are observed. It is supposed then that the panel network observations available will be $(X(t), Z_1(t), \ldots, Z_L(t))$ with t discrete time points: $t_1 < t_2 < \cdots < t_M$ and $M \geq 2$. The individual covariates will be named as $v_l = (v_{l1}, \ldots, v_{lg})$ and the covariates depending on both actors involved will be $w_h = (w_{lij})_{1 \leq i,j \leq g}$ [8].

4.3 Model assumptions

The model deals with directed relations, therefore a sender, called *ego*, and a receiver, called *alter*, can be identified. The relation between ego and alter can be represented as: $i \to j$. The following assumptions are made:

1. The underlying process is in continuous time, with parameter t, and the time steps can vary in length. However, the observations of the

process are in two or more discrete time points; it is therefore possible to refer to the observations as "network panel waves". This assumption allows to consider the process as the creation of a tie depending on the whole network structure and does not allow multiple changes in the same time instant. Since the change will happen in continuous time and the observations are made in discrete time points, $t_1 < t_2 < \cdots < t_M$, the result is that what is shown are the differences between $(X(t_l), Z_1(t_l), \ldots, Z_L(t_l))$ and $(X(t_{l+1}), Z_1(t_{l+1}), \ldots, Z_L(t_{l+1}))$, which are the results of many unobserved sequential changes.

2. The changes of the network are the outcome of a Markov process. Thus, the total network structure is the social context that influences the probabilities of its own changes, which in a mathematical formulation is: $(X(t'), Z_1(t'), \ldots, Z_L(t'))$ with $t' > t$, and t' is independent from all the events happened before time t. This assumption will not be realistic most of the times, but it is the less constraining one in respect to all the other possibilities [7]. Moreover, the assumption will compel the model to refer to lasting linkages, as friendship, more than to ephemeral ones, as the exchange of mails.

3. The actors control their outgoing ties. This implies that the changes are made by the ego actor based on his characteristics, on the alter's characteristics and on their position into the network. This assumption also implies that the focal actor has complete information on the network status and on the alter's characteristics.

4. At a given moment, a probabilistically selected actor - ego - may get the opportunity to change one outgoing tie or the value of one of his behavioural attitudes. No more than one tie or one behavioural attitude can change at any moment. This assumption decomposes the network evolution into the smallest steps possible. This, together with assumption 3., means that the changes happen with the complete knowledge of the network and sequentially, so that the first change can imply the next: «Thus the co-evolution process is separated into a network change

process (social selection) and a behaviour change process (social influence), mutual linked because the transition distribution of each process is determined not only by its own current state but also by the current state of the other process; they are not linked by a joint process where an actor determines simultaneously a change in a network tie and a change in behaviour». (Modelling the co-evolution of networks behaviour, Snijders, Steglich, Schweinberger, 2007 [14]).

The actor-based network change process can be decomposed into stochastic sub-phases, opportunity and determination process:

5. The change opportunity process, modelling the frequency of ties or behavioural changes by actors. The change rate may be constant through all the actors, depend on their position in the network or also depend on their characteristics (the covariates).
6. The change determination process, modelling the exact tie which will be changed when the ego actor is chosen. This process depends on all the possibilities mentioned for the change process, adding the fact that also the alter's characteristics can affect the choice.

Assumption 2., the network seen as a Markov process, implies that the first time point of the panel, including the network status and the initial behavioural values, will be taken as granted and will not be analysed; hence, no assumptions of a dynamic equilibrium are required in this kind of models.

4.4 The Change Opportunity process

The change opportunity process is mainly driven by the rate function, which indicates how frequently the actors make ministeps. In order to have a shorthand notation $(X(t_l), Z_1(t_l), \ldots, Z_L(t_l)) = Y(t)$. The rate function can be formally defined as:

$$\lambda_i(x) = \lim_{dt \to 0} \frac{1}{dt} P\left\{Y_{it}(t + dt) \neq Y_{it}(t) \text{ f.s. j} \in \{1, \ldots, g\} | Y(t) = y\right\} \tag{14}$$

The moment for a change, whichever the component changing is, is randomly determined and follows a Poisson process. The waiting times are modelled by an exponential distribution with rate parameter λ. For each individual there is one rate function for the network, $\lambda_i^{[X]}$, and one for the behavioural changes, $\lambda_i^{[Z]}$; as mentioned in section 3.2, the network can be on more than one level, and the rate functions defined accordingly. In this case, the rate functions for the network will be more than one: $\lambda_i^{[X_1]}, \ldots, \lambda_i^{[X_n]}$, one for each level.

The rate functions will depend on the time period m, on the characteristics of the actors v_{ij}, and on network characteristics. The two rate functions during the time period $t_m < t < t_{m+1}$ are given by:

$$\begin{aligned} \lambda_i^{[X]}(Y, m) &= \rho_m^X exp\left(\sum_k \alpha_k^{[X]} a\left[X\right]_{ki}\left(Y(t)\right)\right) \\ \lambda_i^{[Z_h]}(Y, m) &= \rho_m^{Z_h} exp\left(\sum_k \alpha_k^{[Z_h]} a\left[Z_h\right]_{ki}\left(Y(t)\right)\right) \end{aligned} \tag{15}$$

Where the components in Equation (15) are:

- ρ, which indicates the period-dependence.
- $\alpha_k^{[Z_h]}$, which indicates the dependence on the statistics computed on $a\left[X\right]_{ki}$.

Using the "lack of memory" property of the exponential distribution, the ρ parameter will absorb all the time effect, hence the rate will not be affected from differences in the duration $\Delta t = (t_{m+1} - t_m)$. When the rate parameters are different for every actor, the mass probability distribution for the sampling will be:

$$\frac{\lambda_i^{[X]}}{\lambda_+^{[X]}} \quad \text{where} \quad \lambda_+^{[X]} = \sum_i \lambda_i^{[X]}$$

for the network evolution, and

$$\frac{\lambda_i^{[Z_h]}}{\lambda_+^{[Z_h]}} \quad \text{where} \quad \lambda_+^{[Z_h]} = \sum_i \lambda_i^{[Z_h]}$$

for the behavioural component evolution.

This definition comes from the exponential distribution property explained in section 3.2.1. Using the same property, a unique process allowing to sample contemporary the kind of change, the level and the ego, could be defined. The distribution in a model with a two-levels network and a behavioural variable would be:

$$EXP(\lambda_+) \quad \text{where} \quad \lambda_+ = \sum_i \lambda_i^{[X_1]} + \sum_i \lambda_i^{[X_2]} + \sum_i \lambda_i^{[Z]} \tag{16}$$

where $\lambda_i^{[X_1]}$ is the first level, $\lambda_i^{[X_2]}$ is the second level and $\lambda_i^{[Z]}$ is the behavioural variable. Using the same property, an actor i is the focal actor with probability:

$$\frac{\lambda_i^{[X_1]} + \lambda_i^{[X_2]} + \lambda_i^{[Z]}}{\lambda_+}$$

where λ_+ is defined in equation (16). Given this actor i the discrete probability distribution for level "1", level "2" and behaviour is:

$$\frac{\lambda_i^{[X_1]}}{\lambda_i^{[X_1]} + \lambda_i^{[X_2]} + \lambda_i^{[Z]}}, \quad \frac{\lambda_i^{[X_2]}}{\lambda_i^{[X_1]} + \lambda_i^{[X_2]} + \lambda_i^{[Z]}} \quad \text{and} \quad \frac{\lambda_i^{[Z]}}{\lambda_i^{[X_1]} + \lambda_i^{[X_2]} + \lambda_i^{[Z]}}.$$

4.5 The Change Determination process

Once the focal actor is determined, the alter towards whom the tie will change or the behavioural characteristic to change must be selected; the functions that allow to reach this result are objective functions. An objective function is assumed to be decomposable into the following parts: the *evaluation function f,* the *endowment function g,* and random error term ε. The functions to be maximised are:

$$\begin{gathered} f_i^{[X]}\left(\beta^{[X]}, y\right) + g_i^{[X]}\left(\gamma^{[X]}, y|Y(t)\right)\right) + \varepsilon_i^{[X]}(y) \\ f_i^{[Z_h]}\left(\beta^{[Z_h]}, y\right) + g_i^{[Z_h]}\left(\gamma^{[X]}, y|Y(t)\right)\right) + \varepsilon_i^{[Z_h]}(y) \end{gathered} \tag{17}$$

A convenient choice for the random influence given from $\varepsilon_i^{[Y]}(y)$ is the type 1 extreme value distribution (Gumbel distribution), with mean 0 and scale parameter 1 [15]. These functions are called objective functions because the aim is to maximise them since they are considered as a proxy for the satisfaction that an actor gains when he/she changes the status in the network. This is an assumption based on the fact that when someone makes a change in his/her friendship network or behaviour this is aimed to improve his/her actual condition.

The Actor-based Stochastic Model also assumes that the change will be made in order to myopically optimise the actor's condition. This means that the ego will choose in order to obtain higher values of his/her objective function, subject to the constraints of the actual network status and the chance to make only one ministep per time. The optimisation is myopic because the ego will consider only the changes in his status (values of the objective function) in one ministep, not considering a procedure that could lead him/her to gain higher global values. The reason is that it is not worth to take it into consideration since, before the same focal actor will have again the chance to change a tie, the network could have been totally changed and the same is valid for the behavioural variable. The evaluation function depends only on the new state y while, on the other hand, the endowment function depends on both the current and the new state.

4.5.1 The evaluation functions

«The evaluation function f_i measures the satisfaction of actor i with a given network-behavioural configuration, independently of how this configuration is arrived at.»(Modelling the co-evolution of networks behaviour, Snijders, Steglich, Schweinberger, 2007 [14]). The evaluation functions can be written as follows:

$$\begin{aligned} f_i^{[X]}\left(\beta^{[X]}, y\right) &= \sum_k \beta_k^{[X]} s_{ik}^{[X]}(y) \\ f_i^{[Z_h]}\left(\beta^{[Z_h]}, y\right) &= \sum_k \beta_k^{[Z_h]} s_{ik}^{[Z_h]}(y) \end{aligned} \tag{18}$$

where, as usual, the first is referring to the network and the second to the behaviour.

4.5.2 The endowment functions

«The endowment function g_i, on the other hand, measures a component of satisfaction with a given network-behavioural configuration that will ne lost when the value of a variable X_{ij} or Z_{hi} is changed, but which was obtained without "cost" when it was obtained.»(Modelling the co-evolution of networks behaviour, Snijders, Steglich, Schweinberger, 2007 [14]). Therefore, the aim of the endowment function is to include in the model the differences between the creation and the withdrawal of a tie that the evaluation function is not able to collect. Theories by Van de Bunt (1999) state that the cost of losing a reciprocated friendship is greater than the cost involving the creation of a new one [13]. On the behavioural point of view, this function will evaluate the loss in satisfaction brought by a decrease in a behavioural attitude. The endowment function for the change from y^0 to y is:

$$\begin{aligned} g_i^{[X]}\left(\gamma^{[X]}, y|y^{(0)}\right) &= \sum_k \sum_{i \neq j} \gamma_k^{[X]} I\left\{x_{ij} < x_{ij}^{(0)}\right\} s_{ijk}^{[X]}\left(y^{(0)}\right) \\ g_i^{[Z_h]}\left(\gamma^{[Z_h]}, y|y^{(0)}\right) &= \sum_k \gamma_k^{[Z_h]} I\left\{z_{hi} < z_{hi}^{(0)}\right\} \left(s_{ik}^{[Z_h]}\left(y^{(0)}\right) - s_{ik}^{[Z_h]}(y)\right) \end{aligned} \tag{19}$$

The function $I\{\}$ is the identifier function, which has value 1 when there is a decrease in value, i.e. the tie from 1 becomes 0 or the behavioural variable decreases, and 0 otherwise. $x_{ij}^{(0)}$ is the endowment value, i.e. when in the previous network the value of $x_{ij} = 1$ in $x_{ij}^{(0)} = 0$; and the same holds for the behavioural variable.

4.5.3 The resulting probabilities

The resulting mass probability function can be computed as follows:

$$Pr\left(x(i \to j), x(t), z(t)\right) = \frac{exp\left([f+g]_i^{[X]}\left(\beta^{[X]}, \gamma^{[X]}, x(i \to j)(t)\right)\right)}{\sum_k exp\left([f+g]_i^{[X]}\left(\beta^{[X]}, \gamma^{[X]}, x(i \to k)(t)\right)\right)}$$

$$Pr\left(z(i \updownarrow_h \delta)|x(t), z(t)\right) = \frac{exp\left([f+g]_i^{[Z_h]}\left(\beta^{[Z_h]}, \gamma^{[Z_h]}, z(i \updownarrow_h \delta)(t)\right)\right)}{\sum_{\tau \in \{-1,0,1\}} exp\left([f+g]_i^{[Z_h]}\left(\beta^{[Z_h]}, \gamma^{[Z_h]}, z(i \updownarrow_h \tau)(t)\right)\right)}$$

Where $x(i \to j)$ represents the actor i changing his/her tie towards actor j, and $z(i \updownarrow_h \tau)$ stands for a change in behavioural attitude. The outcome of a change on the adjacency matrix is: $X_{ij}^{(0)} = 1 - X_{ij}$; this will be valid also when considering multi-level networks, since they consist in modelling two one-mode networks and not a two-mode network. One network will be able to affect the other but the definition of adjacency matrix stands, so that it will include only two values: $\{0, 1\}$. On the other hand, the behavioural variable will change with an increase or decrease in its value of only one unit: $Z_{ih}^{(0)} = Z_{ih} + \delta$.

4.6 Basic effects

Once that the objective functions are defined, the s_{ik} effects need a deeper attention. These effects can come from the network or the behavioural variables, and reflect the actor's characteristics. Hereby, a short summary of what can be included in the model is presented; if non-discussed effects are introduced, they will be explained contextually. For a deeper introduction to the effects see Snijders (2001, 2005 [16], [8]).

1. *Outdegree effect* is the number of outgoing ties:
   $s_{i1}(x) = x_{i+} = \sum_j x_{ij}$

2. *Reciprocity effect* is the number of reciprocated ties:
   $s_{i2}(x) = x_{i(r)} = \sum_j x_{ij} x_{ji}$

3. *Tranisitivity effect* is the number of transitive patterns in the focal actor's ties. This includes a third actor h and the relation will include $i \to j \to h$ and at the same time $i \to h$:
   $s_{i3}(x) = \sum_{j,h} x_{ij} x_{ih} x_{jh}$
   (see example in Figure 26.)

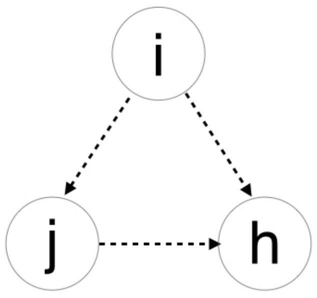


Figure 26: Transitive triad with $i \to j \to h$ and $i \to h$

4. *Same behaviour*, is the sharing of the same behavioural attitude between ego and alter:
   $s_{i4}(x, z) = \sum_j x_{ij} I(z_i = z_j)$
   mainly used for dichotomous behavioural variables, e.g. smoking.

5. *Attribute-related similarity* is the sum of similarities with respect to a behavioural variable between ego and alter:
   $s_{i5}(x, z) = \sum_j x_{ij} \left(1 - |z_{ih} - z_{jh}|/R_h\right)$ with R_h range of Z_h
   this is similar to the effect at point (4) with the difference that it can be used with discrete behavioural variables.

6. *Main effect of a dyadic covariate w* is the sum of the values of w_{ij} for all alters to whom the ego is connected:
   $s_{i6}(x) = \sum_j x_{ij} w_{ij}$

The main effects used for the behavioural objective function are:

1. *Tendency*, indicating the preference for high values:
   $s_{i1}(x, z) = z_{ih}$

2. *Attribute-related similarity*, the sum of similarities with respect to the behavioural function and the alter to whom the ego is tied:
   $s_{i2}(x,z) = \sum_j x_{ij}\,(1 - |z_{ih} - z_{jh}|/R_h)$ with R_h range of Z_h

3. *Dependence on other behaviours* h' $(h \neq h')$:
   $s_{i3}(x,z) = z_{ih} z_{ih'}$

For a deeper introduction to more possible behavioural effects see Steglich et al. (2010, [17]).
It is possible to use the same s_{in} effects in both the network objective function and in the behavioural objective function, and this can lead to unveil correlation effects between the two.

4.7 Intensity matrix

Since the whole process is required to be a continuous time Markov process, it is worth to show how the transition matrix can be written. The process is, indeed, fully described by its initial state and the matrix of transition intensities.

$$\begin{aligned} q_{ij}(y,\hat{y}) &= \lim_{dt\downarrow 0} \frac{1}{dt} P\{Y(t+dt) = y(i\to j) | Y(y) = y\} \\ &= \lambda_i(y) p_{ij}(y,\hat{y}) \end{aligned} \tag{20}$$

where $y = (x,z)$ and $\hat{y}$ is the next outcome. The explicit expression of q_{ij} is:

$$q_{ij}(y,\hat{y}) = \begin{cases} \lambda_i^{[X]}(y) Pr(x(i\to j)|x,z) & \text{if } \hat{y} = (x(i\to j))\,, \\ \lambda_i^{[Z_h]}(y) Pr(z(i \updownarrow_h \delta)|x,z) & \text{if } \hat{y} = (x, z(i \updownarrow \delta))\,, \\ -\sum_i \sum_{j\neq i} q\left(y; \left(x\left(i\to j\right), z\right)\right) + & \\ -\sum_i \sum_{\delta\in\{-1,1\}} q\left(y; \left(x, z\left(i \updownarrow_h \delta\right)\right)\right) & \text{if } \hat{y} = y \\ 0 & \text{Otherwise.} \end{cases}$$

4.8 Estimation

The complexity of the model leads an hypothetical likelihood function not to have an explicit solution in the general case, hence the frequentist and bayesian approach to the estimation of the model parameters is not suggested. On the other hand, the Method of Moments (*MoM*), i.e. the match between the empirical moment (observed on the sample) and the theoretical one, is able to reach a close solution. The *moment equation* is defined as follows:

$$E_{\hat{\theta}}\left(u\left(Y\right)\right) = u(y) \tag{21}$$

where $u(y)$ is a statistic computed on the observations and θ is the parameter. Moreover, through the delta method and the implicit function theorem, under regularity condition, the asymptotic covariance matrix can be computed as follows:

$$\begin{aligned} cov_\theta\left(\hat{\theta}\right) \approx D_\theta^{-1}\Sigma\theta D_\theta^{\prime -1} \quad \text{where} \quad D_\theta &= \left(\frac{\partial E_\theta\left(u(Y)\right)}{\partial\theta}\right) \\ \text{and} \quad \Sigma_\theta &= cov_\theta\left(u(Y)\right) \end{aligned} \tag{22}$$

Therefore, the efficiency of the MoM estimators depends from $u(Y)$.

4.9 Statistics for moment estimation

The empirical statistics on $u(y)$ are easy to compute but, in order to estimate the model with MoM, also the theoretical moment $u(Y)$ is needed. The parameters in θ are the above introduced parameters $\rho, \alpha, \beta, \gamma$; some of them will be constant through all the panel waves and some will vary, e.g. all the waves can have different rate parameters. This difference between the parameters will lead to estimate them differently. The parameters that will be kept constant through the time will have the following moment equation:

$$\sum_{m=2}^{M}\left\{u_m\left(Y(t_{m-1}), Y(t_m)\right)\left|Y(t_{m-1}) = y(t_{m-1})\right.\right\} = u_m\left(y(t_{m-1}), y(t_m)\right) \tag{23}$$

hence all the waves will be considered. On the contrary, the parameters that change through time are affected only by the events happened between t_{m-1} and t and nothing else. The moment equation will be the following:

$$E_\theta \left\{ u_m \left(Y(t_{m-1}), Y(t_m) \right) | Y(t_{m-1}) = y(t_{m-1}) \right\} = u_m \left(y(t_{m-1}), y(t_m) \right) \quad (24)$$

Rate function parameters

The constant rate parameter for the network change, $\lambda_m^{[X]}$, and for the behavioural variable, $\lambda_m^{[Z_h]}$, have natural statistics in, respectively:

$$\begin{aligned} &\sum_{i,j} |X_{ij}(t_m) - X_{ij}(t_{m-1})| \ \text{ for estimating } \lambda_m^{[X]} \\ &\text{and} \\ &\sum_{i} |Z_{hi}(t_m) - Z_{hi}(t_{m-1})| \ \text{ for estimating } \lambda_m^{[Z_h]}. \end{aligned} \quad (25)$$

The extension to the rate parameters depending on the actor's covariate is straightforward: Formula 25, where the parameters $a_k^{[X]}$ and $a_k^{[Z_h]}$ are introduced in order to estimate covariates' weights.

Objective functions

The aforementioned opportunity to distinguish the objective functions into evaluation and endowment functions stands also here. For the sake of simplicity, only the evaluation function will be described into detail. The difference between the two functions, i.e. the fact that evaluation functions only refer to the current time and the endowment functions also refers to the previous time point observation, stands here as well. In order to adapt the evaluation function structure to the endowment one, it will be enough to adapt Formula (19) to the shape of the evaluation function.

The statistics are:

$$\begin{aligned}
&u_m\left(Y(t_{m-1}), Y(t_m)\right) = \sum_i s_{ik}^{[X]}\left(X(t_m), Z(t_{m-1})\right)\\
&\text{for estimating the parameters } \beta_k^{[X]} and\\
&u_m\left(Y(t_{m-1}), Y(t_m)\right) = \sum_i s_{ik}^{[Z_h]}\left(X(t_{m-1}), Z(t_{m-1}), Z(t_m)\right)\\
&\text{for estimating the parameters } \beta_k^{[Z_h]}.
\end{aligned} \tag{26}$$

4.10 Stochastic approximation for parameter estimation

The theoretical moment in the moment equations (23) and (24) cannot be computed in non-trivial cases; however, the stochastic process can be simulated. In particular, the Robbins and Monro (1951) procedure can be used in order to solve the moment equations [8]. These methods are stochastic iterative algorithms, and the basic iteration is:

$$\hat{\theta}_{N+1} = \hat{\theta} - a_N D_0^{-1}\left(U_n - u(y)\right)$$

where U_n is generated according to the probability distribution given from $\hat{\theta}_N$, and a_N is a sequence slowly converging to zero. Hence, the procedure consists in simulating the network with the parameters computed in step N and afterwards updating them in order to obtain a network that is closer to the one observed; the parameters starting value is zero by default.

The above estimator is the *unconditional estimator*, while the *conditional MoM estimator* includes an additional feature. The latter simulates the network and the behavioural variable changes as well, but considers one as conditioning and the other as conditioned. The choice is made on the distance between the simulation and the real value observed; when either the network or the behavioural variable are reproduced in the simulation as it is in the observation, this is chosen as conditioning variable.

5 Implementation of the ABSM

The Actor-Based Stochastic Model (*ABSM*), introduced in section 4, can be useful to analyse the pupil's data introduced in section 2.1. The aim is to study the effects of the balanced and imbalanced triads on the happiness level, used as a proxy for the distress condition of the pupils. The model, as previously explained, using panel data is able to distinguish the effect of *influence*, the effects of the behaviour on the network, and *selection*, the effects of the network on the behaviour. The aim of this section is to estimate the best ABSM and to compare the results with the models estimated in section 2.2. The difference with those models is that, here, all the network aspects are treated in a more adequate way, and the effects, where possible, are treated together with the evolution of the network and the behaviour.

5.1 The data

5.1.1 The networks

Section 3 already showed the network representation; this representation, however, was limited to one class and all the ties outgoing from this limited set were ignored. Here, in order to estimate a complete model, the whole network will be considered. The network will include friendship and dislike relations but, due to the high number of ties to be represented, it is not worth to show the overlapped networks.

In Figure 27, it is possible to observe the network at its first observation time; the grey arrows represent a friendship relation and the orange arrows a dislike relation. The nodes are proportional to the out-degree of the actors, which is, for the *friendship network*, a proxy for the degree of friendliness of an individual, and for the *dislike network* a proxy for the opposite situation (i.e. it represents how many individuals an actor does not like). The higher is the out-degree, the bigger is the node. The node color depends on the happiness score of each individual, ascending from black to green. It is not easy to find a path in the graph which allows to make a statement on the relation between the color and the dimension of the nodes; this will be dealt

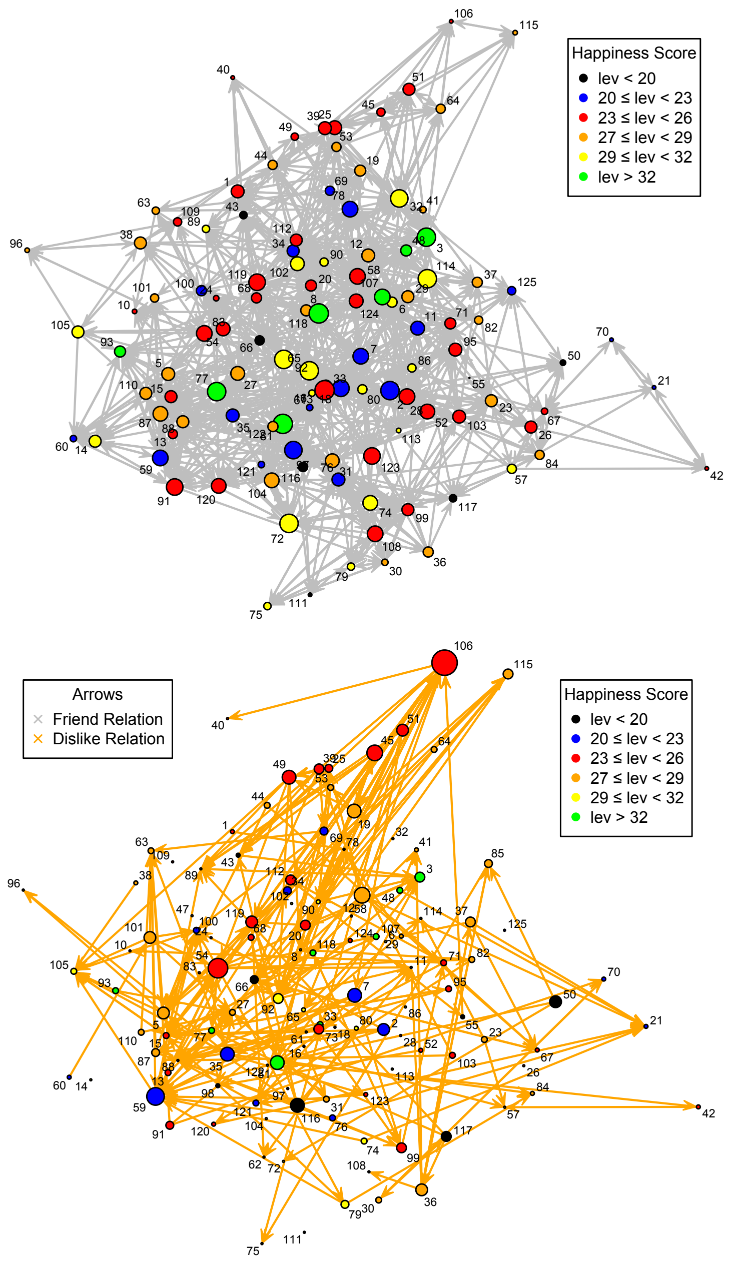


Figure 27: The first time point of the *friendship* and *dislike networks*; the nodes are proportional to the out-degree of the actors and the color is different according to the happiness score.

with later on in the model.

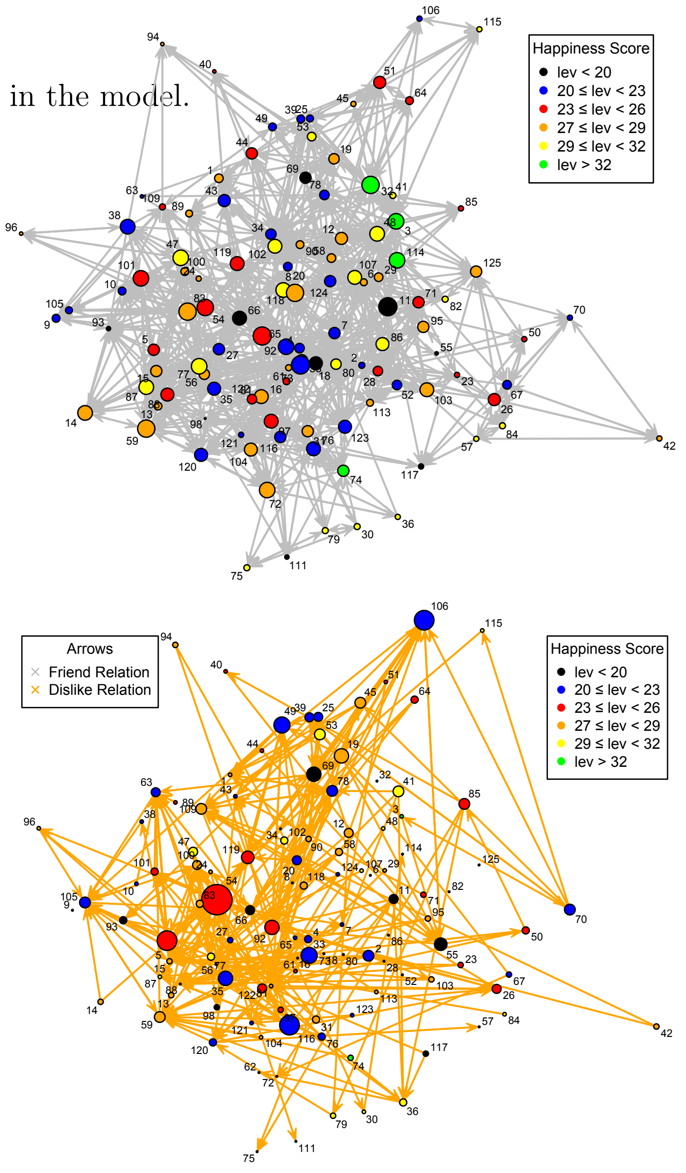
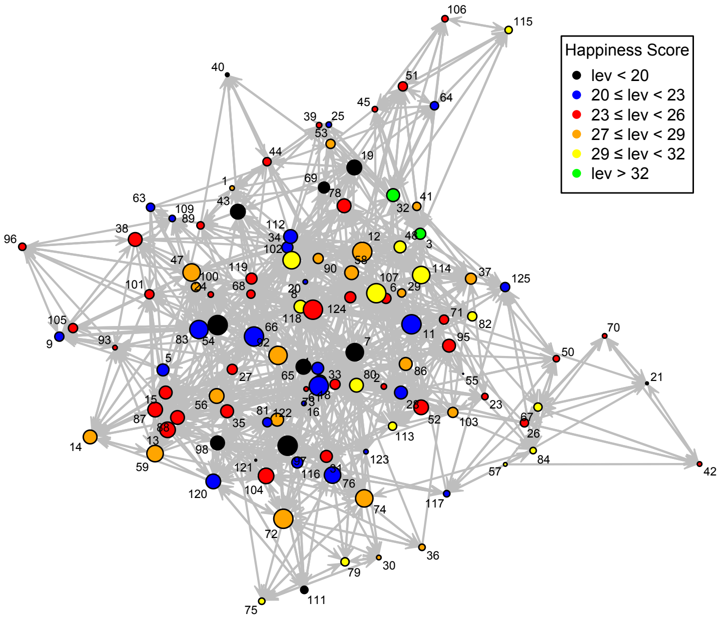


Figure 28: The second time point for *friendship* and *dislike networks*; the nodes are proportional to the out-degree of the actors and the color is different based on the happiness score.

Figure 28 and Figure 29 show, respectively, the second and third network observations, with the same characteristics of the previous figure. They show the network evolution and, if tracked, the difference in behaviour of each individual; however, once again, due to the high density, it is not possible to find any systematic evolution. It can be noticed that the happiness level decreased from the first time point to the last: the number of green actors



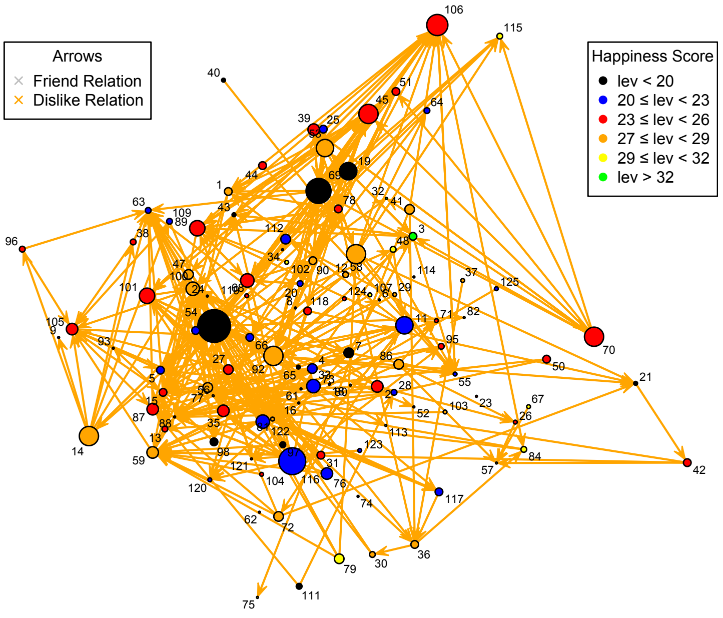


Figure 29: The third time point for *friendship* and *dislike networks*; the nodes are proportional to the out-degree of the actors and the color is different based on the happiness score.

has decreased. The position of each node was kept constant in order to allow insights of the network structure. The last but possibly meaningful representation of the network is the one with the nodes coloured depending on the pupils' class. In Figure 30, the *friendship network* is represented with the nodes dimension proportional to the individuals' out-degree and the color depending on the class. According to this graph, the classes seem to be compact: the individuals belonging to a class are more connected within the class than on the outside of it. Figure 31 shows the *dislike network* in the same manner of Figure 30. Here the arrows are more spread and it is not possible to state that the relations within the class are stronger then on the outside.

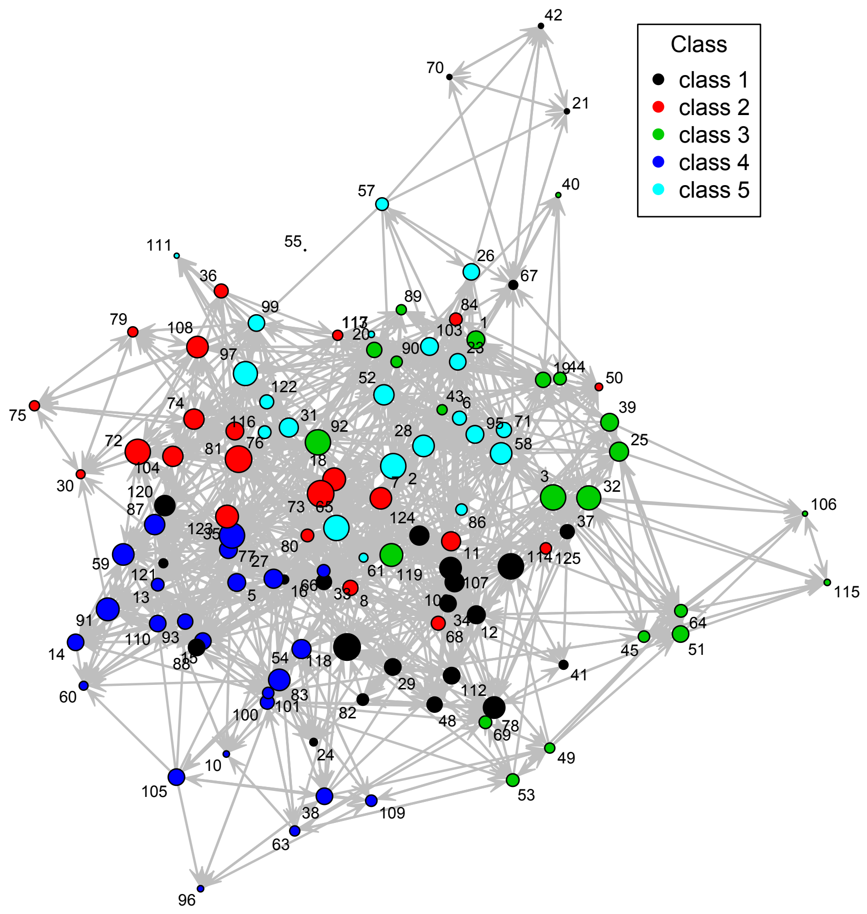


Figure 30: The *friendship network* with the dimension of the nodes proportional to the out-degree and the color depending on the pupils' class

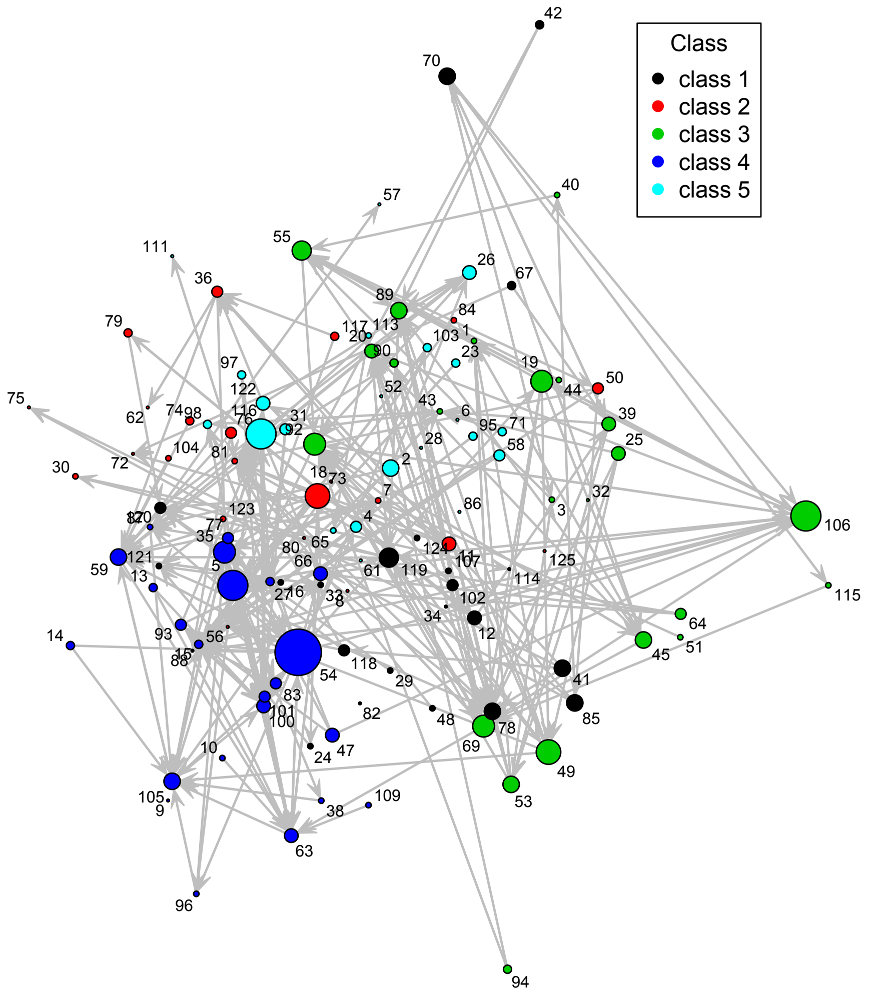


Figure 31: The *dislike network* with the dimension of the nodes proportional to the out-degree and the color depending on the pupils' class

In order to understand if the differences between two waves are enough to estimate the model, the Jaccard index can be useful. This index is a stability measure between successive networks, and can be defined as:

$$\frac{N_{11}}{N_{01} + N_{10} + N_{11}}$$

where N_{hk} is the number of tie variables with value h in one wave and value *k* in the next one [18]. A good value for the Jaccard index is greater than 0.3, although values greater than 0.2 can be acceptable anyway. The values for the *friendship network* are 0.557, for the difference between the first and the second wave, and 0.568, for the difference between the second and the third wave; both values are acceptable. For the *dislike network*, the first value is 0.249 and the second is 0.324; they are worse than the ones of the *friendship network* but still acceptable. The missing data proportion is under the recommended 20% bound.

5.1.2 The behavioural variable

The behavioural variable evolving together with the two-levels network introduced will be the distress proxy measured by the Happiness score. In section 2.1 it was introduced how this variable has been computed. This score

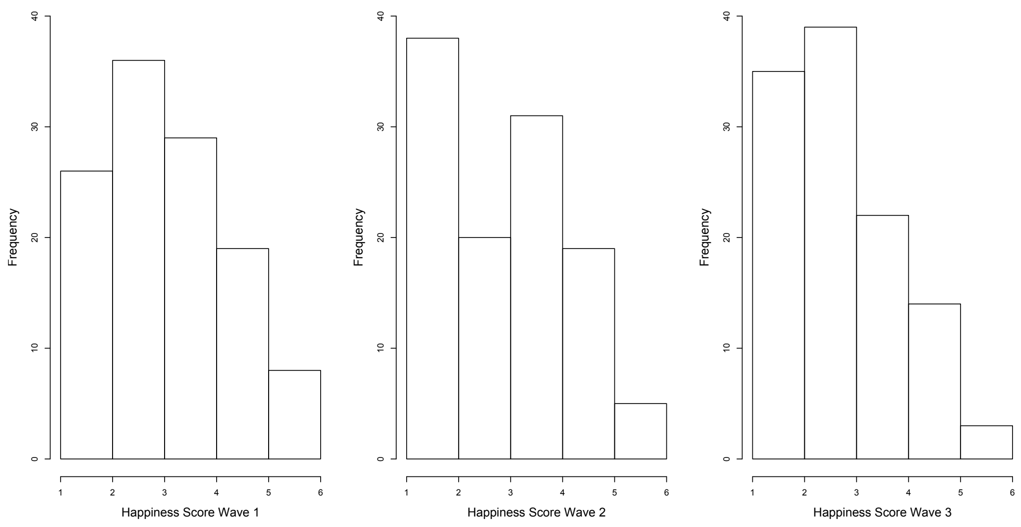


Figure 32: Histogram of the Happiness score variable divided into six classes

had to be rescaled for the purpose of this study; in fact, being a stochastic process evolving with the network, the number recommended for the set of possible outcomes is between 2 and 7. This is due to the fact that, if the possible steps were many, the stochastic process would continuously change, bringing excessive noise to the whole process which may therefore not converge in a feasible solution.

In Figure 32, the histogram of the Happiness score variable divided into six suitable classes is represented; this will allow the process to vary in a sustainable way. The frequency of the changes in the behavioural variable is summarised between the first and the second wave and between the second and the third in, Table 6 and Table 7, respectively.

w1 \ w2	*Score* 1	*Score* 2	*Score* 3	*Score* 4	*Score* 5	*Score* 6
Score 1	5	2	1	0	0	0
Score 2	3	8	2	3	0	0
Score 3	0	12	8	11	0	0
Score 4	0	3	8	6	10	0
Score 5	0	2	1	7	6	3
Score 6	1	0	0	2	3	2

Table 6: Transition matrix between the first and the second wave of the Happiness score

w2 \ w3	*Score* 1	*Score* 2	*Score* 3	*Score* 4	*Score* 5	*Score* 6
Score 1	3	4	2	0	0	0
Score 2	3	11	12	0	1	0
Score 3	3	6	9	1	0	0
Score 4	1	2	14	12	1	0
Score 5	0	0	1	7	11	0
Score 6	0	0	0	1	1	3

Table 7: Transition matrix between the second and the third wave of the Happiness score

The ministep that the model will simulate, as the assumption states, can only increase or decrease of one unit; this implies that, for an individual that in the elapsed time between the first two waves passed from *Score* 1 to *Score* 3, at least two ministeps were done. In Figure 6 and 7, as hoped, the values distant from the diagonal are zero or close to, so that the process can be estimated. As it can be observed in Figure 32, the Happiness score decreases with time; this appears in the transition matrices as well, where the most of the values are under the diagonal. The missing data proportion is under the 20% recommended.

5.1.3 The covariates

The other covariates employed in the analysis are the previous school attended, the age, the gender and the class they are attending. The frequency of the pupils' previous school is summarised in Table 8.

School	1	2	3	4	5	6	7
Frequency	83	12	10	1	16	2	1

Table 8: Frequency table of the pupils' previous school

All the 125 pupils belong to the third year, hence their age is between 13 and 16 years old, with an average of 14.3. 64 of the pupils are male and 61 female, and the five classes are all balanced with 25 kids each.

For the behavioural component, the variables used in the models in section 2.2, i.e. *Happy1, Happy2, Happy3, Dissonant1, Dissonant2* and *Dissonant3* were also introduced. This is due to the fact that it is not possible to include in the model, to date, such variables as effects; therefore, in order to take into account these variables, they were introduced as covariates. The flaw of this approach is that these variables are not estimated in each ministep and are not considered in the network evolution, but at every time point a "picture" of the triads condition is inserted in the model.

5.2 The model estimation

The model includes 59 parameters (6 of which are rate parameters), and took 2685 iterations to converge. The convergence of the parameters can require to re-estimate the model giving as starting values the previous estimations of the parameters; for this model the procedure was repeated 3 times. The model was computed on 8 parallel processors end each estimation required up to 40 minutes.

Given the number of parameters included, it was decided to keep the model simple and to exclusively use the evaluation functions. Some models including the endowment function were carried out but, due to the similarity of the effects used, the correlations between the effects resulted in non significant parameters.

The model description will be divided into its three components: *friendship network*, *dislike network* and Happiness score. The significance of the parameters is evaluated through a T-ratio; although the normal distribution has not been proved yet, it can be considered a good approximation [8].

5.2.1 Friendship network

The analysis of the model begins with the *friendship network*, considering first only the elements referring to this level itself and the covariates. The interactions with the other level will be discussed in the following section.

Effects on the friendship network

In Table 9, the first column is the name of the effect, the second is the parameters value, the third refers to the standard errors. The forth column is the convergence t-ratio; the parameter has a good convergence if this value is below 0.1.

Network Dynamics: effects on the friendship network							
	type	Effect	Value	(	S.E.	)	T.conv
1 .	rate	constant friend rate (period 1)	12.57	(	0.94	)	0.05
2 .	rate	constant friend rate (period 2)	12.08	(	0.96	)	0.01
3 .	eval	friend: outdegree (density)	-2.90	(	0.45	)	-0.00
4 .	eval	friend: reciprocity	2.42	(	0.15	)	-0.01
5 .	eval	friend: transitive triplets	0.29	(	0.02	)	-0.01
6 .	eval	friend: transitive recipr. triplets	-0.18	(	0.04	)	-0.01
7 .	eval	friend: 3-cycles	-0.09	(	0.04	)	-0.00
8 .	eval	friend: transitive ties	0.84	(	0.16	)	-0.04
9 .	eval	friend: indegree - popularity	0.01	(	0.04	)	-0.03
10.	eval	friend: indegree - popularity (sqrt)	0.04	(	0.29	)	-0.06
11.	eval	friend: outdegree - popularity	-0.06	(	0.01	)	-0.05
12.	eval	friend: outdegree - activity	-0.01	(	0.00	)	-0.05
13.	eval	friend: sex alter	0.09	(	0.05	)	-0.02
14.	eval	friend: transitive triplets same sex	0.06	(	0.01	)	-0.02
15.	eval	friend: same prev	-0.08	(	0.05	)	-0.07
16.	eval	friend: same class	0.49	(	0.08	)	-0.04
17.	eval	friend: same class x reciprocity	-0.43	(	0.14	)	-0.02

Table 9: Summary of the ABSM: section concerning the *friendship network*. The first numeric column contains the values of the parameters, the second their standard errors and last the convergence in the parameters computations

«The evaluation function is a weighted sum of "effects" $s_{ik}^{\mathbf{X}}(x)$. These, however, are defined as a function of the whole network $\mathbf{x}$, and in most cases the contribution of a single tie variable x_{ij} is just a simple component of this formula. The contribution to $s_{ik}^{\mathbf{X}}(x)$ of adding a tie $i \rightarrow h$ minus the contribution of adding the tie $i \rightarrow j$ is the log odds ratio comparing the probabilities of i sending the tie to j, if all other effects $s_{ik}^{\mathbf{X}}(x)$ yields the same values for these hypothetical new configurations» [18]. Starting from now, the evaluation of the effect of each parameter will be considered coeteris paribus. The analysis of the parameters estimated in the model will start form the ones in Table 9:

1. *The rate parameter* for period 1 is 12.57; therefore, for every actor, 12.57 changes on the *friendship network* are expected on average.

2. *The rate parameter* for period 2 is 12.08; therefore, for every actor, 12.08 changes on the *friendship network* are expected on average. This value is slightly lower than the first, but they still seems homogeneous.

3. *Out-degree*, the number of outgoing ties, has a parameter of -2.9; this means that the more outgoing ties an individual has, the less probable it is that he is going to create a new tie. The parameter on the t-ratio is strongly significant.

4. *Reciprocity*, i.e. the log odd ratio of adding a new tie when the opposite one is already existing. This effect is obtained from: $\sum_j x_{ij}x_{ji}$. The starting point is $i \leftarrow j$ and the new tie in evaluation is $i \rightarrow j$, leading to the condition $i \leftrightarrow j$. The odd ratio is positive (11.24) and strongly significant.

5. *Transitive triplets*, the new tie completes a new triad. Triads are defined as complete if one of these two structures happen:

   $\{i \rightarrow j \rightarrow h; i \rightarrow h\}$ or $\{i \rightarrow h \rightarrow j; i \rightarrow j\}$. The odd ratio is positive (1.34) and strongly significant.

6. *Transitive reciprocated triplets*, this means that the triad is closed and all the individuals are friends with each other. The starting condition is

$i \leftarrow j \rightarrow h$ and the tie on evaluation is $i \rightarrow j$ which leads to $i \leftrightarrow j \rightarrow h$. The odd ratio is 0.83, and it is statistically significant. The negative value (-0.18) has to be considered together with the value of point 4 because, once a reciprocal relation is closed, this effect has to be taken into account. It is possible to state that the reciprocated ties log odd ratio is 0.18 smaller if this tie is in a triad.

7. *The 3-cycles* consider a different kind of triads structured as $i \rightarrow j \rightarrow h \rightarrow i$. The evaluation is, as always, on $i \rightarrow j$. The parameter is negative, small (-0.09) and statistically significant. This effect is the *Happy1* effect. The odd ratio is 0.91, which means that it is less likely to close this kind of triads leading to a balanced condition. The presence of the aforementioned triads (5.) with an highly positive effect must be furthermore considered.

8. *Transitive ties* are similar to transitive triplets but, instead of considering how many two-paths $i \rightarrow h \rightarrow j$ are there for each actor j, it only consideres whether there is at least one such indirect connection. The parameter is positive and strongly significant.

9. - 10. *In-degree* is the number of incoming ties. These two parameters, one on the sum and one on the squared sum, are not significant but they are included in order to have a better fitting of the network, since the in-degree is one of the aspects evaluated.

11. *Out-degree popularity* reflects tendencies to dispersion in in-degrees of the column units, negative and significant.

12. *Out-degree activity* reflects tendencies to dispersion in out-degrees of the actors, negative and significant.

13. *Sex alter* is the gender of the alter with whom the tie is evaluated; it was found that there is no same gender effect but, generally, males are more likely to receive friend nomination; the log odd ratio between male and female is 0.09 (1.10 odd ratio) and it is statistically significant.

14. *Transitive triplets with the same sex* are more likely to be closed than mixed ones. The odd ratio is 1.06 and it is strongly significant.

15. *Same previous school* evaluates the probability of creating a tie if the two pupils come from the same previous school; the log odd ratio is negative -0.08 (odd ratio 0.92) and barely significant.

16. *Same class* evaluates the probability of a tie given that the two pupils come from the same class. The odd ratio is 1.63; it is therefore more likely to be friend if the same class is attended, as it was observed in the graph.

17. *A reciprocity tie in the same class* is, in on the other hand, less likely than a non reciprocated tie; the odd ratio is 1.53, that means that it is more probable to be nominated as a friend in the class but less probable to find reciprocated friendship than outside the class.

The effect of similar Happiness scores was not significant in the *friendship network*, which could mean that the stress level does not affect the ego nor the alter for what concerns the creation of a tie. Observing these results, it can be noticed that pupils that already nominate a lot of friends are less likely to create new ties; moreover, the reciprocity is very important when a new tie is evaluated, as it could be expected. It is furthermore possible to infer that, within the class, friend nominations are more likely but it is less likely to have reciprocated friendship relations; this could be due to the fact that the pupils nominate class mates as friends by default, but this does not necessarily means a real friendship relation. The more reciprocated friendship relations, i.e. the more tight ones, are out of the belonging class.

Effects of the interactions on the friendship network

The aim now is to evaluate the interaction effects on the network structure. In this section the previous mentioned triads, both alone and in interaction with the behavioural variable, are included.

18. If a tie is evaluated by i towards the individual j but on the *dislike network* there is $j \rightarrow i$, the first tie is more than 9 time less likely to be

Network Dynamics: interactions on the friendship network							
	type	Effect	Value	(	S.E.	)	T.conv
18.	eval	friend: recip. with enemy x age similarity	-2.21	(	1.08	)	0.01
19.	eval	int. friend: happ. ego x 3-cycles (happy1)	0.00	(	0.02	)	0.01
20.	eval	friend: happ.ego x closure of enemy(happy2)	0.00	(	0.13	)	0.05
21.	eval	friend: happ.ego x XWX clos. of enemy(diss1)	-0.00	(	0.04	)	0.00
22.	eval	friend: happ.ego x enemy to agreement(diss2)	-0.05	(	0.08	)	0.05
23.	eval	friend: enemy	-1.55	(	0.83	)	-0.03
24.	eval	friend: enemy to agreement	0.08	(	0.08	)	-0.03
25.	eval	friend: XWX closure of enemy	-0.07	(	0.04	)	-0.05
26.	eval	friend: closure of enemy	0.19	(	0.12	)	-0.07

Table 10: Summary of the ABSM: section concerning the interaction between *friendship network*, *dislike network* and Happiness score. The first column contains the parameters values, the second their standard errors, and the last the convergence in the parameters computations

created; the odd ratio is 0.11, and it is significant. In this parameter it is additionally included the effect of the age similarity; when the age is the same, this effect is stronger and it loses power when the age difference increases;

19. The effect of *Happy1* triads weighted for the Happiness score and the out-degree (this last was inserted by software requirements). The effect is not significant, and very low (0.0047); if linked with an high Happiness score and an high out-degree, it has a positive effect on the creation of a tie.

20. As in point 19, the effect of *Happy2* has a positive parameter but in this case the effect is even lower and the significance basically zero. This was already expected from the models in section 2.2.

21. The effect of *Dissonant1* is negative and non significant. In order to include this variable in the model, the same previous procedure was applied: the effect is weighted for the out-degree and the Happiness score.

22. The effect of *Dissonant2* is negative, bigger than *Dissonant1* but still non significant.

In conclusion, the interactions of the behavioural variable, weighted for the out-degree and the triads, are not significant for the creation of a new tie, even though the sign of the effects is what was expected from the analysis conducted in section 2.2.

23. *Enemy* counts the number of dislikes the alter nominates. The more the dislike nominations, the lower the probability to nominate the individual as a friend. The odd ratio is 1.23 and it is significant.

24. *Enemy to agreement* is the effect of closing a *Dissonant2* triad in the network. Unexpectedly, the outcome is positive: it is more likely to close a dissonant triad; the magnitude of the log odd ratio is small and the parameter is not significant.

25. *XWX closure* is the *Dissonant1* effect and it is negative and significant. This outcome was expected, since this effect was negative and significant also in the models of section 2.2. Furthermore, it confirms the theory, according to which it is less likely to close an imbalanced triad. The odd ratio is 0.93.

26. *Closure of enemy* is the *Happy2* effect; it is positive and significant. This agrees with the theory as well: is more likely to close a triad when the outcome is balanced. The odd ratio is 1.21.

The effects are approximatively as expected: the reciprocity with someone disliking the ego actor is not likely, the interaction between the behavioural variable and the triads in evaluating new ties has the expected direction but it is not significant. On the other hand, when there is the chance, a balanced triad is more likely to be closed; this means that the actor improves the overall satisfaction with the balanced condition, and the opposite is true for the imbalanced condition.

5.2.2 Dislike network

The aim of this section is to evaluate the part of the model referring to the *dislike network*; it is divided into two parts: the network itself and the possible interactions.

Effects on the dislike network

The effects used in the model and summarised in Table 11 are to a large extent the same used for the *friendship network*.

Network Dynamics: effects on the dislike network							
	type	Effect	Value	(	S.E.	)	T.conv
27.	rate	constant enemy rate (period 1)	8.24	(	0.91	)	-0.02
28.	rate	constant enemy rate (period 2)	6.40	(	0.59	)	0.02
29.	eval	enemy: outdegree (density)	-4.88	(	0.32	)	0.04
30.	eval	enemy: reciprocity	2.43	(	0.32	)	-0.01
31.	eval	enemy: transitive triplets	0.17	(	0.08	)	-0.03
32.	eval	enemy: indegree - popularity	-0.06	(	0.05	)	0.03
33.	eval	enemy: indegree - popularity (sqrt)	0.90	(	0.24	)	-0.08
34.	eval	enemy: outdegree - activity	0.03	(	0.01	)	0.01
35.	eval	enemy: sex ego	0.24	(	0.08	)	0.05
36.	eval	enemy: same sex	0.46	(	0.09	)	-0.00
37.	eval	enemy: same sex x reciprocity	-0.71	(	0.29	)	-0.01
39.	eval	enemy: same prev	0.36	(	0.10	)	0.03
39.	eval	enemy: same prev x reciprocity	-0.58	(	0.33	)	0.06
40.	eval	enemy: transitive triplets same prev	-0.44	(	0.23	)	0.00
41.	eval	enemy: happiness ego	-0.11	(	0.05	)	0.02
42.	eval	enemy: age alter	-0.11	(	0.06	)	0.00
43.	eval	enemy: same class	0.97	(	0.10	)	0.00
44.	eval	enemy: same class x reciprocity	-1.43	(	0.32	)	-0.02

Table 11: Summary of the ABSM: section concerning the *dislike network*. The first column contains the parameters values, the second their standard errors, and the last the convergence in the parameters computations

27. - 28. The two rate parameters are lower then the ones in the *friendship network*; only 8.2 changes during the first period on average and 6.2 in the second.

29. *Out-degree*; the more outgoing ties an individual has, the less likely it is for him to create a new one. The odd ratio is approximately 0.01 and it is strongly significant.

30. *Reciprocity*; it is more probable to dislike an individual that dislikes the ego. The odd ratio is 11.35 and it is strongly significant.

31. *Transitive triplets* can be seen as *Dissonant3* triads and, unexpectedly, it has a positive and significant effect. This effect determines that when

a tie can be created and it is going to create a *Dissonant3* imbalanced condition, it is more likely that such a triad will be closed. This does not contradict Heider's theories: indeed it could be that, since the relations are not reciprocated, one actor could be not affected by such a triad because the dislike towards an other actor can bound the first in caring about the opinion of this second actor.

In and out-degree have the same meaning as the parameters estimated in the *friendship network*.

34. *Sex ego* is significant and positive, which means that it is likely that females express a new dislike tie. The odd ratio is 1.27, and it is significant. The effect of the alter is not significant, which means that there is not preference in disliking one of the two genders.

35. *Same sex reciprocity* is negative and significant, which means that it is unlikely to have a reciprocated dislike relation within the same gender.

The effects between 36 and 40 are interpreted in the same way as the one in the *friendship network*.

41. *Happiness ego*; interestingly enough, in this case the happiness level affects the probability to create a new dislike tie. The effect is negative, hence the more an individual is happy, the less it is likely that a new dislike relation is created.

42. *Age alter* is the effect of the age of the receiver. The older the individual, the more unlikely he/she is to receive a dislike relation. The effect is significant.

43. - 44. *Class effects*; is the same as in the *friendship network*, therefore it is more likely that someone in the same class is disliked but, in the same class, it is not likely to have a proper dislike relation, where both individuals dislike each other.

Effects of the interactions on the dislike network

The aim of this section is to analyse the interactions between the two levels. Some effects, although including both levels, affect only one.

Network Dynamics: interactions on the dislike network							
	type	Effect	Value	(	S.E.	)	T.conv
45.	eval	int. enemy: friend x same class	-2.09	(	0.74	)	0.05
46.	eval	en. happ. ego x XWX clo. of friend(happy3)	0.01	(	0.02	)	0.04
47.	eval	int. enemy: happ. ego x 3-cycles(diss3)	0.39	(	0.17	)	-0.01
48.	eval	en.: XWX closure of friend	0.26	(	0.03	)	-0.01

Table 12: Summary of the ABSM: section concerning the interaction between *dislike network*, *friendship network* and Happiness score. The first column contains the parameters values, the second their standard errors, and the last the convergence in the parameters computations

45. *Enemy friend same class* is an interaction with the *friendship network* and the class. If an individual has an high out-degree in the *friendship network* and is in the same class of the ego, it is less likely that he/she will dislike him/her. The odd ratio is 0.12 and it is highly significant.

46. *XWX closure of friend*, is the *Happy3* triad weighted on the out-degree and the Happiness score. It is positive and non significant; this means that the higher the Happiness score and out-degree of an individual, the more likely it is that a triad will be closed.

47. *The interaction between happiness and 3-cycles* is *Dissonant3*. It has a positive and significant effect. This effect is weighted on the out-degree and the Happiness score, as usual. The odd ratio is 1.47.

48. *XWX closure* is the presence of a *Happy3* unweighted triad and it has a positive and significant effect. The odd ratio is 1.3; it is therefore more likely that a tie is created if this can close a balanced triad.

5.2.3 The behavioural dynamics

The aim of this section is to evaluate the behavioural dynamics related to the balanced and imbalanced triads. In this analysis, exclusively those triads and some basic effects were introduced, mainly due the non significance of the parameters.

Network Dynamics: effects on the Happiness score							
	type	Effect	Value	(	S.E.	)	T.conv
49.	rate	rate happiness (period 1)	2.57	(	0.64	)	0.01
50.	rate	rate happiness (period 2)	1.67	(	0.34	)	0.01
51.	eval	behavior happiness linear shape	-0.21	(	0.09	)	0.01
52.	eval	behavior happiness quadratic shape	-0.16	(	0.04	)	-0.01
53.	eval	behavior happiness: effect from age	-0.33	(	0.14	)	0.00
54.	eval	behavior happiness: effect from happy1_c	0.01	(	0.00	)	0.01
55.	eval	behavior happiness: effect from happy2_c	0.01	(	0.03	)	0.02
56.	eval	behavior happiness: effect from happy3_c	0.00	(	0.02	)	0.03
57.	eval	behavior happiness: effect from dissonant1_c	-0.05	(	0.03	)	0.01
58.	eval	behavior happiness: effect from dissonant2_c	0.01	(	0.01	)	0.04
59.	eval	behavior happiness: effect from dissonant3_c	-0.07	(	0.07	)	0.04

Table 13: Summary of the ABSM: section concerning the Happiness score. The first column contains the parameters values, the second their standard errors, and the last the convergence in the parameters computations

49. -50. *Happiness score rate parameters*; as stated in the preliminary analysis, it is low: only 2.57 change on average in the first period and 1.67 in the second.

51. -52. *Linear shape* is the Happiness score itself and the *quadratic shape* is its quadratic transformation; they are both negative and significant.

In Figure 33, the objective function including exclusively the linear and quadratic effects is represented. It has a local maximum in 1; this means that every time an actor has the chance to change, it is likely that its Happiness score will be downgraded. This could be inferred from the transition matrices in Table 6 and 7, since most of the elements are under the diagonal, which means that they had a downgrade in their Happiness score.

53. *Effect from age*, out of many trials, is the only significant covariate-effect in the model; it is negative, hence a downgrade is more likely than an upgrade.

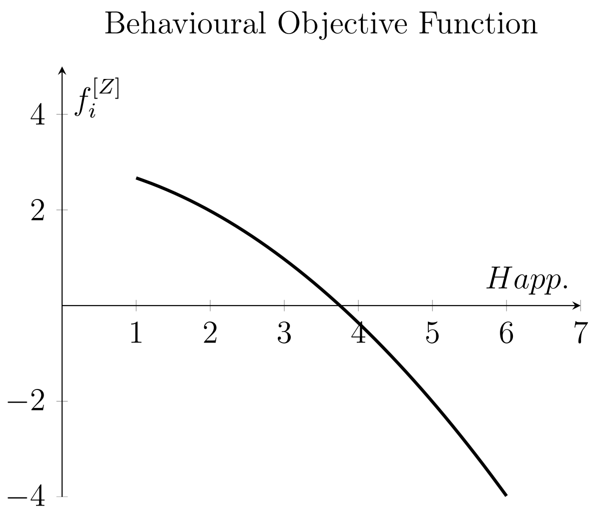


Figure 33: Representation of the behavioural objective function based only on the linear and quadratic effects

The last effects regard the balanced and imbalanced triads; they are included in the model as covariates and not as evolving in the process; this is mainly due to the impossibility, to date, of integrating these effects in the simulation process.

54. *Happy1* is positive and significant, which means that, as expected, the presence of balanced triads of this kind is likely to improve the Happiness score.

55. *Happy2* is positive but not significant.

56. *Happy3* is positive but not significant.

57. *Dissonant1* is negative and significant: the presence of this kind of imbalanced triads, as expected, is likely to decrease the Happiness score of the individual.

58. *Dissonant2* is positive but non significant.

59. *Dissonant3* is negative but non significant.

In order to estimate the Happiness score evolution, other basic effects from the two levels were tested; unfortunately, none resulted significant.

5.3 Goodness of fit

The tests for the significance of each parameter were introduced in the previous section. Moreover, joint tests were conducted; the only set of parameters jointly significant is the effect of the triads on the network itself. Despite the fact that some parameters are singularly significant, the effect of the triads on the behavioural variable is barely significant in the join test. The effect of the Happiness score on the network structure was almost entirely removed because of the lack of significance at the singular level.

Another tool for evaluating the goodness of fit of the model is comparing the observed network with the possible outcomes simulated from the model; the procedure simulates 1000 networks from the model parameters and other statistics of interest, e.g. the in-degree, and compares them with the statistics of the observed network.

The p-value stands for the probability of finding such a network as the one observed in the simulations; the higher the p-value, the better the model. This method of analysing the network can lead to surprising results because is not mandatory that adding more effects, or stressing some others, equals to a more accurate estimation.

Figure 34 is the graphical representation of the test, and on the bottom there is the associated p-value. The observed network (red line) is in-between the 95% confidence band for all in-degrees levels. The p-value is 0.132 and therefore adequate. Figure 35 shows the observed network out-degree distribution compared to the simulated ones. Here, some difficulties in forecasting are faced when the out-degree levels are low. This problem was mitigated by adding to the model some out-degree related elements and by removing some others, consequently reaching a p-value of 0.047 which is barely adequate. Figure 36 shows the observed network against the simulated triads. All the box-plots stand for different triads counts. This plot needed to be rescaled and centered because the large number of triads could be misleading. The p-value is 0.065, which is adequate given the network complexity. The geodesic distance, the shortest path for two nodes to be linked, is another criterium for evaluating the goodness of fit of the model.

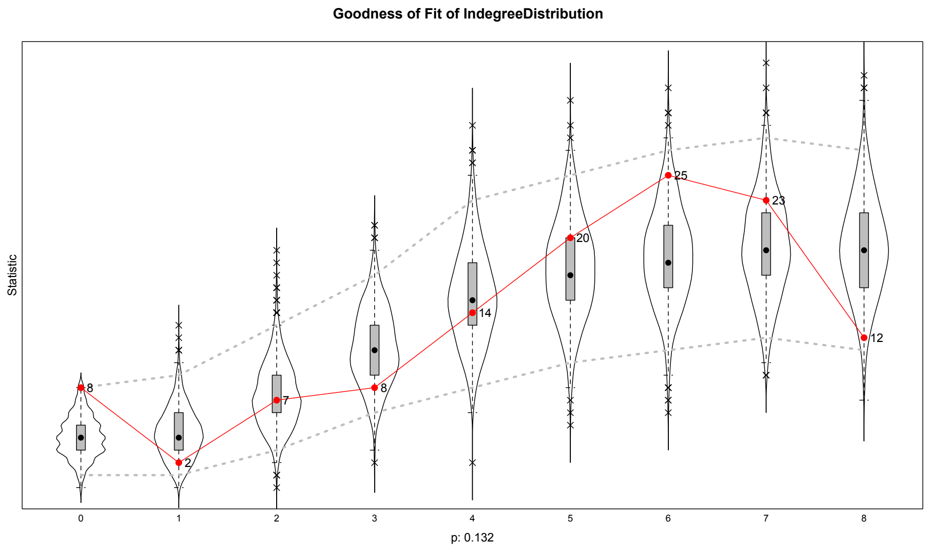


Figure 34: Goodness of fit of the in-degree on the *friendship network*. The observed network is in red and the box-plots represent the in-degree value in the simulated networks. The dotted grey lines indicate the 95% confidence band.

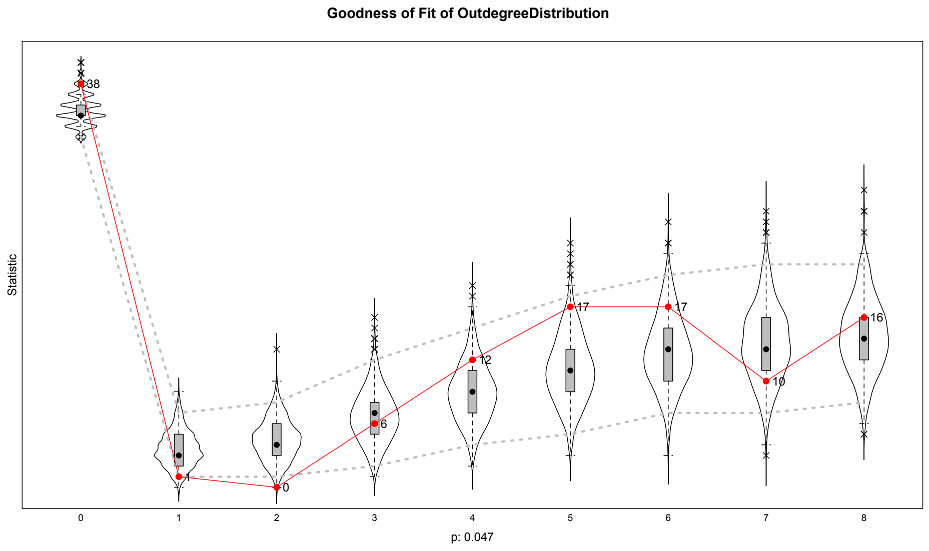


Figure 35: Goodness of fit of the out-degree on the *friendship network*. The observed network is in red and the box-plots represent the out-degree value in the simulated networks. The dotted grey lines indicate the 95% confidence band.

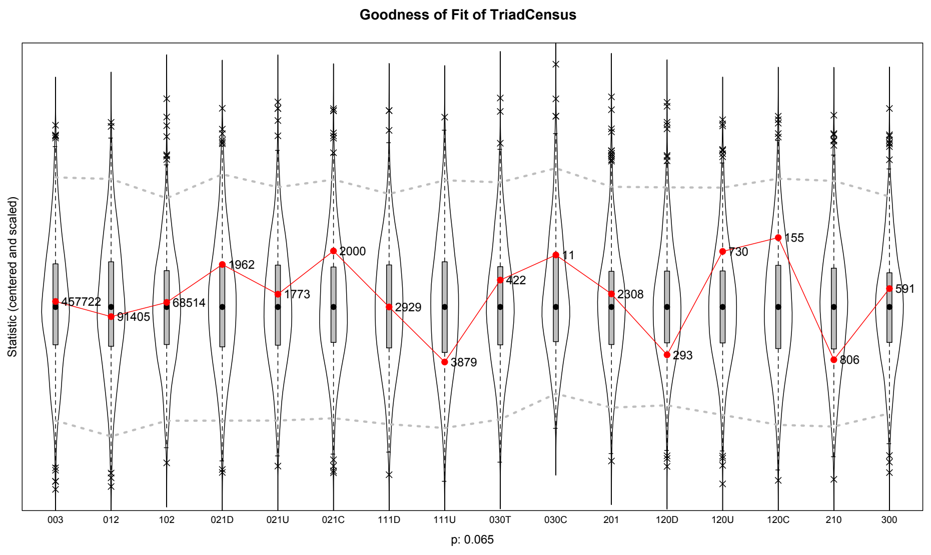


Figure 36: Goodness of fit of the triads on the *friendship network*. The observed network is in red and the box-plots represent the triads number in the simulated networks. The dotted grey lines indicate the 95% confidence band.

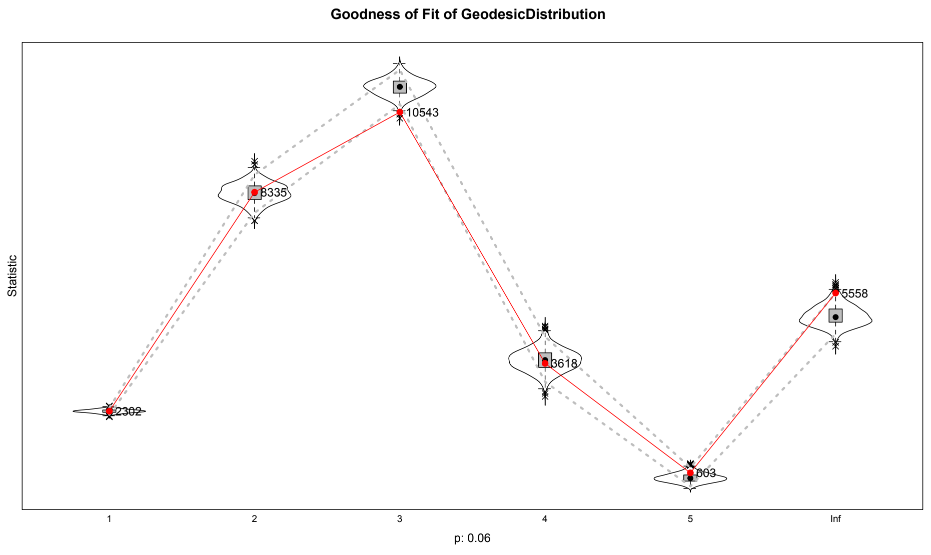


Figure 37: Goodness of fit of the geodesic distance on the *friendship network*. The observed network is in red and the box-plots represent the geodesic distance in the simulated networks. The dotted grey lines indicate the 95% confidence band.

Figure 37 counts the sets of nodes for each geodesic distance, e.g. $1, \ldots, 5$ and infinity (when two nodes are not linked in any path). The fitting is adequate for every geodesic distance except for distance 3 and infinity. The set of nodes with distance 3 is overestimated and the nodes that are not linked in any path are underestimated. This could mean that there is a tendency for the model to keep the nodes linked even if in the observed network they are split; the reason could be that, in the model, only the evaluation function was used. The p-value is 0.06, which can be considered as adequate.

The fitting of the *dislike network* is more accurate, probably due to the fact that this level is less dense and has a smaller change rate. The goodness of fit plot analysed in the *friendship network* will be analysed as well in the *dislike network*.

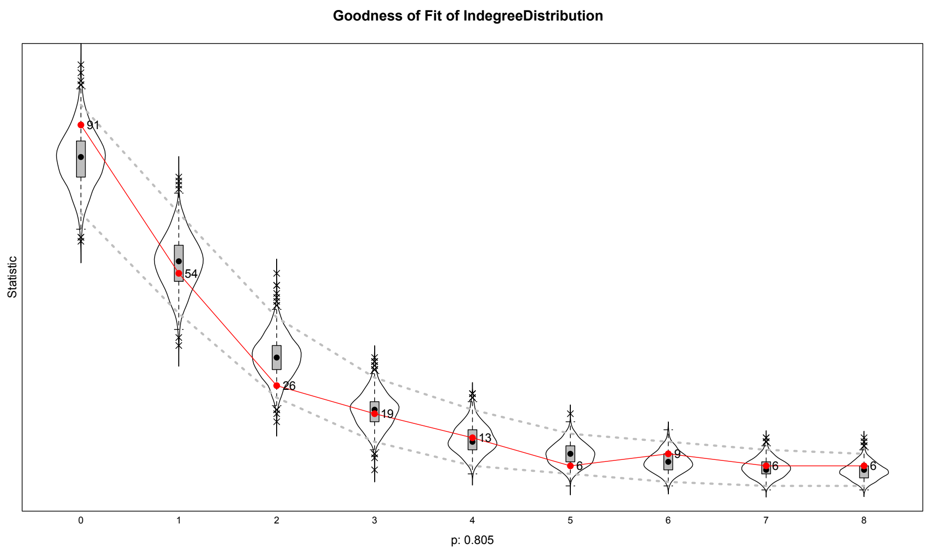


Figure 38: Goodness of fit of the In-degree on the *dislike network*. The observed network is in red and the box-plots represent the in-degree value in the simulated networks. The dotted grey lines indicate the 95% confidence band.

Figure 38 evaluates the goodness of fit of the in-degrees, which is completely satisfactory: the observed network is always included in the 95% confidence band and, moreover, often inside the interquartile range of the box-plot. The p-value is 0.805, which is a good fitting.

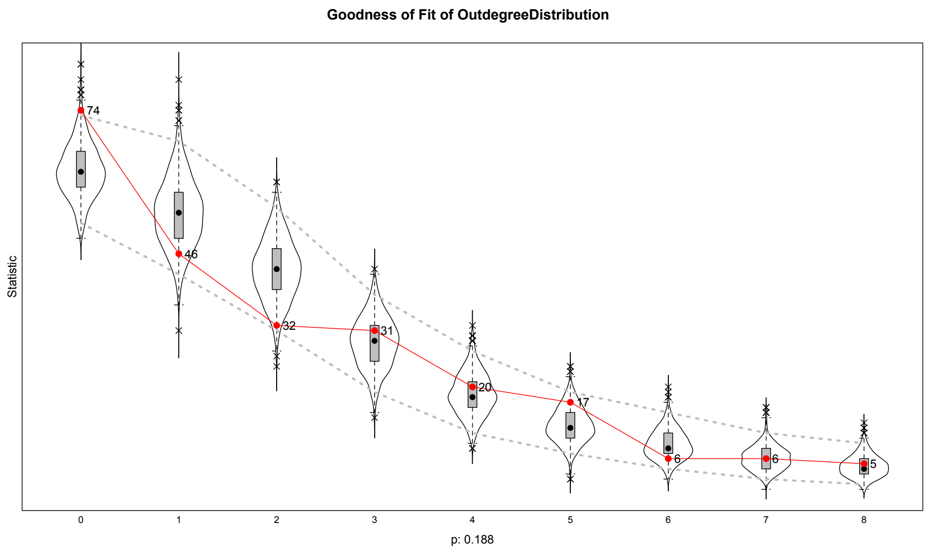


Figure 39: Goodness of fit of the out-degree on the *dislike network*. The observed network is in red and the box-plots represent the out-degree value in the simulated networks. The dotted grey lines indicate the 95% confidence band.

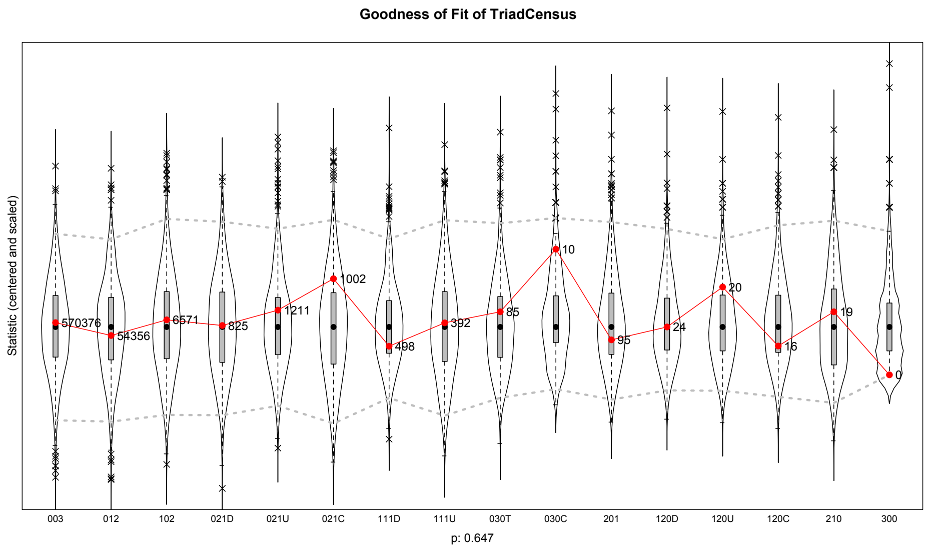


Figure 40: Goodness of fit of the triads on the *dislike network*. The observed network is in red and the box-plots represent the triads number in the simulated networks. The dotted grey lines indicate the 95% confidence band.

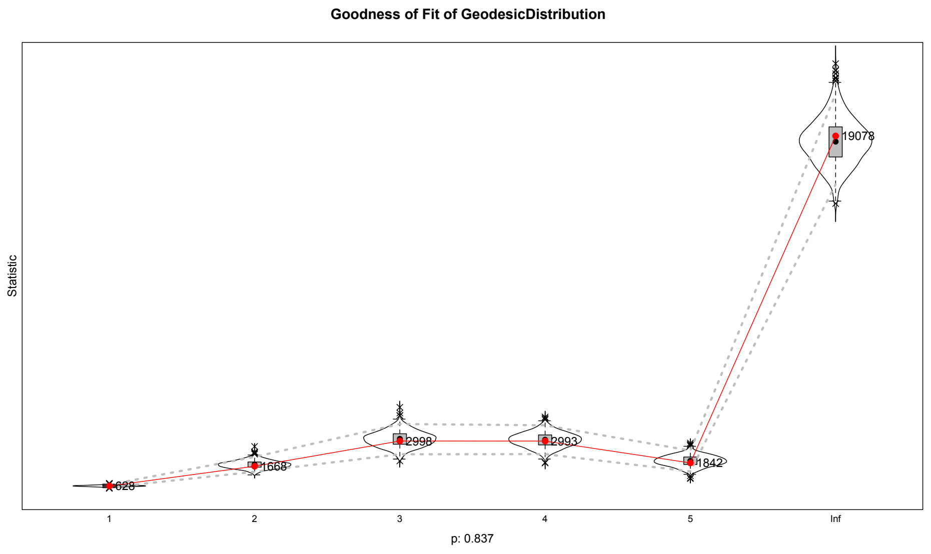


Figure 41: Goodness of fit of the geodesic distance on the *dislike network*. The observed network is in red and the box-plots represent the geodesic distance in the simulated networks. The dotted grey lines indicate the 95% confidence band.

Figure 39 evaluates the goodness of fit of the out-degree; here as well the fitting is satisfactory. The only bad estimated out-degree level is 0, where the model underestimates the number of pupils not disliking any other pupil. The p-value is 0.188, which can be considered as a good fitting.

Figure 40 evaluates the goodness of fit of the triads; the figure was scaled and centered because of the huge difference between the number of different triads. The fitting, here, is good as well; the p-value is 0.647, and the observed network is always in the 95% confidence band.

In Figure 41, the geodesic distance is used as criterium for the goodness of fit; the model is able to satisfactorily simulate the geodesic distance in the *dislike network*. The p-value is 0.837, which ensures a good fitting.

Given these simulations, the model seems to satisfactorily represent the network structure; the fact that, due to its structure, the *dislike network* is fitted better than the *friendship network* stands.

6 Conclusions

In section 2.2, two simple models were introduced; their aim was to evaluate the effect of balanced and imbalanced triads on the stress level of the pupils. Both models were not able to take into consideration some aspects of the network. What emerged from these models is that different kinds of balanced condition have different effects, in terms of sign and significance, on the stress level; accordingly goes for the imbalanced condition. The balanced condition that these two models marked as influencing the stress level is *Happy1*, the agreement in nominating a third individual as friend with someone as well considered friend. The imbalanced condition affecting the stress level is *Dissonant1*, the disagreement in opinion towards a third individual with someone considered as a friend. In the Actor-based Stochastic Model, balanced and imbalanced triads were introduced on more levels: on the behavioural variable, the Happiness score,directly on the network and on the network together with the interaction between Happiness score and out-degree. The balanced triads *Happy1* had a positive effect on the behavioural variable, increasing the Happiness score. The imbalanced triads *Dissonant1*, instead, had a negative effect, therefore decreasing it. All the other balanced and imbalanced triads did not have any significant effect. These result goes accordingly with what was found from the models employed in section 2.2, in particular with the mixed effect model: it as well recognised the two variables as significant. The GAM model estimated as significant also the effect of *Dissonant2*, which in the ABSM model was estimated as positive and non significant.

The effect of the triads on the network structure itself is that individuals are more likely to create a new tie when this allows the creation of a balanced triad. Moreover, the creation of a tie that leads to an imbalanced condition is less likely to happen.

The last series of effects analysed is the interaction between the balanced and imbalanced triads, the Happiness score and the out-degree. The results of the effects are to a large extent statistically non significant, but the direction goes accordingly with the previous results. Actors with an higher Happiness score are more likely to close balanced triads than actors with lower score; actors with higher Happiness score are less likely to close imbalanced triads than actors with lower score. The Happiness score has no effect on the creation of a tie on the *friendship network*, but it has on the *dislike network*. This could mean that the friendship relations are not influenced by the stress level of the actors; the dislike relations, differently, are more likely to happen when the sender has an high stress level.

In conclusion, Heider's theories goes accordingly with the inference on the network data. Not all the balanced triads have the same significant effect and the same stands for the imbalanced triads. What emerged is that the simplest balanced conditions, as agreement with someone considered as a friend towards a third friend, have a significant effect on decreasing the stress level. The same results emerged for the imbalanced triads, where the disagreement with someone considered as a friend toward a third individual, is significant in increasing the stress level.

Triads involving more complex relations, e.g. the triads in which all the actors disliked each other, are more complex to analyse and the focus on their final effect can be easily lost; this could be due to the fact that also other factors, external to the triadic structure, may influence the relations between the actors and consequently the effects.

6.1 Discussion

The model used in the analysis included only the evaluation function, which means that it is more focused in studying the creation of new ties than the maintenance of the existing ones. This choice was led by the autocorrelation in the parameters introduced by the endowment function and the consequent loss of global significance. The improvement in the goodness of fit was not worth this loss of significance in the parameters; moreover, some goodness of fit tests performed worst when the endowment function was used.

The triads were defined as non-reciprocated relations, this could be one of the limiting factors for the study of the more complex triads structure. A different definition could lead to an effects evaluation more focused, but perhaps less reliable since the amount of reciprocated triads would be smaller.

The analysis of the triads effect on the behavioural function was carried out using the count of the triads as a covariate and not including them in the model as evolving effects. This, indeed, weakened the evaluation of those effects. The inclusion in the network had to be weighted also for the out-degree of the actors, with a possible weakening of the meaning of the parameters as well. Further research could focus on extension of the RSiena package in order to estimate those effects directly.

7 Appendix

In this appendix is reported the R code used for all the results in this thesis.

```
## this section creates the functions for extracting the triads from the
## network.
## libraries required
library(snow)
library(RSiena)
library(network)
library(sna)
library("rJava")
library("xlsxjars")
library("xlsx")
library("RSiena")
library("xtable")
library("igraph")
library(psy)
library(mgcv)
library(plm)
library(lmtest)

cl <- makeCluster(3) # this functions runs on 3 parallel processors

##########################################################################
########### function for counting all the unbalanced triads #############
##########################################################################
# the aim of this function is to find all the dissonant triads
dissonant<-function(X){
affected_X<-matrix(NA,ncol=3,nrow=100000) # initialise the triplet matrix
affected_s<-rep(NA,100000) # initialise the matrix
affected_X_2<-matrix(NA,ncol=3,nrow=100000) # initialise the matrix
affected_s_2<-rep(NA,100000) # initialise the matrix
l<-0;n<-0 # initialise the counters
for(i in 1:dim(X)[1]){
        for(j in 1:dim(X)[1]){
         for (k in 1:dim(X)[1]){ # cycle through all the individuals
        if(((X[i,j]==1 & X[j,k]==1 & X[i,k]==-1 )|(X[i,j]==1 & X[j,k]==-1
       & X[i,k]==1 )) & (i!=j & j!=k & i!=k)){ # condition to be dissonant
        l=l+1 # increase the counter
        affected_X[l,] <-c(i,j,k) # save the triplet in matrix form
        affected_s[l]<-paste(i,j,k,sep="-")} # save the triplet in short form
        if(((X[i,j]==-1 & X[j,k]==1 & X[i,k]==1) | (X[i,j]==1 & X[k,j]==1 &
                X[i,k]==-1)) & (i!=j & j!=k & i!=k)){ # condition to bevdissonant
        n=n+1 # increase the counter
        affected_X_2[n,] <-c(i,j,k) # save the triplet in matrix form
        affected_s_2[n]<-paste(i,j,k,sep="-")} # save the triplet in short form
                        }
                }
        }
        aff_c<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X)[1])
                        for(i in 1:dim(X)[1]) aff_c[i,2]<-sum(affected_X[1]==i,na.rm=T)
        # counting the affected triplets for each individual
        aff_c_2<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X[1])
                for(i in 1:dim(X)[1]) aff_c_2[i,2]<-sum(affected_X_2[,1]==i,na.rm=T)

        return(list(unb_X=as.matrix(affected_X[1:l,]),
        unb_s=affected_s[1:l],unb_c=as.matrix(aff_c),
        unb_X_2=as.matrix(affected_X_2[1:n,]),unb_s_2=affected_s_2[1:n],
        unb_c_2=as.matrix(aff_c_2)))
        # returning all the found results
}
#########################################################################
# function for counting all the triads with positive balanced situation#
##########################################################################
# The aim of this function in to find all the happy triplets of both kind, type 1 and type 2
happy<-function(X){
happy_1<-matrix(NA,100000,3) # initialise the 1 type of happiness matrix
happy_2<-matrix(NA,100000,3) # initialise the 2 type of happiness matrix
happy_3<-matrix(NA,100000,3) # initialise the 2 type of happiness matrix
happy_1_s<-rep(NA,100000) # initalise the vector containing the triplets in short form type 1
happy_2_s<-rep(NA,100000) # initalise the vector containing the triplets in short form type 2
happy_3_s<-rep(NA,100000) # initalise the vector containing the triplets in short form type 2
l<-0 ; m<-0;n<-0 # initilise all the counters
for(i in 1:dim(X)[1]){
        for(j in 1:dim(X)[1]){
         for (k in 1:dim(X)[1]) { # cycle through all the individuals
                if((X[i,j]==1 & X[j,k]==1 & X[i,k]==1 ) & (i!=j & j!=k & i!=k)){
                        # condition for happiness of the 1  type
                l=l+1 #increase the counter
```

```
                  happy_1[l,]<-c(i,j,k) # save the triplet in matrix form
                  happy_1_s[l]<-paste(i,j,k,sep="-")} # save the triplet in short form
                  if((X[i,j]==1 & X[j,k]==-1 & X[i,k]==-1) &(i!=j & j!=k &i!=k)){
                           # condition for happiness of the 2  type
                  m=m+1 #increase the counter
                  happy_2[m,]<-c(i,j,k)  # save the triplet in matrix form
                  happy_2_s[m]<-paste(i,j,k,sep="-")} # save the triplet in short form

                  if(((X[i,j]==-1 & X[j,k]==+1 & X[i,k]==-1)| (X[i,j]==-1 &
                           X[j,k]==-1 & X[i,k]==+1))&(i!=j & j!=k &i!=k)){
                             # condition for happiness of the 2  type
                  n=n+1 #increase the counter
                  happy_3[n,]<-c(i,j,k)  # save the triplet in matrix form
                  happy_3_s[n]<-paste(i,j,k,sep="-")} # save the triplet in short form
                                                               }
                                                      }

                                             }
happy_1_c<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X)[1])
                  for(i in 1:dim(X)[1]) happy_1_c[i,2]<-sum(happy_1[,1]==i,na.rm=T)
                  # counting the affected triplets for each individual
happy_2_c<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X)[1])
                  for(i in 1:dim(X)[1]) happy_2_c[i,2]<-sum(happy_2[,1]==i,na.rm=T)
                  # counting the affected triplets for each individual
happy_3_c<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X)[1])
                  for(i in 1:dim(X)[1]) happy_3_c[i,2]<-sum(happy_3[,1]==i,na.rm=T)
                  # counting the affected triplets for each individual
return(list(H1_X=as.matrix(happy_1[1:l,]),H2_X=as.matrix(happy_2[1:m,]),
                           H3_X=as.matrix(happy_3[1:n,]),H1_X_c=as.matrix(happy_1_c),
                           H2_X_c=as.matrix(happy_2_c),H3_X_c=as.matrix(happy_3_c),
                           h_1_s=happy_1_s[1:l],h_2_s=happy_2_s[1:m],
                           h_3_s=happy_3_s[1:n])) # return all the results
                           }

##########################################################################
##########this function finds all the triads  dissonant 3################
##########################################################################
zero<-function(X){
zero_X<-matrix(NA,1000000,3) # initialise the zero matrix
zero_s<-rep(NA,1000000) # initialise the vector for the short form zeros
l<-0 # initialise the counter
for(i in 1:dim(X)[1]){
         for(j in 1:dim(X)[1]){
          for (k in 1:dim(X)[1]){ # cycle through all the individuals

          if((X[i,j]==-1 & X[j,k]==-1 & X[i,k]==-1) & (i!=j & j!=k & i!=k)){
                  # condition for being a zero triplet, mathematically found.
          l<-l+1 # increase the counter
          zero_X[l,] <-c(i,j,k) # save the zeros in a matrix form
          zero_s[l]<-paste(i,j,k,sep="-")} # save the zeros in short form
                           }
                  }
         }
         zero_c<-matrix(c(seq(1:dim(X)[1]),rep(NA,dim(X)[1])),ncol=2,nrow=dim(X)[1])
                           for(i in 1:dim(X)[1]) zero_c[i,2]<-sum(zero_X[,1]==i,na.rm=T)
                           # counting the affected triplets for each individual
return(list(zero_X=as.matrix(zero_X[1:l,]),zero_s=zero_s[1:l],
                           zero_c=as.matrix(zero_c))) # return the results.
}

####################################################################
################## function to create the stability variable ########
# this function counts how many triplets are stable in the time for each unit###

stab<-function(diss,h_1,zero_1){

stability_1<-rep(NA,dim(diss[[1]]$unb_c)[1])
stability_2<-rep(NA,dim(diss[[1]]$unb_c)[1])
for(i in 1:dim(diss[[1]]$unb_c)[1]){
         a <-diss[[1]]$unb_X[,1]==i
         a_1 <-diss[[1]]$unb_X_2[,1]==i
         b <-zero_1[[1]]$zero_X[,1]==i
         c <- h_1[[1]]$H1_X[,1]==i
         d <- h_1[[1]]$H2_X[,1]==i
         e <- h_1[[1]]$H3_X[,1]==i
         stability_1[i]<-(-sum(!(diss[[1]]$unb_s[a]%in%diss[[2]]$unb_s))-
         sum(!(diss[[1]]$unb_s_2[a_1]%in%diss[[2]]$unb_s_2))-
         sum(!(zero_1[[1]]$zero_s[b]%in%zero_1[[2]]$zero_s))-
         sum(!(h_1[[1]]$h_1_s[c]%in% h_1[[2]]$h_1_s))-
         sum(!(h_1[[1]]$h_2_s[d] %in% h_1[[2]]$h_2_s))-
         sum(!(h_1[[1]]$h_3_s[e] %in% h_1[[2]]$h_3_s)) # in 1 and not in 2
         +sum(diss[[1]]$unb_s[a]%in%diss[[2]]$unb_s)+
         sum(diss[[1]]$unb_s_2[a_1]%in%diss[[2]]$unb_s_2)+
          sum(zero_1[[1]]$zero_s[b]%in%zero_1[[2]]$zero_s)+
          sum(h_1[[1]]$h_1_s[c]%in% h_1[[2]]$h_1_s)+
          sum(h_1[[1]]$h_2_s[d]%in% h_1[[2]]$h_2_s)+
          sum(h_1[[1]]$h_3_s[e]%in% h_1[[2]]$h_3_s)) # in 1 and not in 2

          a <-diss[[2]]$unb_X[,1]==i
          a_1 <-diss[[2]]$unb_X_2[,1]==i
          b <-zero_1[[2]]$zero_X[,1]==i
          c <- h_1[[2]]$H1_X[,1]==i
          d <- h_1[[2]]$H2_X[,1]==i
          e <- h_1[[2]]$H2_X[,1]==i

          stability_2[i]<-(-sum(!(diss[[2]]$unb_s[a]%in%diss[[3]]$unb_s))-
          sum(!(diss[[2]]$unb_s_2[a_1]%in%diss[[3]]$unb_s_2))-
          sum(!(zero_1[[2]]$zero_s[b]%in%zero_1[[3]]$zero_s))-
          sum(!(h_1[[2]]$h_1_s[c]%in% h_1[[3]]$h_1_s))-
          sum(!(h_1[[2]]$h_2_s[d] %in% h_1[[3]]$h_2_s))-
          sum(!(h_1[[2]]$h_3_s[e] %in% h_1[[3]]$h_3_s)) # in 2 and not in 3
          +sum(diss[[2]]$unb_s[a]%in%diss[[3]]$unb_s)+
          sum(diss[[2]]$unb_s_2[a_1]%in%diss[[3]]$unb_s_2)+
          sum(zero_1[[2]]$zero_s[b]%in%zero_1[[3]]$zero_s)+
          sum(h_1[[2]]$h_1_s[c]%in% h_1[[3]]$h_1_s)+
          sum(h_1[[2]]$h_2_s[d]%in% h_1[[3]]$h_2_s)+
          sum(h_1[[2]]$h_3_s[e]%in% h_1[[3]]$h_3_s)) # in 2 and in 3

}
return(list(stability_1,stability_2))
}


# importing the daa
#V7a Friends:
#wave1

 friends_w1 <- read.xlsx("data1.xlsx", sheetIndex=1,
 rowIndex = 2:126, colIndex = 9:34,header=F)
 friends_w1[is.na(friends_w1)] <- 0
friendmatrixwave1=mat.or.vec(125,125)
 for (i in 1:125){
 for (j in 1:26){
 k = friends_w1[i,j]
 if(k==1000)friendmatrixwave1[i,]<-matrix(nrow=1,ncol=125)
 else if(k==1001) friendmatrixwave1[i,]<-mat.or.vec(1,125)
 else if(k==0) break
 else {friendmatrixwave1[i,k]<-1}
 j = j+1}
 i=i+1}

teveelvriendenwave1 <- read.xlsx("data1.xlsx",sheetIndex=1,
rowIndex = 2:126, colIndex = 35, header=F)
# om de vrienden van die met te veel op NA te zetten:
for(i in 1:125){
k <- teveelvriendenwave1[i,1]
if (k==1){
for (j in 1:125){
if (friendmatrixwave1[i,j]==1)friendmatrixwave1[i,j]<-NA
j=j+1}
         }
i=i+1}
diag(friendmatrixwave1)=c(0)

#wave 2:

friends_w2 <- read.xlsx("data2.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = 9:36,header=F)
 friends_w2[is.na(friends_w2)] <- 0
 friendmatrixwave2=mat.or.vec(125,125)
 for (i in 1:125){
 for (j in 1:26){
 k = friends_w2[i,j]
 if(k==1000) friendmatrixwave2[i,]<-matrix(nrow=1,ncol=125)
 else if(k==1001) friendmatrixwave2[i,]<-mat.or.vec(1,125)
 else if(k==0) break
 else {friendmatrixwave2[i,k]<-1}
 j = j+1}
 i=i+1}

teveelvriendenwave2 <- read.xlsx("data2.xlsx",sheetIndex=1, rowIndex = 2:126,
colIndex = 37, header=F)
# om de vrienden van die met te veel op NA te zetten:

for(i in 1:125){
k <- teveelvriendenwave2[i,1]
if (k==1) for (j in 1:125){
```

```
if (friendmatrixwave2[i,j]==1) friendmatrixwave2[i,j]<-NA
                  j=j+1}
         i=i+1}
diag(friendmatrixwave2)=c(0)

#wave 3
friends_w3 <- read.xlsx("data3.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = 9:36,header=F)
friends_w3[is.na(friends_w3)] <- 0
friendmatrixwave3=mat.or.vec(125,125)
 for (i in 1:125){
 for (j in 1:26){
 k = friends_w3[i,j]
 if(k==1000) friendmatrixwave3[i,]<-matrix(nrow=1,ncol=125)
 else if(k==1001) friendmatrixwave3[i,]<-mat.or.vec(1,125)
 else if(k==0) break
 else {friendmatrixwave3[i,k]<-1}
 j = j+1}
 i=i+1}

teveelvriendenwave3 <- read.xlsx("data3.xlsx",sheetIndex=1, rowIndex = 2:126,
colIndex = 37, header=F)

# om de vrienden van die met te veel op NA te zetten:

for(i in 1:125){
k <- teveelvriendenwave3[i,1]
if (k==1)
{
for (j in 1:125){
if (friendmatrixwave3[i,j]==1) friendmatrixwave3[i,j]<-NA
                  j=j+1}
         i=i+1}
diag(friendmatrixwave3)=c(0)

friend_network_w1=as.network(friendmatrixwave1)
friend_network_w2=as.network(friendmatrixwave2)
friend_network_w3=as.network(friendmatrixwave3)


# V11 Enemies
#wave 1

enemies_w1 <- read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = 104:119,header=F)
enemies_w1[is.na(enemies_w1)] <- 0
enemies_mat_w1=mat.or.vec(125,125)

 for (i in 1:125){
 for (j in 1:16){
 k = enemies_w1[i,j]
 if(k==1000) enemies_mat_w1[i,]<-matrix(nrow=1,ncol=125)
 else if(k==1001)enemies_mat_w1[i,]<-mat.or.vec(1,125)
 else if(k==0) break
 else {enemies_mat_w1[i,k]<-1}
 j = j+1}
 i=i+1}
diag(enemies_mat_w1)=c(0)

#wave 2
enemies_w2 <- read.xlsx("data2.xlsx", sheetIndex=1,rowIndex = 2:126,
 colIndex = 106:121,header=F)
enemies_w2[is.na(enemies_w2)] <- 0
enemies_mat_w2=mat.or.vec(125,125)
 for (i in 1:125){
 for (j in 1:16){
 k = enemies_w2[i,j]
 if(k==1000) enemies_mat_w2[i,]<-matrix(nrow=1,ncol=125)
 else if(k==1001) enemies_mat_w2[i,]<-mat.or.vec(1,125)
 else if(k==0) break
 else {enemies_mat_w2[i,k]<-1}
 j = j+1}
 i=i+1}
diag(enemies_mat_w2)=c(0)

#wave 3
enemies_w3 <- read.xlsx("data3.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = 106:121,header=F)
enemies_w3[is.na(enemies_w3)] <- 0
enemies_mat_w3=mat.or.vec(125,125)
 for (i in 1:125){
 for (j in 1:16){
 k = enemies_w3[i,j]
```

```
  if(k==1000) enemies_mat_w3[i,]<-matrix(nrow=1,ncol=125)
  else  if(k==1001)enemies_mat_w3[i,]<-mat.or.vec(1,125)
  else  if(k==0) break
  else {enemies_mat_w3[i,k]<-1}
  j = j+1}
  i=i+1}
 diag(enemies_mat_w3)=c(0)

enemy_net_w1<-as.network(enemies_mat_w1)
enemy_net_w2<-as.network(enemies_mat_w2)
enemy_net_w3<-as.network(enemies_mat_w3)

Z<-as.matrix.network(enemy_net_w1)
Z[is.na(Z)]<-0
T<-as.matrix.network(friend_network_w1)
T[is.na(T)]<-0
X1<--Z+T

Z<-as.matrix.network(enemy_net_w2)
Z[is.na(Z)]<-0
T<-as.matrix.network(friend_network_w2)
T[is.na(T)]<-0
X2<--Z+T

Z<-as.matrix.network(enemy_net_w3)
Z[is.na(Z)]<-0
T<-as.matrix.network(friend_network_w3)
T[is.na(T)]<-0
X3<--Z+T


################################################################################
######################## QUESTIONS SELECTION ###########################
################################################################################

happ_wave1 <-read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = c(158:176),header=F)
colnames(happ_wave1)<-c("16g","17a","17b","17c","17d","17e","17f","17g",
       "17h","17i","17j","17k","17l","17m","17n","17o","17p","17q","17r")

happ_wave1[which(is.na(happ_wave1),arr=T)]<-
round(rowMeans(happ_wave1[which(is.na(happ_wave1),arr=T)[,1],],na.rm=T))

happ_wave1[which(happ_wave1==1000,arr=T)]<-NA

neg<-which(names(happ_wave1)%in%c("17c","17f","17i","17l","17p"))

happ_wave1[,c(neg)]<-6-happ_wave1[,c(neg)]

sel <- c("16g","17a","17f","17l","17m","17q","17r")

select<-which(names(happ_wave1)%in%sel)

happ_wave1<-happ_wave1[,c(select)]

######
happ_wave2 <-read.xlsx("data2.xlsx", sheetIndex=1,rowIndex = 2:126,
 colIndex = c(160:178),header=F)

colnames(happ_wave2)<-c("16g","17a","17b","17c","17d","17e","17f","17g",
           "17h","17i","17j","17k","17l","17m","17n","17o","17p","17q","17r")

happ_wave2[which(is.na(happ_wave2),arr=T)]<-
 round(rowMeans(happ_wave2[which(is.na(happ_wave2),arr=T)[,1],],na.rm=T))

happ_wave2[which(happ_wave2==1000,arr=T)]<-NA

neg<-which(names(happ_wave2)%in%c("17c","17f","17i","17l","17p"))

happ_wave2[,c(neg)]<-6-happ_wave2[,c(neg)]

select<-which(names(happ_wave2)%in%sel)

happ_wave2<-happ_wave2[,c(select)]

######
happ_wave3 <-read.xlsx("data3.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = c(160:178),header=F)
colnames(happ_wave3)<-c("16g","17a","17b","17c","17d","17e","17f","17g",
"17h","17i","17j","17k","17l","17m","17n","17o","17p","17q","17r")

happ_wave3[which(is.na(happ_wave3),arr=T)]<-
round(rowMeans(happ_wave3[which(is.na(happ_wave3),arr=T)[,1],],na.rm=T))
```

```
happ_wave3[which(happ_wave3==1000,arr=T)]<-NA

neg<-which(names(happ_wave3)%in%c("17c","17f","17i","17l","17p"))

happ_wave3[,c(neg)]<-6-happ_wave3[,c(neg)]

select<-which(names(happ_wave3)%in%sel)

happ_wave3<-happ_wave3[,c(select)]

round(cor(rbind(happ_wave1[!is.na(happ_wave1[,1]),],
                happ_wave2[!is.na(happ_wave2[,1]),],
                happ_wave3[!is.na(happ_wave3[,1]),])),3)

cor(happ_wave1[!is.na(happ_wave1[,1]),])
cor(happ_wave2[!is.na(happ_wave2[,1]),])
cor(happ_wave3[!is.na(happ_wave3[,1]),])
cronbach(happ_wave3[!is.na(happ_wave3[,1]),])
cronbach(happ_wave1[!is.na(happ_wave1[,1]),])
cronbach(happ_wave2[!is.na(happ_wave2[,1]),])
cronbach(happ_wave3[!is.na(happ_wave3[,1]),])

happ_w1<-rowSums(happ_wave1,na.rm=F)
happ_w2<-rowSums(happ_wave2,na.rm=F)
happ_w3<-rowSums(happ_wave3,na.rm=F)

happiness<-c(happ_w1,happ_w2,happ_w3)

### other covariate selection
happy_1 <- c(h_1[[1]]$H1_X_c[,2],h_1[[2]]$H1_X_c[,2],h_1[[3]]$H1_X_c[,2])
happy_2 <- c(h_1[[1]]$H2_X_c[,2],h_1[[2]]$H2_X_c[,2],h_1[[3]]$H2_X_c[,2])
happy_3 <- c(h_1[[1]]$H3_X_c[,2],h_1[[2]]$H3_X_c[,2],h_1[[3]]$H3_X_c[,2])

zeros <- c(zero_1[[1]]$zero_c[,2],zero_1[[2]]$zero_c[,2],zero_1[[3]]$zero_c[,2])
dissonant <- c(diss[[1]]$unb_c[,2],diss[[2]]$unb_c[,2],diss[[3]]$unb_c[,2])
dissonant_2 <- c(diss[[1]]$unb_c_2[,2],diss[[2]]$unb_c_2[,2],diss[[3]]$unb_c_2[,2])

data_stress_all<-data.frame(happiness, happy_1, happy_2,happy_3, dissonant_1=dissonant,
dissonant_2,dissonant_3=zeros,time=c(rep(1,125),
rep(2,125),rep(3,125)),id=c(seq(1:125),seq(1:125),seq(1:125)))

data_stress_all<-data_stress_all[!is.na(data_stress_all$happiness),]

data_stress_all$id<-as.factor(data_stress_all$id)
data_stress_all$time<-as.factor(data_stress_all$time)

data_stress_all<-data_stress_all[!rowSums(data_stress_all[,2:7])==0,]


data_stress_all_log<-data.frame(log(data_stress_all[,c(1:7)]+1), data_stress_all[,c(8:9)])
##### descriptive

pairs(happiness~happy_1+happy_2+happy_3,data=data_stress_all_log,
      main="Correlation Between Happiness and Balanced Triads")
pairs(happiness~dissonant_1+dissonant_2+dissonant_3,data=data_stress_all_log,
      main="Correlation Between Happiness and Imbalanced Triads")

bau<-data_stress_all_log$id[data_stress_all_log$time==2]
plot(data_stress_all_log$happiness[data_stress_all_log$id %in% bau
& data_stress_all_log$time==1],data_stress_all_lo$happiness[data_stress_all_log$id
%in% bau & data_stress_all_log$time==2])

ht1<-data.frame(ht1=data_stress_all_log$happiness[data_stress_all_log$time==1],
id1=data_stress_all_log$id[data_stress_all_log$time==1])

ht2<-data.frame(ht2=data_stress_all_log$happiness[data_stress_all_log$time==2],
id2=data_stress_all_log$id[data_stress_all_log$time==2])

ht3<-data.frame(ht3=data_stress_all_log$happiness[data_stress_all_log$time==3],
id3=data_stress_all_log$id[data_stress_all_log$time==3])


Time_1<-ht1$ht1[(ht1$id %in% ht2$id) & (ht1$id %in% ht3$id)]
Time_2<-ht2$ht2[(ht2$id %in% ht1$id) & (ht2$id %in% ht3$id)]
Time_3<-ht3$ht3[(ht3$id %in% ht1$id) & (ht3$id %in% ht2$id)]
pairs(~Time_1+Time_2+Time_3, main="Correlation in Log-Happiness in Different Time Points")


summary(log(data_stress_all$happiness))
par(mfrow=c(1,3))
hist(log(data_stress_all$happiness[data_stress_all$time==1]),breaks=25)
hist(log(data_stress_all$happiness[data_stress_all$time==2]),breaks=25)
hist(log(data_stress_all$happiness[data_stress_all$time==3]),breaks=25)
```

```
par(mfrow=c(1,1))


summary(log(data_stress_all$happiness[data_stress_all$time==1]))
summary(log(data_stress_all$happiness[data_stress_all$time==2]))
summary(log(data_stress_all$happiness[data_stress_all$time==3]))
par(mfrow=c(1,3))
hist(log(data_stress_all$happiness[data_stress_all$time==1]),breaks=20,
main="Time Point 1",xlab="Log-Happiness")
hist(log(data_stress_all$happiness[data_stress_all$time==2]),breaks=20,
main="Time Point 2",xlab="Log-Happiness")
hist(log(data_stress_all$happiness[data_stress_all$time==3]),breaks=20,
main="Time Point 3",xlab="Log-Happiness")
par(mfrow=c(1,1))
### GAM Model
gam4<-gam(happiness~s(happy_1,bs="cr")+s(happy_2,bs="cr")+s(happy_3,bs="cr")
+s(dissonant_1,bs="cr")+s(dissonant_2,bs="cr")+s(dissonant_3,bs="cr")+time,
data=data_stress_all_log)
summary(gam4)
plot.gam(gam4,residual=T,page=1)

plot(data_stress_all_log$happiness[data_stress_all_log$time==2],
data_stress_all_log$happiness[data_stress_all_log$time==3])
##################
gam5<-gam(log(happiness)~s(happy_1,bs="cr")+s(happy_2,bs="cr")+s(happy_3,bs="cr")+
s(zeros,bs="cr")+s(I(dissonant+dissonant_2),bs="cr")+time,data=data_stress_all)

summary(gam5)
plot.gam(gam5,residual=T,page=1)
gam.check(gam5)
## diagnostic
rsd<-residuals(gam5)
gam(rsd~s(happy_1,k=20,bs="cs"),gamma=1.4,data= data_stress_all)
gam(rsd~s(happy_2,k=20,bs="cs"),gamma=1.4,data= data_stress_all)
gam(rsd~s(happy_3,k=20,bs="cs"),gamma=1.4,data= data_stress_all)

gam(rsd~s(zeros,k=10,bs="cs"),gamma=1.4,data= data_stress_all)
gam(rsd~s(dissonant,k=20,bs="cs"),gamma=1.4,data= data_stress_all)
gam(rsd~s(dissonant_2,k=20,bs="cs"),gamma=1.4,data= data_stress_all)

plot.gam(gam5,residuals=T,scale=-1,pages=1,
main="Regression Splines and Residuals not all log")


#### Mixed effect Model

panel<-plm.data(data_stress_all_log,c("id","time"))
pdim(panel)
str(panel)
form<-happiness~happy_1+happy_2+happy_3+dissonant_1+dissonant_2+dissonant_3

fre <- plm(happiness~happy_1+happy_2+happy_3+zeros+dissonant+dissonant_2+
        time, model="random",data = panel,random.method ="swar")
        summary(fre)
        coeftest(fre, vcov=vcovHC(fre, cluster="group"))

fpo <- plm(form, model="pooling",data = panel)
        summary(fpo)
        coeftest(fpo)
        coeftest(fpo, vcov=vcovHC(fpo, method = "white1",type = "HC0"))

# within
        pvar(panel)

        fpw=plm(form,panel, effect ="time",model = "within")
        summary(fpw)
        coeftest(fpw)
        coeftest(fpw, vcov=vcovHC(fpw, cluster="group"))

        f=fixef(fpw,type="level")
        as.matrix(f)

        summary(f)

        mean(f)
        var(f)

## test di poolability (modello pooled vs modello within

        pFtest(fpw,fpo)

#       pooltest(fpo, fpw)
```

```
         fpv=pvcm(form,panel,effect="time",model="within")
         summary(fpv)

         pooltest(fpo, fpv)
         pooltest(fpw, fpv)

         fpv=pvcm(form,panel,effect="time",model="within")
#        str(fpv)
         summary(fpv)
         pooltest(fpo, fpv)
# poolability test not refused


################################################################

## Effetti casuali individuali RE model


## random.method = c("swar",
#        "walhus","amemiya","nerlove", "kinla"),



         fre <- plm(form, model="random",effect="individual",data = panel,
         random.method ="swar")
         summary(fre)
         coeftest(fre, vcov=vcovHC(fre, cluster="group"))

ercomp(fre)
par(mfrow=c(1,3))
hist(data_stress_all_log$happy_1[data_stress_all_log$time==1],breaks=25)
hist(data_stress_all_log$happy_1[data_stress_all_log$time==2],breaks=25)
hist(data_stress_all_log$happy_1[data_stress_all_log$time==3],breaks=25)

# choice between FE and RE
         phtest(fpw,fre)
#### RE accepted
plot(fre)
res<-residuals(fre)
qqnorm(res)
qqline(res)

par(mfrow=c(2,3))
plot(res~happy_1,data=data_stress_all_log,ylab="Residuals",
xlab="Happy_1",main="Happy_1 vs Residuals")

plot(res~happy_2,data=data_stress_all_log,ylab="Residuals",
xlab="Happy_2",main="Happy_2 vs Residuals")

plot(res~happy_3,data=data_stress_all_log,ylab="Residuals",
xlab="Happy_3",main="Happy_3 vs Residuals")

plot(res~ dissonant_1,data=data_stress_all_log,ylab="Residuals",
xlab="Dissonant_1",main="Dissonant_1 vs Residuals")

plot(res~ dissonant_2,data=data_stress_all_log,ylab="Residuals",
xlab="Dissonant_2",main="Dissonant_2 vs Residuals")

plot(res~ dissonant_3,data=data_stress_all_log,ylab="Residuals",
xlab="Dissonant_3",main="Dissonant_3 vs Residuals")

par(mfrow=c(1,1))


#V1 Sex
sex <- read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 2,header=F)

#V2 age
#wave 1
age_w1 <- read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 3,header=F)

#wave 2
age_w2 <- read.xlsx("data2.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 3,header=F)

#wave 3
age_w3 <- read.xlsx("data3.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 3,header=F)


#V3 Class
#wave 1
class_w1 <- read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 4,header=F)

#wave 2
class_w2 <- read.xlsx("data2.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 4,header=F)
```

```
#wave 3
class_w3 <- read.xlsx("data3.xlsx", sheetIndex=1,rowIndex = 2:126, colIndex = 4,header=F)

#V4 vorigeschool
previous_school_wave<- read.xlsx("data1.xlsx", sheetIndex=1,rowIndex = 2:126,
colIndex = 5,header=F)

for(i in 1: 125){
k= previous_school[i,1]
if(k==1){
previous_school[i,1]<-2
}
i=i+1
}

#V1 geslacht als coCovar
sex=coCovar(as.vector(as.matrix(sex)))
#V2 leeftijd als varCovar
agewave1matrix=data.matrix(age_w1)
agewave2matrix=data.matrix(age_w2)
agewave3matrix=data.matrix(age_w3)

age=varCovar(cbind(agewave1matrix, agewave2matrix, agewave3matrix))

#V3 klassen als varDyadCovar:
classwave1matrix=data.matrix(class_w1)
classwave2matrix=data.matrix(class_w2)
classwave3matrix=data.matrix(class_w3)

class<-varCovar(cbind(classwave1matrix, classwave2matrix, classwave3matrix))

friend_network <- sienaNet(array( c( friendmatrixwave1,
friendmatrixwave2, friendmatrixwave3),dim = c( 125, 125, 3 ) ) )


###calss 2

cla<-which(classwave1matrix==2)
friend_matrix_w1_2<-friendmatrixwave1[cla, cla]
friend_network_w1_2=as.network(friend_matrix_w1_2)


friend_matrix_w2_2<-friendmatrixwave2[cla ,cla]
friend_network_w2_2=as.network(friend_matrix_w2_2)

friend_matrix_w3_2<-friendmatrixwave3[cla, cla]
friend_network_w3_2=as.network(friend_matrix_w3_2)


par(mfrow=c(1,3))
coordinaten=plot( friend_network_w1_2, xlab = 'Firends class 2 wave 1',label=c(1:125))
plot( friend_network_w2_2, xlab = 'vrienden wave 2',label=c(1:125),coord=coordinaten)
plot( friend_network_w3_2, xlab = 'vrienden wave 3',label=c(1:125),coord=coordinaten)
par(mfrow=c(1,1))

## create the absm model
previous_school=coCovar(as.vector(as.matrix(previous_school_wave)))

data = sienaDataCreate( friend_network, sex, age, class, previous_school)

eff <- getEffects(data)

eff <- includeEffects(eff, sameX, interaction1 = "sex")
eff <- includeEffects( eff, egoX, interaction1="sex")
eff <- includeEffects( eff, altX, interaction1="sex")


print01Report( data, eff, modelname = 'prova1' )



model <- sienaModelCreate(useStdInits = FALSE, projname = 'prova1')
ans <- siena07( model, data = data, effects = eff,
 batch=TRUE, verbose=TRUE, useCluster=TRUE, initC=TRUE, nbrNodes=2,returnDeps=TRUE)
summary(ans)
xtable(ans)
print(xtable(ans))

length(agewave3matrix[which(classwave1matrix==2)])
length(sex[which(classwave1matrix==2)])
length(previous_school_wave$X5[which(classwave1matrix==2)])
####now try with a class only
```

```
sex_2=coCovar(as.vector(as.matrix(sex[which(classwave1matrix==2)])))
age_2=varCovar(cbind(agewave1matrix[which(classwave1matrix==2)],
 agewave2matrix[which(classwave1matrix==2)], agewave3matrix[which(classwave1matrix==2)]))
pre_school_2=coCovar(as.vector(as.matrix(previous_school_wave
$X5[which(classwave1matrix==2)])))

friend_network_2 <- sienaNet(array( c( friend_matrix_w1_2, friend_matrix_w2_2,
 friend_matrix_w3_2),dim = c( 25, 25, 3 ) ) )


data = sienaDataCreate( friend_network_2, sex_2, age_2, pre_school_2)

eff <- getEffects(data)
eff <- includeEffects(eff, inPop)
eff <- includeEffects(eff, inAct)
eff <- includeEffects(eff, outPop)
eff <- includeEffects(eff, outAct)
eff <- includeEffects(eff, sameX, interaction1 = "age_2")
print01Report( data, eff, modelname = 'prova2' )



model <- sienaModelCreate(useStdInits = FALSE, projname = 'prova2')
ans <- siena07( model, data = data, effects = eff,
 batch=TRUE, verbose=TRUE, useCluster=TRUE, initC=TRUE, nbrNodes=2,returnDeps=TRUE)
summary(ans)
xtable(ans)
print(xtable(ans))
#####################################
############simulation
obj<-function(X,i,j,b1=-1.2924,b2=1.6714,b3=0.0680,b4=-0.0297,b5=0.0880){
fi<-exp(b1*sum(X[i,])+b2*t(X[i,])%*%X[,i]+b3*sum(X[,j])+
b4*sum(X[j,])+b5*((sex_s2[i]==sex_s2[j])*X[i,j]))
return(fi)
}

obj(X_0,1,13)
div<-rep(NA,25)
for(j in 1:25) if(i!=j) div[j]=sum(obj(X_0,25,j))
denom<-sum(div,na.rm=T)

simul<-function(X,ti=0,top){
change<-matrix(NA,top,2)
time<-rep(NA,top)
k<-1
t<-1
for(t in 1:top){
ti=ti+rexp(1,rate=4.09)
time[t]<-ti
i<-sample(c(1:25),1)
change[k,1]<-i
div<-rep(NA,25)
for(j in 1:25) if(j!=i) div[j]=sum(obj(X_0,i,j))
prob<-div/sum(div,na.rm=T)
prob[is.na(prob)]<-0
change[k,2]<-sample(c(1:25),1,prob=prob)
X[i,change[k,2]]<-1-X[i,change[k,2]]
k=k+1
t=t+1
}
return(list(X,change,time))
}
result<-simul(X_0,top=10)
str(result)
sum(!result[[1]]== X_0)

X_1=as.network(result[[1]])

result2<-simul(X_0,top=50)

par(mfrow=c(1,2))
coordinaten=plot( as.network(X_0), xlab = 'vrienden wave 1',label=c(1:125))
plot( X_1, xlab = 'vrienden wave 2',label=c(1:125),coord=coordinaten)
plot( as.network(result2[[1]]),
xlab = 'vrienden wave 2',label=c(1:125),coord=coordinaten)

par(mfrow=c(1,1))

#######################################
# simulation for the two mode network ##
#######################################
obj_f<-function(X,Y,i,j,b1=-1.4896,b2=1.6299,b3=0.0972,b4=-0.0234,b5=-2.1670){
```

```
fi<-exp(b1*sum(X[i,])+b2*t(X[i,])%*%X[,i]+b3*sum(X[,j])
+b4*sum(X[j,])+b5*(t(X[i,])%*%Y[,i]))
return(fi)
}
obj_e<-function(X,Y,i,j,b1=-3.6062,b2=-7.2755,b3=0.5022,b4=-0.2340,b5=-0.5554){
fi<-exp(b1*sum(X[i,])+b2*t(X[i,])%*%X[,i]+b3*sum(X[,j])
+b4*sum(X[j,])+b5*(t(X[i,])%*%Y[,i]))
return(fi)
}

sim_2<-function(X,Y,n=100,lf=4.1482,le=1.7266){
X[is.na(X)]<-0;Y[is.na(Y)]<-0
t<-0; sam<-c(1:25)
simul<-matrix(NA,n,6)
colnames(simul)<-c("time","network","focal_act","tie","prev_value","succ_value")
for(i in 1:n){
        dt<-rexp(1,rate=(le+lf)); t=t+dt; simul[i,1]<-round(t,3)
        net<-sample(c(1:2),1,prob=c(lf/(lf+le),le/(lf+le))); simul[i,2]<-net
        act<-sample(c(1:25),1); simul[i,3]<-act
                if(net==1){
                        for(j in 1:25) if(j!=act) div[j]=sum(obj_f(X,Y,act,j))
                        prob1<-div/sum(div,na.rm=T)
                        prob1[is.na(prob1)]<-0
                        prob1<-prob1[-act]
                        act2<-sample(sam[-act],1,prob=prob1)
                                        simul[i,4]<-act2
                                        simul[i,5]<-X[act,act2]
                                        X[act,act2]<-1-X[act,act2]
                                        simul[i,6]<-X[act,act2]

                        }
                else {
                        for(j in 1:25) if(j!=act) div[j]=sum(obj_e(X,Y,act,j))
                                prob1<-div/sum(div,na.rm=T)
                                prob1[is.na(prob1)]<-0
                                prob1<-prob1[-act]

                                act2<-sample(sam[-act],1,prob=prob1)
                                simul[i,4]<-act2
                                simul[i,5]<-Y[act,act2]
                                Y[act,act2]<-1-Y[act,act2]
                                simul[i,6]<-Y[act,act2]

                        }
        }
return(list(X,Y,simul))
}
pr<-sim_2(friend_matrix_w1_2, enemies_mat_w1_2,n=50)

## plots
coordinaten=plot( friend_network_w1_2, xlab = 'Firends class 2 wave 1',label=c(1:125))
plot( enemies_network_w1_2,coord=coordinaten,col=3,edge.col=2,new=F)


plot( as.network(pr[[1]]),label=c(1:25),coord=coordinaten)
plot( as.network(pr[[2]]),coord=coordinaten,edge.col=2,new=F)

barplot(prob,names.arg=c("Friend Net","Dislike Net"),ylim=c(0,1),
ylab="Probability",xlab="Networks",main="Probability Distribution")

barplot(prob2,names.arg=c(1:17,19:25),ylim=c(0,.1),ylab="Probability",
xlab="Actors",main="Probability Distribution in the Friends Network")

barplot(prob3,names.arg=c(1:17,19:25),ylim=c(0,.5),ylab="Probability",
xlab="Actors",main="Probability Distribution in the Dislike Network")

barplot(ps,names.arg=c(1:13,15:25),ylim=c(0,.08),ylab="Probability",
xlab="Actors",main="Probability Distribution the Friendship Network")


#TriadCensus:
TriadCensus <- function(i, data, sims, wave, groupName, varName, levls=1:16){
       unloadNamespace("igraph") # to avoid package clashes
       require(sna)
       require(network)
       x <- networkExtraction(i, data, sims, wave, groupName, varName)
       tc <- sna::triad.census(x)[1,levls]
       # names are transferred automatically
       tc
   }

#GeodesicDistribution:
GeodesicDistribution <- function (i, data, sims, period, groupName,
                            varName, levls=c(1:5,Inf), cumulative=TRUE, ...) {
```

```
      x <- networkExtraction(i, data, sims, period, groupName, varName)
      require(sna)
      a <- sna::geodist(x)$gdist
      if (cumulative)
      {
        gdi <- sapply(levls, function(i){ sum(a<=i) })
      }
          else
      {
        gdi <- sapply(levls, function(i){ sum(a==i) })
      }
      names(gdi) <- as.character(levls)
      gdi
    }

#coord<-plot( friend_network_w1+ friend_network_w2+ friend_network_w3)
happiness
happiness_s<-as.matrix(happiness)
happiness_s[happiness_s==1]<-"black"
happiness_s[happiness_s==2]<-"blue"
happiness_s[happiness_s==3]<-"red"
happiness_s[happiness_s==4]<-"orange"
happiness_s[happiness_s==5]<-"yellow"
happiness_s[happiness_s==6]<-"green"

coord<-plot( friend_network_w1+ friend_network_w2+ friend_network_w3)

lc.txt<-c("class 1","class 2","class 3","class 4","class 5")
lh.txt<-c("lev < 20 ","20 <= lev < 23","23 <= lev < 26",
"27 <= lev < 29","29 <= lev < 32","lev > 32")
le.txt<-c("Friend Relation","Dislike Relation")

friend_network_w1%v%"class"<-class[,1]
friend_network_w1%v%"happiness"<-as.matrix(happiness_s)[1:125]
deg <- rowSums( as.matrix( friend_network_w1 ) )

par(mfrow=c(1,2))
plot(friend_network_w1,vertex.col="happiness",edge.col="grey",
coord=coord, vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.5,
main="Friend time point 1 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(friend_network_w1,vertex.col="class",edge.col="grey",
coord=coord, vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.5,
main="Friend time point 1 with dimension outdegree and color class")

legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class",cex=0.7)
par(mfrow=c(1,1))


friend_network_w2%v%"class"<-class[,2]
friend_network_w2%v%"happiness"<-as.matrix(happiness_s)[126:250]
deg <- rowSums( as.matrix( friend_network_w2 ) )

par(mfrow=c(1,2))
plot(friend_network_w2,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.7,
main="Friend time point 2 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(friend_network_w2,vertex.col="class",edge.col="grey",coord=coord,
 vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.7,
 main="Friend time point 2 with dimension outdegree and color class")
legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class")
par(mfrow=c(1,1))

friend_network_w3%v%"class"<-class[,3]
friend_network_w3%v%"happiness"<-as.matrix(happiness_s)[251:375]
deg <- rowSums( as.matrix( friend_network_w3) )

par(mfrow=c(1,2))
plot(friend_network_w3,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.7,
main="Friend time point 3 with dimension outdegree and color happiness")
legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(friend_network_w3,vertex.col="class",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.7,
main="Friend time point 3 with dimension outdegree and color class")

legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class")
par(mfrow=c(1,1))

enemy_net_w1<-as.network(enemies_mat_w1)
enemy_net_w2<-as.network(enemies_mat_w2)
enemy_net_w3<-as.network(enemies_mat_w3)


enemy_net_w1%v%"class"<-class[,1]
enemy_net_w1%v%"happiness"<-as.matrix(happiness_s)[1:125]
deg <- rowSums( as.matrix( enemy_net_w1 ) )

par(mfrow=c(1,2))

plot(enemy_net_w1,vertex.col="happiness",edge.col="grey",coord=coord,
 vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
 main="enemies time point 1 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(enemy_net_w1,vertex.col="class",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
main="enemies time point 1 with dimension outdegree and color class")
legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class")
par(mfrow=c(1,1))


enemy_net_w2%v%"class"<-class[,2]
enemy_net_w2%v%"happiness"<-as.matrix(happiness_s)[126:250]
deg <- rowSums( as.matrix( enemy_net_w2 ) )

par(mfrow=c(1,2))
plot(enemy_net_w2,vertex.col="happiness",edge.col="grey",coord=coord,
 vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
 main="Enemy time point 2 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(enemy_net_w2,vertex.col="class",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
main="Enemy time point 2 with dimension outdegree and color class")

legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class")
par(mfrow=c(1,1))

enemy_net_w3%v%"class"<-class[,3]
enemy_net_w3%v%"happiness"<-as.matrix(happiness_s)[251:375]
deg <- rowSums( as.matrix( enemy_net_w3) )

par(mfrow=c(1,2))
plot(enemy_net_w3,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
main="Enemy time point 3 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

plot(enemy_net_w3,vertex.col="class",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.7,
main="Enemy time point 3 with dimension outdegree and color class")

legend(12,19,lc.txt,pch=19,col=c(1:5),title="Class")
par(mfrow=c(1,1))


par(mfrow=c(1,2))
deg <- rowSums( as.matrix( friend_network_w1 ) )
plot(friend_network_w1,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.5,
main="Friend time point 1 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

deg <- rowSums( as.matrix( enemy_net_w1 ) )
plot(enemy_net_w1,vertex.col="happiness",edge.col="orange",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.5,
main="enemies time point 1 with dimension outdegree and color happiness")
```

```
legend(-18,19,le.txt,pch=4,col=c("grey","orange"),title="Arrows",cex=0.7)

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)
par(mfrow=c(1,1))

par(mfrow=c(1,2))
deg <- rowSums( as.matrix( friend_network_w2 ) )
plot(friend_network_w2,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.5,
main="Friend time point 1 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

deg <- rowSums( as.matrix( enemy_net_w2 ) )
plot(enemy_net_w2,vertex.col="happiness",edge.col="orange",coord=coord,
vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.5,
main="enemies time point 1 with dimension outdegree and color happiness")
legend(-19,19,le.txt,pch=4,col=c("grey","orange"),title="Arrows",cex=0.7)
legend(12,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)
par(mfrow=c(1,1))

par(mfrow=c(1,2))
deg <- rowSums( as.matrix( friend_network_w3 ) )
plot(friend_network_w3,vertex.col="happiness",edge.col="grey",coord=coord,
vertex.cex = (deg + 1)/15,label=c(1:125),label.cex=0.5,
main="Friend time point 1 with dimension outdegree and color happiness")

legend(11,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)

deg <- rowSums( as.matrix( enemy_net_w3 ) )
plot(enemy_net_w3,vertex.col="happiness",edge.col="orange",coord=coord,
 vertex.cex = (deg + 1)/7,label=c(1:125),label.cex=0.5,
 main="enemies time point 1 with dimension outdegree and color happiness")
legend(-19,19,le.txt,pch=4,col=c("grey","orange"),title="Arrows",cex=0.7)
legend(13,19,lh.txt,pch=19,col=c("black","blue","red","orange","yellow","green"),
title="Happiness Score",cex=0.7)
par(mfrow=c(1,1))

par(mfrow=c(1,3))
hist(happ_w1c,breaks=5,main="Happiness Score",xlab="Happiness Score Wave 1",
cex.lab=1.4,ylim=c(0,40))
hist(happ_w2c,breaks=5,main="Happiness Score",xlab="Happiness Score Wave 2",
cex.lab=1.4,ylim=c(0,40))
hist(happ_w3c,breaks=5,main="Happiness Score",xlab="Happiness Score Wave 3",
cex.lab=1.4,ylim=c(0,40))
par(mfrow=c(1,1))

table(happ_w1c,happ_w2c)
table(happ_w2c,happ_w3c)

par(mfrow=c(1,3))
hist(age_w1,breaks=5,main="Happiness Score",xlab="Happiness Score Wave 1",
cex.lab=1.4)
hist(age_w2,breaks=5,main="Happiness Score",xlab="Happiness Score Wave 2",
cex.lab=1.4)
hist(as.vector(age_w3)[,1],breaks=3,main="Happiness Score",
xlab="Happiness Score Wave 3",cex.lab=1.4)
par(mfrow=c(1,1))

hist(as.matrix(prev))
xtable(table(as.matrix(prev)))

## start with finding a model without the behavioural variable
Jaccard(friendmatrixwave1, enemies_mat_w1)
Jaccard(friendmatrixwave2, enemies_mat_w2)
Jaccard(friendmatrixwave3, enemies_mat_w3)

siena07ToConvergence <- function(alg, dat, eff){
numr <- 0
ans <- siena07(alg, data=dat, effects=eff,
useCluster=TRUE,initC=TRUE,nbrNodes=8) # the first run
repeat {
     numr <- numr+1           # count number of repeated runs
     maxt <- max(abs(ans$tconv[!eff$fix[eff$include]]))
               # convergence indicator, excluding the fixed effects
     cat(numr, maxt,"\n")     # report how far we are
     if (maxt < 0.10) {break} # success
     if (maxt > 5) {break}    # divergence without much hope
                                # of returning to good parameter values
     if (numr > 10) {break}  # now it has lasted too long
     ans <- siena07(alg, data=dat, effects=eff,prevAns=ans,
     useCluster=TRUE,initC=TRUE,nbrNodes=8)
}
ans
}
siena07ToConvergence(multiAlg, multidata_n,multieff)

# data set creation
data_n <- sienaDataCreate(friend,enemy,sex,age,class,prev,
happiness,happy1_n,happy2_n,happy3_n,dissonant1_n,dissonant2_n,dissonant3_n)
# creation of the effects
eff <- getEffects(data_n)

print01Report(data_n,modelname="two_networks")
eff <- includeEffects(eff,cycle3, inPop,inPopSqrt,transRecTrip ,transTies,
outPop,outAct,name="friend")
eff <- includeEffects(eff,inPop,inPopSqrt ,outAct,name="enemy")
### the effects commented were removed for lack of significance

#eff <- includeEffects(eff, sameX, interaction1 = "sex")
#eff <- includeEffects( eff, egoX, interaction1="sex")
eff <- includeEffects( eff, altX, interaction1="sex")
#eff <- includeEffects(eff, sameXRecip, interaction1 = "sex")
eff <- includeEffects(eff, sameXTransTrip, interaction1 = "sex")

#eff <- includeEffects(eff, simX, interaction1 = "age")
#eff <- includeEffects( eff, egoX, interaction1="age")
#eff <- includeEffects( eff, altX, interaction1="age")
eff <- includeEffects(eff, sameX, interaction1 = "class")
#eff <- includeEffects( eff, egoX, interaction1="class")
#eff <- includeEffects( eff, altX, interaction1="class")
eff <- includeEffects(eff, sameXRecip, interaction1 = "class")
#eff <- includeEffects(eff, sameXTransTrip, interaction1 = "class")

eff <- includeEffects(eff, sameX, interaction1 = "prev")
#eff <- includeEffects(eff, sameXRecip, interaction1 = "prev")
#eff <- includeEffects(eff, sameXTransTrip, interaction1 = "prev")
#eff <- includeEffects( eff, altX, interaction1="age")

eff <- includeEffects(eff, sameX,name="enemy", interaction1 = "sex")
eff <- includeEffects( eff, egoX,name="enemy", interaction1="sex")
#eff <- includeEffects( eff, altX, name="enemy",interaction1="sex")
eff <- includeEffects(eff, sameXRecip,name="enemy", interaction1 = "sex")
#eff <- includeEffects(eff, sameXTransTrip, name="enemy",interaction1 = "sex")

#eff <- includeEffects(eff, simX, name="enemy",interaction1 = "age")
#eff <- includeEffects( eff, egoX,name="enemy", interaction1="age")
eff <- includeEffects( eff, altX, name="enemy",interaction1="age")
eff <- includeEffects(eff, sameX,name="enemy", interaction1 = "class")
#eff <- includeEffects( eff, egoX, name="enemy",interaction1="class")
#eff <- includeEffects( eff, altX,name="enemy", interaction1="class")
eff <- includeEffects(eff, sameXRecip, name="enemy",interaction1 = "class")
#eff <- includeEffects(eff, sameXTransTrip,name="enemy", interaction1 = "class")

eff <- includeEffects(eff, sameX, name="enemy",interaction1 = "prev")
eff <- includeEffects(eff, sameXRecip, name="enemy",interaction1 = "prev")
eff <- includeEffects(eff, sameXTransTrip,name="enemy", interaction1 = "prev")
eff <- includeEffects( eff, altX, name="enemy",interaction1="prev")

#happy1
eff <- includeEffects(eff,transTrip,name="friend") #sig
#happy2
eff <- includeEffects(eff,closure,name="friend",interaction1="enemy") # non sig by now
#happy3
eff <- includeEffects(eff,cl.XWX, name ="enemy",interaction1="friend") #sig
#diss1
eff <- includeEffects(eff,cl.XWX, name ="friend",interaction1="enemy") # sig by now
#diss2
eff <- includeEffects(eff,to, name ="friend",interaction1="enemy") # non sig
#diss3
eff <- includeEffects(eff,transTrip,name="enemy") # sig

eff <- includeEffects(eff,crprod,name="friend",interaction1="enemy")

#eff <- includeEffects(eff,crprod,name="enemy",interaction1="friend")

#eff <- includeEffects(eff,crprodRecip,name="friend",interaction1="enemy")

#eff <- includeEffects(eff,crprodRecip,name="enemy",interaction1="friend")


eff <-includeInteraction(eff, crprod,sameX,name="enemy",
interaction1=c("friend","class"))
#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","class"))

#eff <-includeInteraction(eff, crprod,sameX,name="enemy",
#interaction1=c("friend","prev"))
#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","prev"))

#eff <-includeInteraction(eff, crprod,sameX,name="enemy",
#interaction1=c("friend","sex"))
#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","sex"))

#eff <-includeInteraction(eff, crprod,simX,name="enemy",interaction1=c("friend","age"))
eff <-includeInteraction(eff, crprodRecip,simX,name="friend",
interaction1=c("enemy","age"))

#eff <- includeEffects(eff, simX, name="friend",interaction1="happiness")
#eff <- includeEffects(eff, egoX,name="friend", interaction1="happiness")
#eff <- includeEffects(eff, altX, name="friend",interaction1="happiness")


#eff <- includeEffects(eff, simX, name="enemy",interaction1="happiness")
eff <- includeEffects(eff, egoX, name="enemy",interaction1="happiness")
#eff <- includeEffects(eff, altX, name="enemy",interaction1="happiness")


#happy1
eff <-includeInteraction(eff, cycle3,egoX,name="friend",
interaction1=c("","happiness"))
#happy2
eff <-includeInteraction(eff, closure,egoX,name="friend",
interaction1=c("enemy","happiness"))
#happy3
eff <-includeInteraction(eff, cl.XWX,egoX,name="enemy",
interaction1=c("friend","happiness"))

#diss1
eff <-includeInteraction(eff, cl.XWX,egoX,name="friend",
interaction1=c("enemy","happiness"))
#diss2
eff <-includeInteraction(eff, to, egoX,name="friend",
interaction1=c("enemy","happiness"))

#diss3
eff <-includeInteraction(eff,cycle3,egoX,name="enemy",
interaction1=c("","happiness"))

### effects for behaviour
#eff <- includeEffects(eff, indeg, name="happiness", interaction1="friend")
#eff <- includeEffects(eff, outdeg, name="happiness", interaction1="friend")
eff <- includeEffects(eff, effFrom, name="happiness", interaction1="age")
#eff <- includeEffects(eff, effFrom, name="happiness", interaction1="sex")

#eff <- includeEffects(eff, isolate, name="happiness", interaction1="friend")
eff <- includeEffects(eff, behDenseTriads, name="happiness", interaction1="freind")

#eff <- includeEffects(eff, indeg, name="happiness", interaction1="enemy")
#eff <- includeEffects(eff, outdeg, name="happiness", interaction1="enemy")

#eff <- includeEffects(eff, isolate, name="happiness", interaction1="enemy")
#eff <- includeEffects(eff, behDenseTriads, name="happiness", interaction1="enemy")

##### covariate per balanced and imbalanced on behaviour
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy1_n")
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy2_n")
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy3_n")
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="dissonant1_n")
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="dissonant2_n")
eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="dissonant3_n")

multiAlg <- sienaAlgorithmCreate(projname = 'two_networks_try3.R', seed=123 )
ans3 <- siena07(multiAlg, data = data_n, effects = eff,
useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE)
save("ans3",file="ans7.rda")

ans3 <- siena07(multiAlg, data = data_n, effects = eff,
useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE,prevAns= ans3)
save("ans3",file="ans7.rda")

ans3 <- siena07(multiAlg, data = data_n, effects = eff,
useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE,prevAns= ans3)
```

```
save("ans3",file="ans7.rda")

### test
cbind(which(abs(ans3$theta/(ans3$se))>1.5),
(ans3$theta /(ans3$se))[abs(ans3$theta /(ans3$se))>1.5])

gofi <- sienaGOF(ans3, IndegreeDistribution, verbose=TRUE, join=TRUE,
varName="friend",cumulative=FALSE)

gofo <- sienaGOF(ans3, OutdegreeDistribution, verbose=TRUE, join=TRUE,
                     varName="friend", cumulative=FALSE)
goft <- sienaGOF(ans3, TriadCensus, verbose=TRUE, join=TRUE,
                     varName="friend")
gofg <- sienaGOF(ans3, GeodesicDistribution, verbose=TRUE, join=TRUE,
                     varName="friend",cumulative=FALSE)

plot(gofi)
plot(gofo)
plot(goft,scale=T,center=T)
plot(gofg)


gofi_e <- sienaGOF(ans3, IndegreeDistribution, verbose=TRUE, join=TRUE,
varName="enemy",cumulative=FALSE)

gofo_e <- sienaGOF(ans3, OutdegreeDistribution, verbose=TRUE, join=TRUE,
                     varName="enemy", cumulative=FALSE)
goft_e <- sienaGOF(ans3, TriadCensus, verbose=TRUE, join=TRUE,
                     varName="enemy")
gofg_e <- sienaGOF(ans3, GeodesicDistribution, verbose=TRUE, join=TRUE,
                     varName="enemy",cumulative=FALSE)
plot(gofi_e)
plot(gofo_e)
plot(goft_e,scale=T,center=T)
plot(gofg_e)

gofg_e <- sienaGOF(ans3, GeodesicDistribution, verbose=TRUE, join=TRUE,
varName="happiness",cumulative=FALSE)

Multipar.RSiena(ans3,50,25,26,27,7,32)

Multipar.RSiena(ans3,4)


timet<-sienaTimeTest(ans3)
summary(timet)
plot(timet,effect=c(21:40))

###### try with endowment function

data_n <- sienaDataCreate(friend,enemy,sex,age,class,prev,
happiness,happy1_n,happy2_n,happy3_n,dissonant1_n,dissonant2_n,dissonant3_n)
eff <- getEffects(data_n)
eff <- includeEffects(eff,cycle3,
transRecTrip ,transTies,between,inPop ,outPop,outAct,name="friend")
eff <- includeEffects(eff,inPop,inPopSqrt,between,outAct,name="enemy")


eff <- includeEffects(eff,transTrip,between,outAct,inAct,inActSqrt,
inPop,name="friend",type="endow")

eff <- includeEffects(eff,transTrip,outAct,between,inAct,inActSqrt,
inPop,name="enemy",type="endow")

#eff <- includeEffects(eff, sameX, interaction1 = "sex")
#eff <- includeEffects( eff, egoX, interaction1="sex")
eff <- includeEffects( eff, altX, interaction1="sex")
#eff <- includeEffects(eff, sameXRecip, interaction1 = "sex")
eff <- includeEffects(eff, sameXTransTrip, interaction1 = "sex")

#eff <- includeEffects(eff, simX, interaction1 = "age")
#eff <- includeEffects( eff, egoX, interaction1="age")
#eff <- includeEffects( eff, altX, interaction1="age")
eff <- includeEffects(eff, sameX, interaction1 = "class")
#eff <- includeEffects( eff, egoX, interaction1="class")
eff <- includeEffects( eff, altX, interaction1="class")
eff <- includeEffects(eff, sameXRecip, interaction1 = "class")
#eff <- includeEffects(eff, sameXTransTrip, interaction1 = "class")

eff <- includeEffects(eff, sameX, interaction1 = "prev")
#eff <- includeEffects(eff, sameXRecip, interaction1 = "prev")
#eff <- includeEffects(eff, sameXTransTrip, interaction1 = "prev")
#eff <- includeEffects( eff, altX, interaction1="age")
```

```
 eff <- includeEffects(eff, sameX,name="enemy", interaction1 = "sex")
 eff <- includeEffects( eff, egoX,name="enemy", interaction1="sex")
#eff <- includeEffects( eff, altX, name="enemy",interaction1="sex")
 eff <- includeEffects(eff, sameXRecip,name="enemy", interaction1 = "sex")
#eff <- includeEffects(eff, sameXTransTrip, name="enemy",interaction1 = "sex")

#eff <- includeEffects(eff, simX, name="enemy",interaction1 = "age")
#eff <- includeEffects( eff, egoX,name="enemy", interaction1="age")
#eff <- includeEffects( eff, altX, name="enemy",interaction1="age")
 eff <- includeEffects(eff, sameX,name="enemy", interaction1 = "class")
#eff <- includeEffects( eff, egoX, name="enemy",interaction1="class")
#eff <- includeEffects( eff, altX,name="enemy", interaction1="class")
 eff <- includeEffects(eff, sameXRecip, name="enemy",interaction1 = "class")
#eff <- includeEffects(eff, sameXTransTrip,name="enemy", interaction1 = "class")

 eff <- includeEffects(eff, sameX, name="enemy",interaction1 = "prev")
 eff <- includeEffects(eff, sameXRecip, name="enemy",interaction1 = "prev")
#eff <- includeEffects(eff, sameXTransTrip,name="enemy", interaction1 = "prev")
 eff <- includeEffects( eff, altX, name="enemy",interaction1="prev")

#happy1
 eff <- includeEffects(eff,transTrip,name="friend") #sig
#happy2
 eff <- includeEffects(eff,closure,name="friend",
interaction1="enemy") # non sig by now
#happy3
 eff <- includeEffects(eff,cl.XWX, name ="enemy",interaction1="friend") #sig
#diss1
 eff <- includeEffects(eff,cl.XWX, name ="friend",interaction1="enemy") # molto sig by now
#diss2
 eff <- includeEffects(eff,to, name ="friend",interaction1="enemy") # non sig
#diss3
 eff <- includeEffects(eff,transTrip,name="enemy") # sig

 eff <- includeEffects(eff,crprod,name="friend",interaction1="enemy")

#eff <- includeEffects(eff,crprod,name="enemy",interaction1="friend")

#eff <- includeEffects(eff,crprodRecip,name="friend",interaction1="enemy")

#eff <- includeEffects(eff,crprodRecip,name="enemy",interaction1="friend")


 eff <-includeInteraction(eff, crprod,sameX,name="enemy",
interaction1=c("friend","class"))

#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","class"))

#eff <-includeInteraction(eff, crprod,sameX,name="enemy",
#interaction1=c("friend","prev"))
#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","prev"))

#eff <-includeInteraction(eff, crprod,sameX,name="enemy",
#interaction1=c("friend","sex"))
#eff <-includeInteraction(eff, crprodRecip,sameX,name="friend",
#interaction1=c("enemy","sex"))

#eff <-includeInteraction(eff, crprod,simX,name="enemy",
#interaction1=c("friend","age"))
#eff <-includeInteraction(eff, crprodRecip,simX,name="friend",
#interaction1=c("enemy","age"))

#eff <- includeEffects(eff, simX, name="friend",interaction1="happiness")
#eff <- includeEffects(eff, egoX,name="friend", interaction1="happiness")
#eff <- includeEffects(eff, altX, name="friend",interaction1="happiness")


#eff <- includeEffects(eff, simX, name="enemy",interaction1="happiness")
 eff <- includeEffects(eff, egoX, name="enemy",interaction1="happiness")
#eff <- includeEffects(eff, altX, name="enemy",interaction1="happiness")


#happy1
#eff <-includeInteraction(eff, cycle3,egoX,name="friend",
#interaction1=c("","happiness"))
#happy2
#eff <-includeInteraction(eff, closure,egoX,name="friend",
#interaction1=c("enemy","happiness"))
#happy3
#eff <-includeInteraction(eff, cl.XWX,egoX,name="enemy",
#interaction1=c("friend","happiness"))

#diss1
#eff <-includeInteraction(eff, cl.XWX,egoX,name="friend",
#interaction1=c("enemy","happiness"))
#diss2
#eff <-includeInteraction(eff, to, egoX,name="friend",
#interaction1=c("enemy","happiness"))

#diss3
#eff <-includeInteraction(eff,cycle3,egoX,name="enemy",
#interaction1=c("","happiness"))



### effects for
 eff <- includeEffects(eff, indeg, name="happiness", interaction1="friend")
 eff <- includeEffects(eff, outdeg, name="happiness", interaction1="friend")
 eff <- includeEffects(eff, effFrom, name="happiness", interaction1="age")
#eff <- includeEffects(eff, effFrom, name="happiness", interaction1="sex")

#eff <- includeEffects(eff, isolate, name="happiness", interaction1="friend")
#eff <- includeEffects(eff, behDenseTriads,
# name="happiness", interaction1="freind")

 eff <- includeEffects(eff, indeg, name="happiness", interaction1="enemy")
 eff <- includeEffects(eff, outdeg, name="happiness", interaction1="enemy")

#eff <- includeEffects(eff, isolate, name="happiness", interaction1="enemy")
#eff <- includeEffects(eff, behDenseTriads,
# name="happiness", interaction1="enemy")

##### covariate per balanced and imbalanced on behaviour
 eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy1_n")
 eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy2_n")
 eff <-  includeEffects(eff, effFrom, name="happiness", interaction1="happy3_n")
 eff <-  includeEffects(eff, effFrom, name="happiness",
  interaction1="dissonant1_n")
 eff <-  includeEffects(eff, effFrom, name="happiness",
 interaction1="dissonant2_n")
 eff <-  includeEffects(eff, effFrom, name="happiness",
 interaction1="dissonant3_n")

 multiAlg <- sienaAlgorithmCreate(projname = 'two_networks_try3.R', seed=123 )
 ans6 <- siena07(multiAlg, data = data_n, effects =
 eff,useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE)

 save("ans6",file="ans6.rda")
 ans6 <- siena07(multiAlg, data = data_n, effects =
 eff,useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE,prevAns= ans5)

 save("ans6",file="ans6.rda")
 ans5 <- siena07(multiAlg, data = data_n, effects = eff,
 useCluster=TRUE,initC=TRUE,nbrNodes=6,returnDeps=TRUE,prevAns= ans5)

 save("ans6",file="ans6.rda")
### test
 cbind(which(abs(ans6$theta/(ans6$se))>1),
 (ans6$theta /(ans6$se))[abs(ans6$theta /(ans6$se))>1])

 gofi_6 <- sienaGOF(ans6, IndegreeDistribution, verbose=TRUE,
 join=TRUE,varName="friend",cumulative=FALSE)

 gofo_6 <- sienaGOF(ans6, OutdegreeDistribution, verbose=TRUE,
 join=TRUE,varName="friend", cumulative=FALSE)

 goft_6 <- sienaGOF(ans6, TriadCensus, verbose=TRUE, join=TRUE,
 varName="friend")

 gofg_6 <- sienaGOF(ans6, GeodesicDistribution, verbose=TRUE, join=TRUE,
 varName="friend",cumulative=FALSE)

 plot(gofi_6)
 plot(gofo_6)
 plot(goft_6)
 plot(gofg_6)

 gofi_e4 <- sienaGOF(ans5, IndegreeDistribution, verbose=TRUE,
 join=TRUE,varName="enemy",cumulative=FALSE)

 gofo_e4 <- sienaGOF(ans5, OutdegreeDistribution, verbose=TRUE, join=TRUE,
 varName="enemy", cumulative=FALSE)
```

```
goft_e4 <- sienaGOF(ans5, TriadCensus, verbose=TRUE, join=TRUE,
varName="enemy")

gofg_e4 <- sienaGOF(ans5, GeodesicDistribution, verbose=TRUE, join=TRUE,
varName="enemy",cumulative=FALSE)

plot(gofi_e4)
plot(gofo_e4)
plot(goft_e4)
plot(gofg_e4)

Multipar.RSiena(ans5,15,16)
```

References

[1] Snijders Tom, Koskinen Johan and Schweinberger Michael Maximum Likelihood Estimation for Social Network Dynamics The Annals of Applied Statistics (pp 567?588) 2010, Vol. 4, No. 2 doi: 10.1214/09-AOAS313

[2] Rene' Veenstra, Jan Kornelis, Christian Steglich and Maarten H. W. Van Zalk Network-Behaviour Dynamics, Journal Of Research in Adolescence, 23(3) 399-412

[3] Christian Steglich, Nynke Niezink Testing Structural Balance theory, Cluster Statistics and Social Network

[4] Azzalini Adelchi, Scarpa Bruno (2008) Prediction of Quantitative Variables in *Data Analysis and Data Mining, An Introduction*, 2nd edn, Oxford University Press, New York.

[5] Baltagi, Badi H. Econometrics analysis of panel data Wiley editors, 3rd edition

[6] Snijders, Tom. Longitudinal Methods of Network Analysis Encyclopedia of Complexity and System Science, Social Networks section, Springer Verlag, 2009;

[7] Snijders, Tom. Statistical Methods for Network Dynamics Statistical models for social networks, network-based social processes and complex social systems

[8] Snijders, Tom. Models for Longitudinal Network Data Models and methods in social network analysis, Chapter 11 (pp. 215 - 247) in P. Carrington, J. Scott, and S. Wasserman (Eds.),2005;

[9] R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[10] Ripley Ruth, Boitmanis Krists and Snijders Tom RSiena: Siena Simulation Investigation for Empirical Network Analysis R package version 1.1-282. http://www.stats.ox.ac.uk/ snijders/siena

[11] Snijders Tom, et al., A model for the multiplex dynamics of two-mode and one-mode networks, with an application to employment preference, friendship, and advice. Soc. Netw. (2012); http://dx.doi.org/10.1016/j.socnet.2012.05.005

[12] Snijders Tom, et al., Introduction to stochastic actor-based models for network dynamics Social Networks. (2010); doi:10.1016/j.socnet.2009.02.004

[13] Rene' Veenstra, Christian Steglich Actor-Based Model for Network and Behaviour Dynamics Handbook of Developmental Research Methods (pp. 598-618), New York, Guilford

[14] Snijders Tom, Christian Steglich and Schweinberger Michael. Modeling the co-evolution of networks and behavior Longitudinal models in the behavioral and related sciences, (pp. 41-71), 2007.

[15] Maddala, G. S. Limited-Dependent and Qualitative Variables in Economics New York: Cambridge University Press, (pp. 257-91), 1983

[16] Snijders Tom, The Statistical Evaluation of Social Network Dynamics Sociological Methodology, (pp. 361-395), 31 (2001)

[17] Steglich Christian, Snijders Tom, and Pearson Michael Dynamic networks and behaviour, separating selection from influence Sociological Methodology, 40: (pp. 329?393) doi: 10.1111/j.1467-9531.2010.01225.x

[18] Ripley Ruth, Snijders Tom, Boda Zsofia, Andras Voros, Precaido Paulina Manual for RSiena, version: 30-12-2014

[19] F. Doddema 2014 Introductie tot het stochastisch actor-georirenteerd model