Mary Downham:

Introduction:

This is a research project investigating the factors that effect the propagation of ideas/beliefs through a population of agents using computational models. My aim is to move from the long-established mathematical model of social interaction to a more realistic agent-based model – an exploration into artificial culture. Artificial culture is the study of social behaviour of agents using the tools provided by artificial intelligence. The basis of my project stems from work carried out by Damian Zanette (1999) [1] who designed a mathematical deferential to describe the dynamics of idea propagation in a given population. I have created different computational models that apply this equation to a population of agents. Each model investigates different factors that could affect the propagation of ideas. I have investigated into different types of networks: 1D and 2D spatial networks, small world networks, both static and evolving, and the affect of overlaying two different networks on a single population. Additionally I have researched into other attributing individualistic factors such as varying the amount agents are influenced by across the population, positive node correlation and preferential attachment. I finally move towards integrating all the aforementioned factors in one final agent-based model. I aim to analyse the results and present a clear report on my findings.

Structure of this report.

The project will first discuss some important issues in modelling social systems. I shall then review the work done by Zanette. I will then discuss the theory behind the extensions to Zanette's model. This will include looking at spatial relationships within the population. I then review the work done by Duncan Watts[2] (1998) on small world networks and further extend Watt's work by incorporating evolving networks. I shall then look at overlaying network topologies and lastly the role the agent, by looking at the agent's parameters and other attributes in agent behaviour.

Following on from the theoretical background I shall explain how I have implemented the various computational models using java, and the various extensions needed on each model. Leading to a detailed account of my results. Conclusions and a final discussion of my project shall follow. Finally ending with an evaluation of the project as a whole and a discussion of possible future directions.

Modelling Social Systems:

There have been many attempts to model world, and to predict social forecasts. Be it by the stars, by Mathematical models or to understand it firsthand through social anthropology. As yet no model has correctly modelled the world. Within the new domain Artificial Culture, closely linked to Artificial Life, a new way of trying to understand life has developed. The new tool is agent-based modelling which is aided by the use of dynamical theory. The aim – to model life from the ground up.

---

[1] Zanette, D, H, (1999), *Globally coupled bistable elements as a model of group decision making,* C onsejo Nacional de Investigaciones Cientificas y Tecincas, 8400 Bariloche, Rio Negro, Argentina.

[2] Watts, D, J (1998), *Collective dynamics of 'small-world' networks,* Nature Vol 393, 4 June

Why model society?
Many would argue that it's a pointless endeavour to model social systems in such a formal and reductive fashion. Arguing it serves no purpose or suitable answer to the intellectual pursuit of explaining social systems. That since there is such a vast reality gap there can be no successful transfer from simulation to reality. However, the purpose of artificial simulation is not to model everything. It would be a near impossible task to do so. It is to provide cheap, simple and fast ways of trying to describe social systems. Generally, simulations can be used both *'deductively'* and *'heuristically'* providing a means of testing already formed theories or guiding theory construction (Brier, Robinson, 1974)[3].Through modelling social systems from the bottom-up one can '*begin the development of a more unified social science… Artificial society-type models may change the way we think about explanation in the social sciences'* (Epstein, Axtell, 1996)[4]. One must ask what counts as an explanation of social systems, how do we explain life? Maybe through computation we can build what would constitute, look like or behave as a social system. If we can achieve this we can analyse the interactions, the behaviour and the system more accurately since we have designed it. Modelling allows for a generative kind of social science.

What type of model?
Before recent endeavours in the realm of artificial culture and artificial life most models used to model society have traditionally been mathematical models. In many mathematical models the agents within the population have been homogeneous (Axtell, Epstein, 1996). They are all identical. However, a more realistic model would be one that accounts for distinct agents. This is the agent-based model approach. The methodology behind this approach is that agents should have their own individual makeup, internal state and behaviour governing rules. Additionally in traditional differential mathematical models there has been no account varying network topologies, of course the network of an agent will be of significance. As already mentioned the environment plays an important factor in the shaping of an individual.. A more realistic model should take into account an agent's spatial neighbourhood, and their social network. Dynamics systems theory also suggest that the causation will be circular not only will the environment affect the agent but the agent will affect the environment Many agents natural or real directly construct their environment. Additionally in circumstances where the agent cannot directly manipulate their environment their actions and indirect interactions can cause the environment to change – indirectly (Bedau, 1996)[5]. Bearing this in mind equilibriums within a population, as described, will be an emergent structure given by the collective dynamics. It seems then that an agent-based model - such as those used in Artificial Life – is ideal. Mathematical models '*reveal little about the effects of the complex web of interactions binding organisms'* (Bedau,1996). My aim therefore is to move from Zanette's mathematical model towards an agent-based model.

Why is dynamics important?
A dynamical system is a system that changes over time. One uses differential or difference equations to help explain and model a dynamical system. Seeing the states

[3] Brier,A & Robinson, I, 1974, *Computers and the Social Sciences,* Hutchinson and Co Ltd, London.
[4] J, M Epstein and R Axtell, 1996, *Growing Artificial Societies,* MIT Press
[5] Bedau, M, A, 1996, *The extent to which Organisms construct their Environments,* Adaptive Behaviour 4.

of a natural system as a set of parameters one can numerically describe them in a mathematical equation. A change in one parameter will effect how the whole state of the system changes over time. The parameters are said to '*fix the dynamics of the system'* (Van Gelder, 1997)[6]. Dynamical systems also use a notion of a coupling relationship; this is when multiple subsystems have a continuous circular causal influence on each other. That is to say that some *'factor x is continuously affecting and being affected by some other factor y'* (Clark, 2001)[7]. Many if not all systems in the world are a dynamical system since they all change over time.

When studying artificial culture it is important to see how a population evolves over time -*'this involves specifying how the actions and attributes of agents unfold given the topological properties of the network and vice versa'* (Briger, 2003)[8]. We can then view the population, coupled with the environment, as a dynamical system. Indeed dynamics is becoming a key role within Artificial Life and cognitive science together with the ideas of embodiment and situatedness. Agents are seen as being coupled with their environment. In cognitive science it is now proposed that we can understand the mind by taking on the approach laid down by dynamical systems theory. The mind and world are seen as having a coupled relationship – they have a continuous circular causal influence on each other. Simply the body and the world (time, movement etc…) all matter and all play roles in the cognitive system. '*Neural, bodily and environmental elements are intimately intermingled'* (Clark, 2001). It is therefore important to use the tools from dynamical systems theory to aid investigation into social behaviour.

This way of thinking and describing life is not as new as it seems. Indeed back in the early 1970's Jay. W. Forrester founded the System Dynamics Group at MIT. The ideology grew from the disciplines of feedback control and systems engineering (Bloomfield, 1986)[9]. Their approach was to build computer simulation models, which described the behaviour of social systems using the programming language DYNAMO. This resulted in models such as WORLD 2 and WORLD 3 that modelled the *'earth as a closed global ecosystem'* (Bloomfield) and saw people as living in networks that exhibit feedback structures. Since being used from a social systems perspective, system dynamics has been applied to nearly every type of feedback system, technical and social.

Forrester was also knowledgeable in decision-making processes. From his perspective decisions were *'not entirely 'free-will' but are strongly conditioned by the environment'* (Forrester, 1961)[10]. Thus suggesting it was possible to work out the policies that control the process of a decision. It therefore seems appropriate to use a dynamical model as the basis of my research. As mentioned before I shall be using the simple mathematical model designed by Zanette that describes social decision and belief propagation in a dynamical way.

---

6 Van Gelder, T, (1997), *Dynamics and Cognition,* in 'Mind Design 2', ed, John Haugeland, MIT press
7 Clark, A, (2001), *Mindware,* OUP
8 Brieger, R, ed, (2003), *Dynamic social network modelling and analysis,* The National Academies Press.
9 Bloomfield, B, P, 1986, *Modelling the world,* Oxford Publishing Services.
10 Forrester, J,W,1961,*Idustrial Dynamics,* Cambridge Mass, MIT Press.

The basic model:
Zanette's Differential Equation:
The foundation of my project is Zanette's Differential Equation for describing trends in a population. Zanette's model was applied to a voting system – where a population of agents have to decide between two candidates. Using his mathematical model, Zanette aimed to explore how the effects of '*the personal trend and of the average opinion in defining the agent vote combine with each other to lead the group to its collective decision'* (Zanette, 1999). In Zanette's model each agent is exposed to the average opinion at each time step, this dynamically effects how his or her opinion evolves through time.

The average opinion is the arithmetic mean value of the sum of the belief values of the agents in the population at a given time, *t.* The mean average is worked out by the sum of the agent's belief values, *Xi,* divided by the number of agents in the population *N.*

$$\bar{X}(t) = 1/N \sum X i (t)$$

The effect of the average opinion on the development of the agents belief or decision making, is described as:

$$Xi = Xi + (Xi - Xi^3 + ki (\bar{X} - Xi))\, dt$$

The equation states that in a population of agents one can work out how strong a persons belief is at the next time step given his current belief *x,* the average belief value of the population *avg (x bar),* and the influence the average belief has on the agent *k – the coupling constant.* The agents preference at voting is measured as a number between 1 and –1, between two different states or as Zanette demonstrated between two different candidates in a voting system. To have a belief value of 1 is 100% positive belief – yes I agree. To have a belief value of –1 is a 100% negative belief – no I don't agree. To have a belief value of 0 is to say you neither agree nor disagree you are agnostic. Any other belief value is an intermediate belief.

Zanette argues that for *k=1,*the belief of all the population converges to one of the two extremes. For this coupling constant the average opinion is dominant. For *k*=0, Zanette argues that the population converges to both extremes – according to their initial belief value. In the absence of influence the agent reinforces their initial position/belief. Between *k* =1 and *k*=0, a transition is expected between the two behaviours. This model can be simulated computationally by creating a population of agents with a random generated belief value. By running this program one can see how the populations evolve according to the coupling constant.

I have created and ran a program in java to simulate Zanette's differential. The results are in accordance to his predictions.

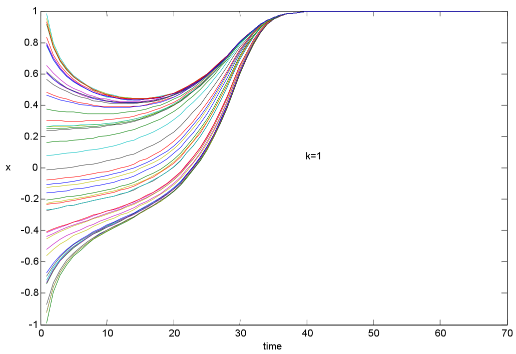


Figure 1: k=1

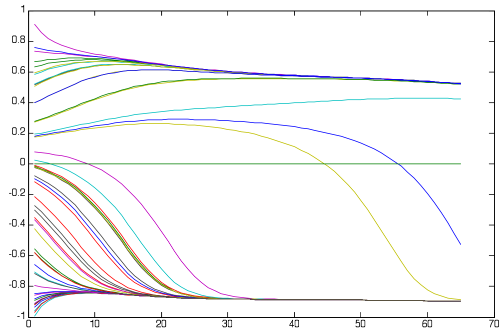


Figure 3: k=0.4

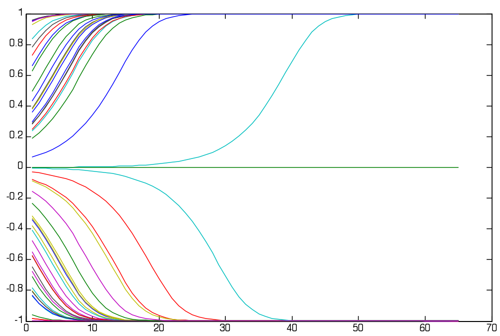


Figure 2: k=0

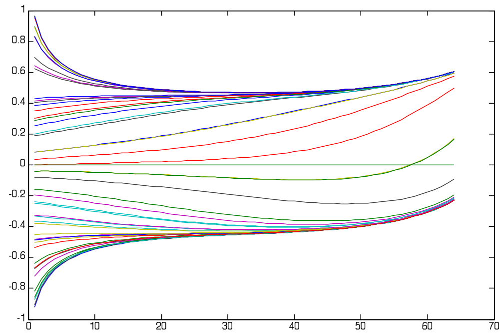


Figure 4: k=0.8

Theoretical basis to the extensions of Zanette's Model:

Here I aim to outline the theoretical basis to the extensions I have created of Zanette's model. All integrate Zanette's differential equation however different factors that could affect the propagation of ideas and make the model more agent-based are explored. Firstly I shall look at how different types of networks can make the model more realistic and then look at properties that affect the agent itself.

What type of Network?

As already stated an agent's environment is of great salience. In Zanette's model there is little account of environment. For research into idea propagation the environment I am concerned with is the social environment of the agent. It is commonly thought that an agents social environment can be successfully viewed as a network. Indeed the social network paradigm '*operationalizes the notion of social structure by representing it in terms of social relations tieing distinct social entities to one another'* (Leinhardt, 1977) [11] and is greatly used throughout the social sciences. I aim to project a variety of network topologies onto Zanette's model with two aim's in

---

[11] Leinhardt, S, 1977, *Social Networks: A developing paradigm,* Academic Press, New York.

mind a) to ascertain what factors effect the propagation of ideas in a population and b) to move towards a more agent-based model of idea propagation.

Spatial Networks:
Firstly I shall be looking at spatial neighbourhood networks. Presently there is no spatial element at work in Zanette's model. The belief value of every agent in the population affects the belief value of every other agent. A global average is taken. However, this is unnatural and unrealistic – spatial concepts such as location, our neighbourhood, distance and immediacy play a large part in our human lives. At the end of the day the land we inhabit is our '*ultimate substrate'* (Nobel, Davy, Franks, 2004)[12] and we are more likely to have ties to agents in our proximity. We are both influenced and influence our neighbours (un-authored, 2004)[13]. We are more likely to be influenced by a local average rather than a global average. In Wellman's Study on community in East York 1968[14] he found that our neighbours were predominate routine ties. This fact should be represented in an agent-based model, to steer away from the homogeneous approach. I have devised two spatial models extending from Zanette's basic model, which take into account an agents relationships within its neighbourhood. In each model an agent can only be affected by the belief values of the agents within its neighbourhood. The first one is a one-dimensional spatial network. An agents neighbourhood consists of those agents either side of him within the dimension stated by the radius, r. For example see fig # below, here the radius is two; so any agent (represented by a node) will have ties or be connected to two neighbouring agents either side of him.

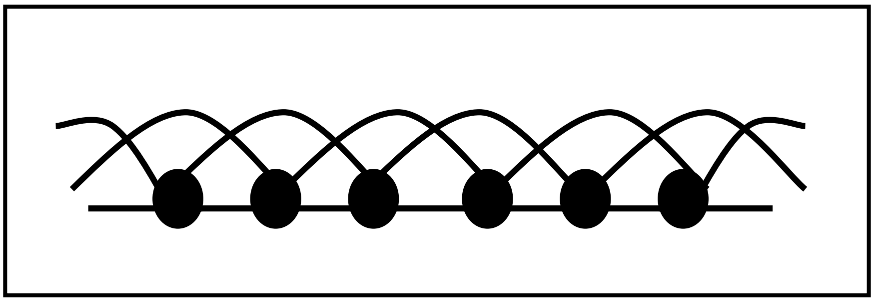


Figure 5: A one-dimensional network, with radius two.

The second spatial extension on Zanette's model is to incorporate two-dimensional spatial networks. These can be perceived as a two-dimensional grid lattice such as those used in cellular automata. It is unrealistic to presume that social networks are one dimensional, in fact they exhibit a highly complex topology that has vast and multiple connections and displays many dimensions. In the two-dimensional grid model a neighbourhood now consists of neighbours on all four immediate sides of the agent. There are several ways one could tackle this extension, for example, an agent could be connected to neighbours at their corners or just those on their direct sides.

[12] Nobel, J, Davy, S & Franks, D, 2004, *Effects of the topology of social network on information transmission,* Univeristy of Leeds yet to be published.
[13] Un-authored website, 2004, *Spatial Networks,* Http://cas.uah.edu/wilhitea/Spatial%20Networks%20Outline.htm.
[14] Wellman, B & Carrington, P,J, 1988, *Networks as personal communities,* in *Social Structures* etd Wellman, B, Cambridge Univeristy Press.

For this model I have looked at neighbours only at their direct sides. Again a radius is set at run time and corresponds to how many neighbours will be included in the neighbourhood. Shown in the diagram below is an agent with a neighbourhood of radius two.

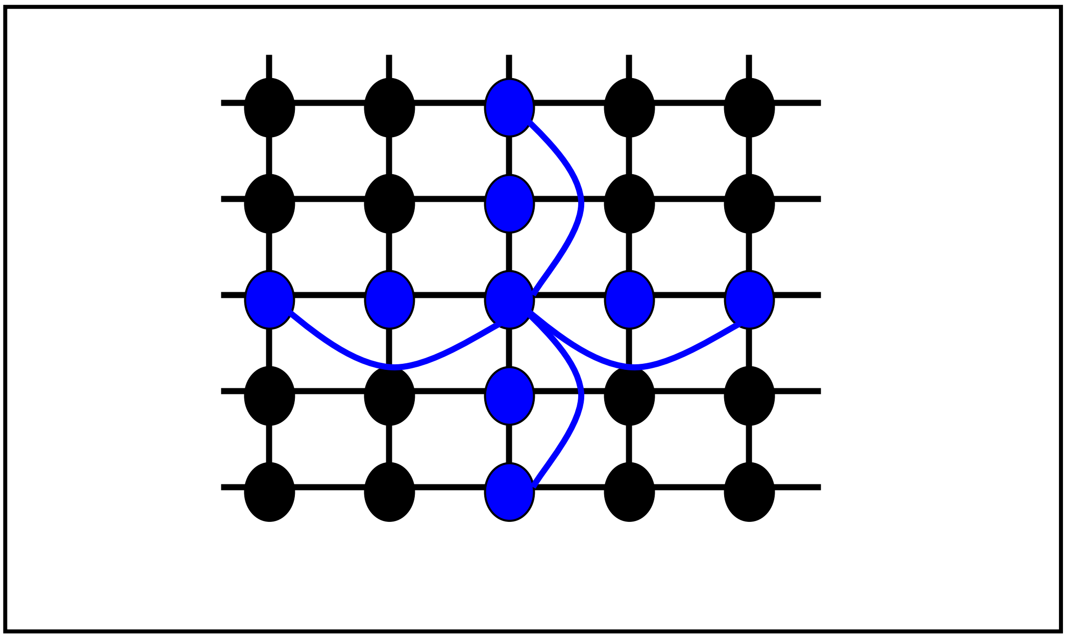


Figure 6: A two-dimensional network, with radius two

Small-World Networks.

The next part of my investigation looks at small-world networks. Recently there has been a burst of growth in the study and research of small-world networks. Small-world networks are now shown to prevail in many networks – neurons, disease, and the Internet (Watts, Strogatz, 1998). Suggesting that they are more common than we first imagined. Although most of the recent enthusiasm has been spurred on by Watts and Strogatz's work on small-world networks, Milgram first presented the small-world phenomenon in 1967[15]. Milgram studied the phenomena by giving a document to a group of people and asking them to pass on the document to someone they knew on first name basis. The aim was to get the document to a target person – a Boston stockbroker. The number of people required for the task to be accomplished was recorded. The results showed that the number of links required to complete the test was 5.2 (mean) – now known as the six-degrees of separation. This suggests that there is, in general, six degrees of separation between two randomly chosen individuals in a population. This is the basis of the small-world phenomena which suggest that '*social networks are in some sense tightly woven, full of unexpected strands linking individuals seemingly far removed from one another'* (Milgram, Travers, 1969)[16].

In my initial extensions, of Zanette's model, the focus was on uniform networks. Uniform networks since agents are in contact with and only with its neighbours, those in his spatial configuration. This uniformed network topology is unrealistic. In the real

[15] Milgram, S. 1967, *The small world problem,* Psycho. Today 2.

[16] Milgram, S & Travers, J. 1969, *An experimental Study of the small word problem,* Sociometry 3. In *Social Networks: A Developing Pardigm,* Edt Leinhardt, S. 1977, Academic Press, New York.

world we know people and come into contact with people from outside our local neighbourhood. As Milgram suggest there are only around six degrees of separation between a random person and us. This fact cannot be present in a network consisting of long-path lengths. Additionally we are not surprised if our friends know each other. These experiences in everyday life suggest that social networks exhibit short path lengths and high transitivity. Indeed social networks are proving to be best represented as small-world networks. In his article ‘Collective dynamics of ‘small-world’ networks’ (1998) Duncan Watts explains that real world networks are not completely uniform nor completely random, but somewhere in-between. In fact a small-world network is a uniformly connected network with a few connections randomly rewired. This means that the clustered structure of the network is kept allowing for transitivity but random short cuts are introduced allowing for short path lengths.

To create a small-world network one takes a uniform network such as those mentioned above and rewires some of the connections. Rewiring happens based on some probability *p*. For probability $p=0$ no rewiring takes place – for probability $p=1$ complete rewiring takes place. Experiments have shown that for a small-world network topology to be achieved the probability of rewiring is around 0.01 – 0.03.

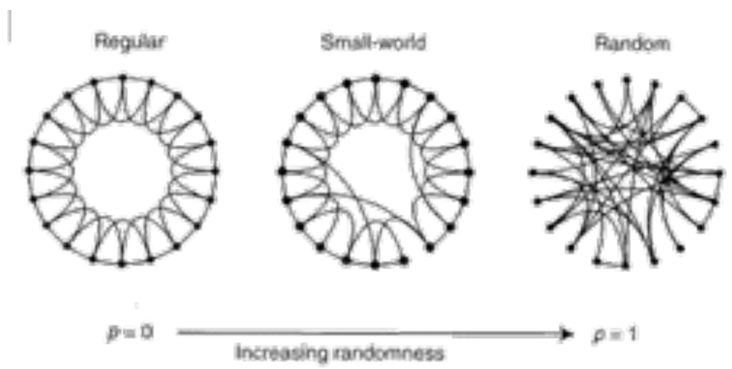


Figure 7: From a uniform - random network, via a small world network.

The growth of small-world networks suggests that any project on social systems should investigate this topology. I thus aim to project the small-world network onto Zanette’s model, and investigate the affects it has on the propagation of ideas. In this model an agents neighbourhood is now made up of local and non-local individuals.

It is important to realise that the agents affect the network topology as well as the topology affecting the agent. Since they are in a dynamic coupled relationship. Therefore I have devised a set of models that takes this into account, where the agents in the population affect the network structure. Here the network evolves over time. This is a result of the agents swapping who is in their neighbourhood. For these models have look at agents choosing new neighbours arbitrary, preferring to keep like-minded neighbours or by selecting new neighbours based on their belief values. I have also looked at incidents where agents may prefer original ideas.

Overlaying Topologies.
The next direction was to look at overlaying topologies most work carried out has only look at single topologies. In reality individuals belong too many different social networks be it spatial, friendship, family or work. However every network influences our inner state. This is when one object/node/agent belongs to various networks. My next model is going to investigate what happens when two networks are present. The networks I will be looking at are spatial and friendship networks – given my small-world networks.

Spatial networks relate to how we are connected in our vicinity. By this I mean our local network, neighbourhood. For this network I will be using the one-dimensional network. Where agents within the population, are influenced by the neighbourhood they belong to.

The second network is a friendship network. The small-world network will represent this. More often than not our friends are from outside of our local neighbourhood, maybe due to academic or leisure related activities. Friendship groups will also influence agents within the population. Additionally an agents friendship group may change according to their selective preferences.

The diagram below represents the overlaying topologies. As one can see there are both influences from an agents neighbours and its friends (seen in red). In the model both networks will have the same radius, this will mean if the radius is two the agent will have four neighbours and four friends. Additionally the constant *k* will be the same for both topologies. The probability of rewiring in the friendship network will be set at run time. Within this model I will investigate different ratios of influence each network has on the agent. Seeing what happens when say the network representing the friendship network has 80% of the influence on the agent and the neighbourhood has 20%.

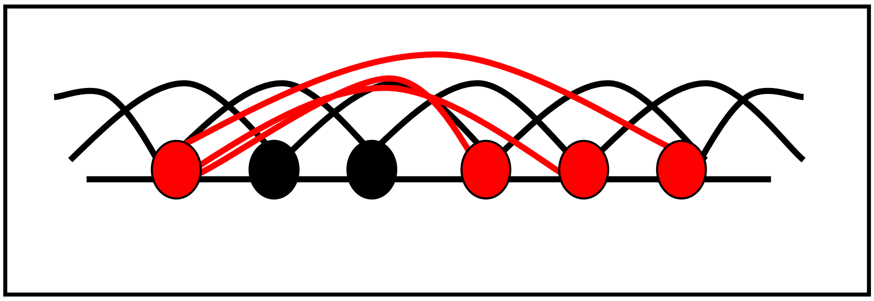


Figure 8: Representation of overlaid network topologies.

So then how can the agent influence their environment?
As already stated the agents and environment affect each other dynamically. For a realistic account of the propagation of ideas I have moved away from Zanette's basic mathematical model by taking into account different more realistic networks – spatial, small-world and overlaying topologies. How then might we make the agents more realistic? And how might this affect the propagation of ideas? Already I have mentioned that an agent might influence how the social networks evolve – by

preferring or selecting people in their environment. But do we only chose friends based on similarity?

Positive Node Degree Correlation.
Newman and Park [17](2003) argue that social networks unlike other networks show a positive node degree. That is to say that an agent is more likely to know another agent if they have the same number of friends/neighbours. A loner is more likely to know other loners, whereas a socialite is more likely to know other equal socialites. They state that social networks display *'assortative'* (Newman, Park, 2003) behaviour due to division of society into communities and groups. I have therefore implemented a model to take into account friend selection due to positive node degree correlation. Where an agent picks another agent as a tie if they have a similar amount of friends. I have used a gaussian distribution to denote how many friends an agent will have.

Preferential Attachment.
Additionally, Barabasi and Albert (1999)[18] argue for preferential attachment. They believe that many computer simulations of real networks don't take into account continuous growth and preferential connectivity. Preferential attachment states that '*the probability of with which a new vertex connects to the existing vertices is not uniform; there is a higher probability that it will be linked to a vertex that already has a large number of connections'* (Barabasi, Albert, 1999). Barabasi and Albert devised a model that when adding nodes the nodes chose to tie with nodes with a great number of connections to other nodes. They found that networks that exhibit preferential attachment have similar characteristics of small-world networks – high transitivity and short path length. Although my project deals with a finite population, the ties within the population change. I therefore created another model that used the number of friends – given a gaussian distribution across the population- to act as a criteria for selection.

Varying Agents internal state:
I also created models that looked at how the agents internal state affected the dynamics of the population. For these models I simply look at varying the amount an agent is influenced across the population and the amount they took up new ideas, by value the rate of change in Zanette's differential equation.

Towards a agent-based model of social decision.
This is the last model I shall be looking at. One aim of the project was to move away from the more traditional mathematical approach to modelling towards the more realistic agent-based modelling approach. So far the computational models mentioned have extended Zanette's model in various ways making them more agent-based. In my last model I aim to weave all these extensions together in one last step towards an agent-based model.

In this model agents are very varied, not only in their belief value, but in their selection process of new friends, their number of friends and their internal states. The

[17] Newman, M,E,J. & Park, J. 2003, *Why Social networks are different from other types of networks.* Physical review E, 68.
[18] Barabasi, A & Albert, R, 1999, *Emergence of Scaling in Random Networks,* Science Vol 286

model is based on the overlaying topology, where there are two networks a friendship network and a neighbourhood network.

What type of ties?

In all the models agents are connected to other agents. These connections are known as, arcs, edges connections or links in network theory and generally termed ties in the social sciences. In many computational models of social dynamics these ties are bi-directional (Nobel, 2004 FILL IN HERE) suggesting that if one agent is linked to another they will both influence/affect each other. However in real life this is rarely the case. Ties with other agents are usually unidirectional: agent A may see agent B as a friend but it may not be reciprocated. Wellman (1988)[19] states that '*ties are usually asymmetrically reciprocal, differing in content and intensity'*. Throughout my project unidirectional ties are used.

Requirement analysis:

This is a scientific research project rather than a development of a software product. Therefore the main end user is myself. My need is to be able to create and analyse different computational models to investigate the different factors that affect the propagation of ideas through society.

This project requires the use of a programming language to test and investigate different theories and models. For this project I'm using Java, to create my various models and run different experiments on various parameters and variables. Java's object-oriented programming structures allow me to create objects that hold both methods and data. This allows me to create objects for both the agents of a population, and their environment. The agent class can hold information that can be acted on and manipulated in different ways as well as holding data about that agent. Variables in any particular experiment can be easily instantiated in Java. A class representing the environment can create a population of agents connected in various ways and provide macro and micro rules on how the agents interact and evolve through time. In general a population of agents is created within the 'environment' and the artificial social structure emerges in time.

Additionally my project needs software capable of performing statistical analysis and creating graphical outputs for the emergent structures. For this I will be using Matlab. Matlab has an exhaustive list of functions that are available to me for both performing statistical analysis and presenting my data to both those who may read it and myself.

For presentational purposes I am also using the program Pajek. Pajek is a program created to present and analysis large networks. It is helpful in dictating how a network evolves over time pictorially. It requires no programming language, only the correct format of data. For the figures used in the project I have displayed the networks using an energy – kamada-kawai – layout. This layout makes use of the kamada-kawai

[19] Wellman, B. 1988, *Structural analysis: from metaphor to substance,* In *Social Structures* Etd Wellman, B. CUP

algorithm that attempts to position nodes so that the (Euclidean) geometric distance between them is as close to the graph-theoretic distance as possible.

The main Java classes:

The Individual Class:
This class relates to the agents, the agents of the artificial population. The class holds the constructors for creating the agents. Each agent holds information that represents its internal state and behavioural rules. The class grows with complexity as the models allow for more complex agents to be created. A simple agent simply holds a variable (double) which represents their *belief value* where as a more complex agent may hold a variety of beliefs, values for how they influence or a vector representing their neighbours/friends. Additionally within this class are simple methods for extracting and returning the information about a particular agent.

The Environment Class:
This class holds the main method and the main methods used for creating an evolving population. Here I shall describe the basic methods used in each model based on Zanette's work. However in the preceding experiments more complex models/methods were introduced which I will be describing later.

Basic methods used in the Environment class:

makePopulation()
This method simply creates a vector of agents by calling the agent constructor, additionally it is here that a random number, using the method supplied by the Maths class, is assigned to the agent indicating their original belief value .

displayPopulation()
The population vector is iterated through. Each agent's belief value is retrieved and added to a string. The end string holds all the agents belief values at that present run.

updateAverage()
Within this method the population vector is iterated through. Each agent's belief value is added to a total. This is then dived by the number of agents to find the mean belief value for the population.

updatePopulation()
Here each agent is taken in turn, their belief value is retrieved and their new one is worked out using Zanette's equation and the mean average for the population.

evolve()
This method runs the sequence for updating the population for a given number of time-steps – by calling the appropriate methods. Within this method a buffered file writer is created to write the agents belief values to a file for each time-step of that experiment.

Main method ()
Here the make population method is called. The evolve method is then called for a certain amount of times depending on how many tests are needed.

Statistical Analysis.
Since this is a research project based on different experiments I need a way of fairly analysing the data from the results. I test the varying parameters of each model five times. This allows me to find a mean average for each parameter combination. While testing one parameter I keep the others at a constant. For example while testing the effect K (the amount of influence) against the probability of reconnection, I keep the relative neighbourhood size constant, and vice versa. For the majority of tests the population size is 100 agents unless stated otherwise. For each trial the population evolves over 500 time steps, unless stated.

There are a few standards/ constants present in the models. That is if k=0 then the same pattern will always emerge, agents will just reinforce their own belief value and thus end at the extreme of their initial position. Additionally the same result occurs when their neighbourhood or friends connected to the agent are 0. This is due to the fact that the average neighbourhood opinion is restricted to just their own view.

Matlab Functions for statistical Analysis.
I have created functions in Matlab to analyse the data from the tests. The same functions are used on every test. The main purpose of these functions is to see how many groups are formed in the population. I do this by seeing how separated agents belief values are from each other. A group is defined when there is difference of 0.05 in adjacent belief values. For example if the population was of 6 people who had the belief values of 0.04, 0.046, 0.009, -1,-0.95 and –0.5 three groups would be formed. However in some cases there are very clear cut and defined groups where in others the groups are ill defined. The functions used are as follows.

Firstly a function called **groupsConverge()** takes the results from a single test. It iterates through each time step – sorting the time steps into ascending order of belief values. The result is a matrix of 200 numbers between -1 and 1 representing the agents belief values at that time step. This matrix is then past to converge() and a number past back representing how many groups are present in that time step. This value is added to a matrix. At the end of the function a matrix has been created which represents how many groups are at each time step.

Converge() is a function which iterates through the matrix comparing adjacent numbers. When there is a difference between adjacent numbers that is greater than 0.05 a *count* variable is increased by one. At the end of the iteration the value of the count variable is past back to the groupsConverge() function.

Groups() is a function which takes the results from 5 runs of the test. This is done by using the built in Matlab function *dlmread*. Dlmread specifies how the files created by the java program should be read into Matlab. The data is then transformed into a matrix and sent to groupsConverge() which returns with a matrix representing the groups formed at each time step. This is done for each of the five runs. These matrixes are then summed and divided by 5 and the average amount of groups for each time step is calculated for the 5 runs.

So in conclusion to my requirements analysis I repeat that there are no end users to my project. I am the sole user. I require a programming language adequate for computationally simulating models. For this I have chosen Java. Additionally I require a package that will aid my analysis and create appropriate graphics to explain my results. For this I have chosen Matlab. I'm also using the computer program Pajek to create network diagrams. Lastly I'm building on work already done by Zanette and his mathematical equation for describing a population's decision behaviour dynamically.

Project development:
Having created a computational model in java to reflect Zanette's mathematical model of social decision. My aim is to expand and further Zanette's mathematical model to get a more agent-based model of the propagation of ideas and to additionally investigate what these developments may suggest about social systems and see what effect different parameters have on the propagation of ideas.

I shall now describe how I have implemented each computational model extension in turn and how I have modified the basic model to achieve this. Following this I shall discuss the results of each model and discuss the implications of the results in the context of the propagation of ideas through society and social decision.

The one-dimensional spatial model.
The one-dimensional model takes an agents spatial configuration into account. An agent is now affected by the local average belief value of their neighbourhood rather than the global average belief value. The neighbourhood is defined given the parameter r, radius. The agent's neighbourhood contains agents within the distance r on each side of him.

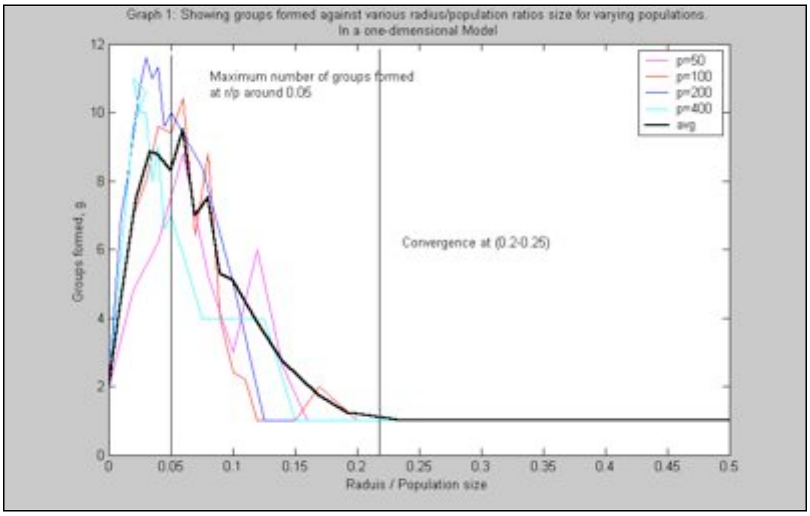
Modifications to the model.
In the environment class there is now a variable that controls how big the spatial neighbourhood is. This is controlled by *r*, the radius. If *r* equals two the agent will have four 'neighbours' that it is connected to – two either side. The sum of the agents belief values in its neighbourhood is calculated and the mean value found. This is now the local average for that agent. This average is then used instead of the global average for that particular agent when Zanette's equation is being applied at each time step.

$$Xi = Xi + (Xi - Xi^3 + ki\ (locAvg - Xi))\ dt$$

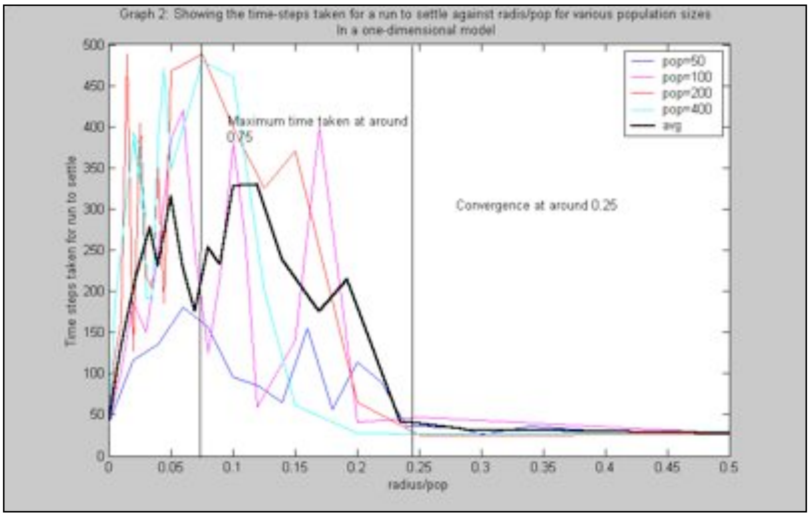
Implementation.
The main bulk of this computational model is the same as the basic model. However within the Individual class a new constructor is added. This constructor creates agents who hold data that represent both their belief value and their local average. Within the environment class there are few subtle changes. Firstly at the makePopulation() method the agents created use the new constructor method. The initial value for their local average is 0. At the start of the program an *int* is required to represent the radius for the agents neighbourhood. Additional to the *double* for the k value which remains constant throughout the run. The radius is the same for each agent in the population. At the updateAverage() method this radius is used to find the mean average belief value for the agents within a certain neighbourhood. This is done by iterating through the population and taking each agent in turn. Using the radius the belief values are then found of the agents neighbours within the radius' dimensions. For example if the average for agent at position 5 is being worked out and the radius is 2, the belief values for agents number 3,4, 6 and 7 are pulled out. The local average for that agent is then found by adding each of his neighbours belief values dived by the number of agents in the neighbourhood. A new agent is created using the original agents belief value and the new local average. The updated agent then replaces the old agent. When all the agents have been updated the new vector representing the population is returned. To make sure that the agents at each end of the vector are connected the '*%'* function is used. Additionally in the updatePopulation() method, instead of using the global average to compute the agents new belief their local average is used instead.

Analysis for the one-dimensional Model:
For this model I tested various parameters of the system. Firstly I tested the number of groups formed (defined above), and the time taken to converge to static group formations against different radius – population ratios. Four population sizes 50, 100, 200 and 400, were used and 19 radius sizes between 0 and half the population size. One must remember that for r = population size/2 the actual number of agents within the neighbourhood would be equivalent to the number of agents in the population.

I then tested different values of *k*: 1, .8, .6, .5, .4, and .2 for 8 different radius/population ratios for a population of 200 agents, again noting how many groups were formed and the time taken to converge to static groups. For each parameter combination five runs were averaged. The constants as mentioned above, such that for r=0 and .k=0 two distinct groups are formed, were present in all experiments. Below are graphs representing the results of the tests, *graph 1* shows how the number of groups formed changed depending on the radius/population ratio and *graph 2* shows how the number of groups formed changed depending on the *k* value used.



Graph 1: Showing groups formed against radius/population for various population sizes, for a one-dimensional model.



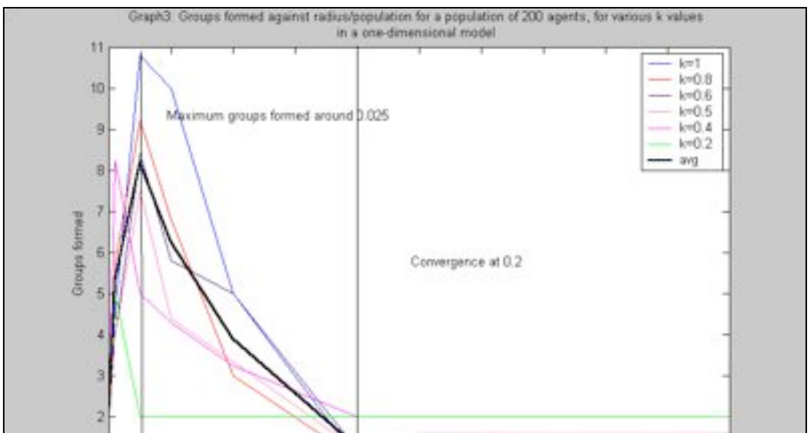
Graph 2: Showing Time taken for population to settle against radius/population for various population sizes. Results are for a one-dimensional model.

The above graph and the results indicate that maximum number of groups are formed when an agent's neighbourhood size is around one tenth of the population size. The number of groups formed does varying depending on how big the population is, with more groups being formed for bigger populations. Additionally the population acts as a whole and one group is formed when an agent's neighbourhood is around 40% of the actual population, this the same for all population sizes. From the graph one can see a rise and fall in the number of groups from two groups being formed when there is no neighbourhood to the maximum and down to one group being formed again. Why should neighbourhoods smaller than a tenth of the population result in less groups being formed than when a neighbourhood is around a tenth of the population and maximum groups are formed?

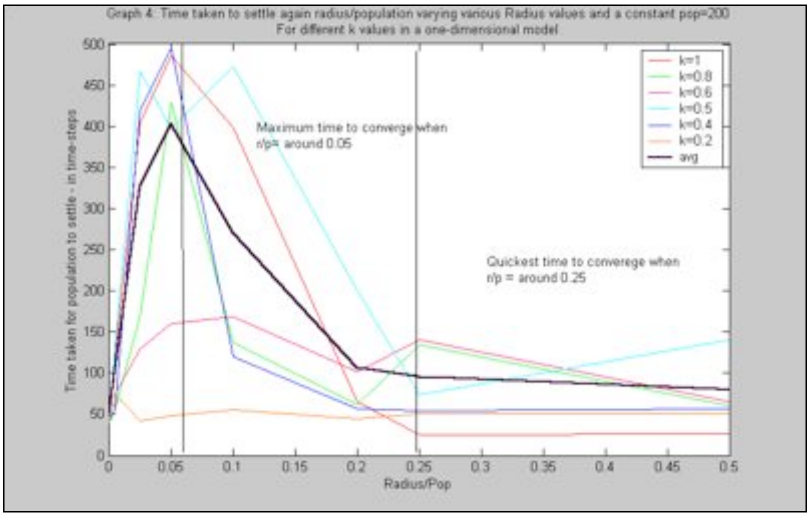
Although not so clear *graph 2,* displaying the amount of time needed for the population to settle into distinct group formations, also shows the same sort of behaviour as the previous results. Where the maximum time needed for the population to settle is when an agent's neighbourhood size is around a tenth of the population Neighbourhood sizes small than this take less time to settle and when an agent's neighbourhood is around or higher than 40% of the total population size the time taken to settle is the same low value.

There is a strong pattern present in all results. That the time needed to settle and the number of groups formed is at a maximum when the neighbourhood size is around a tenth of the population. Starting from when there is no neighbourhood until this point is reached there is a steep rise in number of groups formed and an time taken to settle. After this point is reached there is a distinct fall in number of groups formed and time taken to settle until the neighbourhood is around 40% of the population size. When this point is reached only one group is formed and the time taken to coverage is around 26 time steps. This suggests that there are two critical points. Point 1 is when the neighbourhood size is a tenth of the population and maximum groups are formed and the maximum time is taken. Point 2 is when the neighbourhood is around 40% of the population size and the population acts as a whole and takes minimum time to converge. So why should this be the case?

This result is usual for dynamic systems, but why this system? Before the first critical point small neighbourhoods act as though they are large neighbourhoods. This is due to a kind of chain reaction since the neighbourhoods are small it is as though everyone is influencing everyone else. If you only influence one person and they in turn only influence the next person, you are influencing no-one and yet everyone. With smaller neighbourhoods it is as though the whole population is joined so not many distinct groups can be made. Obviously when the neighbourhoods are so large such as 40% of the population everyone is influencing everyone else and so the population although divided still acts as one. It is only when neighbourhood sizes are not so large and yet not so small that actual group formation happens. So lets us now look at the results for different values of *k*.



Graph 3: Showing the number of groups formed, against different values of radius/population, for different k values with the population at a constant 200. Results for a one-dimensional model.



Graph 4: Showing the time taken for a population to settle in time-steps, for various r/p ratios for a population of 200 agents and different k values in a one-dimensional model.

The results again indicate a definite maximum of groups and time taken to settle when the neighbourhood is around a twentieth to a tenth of the population. The results are not as clear however, since *k* is still affecting how many groups are formed. For low values of *k* two groups are formed rather than one. However, there are still the two critical points: when the maximum number of groups are formed and secondly when the population is acting as a whole and either one or two groups are formed. Again varying *k* means that the results for the time taken to converge are slightly less clear cut. This is due to the intermediate values of *k.* One should remember that for k=0 two groups are formed since an agents only influence is themselves. So for intermediate values of k it is as though the agent is unsure whether to follow suit or to go their own way – the agent is of two minds - thus resulting in a longer time for group formation. However, when *k* is as low as 0.2 this 'indecisiveness' is not present since the influence of an agent's neighbours is trivial.

The two-dimensional spatial model.

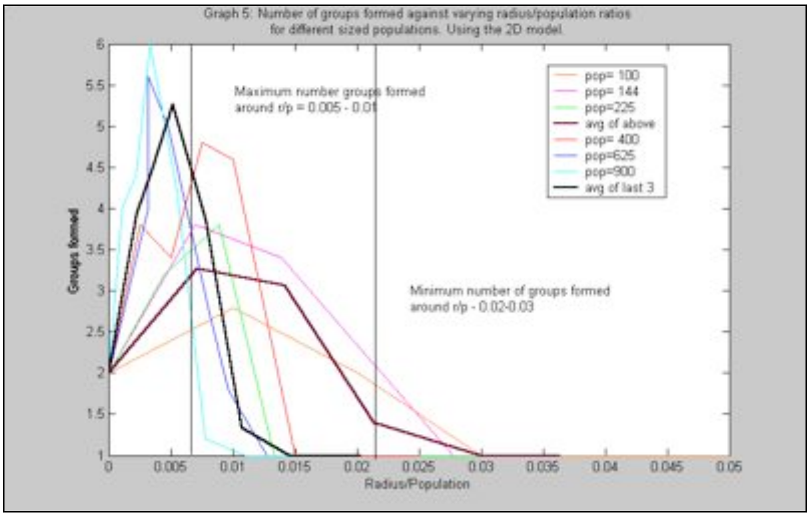
This is a slight variation on the one-dimensional model where the agents for a grid shaped lattice (see figure #). The agents neighbourhood now spans to neighbours on their immediate sides.

Extensions to the computational model.

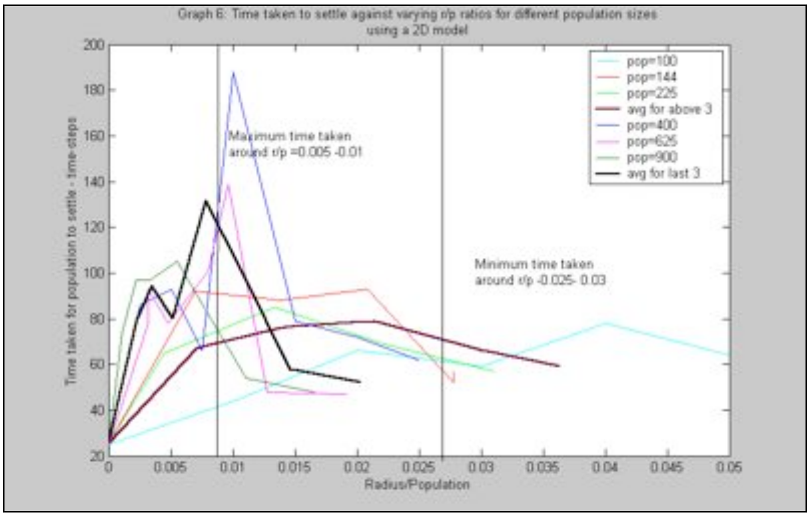
The computational model is an extension of the 1D model. When the makePopulation() method is called in the environment class, a 2D vector is created by creating a vector of vectors containing agents. The agents only hold information for their belief value (random) and local average (0). The rest of the methods are the same as above, but are manipulated to iterate through a 2D vector. This allows me to calculate the local average by referring to agents in adjacent vectors.

Analysis of the two-dimensional model.

Testing was carried out in a very similar way to above. Both number of groups formed and time taken for the population to settle were tested – varying both the population size, *k* value and neighbourhood size. I tested populations of 10x10, 12x12, 15x15, 20x20, 25x25 and 30x30 agents.



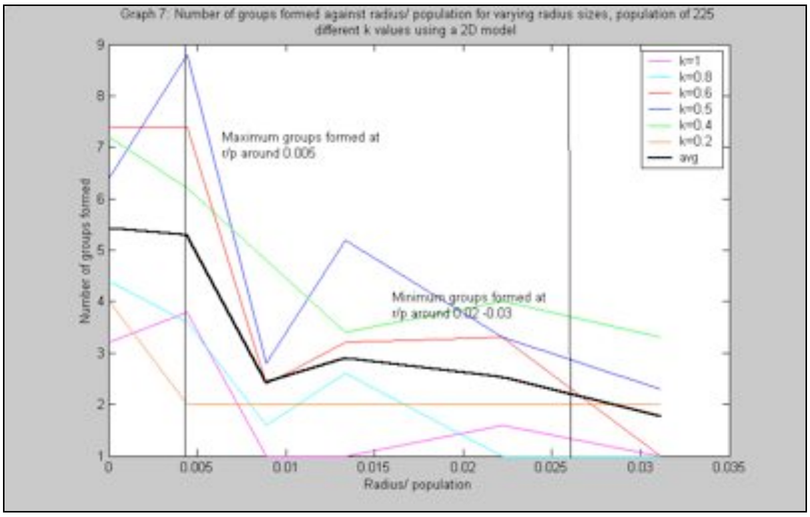
Graph 5: Showing how many groups are formed when varying the radius/population ratio for different sized populations, using the 2D spatial model.



Graph 6: Showing the amount of time needed for population to settle against varying the radius/population ratio for different sized populations, using the 2D spatial model.

Unlike the previous model, the neighbourhood size is four times as big as the radius since a 2D grid formation is being used. *Graph 4* and *Graph 5* indicate a similar result of two critical points being present. The first critical point being that for neighbourhoods around a 50th to a 20th of the population size maximum groups are formed and maximum time is required for the population to settle. The second critical point occurs when the neighbourhood is around a 10^{th} to a 5^{th} of the population size – when this happens the minimum time needed to settle is required and the population acts as a whole forming one group. These critical points are at around half the value of for the critical points present in the one-dimensional model. This is purely because although the radius is the same, the neighbourhoods are twice as big. Therefore for the two-dimensional model a radius half that of the one used in the one-dimensional model is required to achieve the same results.

These results nicely illustrate the fact that for bigger populations more groups or formed at the first critical point.



Graph 7: Showing how many groups are formed when using varying radius sizes with a population size of 200, for varying k values, using the 2D spatial model.

Graph 7 also illustrates that the radius only need be half of the radius needed in the one-dimensional model to achieve the same results.

Results of spatial networks in context:

Having analysed the results of both spatial networks I can now conclude on how space affects the propagation of ideas and belief in a population. As seen throughout the analysis there are two critical points one dictating where the maximum number of groups will occur and the maximum time taken for the population to settle. The second dictating when the population will start acting as a whole. Additionally, I pointed out for small neighbourhoods the population acts like a whole. Finally it was seen that for a two-dimensional network, the radius only had to be half of that for the one-dimensional for the same critical points.

In conclusion therefore:

When an agents neighbourhood is approximately a 10th of the population (5th for 2D) the maximum number of idea/belief groupings are formed.

For smaller neighbourhoods or for neighbourhoods in the region of 40% of the population size, the population acts as one and settles for one (2 depending on k) belief value.

A Static Small World network:
This model implements Watt's small world model and integrates it with Zanette's mathematical model. Here a variable *p* represents the probability of reconnecting an agent's tie from one pre-specified neighbour with a random agent in the population. This model allows for the transformation of a uniformed network into a small world network. There is special emphasis on *p* being around 0.01-0.03 this is where the small world structure is said to take place. The agents still have a local neighbourhood however the higher the probability the more likely their neighbourhood is to contain non-local agents. This model is static that is to say, reconnection happens only once at the beginning and then stays fixed for duration of the 500 time-steps.

Extensions to computational model.
For a small world network a new constructor is introduced to the Individual class. This constructor allows the agent to hold information about who is in its neighbourhood. This information is represented by a vector that holds Integer objects that correspond to the positions of the agents within its neighbourhood. At run time three variables are required, a *double* relating the k constant, an *int* relating the radius of the neighbourhood and another *double* this represents the probability of random connections. The environment class is the same as for the one-dimensional model with the addition of a couple of methods.

makeFinalPop()
This method takes the initial population of agents- containing information about their belief value and local average only. It then iterates through the population. At each iteration a vector is created containing Integer objects representing the positions of their neighbours. At present the neighbourhoods are completely local and uniform. A new agent is then constructed using the new constructor method which adds the vector of neighbours to the agents internal information. The new agent replaces the old agent. This results in a population of agents all containing information about their belief value, local average and who their neighbours are.

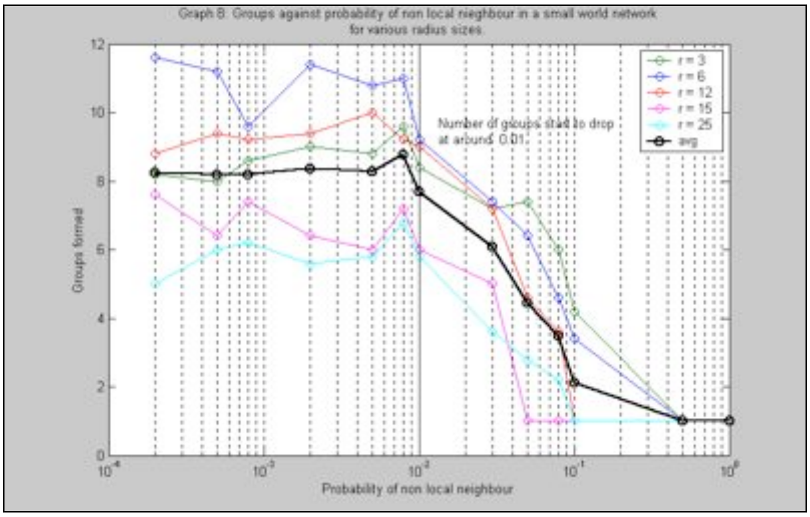
randomisePop()
This method changes the information about the agents neighbours, making use of the probability constant. This method iterates through each agent in the population, and then iterates through their neighbourhood vector. At each step in the neighbourhood vector a random number between 0 and 1 is produced, using the built in Maths class. If this number is smaller than the probability constant, then another agent within the whole population replaces the current neighbour. To chose a random agent I use a number generator that generates a random number within the limit of the population size. This then creates a neighbourhood containing both local and global agents.

The rest of the model works as like the one-dimensional model. Calculating the average belief value of the neighbourhood of a particular agent and updating their belief accordingly. In this model an agent's neighbourhood is only changed once.

Analysis of a Small World Network:
For this model I used a population of 200 agents. I tested four different radius' sizes: 3,6,12,15 and 25 against 14 values for *p:* 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1. For these experiments *k* was constant at 1.

Following this I again used a population of 200 agents but this time tested the affect *k* had on the results. For this investigation I kept the radius size constant at 6 – testing k values of 1; .8., .6, .4 and .2 against the 14 values of *p* aforementioned.

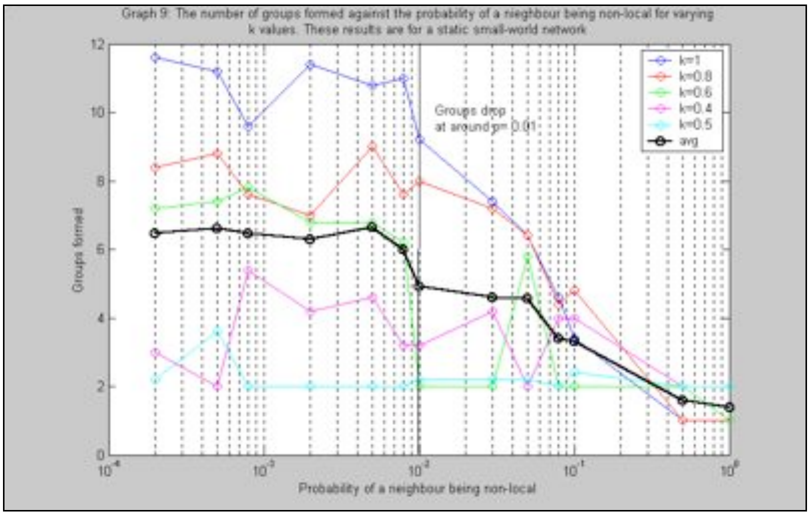
For displaying results from models using a small-world network, a log scale is used.



Graph 8:Showing the number of groups formed within a population of 200, against the probability of a neighbour being non-local. These results are for a static small world network where the k value is constant at 1.

Straight away from *Graph 8* you can see that there is a drop in the number of groups formed at p≈0.03. This result coincides with small world network theory that suggests when p≈0.03 the network stops acting like a uniformed network and starts acting some-what like a randomly connected network. For all values of r, the number of groups formed is pretty much static regardless of the probability of reconnection until p≈0.03. Once you start reconnecting the ties in the network with a probability above 0.03 the number of groups formed falls until the network is randomly connected and one group is formed at around p=0.1. Randomly connected neighbourhoods act as one due to their characteristic short path lengths and by being highly un-clustered.

Again when testing different *k* values (*Graph 9*) the results are not as clear. Low values of *k* affect the agent more than the probability of reconnection since their neighbourhood is no longer influencing them.



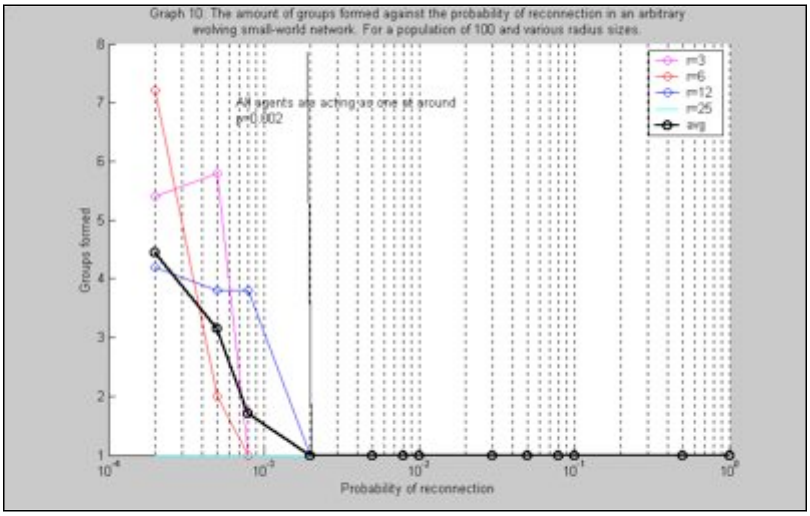
Graph 9:Showing the number of groups formed within a population of 200, against the probability of a neighbour being non-local. These results are for a static small world network where the radius value is constant at 6 and the k values vary.

Results of a Static Small World Network in Context:
I have shown, unsurprisingly, that when a small world network is projected onto Zanette's model the results follow small world network theory. What I mean by this is that at p≈0.03 the number of groups formed start to decrease. It's suggested by Milgram, Watts and others that real social networks are best represented by a small world network structure of p=0.03, they then exhibit six degrees of separation. Evidence of this is in actor networks and scientific referencing. What does this therefore say about the propagation of ideas/beliefs within a population?

Evolving Small-world networks.
I then looked at evolving small-world networks in certain ways – arbitrary, with preference for similar agents and with selection for similar agents. This allows the agents to have a dynamic interaction with their environment. By selecting or preferring agents who are similar to them the agents directly affect the structure of the network. Thus moving toward a more agent-based model than Zanette's provides for. Additionally I looked at the case where agents prefer 'original' belief values.

Arbitrary.
This is an extension of the small-world network model. However, the network does not stay in a constant connection pattern. Instead at each time step, the randomisePop() method is called, and a new set of neighbours is created given the probability constant. This results in a more dynamic system. However in this model the rewiring is arbitrary there is no rule governing how rewiring works. The only change to the program is that the randomisePop() method is called on each time step.

Results:
For this experiment I used a population of 100 agents. For this model I tested radius' sizes of 3, 6, 12 and 15 against probability values of: 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1.



Graph10: Showing how many groups are formed for different probability in an arbitrary evolving small-world network.

In this model every neighbour in an agents neighbourhood can be swapped at each time step given the probability. It is not surprising therefore to see the results revealed in *Graph 10*. The results show for this model that for p values of 0.001 to 0.002 only one group is formed. *Figure 9,* shows the final configurations for an arbitrary evolving network for probability values of 0, 0.001 and 1. It is clear from the final network configuration that when an agent is changing each neighbour at each time step with a probability as low as 0.001 the network quickly resembles a randomly configured network. Thus resulting in only one group being formed.

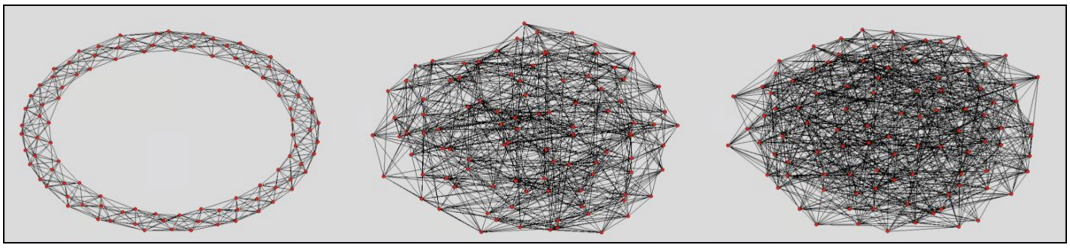


Figure 9: Showing the final network configuration for an arbitrary evolving small world network for values p=0, p=0.001 and p=1 (left -right).

Preference for similar neighbours.

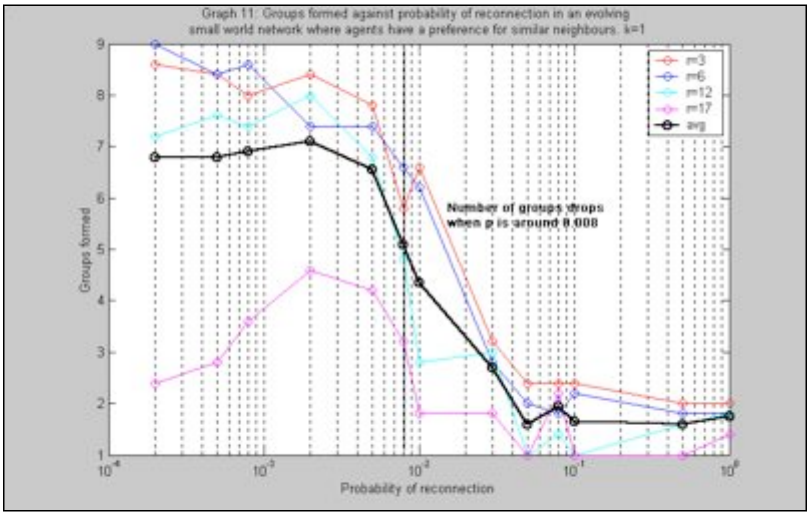
In this model, the agents direct the network topology by choosing which neighbour to remove, if rewiring takes place. If an agent has four neighbours it would chose the neighbour with the most dissimilar belief value to be removed if given the chance for a new acquaintance. In this model the agent has no choice over who replaces the removed neighbour. The new neighbour is selected at random.

The computational model has two extra methods: sortFriends() and swapFriends() together these create a bubble sort algorithm for sorting an agents vector of neighbours into order, determined by the difference in the belief gap. The belief gap is simply the absolute difference of the agents belief and their neighbours. At the end of this method the agent will have their neighbours in order of belief similarity. There is an additional method for finding a new neighbour newFriend(). This simply iterates through each agent in the population. Given the probability constant it swaps its worst neighbour or foe, for a random agent in the population. Again at this stage the agent has no choice in who it's new neighbour is. The only choice they have here is who to swap.

Results:

For this experiment I used a population of 100 agents. For this model I tested radius' sizes of 3, 6, 12, 17 and 25 against probability values of: 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1. For all tests *k* was kept constant at 1. I tested the case where agents prefer to keep similar agents to dissimilar agents and additionally the opposite case where agents prefer to keep dissimilar agents.

Again, unsurprisingly the number of groups formed starts to decrease when p is around 0.01- the '*special spot'* in small-world network theory. The number of groups formed falls to either one or two, when p ≈ 0.03. When agents prefer similar agents, two groups are formed for the majority of the tests. This is likely to happen since agents with more negative belief values are going to get rid of neighbours with positive belief values, and vice-versa. It is as though two *tribes* or parties are being formed. However, this is not always the case in some runs only one group is formed – this could either be because of the initial belief values being in favour of one polar more than the other, or simply due to the random nature of the new neighbour being chosen. Two groups are more likely to be the outcome of smaller neighbourhood sizes. From the networks drawn in Figure 10 you can see that at p=0.01 two groups are starting to form Although the graphical results clearly show that two groups are formed for high probability values, the network shown in figure 10 is not as dense, This is purely due to the computational model. Since the model picks any random agent from the population regardless of the fact that they might already be present in a particular agent's neighbourhood or the fact that it might the agent themselves. The number of individual agents within one neighbourhood might actually fall due to multiple ties with one neighbour or counting themselves as a neighbour. Therefore the network configuration seems less dense than when the probability is smaller. However, the groups formed are more clearly separated. Still the fact still remains that two *tribes* of agents are usually formed for the majority of the tests.



Graph 11: Results for an evolving small world network due to agents having preference for similar neighbours. Varying the probability of reconnection, p, and the radius size for a population of 100 with k at a constant 1.

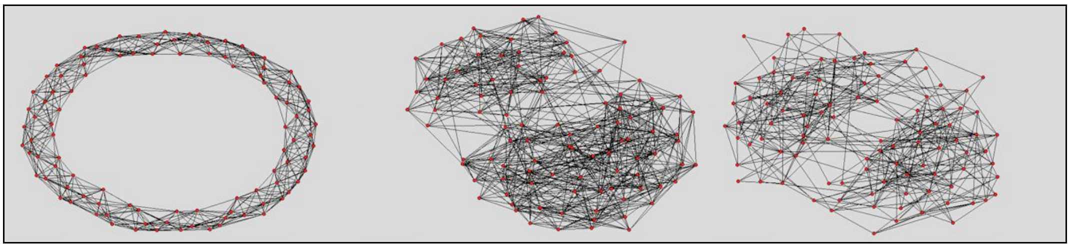
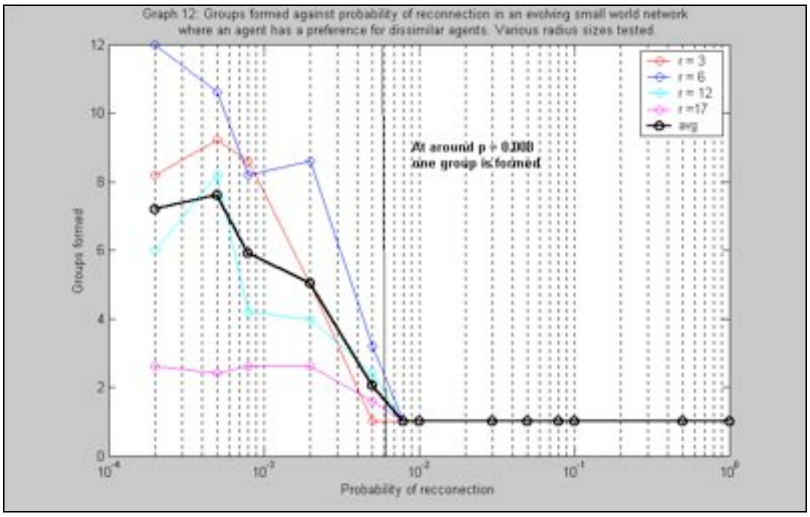


Figure 10: The final network configurations for evolving small world networks due to agents preferring similar agents. Networks are show for p=0; p=0.01 and p=1.

When agents prefer dissimilar neighbours you will see the opposite happen. Instead of the population been separated it is drawn together, while the most extreme opinions cancel each other out. Moreover, there is a sharp change in number of groups formed as soon as the probability is around 0.008 or 0.01. Instead of a more gently decline in groups there is a sudden drop to only one group being formed. As you can see in figure 11, the final network configuration for when p=0.01 resembles a random network, although not as tightly drawn in as the configuration for when p=1.



Graph 12: Results for an evolving small world network due to agents having preference for dissimilar neighbours. Varying the probability of reconnection, p, and the radius size for a population of 100 with k at a constant 1.

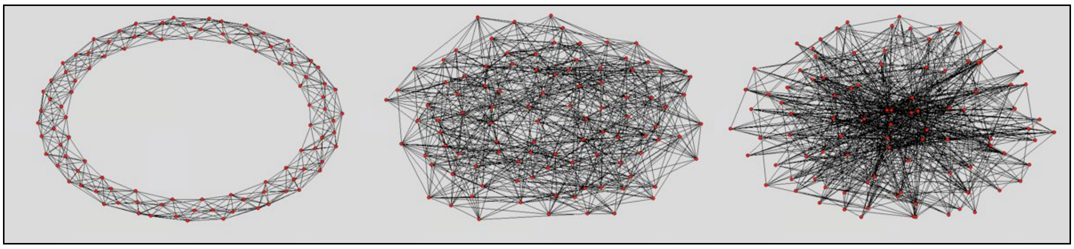


Figure 11: The final network configurations for an evolving small world network due to agents preferring dissimilar neighbours. Networks for when p=0; p=0.01 and p=1 are shown.

Preference and Selection of like agents.
This model is very similar to the above model. However instead of swapping their worst neighbour for a random agent in the population, they swap them for an agent who holds the nearest belief value in the population. They selectively chose their neighbours and therefore the network changes due to the agent's preference and selection. This model is very similar to the above model but contains two additional methods. Together these perform the bubble sort algorithm on the whole population relative to the agent's belief value. This way the agent can pick which the agent most suited to their belief value to take the place of their worst friend. Additionally I have investigated swapping the agents preferences for more original beliefs. Here the agents are after innovation. In this model the agent, sorts their neighbours preferring those who have a bigger belief gap compared to their own belief, they then swap the neighbour closest to them for an agent in the population who has the most unlike belief value. This creates a different dynamic in the model.

Extensions to the computational model:
Again this program is very similar to the former preference model. However, it has a couple of extra methods that allow an agent to chose who to have as their new friend. Additionally within this model the agent cannot chose another if they are already a friend or indeed if it is themselves.

sortPop(): This method is called from the newFriend() method described above. Instead of allotting a random agent of the population to the friendship group the sortPop() method is called and the best-fit friend is returned to replace the worst friend in the agents friendship group. This method has two variables passed to it, the vector representing the whole population and an Individual object representing the agent who is finding a new friend. Details about the agents friendship group, and belief value are pulled out. The population vector is then iterated through, using a for loop. At each iteration the current agent being looked at has its belief value pulled out. The absolute difference between the two agents is worked out. If the difference is smaller than any before – to begin with there is a double set at a 1000 to represent the belief gap – then the isFriend() method is called. If the present individual is not a friend and is not the agent themselves, then that individual is set as being the best possible friend for that agent. The iteration continues until every individual in the population has been examined. At the end of this process, the method returns the position of the individual who is the best possible friend for that agent. This individual then replaces the worst friend of the agent at the newFriend() method.

isFriend(): This boolean method simple takes the agents friendship vector – which represents his friends, and an int representing the position of a new possible friend. It then searches the agents friendship vector, via a for loop, pulling out each friend in turn and retrieving details about the friends position in the population. If any of the friends position equals that of the possible new friend, the method returns true otherwise it returns false and the possible friend can be set as the possible new friend.

Results

For this experiment I used a population of 100 agents. Using this model I tested radius' sizes of 3, 6, 12 and 15 against probability values of*:* 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1. For all tests *k* was kept constant at 1. Again I tested the case where agents prefer to keep similar agents to dissimilar agents and additionally the opposite case where agents prefer to keep dissimilar agents.

Although the results statistically are very similar to the situation when an agent prefers to keep particular neighbours the network configurations in figure 12 tell a different story. At $p \approx 0.01$ there is a very defined population constituting of two groups of agents. This intense network configuration is brought about by the fact that agents actively look for similar agents to have in their neighbourhood. I expected two tightly dense groups at p=1 however this is not the reality. Although two groups are statistically formed based on belief value, the population seems to have broken off into smaller groups. This is due to the nature of the model only allowing a new agent to be part of a particular neighbourhood if they are not already in it. So although the population is divided in two when it comes to their belief value they are actually divided into smaller groups when it comes to their social network.

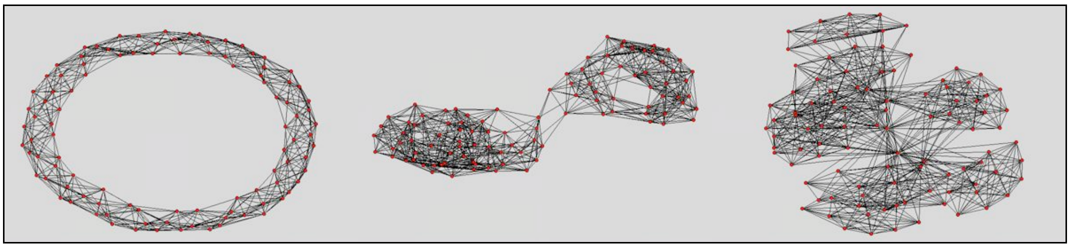


Figure 12: The final network configurations for an evolving small-world network due to agents' selection of similar neighbours. Networks shown for p=0; p=0.01 and p=1.

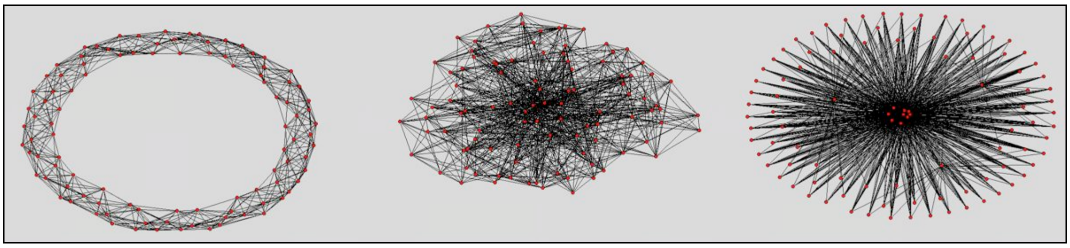
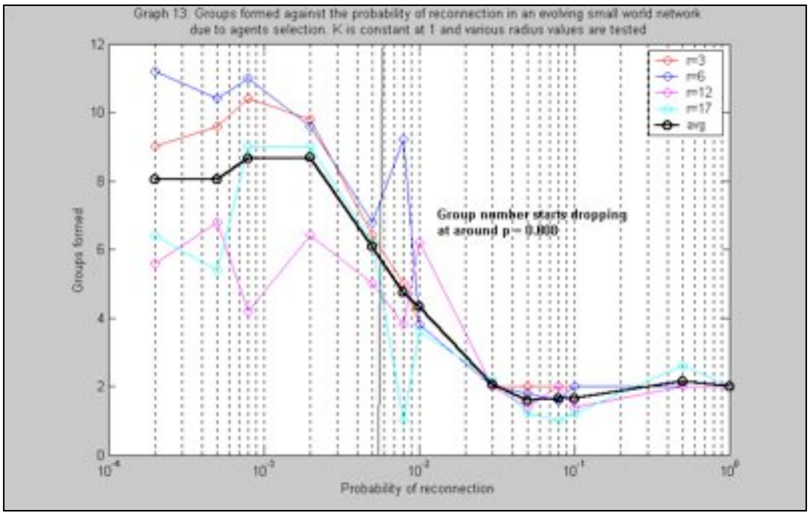
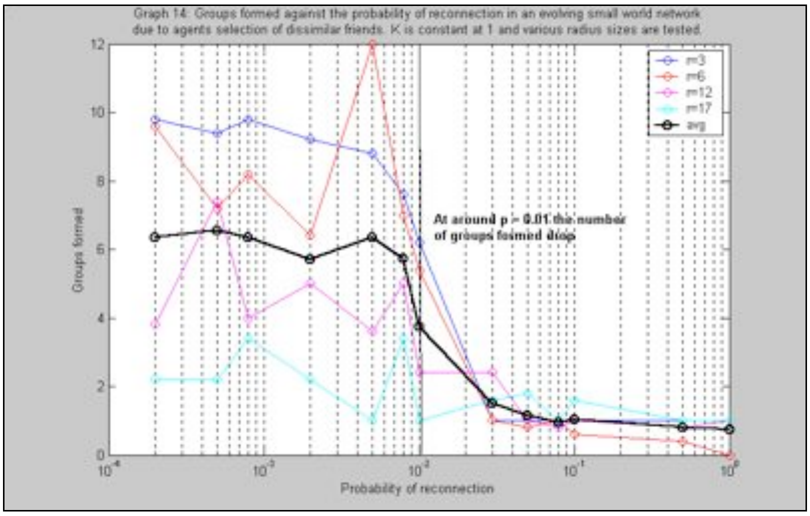


Figure 13: The final network configurations for an evolving small-world network due to agents selection of dissimilar neighbours. Networks shown for when p=0; p=0.01 and p=1.



Graph 13: Results for an evolving small world network due to agents selecting similar neighbours. Varying the probability of reconnection, p, and the radius size for a population of 100 with k at a constant 1.



Graph 14: Results for an evolving small world network due to agents selecting dissimilar neighbours. Varying the probability of reconnection, p, and the radius size for a population of 100 with k at a constant 1.

Overlaying Topologies.

The next model I have devised is constituted of two networks. A one-dimensional model represents an agent's local neighbourhood and a small-world network represents their friendship group. During the testing stage I investigate different ratios of influence the two networks have over the agent. For example, I test when the neighbourhood network has 20% of the influence and the friendship network has 80% of the overall influence. In this model I'm using an evolving small-world network where the agent has preference for similar friends, rather than a static small-world network. However, the agent has no control over who replaces the friend to be swapped.

Extensions to computational model.

This model is basically a combination of two existing models – the one-dimensional model and the evolving small world network. The main difference here is when it comes to update an agent's belief value. Both networks will influence their belief; therefore one must calculate the average belief value of each network. Once the belief value for each network is calculated, it is then divided by ten and multiplied by the appropriate ratio. For instance if the neighbourhood network has 80% of the influence the local average is divided by ten and multiplied by 8. The two average belief values are then totalled and stored for each agent in the population. The average belief value for each agent is then, where N= neighbour and F=friend:

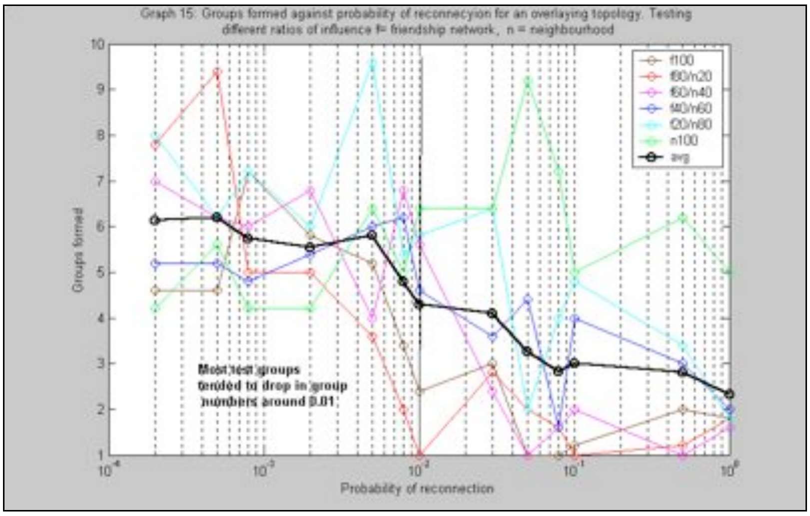
*Avg= (((sum of N belief)/ N)/10)*N ratio) + (((sum of F belief)/ F)/10)*F ratio)*

There are no extra methods in this model

Results

The population was set at 100 agents. For this model I test a variety of different parameters. Firstly I kept *k* constant to 1 and the radius constant at 10, I then tested different ratios (N=neighbourhood, F=friendship group): F=100%, F=80% / N=20%, F=60% / N=40%, F=40% / N=60% , F=20% / N=80% and N=100% against the 14 values of *p:* 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1. With this set of parameters I tested for both when the agents have a preference for similar agents and when they have a preference for dissimilar agents.

Additionally, keeping the ratio constant at 50/50 and the radius at 10, I tested varying *k* values: 1, .8, .6 and .4 against the values of *p* mentioned above. The last investigation was into the factor of radius sizes on the networks. Keeping the ratio constant at 50/50 and *k* constant at 1, I tested radius sizes of 3, 6, 10 and 15 again using the 14 values of *p*.



Graph 15: Shows the results of testing ratios of influence each network has in an overlaid topology. Groups formed are plotted against probability of reconnection. K is constant at 1, the population is of 200 agents and the radius size is 10. In this experiment agents prefer similar friends.

Statistically there isn't much variation in the results for different ratios of influence each network has on the population. Obviously when there is no friendship – small world network – present the number of groups does not change given the probability. However the results indicate that even with the smallest capacity of influence the friendship network affects the number of groups formed based on the value of p. However when the friendship network has little influence the decline in groups formed is much gentler. The results indicate that generally two groups are formed for high values of p for all ratios except when there is no influence from the friendship group. However at the *special spot* for small world networks, p≈0.01, its only for tests where there is a big influence from the friendship network that around two groups of agents are formed. In figure 14 below you can see that for p=0.01 there two groups are starting to form when the friendship group has 80% of the influence. In the other networks there are inklings of a divided population being formed although with greater influence from the neighbourhood network the population stays slightly more uniformed in nature.

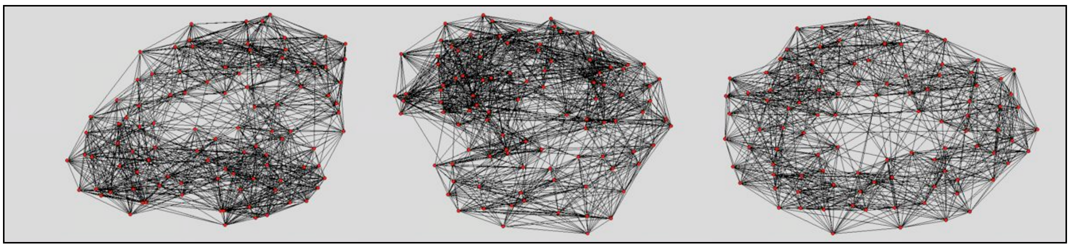


Figure 14: Showing the final network configuration for overlaid topologies when p=0.01. Three ratios of influence are shown: f8/n2; f5/n5 and f2/n8 (left-right).

However, Figure 15 shows that the population does become divided in the end for populations experiencing high influence from the neighbourhood network. From graph 16 one can see that this population division is more likely to happen when there are larger neighbourhoods present. This is probably due to the fact if there are large neighbourhoods then the average belief of that neighbourhood will be more reserved –

any extreme beliefs present will be counteracted. At the same time the friendship group will be just as large but the average belief value for the group will be more acute due to the agent choosing any agent with a belief value that is divergent from his own.

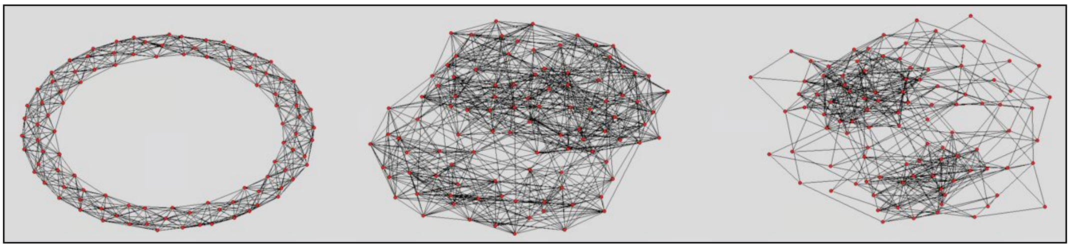
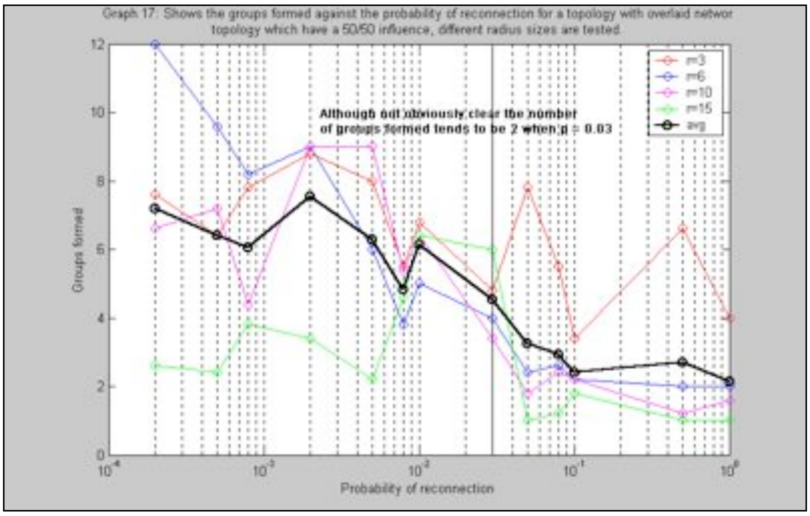


Figure 15: The final configurations of a population with the neighbourhood network having 80% influence of two networks. Networks shown for p=0; p=0.01 and p=1.



Graph 16: Shows the results when allowing both networks 50/50 influence. Various radius sizes are then tested. The number of groups are recorded for each value of probability of reconnection. The population is 200, and K is constant at 1.

Graph 17 shows the affect k has on the population. However since for low values of k the population normally divides into two anyway this has little affect on the statistics. In figure 16 we can see the end network configurations. From these we can tell than even though two groups are formed this is more due to the fact that k is of a low value rather than any major population divisions happening. The networks for a low k value such as 0.4 become more like random networks the higher the probability of reconnection. There are no groups present in the actual population they are due to low k values.

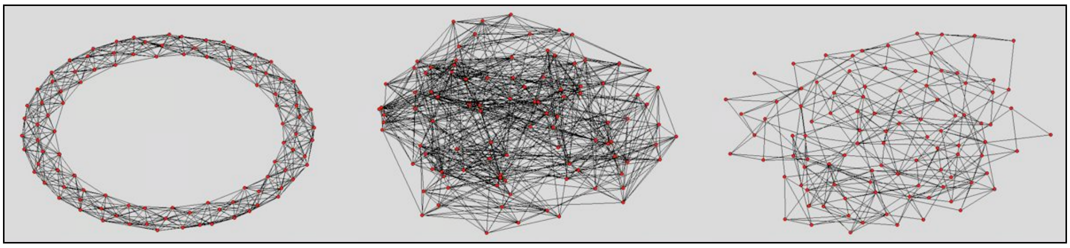
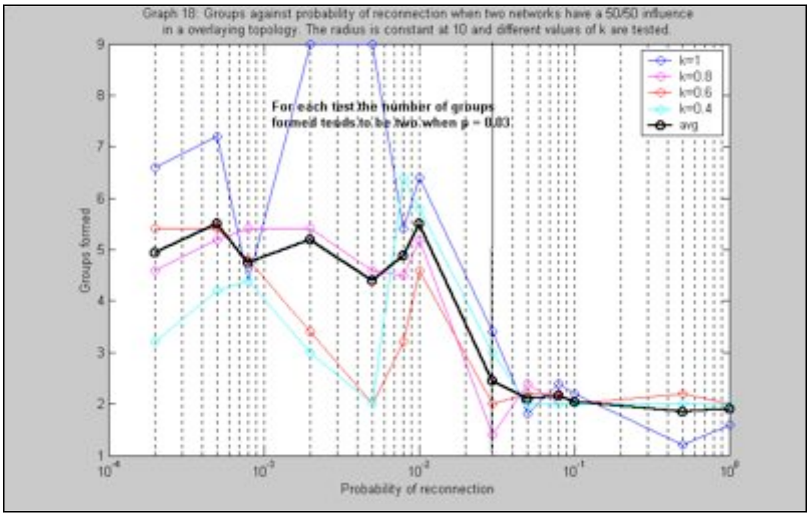
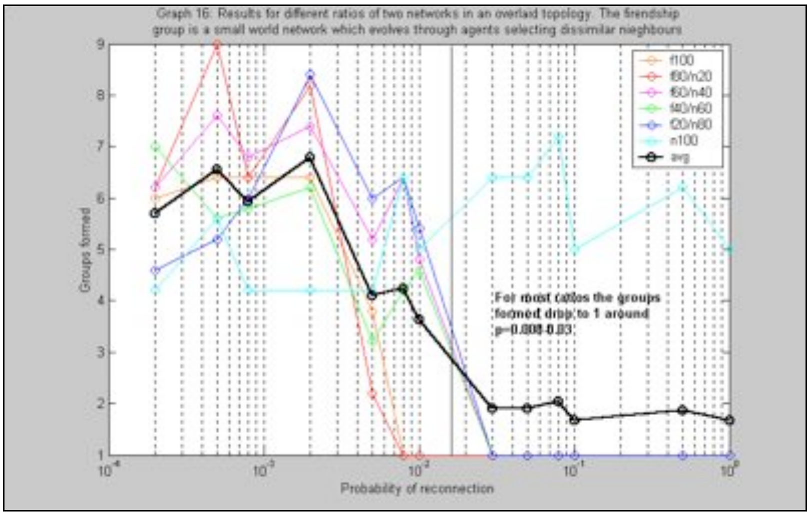


Figure 16: The final network configurations for a 50/50 overlaid topology for k=0.4. Networks for p=0; p=0.01 and p=1 are shown.



Graph 17: Shows the results when allowing both networks 50/50 influence. Various values of k are then tested. The number of groups are recorded for each value of probability of reconnection. The population is 200, and the radius is constant at 10.

The last experiment involving overlaid topologies is for when agents prefer dissimilar friends. As with all preceding experiments the population acts as a whole given that the probability of reconnection ≈ 0.01. For networks with more influence from the neighbourhoods this value rises to 0.03. This behaviour is present for all network ratios, until the friendship group has no influence at all. This again suggests to how effective the small world network is.



Graph 18: Shows the results of testing ratios of influence each network has in an overlaid topology. Groups formed are plotted against probability of reconnection. K is constant at 1, the population is of 200 agents and the radius size is 10. In this experiment agents prefer dissimilar friends.

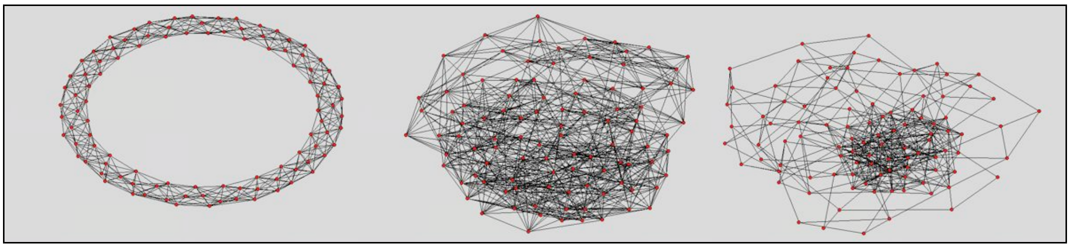


Figure 15: The final configuration of the population when two networks are overlaid with a 50/50 influence one with a preference for dissimilar neighbours. Networks shown for p=0; p=0.01 and p=1.

Positive Node Degree Correlation:

This method is different in the relative neighbourhood sizes. Each agent in this model has a different neighbourhood size, given by a gaussian distribution. The network used in the model is a small-world network. It is evolving and here the agents affect how their new friend is chosen. However, unlike the previous models their worst friend is replaced not by someone who is similar in belief value but who has a similar amount of friends. Although, when deciding which friend to change an agent chooses the one most dissimilar in belief value.

Extensions to the computational model:

As mentioned above the agent’s neighbourhood size is predetermined by a gaussian distribution. This happens in the makeFinalPop() method. During this method an extra for loop is added, which iterates across the population. On each iteration a random number generator is used to find the next gaussian number. However since the majority of these numbers are between –1 and 1 I map these numbers onto some more suitable for neighbourhood sizes. Multiplying them by 4 and adding 10 the result does this. This gives a gaussian distribution with most of the numbers being around 10/11. Since the neighbourhood size cannot be too high or in the minus I also have cut of points. So any number below 0 is 0 and any number above 20 is 20. Below is a diagram giving a sample of the distribution for 100 agents. As one can see most agents have a friendship group between 6 and 14 with few having 20 or 1. Which seems to be more realistic than a uniform distribution across the population.

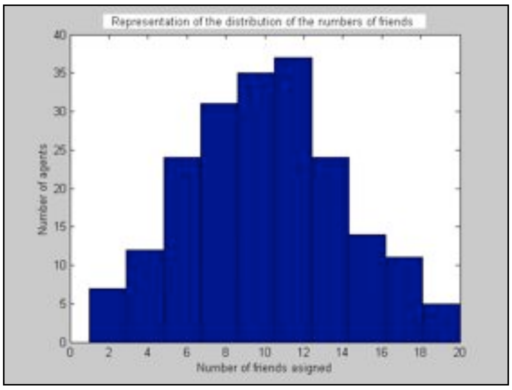
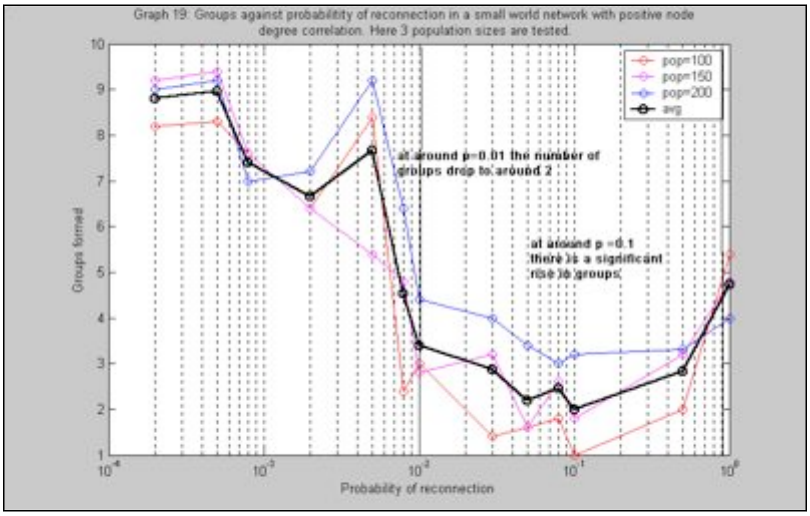


Figure 16: Graph showing the gaussian distribution of number of friends for agents within a population.

The other method that has been modified is the sortPop() method. This method before has found a best-fit friend for any given agent, determined by their belief value. However since we are now dealing with positive node degree correlation, we want this selection to be due to how many friends they have. The main method is still the same: at the newFriend() method the sortPop() method is called passing the population vector and the current agent. At the sortPop() method the current agents size of neighbourhood is retrieved and stored for comparison. For the size of the population vector a for loop is run. At each time step a random individual from the population is pulled out, using the method supplied by the Maths class. I do this purely because by looking at the population in order not everyone will be fairly looked at – if most people have 10 friends, then the first agent in the population will be chosen rather than any other. Once a random agent is found the size of their friendship group is found. If this is either 3 bigger or 3 smaller than the current agents friendship group, they become a possible new friend. As before it is then checked that they are not already a friend and not the current agent themselves. If all is fine then that agent is set as the best agent, until a new one is found or the for loop has finished. The method then returns the position of the new friend, and it gets assigned to the agents friendship group in the newFriend() method.

Results:
Since with these experiments the neighbourhood sizes are internally set, I tested the affect the size of the population has on the network. I kept k constant at 1 and tested Population sizes of 100, 150 and 200 against 14 values of p: 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1.



Graph 19: Shows the results for a small-world network evolving due to positive node degree correlation. The groups formed for different probabilities of reconnection is plotted for three different population sizes.

Graph 19 points to the initial small world pattern whereby at p=0.01 there is a drop in the groups formed. Then for higher values the population either acts as a whole or is divided into two camps. However, there is another pressing pattern emerging from the statistics. That is for p≈1, the number of groups formed rises again. Of course one explanation for this may be that because of the Gaussian distribution a few agents only get allocated 1 friend. This may result in them forming their own little group where their belief value isn't going to change that much. Furthermore if we look at the output from Matlab, figure 17, we can see some very interesting dynamic agent behaviour. It seems as though there are two established belief groups and a few agents that swap between the two. Why should this be? The final network configurations in figure 18 may point to an answer.

From figure 18 we can see that indeed for p above 0.01 two groups are being formed. However also present in the network for p=1 are some 'loner' agents and agents that are connected to both established groups. If an agent has ties to both groups then due to the high probability and having to change a friend at each time step their belief value is likely to oscillate between the two. Therefore the analysis seems to suggest that more groups are formed due to 'loner' agents forming their own groups and agents that are connected to both groups fluctuating between the two.

These are results are true of all population sizes, although are more dramatic for the smaller population.

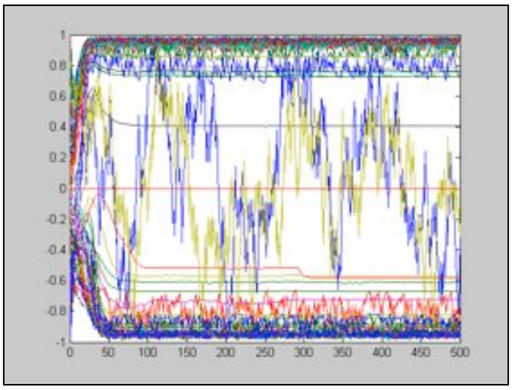


Figure 17: The output from Matlab representing agents belief values over time for positive node degree correlation, for when p=1.

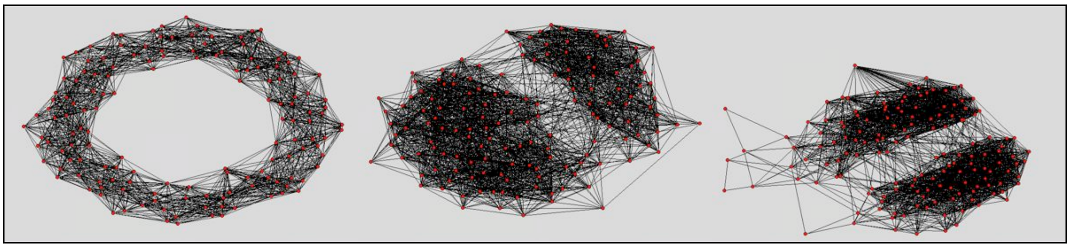


Figure 18: The final network configurations for a small-world network due to positive node degree correlation. Networks shown for p=0; p=0.01 and p=1.

Preferential attachment:

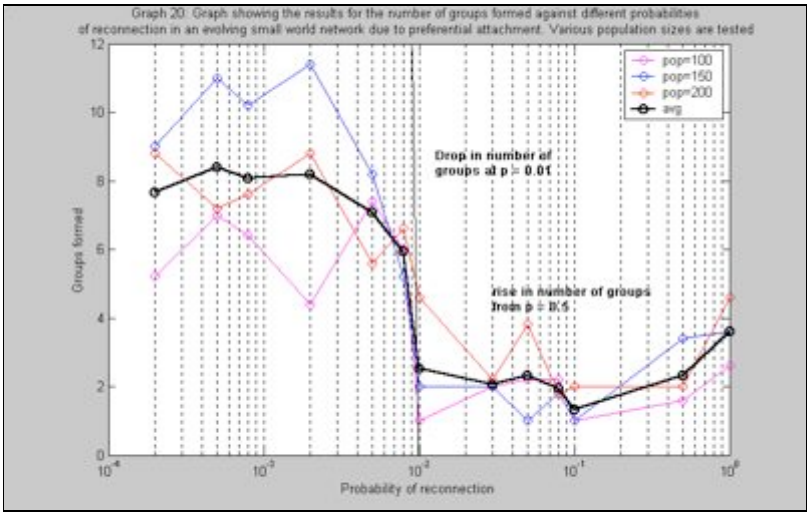
Much like the model which incorporates positive node degree correlation, this model also assigns the agents friendships on a gaussian distribution. Again this is an evolving selective small-world network. However in this model the selection of a new friend is not based on belief value or positive node degree attachment. Instead as Barabasi and Albert (1988) suggest an agent chooses a new friend based on how many friends they have. An agent chooses to have the agent with the most friends as its friend.

Extensions to the computational model:

This model is exactly the same as the above, positive node degree correlation, model. However it differs in its method of selection in the sortPop() method. Here the individual deemed as best-fit as new friend is the agent with the most friends. This is simply done by finding out who has the most friends – using the size of their friendship group- and this agent is set as the best-fit if they are not already a friend or are the current agent themselves. They stay the best-fit until someone better is found or until the for loop finishes they are then returned as the best fit.

Results:

Again as in the previous model since with these experiments the neighbourhood sizes are internally set, I tested the affect the size of the population has on the network. I kept *k* constant at 1 and tested Population sizes of 100, 150 and 200 against 14 values of *p:* 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1.



Graph 20: Shows the results for a small-world network evolving due to preferential attachment. The groups formed for different probabilities of reconnection is plotted for three different population sizes.

Although not quite as exaggerated as for positive node degree correlation, the results for preferential attachment show the same rise in the number of groups formed for high values of p. Likewise this may be due to the Gaussian distribution. This time however nearly half the population will influence an agent who may have around 40 friends. This might result in his belief value being around the middle ground. Additionally if they have to swap a friend at each time step – due to the high probability - they might have a fluctuating behaviour between the two groups, as figure 19 seems to suggest. Looking at the final network configuration for when p=1 we can see that there are agents outside of the two main groups, these outsiders seem to have a great many ties, so this could be a plausible explanation, for rise in group. Obviously since it will only be agents with many ties that have this fluctuating behaviour, and because no individual sub-groups are formed, the number of groups won't be as many as for positive node degree correlation. Again the population size seems to have little affect on the behaviour of the population as a whole.

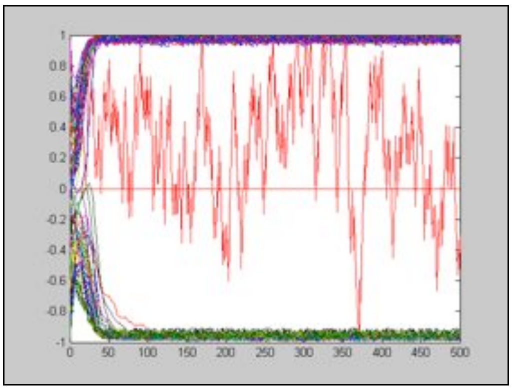


Figure 19: The output from Matlab representing the belief values of the agents over time, for preferential attachment and a probability of 1.

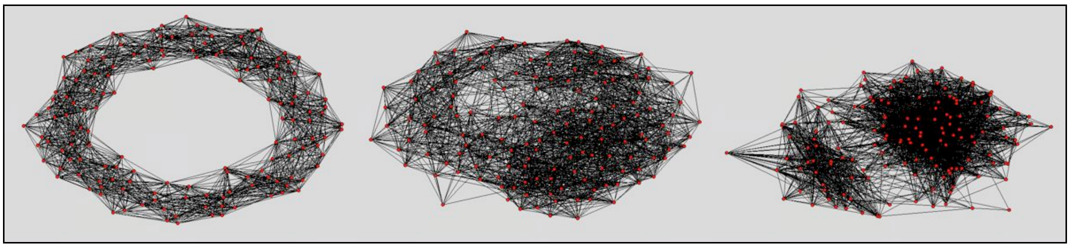


Figure 20: The final network configuration for a small-world network evolving due to preferential attachment. networks where p=0; p=0.01 and p=1 are shown.

Varying the internal states of agents across the population.

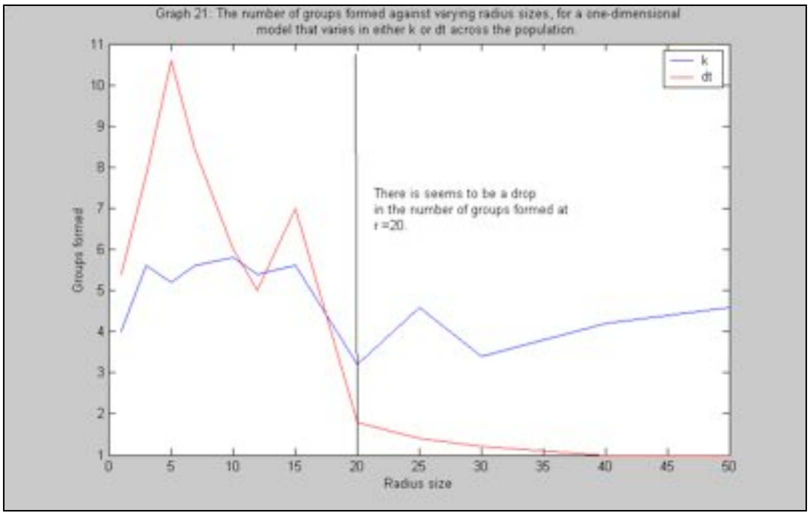
This is a more basic model which looks at what happens when *k* and *dt,* how much an agent is influenced, is varied across the population. I simply take the basic model but as well as assigning a double representing the agents belief they have another for representing how much they are influenced by others or how quickly they adapt to change (the larger *dt* the quicker an agent will change).

Extensions to the computational model:

This is basically the one-dimensional model with an added variable in the Individual class constructor method representing the variable *k or dtfor* each agent. This variable is assigned during the makePop() method, and is given a value using the random number generator which is mapped between 0 and 1. These variables are then pulled out at the updatePopulation() method in the Environment class and used in the final calculations of the new belief value for a given agent. In these models an agent has his own randomly set *k* or *dt* value.

Results:

These experiments were quite simple. I tested what would happen with varied *k* and *dt* across the population on a one-dimensional model, when using different radius' sizes: 1, 3, 5, 7, 10, 12, 15, 20, 25, 30, 40, and 50. For all tests a population of size 100 was used.



Graph 21: This graphs shows the results for when the agents' internal states – k and dt – are varied across the population. Different radius sizes are tested and the number of groups established recorded. For all test the population was 100.

These results are not very novel or exciting. They seem to suggest that again for both varied k and varied dt, there are maximum groups formed when neighbourhood sizes are approximately a 10^{th} of the population. In accordance to the spatial networks, when the neighbourhoods sizes is in the region of 40% of the population size, the number of groups formed drop and stay in here for the duration of the experiment.

Toward an Agent-based model.
Finally I created a model that took into account all the various areas of extension. The agents in this population varied greatly. Firstly they all had a differing numbers of friends, as they did in the positive node degree correlation and preferential attachment models – again given by a gaussian distribution. Secondly the way the influence was calculated was more agent-based. Instead of an agent having only one value representing how much they are influenced, they each have two: one representing how much they get influenced, *k,* and one representing how much they themselves influence *f*. During the calculation of an agents updated belief at a given time step, the amount they are influenced by a particular neighbour is worked out by the total of how much they get influenced and how influential their neighbour is. This is calculated by taking the agents *k* value and their neighbours *f* value. This allows for a more agent-based model.

The network topology was of an overlaying topology where an agent has both a local and a friendship network. However different agents have different ratio values, on how each network influences them. For example some might be more influenced by their local network than their friendship network. This differing ratios formed a gaussian distribution so most agents would be equally influenced by both networks where as few agents would be influenced by just one of the networks. As before the neighbourhood network is based on a one-dimensional spatial network and the friendship network is based on a small world network. In this case the small-world network is evolving, and evolves through agent selection of their new friend. The swapping of a friend is based on the probability of reconnection *p.* However previously when investigating small world networks evolving through selection, only one type of selection has been present at any one time. In this model agents chose their friends due to different selection types – belief value, positive node degree correlation, preferential attachment and for like behaviours. By like behaviours I mean they make ties with someone if they have similar influential *f*/influenced *k* behaviours. Additionally an agent has the predisposition to prefer either like or unlike neighbours. This means within the whole population different agents chose to make different ties for different reasons.

Extensions to the computational model:
Firstly in the Individual class there is a new constructor method to deal with agents with many different behaviours and internal states. The constructor method allows agent objects to now be created with the following information: *x* the agents belief value, *k* the value corresponding to how influenced an agent is, *f* the value corresponding to how influential an agent is, *pos* the agents position within the population which allows me to refer to a point to certain agents, *friends* the amount of friends an agent has, *ratio* this corresponds to how much the frienship network of an agent influences them, *neighbours* which stores an agents group of friends (represented by their position in the population) in a vector, *selec* which is a number that codes what type of selection that agent chooses their friends with and *type* either

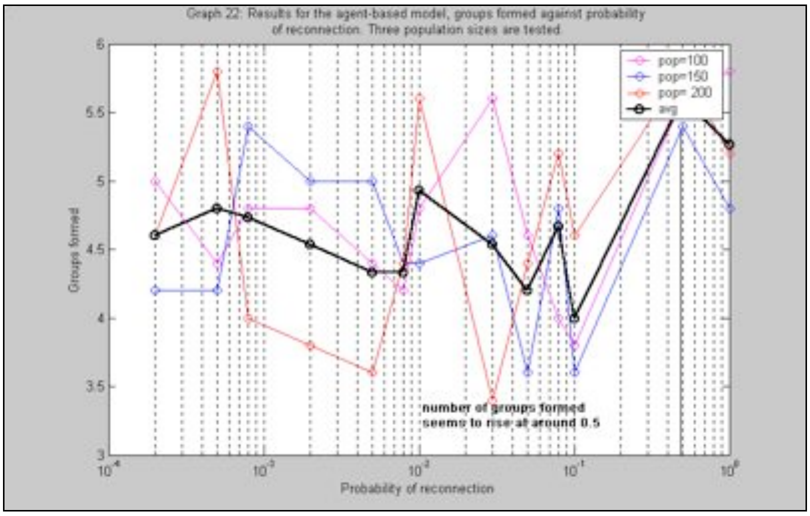
0 or 1 which dictates if they prefer similar or dissimilar friends. All of these data types are set up in either the makeInitialPop() or makeFinalPop() methods.

The makeInitialPop() method simply creates a vector of agents each assigned a random belief value and their position within the population. It is in the makeFinalPop() method that most of the data is assigned to each agent. Here the population is iterated through using a for loop, at each iteration an agent is pulled out and assigned their final make-up. Firstly the k and f values are assigned by using the Maths class to generate a random number between 0 and 1. This is then divided by two, so when it comes to updating the belief value for any agent a combination of their data and their neighbours can be used for the total influenced. Next the data that corresponds to the type of agent is assigned: if it is 1 then the agent prefers similar friends if it is 0 then the agent prefers dissimilar agents. At first all agents are assigned to 1 – they have a preference for similar friends. However given a probability of 0.15 this may swap to 0. That is to say most agents will prefer to lose dissimilar friends when swapping however around 15% of them will prefer to lose similar friends in the hope for more originality. The selection preference is now chosen using a random number generator which picks numbers between 0-5. The number will signify what type of selection a agent will use when picking a new friend. If given 0 they will pick friends based on similar f and k values, if 1 they will choose friends based on preferential attachment, if 2 they will choose friends due to positive node degree correlation and for all others 3,4,and 5 they will choose friends based on belief values. Although there will be a difference in the way agents choose their friends around 50% will choose them due to their belief. As described in the positive node degree correlation model the number of friends an agent has is next established. Following this the particular ratio preference for networks is prescribed to the agent by a gaussian distribution in a similar fashion to the number of friends. However a different mapping is used so that most agents will have a 50/50 ratio when being influenced from the two different networks. Finally their friendship vector is then filled with agents as described in the small-world networks above.

The next significant modification is in the sortPop() method. This is where a new, best-fit, friend is found. In past models only one selection process has been present however now there are four. When an agent is passed to the sortPop() method his selection type is retrieved. Based on this type, using 'if' statements, the right selection process is chosen. Above the processes are described for when an agent picks a new friend due to belief value, preferential attachment and positive node degree correlation. However, now there is a new selection where an agents friend is found based on similar behaviours for influencing and being influenced. Here the absolute difference between each agent in the population and the current agent are found for their values of k and f, and then summed. The most similar agent in behaviour is said to be the one with the least difference in behavioural gap. This agent is then chosen to be of best fit.

The final alteration is in the combining of the updateAvg() and the updatePopulation() methods. Before an average of all the agents contacts belief values have be found and used with acquiring their new belief value using Zanette's equation. However, now there is no constant k value as in all the other models since the way in which any agent is influenced depends not only on how they are influenced but how their contact influences. So instead, for each contact an agent may have, a new belief value is calculated based on the agent's k value and their contacts f value. The new belief values of each contact are then totalled and summed by the number of contacts. This now gives the agent his new belief value.

Results:
As with the models of preferential attachment and positive node degree correlation most details are internal to the program. Here the only variables that can be altered is the probability of reconnection and the population size. Therefore I tested 3 population sizes: 100, 150 and 200 against the 14 values of p: 0, 0.0002, 0.0005, 0.0008, 0.002, 0.005, 0.008, 0.01, 0.03, 0.05, 0.08, 0.1, 0.5 and 1.



Graph 22: This graph shows the results for the tests on three population sizes using the agent-based model.

The results for this model were very static statistically. For all combinations of probability of reconnection and population size, the number of groups formed hardly varied as one can tell from *Graph 22*. However, the network configurations tell a different story. Again there is a change in the configuration when p=0.01 - the network stops so uniform and appears more like a randomly connected network. This composition then becomes denser as the probability gets higher. The output from the analysis with Matlab, figure 22, also points to some very interesting dynamics. With higher probabilities there are *indecisive* agents who cannot decide where to reside. As already suggested this is probably due to agents who have connections to both polar beliefs. If they have to change a friend on each time step then their belief value will probably exhibit a kind of oscillating behaviour depending on how many friends have certain belief at each time step.

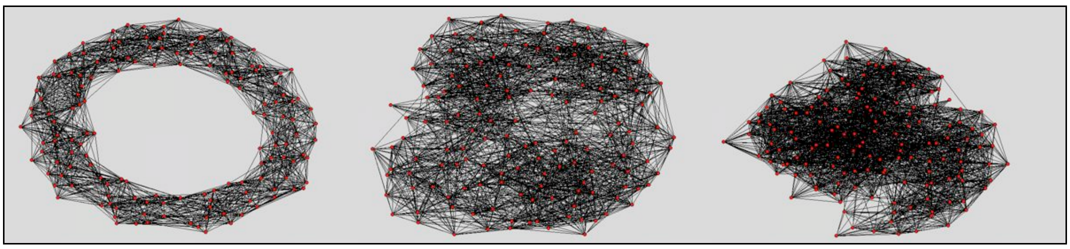


Figure 21: Final network configuration for an agent-based computational model. Networks shown for when p=0; p=0.01 and p=1.

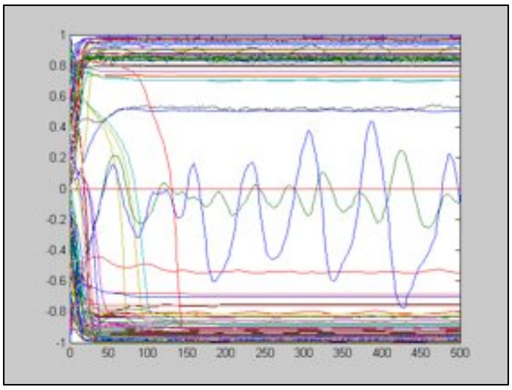


Figure 22: Matlab output for an agent-based model when p=1.