Chapter 5

Clough's Model

5.1 Chapter Overview

This chapter details the implementation of Clough's model [9] and its validation.

5.2 Implementation

Index	Answer
1	The agent calculates a prospective ranking of each candidate based on ideological closeness and the likelihood of the vote being impactful. The likelihood of the votes impact is inferred by treating their neighbouring agent's vote decision in the previous election as direct prediction of the election outcome.
2	Ideological closeness.
3	Share their vote decision from the previous election with each other.
4	The spatial network is a Torus.
5	3 (Clough experiments with 4 but we are only interested in 3.)
6	169

Table 5.1: A completed micro-specification template for Clough's model.

The model is reproduced within the framework detailed in Chapter 4. Therefore, its implementation is described by the content of its noteworthy objects. Please note that the term party is used interchangeably with the term candidate. Table 5.1 shows the completed micro-specification template for the model.

5.2.1 Party

A party has an ideology value, represented by an integer between 1 and 100 (inclusive). It is derived by random sampling of a uniform distribution. An agent also has an ideology value which is derived in the same manner.

5.2.2 Environment

The spatial network takes the form of a torus, as shown in Figure 5.1. This means each agent has an equal number of neighbours. Given an agent with co-ordinates x and y on a 2D representation with dimensions X x Y, the set of their immediate neighbour's co-ordinates N_A are given by:

$$N_A(x,y) = [[X_+, y], [x, Y_-], [X_-, y], [x, Y_+], [X_+, Y_+], [X_+, Y_-], [X_-, Y_-], [X_-, Y_+]] \tag{5.1}$$

where if $x - 1 = 0$ then $X_- = X - 1$, otherwise $X_- = x - 1$. Conversely, if $x + 1 = X - 1$ then $X_+ = 0$, otherwise $X_+ = x + 1$. Y_- and Y_+ follow the same pattern.

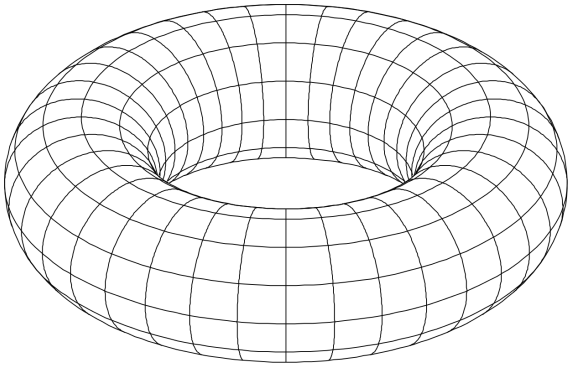

Figure 5.1: A torus network. (Image source: [17]; use permitted under the Creative Commons Attribution License CC BY-SA 3.0)

The parameter *level* determines the size of the neighbourhood. When its value is one the neighbourhood is equal to the immediate neighbours N_A and when it is incremented, the neighbourhood expands to include all agents adjacent to N_A. The maximum level is six, in which the neighbourhood is the entire population.

5.2.3 Agent

5.2.3.1 Utility

In order to place a strategic vote, the agent incorporates both their ideological preference of a candidate and the likelihood that a vote for them would not be wasted. As in Clough's model, an agent with an ideological value A_i calculates the utility (their ideological preference) of a candidate c with ideological value C_i as follows:

$$U_c = -(A_i - C_i)^2 \tag{5.2}$$

The idea is that the utility describes the ideological distance between an agent and a candidate. The exponent ensures that a distance either way is represented the same, for example an agent with an ideological value of 10 should assign the same utility to a candidate with an ideology of 15 and a candidate with an ideology of 5. The result is made negative because an increasing ideological distance should logically represent a decreasing utility.

5.2.3.2 Pivot Probability

Based on interaction with their sample neighbourhood, an agent calculates a pivot probability for each pair of candidates [10]. A pivot probability represents the likelihood that two candidates will tie for first place in the election. The implementation uses Black's method [4], in which a predicted election outcome, inferred directly from the neighbourhood sample, is plotted on a barycentric coordinate system shown in Figure 5.2. Each corner represents a candidate receiving the full vote share and the blue lines represent every outcome in which the candidates either side tie for first place. Perpendicular lines are placed between the blue lines and the predicted outcome point, and their euclidean distance subtracted from 1 represents the pivot probability for the respective candidate pair. The idea is that the ratio between the

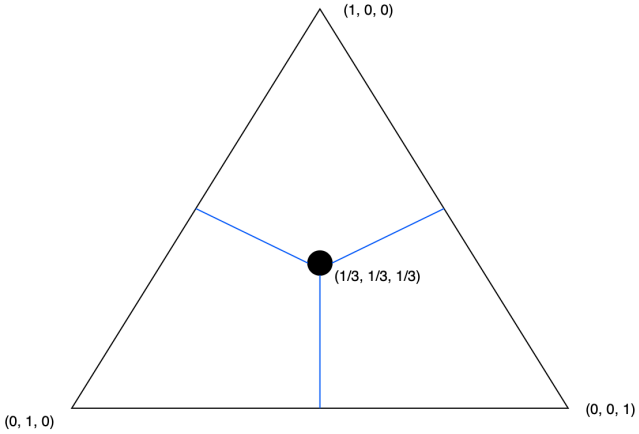

Figure 5.2: A barycentric coordinate system.

probabilities is more important than the exact values themselves, and that this method provides an easy method to represent the probabilities relative to each other.

5.2.3.3 Prospective Rating

An agent votes for the candidate with the highest prospective rating [18], which is calculated for a given candidate a as follows:

$$R_a = \sum_{c \in C} T_{ac}(U_a - U_c) \tag{5.3}$$

where C is the set of all candidates except a, T_{ac} is the pivot probability of the candidate pair and U represents the utility of a candidate. Combining the pivot probability with the utility means that an agent gives value to the potential impact of their vote.

5.3 Validating the Implementation

The simplest approach to validating the implementation is to see if it reproduces similar results. This is important, as it will indicate that the change in results of our Bayesian model are strictly due to the suggested modification. The emergent behaviour in question is the formation of a two-party system. This can be measured by the absolute number of parties present after some rounds of an election. Clough defines this as the number of parties with a vote share greater than 5%.

Figure 5.3 demonstrates that the implementation produces very similar results for three initial candidates. The important takeaway is that the plots share a similar shape and that the implementation reproduces the positive correlation between information completeness and 2-party system formation.

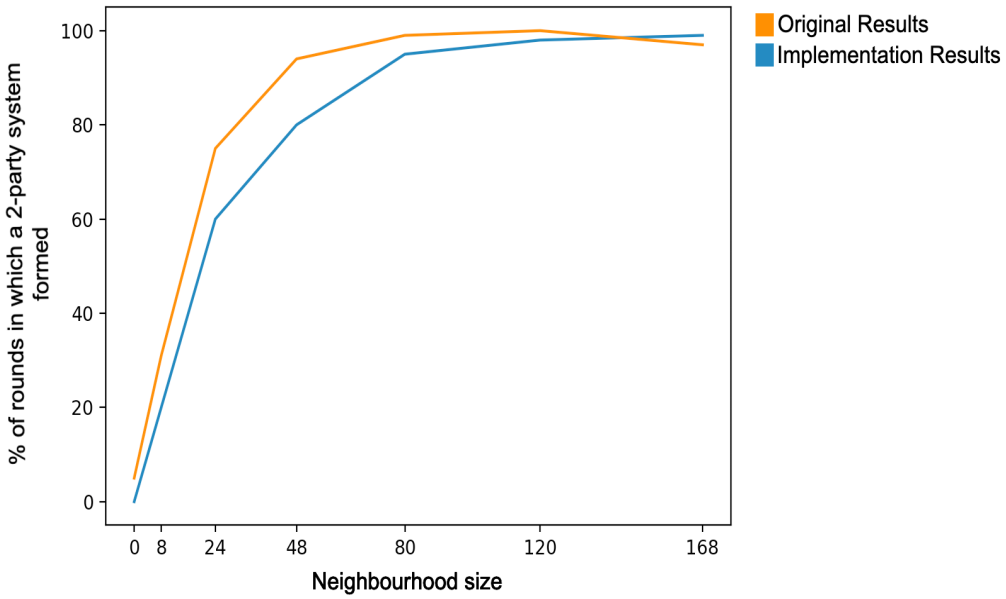

Figure 5.3: Results from the implementation for three initial candidates.

Chapter 6

The Bayesian Model

6.1 Chapter Overview

This chapter discusses the implementation and results of the Bayesian model.

6.2 Implementation

The key difference between the new model and Clough's [9] is how an agent infers pivot probabilities [10] from their neighbourhood sample (assumption index 1 from the micro-specification). Rather than treating the sample as directly representative of the vote share, the agent uses the sample to update its prior beliefs about the vote distribution. The prior distribution (beliefs) of an agent in this model defines them as optimistic about their favoured candidates chance of success in an election.

Because sample completeness and optimism are the key determinants to be explored with the model, it is assumed that the number of candidates is a fixed parameter. This enables the model to be hard-coded for three candidates.

6.2.1 Constructing the Prior Distribution

i	0	1	2	3	4
$P(x_i)$	0.2	0.2	0.2	0.3	0.3

Table 6.1: A naive optimistic distribution attempt.

The model's Bayesian inference uses a discrete variable [11] X, where a possible value x represents an agent's preferred candidate receiving x votes in an election. A naive approach to describing optimism would be to simply define a probability mass function that assigns each value of x that represents a majority of the vote share a higher probability than values representing half or a minority. An example of this is shown in Table 6.1. However, this would only be appropriate for elections with only two candidates, where the value of x represents a single event. Consider a three candidate election, $x = 3$ describes 2 possible election outcomes ([3, 1, 0] and [3, 0, 1]) whereas $x = 4$ describes only one ([4, 0, 0]). As a result, election outcomes that represent a victory would have varying probabilities which would increase with the size of the victory (because the number of events decreases as the size of the victory increases). This isn't particularly intuitive or rational. Additionally, victory outcomes in which the preferred candidate does not receive a majority of the vote share are ignored, for example [2, 1, 1]. The desired consistency and detail motivates building from the joint distribution of three candidate variables.

Assuming voters do not abstain, the optimistic agent's prior probability distribution of election

outcomes is described by the probability mass function of the candidate variables joint distribution:

$$P_{XYZ}(x, y, z) = \begin{cases} 0.6/W_E, & \text{if } x > y \ \& \ x > z \\ 0.4/(T_E - W_E), & \text{otherwise} \end{cases} \tag{6.1}$$

where T_E is the total number of possible election outcomes and W_E is the number of election outcomes in which the preferred candidate wins. The intention is that the agent views election outcomes in which their preferred candidate wins as more likely than others (due to their optimism).

Consider T_E as the number of ways in which T_P votes can be assigned to 3 candidates where T_P is the population size of the electorate. By the Balls and Urns technique [8]:

$$T_E = \binom{n}{2} \tag{6.2}$$

where $n = T_P + 2$. The Balls and Urns technique defines the number of ways in which n indistinguishable balls can be placed into k labelled urns as:

$$\binom{n + k - 1}{k - 1} \tag{6.3}$$

The number of solutions to an equation $a + b + c = d$ can be viewed similarly as the number of ways in which d 'ones' can be placed into each variable on the left hand side. Therefore, 6.3 describes the number of solutions to the equation where n is the number of left hand side variables and k is the sum of these variables (d). Given that an election outcome with no abstentions and three candidates can similarly be described as the equation $a + b + c = T_P$, it follows that T_E equals the number of possible election outcomes.

Calculating W_E, without inefficiently iterating through every possible election outcome, is non-trivial. As such, demonstrating the process in pseudo code provides a clearer picture of the solution than a mathematical approach.

```
1  x = T_P
2  spare = T_P - x
3  majorityCount = 0
4  while x > Ceil(spare / 2) do
5  |  outcomes = spare + 1 //number of outcomes for a given x
6  |  if x > spare then
7  |  |  majorityCount += outcomes
8  |  else
9  |  |  minorityCount = ((spare - x) + 1) * 2
10 |  |  majorityCount += outcomes - minorityCount
11 |  end
12 |  x -= 1
13 |  spare += 1
14 end
15 return majorityCount
```

Algorithm 1: Calculate W_E

Algorithm 1 counts every possible outcome in which a specific candidate wins the election. A possible outcome of the election can be described as $x + y + z = T_P$, where x, y and z represent the vote counts of different candidates. In this case, x represents the vote count of the candidate we are counting successful outcomes for. For a given vote count x, the remaining votes defined as *spare* on line 2, are shared between y and z such that $y + z = spare$. $Ceil(spare/2)$ gives the minimum values to be found in a pair y and z. It follows that a value of x equal to or less than $Ceil(spare/2)$ will be equal to or less than one of y or z, resulting in a loss or a draw. This is the intuition behind line 4, which ignores such outcomes from the counting process.

It is implied by $x > y + z$ that $x > y$ and $x > z$, meaning every election outcome that satisfies this is a victory, hence line 6. The number of outcomes for a given x is equal to the number of ways *spare* can be shared between y and z. While (6.2) could be utilised, the number of outcomes is simply $spare + 1$.

The set of possible outcomes for a given x can potentially include victories, losses and draws. This case is handled in lines 9 and 10. *spare* - x represents how many more votes are available for the other candidates. This means there are *spare* - x opportunities for a candidate to receive a larger vote count and (*spare* - x) + 1 opportunities for a candidate to receive a larger vote count or draw. Given that their are two other candidates, it follows that there are ((*spare* - x) + 1) * 2 possible outcomes in which a draw or loss occurs. Subtracting this from the number of outcomes given x gives the number of victorious outcomes.

6.2.2 Marginalising the Prior Distribution

Performing Bayesian Inference on every value of the joint distribution is inefficient and cannot be reasonably scaled for large values of T_P. A more desirable approach is to marginalise the

distribution to one of the univariate counterparts X, which is the discrete variable representing the vote count of the preferred candidate. This is described as follows:

$$P_X(x) = \sum_{x=0}^{T_P} \sum_{y=0}^{T_P - x} P_{XYZ}(X = x, Y = y, Z = (T_P - x) - y) \tag{6.4}$$

where P_{XYZ} is the PMF detailed by (6.1). Because $x + y + z$ must sum to T_P, there are no wasteful iterations through values that do not satisfy this.

6.2.3 Calculating the Posterior Distribution

Upon observing the vote decision of a neighbour from their sample, the agent performs a Bayesian update on the candidate variable X:

$$g(x|v) = \frac{g(x)\, f(v|x)}{\sum_{x \in X} g(x)\, f(y|x)} \tag{6.5}$$

for every value of X [11]. $g(x)$ represents the the probability of $X = x$ in the prior distribution. For the first update $g(x)$ is calculated with the marginalisation function described by (6.4). The resulting posterior distribution is used as the prior distribution in the next update. The process of treating the posterior from the previous update as the prior for the current one is continued for the rest of the Bayesian inference process. If a vote for a candidate other than the preferred one is observed, a variable representing its count is incremented.

v is a member of the observation variable V, which is equal to one if a vote for the candidate is observed and zero otherwise. $f(v|x)$ is the likelihood function, which returns the probability of observing $V = v$ given that $X = x$. This is defined:

$$f(v|x) = P_{V|X}(v|x) = \begin{cases} \mathrm{X}_S \ / \ R_S, & \text{if } v = 1 \\ (\mathrm{R}_S - X_S) \ / \ R_S, & \text{if } v = 0 \end{cases} \tag{6.6}$$

where R_S is the unobserved sample size including the current observation. X_S is the number of votes for the candidate remaining in the unobserved sample size given how many are in the observed sample and how many exist in the total sample (which is the value of x). Deriving the values of these variables is trivial and unworthy of discussion.

6.2.4 Calculating the Pivot Probabilities

The pivot probability of two candidates is the sum of the probabilities of election outcomes in which they tie for first place. Given the candidate vote counts observed, it is trivial to calculate election outcomes that are not possible. For example, if c represents the observed vote count of a candidate, election outcomes in which the candidate receives less votes than this are not possible. Additionally, it is not possible for the candidate to receive more votes than T_P - O_C where O_C is the count of observed votes that are not for the candidate. With these outcomes ruled out, the calculation of the remaining possible outcome probabilities is described by a joint distribution of three candidate variables representing their possible vote counts. However, the Bayesian inference process outputs only one of these variables (a univariate distribution). The

joint distribution cannot be inferred from the univariate alone, a conditional probability function is required.

$$P_{Y|X}(y|x) = 1/P_E \tag{6.7}$$

where P_E is the number of possible election outcomes in which the preferred candidate receives x votes. This is calculated by subtracting all the impossible election outcomes in which the preferred candidate receives x votes given the candidate vote count variables, from the total election outcomes that were initially possible before the votes were observed. These impossible outcomes are inferred from the observed votes in the manner described in the previous paragraph.

The intuition behind (6.7) is that when a vote decision is observed from the neighbourhood sample, it always rules out the possibility of some value of the candidate variables. The current probability of that value is absorbed by the rest of the distribution, which is reflected by subtracting from the denominator in the conditional probability function.

It is worth noting that the conditional probability function (6.7) does not completely reflect the bias described in the PMF detailed by (6.1). Consequently, cases in which $x < y + z \,\&\, x > y \,\&\, x > z$ are not exactly true to the probability values that would output from Bayesian inference performed on the joint distribution. However, implementing a conditional probability function that is considerate of such cases is overly complex for the requirements of the model, and considering the bias present in the initial prior construction has noteworthy impact on the results of the model, it is wise to deem such an endeavour to be outside of the scope of the project. Additionally, the intention of the initial joint distribution was to construct a bias discrete variable required for the Bayesian inference, there is no harm in deriving a new form of joint distribution from the output of the process.

The conditional probability function can be combined with the univariate distribution X to produce the joint distribution:

$$P_{XYZ}(x, y, z) = P_{XY}(x, y) = P_X(x)P_{Y|X}(y|x) \tag{6.8}$$

where $x + y + z = T_P$. Note that the joint distribution of two variables is equivalent to the joint distribution of three as the values of two variables imply the remaining value. This is used in the calculation of the pivot probabilities:

$$\sum_{i=\lceil T_P/3 \rceil}^{\lfloor T_P/2 \rfloor} P_{XY}(i, i) \tag{6.9}$$

6.3 Profiling the Implementation

While the model described may be theoretically sound, it is imperative that its software implementation executes in a reasonably timely manner. cProfile, a Python library for profiling

code execution, is a useful tool for achieving this. Although 'reasonable' is a criteria that lacks objectivity, this sections aims to sensibly justify the decisions made based on the analysis of the execution profile.

6.3.1 Initial Profile

Process	**Calls**	**Time (s)**	**Accumulative Time (s)**
Election	5	0	11.657
Pivot Probabilities (x)	845	0.004	11.657
Event Probability (x)	215475	5.708	5.708
Marginalise Distribution (x)	845	2.684	4.523
Optimist PMF (x)	12138425	1.827	1.827

Table 6.2: Profile of run with a neighbourhood size of 8. Before optimisation.

Process	**Calls**	**Time (s)**	**Accumulative Time (s)**
Election	5	0	138.944
Pivot Probabilities (x)	845	0.266	138.944
Bayesian Step (x)	215475	10.269	133.214
Likelihood (x)	23991240	114.377	114.377

Table 6.3: Profile of run with a neighbourhood size of 168. Before optimisation.

Table 6.2 and 6.3 show the most significant profile results from a run of the model, with the key parameters being the number of elections and the neighbourhood sizes. These represent the required scope of the project. Five elections is chosen because it is when the results of the model stabilise (vote decisions are not changed), therefore representing the maximum execution times. The neighbourhood sizes of the tables represent the minimum and maximum of Clough's model [9], and subsequently the Bayesian one.

The Time column is the run time of a process excluding child processes, whereas the Accumulative Time column includes child processes. It follows that processes with large values in the Time column should be the targets for optimisation. Where possible, the easiest method is to simply reduce the number of calls to such processes.

6.3.2 Marginalisation

The output of the marginalisation process, along with the Optimist PMF which it calls, is entirely dependent on the number of agents (voters) in the model. Given that this variable is the same for each agent, it follows that it only needs to be performed once. While this could be achieved by passing the process output to each agent as a parameter, this would somewhat violate the notion of encapsulation, which is inherent to the spirit of the project. The model defines an agent by its prior distribution, therefore any functionality pertaining to its construction should remain within the Agent object. This philosophy has the practical benefit of reducing dependencies across objects.

Fortunately, the execution time of the Marginalisation process, as shown by its exclusion from

Table 6.3, does not grow with the neighbourhood size. Given that this is the significant problem size of the model, it is fair to halt further optimisation considerations.

6.3.3 Calculating an Event Probability

Calculating an event probability is interesting because its execution time has a negative relationship with the neighbourhood size. If passed a vote count that is not possible given the observed sample, it returns zero instead of performing any calculations. For example, if five votes for a candidate have been observed, and the process is asked to calculate the probability of an election outcome where the vote count for the candidate is 4, it requires negligible time to infer that such an outcome is not possible. Given that the size of the neighbourhood equates to the size of the observed sample, and that a larger observed sample causes more runs of the process that execute faster, it is clear that a larger neighbourhood size decreases the accumulative execution time of the process.

The size of the electorate population correlates to the execution time and number of calls to the process. As discussed in the previous section, this variable is controlled for the scope of the project, and is therefore not worthy of further consideration.

6.3.4 Performing a Step of Bayesian Inference

As expected the number of calls to Bayesian Step grew significantly with the neighbourhood, however the accumulative execution time was concerning. While its child process Likelihood was being invoked a large amount of times, its execution time was still unsatisfactory. Upon investigation, it was discovered that part of calculating the value of X_s from (6.6) was achieved by counting through every vote in the observed sample. This means that for a neighbourhood of size n, the number of unnecessary iterations is given by:

$$\sum_{i=0}^{N-1} i \tag{6.10}$$

The iterations are labelled unnecessary because the vote counts could be stored and updated every time there is an observation.

After the optimisation was implemented, the execution time of the Likelihood saw a large reduction from 114 to 9 seconds. However, there was a noticeable disparity between it and the accumulative execution time of the Bayesian Step. Given that there were a large number of calls to the Likelihood process, it appeared that this disparity reflected the sum of the invocation overheads. As a result, the Likelihood process was removed from a function and executed within the Bayesian Step one. This halved the accumulative execution time of the Bayesian Step.