Benchmarking Equilibrium Expectation as an Estimation Method for Exponential Random Graph Models

Youhan Cheery
Pavel Krivitsky, Supervisor
School of Mathematics and Statistics, UNSW Sydney

August 2021
Submitted in partial fulfilment of the requirements of the degree of Masters of Statistics

Contents

List of Tables

List of Figures

Abstract

Exponential Random Graph Models (ERGMs) are a family of network models that are useful in modelling complex network structures, such as corporate connections and social relations. ERGMs benefit from many of the properties that make their namesake, the exponential family, such a powerful device in statistical literature and practice. However, despite the benefits from being exponential family models, it is computationally challenging to estimate the parameters that define an ERGM model. Several methods have been devised to estimate parameters of a network model accurately and within computational constraints which will be explored in this work.

The challenge of parameter estimation is largely due to the normalising constant that is core to the definition of an ERGM often making computation intractable. As such, initial methods approximated the likelihood in order to estimate parameters, an approach known as Maximum Pseudo Likelihood Estimation (MPLE) (Strauss and Ikeda (1990)), while more current approaches leverage improvements in computation power and aim to reach the maximum likelihood estimate using Monte Carlo simulation.

Within the class of Monte Carlo estimation methods, a recent approach, Equilibrium Expectation (EE) (Byshkin et al. (2018)), proposes an a fast and scalable means to compute the parameters of large networks, a typically challenging computational problem. By using Markov Chain properties at equilibrium it is able to make infrequent, albeit complex updates to the parameter values to reduce computation effort required. It follows on from the works of Robbins and Monro and Snijders in the form of Stochastic Approximation, as well as Geyer and Thompson and Hunter and Handock respectively in the mode of Markov Chain Maximum Likelihood Estimation (MCMLE) which have been staples in the estimation of ERGM parameters for a number of years. This paper will look to benchmark the Equilibrium Expectation algorithm against these two methods by comparing properties across some of the more challenging network datasets covered in Hummel et al. (2012), namely the famous open source network datasets "E. Coli" and "Kapferer".

This report is structured as follows: the succeeding section introduces ERGMs and the challenge at hand in more detail, followed by a literature review spanning Stochastic Approximation estimation by Snijder as an extension to the Robbins-Monro algorithm, Markov Chain Monte Carlo, and lastly a summary of the Equilibrium Expectation method. Post-literature review this report will then dive into both the method and the results of the analysis in sections 3 and 4 respectively.

1 Introduction

Exponential family models find a wide variety of use: from social networks, to spatial statistics, to even image analysis. Exponential Random Graph Models (ERGMs) take the form:

$$\pi(x, \boldsymbol{\theta}) = \frac{1}{k(\boldsymbol{\theta})} \exp\left(\sum_A \theta_A z_A(x)\right) \tag{1}$$

Where $z_A(x)$ are the statistics that define the network, θ_A are the parameters of the network, and $k(\boldsymbol{\theta})$ is the normalising constant.

Parameter estimation methods have required some ingenuity to traverse the normalising constant $k(\boldsymbol{\theta})$ often being an intractable task. Strauss and Ikeda in 1991 confronted this challenge by approximating the maximum likelihood form with an approximated form, or a 'pseudo maximum likelihood' (MPLE). Using traditional maximum likelihood calculations Strauss and Ikeda then estimate the underlying parameter values. This method had drawbacks of its own: as it's namesake implies, the estimate was calculated for an approximation of the maximum likelihood equation of the network, and not the true likelihood itself. This meant that the possibility of erroneously selecting parameters that defined the observed network came with additional uncertainty for the modeller.

With the advent of computation storage and power, Markov Chain approaches became viable ways of reaching the solution to optimisation problems iteratively. These methods used sampling techniques to gradually direct the model to the maximum likelihood estimate at the global minima. Historically speaking, this began with Roberts and Munro in 1951 with the Stochastic Approximation method that sought to iteratively reach the parameters by the Newton-Raphson method. The relevance of Robberts and Munro's methods were extended to ERGMs by Sneijders in 2002, and onwards to Equilibrium Expectation in 2018 by Stivala et. al. Other methods such as Geyer and Thompson's importance sampling Markov Chain Maximum Likelihood Estimation (MCMLE) approach also sought to leverage the benefits of re-sampling with Markov properties.

In this work, we implement the equilibrium expectation algorithm (Byshkin et al. (2018)) in the R package `ergm` (Hunter et al. (2008)) and benchmark the implementation to ERGM parameter estimation and compare it to the existing Monte Carlo Maximum Likelihood approaches of Stochastic Approximation (Snijders (2002)) and Markov Chain Monte Carlo (Geyer and Thompson (1992)). The implementation of the equilibrium expectation algorithm is important in this case, given that the methods used for comparison are themselves being run from the ERGM R package. Finally, we compare estimates using each method, using the *E. coli* transcriptional regulation network, and *Kapferers* sociational dataset as illustrated by Hummel et al. (2012) in his own analysis of yet another estimation method.

We consider the computational tractability of the MCMC-based approaches by considering three different starting points: starting from all parameters being set to zero, from the MPLE, and from a mostly zero starting point sans having values at the edges of the network pre-calculated with a run of `ergm` at default settings. We then observe traceplots of the Markov Chain and observe time differences for computation time in 4 of this paper.

2 Literature Review

In order to benchmark the computational and accuracy of Equilibrium Expectation as an ERGM estimation method we use the Stochastic Approximation and Markov Chain Maximum Likelihood Estimator as the basis. The aforementioned estimation approaches are detailed in this section, focusing largely on the philosophy of each algorithm reaching meaningful estimates.

We specifically note that though these techniques sought the "actual" MLE, it is worth beginning our discussion with the Maximum Pseudo Likelihood Estimation (MPLE) method (Strauss and Ikeda (1990)) as we use this method as one of the starting points in our benchmarking analysis.

2.1 Maximum Pseudo-Likelihood Estimation and choice of starting point

Due to the normalising constant referred to in Section 1 being a summation over parameters of all possible graphs, estimating the true parameters of the ERGM is often an intractable problem. Strauss and Ikeda proposed an approach where calculating the maximum likelihood function wasn't done on the direct likelihood but rather on a 'pseudo' likelihood which approximated the true underlying model.

For a dyad independent network model, that is, a model where every dyad (or pair of ties) is independent from the next - maximum likelihood can be calculated. While this assumption simplifies the mathematics of estimation, it limits the practicality of the network model greatly. For dyad dependent networks, Strauss and Ikeda proposed the pseudo-likelihood as the product of the probabilities of the y_{ij}, with each of the probabilities conditional on the rest of the data Strauss and Ikeda (1990).

When not conditioning on data, the ERGM takes the log linear form

$$Pr(G) = \frac{1}{Z(\theta)} e^{\theta' x(G)} \tag{2}$$

where θ is a vector of parameters and x(G) is a vector of graph statistics (on the observed graph).

Conditioning on the rest of the graph produces a model form that now does not depend on θ:

$$\begin{aligned} P(y_{ij} = 1|C) &= \frac{Pr(G^-)}{Pr(G^-) + Pr(G^+)} \\ logit(Pr(y_{ij} = 1|C) &= \theta' x(G^+) - x(G^-) \\ logit(Pr(y_{ij} = 1|C) &= \theta' \delta x_{ij} \end{aligned} \tag{3}$$

2.2 Methods for Parameter Estimation

2.2.1 Monte Carlo Maximum Likelihood Estimation (MCMLE)

The Monte Carlo Maximuim Likelihood Estimate was first introduced by Geyer and Thompson (1992), and subsequently extended to curved ERGMs by Hunter and Handcock (2006). MCMLE is a method developed as a consequence to the fact that the Laplace transformations (or normalising constants as is often used in this paper) for exponential family models with dependent data cannot be exactly calculated, and approximations are difficult to find. As such, the general idea comes down to translating from the intractable integral, to a probability distribution such that Monte Carlo methods become applicable, as shown with the method of moments equation below:

$$M_\theta(t) = \int exp\langle t(x), \tau \rangle dP_\theta(x) = \frac{c(\theta + t)}{c(\theta)} \tag{4}$$

Where $t(x)$ are canonical statistics, and P_θ denoting the measure having density f with respect to μ.

The above refers to the "Monte Carlo" side of the eponymous section. Standard Gibbs or Metropolis sampling methods are used to generate an ergodic Markov chain X_1, X_2, ... having equilibrium distribution P_θ. This leads to the observation that

$$d_n(\theta') = \frac{1}{n}\Sigma_{i=1}^{n} exp\langle t(x_i), \theta' - \theta\rangle \rightarrow d(\theta) = \frac{c(\theta')}{c(\theta)} \tag{5}$$

almost surely by the ergodic theorem. For a given observation x the log likelihood can be written as

$$I_x(\theta) = logf_\theta(x) + logc(\theta) = \langle t(x), \theta\rangle - logd(\theta) \tag{6}$$

which has a Monte Carlo approximation given by

$$I_{n,x}(\theta) = \langle t(x), \theta\rangle - logd_n(\theta) \tag{7}$$

We see that Geyer and Thompson consider the log-likelihood ratio $I_{n,x}(\theta)$ at a fixed parameter choice rather than maximising the log-likelihood alone. Then, for any fixed θ,

$$I_{n,x}(\theta) \rightarrow I_x(\theta) \tag{8}$$

almost surely by expression 5. Thus, under the concavity of $I_{n,x}(\theta)$ and $I_x(\theta)$ (a consequence of the exponential nature of the objective function in this case, though not a not necessary condition for the expression above to be true) we have that

$$\hat{\theta}_n \rightarrow \hat{\theta}\text{almost surely} \tag{9}$$

Using simulations from one distribution P_θ we are able to approximate $\hat{\theta}$ regardless of the intractability of the normalising constant. The algorithm has a large dependence on initialising with a reasonable starting point (i.e. something within the realm of reality), which sometimes means requiring a (sometimes infeasibly) large number of Monte Carlo samples. Hunter et al. (2008) demonstrates the sensitivity to starting estimates by applying the MCMLE estimation method to the Sampson Monastery dataset, a dataset with 18 actors and 88 out of 306 possible ties. With a model based on edge count, and a starting point θ^0 of 1, Hunter et al. shows that the model struggles to obtain the MLE in a single iteration, which wouldn't have been the case with a more reasonable starting estimate closer to the (already known) MLE.

In addition, if the first derivative of the log-likelihood ratio in expression 7 with respect to θ is negative (i.e. $g(Y) < g(Y_{obs})$) then 7 will not have a maximum. Again, with a better starting estimate, this isn't likely to be the case but illustrates the sensitivity to challenging starting point estimates.

In his 2006 paper, Hancock applies the methods discussed in Geyer and Thompson (1992):

The MLE $\hat{\boldsymbol{\theta}}$ satisfies the equation

$$\nabla l(\hat{\theta}) = \nabla\eta(\hat{\theta})^t[Z(y_{obs}) - E_{\eta(\hat{theta})}Z(Y) = 0 \tag{10}$$

Where $\nabla\eta(\hat{\theta})$ is a matrix of partial derivatives of η with respect to θ.

Noting that the exponential family form of the models makes the Fisher information matrix

$$I(\theta) = \nabla\eta(\theta)^t[var_{\nabla\eta(\theta)}Z(Y)]\nabla\eta(\theta) \tag{11}$$

It is easier to calculate than the Hessian matrix of second derivatives as in the Newton-Raphson method as used by Stochastic Approximation. The, we calculate iterations of the parameter θ as

$$\theta^{k+1} = \theta^{(k)} + [I(\theta^(k))]^{-1} \nabla l(\theta^{(k)}) \tag{12}$$

The MCMC-MLE is theoretically expected to converge to the maximum likelihood assuming it exists.

2.2.2 Robbins-Monro/ Stochastic Approximation

Stochastic Approximation methods were among the first approaches to estimating the ERGM parameters by finding the actual MLE. First introduced by Robbins and Monro in 1951, the stochastic approximation method aims to iteratively find the roots of an optimisation function as represented by an expected value. It does so by iteratively solving equations of the form

$$E(Z_\theta) = 0 \tag{13}$$

Where Z_θ are the observed network statistics under the unknown θ.

For a root proposed root $\hat{\theta}$ the Robbins and Monro algorithms states that $\lim_{\hat{\theta_n} \to \inf} = \theta$, the true root. This is done by iterating through

$$\theta_{n+1} = \theta_n - \alpha_n (N(\theta) - \alpha) \tag{14}$$

In the context of ERGMs, this is ...

A Markov chain is a sequence of random variables such that the value taken by the random variable only depends upon the value taken by the previous variable. We can hence consider a network in the form of an adjacency matrix in which each entry is a random variable. By switching the values of these variables to 0 or to 1(adding or removing a link from the network) one can generate a sequence of graphs such that each graph only depends upon the previous graph. This would be a Markov chain. The hypothesis is then that if the value at step t is drawn from the correct distribution than so will the value at step t+1. Unlike regular Monte-Carlo methods, the observations that are sampled are close to each-other since they vary by a single link. However, one would need a method for selecting which variable should change state in order to get closer to the MLE, this is done using the Metropolis-Hastings algorithm or the Gibbs sampler

Snijders introduced the Stochastic Approximation as an extension of the original Robbins-Monro algorithm in his paper Snijders (2001). The algorithm in his subsequent paper (Snijders (2002)) takes the ideas proposed for this *stochastic actor oriented model* and uses a simplification due to the scaling matrix used in the first phase (itself a matrix of derivatives) can be estimated using the covariance matrix of the generated statistics rather than by a finite quotient difference.

Due to the normalising constant in ERGMs $k\boldsymbol{\theta}$ typically being unknown, the means of estimating the probability distribution remains an intractable problem. And while stochastic approximation can be used to estimate the parameters, the rise of computing power has meant Markov Chain Monte Carlo MCMC simulation is typically used to address the value of $k\boldsymbol{\theta}$ (and the MLE as a whole).

Using MCMC estimation typically via the Metropolis-Hastings algorithm estimates network parameters by using the Markov process that asymptotically reaches a unique stationary distribution. A new state, x' is proposed with some probability given by $q(x \to x'$.

Snijder's algorithm consists of three main phases, with each phase adopting the Metropolis-Hastings sampling approach to draw from the model. Before diving into each of the phases, the developer has control on two parameters at this stage: the burn-in, and the gain factor. The burn in is a MCMC-specific hyperparameter and needs to be set for any MCMC-based method, while the gain factor controls the size of the steps the algorithm takes in reaching

the MLE. A large gain factor means reaching the general vicinity of the MLE quicker, albeit with less 'precise' movements.

Phase 1: Initialisation

The goal of Stochastic Approximation's first phase is to determine the scaling matrix D_0 and the initial values of the parameters being estimated using a small number of steps. The scaling matrix is used to scale the updates of the different elements of the parameter vector. At the end of the first phase, there is an optional initiation of the Newton-Raphson process (used in the subsequent step), $\hat{\theta}^{(N_1)} = \theta^{(1)} - \alpha_1 D^{-1}(\bar{u} - u_0)$ where the values D and $\bar{u}$ are respectively defined as

Phase 2: Optimisation

The second phase is the main phase of Stochastic Approximation, with a number of subphases each containing several iterations used to calculate $Y(n)$ according to the current parameter value $\hat{\theta}^{(n)}$. After of these minor iterations, the θ is updated according to the equation

$$\hat{\theta}^{(n+1)} = \hat{\theta}^{(n)} - \alpha_n D_0^{-1} Z(n) \tag{15}$$

Where α_n is a constant consistent through each subphase, and $Z_k(n) = P(n)u(1 - Y(n)) + (1 - P(n))u(Y(n)) - u_0$. At the end of each of these subphases, the algorithm estimates a new value for $\theta^{(n)}$, with the final estimate, $\hat{\theta}$, being the average of the $\theta^{(n)}$ estimated at the final subphase.

Note that with each subphase, the value of α_n is halved, so that the steps become incrementally smaller with more iterations (i.e. the algorithm makes smaller, more 'careful' steps as it reaches the final $\hat{\theta}$).

Phase 3: Conversion check and measurement calculation

As the previous phase calculated the final value for $\hat{\theta}$, the final phase of Stochastic Approximation seeks to estimate the covariance matrix of the estimator and $\Sigma(\theta)$.

2.2.3 Equilibrium Expectation

The focus of this paper, the equilibrium expectation algorithm by Byshkin et al. (2018) proposes a fast algorithm for exponential random graph model parameters using maximum likelihood estimation which in turn affords an increase in the scale of the network being estimated. The fast and scalable nature of the equilibrium expectation algorithm are a consequence of the namesake: relying on properties of Markov chains at equilibrium.

As in the prior methods, estimation of the ERGM parameters is obtained from the method of moments equation

$$E_{\pi(\theta)}(z_A(x)) = z_A(x_{obs}) \tag{16}$$

where $z_A(x)$ are the network statistics and x_{obs} is the observed network.

As before, where the distribution of the normalising constant, $k(\theta)$, is not known, Markov chain Monte Carlo simulation is applied. Using MCMC simulation, we can compute the target probability given by expression 1 as well as the expected properties of the model, $E_{\pi(\theta)}(z_A(x)) = \Sigma_A Z_A(x)\pi(x, \theta)$.

Using Metropolis-Hastings or other sampling methods (such as Gibbs) the algorithm asymptotically reaches a unique stationary distribution, $\pi(\theta)a$. After MCMC burnin, and the crux of the algorithm (deciding on a next step based on transition probabilities), the MCMC process eventually leads to equilibrium stationary distribution for all statistics $z_A(x)$

$$\Sigma_{x,x'}\pi(x, \theta)P(x \to x', \theta)(z_A(x') - z_A(x)) = 0 \tag{17}$$

While in other MCMC approaches, whenever a new $x_s(\theta)$ is drawn, the convergence criterion given in equation 17 must be satisfied, in EE the convergence criterion is rewritten:

$$E_{\pi(\theta)}(\Delta z_A(x,\theta)) = 0 \tag{18}$$

That is, if the network x is drawn from the stationary distribution $\pi(x,\theta)$ then the expected value in the change statistics $\Delta z_A(x,\theta)$ is zero. In doing so, the much more computationally expensive 17 is reduced to a 'cheaper' process, noting that 18 is only valid at equilibrium.

On the assumption that a number of simulated networks exists, the left hand side of the stationary distribution **??** can be calculated by Monte Carlo integration

Typically we estimate the value of θ^* by averaging the result of each estimated θ_{s_i} of an observed network x_{s_i}, a la $\hat{\theta*} = \frac{1}{n}\Sigma_i\theta(x_s)$. If the network x_s is very large, then by the ergodicity of systems we can estimate $\theta*$ using $f_A(x_{obs},\theta) = 0$ (**?**), dropping the summation necessitated by MC integration. Thus, the true estimate for $\theta*$ can be found from the equilibrium expectation condition for all A,

$$\Delta z_A(x_s,\theta^{EE}) = 0 \tag{19}$$

where the change statistic can be found by $\Delta z_a(x,\theta) = \Sigma'_x P(x \to x',\theta)(z_A(x') - z_A(x))$. *If* the observed network x_s is very large, then $\theta^{EE}(x_s)$ is the desired estimate for $\theta*$. Otherwise we require a large sample of networks to calculate the summation terms in the above expressions.

Of course, in reality we only have one x_{obs} drawn from some unknown probability distribution, $\pi*(x)$. Networks can be drawn from this unknown distribution by MCMC simulations, meaning we can then iteratively adjust θ until expression 19 is satisfied. In contrast to methods described above (and others not described), the EE algorithm does not draw as many simulated networks for various θ values, making it noticeably quicker. A root finding method is still required once we have reached θ^{EE} that satisfies expression 19 such as stochastic approximation methods (which we indeed use in the implementation of EE in `ergm` for the purpose of benchmarking).

3 Method

Before considering analysis methodology, for an equivalent comparison of Equilibrium Expectation to other estimation methods, we must first implement Stivala's algorithm within the Statnet `ergm` package by Hunter et al. (2008).

The methods we will be benchmarking against are mentioned in Section 1 and detailed in Section 2. Specifically, we use the models used by Hummel et al. (2012) to compare the effectiveness of Equilibrium Expectation against MCMLE and Stochastic-Approximation as implemented in `ergm` under different combinations of starting points, while controlling for other hyperparameters such as burn-in. The datasets we use to compare estimation algorithms are the E. Coli dataset for protein location sites, as well as Kapferer's Zambian tailor shop. For more detail on the datasets refer to Appendix A and B respectively.

For the E. Coli network dataset we consider four models as implemented by Hummels, as well as Krivitsky (2017):

```
ecoli2 ~ edges + degree(2:5) + gwdegree(0.25, fixed = TRUE)
```

The first model considers a count of the edges as well as a count of actors with degrees two through 5 (individually), and the geometrically weighted degree with a fixed decay of 0.25.

```
ecoli2 ~ edges + degree(2:5) + gwdegree(0.25, fixed = TRUE) + nodemix("self"
   ↪ , base = 1)
```

Model 3 is a superset of model 3 but with an additional term for nodal attribute mixing with the edge.

```
kapferer ~ edges + gwesp(0.25, fixed = TRUE) + gwdsp(0.25, fixed = TRUE)
```

Model 3 has parameters: count of edges, `gwesp` (geometrically weighted edgewise shared partnerships) which models triadic closure with a decay of 0.25, and `gwdsp` is a count of shared partnerships regardless of whether actors i and j have a tie with a decay parameter fixed at 0.25.

```
kapferer ~ edges + gwdegree(0.25, fixed = TRUE) + gwesp(0.25, fixed = TRUE)
   ↪ + gwdsp(0.25, fixed = TRUE)
```

Finally, model 3 contains all of model 3 with an additional parameter to count the geometrically weighted degrees (`gwdegree`) decayed at a rate of 0.25.

The terms `gwdegree`, `gwesp`, and `gwdsp` are from Hunter (2007) and explained in more detail in Appendix C

For each of the network, we then considered three different starting point configurations across each of the three aforementioned estimation methods:

1. Starting from a parameter vector of all zeros, i.e. $\boldsymbol{\theta_0} = \mathbf{0}$.
2. Starting from the maximum pseudo-likelihood estimate, $\boldsymbol{\theta_0} = \boldsymbol{\theta_{MPLE}}$
3. Starting from a parameter vector of mostly zeros, except for the edges of the network which were calculated beforehand with a run of the network at the edges only, $\boldsymbol{\theta_0} = \boldsymbol{\theta_{edges}}$. The R package `ergm` was used to calculate the edges using a quick run of MCMLE.

Due to the nature of each estimation algorithm, the hypothesis is that differences in starting points not only may lead to different points of convergence, but the time taken to convergence may also be different. While Stochastic Approximation and Equilibrium Expectation are similar algorithmically, the steps taken to reach the MLE are noticeably different. Stochastic Approximation takes larger steps, while EE squares the change in calculated statistics leading to shorter, more precise steps. We expect that MCMLE will have a somewhat more challenging time with the starting points at zero and mostly zero sans the edges of the observed network, but should be able to converge.

4 Results

We first look at the runtime differences of the different approaches based on different starting points, under the datasets briefly mentioned in Section 3 and detailed in Appendices A and B. Each subsection to follow will also contain the parameter estimates and a measure of the difference between the method and starting point combinations.

4.1 Model Set 1: E. Coli With No Mixing

Equilibrium Expectation (EE)

Starting Point	User	System	Elapsed
zeros	21.052	0.025	21.099
zeros and edges	6.763	0.004	6.777
MPLE	17.863	0.027	17.897

Table 1: E. Coli dataset run-times with various starting point configurations using EE as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg.fixed.0.25
Zeros	-5.737519	-3.323081	-4.305612	-3.518558	-2.636180	2.9922375
Zeros and edges	-4.539951	-2.110865	-3.098017	-3.209092	-3.681314	1.7391186
MPLE	15.534803	32.362031	32.199131	20.771601	12.224798	-196.0182076

Table 2: E. Coli parameter estimates with various starting points using EE as the estimation method

Stochastic Approximation (SA)

Starting Point	User	System	Elapsed
zeros	21.134	0.087	21.795
zeros and edges	896.263	0.264	897.086
MPLE	17.572	0.027	17.610

Table 3: E. Coli dataset run-times with various starting point configurations using SA as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg.fixed.0.25
Zeros	-5.737519	-3.323081	-4.305612	-3.518558	-2.636180	2.9922375
Zeros and edges	-4.539951	-2.110865	-3.098017	-3.209092	-3.681314	1.7391186
MPLE	15.534803	32.362031	32.199131	20.771601	12.224798	-196.0182076

Table 4: E. Coli parameter estimates with various starting points using SA as the estimation method

Markov Chain Maximum Likelihood Estimator (MCMLE)

Starting Point	User	System	Elapsed
zeros	301.087	3.712	373.466
zeros and edges	46.978	3.267	373.466
MPLE	43.854	3.185	125.712

Table 5: E. Coli dataset run-times with various starting point configurations using MCMLE as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg.fixed.0.25
Zeros	-1.680262	497.776421	189.629113	130.370015	35.555459	-446.7001373
Zeros and edges	-5.111108	-1.402672	-2.505505	-2.271429	-3.267121	1.5884273
MPLE	-4.981500	-1.397047	-2.246059	-2.009961	-1.587440	0.8127727

Table 6: E. Coli parameter estimates with various starting points using MCMLE as the estimation method

4.2 Model Set 2: E. Coli With Self Mixing

Equilibrium Expectation (EE)

Starting Point	User	System	Elapsed
Zeros	22.007	0.034	22.041
Zeros and edges	8.160	0.002	8.155
MPLE	19.046	0.043	19.098

Table 7: E. Coli dataset with self-loop run times with various starting point configurations and EE as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg0.25	Mix False	Mix True
Zeros	-10.716486	-6.594619	-6.689404	-7.215315	-4.544823	10.6444020	11.29079006	35.7647417
Zeros and edges	-5.852937	-1.927183	-2.109867	-2.128186	-2.384952	2.4482905	1.83567510	1.8338598
MPLE	2.224877	14.058709	13.586779	8.812083	5.728130	-82.8397531	-1.85029310	-4.1102634

Table 8: E. Coli parameter estimates with various starting points using EE as the estimation method

Stochastic Approximation (SA)

Starting Point	User	System	Elapsed
Zeros	22.128	0.113	22.460
Zeros and edges	932.005	0.493	1215.180
MPLE	18.799	0.094	18.933

Table 9: E. Coli dataset with self-loop run times with various starting point configurations and SA as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg0.25	Mix False	Mix True
Zeros	-6.243631	-5.642640	-5.536328	-4.552708	-1.396533	0.6543856	10.07940661	1.4683291
Zeros and edges	-5.816048	-1.309330	-2.025997	-1.777584	-2.264008	2.0434754	1.55321296	1.1751994
MPLE	3.190518	12.771154	9.827869	8.806080	1.497652	-77.5304319	-1.15712833	-7.7680711

Table 10: E. Coli parameter estimates with various starting points using SA as the estimation method

Markov Chain Maximum Likelihood Estimator (MCMLE)

Starting Point	User	System	Elapsed
Zeros	361.232	2.940	366.182
Zeros and edges	32.287	1.438	33.682
MPLE	22.193	0.835	22.816

Table 11: E. Coli dataset with self-loop run times with various starting point configurations and MCMLE as the estimation method

Starting point	Edges	Degree2	Degree3	Degree4	Degree5	Gwdeg0.25	Mix False	Mix True
Zeros	-1.643217	591.371338	225.284319	154.882969	42.240810	-530.6913835	0.07455501	-0.2704657
Zeros and edges	-5.825112	-1.361900	-2.035242	-1.781577	-2.323788	2.3079946	1.53632447	1.1967021
MPLE	-5.849533	-1.356081	-2.012842	-1.755260	-2.330215	2.3329175	1.56200515	1.2046862

Table 12: E. Coli parameter estimates with various starting points using MCMLE as the estimation method

Starting Point	edges	gwdeg.fixed.0.25	gwesp.fixed.0.25	gwdsp.fixed.0.25
Zeros	-4.29	-9.74	3.41	-1.75
Zeros and edges	-1.05	-45.09	0.60	-0.28
MPLE	-2.95	-0.17	1.39	-0.12

Table 17: Parameters of model set 3 on Kapferer data, with EE as the estimation method across various starting configurations

4.3 Model Set 3: Kapferer With No GWDSP Parameter

Equilibrium Expectation

Starting Point	User	System	Elapsed
Zeros	147.615	0.075	147.761
Zeros and edges	38.984	0.009	39.002
MPLE	134.161	0.064	134.255

Table 13: Runtime for Equilibrium Expectation on Kapferer dataset using model 3 across the three tested starting points

Stochastic Approximation

Starting Point	User	System	Elapsed
Zeros	23.540	0.010	23.585
Zeros and edges	1114.139	0.259	1114.803
MPLE	21.153	0.009	21.193

Table 14: Runtime for Stochastic Approximation on Kapferer dataset using model 3 across the three tested starting points

Starting Point	Edges	gwesp.fixed.0.25	gwdsp.fixed.0.25
Zeros	-1.82	0.74	-0.19
Zeros and edges	-1.06	0.74	-0.37
MPLE	-2.88	1.38	-0.14

Table 15: Parameters of Kapferer dataset under model 3 with Equilibrium Expectation

4.4 Model Set 4: Kapferer With GWDSP Parameter

Equilibrium Expectation

Starting Point	User	System	Elapsed
Zeros	61.388	0.026	61.437
Zeros and edges	19.057	0.004	19.069
MPLE	56.211	0.030	56.253

Table 16: Runtimes of model set 3 on Kapferer data, with EE as the estimation method across different starting points

Stochastic Approximation

Starting Point	User	System	Elapsed
Zeros	61.248	0.036	61.586
Zeros and edges	2945.969	1.102	2950.486
MPLE	56.096	0.042	56.158

Table 18: Runtimes of model set 4 on Kapferer data, with EE as the estimation method of choice across the three different starting points

Starting Point	edges	gwdeg.fixed.0.25	gwesp.fixed.0.25	gwdsp.fixed.0.25
Zeros	-6.99	-24.31	2.26	0.43
Zeros and edges	-2.42	0.09	1.06	-0.15
MPLE	-2.94	0.12	1.38	-0.12

Table 19: Parameters of model set 4 on Kapferer data, with SA as the estimation method across various starting configurations

4.5 Discussion

First, we will consider the parameter estimates across the models in reference to the standard set by Hummel et al. (2012) re-keyed in the table 4.5 below. We note that the parameter values in Hummel's paper were calculated through the stepping algorithm with the default ERGM starting point, the MPLE. Thus, the most comparative analysis can be made when considering the implementations above with respect to an MPLE starting point, though it is still worth exploring the variances in parameter estimates and differences in runtimes across all three models.

Parameter	Model 1	Model 1 plus self-edges
Edges	-5.07 (0.027, 0.012)	-5.83 (0.067, 0.013)
Degree-2	-1.47 (0.126, 0.025)	-1.36 (0.141, 0.014)
Degree-3	-2.36 (0.192, 0.027)	-2.03 (0.206, 0.010)
Degree-4	-2.03 (0.206, 0.020)	-1.79 (0.239, 0.009)
Degree-5	-2.91 (0.415, 0.183)	-2.32 (0.409, 0.035)
GWDEG(0.25)	1.86 (0.174, 0.094)	2.32 (0.336, 0.050)
Node + Self	N/A	1.55 (0.037, 0.0002)
Self + Node	N/A	1.21 (0.118, 0.002)

Table 20: Hummel parameter estimates for the E. Coli models (defined "model set 1" and "model set 2" above) obtained using Stepping (Hummel et al. (2012))

Parameter	Model 1	Model 2
Edges	-3.016163	-3.17598
GWESP(0.25)	1.444937	1.53531
GWDSP(0.25)	0.322320	-0.11757
GWDegree(0.25)	NA	0.43127

Table 21: Parameter estimation of the Kapferer dataset with equivalent models in Hummel et al. (2012)

5 Future Work

A number of improvements can be done to build on this analysis in future, listed below. We categorise these in terms of singularly improving this analysis, as opposed to improving the theme of the analysis.

- While we only considered starting point variation and gain manipulation in this paper, there's an opportunity to continue experimenting with other hyper-parameters of the Equilibrium Expectation algorithm.
- Due to time constraints, we were unable to complete benchmarking of the MCMLE approach with EE and SA for the Kapferer data. Convergence was not achieved on the models.
- Running the models in parallel to show more realistic times in the context of modern computational statistics.
- Experimenting with different root finding methods in the Equilibrium Expectation algorithm to test the runtime.
- Increasing the number of datasets we test on, particularly focusing on a larger variance in network sizes. Looking at much smaller networks as well as much larger networks than those used in this paper.
- Benchmarking with Contrastive Divergence as a starting point for the models (Krivitsky (2017)), and considering other estimation algorithms such as partial stepping.

Appendices

A E. Coli dataset

The E. Coli dataset is a network dataset of protein localisation sites, created and maintained by Kenta Nakai. It is used in the context of this paper as a means to benchmark Equilibrium Expectation given it's well-known challenging structure. The models built on this network are derived from Hummel et al. (2012). The first, referenced as "E. Coli (no self loop)" includes edge counts, counts of actors with degree 2-5 and the gemeotrically weighted degree (GWD) parameter with decay set to 0.25. Credit for the models go to Krivitsky (2017) from his work to assess Contrastive Divergence as a method to seed MCMLE for ERGMs, and Hummel et al. (2012) as the 'ground truth' for these models.

B Kapferer's Zambian tailoring shop dataset

Similar to the above, the Kapferer data is a reputably difficult network dataset containing information of tailor shops in Zambia. The two models used are reproductions of those used in Hummel et al. (2012) so that we had a 'ground truth' from Hummel's et. al to base our own parameter estimates. Much of the terms used in both the Kapferer models and 3 of the E. Coli dataset were discussed in Hunter (2007) and a brief explanation is provided in Appendix C

C ERGM terms from Hunter (2007)

GWDegree

GWESP

GWDSP

D Code

It is impractical to include all the code written verbatim here in the appendix. Instead, find below a referenced list of code created as part of this paper.

- The implementation of the Equilibrium Expectation algorithm as part of the `ergm` package.
  - Reference to each of the scripts modified
  - Reference to each of the scripts modified
- Model code. We note that this code was largely aided by the work done by Krivitsky (2017) in his paper on Contrastive Divergence as a means of starting ERGMs.
- Analysis code for plots and tables cited in the Results section of this paper.
- A utility script used to make some of the analysis more coherent.

References

Byshkin, M., Stivala, A., Mira, A., Robins, G., and Lomi, A. (2018). Fast maximum likelihood estimation via equilibrium expectation for large network data. *Springer Nature*.

Geyer, C. J. and Thompson, E. A. (1992). Constrained monte carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society Series B*.

Hummel, R. M., Hunter, D. R., and Handcock, M. S. (2012). Improving simulation-based algorithms for fitting ergms. *Journal of Computation and Graphical Statistics*.

Hunter, D. R. (2007). Curved exponential family models for social networks. *Soc Networks*.

Hunter, D. R. and Handcock, M. S. (2006). Inference in curved exponential family models for networks. *Journal of Computation and Graphical Statistics*.

Hunter, D. R., Handcock, M. S., Butts, C. T., Goodreau, S. M., and Morris, M. (2008). ergm: A package to fit, simulate and diagnose exponential-family models for networks. *Journal of Statistical Software*.

Krivitsky, P. N. (2017). Using contrastive divergence to seed monte carlo mle for exponential-family random graph models. *Computational Statistics and Data Analysis*.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*.

Snijders, T. A. (2001). The statistical evaluation of social network dynamics. *Sociological Methodology*.

Snijders, T. A. (2002). Markov chain monte carlo estimation of exponential random graph models. *Journal of Social Structure*.

Strauss, D. and Ikeda, M. (1990). Pseudolikelihood estimation for social networks. *Journal of the American Statistical Association*.