Approximate computation and estimation of quantal response equilibrium through simulation*

James R. Bland†

2026-04-17

Abstract: I propose a simulation-based method of approximately computing Quantal Response Equilibrium. The method can also be adapted for Bayesian estimation of a game's parameters, and can estimate Quantal Response Equilibrium when multiple equilibria exist. I demonstrate this approximation and estimation procedure using an Asymmetric Chicken game. Further examples are provided in an online appendix.

1 Introduction

Quantal Response Equilibrium (QRE, McKelvey and Palfrey (1995)) has proven to be a useful lens through which to analyze play by humans in games. This is true for both for making predictions about play, and estimating parameters in a game from data. However QRE is an extension of Nash equilibrium, and solving for Nash equilibrium is a PPAD-complete problem (Daskalakis et al., 2009). These problems are believed to be hard to solve, but verifying that a candidate solution is in fact a solution is easy. That is, we can write a system of equations characterizing a QRE as follows:

$$0 = \sigma - q(\sigma, \lambda, \theta) \doteq H(\sigma, \lambda, \theta)$$

where σ is a mixed strategy profile (a stacked vector), $q(\sigma, \lambda, \theta)$ is the probabilistic best response profile to strategy profile σ, λ is the choice precision parameter (either a scalar or a vector), and θ is a vector of any other parameters that might be of relevance in the game.[1] Since q is typically very easy to evaluate, verifying that a particular $(\sigma, \lambda, \theta)$ is a solution to this equation is also very easy. Some popular choices of q in applications include the logit and Luce (Luce, 1959) specifications, which are, respectively:

*I thank Brian Monroe, attendees of the Purdue VSEEL Grand Reopening Conference, and attendees of the 2024 BSE Summer Forum 'Computational and Experimental Economics' workshop for helpful comments that improved this paper.

†The University of Toledo. james.bland@utoledo.edu https://orcid.org/0000-0002-7117-9998

[1]These could include behavioral parameters describing risk-aversion, other-regarding preferences, and so on.

$$\text{logit:}\quad q_{i,j}(\sigma,\lambda,\theta)=\frac{\exp(\lambda u_{i,j}(\sigma,\theta))}{\sum_k \exp(\lambda u_{i,k}(\sigma,\theta))},\quad \text{Luce:}\quad q_{i,j}(\sigma,\lambda,\theta)=\frac{u_{i,j}(\sigma,\theta)^\lambda}{\sum_k u_{i,k}(\sigma,\theta)^\lambda}$$

where $i = 1, 2, \ldots, n$ indexes the player, $j = 1, 2, \ldots, J_i$ indexes player i's actions, and $u_{i,j}(\sigma, \theta)$ is player i's utility from taking action j given mixed strategy profile σ. These specifications highlight both the difficulty in solving for a QRE (q is non-linear), and the ease of establishing whether $(\sigma, \lambda, \theta)$ is a QRE (q is easy to evaluate). This paper describes how this condition can be used to construct a probability distribution over σ whose modes are quantal response equilibria. Simulation of this distribution then finds a good approximation of these modes.

Extant approaches to computing QRE are many and varied, however perhaps the most efficient of these is the predictor-corrector algorithm developed in Turocy (2005) and Turocy (2010), described in detail with example code in Bland and Turocy (2026), and implemented in *Gambit* (McKelvey et al., 2014).[2] This algorithm starts at a known solution to the QRE system of equations, then uses information contained in this system and its Jacobian to trace out a "branch" of solutions. While the principal branch of the QRE is easily accessible using the predictor-corrector algorithm because we can initialize it at the *centroid* (uniform randomization and zero choice precision), other branches are not necessarily accessible unless we already have a solution on that branch. Unlike the predictor-corrector algorithm, the simulation-based approach discussed here does not require knowledge of a solution on a branch in order to find that branch. Instead, it requires initial conditions of the Monte Carlo chain used for the simulation to be closer, in some sense, to that branch than others.

Consider for example the "Asymmetric Chicken" game shown in Table 1.[3] The *locus* of logit QRE is shown in Figure 1. The locus is the set of mixed strategy profiles that constitute a QRE (for any $\lambda \geq 0$). Nash equilibria are labeled with an "N", and the three points marked "X" denote the logit Quantal Response Equilibria for $\lambda = 1.5$.[4] In order to find all three of these equilibria using the predictor-corrector algorithm, one has to know one solution to $H(\sigma, \lambda, \theta)$ for *each* of the two branches. In principle, the branch on the left of this Figure should be easily accessible because it starts at the *centroid* (in this case $\sigma_{\text{Up}} = \sigma_{\text{Left}} = 0.5$ and $\lambda = 0$). Therefore finding the QRE for $\lambda = 1.5$ on this branch is relatively straightforward. However finding the QRE on the other branch requires knowledge either of one of the Nash

[2]Another approach seen in the literature is to (computationally) minimize $H(\sigma,\lambda,\theta)^\top H(\sigma,\lambda,\theta)$ with respect to σ (see for example pp148-149 of Goeree et al. (2016)). This is similar to the corrector steps in the predictor-corrector algorithm, but does not take advantage of all of the information contained in $H(\sigma, \lambda, \theta)$. Iterating on the sequence $\sigma^{t+1} = q(\sigma^t, \lambda, \theta)$ (i.e. treating it as a contraction) may solve some games, particularly when λ is small, but is unstable for many games.

[3]This game appears in Table 2.3 of Goeree et al. (2016). I use this game as an example for two reasons. First for pedagogical reasons, while the techniques proposed in this paper are probably more useful for more complicated games, choosing a 2-player 2-action game means that everything can be visualized in two dimensions. Second, as an asymmetric game with multiple equilibria, it serves as possibly the most pathological kind of 2-player 2-action game that one could reasonably take a QRE model to.

[4]I choose $\lambda = 1.5$ in this example because there are three logit QRE for this value. For smaller λ there only exists one QRE, which is on the principal branch. For larger values of λ, the equilibrium on the principal branch is closer to the pure-strategy Nash equilibrium at $(0, 1)$, and the other equilibria are more spread out along the non-principal branch.

Table 1: Asymmetric Chicken game

	Left	Right
Up	0, 0	6, 1
Down	1, 14	2, 2

equilibria at the ends of this branch, or another solution to the QRE equation on this branch. While finding all of the Nash equilibria of a 2-player 2-action game is not too strenuous, for more complex games this can become analytically and computationally difficult. Therefore in other games, we might not even know that an equilibrium on a branch like these even exists. For this game, the method described in this paper sets up a probability distribution that will have global modes at all three of these points. Finding *an* equilibrium simply requires that the Markov chain is initialized near that equilibrium, and finding *other* equilibria can be done through changing these initial values.

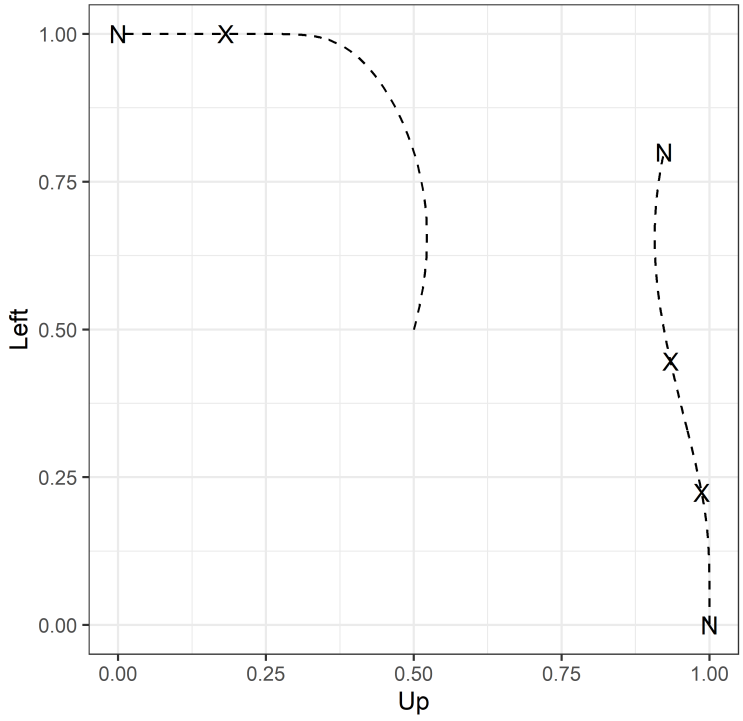

Figure 1: Locus of logit Quantal Response Equilibria in the Asymmetric Chicken game shown in Table 1. Nash equilibria are labeled with an 'N'. Points labeled with an 'X' denote logit QRE corresponding to $\lambda = 1.5$

From here, it is relatively straightforward to *estimate* the parameters λ and θ using data from play in the game using Bayesian techniques.[5] This can be achieved by specifying a prior

[5]This is also true for the predictor-corrector algorithm. The computational burden comes from computing the solutions to $H(\sigma, \lambda, \theta) = 0$. Once these solutions are found, computing the likelihood of observing the

for λ and θ, and in the special cases of logit and Luce error structures, can be estimated in ignorance of the parameter λ.

While this process is certainly not slow, it is probably not recommended for applications where the predictor-corrector algorithm works well.[6] In particular if (i) we are only interested in the principal branch (or we know there is *only* the principal branch), or (ii) we can access other branches easily, then a simulation-based approach will be less precise and more computationally intensive. Instead, it serves as an alternative approach for use when other methods are less helpful.

2 An approximation through simulation

In this section, I describe a method of approximating QRE through Monte Carlo simulation techniques. It relies on being able to sample σ from continuous probability distributions that are characterized in the form:

$$\log p(\sigma \mid \lambda, \theta) = g(\sigma, \lambda, \theta) + c(\lambda, \theta)$$

where $g(\sigma, \lambda, \theta)$ is a function that we construct, and $c(\lambda, \theta)$ is a normalizing constant that ensures that $p(\sigma \mid \lambda, \theta)$ integrates to one. In practice, we do not need to know $c(\lambda, \theta)$ in order to sample from this distribution, and this is exactly how probabilistic programming languages like *Stan* (Carpenter et al., 2017) accept information about probability distributions.[7]

In particular, consider the following (log) probability density function:

$$\log p(\sigma \mid \lambda, \theta) = -H(\sigma, \lambda, \theta)^\top W H(\sigma, \lambda, \theta) + c(\lambda, \theta)$$

where W is a positive definite weighting matrix. Since $H(\sigma, \lambda, \theta) = 0$ if and only if $(\sigma, \lambda, \theta)$ constitutes a QRE, it follows that:

$$\begin{aligned} (\sigma, \lambda, \theta) \text{ is a QRE} &\iff \log p(\sigma \mid \lambda, \theta) = c(\lambda, \theta) \\ (\sigma, \lambda, \theta) \text{ is not a QRE} &\iff \log p(\sigma \mid \lambda, \theta) < c(\lambda, \theta) \end{aligned}$$

In other words, $p(\sigma \mid \lambda, \theta)$ will have global modes at all quantal response equilibria, and will have density strictly less than these modes elsewhere. Furthermore, these modes can be made arbitrarily more peaked by multiplying W by a constant greater than one.

If there exists only one QRE mixed strategy profile σ satisfying $H(\sigma, \lambda, \theta) = 0$, then this distribution will be single-peaked, and the sampler will likely find this solution easily. On the other hand if multiple σs solve the QRE condition, then the sampler may get stuck at one of these solutions, and therefore not explore all equilibria. If one suspects that multiple

data is easy.

[6] All simulations for the Asymmetric Chicken example run on my laptop in about 2.5 minutes.

[7] This is also enough to define a target distribution for the Metropolis-Hastings algorithm.

equilibria exist, it is therefore important that one runs the sampler starting from multiple initial values in order to find these equilibria with a satisfactory probability. In fact, starting a sampler from multiple initial conditions is best practice for Markov Chain Monte Carlo, and is implemented by default in many languages.[8] Furthermore, this provides us with an opportunity to *diagnose* the existence of multiple equilibria, as Monte Carlo chains that converge to different equilibria will likely be flagged with errors based on standard diagnostics used to determine whether the sampler has worked. As I will demonstrate in the examples below, the Gelman-Rubin statistic (Gelman and Rubin (1992), often denoted $\hat{R}$) speaks directly to this kind of non-convergence.

Figure 2 shows an example of solving for logit QRE in the Asymmetric Chicken game (Table 1) for $\lambda = 1.5$. Here I set the weighting matrix to $W = w\mathcal{I}$, where $\mathcal{I}$ is the identity matrix and $w > 0$ is a (scalar) tuning parameter. As w increases, the penalty for drawing samples far away from a logit QRE grows, and so draws from each chain are closer to an equilibrium. This Figure stresses the importance of both (i) appropriately choosing the tuning parameter, and (ii) running several Monte Carlo chains starting at different initial conditions. For example, selecting $w = 10^0$ explores too much of the mixed strategy space away from the equilibria. For $w = 10^1$ we can see the chains starting to narrow down this region into areas that are near the equilibria. For $w = 10^2$ and $w = 10^3$ each chain is either centered on the equilibrium on the principal branch *or* mixes over the equilibria on the other branch, but we do not see good separation between the two equilibria on this branch, which we begin to see for $w = 10^4$. Finally, for $w = 10^5$ the chains' draws are almost indistinguishable from the equilibria at the resolution of the plot. If this excercise were performed using only one Monte Carlo chain (instead of four), we would have either not approached any equilibrium with large enough probability (if w was small), or only found one equilibrium (if w was large).

This exercise can also be performed without fixing λ in order to find the *locus* of QRE. That is, taking our target distribution and combining it with a prior for λ yields:

$$\begin{aligned} p(\sigma, \lambda \mid \theta) &\propto p(\sigma \mid \lambda, \theta) p(\lambda \mid \theta) \\ \log p(\sigma, \lambda \mid \theta) &= \log p(\sigma \mid \lambda, \theta) + \log p(\lambda \mid \theta) + c(\theta) \end{aligned}$$

Sampling from this distribution will result in draws of σ that are spread out along the locus of QRE. Adjusting $p(\lambda \mid \theta)$, the prior for λ, will place more or less probability on different parts of the locus. As with finding equilibria for a given λ, starting multiple Monte Carlo chains is important here, because the locus could consist of multiple, disjoint branches. A single chain will likely explore one branch well, but will likely not discover all branches.

Continuing with the Asymmetric Chicken example (Table 1), Figure 3 shows draws from this target distribution using a prior $\log \lambda \sim N(\log(0.1), 0.7^2)$. Just like the case with fixed choice precision (Figure 2), choosing an appropriate weighting matrix and starting multiple Monte Carlo chains at different initial conditions are both important.

[8] *Stan* (Carpenter et al., 2017) for example starts multiple chains by default.

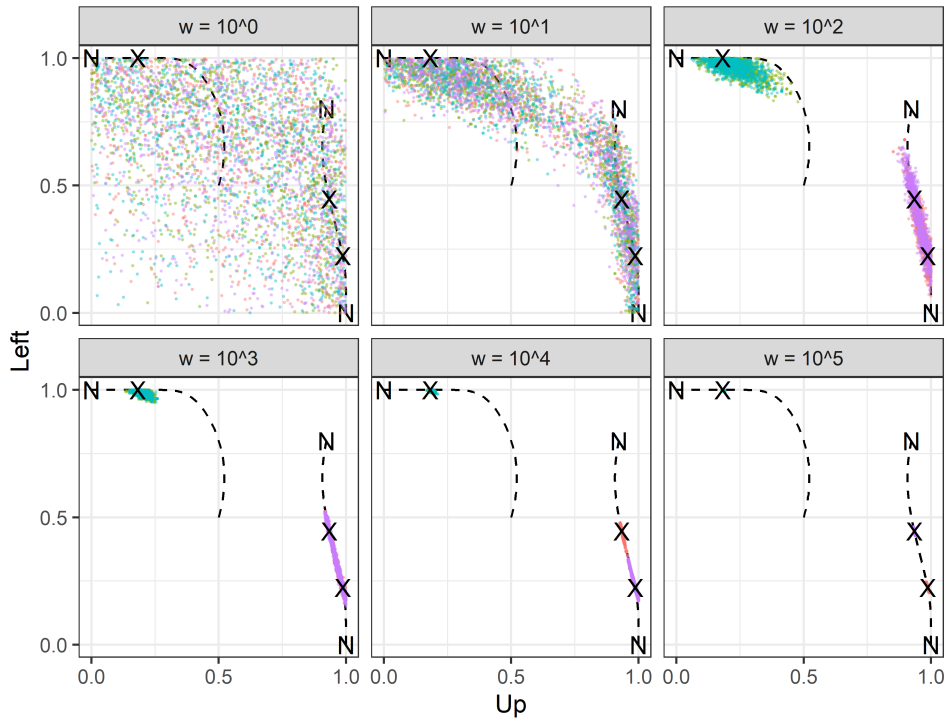

Figure 2: Simulation to find logit QRE for $\lambda = 1.5$ for various tuning parameters (w). Nash equilibria are marked with an 'N', and the logit QRE for $\lambda = 1.5$ are marked with an 'X'. Each Monte Carlo chain is denoted by a different color.

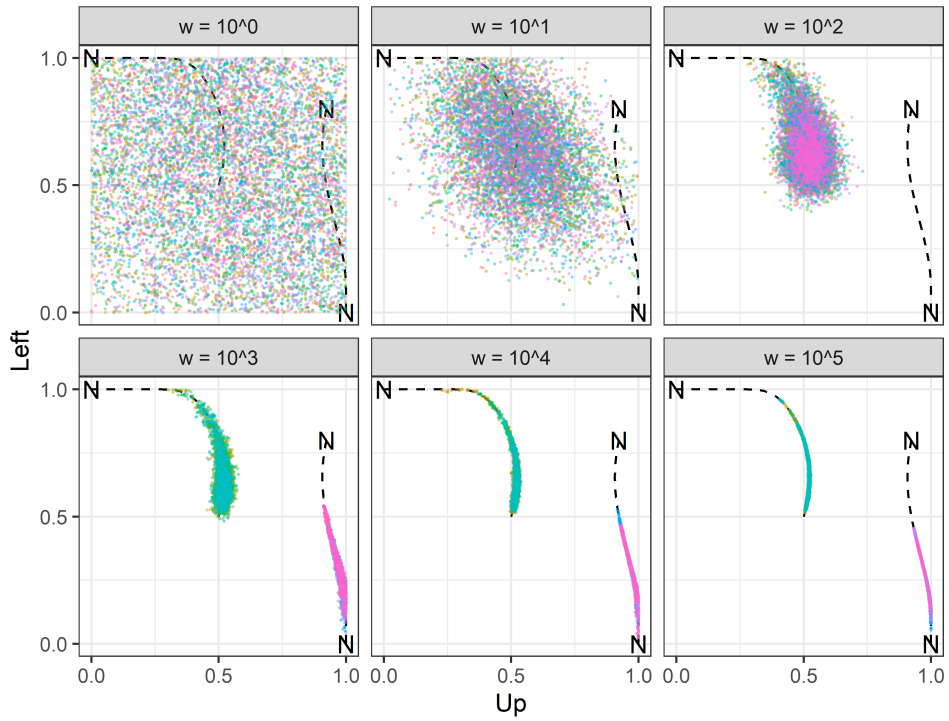

Figure 3: Simulation of logit QRE locus using a prior for λ. Each Monte Carlo chain is denoted by a different color.

3 Eliminating choice precision in some special cases

In some important special cases, it is possible to further *eliminate* λ from the target distribution. This is because expressing the QRE condition in log probability differences can linearize the problem. That is, for the logit specification of q, we can write the QRE condition as:[9]

$$0 = \sum_{i=1}^{n} \sum_{j=1}^{J_i - 1} \left(\log \sigma_{i,j+1} - \log \sigma_{i,j} - \lambda \left[u_{i,j+1}(\sigma) - u_{i,j}(\sigma) \right] \right)^2$$

and for the Luce specification:

$$0 = \sum_{i=1}^{n} \sum_{j=1}^{J_i - 1} \left(\log \sigma_{i,j+1} - \log \sigma_{i,j} - \lambda \left[\log u_{i,j+1}(\sigma) - \log u_{i,j}(\sigma) \right] \right)^2$$

Minimizing the right-hand sides of these with respect to $\lambda \geq 0$ yields minimizers for the logit and Luce specifications, respectively:

[9]Note that one could have constructed the function $H(\sigma, \lambda, \theta)$ using this log-probability difference representation. In fact, this is advisable when using the predictor-corrector algorithm, because the problem is more stable. In the cases studied here and the Appendix, I find no appreciable differences in the properties of the sampler if I use this specification.

$$\hat{\lambda}^{\text{logit}}(\sigma,\theta) = \max\left\{\frac{\sum_{i=1}^{n}\sum_{j=1}^{J_i-1}(\log\sigma_{i,j+1}-\log\sigma_{i,j})(u_{i,j+1}(\sigma)-u_{i,j}(\sigma))}{\sum_{i=1}^{n}\sum_{j=1}^{J_i-1}(u_{i,j+1}(\sigma)-u_{i,j}(\sigma))^2},0\right\},$$
$$\hat{\lambda}^{\text{Luce}}(\sigma,\theta) = \max\left\{\frac{\sum_{i=1}^{n}\sum_{j=1}^{J_i-1}(\log\sigma_{i,j+1}-\log\sigma_{i,j})(\log u_{i,j+1}(\sigma)-\log u_{i,j}(\sigma))}{\sum_{i=1}^{n}\sum_{j=1}^{J_i-1}(\log u_{i,j+1}(\sigma)-\log u_{i,j}(\sigma))^2},0\right\}$$

In fact, if (σ,λ,θ) is a (say) logit QRE, then $\hat{\lambda}^{\text{logit}}(\sigma,\theta)=\lambda$. This means that this "estimate" will correctly identify λ when σ is on the locus. Furthermore, the following target density:

$$\log p(\sigma \mid \theta) = -H(\sigma,\hat{\lambda}(\sigma,\theta),\theta)^{\top}WH(\sigma,\hat{\lambda}(\sigma,\theta),\theta)+c(\theta)$$

will have global modes along the locus, and less density elsewhere. Put differently, conditional on σ being on the locus, σ will be uniformly distributed along the locus.[10]

Figure 4 shows this technique applied to finding the locus of logit QRE in the Asymmetric Chicken game (Table 1). Again, choosing an appropriate weighting matrix (so that draws are close to the locus) and using multiple starting values (so that we find all branches with sufficiently large probability) are both important considerations.

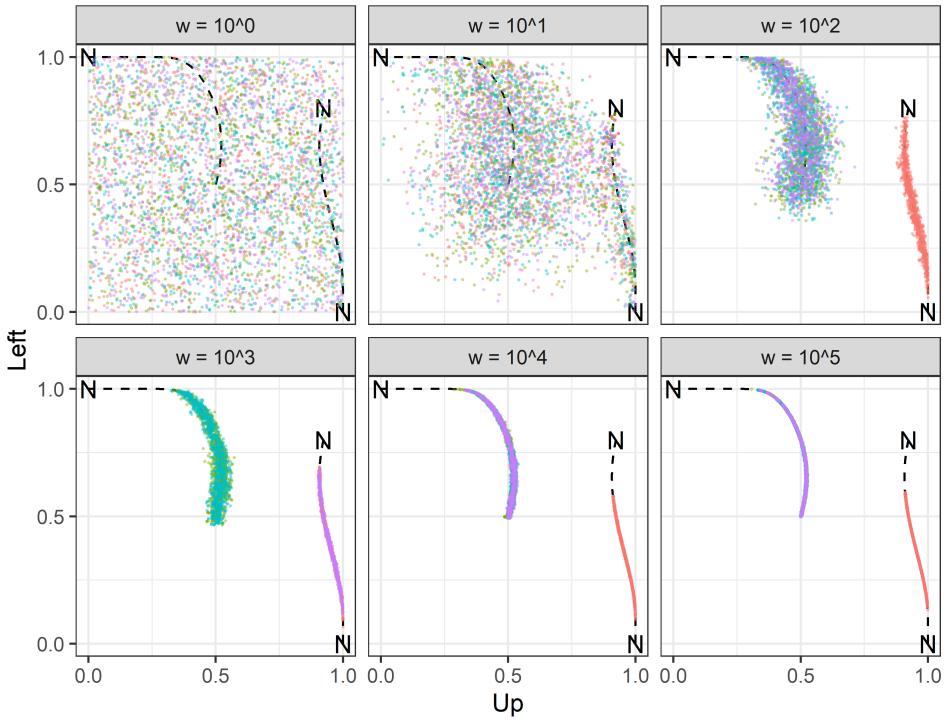

Figure 4: Simulation of logit QRE locus by eliminating λ. Each Monte Carlo chain is denoted by a different color.

[10]This is similar to the prior calibrated for the logit equilibria in Bland (2026).

4 Estimation

Now suppose that we have data y of action counts from play in our game. It is fairly straightforward to transform our target distribution $\log p(\sigma \mid \lambda, \theta)$ into a (data-augmented by σ) posterior distribution over the game's parameters λ and θ as follows:

$$\begin{aligned} p(\sigma, \lambda, \theta \mid y) &\propto p(y \mid \sigma, \lambda, \theta) p(\sigma, \lambda, \theta) \\ &= p(y \mid \sigma, \lambda, \theta) p(\sigma \mid \lambda, \theta) p(\lambda, \theta) \\ \log p(\sigma, \lambda, \theta \mid y) &= y^\top \log \sigma - H(\sigma, \lambda, \theta)^\top W H(\sigma, \lambda, \theta) + \log p(\lambda, \theta) + c \end{aligned}$$

where $y^\top \log \sigma$ is the log-likelihood of observing our data y conditional on (approximate) QRE mixed strategy profile σ, and $p(\lambda, \theta)$ is the prior over the parameters λ and θ.[11]

Continuing with the Asymmetric Chicken example (Table 1), I simulate the posterior distribution for three hypothetical datasets with 100 decisions each for the row and column players using a prior of $\log \lambda \sim N(\log(0.1), 0.7^2)$ and a weighting matrix of $W = 1,000\mathcal{I}$.[12] The posterior draws for σ are shown in Figure 5. "D" denotes the empirical choice frequencies in each dataset. As with the previous examples, we can see that different Monte Carlo chains have found different branches of the logit QRE locus. For estimation though, this is a bug not a feature: some chains which were initialized far away from the data have converged to *local* modes of the posterior distribution. These local modes are, loosely, located at the "closest" parts of the wrong branch of the locus. For example, in the dataset where 25% of row player decisions are "Up" and 90% of column player decisions are "Left" (left-most panel), the data are close to the principal branch, but the chain denoted with red dots has converged to a region of the other branch. This is likely because this chain was initialized near this branch, and could not in its warm-up overcome the large penalty associated with exploring the space *between* the two branches. It is important to remember, though, that our understanding of the problem (in particular that multiple equilibria may exist), can help us diagnose and solve this issue. Firstly, the diagnostics for these chains show that the largest $\hat{R} = 1.54$, so we will immediately know that the chains have not mixed well.[13] Secondly, we can investigate the target density of each chain.[14] A summary of this is shown in the rightmost column of Table 2. Chain 1 (first row of the Table) is the red chain in Figure 5, and we can see that the posterior mean target log-density is much lower than the other three chains. This is indicative of Chain 1 getting stuck at a local mode of the posterior distribution, and since this variable is measured in log-units, it is in a *much* less likely region of the posterior than are Chains 2-4. Furthermore, Chains 2-4 generally agree on λ and the equilibrium mixed strategy profile. In this instance one should therefore discard Chain 1, and report summary statistics based on the other chains.

[11] See Bland (2026) for a discussion of choosing appropriate priors in QRE applications.

[12] To see what these choices imply about the prior distribution of strategy profiles, see the bottom-left panel of Figure 3.

[13] *Stan* computes this and other diagnostics by default. These chains were also flagged with problems with the bulk and tail effective sample sizes, which is also indicative of non-convergence of the chains.

[14] In *Stan*, this (ignoring constants of proportionality) is output as a parameter called `lp__` by default.

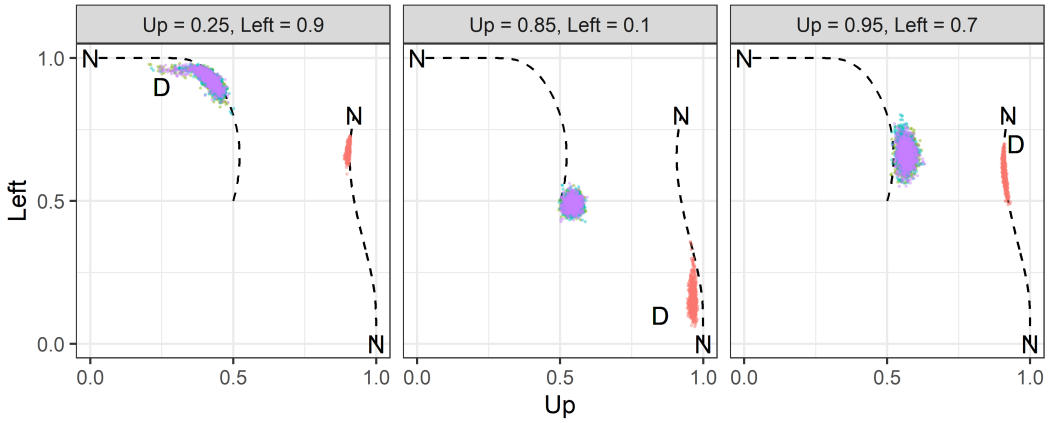

Figure 5: Posterior draws from the logit QRE model and (hypothetical) data from the Asymmetric Chicken game. 'D' denotes empirical choice frequencies in the data, which include 100 decisions for the row and column player each. Each Monte Carlo chain is denoted by a different color.

Table 2: Summary statistics of the four Monte Carlo chains shown in the left panel of Figure 5

	posterior means			
chain	Up	Left	lambda	target log-density
1	0.90	0.67	3.17	-244.18
2	0.41	0.92	0.43	-104.52
3	0.42	0.92	0.41	-104.41
4	0.41	0.93	0.45	-104.56

4.1 An alternative sampler for estimation

As mentioned above, Monte Carlo chains settling on different branches when we are *estimating* a QRE is a problem. Here, we worry that a chain is stuck at a local mode in the posterior. This mode is likely to occur on a branch different to the branch where the global mode is located. The problem lies not with the target distribution, but with the sampler used to draw from this distribution. To see this, note that samplers like *Stan* (which implements Hamiltonian Monte Carlo) and the Metropolis-Hastings algorithm are *local* samplers. This means that these samplers propose jumps for the next iteration that are centered on and close to the current iteration. Hamiltonian Monte Carlo may be particularly trapped in local modes because it is gradient-based, and will typically try to propose a next iteration that travels *up* the gradient, whereas traversing between two branches requires initially moving *down* this gradient.

One fix to this problem is to use a "population-based" sampler. One such sampler is the Differential Evolution Markov Chain (DE-MC, ter Braak (2006), ter Braak and Vrugt (2008), Hartig et al. (2023)). This approach initializes several chains at different points, and constructs proposals that include differences between the chains. As such, one chain that is currently in a low-density branch will have opportunities to jump to the high-density branch, as long as at least one of the other chains is exploring that branch. Figure 6 replicates the estimation exercise using the DE-MC sampler. Here we can see that all chains have found the global mode near the data, irrespective of their starting point.

Of course, it is also possible that our target distribution for estimation has two or more (roughly) equally important modes. This could be the case if the data fall somewhere in between the branches, and so perhaps the model cannot distinguish between points on one branch or another. This is not a problem as long as an appropriate sampler is used. For example, the DE-MC sampler will cover these modes well. On the other hand, a local sampler's chains will likely get stuck in one mode. As such, it is important to use an appropriate sampler if one suspects multiple branches.

5 Additional examples

I provide additional examples of these techniques with replication code in *R* (R Core Team, 2021) and *Stan* (Carpenter et al., 2017, Stan Development Team (2024)) here: https://github.com/JamesBlandEcon/ApproxQRE

These examples are:

- Generalized matching pennies games with and without risk-aversion using data from Selten and Chmura (2008). This is a replication of parts of Bland (2026).
- The Volunteer's Dilemma using data from Goeree et al. (2017), including participant-level heterogeneity in parameter θ: warm glow volunteering and duplicate aversion.
- A first-price private value auction with risk-averse bidders (using simulated data)

The last two of these examples include an application of Bland (2023), which notes that in some Bayesian games QRE can be computed by just focusing on the aggregate mixed strategy

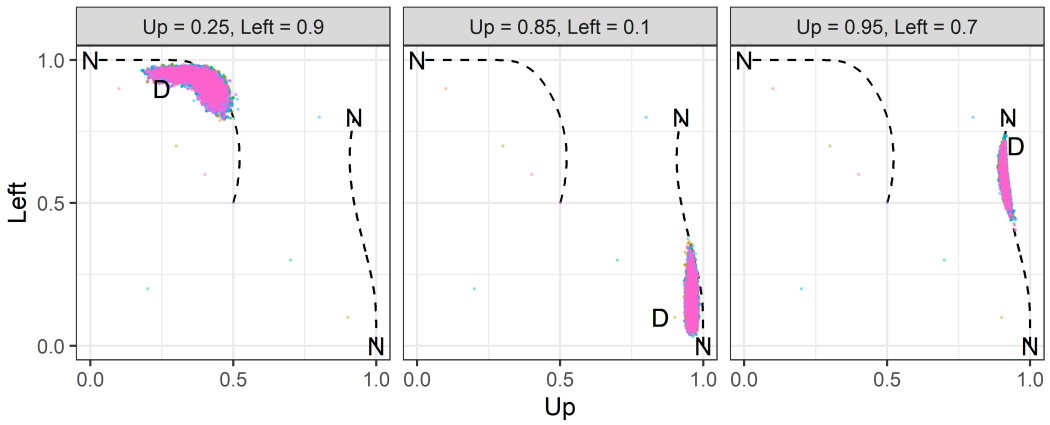

Figure 6: Posterior draws from the logit QRE model and (hypothetical) data from the Asymmetric Chicken game, using the DE-MC sampler. 'D' denotes empirical choice frequencies in the data, which include 100 decisions for the row and column player each. Each Monte Carlo chain is denoted by a different color.

profiles (i.e. unconditional on a player's type). This substantially reduces the number of elements in $H(\sigma, \lambda, \theta)$.

6 Conclusion

Solving for Quantal Response Equilibrium is computationally difficult. This paper presents a framework for approximately computing and estimating Quantal Response Equilibrium models using Monte Carlo simulation. To simulate mixed strategy profiles that are close to a QRE, we can construct a distribution whose global modes are QRE. Draws from this simulation can be made arbitrarily close in probability to these modes by adjusting a tuning parameter. If choice precision λ is fixed, each Monte Carlo chain in the sampler will likely find one equilibrium. As a single chain will likely converge to an equilibrium that is "close" to its initial conditions, initializing a chain to find an equilibrium requires less information than (say) the predictor-corrector algorithm, which requires knowledge of an exact solution on the QRE branch in order to find a particular solution. Multiple equilibria can be searched for by running multiple Monte Carlo chains with different initial conditions, and in principle the existence of multiple equilibria can be identified by standard Monte Carlo chain diagnostics. Once we can *sample* from a QRE, it is relatively straightforward to *estimate* a QRE by specifying priors on choice precision λ and any other unknown parameters that describe play in the game, although some care needs to be taken in selecting an appropriate sampler for this task.

In the Asymmetric Chicken example discussed above, the sampler was able to (i) find multiple equilibria given a choice precision λ, and (ii) trace out the QRE locus. When taken to data though, some of the good features of the sampler, namely finding multiple equilibria, stymied estimation: chains can get stuck at local modes in the distribution. This problem can be overcome by using a population-based sampler that is robust to multiple modes.

This method is perhaps more computationally intensive, but less user-intensive than implementing the predictor-corrector algorithm. This is because the predictor-corrector algorithm requires information about the Jacobian of the system of equations characterizing the equilibrium condition in addition to the system of equations itself. This must either be determined analytically, or numerically approximated. When used for estimation, this method makes no distinction between principal and non-principal branches. In some cases this could be a benefit, because we can allow our data to select the branch that best represents play in our experiment. In other cases where we may want to focus just on the principal branch, this method may not be so useful. In these cases the predictor-corrector algorithm, which will trace out one branch at a time, may be preferable for estimation.

References

Bland, J. R. (2023). Computing quantal response equilibrium in some bayesian games using marginal strategy profiles. *Available at SSRN 4551341*.

Bland, J. R. (2026). Bayesian inference for quantal response equilibrium in normal-form games. *Games and Economic Behavior 157*, 619–636.

Bland, J. R. and T. L. Turocy (2026). Quantal response equilibrium as a structural model for estimation: The missing manual. *Games and Economic Behavior 157*, 592–618.

Carpenter, B., A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. A. Brubaker, J. Guo, P. Li, and A. Riddell (2017). Stan: A probabilistic programming language. *Journal of statistical software 76*.

Daskalakis, C., P. W. Goldberg, and C. H. Papadimitriou (2009). The complexity of computing a nash equilibrium. *Communications of the ACM 52*(2), 89–97.

Gelman, A. and D. B. Rubin (1992). Inference from iterative simulation using multiple sequences. *Statistical science 7*(4), 457–472.

Goeree, J. K., C. A. Holt, and T. R. Palfrey (2016). *Quantal response equilibrium: A stochastic theory of games*. Princeton University Press.

Goeree, J. K., C. A. Holt, and A. M. Smith (2017). An experimental examination of the volunteer's dilemma. *Games and Economic Behavior 102*, 303–315.

Hartig, F., F. Minunno, and M. Paul (2023). *BayesianTools: General-Purpose MCMC and SMC Samplers and Tools for Bayesian Statistics*. R package version 0.1.8.

Luce, R. D. (1959). Individual choice behavior.

McKelvey, R. D., A. M. McLennan, and T. L. Turocy (2014). Gambit: Software tools for game theory.

McKelvey, R. D. and T. R. Palfrey (1995). Quantal response equilibria for normal form games. *Games and economic behavior 10*(1), 6–38.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Selten, R. and T. Chmura (2008). Stationary concepts for experimental 2x2-games. *American Economic Review 98*(3), 938–966.

Stan Development Team (2024). RStan: the R interface to Stan. R package version 2.32.5.

ter Braak, C. J. F. (2006). A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: easy Bayesian computing for real parameter spaces. *Statistics and Computing 16*(3), 239–249.

ter Braak, C. J. F. and J. A. Vrugt (2008). Differential Evolution Markov Chain with snooker updater and fewer chains. *Statistics and Computing 18*(4), 435–446.

Turocy, T. L. (2005). A dynamic homotopy interpretation of the logistic quantal response equilibrium correspondence. *Games and Economic Behavior 51*(2), 243–263.

Turocy, T. L. (2010). Computing sequential equilibria using agent quantal response equilibria. *Economic Theory 42*, 255–269.