The Cross Entropy Method for the N-Persons Iterated Prisoner's Dilemma

Tzai-Der Wang

Artificial Intelligence Economic Research Centre,
National Chengchi University,
Taipei, Taiwan.
email: dougwang@nccu.edu.tw

Colin Fyfe

Applied Computational Intelligence Research Unit,
The University of the West of Scotland, U.K.
email: colin.fyfe@uws.ac.uk

We apply the Cross-entropy method to the N persons Iterated Prisoners Dilemma and show that cooperation is more readily achieved than with existing methods such as genetic algorithms or reinforcement learning.

Keywords: optimization, cooperation

1. Introduction

The cross entropy method has been well introduced in[1] and was motivated as an adaptive algorithm for estimating probabilities of *rare events* in complex stochastic networks. In such a situation, a Monte Carlo simulation which draws instances from the true distribution of events would require an inordinate number of draws before enough of the rare events were seen to make a reliable estimate of their probability of occurring. It was soon realized that the cross entropy method can also be applied to solving difficult combinatorial and continuous optimization problems with a simple modification of the method. Generally speaking, the basic mechanism involves an iterative procedure of two phases:

(1) draw random data samples from the currently specified distribution.
(2) identify those samples which are, in some way, "closest" to the rare event of interest and update the parameters of the currently specified distribution to make these samples more representative in the next

iteration.

In this paper, we wish to apply the method of cross entropy to the N-persons Iterated Prisoner's Dilemma, an abstract mathematical game which has close links to the formation of oligopolies.[2]

2. The Cross Entropy Method

The Cross Entropy method is best approached from the perspective of its use in estimates of statistics concerning rare events such as the probability measure associated with the rare event.

2.1. *The Cross Entropy Method for Rare Event Simulations*

Let $l = (S(\mathbf{x}) > \gamma)$ be the event in which we are interested and typically we will be interested in problems in which l is very small. We could use Monte Carlo methods to estimate l but if l is very small this would lead to a very large number of samples before we could get reliable estimates of l. The cross entropy method uses *importance sampling* rather than simple Monte Carlo methods: if the original pdf of the data is $f(\mathbf{x})$, then we require to find a pdf, $g(\mathbf{x})$, such that all of $g()$'s probability mass is allocated in regions in which the samples are close to the rare-event. More formally, we have the deterministic estimate

$$l = \int I_{\{S(\mathbf{x})>\gamma\}} f(\mathbf{x}) d\mathbf{x} = \int I_{\{S(\mathbf{x})>\gamma\}} \frac{f(\mathbf{x})}{g(\mathbf{x})} g(\mathbf{x}) d\mathbf{x} = E_{g()} \left[I_{\{S(\mathbf{X})>\gamma\}} \frac{f(\mathbf{X})}{g(\mathbf{X})} \right]. \tag{1}$$

where I_L is the indicator function describing when L in fact occurred. An unbiased estimator of this is

$$\hat{l} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\mathbf{X_i})>\gamma\}} \frac{f(\mathbf{x}_i)}{g(\mathbf{x}_i)} = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\mathbf{X_i})>\gamma\}} \mathbf{W}(f(\mathbf{x}_i), g(\mathbf{x}_i)) \tag{2}$$

where $\mathbf{W}()$ is known as the likelihood ratio.

The best $g()$ in (1) is $g^*(\mathbf{x}) = \frac{I_{\{S(\mathbf{x})>\gamma\}} f(\mathbf{x})}{l}$, which would have the same shape as $f()$ but all its probability mass in the interesting region. Note that for the optimal $g()$, $\int_{\mathbf{x}:S(\mathbf{x})>\gamma} g*(\mathbf{x}) d\mathbf{x} = 1$ while $\int_{\mathbf{x}:S(\mathbf{x})>\gamma} f(\mathbf{x}) d\mathbf{x} = l$.

However we don't know l. So we pick a family of PDFs $g(\mathbf{x}, \mathbf{v})$, parameterised by $\mathbf{v}$ and minimise the Kullback Leibler divergence between g^* and $g()$,

$$\min KL(g^*, g) = \int g^*(\mathbf{x}) \ln g^*(\mathbf{x}) d\mathbf{x} - \int g^*(\mathbf{x}) \ln g(\mathbf{x}, \mathbf{v}) d\mathbf{x} \tag{3}$$

So we maximise the cross entropy $\int g^*(\mathbf{x}) \ln g(\mathbf{x}, \mathbf{v}) d\mathbf{x}$.

We pick $\mathbf{v} = \arg\max \int \frac{I_{\{S(\mathbf{x})>\gamma\}} f(\mathbf{x})}{l} \ln g(\mathbf{x}, \mathbf{v}) d\mathbf{x}$, and we may discard l, a constant. But getting an optimal $g(\mathbf{x}, \mathbf{v})$ for a particular γ may not be an easy task. Therefore we create a set of γ_t for which we estimate the corresponding $\mathbf{v}_t$. The γ_t are chosen such that

$$P(\mathbf{x} : S(\mathbf{x}) > \gamma_t) > P(\mathbf{x} : S(\mathbf{x}) > \gamma_{t+1}) \tag{4}$$

i.e. at each iteration, the events are becoming more rare under $f()$. Thus

$$\max \int I_{\{S(\mathbf{x})>\gamma\}} f(\mathbf{x}) \ln g(\mathbf{x}, \mathbf{v}) d\mathbf{x} \tag{5}$$

$$= \max_{\mathbf{v}_t} \int I_{\{S(\mathbf{x})>\gamma_\mathbf{t}\}} \frac{f(\mathbf{x})}{g(\mathbf{x}, \mathbf{v}_{t-1})} \ln g(\mathbf{x}, \mathbf{v}_t) g(\mathbf{x}, \mathbf{v}_{t-1}) d\mathbf{x} \tag{6}$$

$$= \max_{\mathbf{v}_t} E_{g(\mathbf{x}, \mathbf{v}_{t-1})} \left\{ I_{\{S(\mathbf{x})>\gamma_t\}} \mathbf{W}(f(\mathbf{x}), g(\mathbf{x}, \mathbf{v}_{t-1})) \ln g(\mathbf{x}, \mathbf{v}_t) \right\} \tag{7}$$

Since we are working with samples, we pick $\mathbf{v}_t$ to maximise

$$\max_{\mathbf{v}_t} \frac{1}{N} \sum_{i=1}^{N} I_{\{S(\mathbf{x}_i)>\gamma_t\}} \mathbf{W}(f(\mathbf{x}_i), g(\mathbf{x}_i, \mathbf{v}_{t-1})) \ln g(\mathbf{x}, \mathbf{v}_t) \tag{8}$$

For example, if $g(\mathbf{x}, \mathbf{v}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{\mathbf{x}-\mu}{\sigma})^2}$, we find minimum of

$$\frac{1}{N} \sum_{i=1}^{N} I_{\{S(\mathbf{x}_i)>\gamma_t\}} \mathbf{W}_{t-1} \left\{ \ln(\sigma) + \frac{1}{2\sigma^2} (\mathbf{x} - \mu)^2 \right\} \tag{9}$$

We calculate the derivative of this with respect to the parameters, and set this equal to 0, to determine

$$\hat{\mu} = \frac{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)>\hat{\gamma}_t\}} W(\mathbf{X}_i, \mathbf{u}, \hat{\mathbf{v}}_{t-1}) x_i}{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)>\hat{\gamma}_t\}} W(\mathbf{X}_i, \mathbf{u}, \hat{\mathbf{v}}_{\mathbf{t-1}})} \tag{10}$$

$$\hat{\sigma^2} = \frac{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)>\hat{\gamma}_t\}} W(\mathbf{X}_i, \mathbf{u}, \hat{\mathbf{v}}_{t-1}) (\mathbf{x}_i - \hat{\mu})^2}{\sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)>\hat{\gamma}_t\}} W(\mathbf{X}_i, \mathbf{u}, \hat{\mathbf{v}}_{\mathbf{t-1}})} \tag{11}$$

2.2. *Cross Entropy Method for Optimization*

For optimization, we turn the problem into the so-called *associated stochastic problem*(ASP) first. The basic method is

- Generate random samples from the associated stochastic problem using some randomization method.
- Update the parameters to make the production of better samples more likely next time.

Note that, unlike the rare event simulations, we do not have a base parameterisation to work to and hence we don't have the $W(\mathbf{X}_i, \mathbf{u}, \hat{\mathbf{v}}_{t-1})$ term in the calculation.

We usually wish to maximize some performance function $S(\mathbf{x})$ over all states $\mathbf{x}$ in data set $\aleph$. Denoting the maximum by $\gamma^* = \max_{\mathbf{x}\in\aleph} S(\mathbf{x})$ Thus, by defining a family of pdfs $\{f(.\,;\mathbf{v}), \mathbf{v} \in \nu\}$ on the data set $\aleph$, we follow[3] to associate with this the following estimation problem

$$l(\gamma) = \mathbf{P}_v(S(\mathbf{x}) \geq \gamma)) = \mathbf{E}_v I_{\{S(\mathbf{x})>\gamma\}} \tag{12}$$

where $\mathbf{x}$ is a random vector with pdf $f(.\,;\mathbf{v}), \mathbf{v} \in \nu$. Thus we create a sequence $f(.\,;\mathbf{v}_0), f(.\,;\mathbf{v_1}), f(.\,;\mathbf{v_2}), \ldots$ of pdfs that are optimized in the direction of the optimal density and for the fixed $\hat{\gamma}_t$ and $\hat{\mathbf{v}}_{t-1}$, we derive the $\hat{\gamma}_t$ from the following program

$$\max_{\mathbf{v}} \hat{D}(\mathbf{v}) = \max_{\mathbf{v}} \frac{1}{\mathbf{N}} \sum_{i=1}^{N} I_{\{S(\mathbf{X}_i)>\hat{\gamma}_t\}} \ln f(\mathbf{X}_i; \mathbf{v}) \tag{13}$$

[3] shows that if we have a finite support discrete distribution such as the Bernoulli distribution, then we can have the elements of the probability vector updated according to

$$\hat{p}_{t,ij} = \frac{\sum_{k=1}^{N_{elite}} I_{S(x_k)>\hat{\gamma}_t} I_{x_{ki}=j}}{\sum_{k=1}^{N_{elite}} I_{S(x_k)>\hat{\gamma}_t}} \tag{14}$$

where $\hat{p}_{t,ij}$ is the estimated probability that the i^{th} element of the probability vector will equal j at iteration t. This is the method we use in this paper. We also use the smoothing technique of[3] so that the parameter vector at time t is

$$\tilde{\mu}_t = \alpha\hat{\mu}_t + (1-\alpha)\tilde{\mu}_{t-1} \tag{15}$$

where $\hat{\mu}_t$ is the outcome of the calculation (**??**) and $\alpha = 0.2$.

3. The Iterated Prisoner's Dilemma

The 2-players' Prisoner's Dilemma was peoposed by Merrill Flood and Melvin Dresher, and formalised by Albert Tucker in 1950 and is well known to researchers in the artificial intelligence and economics fields. The story of the prisoners' dilemma is based on two men, charged with a joint violation of a law, and held separately by the police. Each is told that

(1) if one confesses and the other does not, the former will be given a reward and the latter will be jailed.

(2) if both confess, each will be fined.

At the same time, each has good reason to believe that if neither confesses, both will go clear.

The N-player Prisoners' Dilemma game can be defined by the following three properties:

(1) each player faces two choices between cooperation (C) and defection (D);
(2) the D option is dominant for each player;
(3) and the dominant D strategies intersect in a deficient equilibrium. In particular, the outcome if all players choose their non-dominant C strategies is preferable from every player's point of view to the one in which everyone chooses D, but no one is motivated to deviate unilaterally from D.

Thus the payoff matrix can be represented as in Table 1 which shows the gain for a single prisoner in a population of N players. It is important to note that the return is dependent on the actions of the other $N-1$ players in the population. The term C_i (D_i) refers to the payoff to the current strategy if it cooperates (defects) when there are i *other* cooperators in the population.

Number of Cooperators	0	1	2	$\cdots$	$N-1$
Cooperate	C_0	C_1	C_2	$\cdots$	C_{N-1}
Defect	D_0	D_1	D_2	$\cdots$	D_{N-1}

The payoff matrix of the NIPD must satisfy

(1) It pays to defect: $D_i > C_i, \forall i \in \{0, \ldots, N-1\}$.
(2) Payoffs increase when the number of cooperators in the population increases: $D_{i+1} > D_i$ and $C_{i+1} > C_i, \forall i \in \{0, \ldots, N-1\}$
(3) The population as a whole gains more, the more cooperators there are in the population: $C_{i+1} > \frac{(C_i+D_{i+1})}{2}, \forall i \in \{0, \ldots, N-2\}$.

Notice that this last gives a transitive relationship so that the global maximum is a population of cooperators.

We have previously investigated the evolution of cooperation in the IPD with evolutionary algorithms,[2,4] artificial immune systems[5] and reinforcement learning. In the next section, we use the Cross Entropy method.

3.1. *CE for NIPD*

We encode the problem so that the first position in the probability vector gives the probability that the strategy will cooperate (”1”) or defect (”0”) on the first round. In round 2, the decision taken by each agent depends on whether he himself cooperated or defected in the first round and how many other cooperators there were. Thus we have 2*N+1 bits. In round 3 the decision depends on the strategy’s previous decisions: DD, CD, DC, or CC. Within each block there are $(N+1)*(N+1)$ histories determined by the number of cooperators in each of the first two rounds, i.e., 00, 10, 20, 30, 01, 11, 21, and so on. The fourth and subsequent round decisions use a memory of the previous three rounds.

The probability vector is initialised to 0.5 in every position. At each instant in time, we generate M samples from the current distribution. For each sample, we generate K sets of $N-1$ players to play with, one of which is the current sample itself. Each group of N samples plays R rounds, typically. Unless stated otherwise, M =1000, K = 10 and R = 20. The payoffs for each cooperator is $2n$ if there are n cooperators in the population while each defector earns $2n+1$.

In our previous investigations with genetic algorithms etc., we have found that cooperation is more difficult to achieve as the number of players increases: typically, when $N=6$ or 7, total cooperation becomes impossible to achieve. This is also our finding with the basic Cross Entropy method: partial cooperation is achieved but the global optimum is not.

However we have found that by adding noise to the simulation and decreasing the amplitude of the noise, we can achieve greater cooperation. Typically we add noise drawn from a uniform distribution $U[-0.1, 0.1]$ after the usual update of the probability vector and then re-normalise so that the probabilities sum to 1.

Moreover, we can with the CE method hope to get cooperation in even larger problems. Typically the sole defection is in the first element of the probability vector. For example, in the 15 IPD, the probability vector is greater than 8×16^3, the last part of the probability vector which is used in rounds 4 to 20. This section of the probability vector is greater than 4000 dimensional and the evaluation of the performance function is dominated by this part. Thus early in the simulation this part dominates and if, by chance, the elite samples happen to defect in round 1, that is what is learned by the probability vector. The noise gives some chance to partly escape from local optima. Figure 1 shows a 20 IPD simulation under the same conditions.

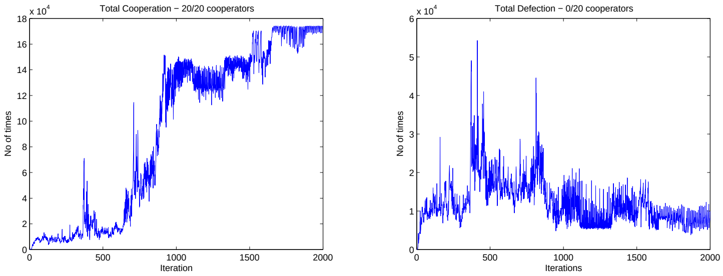

Fig. 1. Left: the number of times total cooperation was achieved in the 20 IPD simulation. Right: the number of times total defection was achieved. The amplitude of the noise declined to 0 during the first half of the simulation and remained 0 subsequently.

4. Conclusion

The resulting implementation of the Cross Entropy algorithm is very like Population Based Incremental Learning,[6] however, whereas that algorithm was introduced as an heuristic abstraction of the Genetic Algorithm, the Cross Entropy method comes with strong theoretical credentials.

References

1. P.-T. de Boer, D. P. Kroese, S. Mannor and R. Y. Rubenstein, *Annals of Operations Research* **134**, 19 (2004).
2. T. Wang, C. Fyfe and J. P. Marney, A comparison of an oligopoly game and the n-person iterated prisoner's dilemma, in *Fifth International Conference of the Society for Computational Economics,Computing in Economics and Finance, CEF99*, 1999.
3. R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method* (Springer, 2004).
4. T. Wang and C. Fyfe, Structured chromosomes for the n-person iterated prisoner's dilemma, in *Second Internaional Conference on Intelligent Data Engineering and Automated Learning, IDEAL2000*, 2000.
5. T. Wang and C. Fyfe, Achieving cooperation using artificial immune systems, in *4th International Workshop on Computational Intelligence in Economics and Finance*, 2005.
6. C. Fyfe, Noise, neighbourhoods and niches, in *Second International Conference on Soft Computing, SOCO97*, Sept. 1997.