Coupling From The Past

Kim Ward

February 2020

1 Overview

In Mathematics, often you don’t have a neat formula for the things you want to calculate. However, you can sometimes use an algorithm to get an answer that’s as close as you want, for example finding the root of an equation by rearranging it in your calculator and then pressing the = sign a lot of times.

These algorithms usually have starting values, and picking the wrong starting value can affect how long the algorithm needs to run to get a ”good enough” answer. But what if you didn’t need to pick a starting value or an amount of time to run the algorithm, and it just gave you the exact right answer? This report examines one way to do this, called Coupling From The Past or CFTP for short.

Instead of running our algorithm into the future (where no matter how long we run it we’ll always be a tiny bit away from the true answer), CFTP asks us to pretend that our algorithm has been running for an infinite time in the past up until now. It then finds out where we would be if that was the case. Since the algorithm has been running for an infinitely long time, it turns out we must be exactly at the right answer. CFTP works by considering every single possible starting value, and then tries to make them ”couple up” to end up at the same place as quickly as possible by fiddling about with the random numbers the algorithm uses. Then it doesn’t matter where the algorithm started from infinitely far in the past, because all the roads lead here!

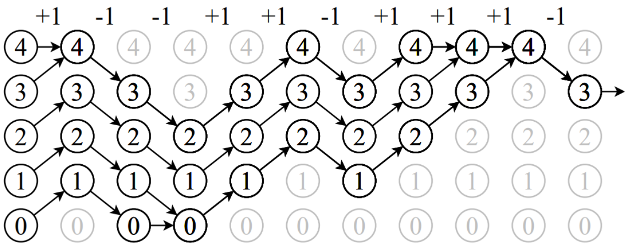

Figure 1: Coupling From The Past makes every starting value end up at the same place.

CFTP is quite a tricky method. It only works for some algorithms and needs to have random numbers, and running something ”from the past” is a lot harder to get your head around than running it the normal way. In particular, you need to be very careful where you’re getting those random numbers from, or the end answer won’t be right. Luckily, we can tell the computer to give us the same random numbers we asked it for a while ago by setting a random ”seed”. Without this, CFTP wouldn’t be possible.

2 Introduction

Monte Carlo simulation is the process of calculating a quantity Q of interest about a random variable X by phrasing that quantity as an expectation of some function $f(X)$ and then simulating many draws X_i from the distribution π of X in order to approximate Q.

$$Q(X) = \mathbb{E}_{\pi}[f(X)] \approx \frac{1}{N}\sum_{i=1}^{N} f(X_i)$$

Often the random variable X is complicated and cannot be easily drawn from, so instead a Markov chain is constructed with stationary distribution π and allowed to run for sufficient burn-in time b to minimise the effects of starting bias before using the chain's values as the draws X_i. This process is called Markov chain Monte Carlo.

$$Q(X) = \mathbb{E}_{\pi}[f(X)] \approx \frac{1}{N-b}\sum_{i=b+1}^{N} f(X_i)$$

Choosing an appropriate b is a non-trivial problem [12], and lots of computation time may be spent on running the chain for the first b steps with the data collected there being of no use. This problem is exacerbated by the rise of parallel computing, as by running multiple Markov chains in parallel the bias introduced (and therefore computational effort needed to deal with it) scales linearly with the degree of parallelisation.

Unbiased Monte Carlo methods instead draw directly from the distribution of X, at a cost of quite some computational effort. This draw can then be used as the starting point for a Markov chain in order to provide a computationally effective and unbiased approximation for Q that has no burn-in time.

In this report, we focus on one such method, known as coupling from the past (CFTP) and first proposed in 1996 by Propp & Wilson [11]. In Section 3 we place this method in context of an earlier known method to show the advances CFTP made, look at ways it was extended beyond the original paper, and point out problems with the method that limit the range of situations it can be applied to. In Section 4 we discuss pitfalls a practical user of CFTP must avoid via a computational study on a toy example, and show empirically how carelessness in how one thinks about randomness or the flow of time can introduce bias into the algorithm's results.

The ability to make draws directly from the steady state of a system is of interest to the modelling of physical systems [11] [4], simulation [3], and Bayesian statistics [8]. For a more comprehensive list of applications, see [2].

3 The Rise and Fall of CFTP

3.1 Eigenvectors

For an ergodic Markov Chain on a finite state space of size n where the transition matrix P is known, the unique invariant distribution π must satisfy $\pi P = \pi$, and so π^T must be the unique eigenvector of P^T with eigenvalue 1. It is possible to solve for all the eigenvalues and eigenvectors of a general matrix of size n in time $O(n^3)$ [6], so π can be obtained as a normalised finite-dimensional vector. Draws from π can then be simulated using a random number generator.

Since this problem is trivial for small n, we instead look at what can be done when n is sufficiently large that $O(n^3)$ is not an acceptable computational complexity, or indeed when n is infinite.

3.2 Poissonification

Asmussen et al (1992) [3] proved that unbiased Monte Carlo by observing a Markov chain was in theory possible on a finite state space via the construction of an explicit algorithm for doing so. For a state space $\{1, 2\}$ of size 2 and stationary distribution π, the algorithm is as follows:

- Run the chain starting from state 1 until state 2 has been visited and state 1 returned to.
- Let T_1, T_2 be the number of discrete timesteps spent on the first visits to state 1, 2 respectively.
- Draw $H_1 \sim \text{Gamma}(T_1, 1)$ and $H_2 \sim \text{Gamma}(T_2, 1)$.
- If $H_1 > H_2$, return state 1, else return state 2.

This works because if the discrete-time chain is turned into a continuous-time chain by *Poissonification*: assigning i.i.d. $\sim \text{Exp}(1)$ times to each discrete timestep, the first holding times H_1 and H_2 of the continuous chain satisfy $P(H_1 > H_2) = \pi_1$.

Extending this algorithm for an n-state Markov chain is possible via a recursion argument, although the algorithmic complexity is too high for practical use with large n. Nevertheless, this proof of existence opened up the search for other algorithms based not on the transition matrix but on running and observing the Markov chain itself.

3.3 Coupling From The Past

Coupling from the past (CFTP) [11] was first proposed as a way to deal with finite state spaces with large n. With notation and image borrowed from [7], the informal algorithm is as follows:

- Construct stochastic maps $f_1, f_2, ...$ from the state space S to itself, in a way that is consistent with P i.e. each map represents an application of P to each element of the state space.
- Let $\overleftarrow{F_t} := f_1 \circ f_2 \circ ...f_t$ be the backwards composition, and $C := \inf_{t \in N}\{\overleftarrow{F_t}$ is a constant map$\}$ be the *backwards coalescence time*. If a map $\overleftarrow{F_t}$ is constant, we denote by $\overleftarrow{F_t}(*)$ the unique value in its range.
- If C is finite, then for $T > C$ we must have that $\overleftarrow{F_T}(*) \equiv \overleftarrow{F_C}(*)$ and in particular, the limit as $T \to \infty$ exists and is equal to $\overleftarrow{F_C}(*)$.
- Since this limit represents "running the chain in the past for an infinite time until now", it can be argued that $\overleftarrow{F_C}(*)$ is drawn exactly from π.

The main choice a user of the algorithm must make is how to construct the maps f_t to both ensure that C is as small as possible and to allow coalescence to be easily detected. The default choice for f_t is to independently apply a transition from P to each element of the state space, but use of different types of common random numbers can lead to non-independence between $f_t(i)$ and $f_t(j)$ in a way that is exploitable. A transition can be thought of as a combination of a deterministic update rule ϕ_t and a source of randomness $U_{i,t}$.

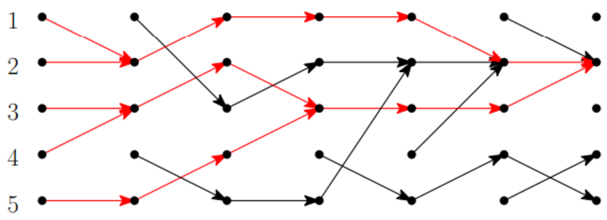

Figure 2: Coupling From The Past with independent random maps on a generic five-state Markov Chain. Red lines indicate a path in the image of all $\overleftarrow{F}_t$ up to the coalescence time.

$$f_t(i) = \phi_t(i, U_{i,t})$$

For example, continuous-space CFTP [10] partitions a well-behaved but infinite state space and uses maximal coupling on each partition to get the number of active states down to a finite number.

For any given choice of map construction, C is either a.s. finite or a.s. infinite (and thus will be a.s. finite in working CFTP algorithms), but can have heavy tails (see Figure 6 for an example and an explanation of problems this may cause).

3.4 Monotone CFTP

Monotone CFTP [11] chooses maps f_t with monotonicity properties on a partially ordered state space with greatest and least elements $\hat{0}$ and $\hat{1}$.

A map f_t is monotonic if $i \leq j \implies f_t(i) \leq f_t(j) \; \forall i, j \in S$. Ensuring all the random maps f_t are monotonic can be achieved by using a common source of randomness U_t for all states and a deterministic update rule ϕ_t that is left-monotonic.

Composition of maps preserves monotonicity, so the $\overleftarrow{F}_t$ will be monotonic as well. If for any T we have $\overleftarrow{F}_T(\hat{0}) = \overleftarrow{F}_T(\hat{1})$, by monotonicity we know that $\overleftarrow{F}_T$ is a constant map and coalescence has occurred. Therefore we are free of the requirement of tracking every element of the state space S, making monotone CFTP computationally efficient on large state spaces.

To use monotone CFTP, choose a time T in the past and run the Markov chain on $\hat{0}$ and $\hat{1}$ from this starting point. If they have not coupled by time 0, you restart the chain at a different T farther into the past, being careful about your choices of random numbers to ensure they line up with your previous ones on the shared time interval. Most monotone CFTP implementations use *binary backoff*, where each time the chain is restarted T is increased by a factor of 2.

3.5 Criticisms of CFTP

A Markov chain X with invariant distribution π is called *uniformly ergodic* if the chain converges to π at worst geometrically in a way which is independent of its starting position (see [5] for a formalisation).

[5] proved that CFTP algorithms with finite backwards coalescence times C can only occur if the chain in question is uniformly ergodic. This is quite a severe restriction: for example, a random walk Metropolis algorithm on $\mathbb{R}^d$ is not uniformly ergodic because the expected burn-in period is dependent on the starting position in an unbounded way.

Attempts to solve this problem lead to *domCFTP* [9], which couples a uniformly ergodic chain to a non-uniformly ergodic one, but also has hard-to-satisfy conditions and can be too complex to implement in practice. Other methods of using coupling to generate infinite samples have also been developed that are unrelated to CFTP, such as the recent paper [8] that inspired this report.

4 Computational Study on CFTP

4.1 Toy Example

In the following section, consider a 5-state Markov chain on the ordered state space $S = \{0, 1, 2, 3, 4\}$ with transition function "flip a coin: if heads add one if possible, if tails subtract one if possible". We will use a monotone deterministic update rule consisting of adding the same random number to each element of the state space. Upper and lower clipping ensures the chain will almost surely eventually coalesce.

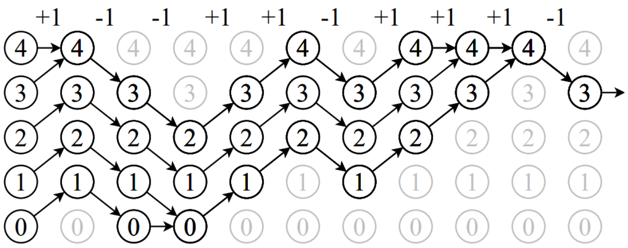

Figure 3: Coupling From The Past with monotone random maps on the five-state Markov Chain example, taken from [13].

This Markov chain has a uniform stationary distribution $\pi = (0.2, 0.2, 0.2, 0.2, 0.2)$ which can be verified analytically. By running many simulated trials and using the normal approximation to the multinomial, we can create 95% confidence intervals (shown as dashed lines in later plots) allowing us to identify when misuse of the CFTP algorithm gives results not distributed according to π. Code for the computational study can be found at [1].

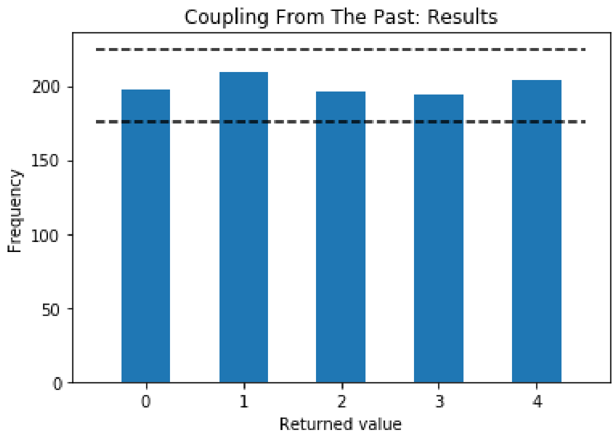

Figure 4: Coupling From The Past set up correctly, giving results within confidence intervals.

4.2 Coupling Into The Future

Let $\overrightarrow{F_t} := f_t \circ f_2 \circ ... f_1$ be the forward composition of random maps, as opposed to the backward composition $\overleftarrow{F_t}$ defined before. Forwards compositions are easier to calculate, as only values in the image of the current map composition need be tracked. It is tempting to swap out backwards for forwards compositions because of this, and such a swap can easily occur accidentally as a result of an error in one line of code.

Replacing $\overleftarrow{F_t}$ by $\overrightarrow{F_t}$ in the algorithm leads to a result no longer distributed according to π. This is because even when $\overrightarrow{F_T}$ is a constant map we no longer have $\overrightarrow{F_T}(*) = \overrightarrow{F_{T+1}}(*)$ as this transition is a step of the Markov chain itself, so our convergence results are no longer valid.

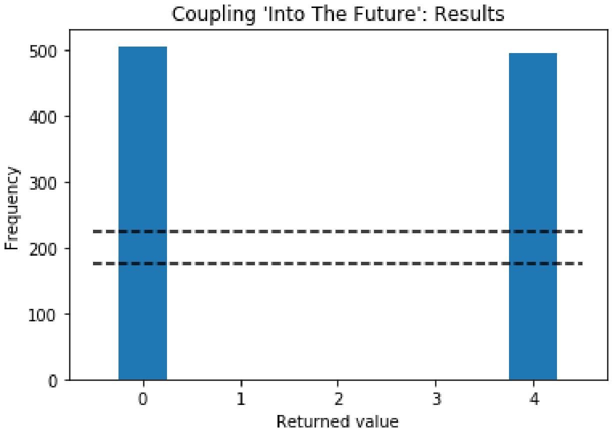

Figure 5: ”Coupling into the future” using forward composition.

In our example, when using forward composition, coalescence can only happen as a result of clipping, and so can only occur at the value 0 or 4. This leads to a result very far from π.

4.3 Interruptability bias

Although monotone CFTP used correctly draws samples exactly from π, this ”perfectly unbiased” algorithm has two features that can threaten that status:

- CFTP often has heavy tails: if the algorithm has been running a certain amount of time, the expected future running time of the algorithm is much greater.
- The running time of the algorithm is not independent of the value it returns.

See Figure 6 for a demonstration of this on our example. Here, returning a result of 4 requires four upper clips, whereas a result of 2 requires two upper clips and two lower clips. This requires on average more time, as the chain cannot directly alternate between upper and lower clips.

Fill [4] argued that these factors could introduce the sociological phenomenon of *interruptability bias*, where an impatient user aborts the algorithm and runs it again to return samples quicker, biasing against values with longer running times.

4.4 Random Seed Bias

A problem very similar to interruptability bias can also arise from careless use of random numbers in the monotone CFTP algorithm. If, each time you restart the chain farther into the past, you draw a fresh set of

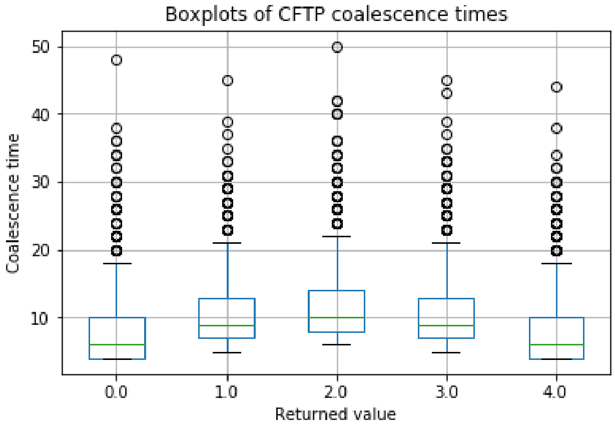

Figure 6: Boxplots of algorithm running times by end result, showing heavy upper tails and non-independence.

random numbers, you are essentially aborting the run and restarting rather than continuing the same run. This will bias your returned result towards values with shorter expected running times. Figure 7 shows an extreme version of this, by using linear backoff (increasing T by one each time) rather than binary backoff to ensure the bias at the moment of coalescence is undampened by further iterations of the chain.

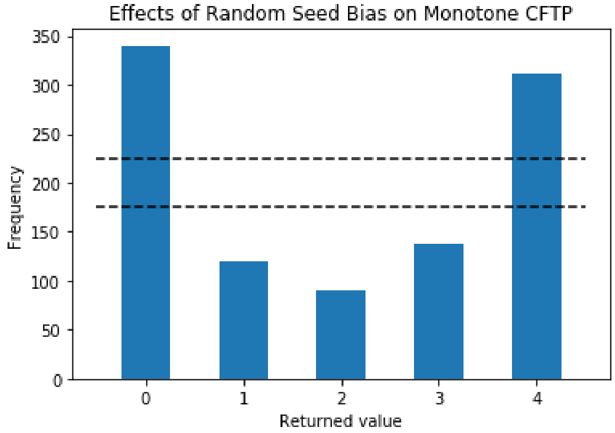

Figure 7: Careless use of random numbers will bias the results.

5 Conclusion

CFTP is a powerful algorithm, though tricky to understand and implement and only viable for certain scenarios. Making smart choices of common random numbers can have large effects on the success of an algorithm. Coupling as a broader technique is also useful for unbiased Monte Carlo beyond the specific implementations of CFTP and its derivatives, and many of the specific techniques and results formalised here (such as deterministic update rules and random algorithm completion times possibly leading to interruptability bias) continue to be of use in later research. [8]

References

[1] Code for computational study : https://github.com/kimgtward/mres/blob/master/cftp.ipynb.

[2] Web site for perfectly random sampling with markov chains: http://www.dbwilson.com/exact/.

[3] Søren Asmussen, Peter W. Glynn, and Hermann Thorisson. Stationarity detection in the initial transient problem. *ACM Trans. Model. Comput. Simul.*, 2(2):130–157, April 1992.

[4] James Allen Fill. An interruptible algorithm for perfect sampling via markov chains. *Ann. Appl. Probab.*, 8(1):131–162, 02 1998.

[5] S.G. Foss and R.L. Tweedie. Perfect simulation and backward coupling. *Communications in Statistics. Stochastic Models*, 14(1-2):187–203, 1998.

[6] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[7] Geoffrey R. Grimmett and Mark Holmes. Non-coupling from the past. *arXiv e-prints*, page arXiv:1907.05605, Jul 2019.

[8] Pierre E. Jacob, John O'Leary, and Yves F. Atchadé. Unbiased Markov chain Monte Carlo with couplings. *arXiv e-prints*, page arXiv:1708.03625, Aug 2017.

[9] Wilfrid S. Kendall and Jesper Møller. Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. *Advances in Applied Probability*, 32(3):844–865, 2000.

[10] D. J. Murdoch and P. J. Green. Exact sampling from a continuous state space. *Scandinavian Journal of Statistics*, 25(3):483–502, 1998.

[11] James Gary Propp and David Bruce Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. In *Proceedings of the Seventh International Conference on Random Structures and Algorithms*, page 223–252, USA, 1996. John Wiley Sons, Inc.

[12] Adrian E. Raftery and Steven M. Lewis. [practical markov chain monte carlo]: Comment: One long run with diagnostics: Implementation strategies for markov chain monte carlo. *Statist. Sci.*, 7(4):493–497, 11 1992.

[13] David Bruce Wilson. Layered Multishift Coupling for use in Perfect Sampling Algorithms (with a primer on CFTP). *arXiv Mathematics e-prints*, page math/9912225, Dec 1999.