Coupling From The Past

Matteo Silvestro

June 22, 2016

Abstract

Markov chain Monte Carlo methods allow to compute difficult integrals and sample from complex distributions, opening new possibilities. Gibbs sampler gained a lot of popularity, dealing with multidimensional problems. The main issue is to be able to sample directly from the stationary distribution of the Markov chain, that is to get perfect samples. An efficient algorithm to accomplish that is the coupling from the past (CFTP) algorithm. In this essay, we will describe and prove the functionality of CFTP algorithm. Furthermore, a practical application in image restoration is given, applying CFTP to a Gibbs sampler.

1 Introduction to MCMC

In probability and statistics it is not rare to find integrals to compute, e.g.

$$\mathbb{E}[f(X)] = \int f(x)p(x)dx$$

and, unfortunately, most of them are not available analitically in a closed form or they are too expensive to compute. Traditional methods involves approximation, mainly by numerical methods, than can achieve good results. But there is another way, stochastical approximation, that is more related to probability theory and is more practical in higher dimensional problems, which can be very complex to deal with for numerical methods.

In the last century, there has been a great improvement in computational statistics, mainly due to the development of more and more powerful computers. In fact, the main requirement of Monte Carlo methods is the ability to compute a large quantity of random numbers from a given distribution, that has been possible only in recent times.

Thanks to Monte Carlo methods, we are now able to tackle problems that not long ago were considered as unsolvable due to their high complexity. In statistics, as an example, it is not uncommon to get quite complicated distributions and, in the past, the only viable solution was to fall back to simple standard models, with tractable and known distributions, even if they did not fit the problem very well.

Furthermore, the field of physics was the one in which the first Monte Carlo methods developed, since it is very often required the computation of high dimensional integrals on complicated boundaries or the simulation of systems with many degrees of freedom. It is now still a very demanding sector, in which is important to have reliable and efficient algorithms.

1.1 Monte Carlo methods

In general, a Monte Carlo method is a computational algorithm that relies on repeated random sampling to obtain numerical results.

Suppose p is a probability density on an arbitrary space related to the random variable X, we may be interested in, for example:

- expectation of a function of X, i.e. $\mu = \mathbb{E}[f(X)] = \int f(x)p(x)dx$; in particular if $f(x) = x^n$ we get moments and if $f(x) = \mathbb{1}_A(x)$ we get probabilities;
- normalizing constants of the density, if only the unnormalized density p^u is known;
- marginal distributions, i.e. computing $\int p(x_1, x_2)dx_2$.

Those quantities may be too complex to evaluate analitically, if not impossible.
A Monte Carlo method usually follows this structure, to obtain $\mu = \mathbb{E}[f(X)]$ as result:

- sample $X_1, \dots, X_n \sim p$ i.i.d.;
- approximate p with the empirical distribution

$$p_N(\cdot) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X_i}(\cdot)$$

  with δ_x Dirac measure concentrated at x;
- use the empirical distribution to approximate the integral, computing the integral of f with respect to p_N

$$\hat{\mu}_N = \int f(x)p_N(x)dx = \frac{1}{N}\sum_{i=1}^{N}\int f(x)\delta_{X_i}(x)dx = \frac{1}{N}\sum_{i=1}^{N} f(X_i).$$

The estimator $\hat{\mu}_N$ has nice properties, in fact it is an unbiased estimator for μ

$$\mathbb{E}(\hat{\mu}_N) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[f(X_i)] = \int f(x)p(x)dx = \mu,$$

and using the law of large numbers we have

$$\frac{1}{N}\sum_{i=1}^{N} f(X_i) \to \mathbb{E}[f(X)] \quad \text{as} \quad N \to \infty,$$

thus we proved that the Monte Carlo method converges to the desired value as N goes to infinity.

As we have seen, it is critical to be able to sample from any p, that is quite often not available analitically or computationally. This is the main concern of Monte Carlo methods.

1.2 Markov chains

A Markov chain (MC) is a discrete time stochastic process with countable state space $\mathcal{S}$ that satisfies the Markov property, i.e.

$$\mathbb{P}(X_t = i_t \mid X_0 = i_0, \dots, X_{t-1} = i_{t-1}) = \mathbb{P}(X_t = i_t \mid X_{t-1} = i_{t-1}),$$

assuming that the chain starts at time $t = 0$ and has $t \in \mathbb{N}$ without loss of generality. We can however also suppose that the chain starts at time $-\infty$ and has $t \in \mathbb{Z}$, as we will do in the next section, as a natural extension.

We can define the *transition probability* as $p_{ij}^{(s,t)} = \mathbb{P}(X_t = j \mid X_s = i)$. If the chain is time-homogeneous, i.e. the transition probability does not depend on s, we simply write $p_{ij}^{(t)} = p_{ij}^{(0,t)} = p_{ij}^{(s,t+s)}, \forall s$ and $p_{ij} = p_{ij}^{(t,t+1)}$. The p_{ij} can be arranged in a square stochastic matrix P, called *transition matrix*.

Our main interest will be on Markov chains that are irreducible, recurrent and aperiodic.

A Markov chain is *irreducible* if $p_{ij}^{(t)} > 0$ for all $i, j \in \mathcal{S}$ and some t.

A Markov chain is *recurrent* if it returns to its initial state, say i, in a finite time with probability 1, for any state $i \in \mathcal{S}$. This is equivalent to require that the expected number of visits of the chain to any state of $\mathcal{S}$ is infinite.

A Markov chain is *aperiodic* if, for some state i, it holds that $\gcd\{t > 0 : p_{ii}^{(t)} > 0\} = 1$.

With these hypotheses, the Markov chain is said *ergodic* and has a unique *stationary distribution*, that is a distribution $\pi = (\pi_i, i \in \mathcal{S})$ such that

$$\pi P = \pi$$

or equivalently

$$\sum_{i \in \mathcal{S}} \pi_i p_{ij} = \pi_j \quad \forall j \in \mathcal{S},$$

and from any initial state the chain converges to that unique stationary distribution, that is for any $i \in \mathcal{S}$

$$p_{ij}^{(t)} \to \pi_j \quad \forall j \in \mathcal{S}$$

as $t \to \infty$. This means that, if we make the chain run long enough we get the stationary distribution.

From now on, if not stated otherwise, all Markov chain are supposed to be ergodic.

1.3 Generation of random numbers

We make a fundamental assumption: that we know a reliable way to generate a sample from $\mathcal{U}(0,1)$, i.e. a uniform distribution in $(0,1)$. Every programming language has a function `rnd()` that generates our desired random sample. Its working principle is not trivial and will not be discussed here. Anyway, it is useful to know that a pseudo-random number generator (PRNG) `rnd()` takes an initial seed, say x_0, and use a deterministic function D to obtain the sequence of random numbers $\{x_i\} = \{D^i(x_0)\}$ in $(0,1)$, with $D^i(x_0) = D(D^{i-1}(x_0))$. For every seed the entire sequence is the same, but the numbers of the sequence are distributed according to a uniform distribution in $(0,1)$. Usually the seed is set as a number related to the current date and time, to have a different one every time the algorithm is run.

Being the sampling of random number the central problem of Monte Carlo methods, it is no surprise that the introduction of such algorithms accelerated the development of reliable PRNG.

If a cumulative distribution function F is available analitically, it is possible to resort to simple methods such as the *inverse transform method*, that states that if $U \sim \mathcal{U}(0,1)$, then the random variable $F^{-}(U)$ is distributed according to F, with $F^{-}(u) = \inf\{x : F(x) \geq u\}$ *generalized inverse*. The problem is that F usually does not have a nice form and most of the time the generalized inverse is not computable.

Other algorithms were then developed, requiring only the function form of the density f (usually nicer than the distribution), usually up to a multiplicative constant.

We may use *accept-reject method*, in which the main idea is that simulating $X \sim f$ is equivalent to simulating $(X, U) \sim \mathcal{U}\{(x,u) : 0 < u < f(x)\}$. That can be accomplished, simplifying things a lot, by generating a uniform random variable in an adequate interval and accepting or rejecting it if it is under the density or not, respectively.

Another way is the *importance sampling*, in which we sample from a different (usually simpler) density g (called *importance function*) and then reweight it by using the original density f.

Anyway, all those methods require a good knowledge of f, and this is not always the case.

1.4 Markov chain Monte Carlo

The main idea is that, instead of sampling directly from f, we sample from an ergodic Markov chain that has stationary distribution f.

Different algorithms were developed to find such a Markov chain, the most important ones being:

- Metropolis-Hastings algorithm;
- Gibbs sampler,
- slice sampler.

For instance, a two-stage Gibbs sampler for X, Y random variables with joint density $f(x, y)$ generates a Markov chain (X_t, Y_t) performing the following steps

```
Take $X_0 = x_0$
For $t = 1, 2, \dots$ generate
  $Y_t \sim f_{Y|X}(\cdot \mid x_{t-1})$
  $X_t \sim f_{X|Y}(\cdot \mid y_t)$
```

where $f_{Y|X}$, $f_{X|Y}$ are the conditional densities. In this case the stationary distribution is our desired f.

All these methods are really powerful, since sampling from a Markov chain is way simpler than sampling directly from a complex distribution. There is, anyway, a big problem related to the starting state.

We start a Markov chain from X_0 and, at the beginning, we know that $\mathbb{P}(X_t \mid X_0) \neq \mathbb{P}(X_t)$, so its successive states depend on X_0. As time goes on, since we have an ergodic MC, we are assured that $\mathbb{P}(X_t)$ converges to its unique stationary distribution and, hence, it forgets its initial state X_0. This happens at a certain time, called *mixing time*, and the chain is thereafter said to be in *equlibrium*: from this time on, the samples are from the stationary distribution. The period between $t = 0$ to the mixing time is said *burn-in period*, and is always discarded.

The problem is that it is not trivial to find such mixing time.

There have been a lot of studies in Markov chains convergence, but ultimately there is not a reliable and universal way to find a time t^* big enough such that the samples are from the stationary distribution.

Anyway, Propp and Wilson (1996) found a way to generate samples that are *perfect*, i.e. that comes from the stationary distribution, exploiting a clever way to determine the burn-in period by using the concept of *coupling*. This is known as *coupling from the past* (CPFT) algorithm.

2 CFTP

The coupling from the past algorithm, developed by Propp and Wilson [1], allows for perfect and independent sampling from a certain distribution. The sampling works by running an ergodic Markov chain that has as stationary distribution the target distribution. The main idea is running coupled Markov chains, starting from all initial states. After some time, all chains meet, i.e. they *coalesce*, and they follow the same path: this means that the effect of initial states is worn off. A critical tool of the method is called *coupling*.

2.1 Coupling

Suppose to start a Markov chain for all initial states. We require a valid technique that ensures that coalescence at a finite time occurs with a large enough probability. For instance, if we run the chains independently the probability that they all take the same value at some time decreases as a power of the number of initial states. Hence, a *coupling scheme* is needed, so that all chains are marginally generated from the original transition probabilities and that two chains that once take the same value remain identical forever after.

2.1.1 Random walk coupling

We illustrate it with an example.

Example 1 (A random walk.)**.** *Suppose we have three balls disposed in two urns. With probability* $\frac{1}{2}$ *we pick a ball from the left urn and we put it inside the right urn. Alternatively, we pick a ball from the right urn and we put it into the left urn. If we find a chosen urn empty we do nothing. What is the long-run average number of balls in the right urn?*

We might represent the problem as a Markov chain, as shown in the diagram in Figure 1.

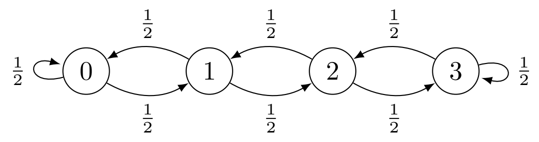

Figure 1: A random walk displayed as a Markov chain diagram.

In this case it is straightforward to compute the stationary distribution $\pi = (\pi_0, \pi_1, \pi_2, \pi_3)$ using the equation $\pi = P\pi$, that gives us the (not so surprising) solution $\pi = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$. But, for the sake of the example, suppose we can not compute directly such stationary distribution with analytical methods and we resort to a simulation method based on perfect samples. How can we know when the chain has reached equilibrium and, hence, the sample comes from the stationary distribution?

As said before, we start a different Markov chain from all four initial states, starting from time $t = 0$. Suppose that, for any time, we toss a fair coin that can give heads (H) or tails (T). If the result is H we move to the next state (from i to $i+1$, if possible), otherwise we come back to the previous state (from i to $i-1$, if possible). Let us consider the following possible scheme in Figure 2.

As we have pointed out before, at each time t all the chains use the same random number (in this case, the result of a tossed coin) to make the transition, i.e. they are *coupled*, hence they are not run independently. We notice that at time $t = 5$ coalescence occurs and so, from this time on, all the chains follow the same path and we can say that the initial state effect is worn off.

We may call this procedure *forward coupling*.

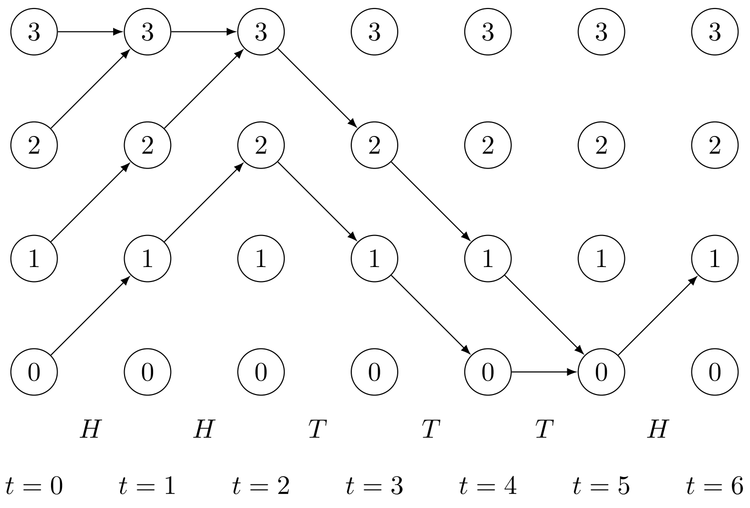

Figure 2: A random walk displayed as a Markov chain diagram.

2.1.2 Formalization of coupling

A convenient representation of coupling in a formal way uses the concept of *random mapping* $\phi(\cdot, \cdot)$, that specifies the next state of the chain as a function of the current state and random numbers.

Given that a Markov chain transition can be generally represented as

$$x_{t+1} = \phi(x_t, u_{t+1}),$$

with u_t's are iid from a fixed distribution (usually a $\mathcal{U}(0, 1)$). Of course, the random mapping is derived from the Markov chain and can be represented in different ways.

We can define the successive iterations of a MCMC algorithm as a sequence of random mappings on $\mathcal{S}$,

$$\phi_1, \phi_2, \ldots, \phi_t, \ldots,$$

where $\phi_t(x) = \phi(x, u_t)$. In particular we can write

$$x_t = \phi_t \circ \cdots \circ \phi_1(x_0),$$

which is also called *stochastic recursive sequence.* We note two important features of such random mappings

- for a give starting state x_0, the successive images of x_0 by the composition $\phi_t \circ \cdots \circ \phi_1$ are correctly distributed from the t-th transition of the original Markov chain;
- starting from the same state x_0 and with the same random mappings, the path will always be the same, being based on the same sequence of random numbers u_t.

This means that if we have two starting values $x_0^{(1)}$ and $x_0^{(2)}$, the sequences

$$x_t^{(1)} = \phi_t \circ \cdots \circ \phi_1(x_0^{(1)}) \quad \text{and} \quad x_t^{(2)} = \phi_t \circ \cdots \circ \phi_1(x_0^{(2)})$$

will be both distributed from the correct Markov transition and, since they follow the same random mappings, the two sequences will be identical from time t^* at which the coalescence occurs on, that is exactly what we wanted.

There seems to be a problem about random mappings. In fact, we do not now in advance how long the sequence $\{u_t\}$ should be, so we should generate an infinite sequence. Furthermore, such sequence must me stored somewhere, taking a lot of space.

This concern is easily overcame by using pseudo-random number generators. In fact, we may use as u_t the t-th element of the sequence of random numbers generated from a fixed seed u_0. Hence, there is no need to store any number but the seed, and we can generate a sequence as long as wanted time by time.

With all this construction, we can say that a coalescence happened by time t if the composition $\phi_t \circ \cdots \circ \phi_1$ is *constant*, i.e. if

$$\phi_t \circ \cdots \circ \phi_1(x_0) \equiv x$$

for a fixed $x \in \mathcal{S}$ and $\forall x_0 \in \mathcal{S}$.

2.1.3 Problem of forward coupling

Let us come back to Example 1. With the notation we introduced we may say that

$$u_1 = H, u_2 = H, u_3 = T, u_4 = T, u_5 = T, u_6 = H, \ldots$$

with u_i iid from a $\mathcal{B}ernoulli(\frac{1}{2})$, while our random mapping was defined as

$$x_{t+1} = \phi(x_t, u_{t+1}) = \begin{cases} \min\{x_t + 1, 3\} & \text{if } u_{t+1} = H \\ \max\{x_t - 1, 0\} & \text{if } u_{t+1} = T \end{cases}$$

with $x_t \in \mathcal{S} = \{0, 1, 2, 3\}$.

The time at which coalesce occurs is $t = 5$, in fact we notice that $\phi_5 \circ \cdots \circ \phi_1(x_0) \equiv 0$ for $\forall x_0 \in \{0, 1, 2, 3\}$, i.e. it is constant.

We might then say that the value $x_5 = 0$ is a sample from the stationary distribution, since the initial state effect is worn off, but we would be wrong. As a matter of fact, we know that the stationary distribution is $\pi = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$, but, by the way the chain is constructed, when coalescence happens at a time τ we will be either in state 0 or 3 and we will conclude that $\pi_1 = \pi_2 = 0 \neq \frac{1}{4}$, in contrast with the true stationary distribution. We might argue that we can overcome this problem by just taking one step more, in fact at time $\tau + 1$ we will be with equal probability in all states. This is, once more, not correct because, while it may hold for this chain, it does not hold in general: it is easy to see that, for example, taking another random walk with one more state 4, this is not true.

The problem is, in fact, that τ is a random time and, hence, sampling from X_τ will yield biased samples. To overcome this problem we must sample from a *fixed* time $T \geq \tau$: X_T is now a sample from the stationary distribution. However, we do not know how to choose such a T.

2.2 Coupling from the past

The idea of CFTP is to introduce a simple but fundamental modification to the forward coupling so that the produced samples are perfect. The basic difference is that in CFTP the Markov chains are run from the past ($t < 0$) to the present $t = 0$ and, most importantly, the samples are always drawn from a fixed time, i.e. $t = 0$, provided that coalescence happened before.

2.2.1 Heuristics of CFTP

The CFTP works in the following way. Suppose to start a Markov chain from all initial states, say there are k of them. These chains will eventually coalesce before arriving to the present and, from some time on, will follow the same path. Then, at $t = 0$ we have a sample from the stationary distribution.

We can not, however, start the chain from time $-\infty$, so we will use a clever trick and proceed backwards. We begin by starting at time $T = -1$ and check for coalescence at time $t = 0$. If coalescence occurred, we accept the state of the chain at time $t = 0$ as a sample from the stationary distribution. Otherwise, the starting time is moved back at $T = -2$ and chains are evolved again and checked for coalescence. If by time $t = 0$ coalescence took place, we accept the state at $t = 0$ as a perfect sample, otherwise the starting time is moved further back to $T = -3$. We continue this way, and a sample is drawn if coalescence occurs or the starting time is shifter further back. This process continues until coalescence occurs by time $t = 0$.

The are two important points that guarantee the correctness of the algorithm.

1. To achieve coalescence with a large enough probability, as we said before, we must use a random mapping to evolve the chain from the starting states. That means that at each transition the same number u_t is used to decide where to go next. Let us suppose, for example, that we started at time $T = -1$ and generated a random number u_0, we use it in the transition from $t = -1$ to $t = 0$ by the random mapping $x_0 = \phi(x_{-1}, u_0)$. If coalescence did not occur, we move back to $T = -2$ and perform two transitions. Anyway, we only produce a new random number, u_{-1}, to transition from $t = -2$ to $t = -1$, and use the *old* random number u_0 to transition from $t = -1$ to $t = 0$. Therefore, every time we move back the starting time T, we generate only one new random number u_{T+1} to be used for the first transition; for the remaining transitions the old random numbers $u_{T+2}, \ldots, u_0$ are used. This way the paths of the Markov chains are coupled together the same way every iteration of the algorithm.

2. The samples are drawn always at time $t = 0$, even if coalescence occurred earlier. This time it is not a mistake drawing exactly at time $t = 0$ even if coalescence took place exactly at time $t = 0$, since we do not always sample at such random time.

Let us go back once more to Example 1 and apply CFTP.

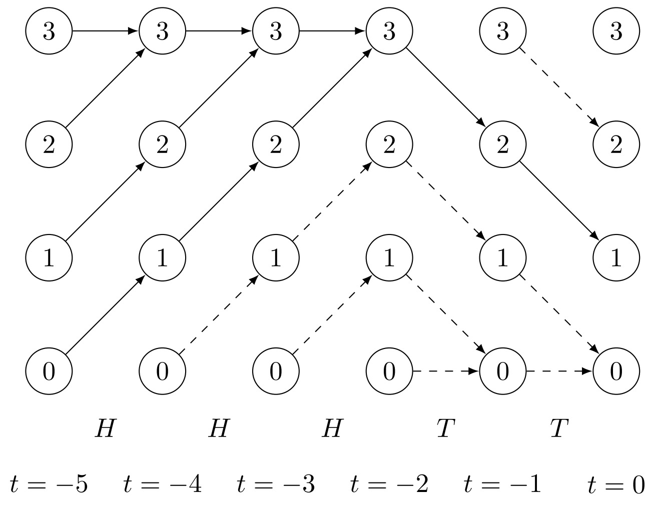

Figure 3: CFTP applied to a random walk.

Referring to Figure 3, we must go back to time $t = -5$ to have coalescence at time $t = -2$. From that time on, all four chains starting from $t = -5$ proceed together. We may be tempted to say that 3 is a sample from the stationary distribution but, as said before, this is not correct since we stopped at a random time. The perfect sample, in this case, is 1, attained at fixed time $t = 0$. Dashed lines represent Markov chains started at a certain time $t > -5$ (in previous iterations) that are no more available starting from time $t = -5$, since their initial states are no more reached.

Now we provide a heuristic argument why the CFTP returns a perfect sample in finite time. Let T be the starting time for which the trajectories coalesce by time $t = 0$, say at time $t = \tau$. In addition, suppose that the Markov chain is run from the infinite past. We know that from time $t = \tau$ on, the chain is in equilibrium, since coalescence occurred, and the chain is following a stationary distribution. Note that we do not know the state of the chain at T, and it does not really matter since, from such time, we are starting from all possible states. Furthermore, we do not know either the state of the chain at time $t = \tau$, but this is not relevant since all what matters is that from time $t = \tau$ to time $t = 0$ the state space remains a singleton (or, equivalently as before, the composition of random maps remains constant). Therefore, we can state that x_0 (recall that this time $t = 0$ is not the starting time but the time at which we stop) is a result of infinite time simulation and that is a perfect sample from the stationary distribution. The method can thus be referred to as a virtual simulation from time $-\infty$ as it allows to sample from an infinitely long simulation reconstructing it on a finite time interval.

2.2.2 Propp and Wilson algorithm

In summary, the CFTP algorithm can be described in pseudo code as follows [3], based on the version proposed by Propp and Wilson [1].

Algorithm 1: Coupling from the past

```
$T \leftarrow -1$
Repeat
  Sample $u_{T+1}$ from $\mathcal{U}(0,1)$
  $\mathcal{S}^{(T)} \leftarrow \mathcal{S}$
  For $t = T, T+1, \dots, -1$
    $\mathcal{S}^{(t+1)} \leftarrow \phi(\mathcal{S}^{(t)}, u_{t+1})$
  $T \leftarrow T - 1$
Until $\mathcal{S}^{(0)}$ is a singleton.
```

Now we state the theorem that gives theoretical validity to CFTP algorithm [1] and present its proof.

Theorem 1. *The CFTP algorithm returns a value with probability* 1*, and this value is distributed according to the stationary distribution of the Markov chain.*

Proof. Following [5], the proof is based on the fact that the k Markov chains do coalesce at some finite time into one chain, say $\{X_t^*\}$.

We recall that the Markov chain is ergodic and that π is its unique stationary distribution. Hence, each chain $\{X_t^{(j)}\}$ starting from $j \in \mathcal{S}$ is irreducible and $\mathcal{S}$ is finite, there exists $T_j < \infty$ such that, for $T \geq T_j$

$$\mathbb{P}(X_T^{(j)} = x \mid X_0^{(j)} = j) > 0, \quad \forall x \in \mathcal{S}.$$

Then, each chain has a positive probability of being in any state at time $T \geq \max\{T_1, T_2, \dots, T_k\}$ and for some $\varepsilon > 0$

$$\mathbb{P}(X_T^{(1)} = X_T^{(2)} = \cdots = X_T^{(k)}) > \varepsilon.$$

If we define the events

$$C_i = \{\text{the } k \text{ chains coalesce in } (-iT, -(i-1)T)\},$$

under the assumption that all chains are started at time $-iT$ for the associated CFTP algorithm, then we have $\mathbb{P}(C_i) > \varepsilon$. Moreover, the C_i are independent because coalescence in $(-iT, -(i-1)T)$ depends only on $\phi_{-iT}, \phi_{-iT-1}, \dots, \phi_{-(i-1)T}$, that are independent by definition.

Thus, since

$$\sum_{i=1}^{\infty} \mathbb{P}(C_i) = \infty,$$

we can apply second Borel-Cantelli lemma and conclude that

$$\mathbb{P}\{C_i \text{ happens infinitely often}\} = 1,$$

so our first claim is proved.

Then, we know that the CFTP algorithm starts from all possible states, this implies that the realization of a Markov chain $\{X_t\}$ started from $-\infty$ will coalesce at a time $-\tau$ and thereafter be equal to $\{X_t^*\}$. Therefore, X_0 and X_0^* will have the same distribution and, in particular, $X_0^* \sim \pi$.

□

In the CFTP algorithm given above values $T = -1, -2, -3, \dots$ are taken as starting times of the chains. Anyway, this is not necessary and, in principle, it is possible to take any decreasing sequence of T. In particular, Propp and Wilson [1] recommended using $T_i = -2^{i-1}, i = 1, 2, 3, \dots,$ which doubles at every step. This choice minimizes the worst-case number of required simulation steps and almost minimizes the expected number of steps.

In fact, to realize that, it is possible to calculate the asymptotic running time of the algorithm in the two cases. Suppose that $-\tau$ is the smallest T such that all the k chains coalesce by time $t = 0$. If starting times are set as $T_i = -i$ the number of iterations needed is τ and the running time (using big $\mathcal{O}$ notation) is

$$N_1(\tau) = k \sum_{i=1}^{\tau} i = k \frac{\tau(\tau+1)}{2} = \mathcal{O}(\tau^2);$$

while if the starting times are set as $T_i = -2^i$ the number of iterations needed is the minimum integer greater than $\log_2(\tau)$, i.e. $\lceil \log_2(\tau) \rceil$, so the running time is

$$N_2(\tau) = k \sum_{i=0}^{\lceil \log_2(\tau) \rceil} 2^i = k(2^{\lceil \log_2(\tau) \rceil + 1} - 1) \leq k 2^{\log_2(\tau)+2} = \mathcal{O}(2^{\log_2(\tau)}) = \mathcal{O}(\tau),$$

so there is an increase of asymptotic performance from quadratic (not good) to linear (very good).

2.3 Monotone CFTP

The CFTP algorithm as described above is not very simple from a computational point of view, since it needs to keep track of k Markov chains at once, and can be very intensive if the state space $\mathcal{S}$ is very large. Therefore, its practicality is quite limited. In some cases it is possible to define a random map that possesses the property of monotonicity for a partial order imposed on the state space. This way we would work only on two chains.

2.3.1 Monotonicity

Suppose that there is a *partial order* on $\mathcal{S}$, that is, for all $i, j, k \in \mathcal{S}$, it exists a binary relation $\preceq$ such that

- $i \preceq i$ (reflexivity),
- if $i \preceq j$ and $j \preceq i$ then $i = j$ (antisimmetry),
- if $i \preceq j$ and $j \preceq k$ then $i \preceq k$ (transitivity),

holds. Then a random mapping is said *monotone* if $\phi(i,u) \preceq \phi(j,u)$ when $i \preceq j$, $\forall u$.

Now, denote with $\tilde{0}$ and $\tilde{1}$ respectively the minimum and the maximum element of $\mathcal{S}$ with respect to the partial order, that is $\tilde{0} \preceq i \preceq \tilde{1}, \forall i \in \mathcal{S}$. If there is a monotone random mapping $\phi(\cdot,\cdot)$ for the Markov chain, the use of such transition rule preserves the order in the state space. Then, it is necessary to monitor the two chain whose starting space are $\tilde{0}$ and $\tilde{1}$. In fact, when coalescence happens by time $t = 0$, i.e. $\phi_\tau \circ \cdots \circ \phi_0(\tilde{0}) = \phi_\tau \circ \cdots \circ \phi_0(\tilde{1})$, the (virtual) chains started from all possible initial states in $\mathcal{S}$ must have coalesced. Therefore, the intermediate chains need not to be monitored, resulting in a big advantage in computing if the state space is large.

If we go back to Example 1, we may introduce the partial order

$$0 \preceq 1 \preceq 2 \preceq 3,$$

for which is trivial to show that

$$\phi(0,u) \preceq \phi(1,u) \preceq \phi(2,u) \preceq \phi(3,u) \quad \forall u,$$

so that it is necessary to monitor only the Markov chain starting from $\tilde{0} = 0$ and $\tilde{1} = 3$. Check also Figure 3 to have a graphical understanding of this.

2.3.2 Antimonotonicity

In some situations, when a monotone random mapping can not be found, it may still be possible to monitor only two paths. This can be achieved by the *crossover method*, which can be applied if the random mappings are antimonotone.

A random mapping is *antimonotone* if $\phi(i,u) \succeq \phi(j,u)$ when $i \preceq j$, $\forall u$ for a certain partial order $\preceq$.

As an example, take a modification of the random walk shown in Exercise 1 as seen in Figure 4 [2].

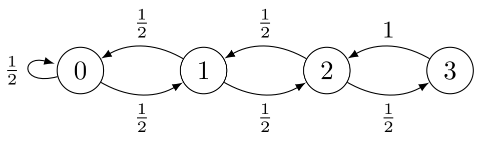

Figure 4: Another random walk displayed as a Markov chain diagram.

To make transitions we can use a rule such that

- if the coin comes up heads, we remain in state 0 if we are in 0, we move up if we are in 1 or 2 and we move a step down if we are in 3;

- if the coin comes up tails, we move a step up from 0 and we move a step down if we are in 1, 2 or 3;

that can be formalized in a random mapping as follows

$$x_{t+1} = \phi(x_t, u_{t+1}) = \begin{cases} 0 & \text{if } u_{t+1} = H \text{ and } n = 0 \\ 1 & \text{if } u_{t+1} = T \text{ and } n = 0 \\ x_t + 1 & \text{if } u_{t+1} = H \text{ and } n \in \{1, 2\} \\ x_t - 1 & \text{if } u_{t+1} = T \text{ and } n \in \{1, 2\} \\ 2 & \text{if } n = 3 \end{cases}$$

with u_t result of the coin toss. Such random mapping is not monotone. Anyway, if we define the following partial order

$$2 \preceq 0 \preceq 1 \preceq 3,$$

then we have $\phi(i, u) \succeq \phi(j, u)$ when $i \preceq j, \forall u$, in fact

$$\begin{cases} 3 \succeq 0 \succeq 2 \succeq 2 & \text{if } u = H \\ 1 \succeq 1 \succeq 0 \succeq 2 & \text{if } u = T \end{cases}$$

so it is possible to monitor only states $\tilde{0} = 2$ and $\tilde{1} = 3$, as it is seen in Figure 5.

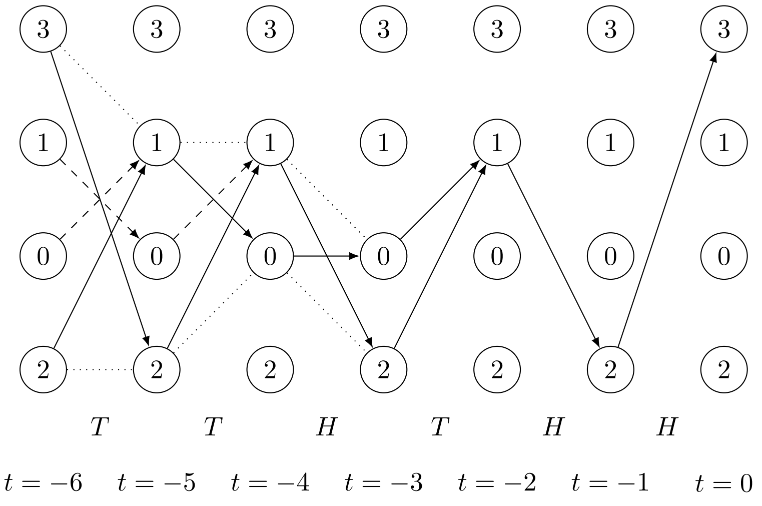

Figure 5: CFTP applied to another random walk, using an antimonotone random mapping.

Dashed lines represent unnecessary Markov chains, started from intermediate states. Dotted lines represent the envelope in which the random mapping works by property of antimonotonicity. We accept 3 as a perfect sample.

2.4 Application to a real world example

We will now show an application to a real world example, in particular to estimate a posterior distribution, not computable directly, with as final aim restoring a degraded image.

2.4.1 Restoration of a degraded image

Consider the restoration of a degraded $N \times N$ binary image, whose pixels $x \in \mathcal{X}$ are either black $(+1)$ or white (-1) [3], with $\mathcal{X}$ state space of all possible $N \times N$ binary images. Suppose that the original image has been contaminated by a Bernoulli noise, i.e. each pixel is reversed to its complementary $(+1 \to -1$ and $-1 \to +1)$ independently and with probability p.

The likelihood of observing y when x is the true image can be expressed as

$$\begin{aligned} p(y \mid x) &= \prod_{i=1}^{N\times N} p^{\mathbb{1}_{\{x_i \neq y_i\}}}(1-p)^{\mathbb{1}_{\{x_i = y_i\}}} \\ &= \prod_{i=1}^{N\times N} p \left(\frac{1-p}{p}\right)^{\mathbb{1}_{\{x_i = y_i\}}} \\ &= p^{N\times N} \left(\frac{1-p}{p}\right)^{\sum_{i=1}^{N\times N} \mathbb{1}_{\{x_i = y_i\}}}, \end{aligned}$$

where we used the fact that $\mathbb{1}_{\{x_i \neq y_i\}} = 1 - \mathbb{1}_{\{x_i = y_i\}}$.

Then we can note that $x_i = y_i \iff x_i y_i = 1$ and $x_i \neq y_i \iff x_i y_i = -1$, since we know that $x_i = \pm 1$ and $y_i = \pm 1$ for all $i \in \{1, 2, \ldots, N \times N\}$. Thus, we can write in a better way the indicator function: $\mathbb{1}_{\{x_i = y_i\}} = \frac{x_i y_i + 1}{2} = \frac{1}{2} x_i y_i + \frac{1}{2}$.

So our previous likelihood becomes

$$\begin{aligned} p(y \mid x) &= p^{N\times N} \left(\frac{1-p}{p}\right)^{\sum_{i=1}^{N\times N}(\frac{1}{2}x_i y_i + \frac{1}{2})} \\ &= p^{N\times N} \left(\frac{1-p}{p}\right)^{\frac{1}{2}N\times N} \left(\frac{1-p}{p}\right)^{\frac{1}{2}\sum_{i=1}^{N\times N} x_i y_i}, \end{aligned}$$

and hence

$$p(y \mid x) \propto \left(\frac{1-p}{p}\right)^{\frac{1}{2}\sum_{i=1}^{N\times N} x_i y_i}. \tag{1}$$

Furthermore, since the images in reality are often composed of relatively smooth pieces, a smooth prior can be chosen as

$$p(x) \propto \exp\left\{\beta \sum_{\langle i,j\rangle} x_i x_j\right\}, \tag{2}$$

where $\langle i, j\rangle$ indicates all neighbor pixel pairs in the image that are adjacent in either the horizontal or the vertical directions, and $\beta > 0$. The coefficient β determines the smoothness of the original image, the larger the β the smoother the prior. Such a prior is, in fact, a simplified Ising model, that is a useful Markov random field model.

Now, the posterior distribution can be expressed as

$$\begin{aligned} p(x \mid y) &\propto p(y \mid x) p(x) \\ &\propto \exp\left\{\beta \sum_{\langle i,j\rangle} x_i x_j + \frac{1}{2}\log\left(\frac{1-p}{p}\right) \sum_{i=1}^{N\times N} x_i y_i\right\}. \end{aligned} \tag{3}$$

To obtain an estimate of the true image, various Bayesian estimators can be used, such as *maximum a posteriori* (MAP) and *marginal posterior mode* (MPM) estimators. Both involve the evaluation of the posterior distribution, which for large N is computationally unavailable. Alternatively, a perfect sampling algorithm such as a monotone CFTP can be applied to simulate the posterior distribution. Recall that a Markov chain is required. For this problem, such chain can be realized using a Gibbs sampler. The Gibbs sampler takes samples from the full conditional distributions which can be obtained from Equation 3 as

$$p(x_k \mid x_{-k}, y) \propto \exp\left\{\beta \sum_{\langle k,i\rangle} x_k x_i + \frac{1}{2}\log\left(\frac{1-p}{p}\right) x_k y_k\right\},$$

where $x_{-k} = (x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_{N\times N})$, i.e. the vector of all pixels except the k-th one.

This formula is derived from 3 by just removing all the stuff not depending on x_k and taking all the rest as normalizing constant. The fact is, such normalizing constant is needed to sample from it. It is possible to use a little trick, given that $x_k = \pm 1$, that is it can have only two values, so that

$$p(x_k = 1 \mid x_{-k}, y) = \frac{1}{C}\exp\left\{\beta \sum_{\langle k,i\rangle} x_i + \frac{1}{2}\log\left(\frac{1-p}{p}\right) y_k\right\}$$
$$p(x_k = -1 \mid x_{-k}, y) = \frac{1}{C}\exp\left\{-\beta \sum_{\langle k,i\rangle} x_i - \frac{1}{2}\log\left(\frac{1-p}{p}\right) y_k\right\}$$

with C normalizing constant.

For simplicity, call

$$\alpha = \beta \sum_{\langle k,i\rangle} x_i + \frac{1}{2}\log\left(\frac{1-p}{p}\right) y_k$$

so that $p(x_k = 1 \mid \mathbf{x_{-k}}, \mathbf{y}) = \frac{1}{C}\exp\{\alpha\}$ and $p(x_k = -1 \mid \mathbf{x_{-k}}, \mathbf{y}) = \frac{1}{C}\exp\{-\alpha\}$.

Since the density must sum up to one, it must hold that

$$\frac{1}{C}\exp\{\alpha\} + \frac{1}{C}\exp\{-\alpha\} = 1,$$

that gives as a result

$$C = \exp\{\alpha\} + \exp\{-\alpha\},$$

than can be plugged again in the densities, obtaining that

$$\begin{aligned} p(x_k = 1 \mid x_{-k}, y) &= \frac{\exp\{\alpha\}}{\exp\{\alpha\} + \exp\{-\alpha\}} \\ &= \frac{1}{1 + \exp\{-2\alpha\}}. \end{aligned}$$

Finally, the full conditional to sample from is

$$p(x_k = 1 \mid x_{-k}, y) = \left(1 + \exp\left\{-2\beta \sum_{\langle k,i\rangle} x_i - \log\left(\frac{1-p}{p}\right) y_k\right\}\right)^{-1}, \tag{4}$$

and $p(x_k = -1 \,|\, x_{-k}, y) = 1 - p(x_k = 1 \,|\, x_{-k}, y)$, with $k = 1, 2, \ldots, N \times N$. The random mapping based on the Gibbs sampler will then be

$$x_k^{(t+1)} = \phi(x_{-k}^{(t)}, u_k^{(t+1)}) = \begin{cases} 1, & \text{if } u_k^{(t+1)} < p(x_k = 1 \mid x_{-k}^{(t)}, y) \\ -1, & \text{otherwise} \end{cases} \tag{5}$$

where $u_k^{(t+1)} \sim \mathcal{U}(0,1)$.

Now impose the partial order $\preceq$ such that $x \preceq x'$ if $x_i = 1$ whenever $x'_i = 1$, $\forall i$. We can see that the distribution in Equation 4 is an increasing function of x_{-k} on the partial order, i.e. $\phi(x_{-k}, u_k) \preceq \phi(x'_{-k}, u_k)$. Hence we can apply the monotonicity and monitor only two Markov chains, one starting in $\tilde{1}$, where all pixels are black, and the other starting in $\tilde{0}$, where all pixels are white. Then we check coalescence of these two chains by time $t = 0$.

We can then apply *marginal posteriod mode* estimation [4] to obtain an estimate of the original image. Setting $p(x_k \,|\, y) = \mathbb{P}\{z \in \mathcal{X} : z_k = x_k \,|\, y\}$, a configuration $\hat{x}$ is called a *marginal posterior mode estimate* (MPME) of x given y if each $\hat{x}_k$ is a (marginal posterior) mode of $x_k \mapsto p(x_k \,|\, y)$.

In other words, it is just necessary to take the mode of every single pixel x_k at a time, given our sample resulting from the CFTP algorithm. The image $\hat{x}$ which has as each pixel the mode of that pixel with respect to all the samples is our estimate.

2.4.2 Implementation and results

We show the performance of the monotone CFTP on a 64×64 binary image of a penguin. Different values for p are chosen, respectively 0.1, 0.2 and 0.3. In all cases, $\beta = 0.45$ and 1000 samples from the posterior distribution were drawn. To recover the original image, MPM estimates were computed by using the obtained samples. In Figure 6 the original image of a penguin is displayed, together with the degraded and recovered images. The misclassification rates e in each experiment was also provided. The rates are computed as the ratio between the number of incorrectly restored pixels and the total number of pixels $N \times N$.

The algorithm was implemented using the R programming language. The code is available at `http://matteosilvestro.altervista.org/files/sds/image_restore.R`.

References

[1] Propp, James Gary and Wilson, David Bruce, "Exact sampling with coupled Markov chains and applications to statistical mechanics," in *Random Structures & Algorithms*, vol. 9, issue 1-2, pp. 223-252, Aug-Sep 1996.

[2] Thönnes, Elke, "A primer on perfect simulation," in *Statistical Physics and Spatial Statistics*, vol. 554 of the series *Lecture Notes in Physics*, pp. 349-378, Dec 2000.

[3] P. M. Djuric, Yufei Huang and T. Ghirmai, "Perfect sampling: a review and applications to signal processing," in *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 345-356, Feb 2002.

[4] Winkler, Gerhard, "Image Analysis, Random Fields and Markov Chain Monte Carlo Methods", ed.2, pp. 25, 2003.

[5] Robert, Christian and Casella, George, "Monte Carlo Statistical Methods", ed.2, pp. 267-543, 2004.

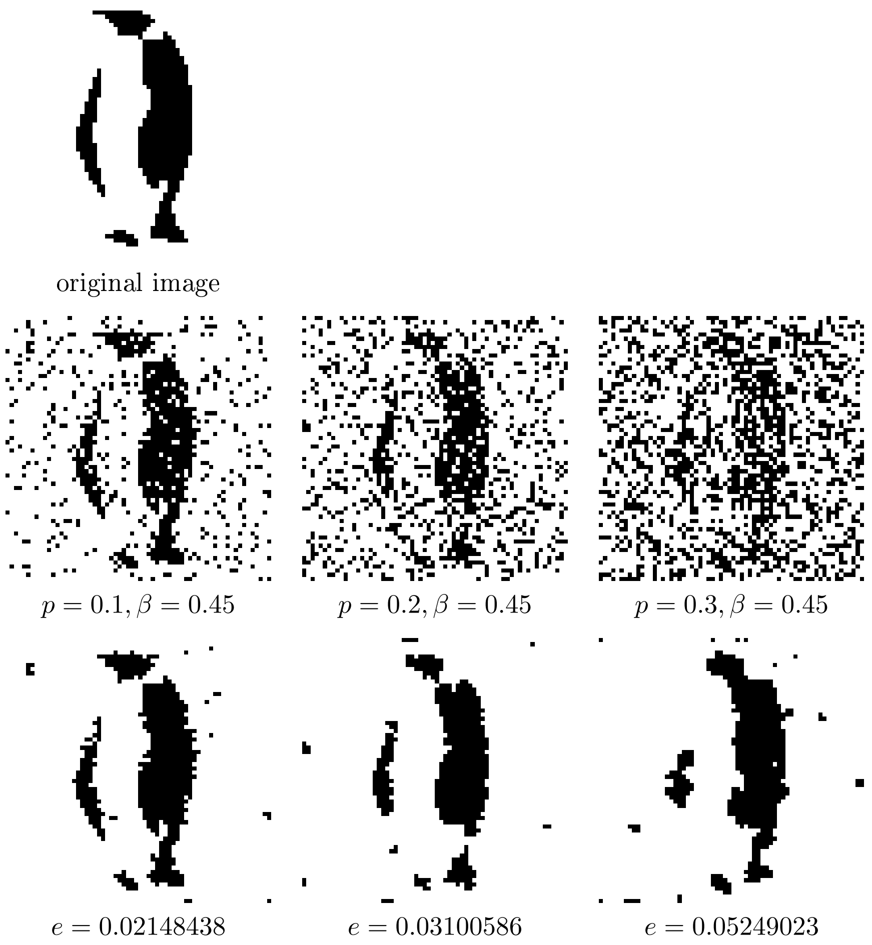

Figure 6: Image in the top row is the original binary image of a penguin. The three images on the second row are the degraded images after Bernoulli noises with different values of p, respectively $0.1, 0.2, 0.3$. The three images on the third row are the restore images, after using a MPM estimate of the posterior. Misclassification rates e are also provided.