UNIVERSITÀ DEGLI STUDI DI MILANO
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di laurea triennale in Fisica

Exact Sampling of Corrugated Surfaces via Coupling From The Past

Relatore: Prof. Sergio Caracciolo
Correlatore: Dott. Andrea Sportiello

PACS: 64.60.Cn
02.50.Ga

Tesi di laurea di
Rinaldi Enrico
matricola 677693

Anno Accademico 2006 – 2007

Contents

Introduzione

Lo studio teorico e la realizzazione pratica di simulazioni numeriche è un ingrediente spesso fondamentale per lo sviluppo di nuove teorie nella fisica contemporanea. Per questo motivo, in questo lavoro di tesi, abbiamo scelto di affrontare l'analisi di algoritmi di simulazione piuttosto recenti, e la loro applicazione a modelli teorici nell'ambito della meccanica statistica.

Una vasta area della teoria delle probabilità riguarda lo studio delle catene di Markov, ovvero una certa classe di processi stocastici caratterizzati da assenza di memoria. Queste sono la teoria matematica alla base dell'implementazione al calcolatore di algoritmi che usano generatori di numeri random. In particolare i metodi di Monte Carlo, basati sulle catene di Markov, sono uno degli strumenti più usati nelle simulazioni in molti campi delle scienze moderne. Estensioni e miglioramenti di questi metodi (in breve MCMC → Markov Chain Monte Carlo) sono molto studiati e il nostro lavoro parte proprio dallo studio di un nuovo algoritmo ideato da James Propp e David Wilson [1] chiamato Coupling From The Past (in breve CFTP).
L'algoritmo CFTP permette un campionamento dello spazio degli stati della catena di Markov che segue esattamente la distribuzione di probabilità desiderata, diversamente dal MCMC che la approssima (seppure con buona precisione). In questo caso si parla allora di *campionamento esatto* contro *campionamento approssimato*. È questo il contenuto trattato nel primo capitolo.

Tra i sistemi fisici a cui è possibile applicare in maniera particolarmente efficace il CFTP ce ne sono alcuni di interesse per la meccanica statistica, come il modello di Ising o il modello di Potts, in cui la distribuzione di probabilità sullo spazio delle configurazioni è la misura di Gibbs del corrispondente sistema termodinamico. Il modello di superfici corrugate da noi studiato ha la stessa caratteristica dei modelli sopracitati, ed è anch'esso un modello di *reticolo*, in breve una griglia di siti ad ognuno dei quali è associata una variabile casuale con una particolare distribuzione di probabilità. Nel secondo capitolo descriviamo il modello e la possibilità di applicare ad esso il CFTP.

Un fine del nostro studio è di ottenere un valore numerico per l'energia libera del sistema, grazie ad una media numerica sulle configurazioni generate con il CFTP. La difficoltà consiste nel fatto che nessun MCMC permette di misurare la funzione di partizione da cui si calcola l'energia libera per un sistema con uno spazio delle configurazioni vasto: infatti il campionamento viene

fatto proporzionalmente alla misura di probabilità e la costante di proporzionalità non può essere determinata in alcun modo.
Noi riusciamo ad aggirare questo problema, parametrizzando opportunamente la misura sullo spazio degli stati, e realizzando una interpolazione su una famiglia di sistemi. In pratica, troviamo una osservabile locale sul reticolo che corrisponde alla derivata dell'energia libera rispetto al parametro che descrive la famiglia di misure; dalla derivata, calcolata numericamente in vari punti della curva del parametro, possiamo ottenere il valore dell'energia libera con una integrazione.

Il risultato finale mostra che è presente una discrepanza tra il nostro calcolo numerico e le previsioni ottenute da S. Majumdar e O. Martin [2], i quali, a nostro avviso, utilizzano una approssimazione di campo medio non del tutto giustificata. Rimarchiamo che l'utilizzo di un campionamento esatto è stato cruciale nel fugare ogni possibile dubbio sulla presenza di bias nel campione, e dare delle barre di errore affidabili in maniera rigorosa.

Riassunto

Capitolo 1

In questo capitolo introduciamo inizialmente i concetti basilari della teoria delle catene di Markov, per poi spiegare brevemente come sia possibile, tramite di esse, estrarre degli stati da uno spazio delle configurazioni secondo una distribuzione di probabilità conosciuta. Il metodo più noto che risale agli inizi degli anni $'50$ è un metodo di campionatura approssimata chiamato *Monte Carlo Markov chain*, il quale permette, data una catena di Markov ergodica, di estrarre stati, nel limite di tempo molto grande, che seguono la distribuzione di equilibrio.

Un metodo più moderno, quello che sarà utilizzato in questo lavoro di tesi, è invece un metodo di campionatura esatta, il quale permette di estrarre stati dallo spazio delle configurazioni esattamente secondo la distribuzione di equilibrio di una catena di Markov, ovvero di determinare a run-time il tempo al di là del quale l'estrazione avviene secondo la distribuzione stazionaria. Questo algoritmo dovuto a Jim Propp e David Bruce Wilson utilizza una catena di Markov ergodica che viene iterata dal passato al presente partendo idealmente da tutti gli stati possibili e si chiama *coupling from the past*.

Ovviamente nel caso di uno spazio degli stati molto grande (ovvero nella maggior parte dei casi di interesse in meccanica statistica) non è praticamente realizzabile un programma che tenga traccia di tutte le catene che partono da tutti gli stati possibili, per cui sembra che il coupling from the past non abbia utilità pratica. Tuttavia è possibile utilizzare una versione di questo algoritmo anche per spazi delle configurazioni molto grandi nel caso in cui questi ultimi possiedano una struttura di ordinamento parziale, e la catena di Markov abbia una matrice di transizione che lo conserva: questa proprietà della catena si chiama monotonicità.

Capitolo 2

In questo capitolo esponiamo il modello del sistema che vogliamo studiare. Si tratta di un *lattice bipartito* in cui esistono dunque due tipi diversi di vertici, ognuno dei quali ha come primi vicini solo vertici dell'altro tipo: li chiameremo vertici **pari** e vertici **dispari** dal momento che,

nella realizzazione del lattice su un reticolo quadrato (bidimensionale), potremo assegnare ad ognuno di essi una coppia di coordinate la cui somma determinerà la parità corrispondente. In ogni vertice è presente una random variabile reale (chiamata *altezza*) appartenente all'intervallo chiuso $[0, 1]$ e distribuita secondo una funzione di probabilità che dipende dal tipo di vertice.

Il sistema in esame è composto da configurazioni che possiedono un massimo locale su ogni sito pari e un minimo locale su ogni sito dispari. Quando la distribuzione di probabilità è la stessa per tutti i siti, ed è la distribuzione uniforme in $[0, 1]$ normalizzata a 1, stiamo considerando il modello di superfici corrugate; l'analisi di alcune sue proprietà statistiche è lo scopo di questa tesi. In particolare ci proponiamo di misurare l'energia libera del sistema nel limite termodinamico (cioè quando il volume del reticolo, su cui è basato il modello, tende all'infinito). Sappiamo tuttavia che un metodo di Monte Carlo non può in alcun modo dare informazioni sull'energia libera a causa del metodo di campionamento, ma può dirci qualcosa della sua derivata. Infatti è possibile, a partire dalla misura di una specifica osservabile locale su reticolo finito, ottenere la derivata dell'energia libera lungo una famiglia a un parametro di misure: vogliamo infatti dimostrare come, introducendo sul reticolo delle misure di probabilità dipendenti da un parametro libero τ, si possa utilizzare il metodo del *reweighting* per trasformare una piccola variazione del parametro libero in una osservabile locale.

Facendo simulazioni a valori diversi del parametro τ otteniamo dei dati che, opportunamente trasformati ed analizzati, corrispondono alla derivata dell'energia libera valutata nel punto τ, per cui possiamo ottenere l'energia libera del modello di superfici corrugate semplicemente integrando numericamente i dati. Questo è possibile se la famiglia di misure scelta ha certe caratteristiche e se non ci sono singolarità nella derivata agli estremi di integrazione.

L'osservabile locale misurata è una densità (quindi una quantità intensiva) di quelli che, in una configurazione, chiameremo difetti, ovvero punti che hanno un'altezza maggiore di $\frac{1}{2}$.

Capitolo 3

L'analisi dei dati, ottenuti da tutte le simulazioni effettuate, è spiegata in questo capitolo. Per prima cosa spieghiamo quali sono stati i requisiti che il programma di simulazione, e prima ancora lo stesso CFTP applicato al nostro modello, hanno dovuto mostrare affinchè noi potessimo proseguire con le simulazioni vere e proprie. Poi analizziamo la distribuzione delle misure effettuate sulle configurazioni *exactly sampled* per mostrare come il particolare approccio di campionamento usato (CFTP, per l'appunto), ci dia la possibilità di trascurare tutti quegli effetti presenti invece nelle simulazioni con MCMC approssimato: stiamo parlando delle correlazioni tra i dati dovute all'utilizzo di catene di Markov e del processo di termalizzazione che costringono ad eseguire analisi dati con tecniche specifiche. Nel nostro caso, dunque, l'analisi degli errori

effettuata sarà quella gaussiana e sapremo quindi dare facilmente una incertezza ad ogni misura.

Di seguito descriviamo il procedimento di estrapolazione della densità di difetti per il valore della dimensione del reticolo che tende all'infinito. Infatti, siccome l'energia libera che ci interessa è quella corrispondente al limite termodinamico del sistema (volume infinito), mentre le nostre simulazioni si basano tutte sull'utilizzo di una porzione quadrata di reticolo (ovviamente finita), dobbiamo ottenere la misura dell'osservabile locale mediante una estrapolazione da un fit sui dati.

Per concludere, mostriamo il calcolo dell'integrazione che ci porta al valore dell'energia libera. Dal momento che i dati rappresentano in qualche modo la derivata dell'energia libera rispetto al parametro τ della famiglia di misure, quello che ci troviamo ad avere in mano dopo l'analisi dei dati, è una serie di punti. Quindi cerchiamo un polinomio che si adatti nel modo migliore all'andamento dei dati e lo integriamo, tenendo conto della propagazione degli errori dovuti al fit.

Capitolo 4

In questo capitolo descriviamo brevemente il codice utilizzato nelle simulazioni. Dal momento che, in fasi diverse del lavoro, abbiamo usato sia codice di Mathematica 6.0 sia di C++, teniamo separati i due linguaggi in due sezioni.

La prima riguarda il codice di Mathematica, usato sia per fare simulazioni premilinari (descritte nel Capitolo 3), inerenti allo studio della dinamica dell'algoritmo di Propp e Wilson nel caso specifico da noi preso in esame, sia per visualizzare graficamente le configurazioni come matrici di punti permettendoci un approccio più immediato all'analisi del sistema.

La seconda invece espone in modo dettagliato le funzioni del programma di simulazione scritto in C++ che implementano l'algoritmo di Coupling From The Past. È una parte importante questa, perchè permette di approfondire aspetti dell'algoritmo che potrebbero non risultare chiari nella teoria generale esposta nel Capitolo 1.

Capitolo 5

Questo capitolo contiene alcune riflessioni finali sul lavoro svolto e degli spunti per future analisi.

Per prima cosa confrontiamo il risultato ottenuto nel capitolo precedente con quello ottenuto da S. Majumdar e O. Martin in [2], cercando di spiegare la discrepanza evidente tra i due.

Poi proseguiamo mostrando la possibilità che il modello di superfici corrugate possa presentare una transizione di fase. Questa idea nasce dallo studio di simulazioni preliminari del sistema effettuate per valori del parametro della misura (τ) al di fuori del range $[1/2, 1]$ consid-

erato in questa tesi. Si è infatti notata la presenza di effetti tipici di transizioni di fase durante l'esecuzione del codice (*critical slowing down*) e nell'analisi delle configurazioni (*clustering*).

1. Exact sampling algorithms

1.1 Theory of Markov chains

Let us begin with a simple example to introduce useful notations. Consider a *random walker* in a very small town consisting of four streets, and four street-corners v_1, v_2, v_3 and v_4 arranged as in figure 1.1. At time 0 the random walker stands in corner v_1. At time 1 he flips a fair coin

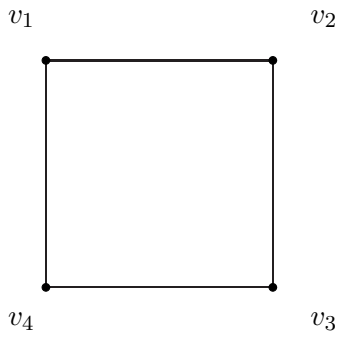


Figure 1.1 Random walker's town

and moves immediatly to v_2 or v_4 according to whether the coin comes up heads or tails. At time 2 he flips the coin again to decide which of the two adjacent corners to move to, with the decision rule that if the coin comes up heads, then he moves one step clockwise, while if it comes up tails, he moves one step counterclockwise. This procedure is then iterated at times 3,4, ... For each n let X_n denote the index of the street-corner at which the walker stands at time n. Hence $(X_0, X_1, \dots)$ is a random process taking values in $\{1, 2, 3, 4\}$. Since the walker starts at time 0 in v_1, we have

$$\mathbf{P}(X_0 = 1) \;=\; 1$$

Next, he will move to v_2 or v_4 with probability $\frac{1}{2}$ each, so that

$$\mathbf{P}(X_1 = 2) \;=\; \frac{1}{2}$$

and

$$\mathbf{P}(X_1 = 4) \;=\; \frac{1}{2}$$

To this end it is useful to consider conditional probabilities. Suppose that at time n the walker stands at, say, v_2. Then we get the conditional probabilities

$$\mathbf{P}(X_{n+1} = v_1 \,|\, X_n = v_2) \;=\; \frac{1}{2}$$

and

$$\mathbf{P}(X_{n+1} = v_3 \,|\, X_n = v_2) \;=\; \frac{1}{2}$$

because of the coin-fliping mechanism for deciding where to go next. In fact, we get the same conditional probabilities if we condition further on the full history of the process up to time n, i.e.,

$$\mathbf{P}(X_{n+1} = v_1 \,|\, X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = v_2) \;=\; \frac{1}{2}$$

and

$$\mathbf{P}(X_{n+1} = v_3 \,|\, X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = v_2) \;=\; \frac{1}{2}$$

for any choice of $i_0, \ldots, i_{n-1}$. (This is because the coin-flip at time $n+1$ is independent of all previous coin-flips, and hence also independent of $X_0, \ldots, X_n$.) This phenomenon is called the **memoryless property**, also known as the **Markov property**: the conditional distribution of X_{n+1} given $(X_0, \ldots, X_n)$ depends only on X_n.

Another interesting feature of this *random process* is that the conditional distribution of X_{n+1} given that $X_n = v_2$ (say) is the same for all n. This property is known as **time homogeneity**. These observations call for a general definition:

Definition 1.1. *Let P be a $(k \times k)$-matrix with elements $\{P_{i,j} : i, j = 1, \ldots, k\}$. A random process $\{X_t\} = (X_0, X_1, \ldots)$ with finite state space $\mathcal{S} = \{s_1, \ldots, s_k\}$ is said to be a **(homogeneous) Markov chain with transition matrix** P, if for all n, all $i, j \in \{1, \ldots, k\}$ and all $i_0, \ldots, i_{n-1} \in \{1, \ldots, k\}$ we have*

$$\begin{aligned} \boldsymbol{P}(X_{n+1} = s_j \,|\, X_0 = s_{i_0}, X_1 = s_{i_1}, \ldots, X_{n-1} = s_{i_{n-1}}, X_n = s_i) & \\ = \; \boldsymbol{P}(X_{n+1} = s_j \,|\, X_n = s_i) & \qquad (1.1) \\ = \; P_{i,j} & \end{aligned}$$

The elements of the transition matrix P are called transition probabilities.

Another important characteristic (besides the transition matrix) of a Markov chain $(X_0, X_1, \ldots)$ is the **initial distribution**, which tells us how the Markov chain starts. It is represented by a row vector $\mu^{(0)}$ given by

$$\begin{aligned} \mu^{(0)} \;&=\; (\mu_1^{(0)}, \mu_2^{(0)}, \ldots, \mu_k^{(0)}) \\ &=\; (\mathbf{P}(X_0 = s_1), \mathbf{P}(X_0 = s_2), \ldots, \mathbf{P}(X_0 = s_k)) \end{aligned}$$

It turns out that once we know the initial distribution $\mu^{(0)}$ and the transition matrix P, we can compute all the distributions $\mu^{(1)}, \mu^{(2)}, \dots$ of the Markov chain due to the following result

Theorem 1.1. *For a Markov chain $(X_0, X_1, \dots)$ with state space $\{s_1, \dots, s_k\}$, initial distribution $\mu^{(0)}$ and transition matrix P, we have that for any n the distribution $\mu^{(n)}$ at time n satisfies*

$$\mu^{(n)} = \mu^{(0)} P^n \tag{1.2}$$

i.e. $\mu_i^{(n)} = \sum_{j=1}^{k} \mu_j^{(0)} P_{j,i}^N$ *for all* $i \in \{1, \dots, k\}$.

Lets see what happen if we look at the distribution of X_n in any nontrivial Markov chain: X_n will keep fluctuating infinitely many times as $n \to \infty$ and therefore we cannot hope to get results about X_n, but we may hope that its *distribution* settles down to a limit. I f this limit exists we call it a **stationary distribution** for the Markov chain. The general definition is as follows

Definition 1.2. *Let $\{X_t\}$ be a Markov chain with state space $\mathcal{S} = \{s_1, \dots, s_k\}$ and transition matrix P. A row vector $\pi = (\pi_1, \dots, \pi_k)$ is said to be a* stationary distribution *for the Markov chain (or for the transition matrix P), if it satisfies*

(i) $\pi_i \geq 0$ *for* $i = 1, \dots, k$, *and* $\sum_{i=1}^{k} \pi_i = 1$ *and*

(ii) $\pi P = \pi$, *meaning that* $\sum_{i=1}^{k} \pi_i P_{i,j} = \pi_j$ *for* $j = 1, \dots, k$.

Property (i) simply means that π should describe a probability distribution on $\mathcal{S}$.
Property (ii) implies that if the initial distribution $\mu^{(0)}$ equals π, then the destribution $\mu^{(1)}$ satisfies

$$\mu^{(1)} = \mu^{(0)} P = \pi P = \pi$$

and by iterating we see that $\mu^{(n)} = \pi$ for every n.

Now we introduce two important properties which play a key role in the study of **stationary distributions** of Markov chains: **irreducibility** and **aperiodicity**. These are the properties that guarantee that all the states of the Markov chain can be reached from all the others. Hence, we say that two states s_i,s_j *intercommunicate* (denoted as $s_i \leftrightarrow s_j$) if the chain has a positive probability $\mathbf{P}(X_{m+n} = s_j \,|\, X_m = s_i) > 0$ for a certain n and if the same happens when we change i and j. This takes us directly to the definition

Definition 1.3. *A Markov chain $\{X_t\} = (X_0, X_1, \dots)$ with state space $\mathcal{S} = \{s_1, \dots, s_k\}$ and transition matrix P is said to be* irreducible *if for all $s_i, s_j \in \mathcal{S}$ we have that s_i and s_j intercommunicate.*

The *period* $d(s_i)$ of a state $s_i \in \mathcal{S}$ is defined as

$$d(s_i) \;=\; \gcd(n \geq 1 \,:\, (P^n)_{i,i} > 0)$$

If $d(s_i) = 1$, then we say the state s_i is *aperiodic*.

Definition 1.4. *A Markov chain is said to be* aperiodic *if all its states are aperiodic.*

A Markov chain with these two properties is called **regular** (or **ergodic** if it also has $P_{i,i} > 0 \,\forall\, s_i \in \mathcal{S}$) and is of primary importance in every random algorithm application because of the following theorems

Theorem 1.2. *For any* regular *Markov chain, there exists at least one stationary distribution.*

Theorem 1.3. *Let $\{X_t\}$ be a regular Markov chain with state space $\mathcal{S} = \{s_1, \ldots, s_k\}$, transition matrix P and an initial distribution $\mu^{(0)}$. Then, for any distribution π which is stationary for the transition matrix P, we have*

$$\lim_{n\to\infty} d_{TV}(\mu^{(n)}, \pi) = \lim_{n\to\infty} \left[\sum_{i=1}^{k} \left| \mu_i^{(n)} - \pi_i \right| \right] = 0 \tag{1.3}$$

Here, d_{TV} is a metric on probability distribution called **total variation distance** and equation (1.3) is often referred to as the Markov chain approaching *equilibrium* as $n \to \infty$, equilibrium because, as a corollary, also $\sum_i \left| \mu_i^{(n+t)} - \mu_i^{(n)} \right| \to 0$.

Theorem 1.4. *Any* regular *Markov chain has exactly one stationary distribution.*

If we add another property to a regular Markov chain we can get a stronger condition on the stationary distribution.

Definition 1.5. *Let $\{X_t\}$ be a Markov chain with state space $\mathcal{S} = \{s_1, \ldots, s_k\}$ and transition matrix P. A probability distribution π on $\mathcal{S}$ is said to be* ***reversible*** *for the chain (or the transition matrix P) if for all $i, j \in \{1, \ldots, k\}$ we have*

$$\pi_i P_{i,j} \;=\; \pi_j P_{j,i} \tag{1.4}$$

The Markov chain is said to be reversible *if there exists a reversible distribution for it.*

This looks like a strong form of equilibrium, as the following result further suggests

Theorem 1.5. *Let $\{X_t\}$ be a Markov chain with state space $\mathcal{S} = \{s_1, \ldots, s_k\}$ and transition matrix P. If π is a reversible distribution for the chain, then it is also a stationary distribution for te chain.*

The reader interested in the proofs of the previous theorems, can find them in Chapter 5 of [3].

1.2 Coupling from the past algorithm

Markov chains introduced in Section 1.1 are useful in the cases we have to simulate a random variable X distributed according to a given probability distribution π on a state space $\mathcal{S}$. **Markov chain Monte Carlo (MCMC)** method is the name of all algorithms using Markov chain to simulate state spaces and originates in physics in the 1950's.
The idea is the following: suppose we can construct an irreducible and aperiodic Markov chain whose stationary distribution is π. If we run the chain with arbitrary initial distribution, then Theorem 1.3 guarantees that the distribution of the chain at time n converges to π, as $n \to \infty$. Hence, if we run the chain for sufficently long time, then the distribution of X_n will be very close to π. Of course this is just an approximation, but the point is that the way μ^n approximates π is exponential in n, under mild hypotheses.

Provided that you have a technique to construct a Markov chain knowing only the equilibrium distribution from which you want to sample, the key question is "How long is a sufficently long run?". To decide the number of steps various convergence tests have been proposed but they are generally difficult to find, and upper bounds are often too weak to be of a practical interest. In particular they all suffer from the possibility that the Markov chain reaches a *metastable* state, so even if the chain seems to be stable, there may exist domains in the state space which have not yet being visited; in this case, the only way to ensure that we have convergence is to check that we have visited every state, but if we could do this in a practical way there generally would be no need to do the simulation at all!.
A possible solution of the problem came in a 1996 paper by Propp and Wilson [4]. They yielded an algorithm which simultaneously produces an output whose distribution is *exactly* the equilibrium distribution π, and determines automatically when to stop running the Markov chain. This method is denoted **exact** or **perfect sampling** versus the approximate sampling of MCMC. Propp and Wilson algorithm introduces a new intuitive idea, that is *backward coupling* or better what they called **coupling from the past (CFTP)**. This name anticipates the fact that we have to work not with one, but several Markov chains at once and that the simulations run from some time in the past up to time 0.
Consider to simultaneously run $N = |\mathcal{S}|$ copies of the Markov chain from every possible initial state; let the rule to update the Markov chain be the same for all the chains; this implies in particular that it should be configuration-independent. We say that two chains have **coalesced** if at some step they reached the same state. From that step onward the two chains will follow the same path in the state space, due to the fact that the same updating function is used for all the chains. Now, suppose all N chains have coalesced, such that the effect of the initial state has worn off. A simple MCMC algorithm (*forward simulation*) would then be to start N chains in every possible initial state and output the current at once all chains had coalesced. However,

this will not yield a correct algorithm, because if states in the Markov chains have a unique predecessor we will obtain *biased* samples (see example at page 79 in [3]). The solution to the problem of biased samples comes from the possibility to run the chains from the past to the present instead of running them from the present to the future.

To better understand the meaning of *backward simulations* and *coalescence* in this case we introduce a new useful notation.
Consider a situation where an ergodic Markov chain with transition matrix P is iterated for M(fixed) step via an **update function**

$$\phi : \mathcal{S} \times \mathcal{U} \to \mathcal{S} \quad \text{such that} \quad \mathbf{P}(\phi(s_i, U) = s_j) = P_{i,j} \tag{1.5}$$

where U is a random number and for convenience we assume that the start and finish of the simulation are designated as time $-M$ and time 0: we call this *fixed-time forward simulation.*
An approximate sampling procedure whose output is governed by the same probability distribution as M-step forward simulation is *fixed-time backward simulation* in which the iteration starts at time 0 and moves into the past stopping at time $-M$; this procedure takes fewer steps then forward simulation when M is large. Since the state of the chain at time -1 is determined by its history from time $-M$ to time -1, it is unknown to us when we begin our backward simulation; therefore we run the chain from time -1 to time 0 not just once but $|\mathcal{S}|$ times, once for each of the $|\mathcal{S}| = k$ states of the chain that might occur at time -1. That is we can define a map $f_{-1} : \mathcal{S} \to \mathcal{S}$ by putting $f_{-1}(s) = \phi(s, U_0)\ \forall\, s \in \mathcal{S}$. Similarly, for all time t with $-M \leq t < -1$, we can define a random map f_t by putting $f_t(s) = \phi(s, U_{t+1})$ or better a recursively defined function $F_t^0(s)$ whose values are actually *functions* from the state space to itself. The output of fixed-time simulation is given by $F_{-M}^0(s^*)$ where s^* is the initial state and $F_{t_1}^{t_2}$ is defined as the composition $f_{t_2-1} \circ f_{t_2-2} \circ \cdots \circ f_{t_1+1} \circ f_{t_1}$. We see that under backward simulation there is no need to keep track of all the maps f_t individually, but rather we only need to keep track of the composition F_t^0, which can be updated via the rule

$$F_t^0(s, U_0, U_{-1}, \ldots, U_{t+1}) = F_{t+1}^0(\phi(s, U_{t+1}), U_0, U_{-1}, \ldots, U_{t+2}) = F_{t+1}^0 \circ f_t \tag{1.6}$$

More to the point is the observation that if the map $F_{t'}^0$ is a **constant map** for some $t' \in [-M, 0)$, with

$$F_{t'}^0(s_i) \;=\; F_{t'}^0(s_j) \quad \forall\, i, j \in \{1, \ldots, k\},$$

then this will remain true from that point onward (that is for all $t \leq t'$), and the value $F_{-M}^0(s^*)$ must equal the common value of $F_{t'}^0(s_i)\ i \in \{1, \ldots, k\}$; there is no need to go back to time $-M$ once the composed map $F_{t'}^0$ has become a constant map.
It must be clear that a constant map is *equivalent* with coalescence of all chains with different initial states. However, the value of the mapping $F_{t'}^0$ is not necessarily the same as the state

of the chains at the time of coalescence: that is the chain may coalesce before time 0. This explains why the idea of simulationg from the past up to the present is crucial; on the opposite, if we were doing forward simulation from time 0, then the time t' at which the map $F_0^{t'}$ became constant would coincide with the time of coalescence. Also the value $F_0^{t'}$ is not necessarily the same as the value F_0^{∞}, the latter being a correct sample from π.

Now, by removing the cut-off M setting it equal to infinity we are achieving an output $F^0_{-\infty}$ distributed according to the stationary distribution π of the Markov chain. An algorithm could then be stated as follows

$t \leftarrow 0$
`repeat`
$t \leftarrow t-1$
`for all` $s \in \mathcal{S}$
$F_t^0(s, U_0, U_{-1}, \dots, U_{t+1}) \leftarrow F_{t+1}^0(\phi(s, U_{t+1}), U_0, U_{-1}, \dots, U_{t+2})$
`until` F_t^0 `is a constant map`
`return the unique value` $F_t^0(\cdot)$

This is just what CFTP does, and if we want to be more specific we can ask ourselves if the process always terminates or if the loop above is unbounded. The answer is in the following theorem and under certain hypotheses it can be more precise.

Theorem 1.6. *With probability* 1 *the CFTP protocol returns a value, and this value is distributed according to the stationary distribution of the Markov chain.*

Proof. Since the chain is ergodic, there is an n such that for all states i and j, there is a positive chance of going from i to j in n steps (see Definition 1.3). Hence for each t, $F_{t-n}^t(\cdot)$ has a positive chance of being constant.
Since each of the maps $F_{-n}^0(\cdot), F_{-2n}^{-n}(\cdot), \dots$ has some positive probability $\epsilon > 0$ of being constant, and since these events are independent, it will happen with probability 1 that one of these maps is constant, in which case F_{-M}^0 is constant for all sufficiently large M.
When the algorithm reaches back M steps into the past, it will terminate and return a value that we may as well call $\bar{F}^0_{-\infty}$. Note that $\bar{F}^1_{-\infty}$ is obtained from $\bar{F}^0_{-\infty}$ by running the Markov chain one step, and that $\bar{F}^1_{-\infty}$ and $\bar{F}^0_{-\infty}$ have the same probability distribution. Together these last two assertions imply that the output $\bar{F}^0_{-\infty}$ is distributed according to the unique stationary distribution π. □

This is the so-called $0-1$ law, meaning that the probability for the termination of CFTP algorithm must be either 0 or 1; hence, it is enough to show that $\mathbf{P}(\text{algorithm terminates}) > 0$ in order to show $\mathbf{P}(\text{algorithm terminates}) = 1$. However Theorem 1.6 does not bound the time

of the iterations and this can bring to computational problems or biased samples: we could see that, choosing a bad update function, the algorithm may fail to terminate and interrupting a long run introduces a bias. Another natural question about Propp and Wilson algorithm is what do to in the case the state space $\mathcal{S}$ is huge, that is how can we run the chains from all possible starting values: we shall see an ingenious technique in Section 1.3.

It is sometimes desiderable to view the process as an iterative one, in which one successively starts up k copies of the chain at times $-1, -2, \dots$, until one has gone sufficently far back in the past to allow the different histories to coalesce at time 0. In this setup it is very important to bear in mind that the random variables $U_t \in \mathcal{U}$ that one uses in going from time $t-1$ to time t must be the same for the many sweeps one might make through this time-step for all the chain's copies. If not, biased samples would again be obtained. We may accomplish this needing storing the random variables U_t or, if our U_t's are given by some pseudo-random number generator, we may suitably reset the generator's seed in order to re-use the same variables in the right sweeps. In Figure 1.2 we consider a simple example with a sequence of times $(t_1, t_2, \dots) = (1, 2, 4, \dots)$ such that the coupled chain runs during a step following the update function $f_{-t}(\cdot) = \phi(\cdot, U_{-t})$. Moreover we take a very small state space $\mathcal{S} = \{s_1, s_2, s_3\}$ in order to keep easily track of chain's trajectories. Since $t_1 = 1$, we start running the chain from time -1 to time 0. Suppose (as in the top part of Figure 1.2) that it turns out that

$$\begin{cases} \phi(s_1, U_0) = s_1 \\ \phi(s_2, U_0) = s_2 \\ \phi(s_3, U_0) = s_1 \end{cases}$$

Hence the state at time 0 can take two different values (s_1 or s_2) depending on the state at time -1, and we therefore try again with starting time $-t_2 = -2$. We then get

$$\begin{cases} \phi(\phi(s_1, U_{-1}), U_0) = \phi(s_2, U_0) & = s_2 \\ \phi(\phi(s_2, U_{-1}), U_0) = \phi(s_3, U_0) & = s_1 \\ \phi(\phi(s_3, U_{-1}), U_0) = \phi(s_2, U_0) & = s_2 \end{cases}$$

which again produces two different values at time 0. Note the use of U_0 for the step from time -1 to time 0 as before. Continuing the algorithm forced us to start the chains from an earlier starting time $-t_3 = -4$. This yields

$$\begin{cases} \phi(\phi(\phi(\phi(s_1, U_{-3})U_{-2}), U_{-1}), U_0) = \cdots & = s_2 \\ \phi(\phi(\phi(\phi(s_2, U_{-3})U_{-2}), U_{-1}), U_0) = \cdots & = s_2 \\ \phi(\phi(\phi(\phi(s_3, U_{-3})U_{-2}), U_{-1}), U_0) = \cdots & = s_2 \end{cases}$$

This time we get the state s_2 at time 0 regardless of the starting value at time -4. The algorithm therefore stops with output equal to s_2. Note that if we would continue and run the chains starting at times $-8, -16$ and so on, then we will keep getting the same output forever.

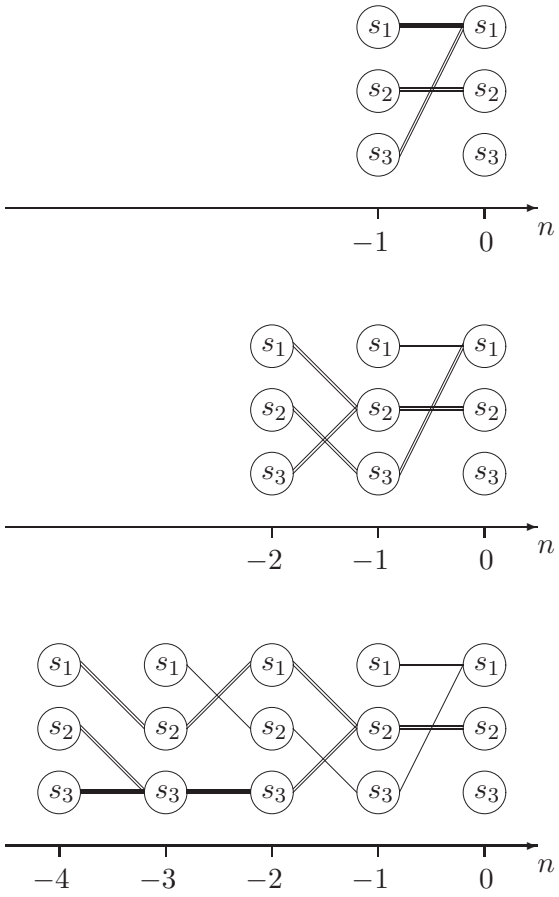


Figure 1.2 A run of the CFTP algorithm with $t_1 = 1$, $t_2 = 2$, $t_3 = 4$, and state space $\mathcal{S} = \{s_1, s_2, s_3\}$. Transitions that are carried out in the running of the algorithm are indicated with double lines; others are simple.

1.3 The monotone case

As the wise reader can see the CFTP algorithm presented so far will not be of any practical utility if the state space $\mathcal{S}$ is large: if a Markov chain has 2^N states, for N large, it will not be feasible to visit all of them. Efficient simulation is possible only if we can reduce the number of

chains to run. Many Markov chains from which we wish to sample and which are of particular interest in Statistical Mechanics, have a special property called **monotonicity**. The CFTP algorithm can take advantage of this condition as we will soon see, becoming very useful for statistical simulations in huge state spaces.

Suppose now that the (possibly huge) state space $\mathcal{S}$ of our ergodic Markov chain admits a natural *partial ordering* $\preceq$: we say that $(\mathcal{S}, \preceq)$ is a **poset** (partially ordered set).

Definition 1.6. *Let P be a transition matrix on a poset $(\mathcal{S}, \preceq)$. We say P is* monotone *if P preserves the partial order,i.e. if $P_{i,\cdot} \preceq P_{j,\cdot}$ stochastically whenever $s_i \preceq s_j$*

The product of monotone transition matrices is monotone.
Monotonicity of P is guaranteed when one can couple transitions using a monotone transition rule.

Definition 1.7. *A* monotone transition rule *for a transition matrix P on a poset $(\mathcal{S}, \preceq)$ is an update function $\phi : \mathcal{S} \times \mathcal{U} \to \mathcal{S}$ such that $\boldsymbol{P}(\phi(s_i, U) = s_j) = P_{i,j}$ and $\phi(s_i, U) \preceq \phi(s_j, U)$ for all $U \in \mathcal{U}$ whenever $s_i \preceq s_j$.*

When a monotone transition rule exists, one can simultaneously generate transitions from various states in such a way as to maintain ordering relations for each realization.
If, in addition, we can find minimal and maximal elements $\hat{0}, \hat{1}$ with $\hat{0} \preceq s \preceq \hat{1}$ for all $s \in \mathcal{S}$, that is, if the poset is a *lattice*, we have the **monotone case**.

We can now proceed as in Section 1.2 defining a random map

$$\Phi_{t_1}^{t_2}(s, U) \;=\; \phi(\phi(\dots(\phi(s, U_{t_1}), U_{t_1+1}), \dots, U_{t_2-2}), U_{t_2-1})$$

where U is short for $(U_0, U_{-1}, \dots)$. If $U_{-T}, U_{-T+1}, \dots, U_{-1}, U_0$ have the property that

$$\Phi_{-T}^{0}(\hat{0}, U) \;=\; \Phi_{-T}^{0}(\hat{1}, U) \tag{1.7}$$

then the monotonicity property assures us that $\Phi_{-T}^{0}(s, U)$ takes on their common value for all $s \in \mathcal{S}$. This frees us from the need to consider trajectories starting in all $|\mathcal{S}|$ possible initial states; two states will suffices because the others are"sandwiched" and if these two coalesce then so will all the others.

In the following chapters we are going to deal with a continuous state space and it is obviously impossible to track the chains starting from all possible states. However there exists a possible formulation of this theory dealing with continuous variables as explained in the work [5]. We can now sketch an argument for which coalescence is assurable even for configurations in continuous space. The point is that we are using the same set $\{U_0, U_{-1}, \dots\}$ of random numbers on the two coupled chains, so that when we update a configuration of a chain, we use the exactly

same variable to update the other configuration too. This lead to the fact that, if the update is accepted in both chains, the updated variable is the same for both; thus, even for real-valued variables, we have a non-zero probability for the configurations to be equal.

2. Model of corrugated surfaces

2.1 Random minima model

Our model consists in a lattice (the underlying space for this problem) where the number of minima (maxima) is the highest possible, which is to say we have two distinct types of sites on the lattice, one on which always resides a minimum and one on which always resides a maximum of the energy, and these sites are alternated.

More generally speaking, we have a bipartite lattice $\mathcal{L}$ where V_e and V_o are the sets of distinct types of alternated sites, even and odd, and E is the set of edges linking two sites (one even and one odd): $\mathcal{L} = (V_e, V_o; E)$.
We denote the set of all sites $V = V_e \cup V_o$ and we label each site with a latin index $i \in \{1, \dots, V\}$; moreover we say two sites i, j are adjacent if the edge $e = (i, j)$ is in E (in this case we write $e = \langle i, j \rangle$).

In such a defined space we associate to each site a *height* (or *energy*, if one prefers) x_i which can take values in the real interval $[0, 1]$; according to the site parity, x_i is choosen from the following normalized probability distributions with support on $[0, 1]$

$$\begin{aligned} \mu(x_i) &= \mu_e(x_i) \quad \forall\, i \in V_e \\ \mu(x_i) &= \mu_o(x_i) \quad \forall\, i \in V_o \end{aligned}$$

and the two measures are such that $\mu_e(x_i) = \mu_o(1 - x_i)$. This choice of symmetry is made for simplicity, and is not strictly necessary.

We write $\mathbf{X} = \{x_i\}_{i \in V}$ to denote a configuration and we call this configuration **feasible** if for any edge $\langle i, j \rangle$, with $i \in V_e$ and then $j \in V_o$, we have $x_i \geq x_j$.[1] So we obtain the partition function integrating the measure $\mu(x)$ over all the feasible configurations

$$\mathcal{Z} = \int \prod_{i \in V_e} d\mu_e(x_i) \int \prod_{j \in V_o} d\mu_o(x_j) \prod_{\langle i,j \rangle} \Theta(x_i - x_j) \tag{2.1}$$

[1] This model can be seen a a limit $\lambda \to \infty$ of the general model in which all configurations are allowed, but we have a price λ for each minimum (maximum).

The integration runs over all the possible combinations of V real numbers in $[0,1]$, that is $[0,1]^V$ and the last product in the integral runs over the nearest neighbours $j \in V_o$ of each even site $i \in V_e$.

If a configuration is not feasible it means there is an even site i such as x_i is not a local maximum and the Θ function in (2.1) prevents this configuration from being counted in the partition function. We can also note that if the measure on the even sites gives zero chance to small x_i to appear (i.e. $\mu_e(x_i) = 0$ if $x_i \in [0, 1/2]$), all the Θ's are automatically satisfied and the partition function reduces to the product of the one-site measure's normalization which is 1. We will call this the *trivial model*.

Our final goal is to calculate the intensive **free energy** f of the system, which is strictly related to the partition function by the formula

$$f = \frac{1}{V} \ln \mathcal{Z}(V) \tag{2.2}$$

where V is the number of sites, in the situation where the measure is uniform in $[0,1]$, that is all the real numbers in the interval are equiprobable (for a statistical mechanics system, the free energy is defined as $-\frac{1}{\beta V} \ln \mathcal{Z}(\beta; V)$. Here we omitted the sign for convenience, and the factor β because our system hasn't got a temperature). This last case is the classic model of **corrugated surfaces** (for further readings see bibliography in [2]).

2.2 Lattice and poset structure

Our model is easily implemented on a planar square portion of the lattice $\mathcal{L}$ and the following is the starting point of every simulation we will make:

- a grid of L^2 points with 2 coordinates $(i, j) \in \{1, 2, \ldots, L\}^2$. Even sites are those such that $i + j$ is even (see Figure 2.1);
- a real number in $[0, 1]$ is placed over each point. Every even site must have a value larger than each of the four neighbouring odd sites;

Every realization of the second item above is a feasible configuration $\mathbf{X}$ and we can perform on this whatever kind of measure we want.

Now our purpose is to find a CFTP algorithm, especially a Markov chain, whose states are all the feasible configurations of the corrugated surfaces model, but we want a monotone CFTP to concretely realize our simulations. To have a chance to find a such a monotone Markov chain (see Definition 1.6) we should guess a natural partial ordering on the chain's state space and now we show how our model in Section 2.1 can be seen as a poset and how we can define a monotone transition rule (in the sense of Definition 1.7) for the Markov chain.

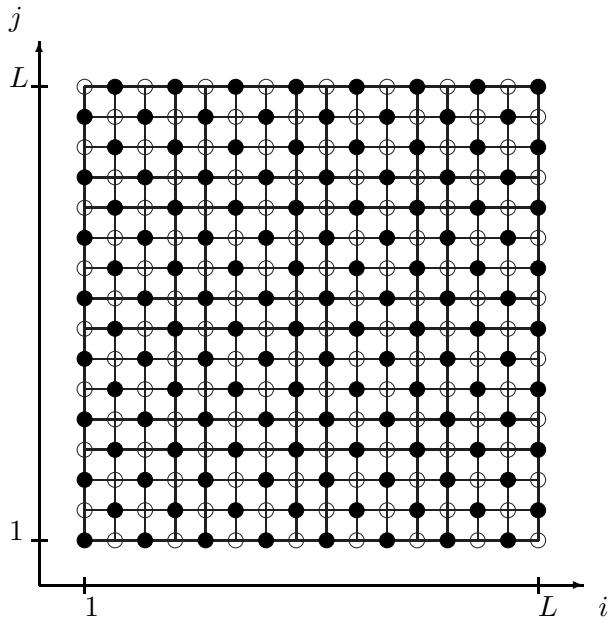


Figure 2.1 $L \times L$ grid with a white circle on each odd site and a black one on each even site

Consider two configurations $\mathbf{X}$ and $\mathbf{Y}$ in the space of feasible configurations $\mathcal{S}$. We say that they are partially ordered ($\mathbf{X} \preceq \mathbf{Y}$) if $x_v \leq y_v$ for any $v \in V_e$, and $x_w \geq y_w$ for any $w \in V_o$. The structure of lattice is completed by the presence of an *infimum* and a *supremum* state

$$\begin{aligned} \hat{0} &: \mathbf{X} \text{ such that} \quad x_v = 0 \qquad \forall\, v \in V \\ \hat{1} &: \mathbf{X} \text{ such that} \quad x_v = 1 \qquad \forall\, v \in V \end{aligned} \tag{2.3}$$

so that $\forall\, \mathbf{X} \in \mathcal{S}$ we have $\hat{0} \preceq \mathbf{X} \preceq \hat{1}$.

The transition rule from $\mathbf{X} = \{x_v\}_{v \in V}$ to $\mathbf{X}' = \{x'_v\}_{v \in V}$ (i.e. the dynamics of the Markov chain), which preserves the lattice structure for any one-site measure μ, is the following:

$$\text{define} \qquad x_v^{(n)} = \begin{cases} \max\limits_{(w \text{ neigh } v)} x_w & \text{if} \quad v \in V_e \\ \min\limits_{(w \text{ neigh } v)} x_w & \text{if} \quad v \in V_o \end{cases}$$

1. choose randomly and uniformly a vertex $v \in V$ (i.e. sample 2 integers in $\{1, 2, \ldots, L\}$);
2. choose randomly a real number z in $[0, 1]$ according to the measure $\mu(z)$ (the *test height*);
3. - if v is even: we substitute x_v with z setting $x'_v = z$ if $z \geq x_w$ for all w neighbours of v, that is $z \geq x_v^{(n)}$. Otherwise, we leave X_v;

- if v is odd: we use $1-z$ instead of z (because $\mu_o(x) = \mu_e(1-x)$), and we set $x'_v = (1-z)$ if $(1-z) \leq x_w$ for all w neighbours of v, that is $(1-z) \leq x_v^{(n)}$. Otherwise, we leave x_v.

We can easily show that if $\mathbf{X} \preceq \mathbf{Y}$, then the same relation is true for their one-step evolution in the Markov chains due to the above transition rule ($\mathbf{X}' \preceq \mathbf{Y}'$) and this is the important topic in the monotone CFTP algorithm because starting from $\hat{0}$ and $\hat{1}$ we have coalescence for all the states if the two of them coalesce.
The two cases (even and odd site) must be treated in different but symmetric ways. Suppose first we are updating an even site $v \in V_e$; all the other sites in $\mathbf{X}$ and in $\mathbf{Y}$ (we have to use the same update for both the chains) remain unchanged, namely the neighbours of v are the same before and after the sampling of z, that is $x_v^{(n)} \leq y_v^{(n)}$. If $z \geq y_v^{(n)}$, then it is a local maximum and we set $y'_v = z$; at the same time it is also true that $z \geq x_v^{(n)}$, because of the ordering of the configurations, and we can write $x'_v = z$. However we can have $z \geq x_v^{(n)}$ but $z \leq y_v^{(n)}$, and this leads to $x'_v = z$ and $y_v = y'_v$, but still $z \leq y_v$ and the partial order of the configurations $\mathbf{X}' \preceq \mathbf{Y}'$ is fulfilled. Of course, if $z \leq y_v^{(n)}$ and $z \leq x_v^{(n)}$, no site is updated and $\mathbf{X}' \preceq \mathbf{Y}'$ is obviously true. Now suppose the site is odd: we must repeat all the reasonings above changing z with $1-z$ and inverting all the binary relations; what we see is that also in this second case $\mathbf{X}' \preceq \mathbf{Y}'$.

The Markov chain that follows the dynamic given by the above transition rule is ergodic, aperiodic and irreducible. These properties guarantee that this chain has a unique equilibrium distribution, and that we will sample just from that distribution. We want to remark here that the CFTP algorithm, when the chains have coalesced, outputs only a configuration, and in Chapter 1 we have demonstrate that the outputted configuration follow the equilibrium distribution of the ergodic Markov chain. If we run the algorithm for, we say, N times, we end with a statistical ensemble of N configurations without any correlations between them. This is true if we pay attention to use a different initial seed for the random number generator in each different run. For better understanding of the algorithm we suggest to read now the code implementation in Chapter 4.

Now, although we can say the monotone case arise (see Section 1.3), and that this is true for all the distribution $\mu(x)$ so that a fast CFTP algorithm can be implemented, we have 2 open questions we would answer to:

- can we find a new parametrization of the model so that it will be parity invariant, so that there is a sigle measure to sample from and all the sites are of the same kind? We prefer this new situation because all the relations can be written more easily.

- is the transition rule one of the best to make the chains coalesce or the CFTP loops remains unbounded? This aspect needs test simulations with Mathematica as we will see

in section 3.1.1.

The first question has a simple answer: we can transform a configuration $\mathbf{X}$ with the notation of Section 2.1 in a new one $\tilde{\mathbf{X}}$ which has at every site a height

$$\tilde{x}_v = \begin{cases} x_v & \text{if} \quad v \in V_o \\ 1 - x_v & \text{if} \quad v \in V_e \end{cases} \tag{2.4}$$

With these new configurations $\tilde{\mathbf{X}}$ we have a symmetrical representation of even and odd sites, and we have only one distribution for each height of a site from which we have to sample. In this notation we have another explicit form for the Θ's in the partition function because now a feasible configuration hasn't got local minima and maxima but has to satisfy the constraint

$$\Theta(1 - x_v - x_w) \qquad \forall \langle v, w \rangle \tag{2.5}$$

that is the sum over every nearest neighbour must be less than 1. This condition is easily implemented in the code of the simulation as we will see in Chapter 4 and this is why we deal with configurations without maxima and minima.
The expression of the partition function in this new space of configurations takes the following form

$$\mathcal{Z} = \int \prod_{i \in V} d\mu(x_i) \prod_{\langle i,j \rangle} \Theta(1 - x_i - x_j) \tag{2.6}$$

where the new one-site measure $\mu(x)$ corresponds to the old $\mu_e(x)$. With this notation, the trivial model we have seen in Section 2.1 have only *low* heights, i.e. less than $\frac{1}{2}$, so that any sum of pairs is always less than 1 and the total free energy is simply $V \ln \int d\mu(x)$.

The second answer requires a lot of work because we don't know *a priori* if this system, even if in the monotone case, will rapidly coalesce, but we can see how it behaves in the simple framework of forward simulations and we can analize the time of first coalescence. This study is shown in subsection 3.1.1 and make use of the symbolic math software Mathematica 6.0.

2.3 The one-parameter family of measures

Even if we can simulate our system with a CFTP algorithm we know there is no chance of measure directly a quantity like the free energy or the partition function because they are not local observables but depend on an overall factor not derivable from no one of the known random algorithms. The measure of a local observable related to the free energy can be found in analogy with a well-known relation in statistical mechanics and thermodynamics, that is when we derive with respect to the temperature the logarithm of the partition function we obtain a quantity related to the average energy of the system. Equation (2.7) shows that along a one-parameter

family of measures (i.e. the Gibbs measure with parameter β), the derivative of the partition function gives a local observable.

$$\langle E \rangle \;=\; -\frac{\partial \ln \mathcal{Z}(\beta)}{\partial \beta} \tag{2.7}$$

We can now introduce such a one-parameter family of measures with the hope that there exists an easy observable related to the derivative of the free energy along the measure parameter.

We know from equation (2.6) (equivalently equation (2.1)) that our system is totally entropic because the partition function has a combinatorial form. That is, the Hamiltonian of the system just consists of the constraint given by the Θ functions, and there is no quantity or parameter such as the inverse of the temperature β, on which the free energy depends. The only part we can vary in the partition function to obtain an expression like (2.7) is the one-site measure and we introduce a real parameter τ with these points in mind:

- we want to recover our corrugated surfaces model at one value of the parameter and we want to pass through a trivial model along the one parameter curve, that is the model where all the Θ's are satisfied and the partition function turns out to be the normalization of the one-site measure. Hence we ask

  $$\mu^{\tau=start}(x) \;=\; 0 \qquad \text{for all } x \in \left[\frac{1}{2}, 1\right] \tag{2.8}$$

  and

  $$\mu^{\tau=end}(x) \;=\; 1 \qquad \text{for all } x \in [0,1] \tag{2.9}$$

  and we will study an intervall of the form [start,end];

- we also want to guarantee the continuity of the $\mathbb{L}^2$ norm of the measure in the parameter and some sort of continuity given by the fact that $\mu^{\tau+d\tau}(x) \leq (1 + r(\tau)d\tau)\mu^{\tau}(x)$ for all $x \in [0,1]$, with an infinitesimal $d\tau$ and a certain finite function $r(\tau)$.

The behaviour of the measure between the above important values of τ, start and end, is choosen in a convenient way for the numerical simulations.
All these choices are made to allow an exactly sampled configuration $\mathbf{X}$ from the measure μ^{τ} to be sampled in a biased way from the measure $\mu^{\tau+d\tau}$, but with a known bias which we can show it is something like the derivative of the free energy with respect to τ. This topic is the main core of the theoretical work and we are going to show it in details.

There are a lot of one-parameter families of measures $\mu^{\tau}(x)$ meeting all the conditions above and we will use two of them that we have found particular suitable, one for the numerical implementation of the CFTP algorithm, and one for the theorical demonstration of the main topic (they are actually the same up to a trivial rescaling). These two families of measures have different typical features and since we want to compare simulation's results and theorical hypotheses,

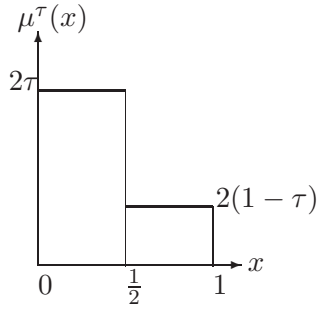


Figure 2.2 Simple plot of the family of measures parametrized by τ. In the simulations, each real number in $[0, \frac{1}{2})$ is picked with probability 2τ, whereas numbers in $[\frac{1}{2}, 1]$ with probability $2(1-\tau)$. The factor 2 ensures normalization.

at the end we must take into account the free energy shift coming from the rescaling.
The first parametrization we have choosen is the one that simply gives different weights to numbers greater than $\frac{1}{2}$ and less than $\frac{1}{2}$ and that is always normalized when varying the parameter τ. We can represent this family of measure as

$$\mu^\tau(x) := \begin{cases} 2\tau & \text{if} \quad x \in \left[0, \dfrac{1}{2}\right) \\ 2(1-\tau) & \text{if} \quad x \in \left[\dfrac{1}{2}, 1\right] \end{cases} \tag{2.10}$$

and in Figure 2.2 you can see a visual representation of what happens when, in the code, we sample a height: if $\tau = 1$, no *high* height are choosen and the trivial model occurs (i.e. the partition function is the normalization, that is 1), but changing τ towards the value of corrugated surfaces model, more points on the grid have a non-zero probability to be high (increasing in high points corresponds to a symmetric decreasing of low points and it becomes more and more difficult to satisfy the Θ functions).

We will show in Chapter 4 how we implement the sampling of each site's height from this measure. Furthermore we will show that, to get our final numeric result, we have ran several simulations at different values of the parameter τ.
The other parametrization is, in a way, more natural, but it is not normalized and this is why we use it in the calculations, where one can easily deal with this, rather than in the code where it is understood that any sampling procedure is normalized. In this case, values smaller than $\frac{1}{2}$ have always the same weight when changing the parameter, whereas values larger than $\frac{1}{2}$ are more and more damped as the parameter approaches the value corresponding to the trivial model. In

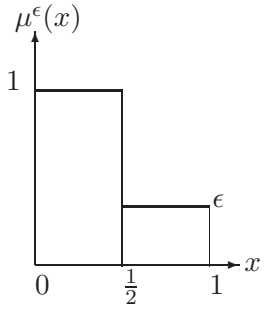


Figure 2.3 Plot of the family of measures parametrized by ϵ. This measures are not normalized except for the final model: $\epsilon = 1$.

this case we call the parameter ϵ (not to make confusion among the two families), and we write

$$\mu^\epsilon(x) := \begin{cases} 1 & \text{if} \quad x \in \left[0, \frac{1}{2}\right) \\ \epsilon & \text{if} \quad x \in \left[\frac{1}{2}, 1\right] \end{cases} \tag{2.11}$$

The normalization of $\mu^\epsilon(x)$ is 1 only for the final model (see Figure 2.3) so that the partition functions, using μ^ϵ and μ^τ, match in the relevant case with uniform measure, and the same do the free energies. As we will soon see the difference in the normalization, passing from μ^τ to μ^ϵ, that is from simulation's outputs to analytical calculations, will be easily handled.

2.3.1 Local observables and reweighting

We are now ready to introduce the underlying idea which will allow us for calculating the free energy. The first step is to understand the concept of **reweighting**, a statistical tool to measure observables when changing the probability distribution on the state space.
Let's make a clear example: in statistical mechanics with the Gibbs measure on the state space (i.e. $\exp(-\beta\mathcal{H})$) one may want to know the mean value of an observable $\mathcal{O}$ (i.e. $\langle\mathcal{O}\rangle_\beta$), depending explicitly on the parameter of the measure, β. With the reweight method it is possible to obtain the measure of the observable at another value of the parameter, say β', but without changing the probability distribution; starting from the Gibbs measure a temperature β one can calculate $\langle\mathcal{O}\rangle_{\beta'}$ and this is very useful doing simulations, because data sampled with β as parameter can be used to measure an observable at β' without the need of re-run the simulation. We give here the relation of what is called *simple histogram reweighting*

$$\langle\mathcal{O}\rangle_{\beta'} = \frac{\langle\mathcal{O}e^{-(\beta'-\beta)\mathcal{H}}\rangle_\beta}{\langle e^{-(\beta'-\beta)\mathcal{H}}\rangle_\beta} \tag{2.12}$$

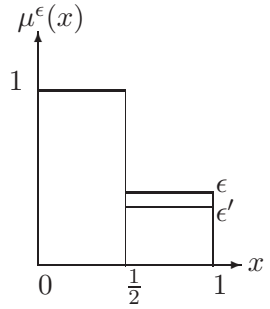


Figure 2.4 Visualization of the pointwise relation between different measures.

and we underline the fact that this can be viewed such as part of the measure being transformed in an observable quantity.

In our case we use the family of measures μ^ϵ and we try to vary the parameter by a small amount. What we note is

$$\mu^{\epsilon'}(x) \leq \mu^\epsilon(x) \quad \forall\, x \in [0,1] \qquad \text{if } \epsilon' \leq \epsilon \tag{2.13}$$

and also graphically (see Figure 2.4) it is clear that only *high* site are interested in the reweight.

Now call $\epsilon' = \epsilon - \delta$, for a small $\delta \geq 0$, and try to understand the meaning of changing the value of δ.

As we said before, μ^ϵ (and also μ^τ) is chosen fulfilling a sort of continuity, just the same we see in equation (2.13). So we claim that a configuration, sampled from $\mu^{\epsilon-\delta}$, is sampled from μ^ϵ in a biased way, and we can access the bias just looking at Figure 2.4! Rigorously speaking, since only numbers greater than $\frac{1}{2}$ are affected by the value of δ, if we measure how much of them there are in an exactly sampled configuration, we obtain a number related to the bias. The probability that a number x in $[\frac{1}{2}, 1]$ is choosen by the first measure μ^ϵ is just ϵ whereas the second measure $\mu^{\epsilon-\delta}$ would have choosen x with weight $\epsilon - \delta$: so the given height value appears in the two configurations, sampled from μ^ϵ and $\mu^{\epsilon-\delta}$, with relative probability

$$\begin{matrix} 1 & \text{if } x \in \left[0, \dfrac{1}{2}\right) \\ 1 - \dfrac{\delta}{\epsilon} & \text{if } x \in \left[\dfrac{1}{2}, 1\right] \end{matrix} \tag{2.14}$$

Since numbers in the second half interval of $[0,1]$ are so special in this case, and since they are the ones making effective the Θ functions, we want to call them **defects** (also cfr. Section 2.4). Graphically one can view the probability of equation (2.14) in Figure 2.5

If we sum over all feasible configurations we obtain a ratio for the partition functions that can be interpreted as the probability for a configuration $\mathbf{X}$ to be chosen in both the measures.

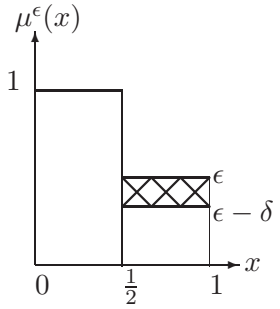


Figure 2.5 Visualization of the acceptance rate of the configurations sampled from different measures.

Since in a configuration there is a variable number of defects, the ratio becomes

$$\frac{\mathcal{Z}(\epsilon-\delta;L)}{\mathcal{Z}(\epsilon;L)} = \left\langle \left(1-\frac{\delta}{\epsilon}\right)^{\#\{\text{of defects}\}} \right\rangle_\epsilon \tag{2.15}$$

where the subscript ϵ of the expectation value means that we made the measure of the observable on a configuration sampled from μ^ϵ.
Now we want to manipulate this equation in order to obtain the expression of a free energy as a function of the measurable observable #{of defects}. We remark that the partition function depends on the parameter ϵ of the one-site probability distribution, but also on the grid's dimension L, because the realization of a particular configuration is strictly related to the volume $|V| = L^2$ of points.
Remembering equation (2.2) we write the ratio of partition functions as a single exponential function

$$\exp\left\{|V|\left[f(\epsilon-\delta)-f(\epsilon)\right]\right\} = \left\langle \left(1-\frac{\delta}{\epsilon}\right)^{\#\{\text{of defects}\}} \right\rangle_\epsilon \tag{2.16}$$

and, by multiplying and dividing the first exponent by δ, we obtain

$$\exp\left\{|V|\delta\left[\frac{f(\epsilon-\delta)-f(\epsilon)}{\delta}\right]\right\} = \left\langle \left(1-\frac{\delta}{\epsilon}\right)^{\#\{\text{of defects}\}} \right\rangle_\epsilon \tag{2.17}$$

We have said that the perturbation δ of the measure must be a small non negative quantity and in particular we now want to calculate the limit for $\delta \to 0$ because we are searching for a linear term, a derivative, of the free energy. We recognize in the exponent

$$\frac{f(\epsilon-\delta)-f(\epsilon)}{\delta} \tag{2.18}$$

an incremental ratio, that is the first order term in the Taylor series of the derivative. At the same time, if we do not consider terms of second or higher orders in δ, but only linear ones, the

quantity

$$\left(1-\frac{\delta}{\epsilon}\right)^{\#\{\text{of defects}\}} \tag{2.19}$$

is the Taylor expansion of an exponential function.
This allow us to write

$$\exp\left\{|V|\delta\left[-\frac{\partial f(\epsilon)}{\partial\epsilon}\right]\right\} = \left\langle \exp\left\{-\frac{\delta}{\epsilon}\#\{\text{of defects}\}\right\}\right\rangle_{\epsilon} \tag{2.20}$$

and equating the exponents, the equation becomes

$$|V|\delta\left[-\frac{\partial f(\epsilon)}{\partial\epsilon}\right] = \left\langle -\frac{\delta}{\epsilon}\#\{\text{of defects}\}\right\rangle_{\epsilon} \tag{2.21}$$

The $\#\{\text{of defects}\}$ is obviously related to the size L of the grid, so we change it with a density of defects that is normalized by the volume $|V|$:

$$\rho_{\text{def}}(L) = \frac{\#\{\text{of defects}\}}{L^2} \tag{2.22}$$

Simplifying $|V|$ and δ, equation (2.21) takes the form we want:

$$\frac{\partial f(\epsilon)}{\partial\epsilon} = \frac{\langle\rho_{def}\rangle_{\epsilon}}{\epsilon} \tag{2.23}$$

Now it's time to show out the final relation between the free energy of corrugated surfaces model (i.e. $f(\epsilon=1)$) and the local observable measuring the density of defects. Integrating equation (2.23) brings to

$$f(\epsilon=1)-f(\epsilon=0) = \int_0^1 d\epsilon\,\frac{\partial f(\epsilon)}{\partial\epsilon} = \int_0^1 d\epsilon\,\frac{\langle\rho_{def}\rangle_{\epsilon}}{\epsilon} \tag{2.24}$$

Hence, if we had an analityc expression for $\langle\rho_{def}(\infty)\rangle$ as a function of ϵ, we could integrate it and, knowing the free energy of the trivial model $f(\epsilon=0)$, calculate the free energy of our model.
This can be done numerically paying attention at the integration limit $\epsilon=0$, because it could be a singularity of the integrand function.

The free energy for the trivial model is known because we already said that the free energy is equal to the logarithm of the normalization of the measure times $|V|$ (see equation (2.6)) and, for the family μ^{ϵ} that we are using, it is equal to

$$f(\epsilon=0) = \frac{\ln(\mathcal{Z}(0;L))}{|V|} = \frac{\ln\left(\frac{1}{2}\right)^{|V|}}{|V|} = -\ln 2 \tag{2.25}$$

as we can see in Figure 2.3. Consequently

$$f(\epsilon=1) = \int_0^1 d\epsilon\,\frac{\langle\rho_{def}\rangle_{\epsilon}}{\epsilon} - \ln 2 \tag{2.26}$$

What we have to do in order to calculate $f(\epsilon = 1)$ is giving a specific form to $\langle \rho_{def} \rangle_\epsilon$, and for this we use the CFTP algorithm sampling each configuration with the one-site measure parametrized by τ (we already said this is the right choice because the probability distributions remain normalized at 1 for every value of the parameter τ).
We proceeded as follow: for each size L of the grid in the form $L = 2^k$, with $k \in \{4, 5, 6, 7, 8\}$, we sampled $N = 10^4$ configurations using μ^τ, with $\tau \in \{0.500, 0.525, 0.550, 0.575, \ldots, 0.975\}$ (and $\mu^\tau \sim \mu^\epsilon$ for $\epsilon = \frac{1-\tau}{\tau}$). On each configuration we performed the measure of how many defects there are. Hence we obtained, for each simulation with fixed L and τ, a statistical ensemble $\{\rho^i_{def}\}_{i=1\ldots N}$ on which we applied the standard statistical analysis, as we will show in section 3.2, resulting in the values $\langle \rho_{def}(L) \rangle_\tau$ with their appropriate statistical errors. Then, for each value of τ, we extrapolate, with a fit procedure, the limit $L \to \infty$ of the density of defects. The integral (2.26) can be then evaluated numerically using a polynomial fit for $\langle \rho_{def}(\infty) \rangle_\tau$. We defer to Section 3.2 any further discussion on simulations.

2.4 Cluster expansion

When we began to study the model of corrugated surfaces, in particular the behavior of defects in a configuration at certain ϵ, we found a resemblance between the defects lattice and a typical hard-core lattice model. By this, we mean that there can't be two adjacent (nearest neighbours) defects (i.e. equation (2.6) states that two numbers greater than $\frac{1}{2}$ violate the Θ function), and the defects of a feasible configuration can be interpreted as a perfect lattice gas, where a defect is a particle. Two particles cannot overlap because of the hard-core repulsion given by the Hamiltonian which is, in our case, just the Θ function. Moreover, in this scenario, we can think at the parameter ϵ as the gas *temperature*, with the gas becoming less and less dense when $\epsilon \to 0$ (low "excitation" energy).

Follow these considerations, we decide to apply the classical cluster expansion (cfr. [6] for a general overview of the theory) to find a perturbative series, in ϵ, for the free energy of the system.
Let us introduce some useful notations: we have a graph G with a lattice structure, $G = (V, E)$, and each vertex i has a random variable $x_i \in [0, 1]$; associated to each configuration, we have a list of positions $I = \{i_1, \ldots, i_k\}$ that univocally determines a connected subgraph H_I on G and a vector n on its vertex set, with $n_i = 1$ on vertices $i \in I$. This list is consistent, and generates a valid pair (H_I, n), if

- for each $(i, j) \in E(H_I)$, at least one among i and j is a defect;
- each $i \in V(H_I)$ such that $\deg_{H_I}(i) \leq \deg_G(i)$ is not a defect.

In this case we call (H_I, n) a *cluster*.

The cluster expansion allow us to write the partition function as a sum of weights i.e. $W(I)$, associated to each cluster. Each $W(I)$ is to be calculated using the explicit form of the Hamiltonian and we will see the form it takes in the model of corrugated surfaces. To continue this way, we have to make the assumption that $W(I) \sim W'(I)\epsilon^{|I|}$ for $\epsilon \to 0$, where $|I|$ stands for the cardinality of I, that is, for the number of defects. Then, the cluster of order ϵ are the ones looking like a"star" graph around a vertex i in the graph, i.e. $H_{\{i\}}$. The clusters of order ϵ^2 have two vertices i and j with $n = 1$, at distance at most 2, and $V(H)$ contains them and all their neighbours, and so on. We depict some clusters for the model of corrugated surfaces in Figure 2.6.

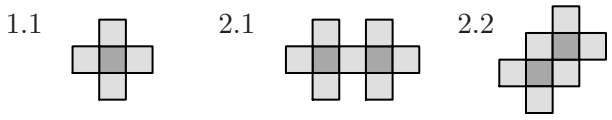


Figure 2.6 Valid clusters for the model of corrugated surfaces

Now consider the term of order ϵ in the expansion of Z. It is just the sum over all possible locations of a single defect, i.e.

$$\mathcal{Z} = 1 + \epsilon \sum_i W'(\{i\}) + \mathcal{O}(\epsilon^2) \tag{2.27}$$

Now, if we believe that the good strategy is understanding the logarithm of $\mathcal{Z}$, we will write that

$$Z = \exp\left[\epsilon \sum_i W'(\{i\}) + \mathcal{O}(\epsilon^2)\right] \tag{2.28}$$

Up to the corrections, this would be a *perfect gas* of isolated defects, at a density linear in ϵ. This is in accord with the naïve idea of assiomatic thermodynamic, such that any interacting gas at sufficiently low densities is well approximated by a perfect gas.

However there are a lot of wrong terms in the expansion of the quantity above, even at order ϵ^2. For example, we have terms of the form $\frac{1}{2}W'(\{i\})^2$, while we cannot have more than one defect at each site, and terms like $W'(\{i\})W'(\{j\})$ for i and j nearby, instead of the appropriate $W'(\{i, j\})$.

The point is that the wrong terms are of the same order in ϵ of the clusters we still have to consider. So, working at order ϵ^2, we would had anyhow to describe a perfect gas of isolated and paired defects, but, in order to consistently correct the error terms, we have to add other

"pseudo-clusters" to our collections, which can cancel the spurious contributions. Contrarily to the original clusters, where the factors $W'(I)$ come from an integration, and are positive at sight for a real-valued Hamiltonian, the correction terms can be real negative. The calculation of each order is a combinatorial work and the number of terms grows exponentially with the order of the parameter.

The model of corrugated surfaces is a special case where the factors $W(I)$ corresponding to sets I with adjacent defects would just be zero, and we have only to deal with sets I in which defects are a connected set of second-neighbour defects. We have that variables $x_i \in [0,1]$ can be replaced by a pair (y_i, n_i), with $y_i \in [0,1]$ and $n_i \in \{0,1\}$, and $x_i = (y_i + n_i)/2$. As we said at the beginning of this section, if both n_i and n_j are zero, the Θ function is always satisfied, and we are thus in the condition of using $n = 0$ as the 'zero-order' of a cluster expansion. Furthermore, writing the Θ function with the new variables as $\Theta(1 - (n_i + n_j + y_i + y_j)/2)$, makes clear that, if both n_i and n_j are 1, for any values of y_i and y_j the argument is negative. Thus, in this specialization of the cluster expansion, many terms $W(I)$ vanish (the ones for which 0's and 1's are not alternating over H). This implies that the exponential growth in the number of diagrams has a smaller rate for this model.

We have translational invariance in the system, up to boundary corrections (which are only of order $\sqrt{N}$, thus subleading at every order in our cluster expansion, in ϵ, of the free energy at order N).

At order 1 we only have to calculate $W(\{i\})$, and the free energy would read

$$f \;=\; \ln\big(1 + \epsilon W(\{i\})\big) + \mathcal{O}(\epsilon^2) \tag{2.29}$$

At order 2 we do not have to calculate $W(I)$ for sets I of adjacent defects, because of the remark above, but only for I with two defects at distance 2. There are two such diagrams (up to symmetries of the lattice), the one with relative defect positions at a vector $(1,1)$, and the one at position $(2,0)$. Each of them come with a symmetry factor 2 (the diagonal can be oriented at $\pi/4$ or $-\pi/4$, while the distance-2 dimer can be horizontal or vertical). Then we have the pseudo-clusters corresponding to these same configurations, and to the one of two adjacent defects. So we have

$$\begin{aligned} f = {} & \ln\big(1 + \epsilon W(\{i\})\big) \\ & + 2\ln\big(1 + \epsilon^2 W(\{i, i+(1,1)\})\big) + 2\ln\big(1 + \epsilon^2 W(\{i, i+(2,0)\})\big) \\ & - 6\ln\big(1 + \epsilon^2 W(\{i\})^2\big) + \mathcal{O}(\epsilon^3) \end{aligned} \tag{2.30}$$

Making the integrals of the corrisponding product of theta functions gives

$$W(\{i\}) = \int_0^1 dx_0 (1-x_0)^4 = \frac{1}{5} \tag{2.31}$$

$$W(\{i, i+(1,1)\}) = \int_0^1 dx_0 \left(\int_0^{1-x_0} dx_1 (1-x_1)^3 \right)^2 = \frac{1}{20} = \left(\frac{1}{5}\right)^2 + \frac{1}{100} \tag{2.32}$$

$$W(\{i, i+(2,0)\}) = \int_0^1 dx_0 2x_0 \left(\int_0^{1-x_0} dx_1 (1-x_1)^2 \right)^2 = \frac{2}{45} = \left(\frac{1}{5}\right)^2 + \frac{1}{225} \tag{2.33}$$

Substituting into (2.30) and expanding in series gives

$$\begin{aligned} f &= \epsilon \cdot \frac{1}{5} + \epsilon^2 \left[-\frac{1}{2} \left(\frac{1}{5}\right)^2 - 2 \left(\frac{1}{5}\right)^2 + 2 \left(\frac{1}{100} + \frac{1}{225} \right) \right] + \mathcal{O}(\epsilon^3) \\ &= \epsilon \cdot \frac{1}{5} - \epsilon^2 \cdot \frac{16}{225} + \mathcal{O}(\epsilon^3) \end{aligned} \tag{2.34}$$

This is an important result because it gives us the possibility to check out if our simulation data behave properly. Indeed, we will see in Chapter 3 that our data are compatible with a distribution $\langle \rho_{\text{def}} \rangle_\epsilon$ well-approximated by a polynomial in ϵ, at least for some neighbourhood of $\epsilon = 0$. The cluster expansion we developed in this section provides us a prediction for the first three terms of this polynomial (included the fact that in $\epsilon = 0$ the density is zero), namely

$$\begin{aligned} \langle \rho_{\text{def}} \rangle_\epsilon \;&=\; \epsilon \, \frac{\partial f(\epsilon)}{\partial \epsilon} \;=\; \epsilon \left[\frac{1}{5} - \frac{32}{225} \epsilon + \mathcal{O}(\epsilon^2) \right] \\ &=\; 0.2\,\epsilon - 0.142222\,\epsilon^2 + \mathcal{O}(\epsilon^3) \end{aligned} \tag{2.35}$$

We will see in Chapter 3 that the numerical values of our fit are remarkably compatible with these values.

3. Data and simulations analysis

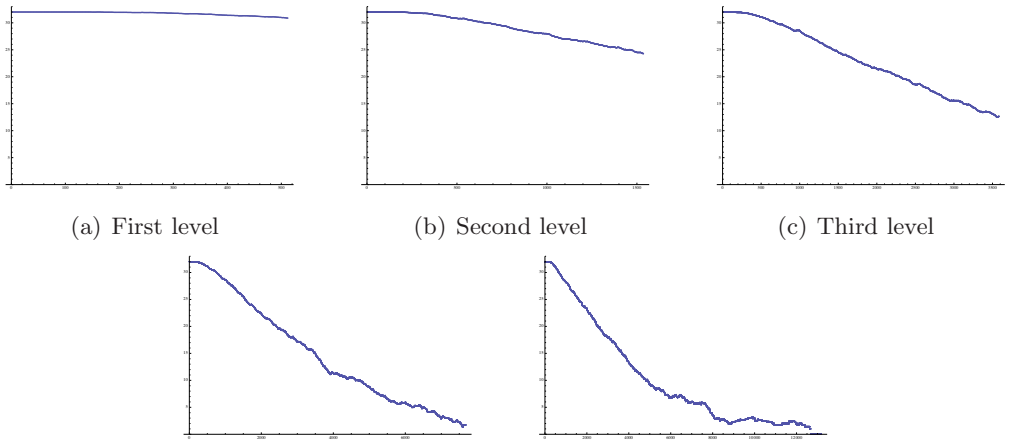
3.1 Preliminary checks with Mathematica

3.1.1 Coalescence time

The simulation algorithm described, both in the general framework of MCMC (Section 1.2) and for the specific model of corrugated surfaces (Section 2.2), requires some preliminary work to find out if our choice for the update function of the Markov chain could bring, in a resonably short time (at least in polynomial time), to coalescence. It is a subtle matter because we know there is no *a priori* knowledge of the result.

Then we decide to perform some test simulations using our update function, and to plot, run-time, a specific variable related to the differences between the two coupled Markov chains, one starting from the $\hat{1}$ configuration and one strating from $\hat{0}$. This variable, called *distance*, get the number of sites that are different between the two ran chains, and it is obviously 0 when the chains have coalesced. For almost every lattice size L and parameter value τ, we run a CFTP algorithm implemented in Mathematica code (cfr. Chapter 4). When the algorithm starts iterating the chains it may fail to reach coalescence, and at that time it doubles the running time and restarts the chains: we call this a level of the CFTP run. We plot, at every level, the distance between the chains, and we study how the distance behave during the simulation.

We can see, from a couple of figures (Fig. 3.1) got from Mathematica, that initially the plot presents a plateau and the distance doesn't change too much. Then the distance becomes more and more smaller, and the behavior seems monotone. But at the very end it happens that another effects interacts with the decreasing of the distance. In the plots we can see a fluctuations of the distance until it reaches the 0, and it seems that the coalescence process has to pass through a tunnelling effect before it successes. This trend appears in all our simulations, so we think it is characteristic of the coalescence process, but we don't investigate this aspect any more.
However, the coalescence time doesn't seems to grow exponentially with the lattice size L.



(a) First level (b) Second level (c) Third level

(d) Fourth level (e) Fifth level

Figure 3.1 Trend of the distance between two Markov chains starting from the extremal of the lattice whit $L = 32$ and $\tau = 0.500$

3.1.2 3×3 *distribution*

As we have shown in Chapter 1, our algorithm should output samples with the desired probability distribution. Before running it intensively, we want some strong test that this is the case. Hence we find a simple case where the partition function is easily calculated and we can compare simulated data and analityc results: the simplest but not trivial situation fitting our requirements is the 3×3 squared lattice with free boundary condition, where we want to recover the appropriate marginal distribution of the "middle" variable z.

y_4	x_1	y_1
x_4	z	x_2
y_3	x_3	y_2

Figure 3.2 The square lattice used in the calculation of the distribution probability for the central site's height z.

Firstly we run a simulation for our model ($\tau = 0.5$, i.e. uniform measure in $[0, 1]$) in this small square and output the central site's height z for 10^6 (conjecturally) exactly sampled configurations.
Then we calculate the distribution

$$p(z) = \int_0^1 dx_1 \cdots dx_4 \int_0^1 dy_1 \cdots dy_4 \prod_{i=1}^{4} [\Theta(1 - z - x_i)\Theta(1 - x_i - y_i)\Theta(1 - x_i - y_{i-1})] \quad (3.1)$$

in which we use labels according to Figure 3.2.

Examining the expression (3.1) in details, especially the Θ functions, we note that there are symmetries according to the values x_i can take: if we choose $x_1 = \max_i(x_i)$ (and multiply by 4 the resulting expression because the maximum could have been in any of the 4 locations), the first Θ gives $x_1 \leq (1-z)$ and we can write

$$p(z) \;=\; 4 \cdot \int_0^{1-z} dx_1 \int_0^{x_1} dx_2 \cdots dx_4 \int_0^1 dy_1 \cdots dy_4 \prod_{i=1}^{4} [\Theta(1-x_i-y_i)\Theta(1-x_i-y_{i-1})]$$

Now we must choose which x_i ($i = 2,3,4$) is the second maximum, and we clearly see three different cases of which two are symmetrical:

A if we take x_2, then x_4 is symmetrical by the swappings $y_1 \leftrightarrow y_4$, $y_2 \leftrightarrow y_3$, so it suffices to multiply by 2 the integral; moreover the integrals of y_i can be calculated because

$$\begin{aligned}
\Theta(1-x_1-y_1)\Theta(1-x_2-y_1) \;&=\Theta(1-x_1-y_1) \quad (\text{because } x_1 \geq x_2)\\
\Theta(1-x_1-y_4)\Theta(1-x_4-y_4) \;&=\Theta(1-x_1-y_4) \quad (\text{because } x_1 \geq x_4)\\
\Theta(1-x_2-y_2)\Theta(1-x_3-y_2) \;&=\Theta(1-x_2-y_2) \quad (\text{because } x_2 \geq x_3)
\end{aligned}$$

and this leads to

$$\int_0^{1-x_1} dy_1 \int_0^{1-x_2} dy_2 \int_0^{1-x_1} dy_4 \;=\; (1-x_2)(1-x_1)^2$$

B if we take x_3, then the integration of x_2 and x_4 runs in $[0, x_3]$ and the same argument on the Θ's applies here

$$\begin{aligned}
\Theta(1-x_3-y_3)\Theta(1-x_4-y_3) \;&=\Theta(1-x_3-y_3) \quad (\text{because } x_3 \geq x_4)\\
\Theta(1-x_3-y_2)\Theta(1-x_2-y_2) \;&=\Theta(1-x_3-y_2) \quad (\text{because } x_3 \geq x_2)\\
\Theta(1-x_1-y_1)\Theta(1-x_2-y_1) \;&=\Theta(1-x_1-y_1) \quad (\text{because } x_1 \geq x_2)\\
\Theta(1-x_1-y_4)\Theta(1-x_4-y_4) \;&=\Theta(1-x_1-y_4) \quad (\text{because } x_1 \geq x_4)
\end{aligned}$$

and leads to

$$\int_0^{1-x_1} dy_1 \int_0^{1-x_1} dy_4 \int_0^{1-x_3} dy_2 \int_0^{1-x_3} dy_3 \;=\; (1-x_1)^2(1-x_3)^2$$

At this point the expression 3.1 simplifies as follows

$$\begin{aligned}
p(z) = 2 \cdot 4 \cdot \int_0^{1-z} dx_1 \int_0^{x_1} dx_2 \int_0^{x_2} dx_3 dx_4 (1-x_2)(1-x_1)^2 \int_0^1 dy_3\, \Theta(1-x_3-y_3)\Theta(1-x_4-y_3)\\
+\, 4 \cdot \int_0^{1-z} dx_1 \int_0^{x_1} dx_3 \int_0^{x_3} dx_2 dx_4 (1-x_3)^2(1-x_1)^2
\end{aligned}$$

The last two Θ's are satisfied if we choose the third maximum among x_3 and x_4. We take $x_3 \geq x_4$ and multiply by 2 because our choice is arbitrary.
So we have

$$p(z) = 2 \cdot 2 \cdot 4 \cdot \int_0^{1-z} dx_1 \int_0^{x_1} dx_2 \int_0^{x_2} dx_3\, x_3(1-x_3)(1-x_2)(1-x_1)^2 + 4 \cdot \int_0^{1-z} dx_1 \int_0^{x_1} dx_3 x_3^2(1-x_3)^2(1-x_1)^2$$

These integrals can be easily calculated either by hand or with Mathematica and we obtain the probability distribution

$$p(z) = \frac{17}{630} - \frac{8}{45}z^3 + \frac{4}{15}z^5 + \frac{2}{9}z^6 - \frac{4}{7}z^7 + \frac{7}{30}z^8 \tag{3.2}$$

which is not normalized. The normalized one is

$$p^{norm}(z) = \frac{3}{25}(z-1)^4(17 + 68z + 170z^2 + 228z^3 + 147z^4) \tag{3.3}$$

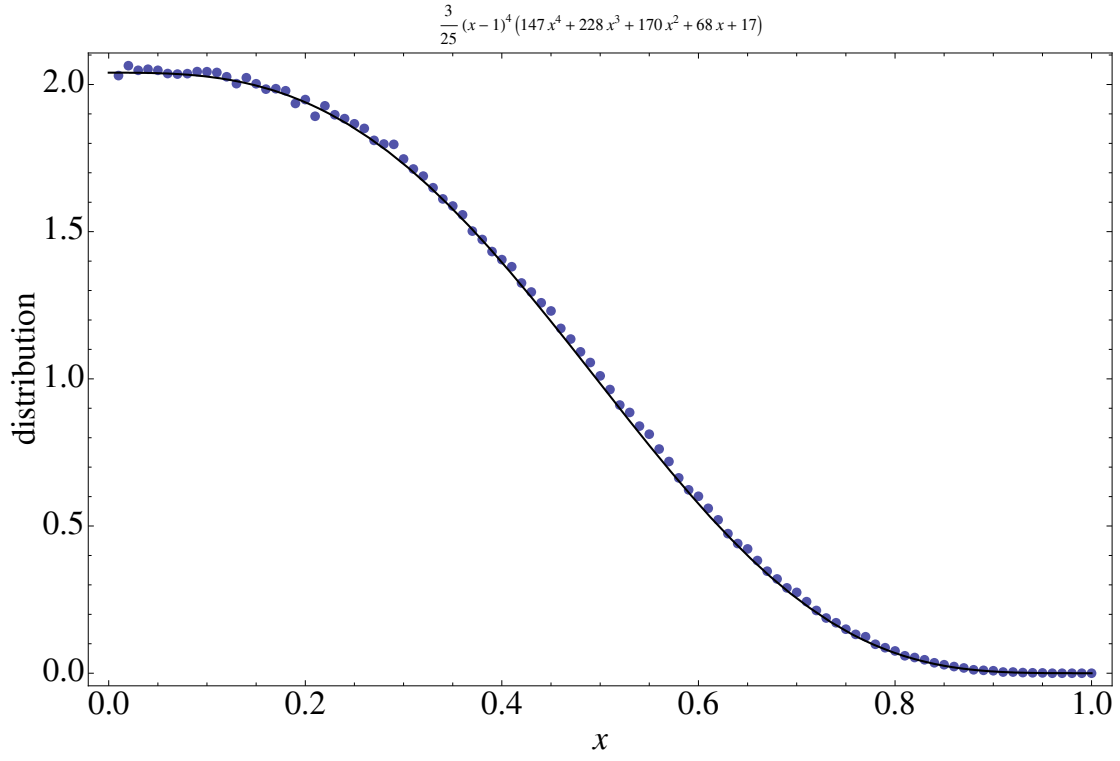


Figure 3.3 Comparison between an exactly sample of 10^6 measure of the central site's height in a 3×3 lattice and the analytical form of its distribution written in equation (3.3).

Finally we arrange our simulation's data in a normalized histogram so that we can compare their distribution form with our formula (3.3). The superimposed plots are shown in Figure 3.3 and look convincing.

In statistical data analysis there are several tests for the goodness of a set of data. When one have a population with a specific statistical distribution and want to decide if a particular sample comes from that population, it must use a parameter-free test because the distribution is exactly know without parameters. That's our case and we use the Kolmogorov-Smirnov test to decide if the sample output by the simualtion differs from the hypothesized distribution $p^{norm}(z)$ in equation (3.3). Since we have seen, at least graphically, that all works fine, we presume that the test will not reject our hypothesis.
Briefly, the test compares the empirical cumulative distribution function to the cumulative function and it use the maximum difference between them as a statistic, called Kolmogorov-Smirnov statistic. For a sample of n data, $\{x_i\}_{i=1\ldots n}$, we have the empirical distribution function

$$F_n(x) \;=\; \frac{1}{n}\sum_{i=1}^{n} I_{x_i \leq x} \tag{3.4}$$

where $I_{x_i \leq x}$ is the characteristic function, and the statistic is

$$D_n \;=\; \sup_x |F_n(x) - F(x)| \tag{3.5}$$

where $F(x)$ is the hypothesized cumulative distribution, i.e. $F(x) = \displaystyle\int_{-\infty}^{x} p^{norm}(z)\, dz$.
The variable $\sqrt{n}D_n$ follow the Kolmogorov distribution and the test is constructed by using the critical values of this distribution. The hypothesis regarding the distributional form is rejected if the test statistic, $\sqrt{n}D_n$, is greater than the critical value obtained from statistical tables.
The test performed on our data bring to the following result

$$\sqrt{n}D_n \;=\; 1.022 \tag{3.6}$$

with $n = 10^6$, and this is lower than the critical value $\alpha = 1.073$ corresponding to a probability greater than 20% to have a good fit.

3.2 Analysis of simulation's output

As we said at the end of Section 2.3.1, we have ran simulations both varying the parameter τ and the lattice size L, in order to obtain an amount of data sufficiently for a statistical analysis. Each simulation consists in $N = 10^4$ runs of the monotone CFTP algorithm, each run exactly sampling a feasible configuration and outputting the measure of the number of defects (i.e.

heights greater than $\frac{1}{2}$) into a file, together with the number of sweeps of the Markov chain needed to reach coalescence.
Since the output comes from an exact sampling procedure, we know that our data are not correlated and that it is possible to straightforwardly use the gaussian error analysis. All this hypothses will be probed in the following sections.

3.2.1 Run-time

First of all, we want to know how fast our implemented algorithm is, and how long we have to wait until coalescence is reached. This survey is needed for us to know if all in the code is working in the right way: if we think at the family of measure μ^τ used in the simulation, we presume that, at distinct values of the parameter τ, the run-times shoul be considerably different due to the quantity of inequalities the code must proceed. Thus, if there is a small probability for a number greater than $\frac{1}{2}$ to be sampled (i.e. the parameter τ is near 1), a feasible configuration of heigths is easily accomplished, remembering that all Θ's functions in equation (2.6) are satisfied and we are near the trivial model along the parameter curve.

We now show in Table 3.1 a summary of how long, in seconds, does it take for 10 runs to complete, that is to have 10 exactly sampled configurations, with a different set of initial conditions (different lattice size and one-site measure). Clearly, this time is processor-dependent, but the qualitative argument is properly shown.

		time $[sec]$		
	τ	0.500	0.750	0.900
L	16	0.08	0.03	0.02
	32	0.41	0.21	0.09
	64	2.02	0.79	0.58
	128	10.4	3.11	2.74
	256	45.8	16.2	13.2

Table 3.1 Time for outputting 10 exactly sampled configurations with different values of L and τ

3.2.2 Data distribution

Now we focus our attention on the output of each simulation, which is a set of 10^4 measures of the configuration's density of defects.

If it was the case of a typical Monte Carlo Markov Chain simulation, we would expect samples, that is measure, correlated in some way. We should then remove correlations and

bias, due to the approximate sampling, if we would accomplish the right result of the mesure. But, since we are not dealing with an approximate sampling method, we presume our data to be statistically uncorrelated and obviously unbiased (an exact sampling method always output configuration with the right equilibrium distribution). There is neither thermalization time nor correlation one, so each measure we find in the output file has to be consider a normal random variable, whose distribution has got a mean value and a standard deviation.

We present, in Figure 3.4 and 3.5, the histogram distributions of the number of defects for the $L = 256$ lattice with different values of the measure parameter τ. It is clear that the distribution is a gaussian one even from the plot, but the reader can easily verify the matching with the superimposed plot of a fitted gaussian. In both figure's legend there is the χ^2 value of the gaussian fit over the histogram and both look very good. However we decide not to show the mean and variance fitted value, because we prefer to run an analysis program we wrote and which use the all data information instead of the histogram data information. Now that we are sure all our data are distributed normally (the same fitting procedure as above was performed on all data sets), we can straightforward analyze them.

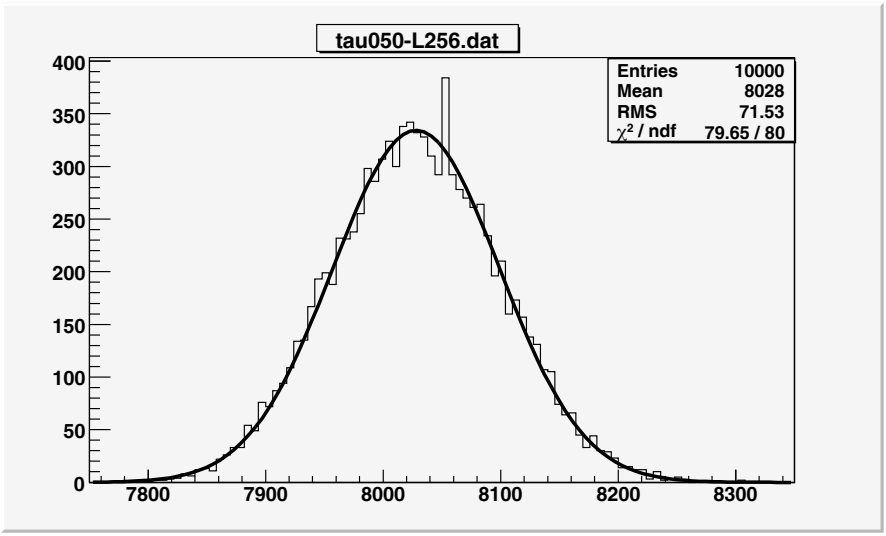


Figure 3.4 Distribution of defects in a 256×256 lattice with $\tau = 0.50$ fitted by a gaussian.

What we are interested in is clearly the average value of the number of defects, and the standard deviation of the distribution as the statistical error for the measure. In particular we

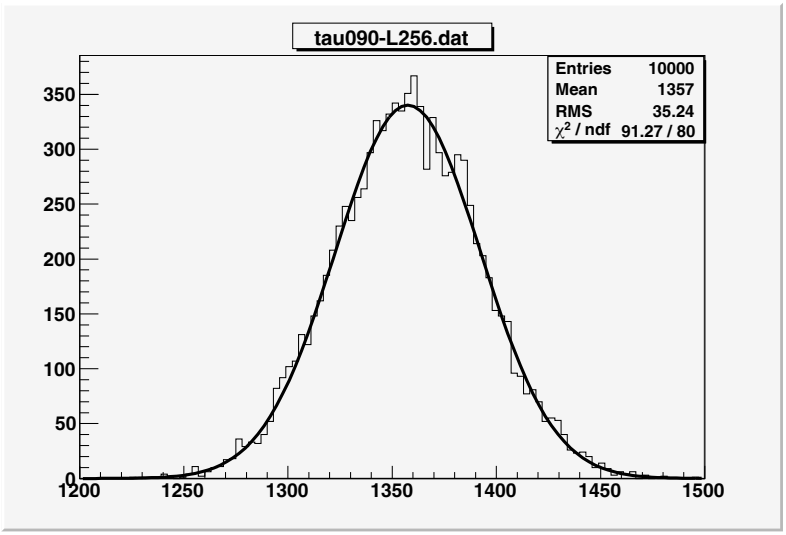


Figure 3.5 Distribution of defects in a 256×256 lattice with $\tau = 0.90$ fitted by a gaussian.

use the following expressions

$$\begin{aligned} \text{mean value:} \quad \langle n \rangle &= \frac{\sum_{i=1}^{N} n_i}{N} \\ \text{variance:} \quad \sigma^2_{\langle n \rangle} &= \frac{\langle n^2 \rangle - \langle n \rangle^2}{N} \end{aligned} \tag{3.7}$$

for the set of data $\{n_i\}_{i=1,\dots,N}$. We show the result of the statistical analysis on each set of measures in Table 3.2.

3.2.3 Finite-size scaling

Since we are going to use a density of defects, we perform a transformation over each file of data in order to obtain, for each parameter τ, a table with the measured density and its statistical error. This is important because we want to fit these averaged values and we need an error on them: we decide to set the error equal to the standard deviation of the distribution (from equation (3.7)) divided by the square root of the total number of events in the distribution (e.g. $\sqrt{10^4}$) and obviously by the volume of the lattice because it is a density.

We imagine that the behavior of the density as a function of the lattice size L, for a fixed value of τ, is not affected by any of the typical effects one could find in studying a phase transition with a Monte Carlo simulation. That is we think there are no collettive phenomena, and the correlation function of two adjacent variable is exponentially damped. All these hypotheses are supported by the interpretation, given in Section 2.4, of the defects as a particle-interacting gas. Since we are not passing through a critical point of a phase transition (as it would be if we

		Measures of the number of defects				
	τ	0.500	0.600	0.700	0.800	0.900
L	16	33.00±4.62	25.12±4.21	18.06±3.74	11.60±3.17	5.60±2.29
	32	128.34±8.98	97.552±8.41	70.125±7.36	45.007±6.21	21.721±4.47
	64	506.40±18.09	384.64±16.38	276.16±14.65	177.23±12.25	85.712±8.88
	128	2013.05±35.50	1528.53±32.54	1096.80±29.05	703.69±24.46	340.21±17.80
	256	8027.81±71.50	6094.54±66.20	4373.35±58.88	2806.17±49.31	1356.91±35.24

		Measures of the number of defects				
	τ	0.550	0.650	0.750	0.850	0.950
L	16	28.95±4.42	21.49±3.99	14.76±3.49	8.53±2.79	2.77±1.64
	32	112.48±8.65	83.438±7.92	57.316±6.84	33.175±5.44	10.71±3.20
	64	443.64±17.16	329.03±15.49	225.63±13.49	130.677±10.7791	42.24±6.45
	128	1763.11±34.17	1306.99±30.89	896.33±26.79	518.81±21.51	167.54±12.76
	256	7030.20±68.39	5212.14±62.83	3573.18±54.19	2068.50±42.98	667.86±25.40

		Measures of the number of defects				
	τ	0.525	0.625	0.725	0.825	0.925
L	16	30.94±4.52	23.27±4.09	16.40±3.62	10.06±2.99	4.18±1.98
	32	120.30±8.82	90.38±8.19	63.64±7.07	39.02±5.85	16.21±3.94
	64	474.46±17.60	356.48±15.93	250.61±14.05	153.65±11.52	63.83±7.83
	128	1886.02±34.91	1416.19±31.53	995.47±27.99	610.49±23.15	253.27±15.57
	256	7520.64±70.20	5647.56±64.62	3968.96±56.47	2433.84±46.31	1009.44±31.04

		Measures of the number of defects				
	τ	0.575	0.675	0.775	0.875	0.975
L	16	26.97±4.32	19.77±3.88	13.20±3.34	7.06±2.57	1.37±1.15
	32	104.85±8.48	76.71±7.63	51.12±6.56	27.41±5.00	5.31±2.28
	64	413.61±16.80	302.19±15.01	201.22±12.90	108.05±9.91	20.96±4.53
	128	1644.15±33.28	1200.62±29.89	798.92±25.66	428.72±19.69	83.15±9.06
	256	6555.08±67.85	4787.43±60.97	3185.86±51.91	1709.34±39.27	331.55±18.1

Table 3.2 Average and standard deviation summarizing all our simulation's data.

try to extend our model at values of the parameter τ less than one-half), an intensive physical quantity as the density of defects scales, with the size of the lattice, in a simple way. We have tested our predictions by fitting the data, exposed in the previous section, with the following function

$$\langle \rho_{def}(L) \rangle_\tau \; = \; \langle \rho_{def}(\infty) \rangle_\tau + \frac{A_\tau}{L} + \frac{B_\tau}{L^2} + \mathcal{O}(L^{-2}) \tag{3.8}$$

Corrections of order $1/L$ to the "true" physical observable $\rho_{def}(\infty)$, are due to the border effects on the lattice. Other greater orders are subleading and we chose them in order to make the best fit (χ^2 test), and they are of some utility only for small L. Indeed, we can plot, in a logarithmic scale, the density versus the lattice size, and we should see how points are linearly dependent from $\log_2 L$ (this procedure is show in Figure 3.8). The reader can see fits for the size scaling in Figure 3.6 and 3.7. Moreover, we show the values for the extrapolated density of defects ($\rho_{\rm def}(\infty)$) with their error bars (σ_ρ) in Table 3.3. As error, σ_ρ, we choose the one given us by the fitting procedure for the constant term in the expression (3.8).[1]

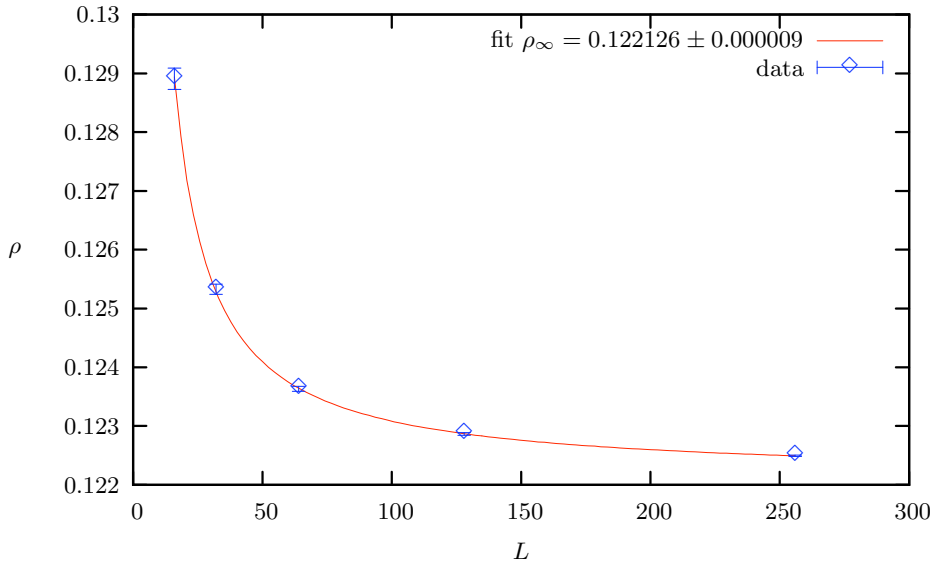


Figure 3.6 Scaling of the density of defects ρ with the lattice size L for $\tau = 0.500$: the fitting function gives a value $\rho_\infty = 0.122126 \pm 0.000009$ for the density in the approximation of infinite size.

3.3 Fitting the data

Giving that this is the right procedure to extrapolate the density of defects in the thermodynamic limit, we obtain a sequence of points, one for each parameter τ, representing the *experimental*

[1] We remark that we use the open source program Gnuplot for all the fits in this thesis.

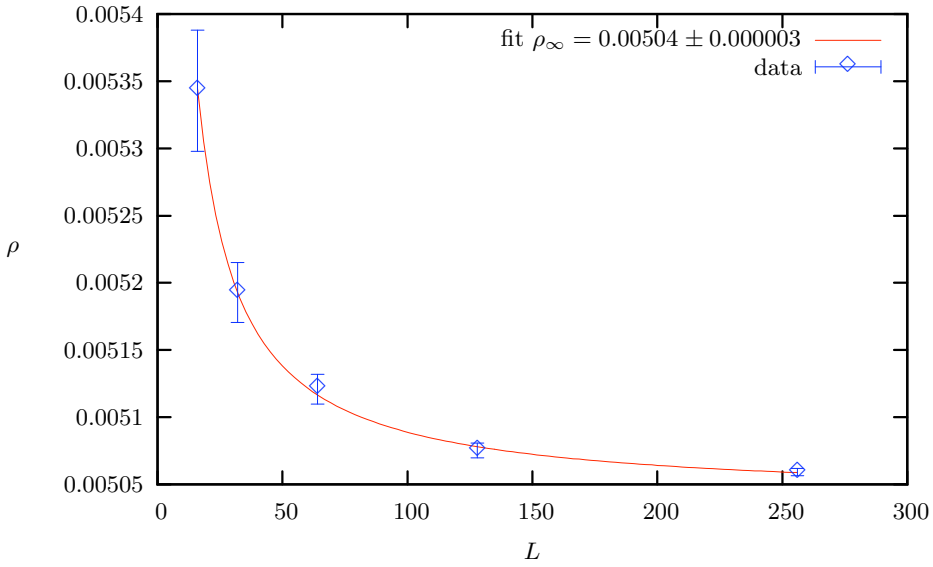


Figure 3.7 Scaling of the density of defects ρ with the lattice size L for $\tau = 0.975$: the fitting function gives a value $\rho_\infty = 0.00504 \pm 0.000003$ for the density in the approximation of infinite size.

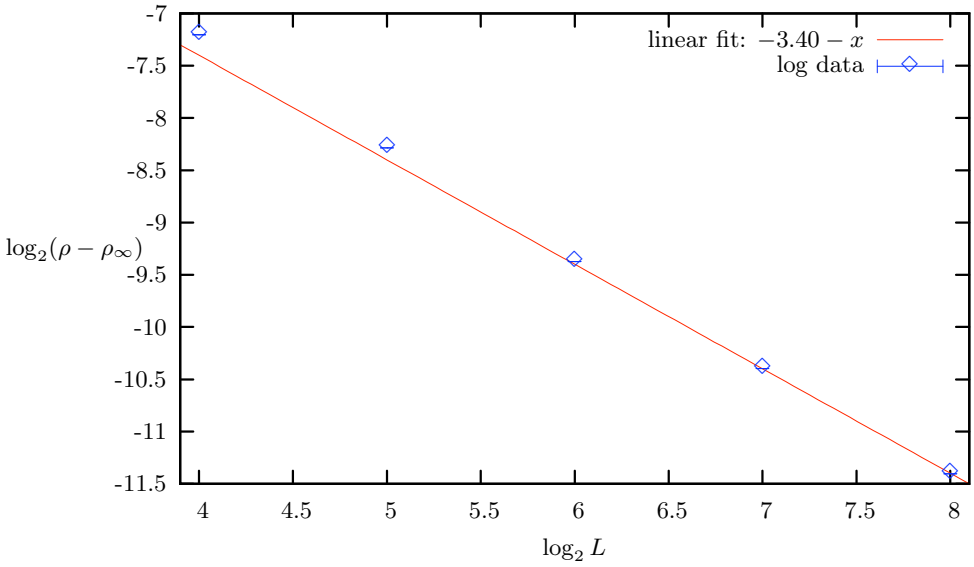


Figure 3.8 Logarithmic plot for the density of defects as a function of the lattice size ($L = 256$ and $\tau = 0.500$). If we make the logarithm of equation 3.8 we have the -1 slope of this plot as we would. We also note that subleading corrections are more important on small lattice and the corresponding points have a systematic deviation from the fitted line.

τ	$\rho_{\mathrm{def}}(\infty)$	σ_ρ
0.500	0.122126	$9.13{\cdot}10^{-06}$
0.525	0.114403	$1.21{\cdot}10^{-05}$
0.550	0.106936	$6.27{\cdot}10^{-06}$
0.575	0.099710	$9.28{\cdot}10^{-06}$
0.600	0.092699	$7.88{\cdot}10^{-06}$
0.625	0.085900	$1.17{\cdot}10^{-05}$
0.650	0.079275	$1.30{\cdot}10^{-05}$
0.675	0.072816	$1.24{\cdot}10^{-05}$
0.700	0.066505	$1.65{\cdot}10^{-05}$
0.725	0.060357	$1.18{\cdot}10^{-05}$
0.750	0.054334	$1.20{\cdot}10^{-05}$
0.775	0.048448	$1.41{\cdot}10^{-05}$
0.800	0.042673	$1.27{\cdot}10^{-05}$
0.825	0.037013	$7.71{\cdot}10^{-06}$
0.850	0.031449	$8.24{\cdot}10^{-06}$
0.875	0.025986	$7.95{\cdot}10^{-06}$
0.900	0.020636	$6.94{\cdot}10^{-06}$
0.925	0.015344	$2.43{\cdot}10^{-06}$
0.950	0.010153	$3.74{\cdot}10^{-06}$
0.975	0.005039	$2.58{\cdot}10^{-06}$

Table 3.3 Density of defects extrapolated at $L \to \infty$ for all the values of τ.

(simulated) trend of the free energy's derivative (unless of a factor). What we have shown in equation (2.23) is true for the parameter ϵ, but all we have to do, to take into account the different parametrization of the measure, is to assign at each τ the corresponding value of ϵ which brings to the same measure (exept for the normalization factor). The substitution is $\epsilon = \frac{1-\tau}{\tau}$.

If we look at Table 3.3, the only thing that changes is the first column, where we are going to put ϵ instead of τ, but the density doesn't change because the probability distribution is the same (we take into account the shift of the normalization only at the end of the procedure). From Section 2.4, we are searching for a polynomial in ϵ fitting our data, and then we would like to integrate this polynomial divided by ϵ. Firstly we have to pay attention at the constant term of the polynomial, because one of our integration bound is 0, in which $\frac{1}{\epsilon}$ diverges. For this reason we choose to set "by hand" the first parameter of the fitting function, and we clearly put it to 0 so that the possibly divergent term vanishes. The final result of our fitting analysis is shown in Figure 3.9.

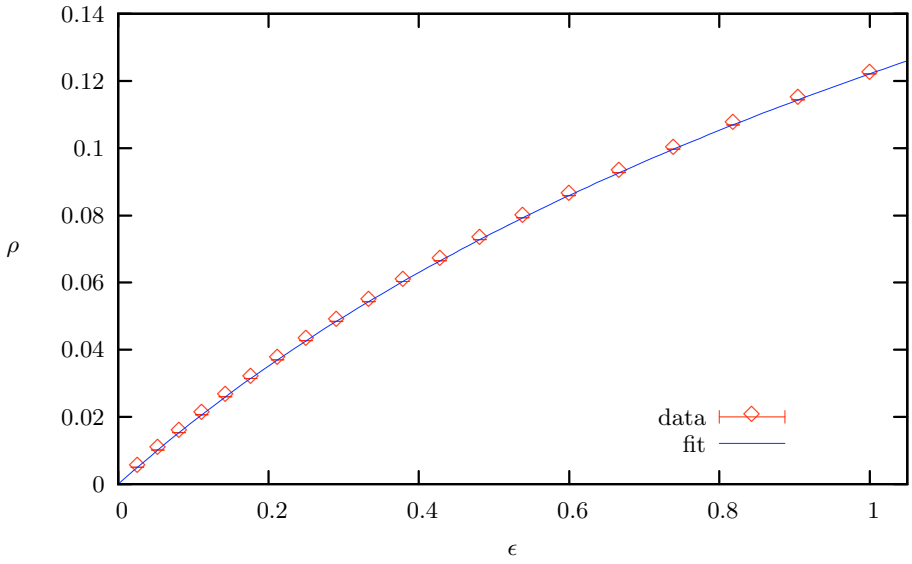


Figure 3.9 Data fitted with the polynomial $0. + 0.199913\ x - 0.139179\ x^2 + 0.102073\ x^3 - 0.054643\ x^4 + 0.01397\ x^5$ where the absence of a constant terms is forced.

Now, one may asks why we have chosen just a fifth order polynomial, and the answer is graphically shown in Figure 3.10 and 3.11. As the labels of the two figures explain, we have tried fitting polynomials of several different orders, with the purpose of find the best fit. Looking only at the value of the reduced χ^2 would have get the fifth order polynomial as the best fit, but a more intuitive and simper way to see if the fit was right is to plot the difference between

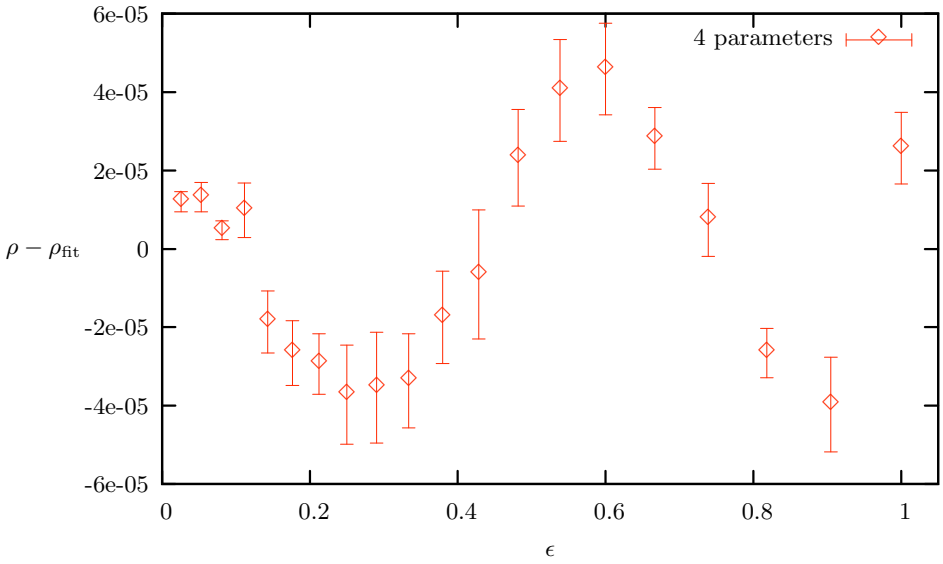


Figure 3.10 Plot of the difference between the density data and the fitting function with 4 parameters: a fourth order polynomial. We note a *structure* in the plot which show the bad quality of the fit: the resultant $\tilde{\chi}^2$ is greater than 10.

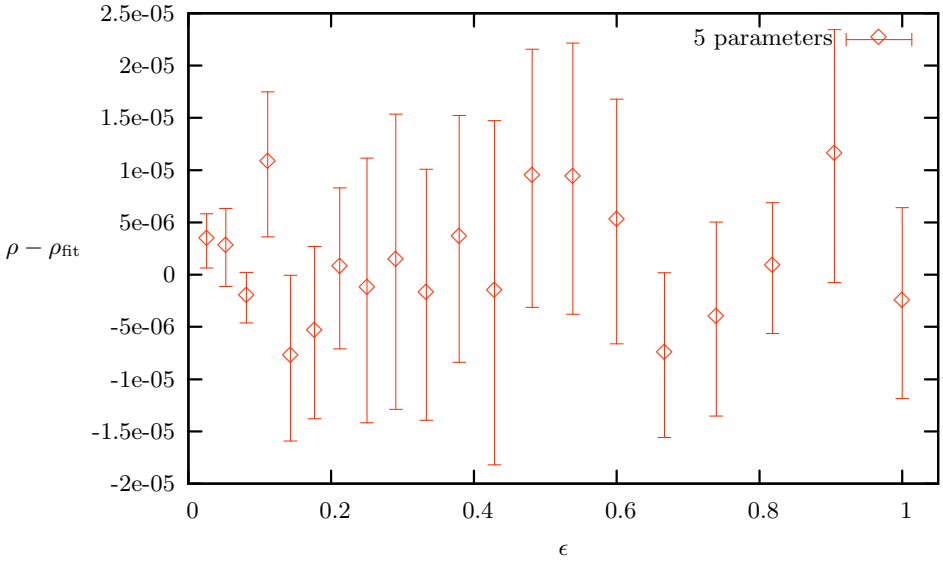


Figure 3.11 In this plot where the fitting function is a fifth order polynomial, the structure is less visible and however fluctuate inside the error bars. The resultant $\tilde{\chi}^2$ is 0.68 which is good.

the data and the fitted function evaluated in the same points. This method resembles the least squares method for fitting linearly arranged data. Indeed, when we look at this differences for polynomial of order less than 5, we find that they fluctuate with a clearly visible pattern, that is, there is an underlying *structure* which our fit doesn't take into account. In this way we find that the fit of Figure 3.9 is the best we can do.

This is not yet the end of our work, because each coefficient of the polynomial is a fitted parameter, and it has an error that we must consider in order to get the uncertainty on the final result for the free energy. It follows a table (3.4) with the fitted polynomial and the error bars assigned to each coefficient. We are glad to see that the coefficients a and b are of the same order of the one we have theoretically predicted in equation (2.35), using the cluster expansion method.

$a\,\epsilon + b\,\epsilon^2 + c\,\epsilon^3 + d\,\epsilon^4 + e\,\epsilon^5$	
a	$+0.199913 \pm 0.000044$
b	-0.139179 ± 0.000489
c	$+0.102073 \pm 0.001628$
d	-0.054643 ± 0.002078
e	$+0.01397 \pm 0.0009$

Table 3.4 Fitted parameters for the polynomial.

3.3.1 The free energy

Now we have to integrate, like we wrote in equation (2.26), the fitted polynomial, and we will get back with the free-enery of the system in the thermodynamic limit. Since all we have done is concretely a numerical procedure, we have to handle the statistical errors given by the fit, and we have to show the free energy together with its error bar.

Helped by Mathematica 6.0, we calculate

$$f(\epsilon = 1) + \ln 2 \;=\; \int_0^1 \frac{a\,\epsilon + b\,\epsilon^2 + c\,\epsilon^3 + d\,\epsilon^4 + e\,\epsilon^5}{\epsilon}\, d\epsilon \tag{3.9}$$

where the parameters of the numerator in the integral are shown in Table 3.4. We obtain

$$f(\epsilon = 1) \;=\; -0.539666 \tag{3.10}$$

The method we used to devise the error on the free energy, starting from the parameters error, is called OAT procedure, because it consists in varying, one at a time, each fitted parameter while other are fixed and supposed without uncertainty. Doing so we can calculate a lower and an

upper limit for the free energy. The lower limit is -0.540209 and the upper limit is -0.539123. They bring together to the final result for our system's free energy

$$f = -0.539666 \pm 0.000543 \tag{3.11}$$

4. The code

4.1 Mathematica code

We initially performed some simulations with Mathematica because of the ease of visualizing the configurations on a grid, and of implementing some kind of actions with simple code.
We created a grid, or matrix if one preferes, with the command `Table[]`. Adding an extra frame of quenched variables makes the free-boundary conditions:

```
frame = Table[
   If[i != 1 && i != L + 2 && j != 1 && j != L + 2, 0, 1], {i, 1,
    L + 2}, {j, 1, L + 2}];
```

Then we created the starting configurations, $\hat{0}$ and $\hat{1}$, named here as `X` and `Y`, of the Markov chains of CFTP algorithm. The variable `dist` takes into account the differences between the chains and when it turn out to be 0 then the chains have coalesced. We update it on the run, with $\mathcal{O}(1)$ complexity, instead of calculating it.

```
initialize := Module[{},
  thedists = {};
  X = Table[
    If[EvenQ[i + j] && i != 1 && i != L + 2 && j != 1 && j != L + 2,
     1, 0], {i, 1, L + 2}, {j, 1, L + 2}];
  Y = 1 - X - frame;
  dist = L^2;]
```

The real grid of numbers, excluding the frame, is obtained through the function

```
interiore[mat_] := Table[mat[[i, j]], {i, 2, L + 1}, {j, 2, L + 1}];
```

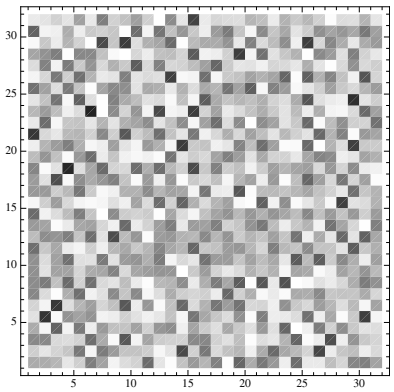
and the visualization tool is the following complicated function

```
showstate := Module[{},
   If[Length[thedists] > 0 && dist > 0,
     Print["Run at level ", levelnow,
```

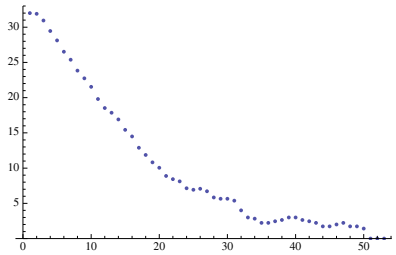
```
 ", but not succeeded. Distance is ", dist, " out of ", L^2,
 "."];
Print[GraphicsRow[{
    ListDensityPlot[1 - interiore[X],
     ColorFunctionScaling -> False, Mesh -> False,
     InterpolationOrder -> 0, DisplayFunction -> Identity],
    ListDensityPlot[1 - interiore[Y],
     ColorFunctionScaling -> False, Mesh -> False,
     InterpolationOrder -> 0, DisplayFunction -> Identity],
    ListDensityPlot[1 - interiore[Abs[X - Y]],
     ColorFunctionScaling -> False, Mesh -> False,
     InterpolationOrder -> 0, DisplayFunction -> Identity],
    ListPlot[Sqrt[thedists], PlotRange -> {0, L + 1},
     DisplayFunction -> Identity]
    }, ImageSize -> 800]],
If[Length[thedists] > 0 && dist == 0,
 Print["Run at level ", levelnow, ", and succeeded."];
 Print[GraphicsRow[{
     ListDensityPlot[1 - interiore[X],
      ColorFunctionScaling -> False, Mesh -> False,
      InterpolationOrder -> 0, DisplayFunction -> Identity],
     ListPlot[Sqrt[thedists], PlotRange -> {0, L + 1},
      DisplayFunction -> Identity]
     }, ImageSize -> 600]],
 Print["Distance is ", dist, " out of ", L^2, "."];
 Print[GraphicsRow[{
     ListDensityPlot[1 - X, ColorFunctionScaling -> False,
      DisplayFunction -> Identity],
     ListDensityPlot[1 - Y, ColorFunctionScaling -> False,
      DisplayFunction -> Identity],
     ListDensityPlot[1 - Abs[X - Y],
      ColorFunctionScaling -> False,
      DisplayFunction -> Identity]
     }, ImageSize -> 600]]];];
```

that plot the matrices in different ways depending on reaching or not coalescence.

The interesting part of the Mathematica code is the graphic one we have just seen, so we do not go on showing functions implementing the core of the simulation's algorithm, which is the purpose of the following section, but we want to display some figures.

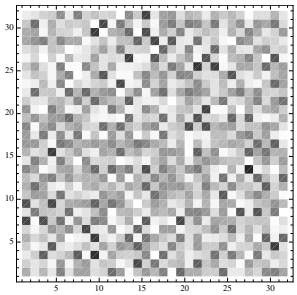


(a) Configuration of *heights* exactly sampled

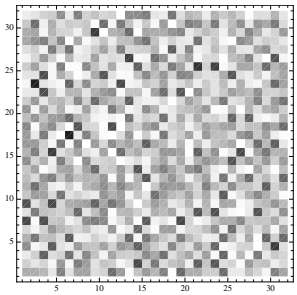


(b) Plot of the *distance* between the chains

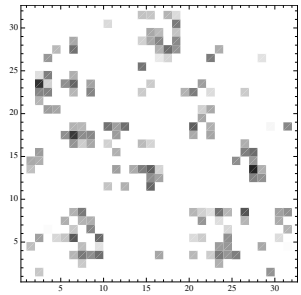
Figure 4.1 Example of a plot when the chains have coalesced.



(a) Configuration of *heights* from the first Markov chain



(b) Configuration of *heights* from the second Markov chain



(c) Grid with the differences between the Markov chains

Figure 4.2 Example of a plot when the chains have not coalesced.

4.2 C++ code

An intense use of Mathematica code for simulations, in this case we mean, is not the best. If the grid size L becomes too large, from $L = 128$ onward, the algorithm will take several minutes to output just one exactly sampled configuration. Thus we decided to run the main simulations written in C++ language and we noted a great improvement of time performances.
We chose for a dynamic use of the memory, and every grid of numbers (i.e. height's configuration)

is a double pointer to a real number. As we know, every monotone CFTP run must start from the extremal configurations, $\hat{0}$ and $\hat{1}$, here in the parity-invariant notation. We thus introduced a function that initialize the starting grids to the right values.

```
void starting()
{
  for(int i=1; i<(Lx+1); i++)
    for(int j=1; j<(Ly+1); j++)
      {
frame[i][j] = 0;
X[i][j] = (i+j)%2 == 0 ? 0 : 1;
Y[i][j] = (i+j)%2 == 0 ? 1 : 0;
      }
  for(int i=0; i< (Lx+2); i++)
    {
      frame[0][i] = frame[Lx+1][i] = frame[i][0] = frame[i][Ly+1] = 1;
      X[0][i] = X[Lx+1][i] = X[i][0] = X[i][Ly+1] = 0;
      Y[0][i] = Y[Lx+1][i] = Y[i][0] = Y[i][Ly+1] = 0;
    }
  N = Lx*Ly;
}
```

The variable `N` above is initially set to the number of points in the grid but has the same meaning as `dist` in the Mathematica code, that is two chains have coalesced if `N` is 0.
Now we show the update function which uniformly pick a vertex (i, j) of the grid and a height h; the variable `h` has a probability distribution depending on the parameter `tau` in the code. The reader should note we used one of the Gnu Standard Library random numbers generators (here `RANDOM_GEN`): this choice came after some testing simulations and looked convincing. More specifically, the generator is `ranlxs0()`.
During one step of the coupled Markov chain, the algorithm must check 4 inequalities for each chain and then 2 for testing the distance as the reader can see in the following function.

```
void update()
{
  int i,j;
  double h;
  i = 1 + gsl_rng_uniform_int (RANDOM_GEN, Lx);
  j = 1 + gsl_rng_uniform_int (RANDOM_GEN, Ly);
```

```
  h = 0.5*gsl_rng_uniform (RANDOM_GEN);
  if ( gsl_rng_uniform (RANDOM_GEN) > tau )
    h += 0.5;
  if ( fabs(X[i][j] - Y[i][j]) < 1e-9 ) N++;
  if ( (h+X[i+1][j])<=1 && (h+X[i-1][j])<=1 && (h+X[i][j+1])<=1 && (h+X[i][j-1])<=1 )
    X[i][j] = h;
  if ( (h+Y[i+1][j])<=1 && (h+Y[i-1][j])<=1 && (h+Y[i][j+1])<=1 && (h+Y[i][j-1])<=1 )
    Y[i][j] = h;
  if ( fabs(X[i][j] - Y[i][j]) < 1e-9 ) N--;
}
```

Now let's look at the main part of the CFTP algorithm written below.

```
int CFTPrun()
{
  N = Lx*Ly;
  level0 = log(Lx*Ly+1)/log(2);
  int levelnow = level0;
  while ( N>0 )
    {
      starting();
      for (int level=levelnow; level>level0; level--)
           runlevel(level);
      runlast();
      levelnow++;
    }
  return levelnow-1;
}
```

The reader should recognize the structure introduced in Section 1.2:

- the `while` loop proceeds until coalescence happens (the test variable is `N`, the distance between the chains);
- every time we restart the Markov chain we have to initialize the configurations to $\hat{1}$ and $\hat{0}$ with the `starting()` function;
- a run of the Markov chain is divided in sequent steps, each one remembering the random variable used, characterized by the use of the variable `level`: increasing this variable results in doubling the time-steps of the chain;

The importance of using the same random numbers at the same sweeps is known and we can realize this condition by suitably re-setting our random number generator using a seed. Hence we associate each `level` with a seed `rseed` and we define a function with `level` as argument.

```
void runlevel( int lev)
{
  int tmax = 1;
  tmax = tmax << lev;
  gsl_rng_set (RANDOM_GEN,(rseed+lev));
  for (int t=0; t<tmax; t++)
    update();
}
```

This function simply set the random generator to the right seed and run the update step for `tmax` times, where `tmax` is in the form $2^{\texttt{level}}$. Note how we use the bitwise operator `<<` to do the power operation. This trick speeds up all the process as we had seen with a debugging session on the code.

To conclude the description of the code we show a function that is called by the `CFTP` function at the end of every run: its meaning is clear if one note how much steps the update do. We define above the variable `level0` and if we look at the line `tmax = tmax << level0;` in the code below we can say that the Markov chain is updated a number of times equal to the number of site in the grid plus one. This means we have made the hypothesis that the last sweep which the Markov chain should do, must have the dimension, at least, of the cover time of the grid (since each update step pick up only a vertex).

```
void runlast()
{
  int tmax = 1;
  tmax = tmax << level0;
  gsl_rng_set (RANDOM_GEN,rseed);
  for (int t=0; t<tmax; t++)
      update();
}
```

For reasons of completeness, we show the main program and the definitions.

```
#define Lx 256
#define Ly 256
```

```
#define tau 0.50
#define runs 10000

gsl_rng * RANDOM_GEN;
double **frame;
double **X;
double **Y;
int **defects;

int main()
{
  RANDOM_GEN = gsl_rng_alloc(gsl_rng_ranlxs0);
  int final_level;
  ofstream output("defects.dat");
  initialize();
  initdefects();
  for (int count=0; count<runs; count++)
    {
      rseed += 32;
      final_level = CFTPrun();
      output << measuredefects(X) << ' ' << final_level << endl;
    }
  output.close();
  freemem();
  freedefects();
  gsl_rng_free (RANDOM_GEN);
}
```

5. Conclusions and perspectives

In this thesis we have been concerned with the calculation of the free energy in the classical model of corrugated surfaces. We decided for a computational approach to the final result using a Markov Chain Monte Carlo method. To avoid thermalization, correlation times and other effects affecting physical observables in these methods, we used an exact sampling algorithm called *coupling from the past*, devised by Propp and Wilson [4], which, on the other hand, outputs configurations following the exact equilibrium distribution of the Markov chain. The applicability of this algorithm to our model wasn't yet tested, so we analized the whole procedure from the beginning, finding that the simulations terminated and that the outputs were following the right distribution.

After this, it came a theoretical topic related to the mesure of a free energy from a sampling algorithm. We knew that, in a sampling procedure, the weight of each configuration is determined with an overall multiplicative constant, impossible to identify. This lead to the impossibility of measuring the free energy directly from an average procedure on the sampled configurations. We then devised the trick of studying a family of systems along a parameter curve, such that on an extremum there was our model and on the other extremum a trivial and solvable model. Along the parameter curve, we found a local observable on the lattice representing the derivative of the free energy. In this way we managed to integrate the observable to obtain the final result.

5.1 Possible issue of the final result

An article [2] appeared last year, but reviewed just this summer, is strictly related to the topic of our work. The authors, S. Majumdar and O. Martin, study the statistical properties of the number of maxima M in the model of random energy landscape, and, in a section of their work, they compute the probability of a maximally packed configuration of maxima with size N (e.g. $P(M_{max}, N)$). Following a "large deviation" law, they write this probability as

$$P(M_{max}, N) \sim \exp[-N\Phi(y_{max})] \sim \gamma^N$$

where Φ is a large deviation function and γ is a lattice dependent constant whose logarithm can be interpreted as the free energy of the system. With their simulation in 2-dimensional lattice, they obtain a value for γ, that is

$$\gamma_{\text{2-d}} \;=\; 1.6577 \pm 0.0006$$

Noting that a maximally packed configuration of N sites is just a feasible configuration with $|V| = N$ in the model of corrugated surfaces, and that its probability is the partition function in equation (2.1), the constant γ is related to our free energy, such that

$$\gamma \;=\; \exp[-f]$$

We then have too a value for γ that is

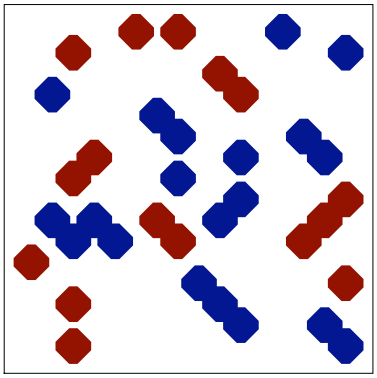
$$\gamma_{\text{corr}} \;=\; 1.7154 \pm 0.0009$$

and we see that they are not consistent.
A possible way to deal with this discrepancy could be to talk with the authors of the article and to find, if there is, an error in one of the two procedures. Our present understanding of this issue is that the authors are implicitly making an approximation of mean-field fashion (although a refined one), and the discrepancy comes from loop corrections to their procedure. A possibility could be to test their procedure against the "ladder graph", i.e. the strip of width 2, which is both exactly solvable, and has some cycles.

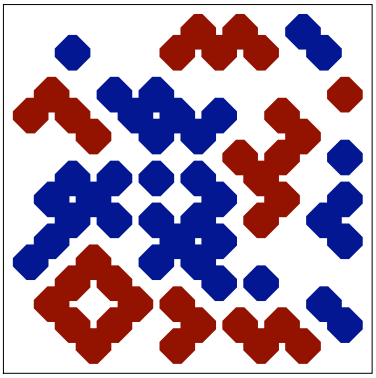
5.2 Possible presence of a phase transition

The study of the model alongside a parameter family of measures has brought to a great result, but we have explored only a restricted part of the systems dependent from the parameter ϵ. Since we now know what's the behavior of defects as ϵ goes from 0 to 1, we could ask what happens if $\epsilon \to \infty$. We could analyze, with the same statistical tools we have used in this thesis, simulations made at parameter τ values less than $1/2$.

Indeed, we have made these simulations during our preliminary checks with Mathematica. What we have noted is a significant slowing down of the coalescence time and the presence of defects collected together in clusters. These are clear clues of a possible phase transition. Since we have not investigated further on, we can only show some figures of the defects configurations, with in mind the cluster expansion of Section 2.4.

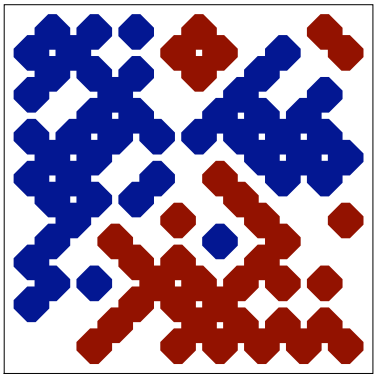


(a) Configuration of defects at $\tau = 0.500$

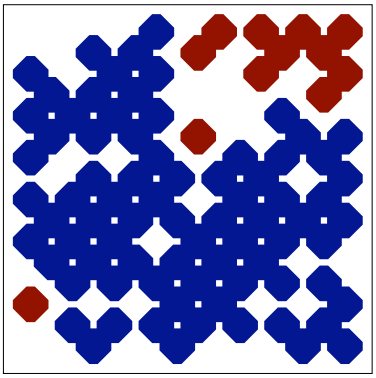


(b) Configuration of defects at $\tau = 0.200$

Figure 5.1 Example of a plot of defects for a lattice 16×16. Toward the phase transition



(a) Configuration of defects at $\tau = 0.150$



(b) Configuration of defects at $\tau = 0.100$

Figure 5.2 Example of a plot of defects for a lattice 16×16. We can see an ordered phase.

Acknowledgments

This thesis is fruit of a collaboration with Prof. Sergio Caracciolo and Dr. Andrea Sportiello. I wish to thank them for this great oppurtunity, for all the things they have thaught me and, in particular, for their patience.
I also want to thank Dr. Antonio Rago who has introduced me in the beautiful world of Monte Carlo simulations and who has given me so many advices during this last year.
A special gratitude goes to my family and to my girlfriend Valentina for their tireless presence at my side.

Bibliography

[1] David Bruce Wilson. *Exact Sampling with Markov Chains*. PhD thesis, Massachusetts Institute of Technology, June 1996.

[2] Satya N. Majumdar and Olivier C. Martin. The Statistics of the Number of Minima in a Random Energy Landscape. *Phys. Rev. E*, 74(061112), 2006.

[3] Olle Häggström. Finite Markov Chains and Algorithmic Applications. Lecture's notes, January 2001.

[4] James Propp and David Bruce Wilson. Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. *Random Structure and Algorithms*, (9):223–252, 1996.

[5] Faheem Mitha. *Perfect Sampling on Continuous State Spaces*. PhD thesis, University of North Carolina, 2003.

[6] Kerson Huang. *Statistical Mechanics*, pages 213–230. John Wiley & Sons, 1987.

[7] James Allen Fill. An Interruptible Algorithm for Perfect Sampling via Markov Chains. *Annals of Applied Probability*, (8):131–162, 1998.

[8] Morten Fismen. Exact Simulation using Markov Chains. Master's thesis, The Norwegian University of Science and Technology, December 1997.

[9] Alan D. Sokal. Monte Carlo Methods in Statistical Mechanics. Lecture's notes, September 1996.