Title

Acyclic Monte Carlo: Efficient multi-level sampling of undirected graphical models through fast marginalization

Author

Kominiarczuk, Jakub

Publication Date

2013

Peer reviewed|Thesis/dissertation

ACYCLIC MONTE CARLO

Efficient multi-level sampling of undirected graphical models through fast marginalization

by

JAKUB K. KOMINIARCZUK

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Alexandre J. Chorin, Chair
Professor Per-Olof Persson
Professor Jonathan Wilkening
Professor Andrés Rodríguez-Clare

Fall 2013

ABSTRACT

Acyclic Monte Carlo: Efficient multi-level sampling of undirected graphical models through fast marginalization

by

Jakub K. Kominiarczuk

Doctor of Philosophy in Mathematics

University of California, Berkeley

Professor Alexandre J. Chorin, Chair

We present a method for sampling high-dimensional probability spaces, applicable to Markov fields with both discrete and continuous variables, based on an approximate acyclic representation of the probability density. Our method generalizes and places in a common framework some recent work on computing renormalized Hamiltonians and stochastic multigrid sampling.

An acyclic representation of a probability distribution function (PDF) is obtained when one chooses an ordering of the variables and writes the PDF as a product of conditional probabilities, so that the probability of any variable is conditional only on the variables that precede it in the ordering. An acyclic representation makes the sampling efficient, because it uses the sparsity present in the model. We derive an approximate acyclic representation for general graphs by finding marginals through a fast marginalization scheme. The partial derivatives of the logarithm of the marginal probability are computed approximately through stochastic linear projection onto a polynomial basis, followed by reconstruction of the marginal through integration. The projection is based on an optimized inner product, making possible the use of Gaussian quadrature. Probability distributions involving discrete variables are handled by embedding the PDFs in differentiable extensions. Our algorithm can be extended to the evaluation of renormalized Hamiltonians formed using general renormalization schemes.

The approximate acyclic representation of the PDF is then used for sampling. The variables are sampled in a fixed order, producing independent samples together with their sampling weights. We present an optimized sampling strategy that uses a maximum amount of information to choose individual variable values. The samples are further improved using techniques from particle filtering. We also introduce a block Markov chain

Monte Carlo scheme based on the sampling weights. Finally, we present applications of our methodology to the Ising model.

ABSTRACTUM

Acyclicus Mons Caroli (Monte Carlo): Efficax multiæquilibris ratio ad exempla casualia formarum graphorum efficienda per quadraturam citam

Iacobus Carolus Kominiarczuk

Ph. doctor scientiarum mathematicarum

Universitas Californiensis, Berkeley

Professor Alexandre J. Chorin, qui Iacobum C. Kominiarczuk doctorem creat ac renuntiat

---

Demonstramus rationem exemplorum casualium efficiendorum ex multis dimensionibus distributionibus probabilitatis, quam possumus usu adhibere ad aream Marcovi iuxta cum quantitatibus variabilibus casualibus discretis ac variabilibus casualibus continuis, nisam in acyclicam formam distributionis probabilitatis. Ratio nostra generatim rem exponit atque communi lingua aliquam partem methodorum calculandi mediocres hamiltonianos ac probandi casualiter multis cum modis adhibendis.

Acyclica forma functionis distributivæ probabilitatis accipitur disponendo quantitatum variabilium in electo ordine scribendoque functionem distributivam ut summam functionis probabilitatis conditionalis indicatarum quantitatum variabilium ex multiplicatione effectam. Quibus specialitas est: probabilitas electæ quantitatis variabilis sita est in quantitatibus variabilibus, quæ exsistebant iam in electo ordine. Usus formæ acyclicæ distributionis propabilitatis dat facultatem fingendi citius exempla casualia, quoniam utitur paucis inter quantitates variabiles coniunctionibus. Introducimus appropinquatam acyclicam formam distributionis adhibitam usu ad quælibet grapha subiecta per computationem distributiones marginales methodo quadraturæ celeris usa. Derivatæ particulares logarithmi marginalis distributionis propabilitatis calculantur modo appropinquato per proiectionem casualem perpendiculatam in basem polynomiorum, postea calculatur distributio marginalis recuperata ex derivatis per quadraturam. Proiectio ab optima producto scalare nitens licet modo Gaussi quadratura uti. Introducto extensione differentionali ratio ad distributiones probabilitatis cum quantitatibus variabilibus casualibus discretis adhibitur. Methodon nostra accomodari potest ad calculandum mediocres hamiltonianos assecutos variarum rationum gratia.

Calculata acyclica forma distributionis probabilitatis ea utimur ad exempla casualia efficienda. Quantitates variabiles casuales in electo ordine

probantur atque libera exempla eorumque pondera dant. Demonſtramus optimatum modum ad exempla casualia efficienda a quo maximus numerus informationum ad certam quantitatem variabilem probandam adhibitur. Usis particulas colandi modis qualitas finalium exemplorum casualium in meliorem ſtatum mutatur. Introducimus etiam methodon Mons Caroli (Monte Carlo) nisam in calculatam a nobis acyclicam formam diſtributionis probabilitatis ac ponderis exemplorum casualium. Dissertationem finimus demonſtrantes eventus accomodandi rationem noſtram ad formam Isingi.

ABSTRAKT

Acykliczne Monte Carlo: Metoda efektywnego próbkowania losowego modeli graficznych poprzez szybkie ubrzegowianie

Jakub K. Kominiarczuk

Doktor nauk matematycznych

Uniwersytet Kalifornijski, Berkeley

Profesor Alexander J. Chorin, Promotor

Przedstawiamy metodę tworzenia próbek losowych z wielowymiarowych rozkładów prawdopodobieństwa, mającej zastosowanie do pól Markowa zarówno o dyskretnych, jak i ciągłych zmiennych, opartą o acykliczną formę rozkładu prawdopodobieństwa. Nasza metoda uogólnia i opisuje we wspólnym języku pewną klasę metod obliczania zrenormalizowanych hamiltonianów oraz próbkowania losowego z użyciem wielu skal.

Acykliczna forma funkcji rozkładu prawdopodobieństwa jest otrzymywana poprzez uszeregowanie zmiennych w wybranym porządku oraz zapisanie funkcji rozkładu jako iloczynu funkcji prawdopodobieństwa warunkowego poszczególnych zmiennych posiadających swoistą cechę: prawdopodobieństwo danej zmiennej jest zależne jedynie od zmiennych występujących wcześniej w wybranym porządku. Użycie acyklicznej formy rozkładu prawdopodobieństwa pozwala na efektywne tworzenie próbek losowych, ponieważ wykorzystuje niską gęstość zależności pomiędzy zmiennymi losowymi. Wprowadzamy przybliżoną acykliczną formę rozkładu stosowalną w przypadku dowolnych grafów zależności poprzez obliczanie rozkładów brzegowych z użyciem metody szybkiego ubrzegowiania. Pochodne cząstkowe logarytmu brzegowego rozkładu prawdopodobieństwa są obliczane w sposób przybliżony poprzez stochastyczne rzutowanie prostopadłe na bazę wielomianową, po czym obliczany rozkład brzegowy jest odzyskiwany z pochodnych poprzez całkowanie. Rzutowanie jest oparte o zoptymalizowany iloczyn skalarny, pozwalający na użycie całkowania metodą Gaussa. Metoda jest stosowalna do rozkładów prawdopodobieństwa ze zmiennymi dyskretnymi po wprowadzeniu rozszerzenia różniczkowalnego danego rozkładu. Nasza metoda znajduje zastosowanie do obliczania renormalizowanych hamiltonianów powstałych przy użyciu dowolnych metod renormalizacji.

Po obliczeniu acyklicznej formy rozkładu prawdopodobieństwa, używamy jej do tworzenia próbek losowych. Zmienne są próbkowane w ustalonym wcześniej porządku, dając niezależne próbki oraz ich wagi. Prezentujemy zoptymalizowaną strategię próbkowania losowego używającą maksymalną ilość informacji dostępnych do próbkowania danej zmiennej. Jakość wynikowych próbek losowych jest polepszana z użyciem technik filtrowania cząsteczek. Wprowadzamy również metodę Monte Carlo opartą o obliczoną przez nas acykliczną formę rozkładu prawdopodobieństwa oraz wagi próbek losowych. Rozprawę kończy prezentacja wyników zastosowania naszej metody do modelu Isinga.

ΠΕΡΙΛΗΨΗ

Ακυκλικά Monte Carlo: Αποτελεσματική πολυεπίπεδη δειγματοληψία ακυκλικών γραφικών μοντέλων μέσω ταχείας περιθωριοποίησης Από

Jakub K. Kominiarczuk

Δόκτωρ Φιλοσοφίας στα Μαθηματικά

Πανεπιστήμιο Καλιφόρνιας, Μπέρκλεϊ

Καθηγητής Alexander J. Chorin, προεδρεύων

Παρουσιάζουμε μια μέθοδο δειγματοληψίας χώρων πιθανότητας μεγάλων διαστάσεων, εφαρμόσιμη σε πεδία Markov τόσο με διακριτές όσο και συνεχείς μεταβλητές, βασισμένη σε μια προσεγγιστική ακυκλική αναπαράσταση της συνάρτησης πυκνότητας πιθανότητας. Η μέθοδός μας γενικεύει και τοποθετεί σε ένα κοινό πλαίσιο πρόσφατες εργασίες σχετικά με τον υπολογισμό επανακανονικοποιημένων Χαμιλτονιανών και τη στοχαστική πολυπλεγματική δειγματοληψία.

Μια ακυκλική αναπαράσταση μιας συνάρτησης πυκνότητας πιθανότητας (σ.π.π.) επιτυγχάνεται όταν επιλεγεί μια διάταξη των μεταβλητών και γραφεί η σ.π.π. ως γινόμενο δεσμευμένων πιθανοτήτων, έτσι ώστε η πιθανότητα κάθε μεταβλητής να εξαρτάται μόνο από τις μεταβλητές που προηγούνται αυτής στη διάταξη. Μια ακυκλική αναπαράσταση κάνει τη δειγματοληψία αποτελεσματική, επειδή χρησιμοποιεί τη σποραδικότητα που υπάρχει στο μοντέλο. Εξάγουμε μια προσεγγιστική ακυκλική αναπαράσταση για γενικά γραφήματα βρίσκοντας τις περιθώριες συναρτήσεις μέσω ενός γρήγορου συστήματος περιθωριοποίησης. Οι μερικές παράγωγοι του λογαρίθμου της περιθώριας συνάρτησης πιθανότητας υπολογίζονται προσεγγιστικά μέσω στοχαστικής γραμμικής προβολής σε μία πολυωνυμική βάση, ακολουθούμενη από την ανακατασκευή της περιθώριας συνάρτησης μέσω ολοκλήρωσης. Η προβολή βασίζεται σε ένα βελτιστοποιημένο εσωτερικό γινόμενο, που καθιστά δυνατή τη χρήση των Gaussian τετραγωνισμών. Οι κατανομές πιθανότητας που αφορούν διακριτές μεταβλητές αντιμετωπίζονται με την ενσωμάτωση της σ.π.π. σε διαφορίσιμες επεκτάσεις. Ο αλγόριθμός μας μπορεί να επεκταθεί στην αξιολόγηση των επανακανονικοποιημένων Χαμιλτονιανών που σχηματίζονται χρησιμοποιώντας γενικές μεθόδους επανακανονικοποίησης.

Η προσεγγιστική ακυκλική αναπαράσταση της συνάρτησης πυκνότητας πιθανότητας χρησιμοποιείται στη συνέχεια για δειγματοληψία.

Παίρνουμε δείγματα μεταβλητών με μία συγκεκριμένη σειρά, παράγοντας ανεξάρτητα δείγματα σε συνδυασμό με τα βάρη δειγματοληψίας τους. Σας παρουσιάζουμε μια βελτιστοποιημένη στρατηγική δειγματοληψίας που χρησιμοποιεί μια μέγιστη ποσότητα πληροφορίας για να επιλέξει μεμονωμένες τιμές μεταβλητών. Τα δείγματα βελτιώνονται περαιτέρω χρησιμοποιώντας τεχνικές φιλτραρίσματος σωματιδίων. Επίσης, παρουσιάζουμε ένα μπλοκ σύστημα Markov Chain Monte Carlo με βάση τα βάρη της δειγματοληψίας. Τέλος, παρουσιάζουμε εφαρμογές της μεθοδολογίας μας στο μοντέλο Ising.

תקציר

דגימת מונטה-קארלו אציקלית: דגימה רב-שכבתית יעילה של מודלים גרפיים לא מכוונים בעזרת דחיקה לשוליים מהירה

נכתב על ידי

יקוב קומיניארצ'וק

דוקטור לפילוסופיה במתמטיקה

אוניברסיטת קליפורניה, ברקלי

אלכסנדר צ'ורין, יושב ראש

אנו מציגים שיטה לדגימת מרחבי הסתברות בעלי מימד גבוה, עם ישומים לשדות מרקוב עם משתנים בדידים ורציפים, המבוססת על ייצוג אציקלי מקורב של פונקצית הסתברות מצטברת. השיטה שלנו מכלילה ומאחדת במסגרת אחידה את חלק מהמחקר שיצא לאחרונה על חישוב המילטוניאנים מנורמלים ודגימה סטוכסטית על מולטי-שריגים.

הייצוג האציקלי של פונקציית הסתברות מצטברת מתקבל כאשר בוחרים סידור של המשתנים וכותבים את פונקציית הסתברות כמכפלה של הסתברויות מותנות, כך שההתסברות של המשתנה מותנה בהסתברויות של משתנים הקודמים לו בסידור. הייצוג האציקלי מייעל את הדגימה, מכיוון שהוא מתשמש בדלילות שישנה במודל. אנו מפתחים את הייצוג האציקלי המקורב לגרפים כלליים על ידי מציאת הסתברות שולית דרך שיטה מהירה של "דחיקה לשוליים". הנגזרת החלקית של הלוגריתם של הסתברות שולית מחושב בקירוב דרך הטלה לינארית אקראית לבסיס פולינומיאלי ואחריה שיחזור של הסתברות השולית בעזרת אינטגרציה. ההטלה מבוססת על מכפלה פנימית מיועלת שמתאפשרת עקב השימוש בתרבועי גאוס. התפלגויות הסתברות של משתנים בדידים מחושבות בעזרת שיכון של פונקצית הסתברות מצטברת בתוך הרחבה חלקה. את האלגוריתם שלנו ניתן להכליל להמילטוניאנים מנורמלים המתקבלים מתוך שיטות רנורמליזציה כלליות.

לאחר מכן, הייצוג האציקלי מיושם לדגימה. המשתנים נדגמים בסדר קבוע מראש, בכך מתקבלים מדגמים בלתי תלויים יחד עם משקולות הדגימה שלהם. אנו מציגים שיטת דגימה יעילה אשר משתמשת בכמות המקסילית של המידע כדי לבחור ערכים של משתנים בודדים. בנוסף, אנו משפרים את המדגמים על ידי שימוש בטכניקות של סינון חלקיקים. אנו מציעים שיטת מונטה-קרלו של בלוקים של שרשראות מרקוב המבוססת על משקולות דגימה. לסיום, אנו מציגים יישומים של המתודולוגיה שלנו למודל אייסינג.

ZUSAMMENFASSUNG

Azyklisches Monte Carlo: die Methode des effizienten stochastischen Samplings durch schnelle Marginalisierung

Jakub K. Kominiarczuk

Doktor der mathematischen Wissenschaft

Die kalifornische Universität, Berkeley

Professor Alexander J. Chorin, Betreuer

Wir praesentieren eine Methode um Zufallsvariablen in hochdimensionale Wahrscheinlichkeitsraeumen darzustellen. Die Methode findet Anwendung in Markov Feldern mit diskreten und kontinuierlichen Variablen und basiert auf einer angenaeherten, azyklischen Repraesentation der Wahrscheinlichkeitsdichte. Unsere Methode generalisiert und vereinigt Ansaetze fuer renormalisierte Hamiltonische Systeme und stochastische Multigrid Verfahren.

Eine azyklische Repraesentation einer Wahrscheinlichkeitsdichte wird erreicht in dem den Zufallsvariablen eine Ordung zugewiesen wird und die Wahrscheinlichkeitsdichte als Produkt konditionierter Wahrscheinlichkeitsdichten geschrieben wird, wobei die Wahrscheinlichkeit einer Variable nur auf jene Zufallsvariablen konditioniert ist die in der gegebenen Ordnung vorangehen. Die azyklische Representation beschleunigt das generieren von Realisationen der Variable da die duennbesetzte Modellstruktur genutzt werden kann. Wir leiten eine Annaeherung an die azyklische Representation fuer allgemeine Graphen her, in dem wir schnelle Marginalisierungen nutzen. Die partiellen Ableitungen des Logarithmus der Marginale werden durch stochastische lineare Projektionen auf eine Polynom-Basis angenaehert, welche dann einfach integriert werden koennen. Die Projektion basiert auf einem optimalen inneren Produkt, so das Gauss-Quadratur genutzt werden kann. Differenzierbare Erweiterungen werden fuer diskrete Zufallsvariablen angewendet. Unsere Methode kann auch zur Auswertung renormalisierter Hamiltonischer Systeme, die aus generalisierter Renormalisierung hervorgehen, genutzt werden.

Wir nutzen die azyklische Representation der Wahrscheinlichkeitsdichte um Stichproben zu generieren. Die Stichproben der einzelnen Zufallsvariablen werden der gegebenen Ordnung nach erzeugt, so das die Stichproben und deren Gewichte unnabhaengig voneinander sind. Wir presentieren eine optimierte Strategie die maximale Information benutzt um

einzelne Stichproben zu generieren. Die Stichproben werden dann mit Hilfe von Methoden der “particle filter” weiter verbessert. Daneben stellen wir eine Strategie fuer blockweise Markov-Ketten-Monte-Carlo vor das auf den Gewichten der Stichproben basiert. Schliesslich zeigen wir die Anwednung unserer Methoden am Ising Modell.

АННОТАЦИЯ

Ациклический метод Монте-Карло: многоуровневая выборка моделей на неориентированных графах посредством быстрого интегрирования

Якуб Кониарчук

Кандидат физико-математических наук

университет Калифорнии, Беркли

Председатель диссертационной комиссии: профессор Александр Чорин

Представлен метод выборки в многомерных вероятностных пространствах, применимый к марковским полям как с дискретными, так и с непрерывными переменными, основанный на приближенном ациклическом представлении функции плотности. Данный метод обобщает некоторые недавние работы по вычислению ренормализованных гамильтонианов и выборок многосеточным методом и дает им новую интерепретацию.

Ациклическое представление вероятностной функции распределения получается при выборе порядка следования переменных и записи функции распределения как произведения условных вероятностей таким образом, чтобы условная функция распределения каждой переменной зависела только от предшествующих переменных. Ациклическое представление повышает эффективность выборки, так как оно использует разрешенность исследуемой модели. С использованием схемы с быстрым интегрированием для поиска полных вероятностей получено приближенное ациклическое представление для произвольных графов. Частные производные логарифма полных вероятностей вычисляются приближенно через вероятностную линейную проекцию на базис, состоящий из полиномов; затем полная вероятность восстанавливается интегрированием. Проекция основана на оптимизированном скалярном произведении, позволяющем использовать метод численного интегрирования Гаусса. Для вероятностных пространств с дискретными переменными применяется вложение в дифференцируемые расширения. Предложенный алгоритм может быть обобщен для вычисления ренормализованных гамильтонианов, полученных при помощи общих схем ренормализании.

Приближенное ациклическое представление функции плотности затем используется для выборки. Выборка переменных производится в фиксированном порядке, в результате чего получаются независимые выборки с соответствующими весами. Представлена оптимизированная стратегия выборки, использующая наибольшее количество информации для выбора значений каждой переменной. Затем выборки улучшаются посредством методов фильтрования частиц. Также описана схема Монте-Карло на блочных марковских цепях, использующая веса выборки. В завершение представлены приложения разработанных методов к модели Изинга.

ÖZET

Çevrimsiz Monte Carlo: Yönsüz grafiklerin çok katlı örneklenmesi için hızlı marjinal almaya dayanan etkili bir yöntem

Jakub K. Kominiarczuk

Matematik Doktorası

Kaliforniya Üniversitesi, Berkeley

Danışman: Profesör Alexandre J. Chorin

Bu tezde, çok boyutlu uzaylarda tanımlanan olasılık dağılımlarından örnekleme yapmak için geliştirilen ve hem ayrık hem de sürekli değişkenli Markov alanlarına uygulanabilen bir yöntem sunulmaktadır. Bu yöntem, verilen bir olasılık dağılımının çevrimsiz yaklaşık bir gösterimini temel almaktadır. Yöntemimiz, yeniden normalize edilmiş Hamiton hesaplama ve stokastik çok katmanlı ızgara örneklemesi üzerine yapılmış bir takım güncel çalışmaları genellemekte, bunları bir çerçeve içine almaktadır.

Bir olasılık dağılımının çevrimsiz gösterimi, dağılımdaki değişkenlerin koşullu olasılıklarının çarpımıdır. Bu koşullu olasılıklar, tercih edilen belli bir sıraya göre her bir değişkenin sadece kendinden önceki değişkenlere koşullandırılmasından elde edilir. Çevrimsiz gösterim, verilen bir modeldeki seyrekliği kullanarak örneklemenin verimli olmasını sağlayabilir. Bu çalışmada, genel grafikler için verilen bir dağılımın marjinalleri bir hızlı marjinal hesaplama yolu ile hesaplanarak bu dağılımın çevrimsiz yaklaşık bir gösterimi türetilmiştir. Marjinal olasılığın logaritmasının kısmi türevlerinin yaklaşık olarak hesaplanması ise önce bir polinom tabanına yapılan stokastik izdüşüm, ardından da marjinal dağılımın integralle geri çatılması işlemleri ile gerçekleştirilmiştir. Söz konusu izdüşüm bir eniyilenmiş iç çarpıma dayanmakta olup, Gauss dördününün kullanılmasını mümkün kılmaktadır. Ayrık değişken içeren olasılık dağılımları ile, bu dağılımların türevlenebilen uzantılarına gömülmesi suretiyle çalışılmıştır. Yöntemimiz, genel yeniden normalize etme yolları kullanılarak oluşturulmuş yeniden normalize edilmiş Hamitonları hesaplamak için de uyarlanabilir.

Olasılık yoğunluk dağılımının yöntemimiz ile elde edilen yaklaşık gösterimi daha sonradan örnekleme için kullanılmaktadır. Değişkenler belli bir sırayla örnekleme ağırlıklarıyla birlikte üretilmektedir. Tek tek değişkenlerin değerlerini seçmek için eldeki bilgiyi olası en yüksek miktarda kullanan bir örnekleme stratejisi geliştirilmiştir. Elde edilen örnekler parçacık süzgeci teknikleri kullanılarak bir iyileştirmeye tabi tutulmuştur.

Ayrıca, örnekleme ağırlıklarına dayanan bir blok Markov zinciri Monte Carlo yöntemi tanıtılmıştır. Son olarak, yöntemimizin Ising modeline uygulaması gösterilmiştir.

When we try to pick out anything by itself,
we find that it is bound fast by a thousand
invisible cords that cannot be broken,
to everything in the universe.

— John Muir

ACKNOWLEDGMENTS

My journey as a graduate student at University of California, Berkeley, was a wonderful experience, whose greatest component were the people I had the privilege to meet along the way.

First and foremost, I would like to sincerely thank my doctoral advisor Professor Alexandre J. Chorin for his unwavering support throughout my graduate career. Without him my studies would not have been possible and for that, his friendship and advice, I will forever be in his debt.

I am thankful to the host of friends that surrounded me at Berkeley, both at the Department of Mathematics and the Lawrence Berkeley National Laboratory. I would like to thank Семён Владимирович Дятлов (Semyon Vladimirovich Dyatlov), whose dancing talent is matched only by his mathematical acumen and sense of humor, for sharing our student office for the past five years, the countless discussions and help in more matters than could be listed. I am most glad I had the pleasure to meet and befriend Valerie Heatlie, Matthias Morzfeld, Robert Saye, Chris *VoroMaster* Rycroft and Per-Olof Persson, who all spent their time discussing research problems, helped in a plethora of ways over the years and continue to do so on a moment's notice. I would like to thank my supervisor Professor James A. Sethian for his patience, advice and support in times of need. My thoughts are with Valerie Heatlie and Professor Grigory Isaakovich Barenblatt, who were always the heart and soul of the mathematics group.

I extend my gratitude to the members of my dissertation committee, Per-Olof Persson, Jon Wilkening and Andrés Rodríguez-Clare for their time, guidance and support along the way. I would like to thank Panayiota Constantinou, Sinan Yildirim, Matthias Morzfeld, Semyon Dyatlov, Boris Ettinger and my mother, Renata Kominiarczuk, for providing the abstract translations for this thesis, turning it into the modern day Rosetta stone.

There is not enough space to express the sense of indebtedness to my family — parents, siblings and grandmother — for always being the rock that one can stand on when all else crumbles and supporting me through-

out my life; for that, and everything that is to come, I sincerely thank you.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

PDF Probability Distribution Function

MCMC Markov Chain Monte Carlo

IS Independent Set

VC Vertex Cover

MVC Minimum Vertex Cover

MIS Maximum Independent Set

DAG Directed Acyclic Graph

CSD Conditional Sampling Distribution

SIS Sequential Importance Sampling

SIR Sequential Importance Resampling

FRC Full Rejection Control

PRC Partial Rejection Control

LAPACK Linear Algebra Package

VTK Visualization Toolkit

MCRG Monte Carlo Renormalization Group

CHMC Chainless Monte Carlo

COLAMD Column Approximate Minimum Degree Ordering Algorithm

CSD Conditional Sampling Distribution

PAC Product of Approximate Conditionals

PDE Partial Differential Equation

NOTATION

PROBABILITY

x, y, z	Random variables
$\boldsymbol{x}_V$	Vector of random variables whose components correspond to a lattice (set of nodes) V
$\boldsymbol{x}_{V \setminus U}$	Vector of random variables corresponding to the subset $V \setminus U \subset V$
x_u	Component of a vector of random variables corresponding to a node u on a lattice
$P(\boldsymbol{x}_V)$	Probability distribution of random variables corresponding to the nodes in V
$W(\boldsymbol{x}_V)$	Hamiltonian corresponding to $P(\boldsymbol{x}_V)$, defined as $P(\boldsymbol{x}_V) = \exp(W(\boldsymbol{x}_V))/Z_V$
Z_V	Partition function, a normalization constant defined as $Z_V = \int \exp(W(\boldsymbol{x}_V)) d\boldsymbol{x}_V$
x_u	Component of a vector of random variables corresponding to a node u on a lattice
$P(x \mid y)$	Conditional probability distribution of x given y
$x \perp\!\!\!\perp y \mid z$	x is conditionally independent of y given the value of z
$\mathbf{E}[f(\boldsymbol{x}_V)]$	Expected value of a function $f(\boldsymbol{x}_V)$ with respect to $P(\boldsymbol{x}_V)$

G	Graph, written as $G = (V, E)$
V	Set of nodes u
E	Set of undirected edges (u, v)
$N(u)$	Set of neighbors of a node $u \in V$
$\bar{N}(u)$	*Closed* set of neighbors, $\bar{N}(u) = N(u) \cup u$

D	Directed graph (digraph), written as $D = (V, A)$
A	Set of directed edges (arcs) $(\overrightarrow{u, v})$ from node u to node v, for $u, v \in V$
$N_p(u)$	Set of direct predecessors (parents) of a node $u \in V$, defined as a set of nodes $v \in V$ such that $(\overrightarrow{v, u}) \in A$
$N_s(u)$	Set of direct successors (children) of a node $u \in V$, defined as a set of nodes $v \in V$ such that $(\overrightarrow{u, v}) \in A$
$S(u)$	Set of successors of a node $u \in V$, defined as a set of nodes that can be reached from u using arcs in A

1 INTRODUCTION

The concept of renormalization arose in the study of critical behavior in thermodynamical systems, where one is interested in studying the scaling behavior of the system. Renormalization is often presented in the context of the Ising model, in which spins — as detailed below — on a regular Cartesian lattice interact with their nearest neighbors. The Ising model is then renormalized by integrating away a fraction of the spin variables. The study of the scaling of various properties of the system allows the investigation of the properties of the critical transition occurring in the model.

The random variables $\boldsymbol{x}_V$ of the Ising model live on a Cartesian lattice V and can take only two values, either -1 or 1, corresponding to a spin pointing down or up. The probability of a configuration is given by

$$P(\boldsymbol{x}_V) = \exp\left(\frac{J}{2T}\sum_u x_u \sum_{v\in N(u)} x_v\right),$$

where T is the temperature, J a coupling constant and $N(u)$ the set of nearest neighbors of the node u. The Ising model can be generalized to an arbitrary number of dimensions and will be used throughout this work as an model example, due to the wide array of results available.

1.1 RENORMALIZATION

We begin the discussion with definitions of terms that will be mentioned throughout the thesis. Let $P(\boldsymbol{x}_V)$ be a probability distribution function defined for a finite vector of random variables $\boldsymbol{x}_V$, which form a lattice V. Assuming that $P(\boldsymbol{x}_V) > 0$, i.e. it is a Gibbs measure, we define the *Hamiltonian* $W(\boldsymbol{x}_V)$ to be the logarithm of the probability distribution,

$$P(\boldsymbol{x}_V) = \exp(W(\boldsymbol{x}_V))\Big/Z_V,$$

where the partition function $Z_V = \int \exp(\boldsymbol{x}_V)d\boldsymbol{x}_V$ is a normalization constant ensuring that $\int P(\boldsymbol{x}_V)d\boldsymbol{x}_V = 1$. For reasons of generality and simplicity of notation, the definition of Hamiltonian used within the present thesis is the negative of the potential energy of the system and absorbs

all physical parameters, such as the temperature T or coupling strength J.

By the Hammersley-Clifford theorem, the Hamiltonian may be decomposed into a sum of interactions between the variables. Therefore, we may write

$$W(\boldsymbol{x}_V) = \sum_i c_i \Phi_i(\boldsymbol{x}_V). \tag{1.1}$$

where the functions $\Phi_i(\boldsymbol{x}_V)$ represent the interactions between the random variables $\boldsymbol{x}_V$. The coupling coefficients c_i specify the relative strengths of these interactions and may be written as a vector $\boldsymbol{c} = (c_1, c_2, \dots, c_K)$. Among the common interactions are the nearest-neighbor interaction $x_u x_v$, where the nodes u and v are nearest neighbors on the lattice V, and the plaquette $x_u x_v x_w x_t$, with the nodes u, v, w and t forming a square tile on the lattice.

We renormalize by coarsening the finite system, dividing the original *fine variables* $\boldsymbol{x}_V$ into subsets and assigning each subset a group variable, thus obtaining a set of *coarse variables* $\boldsymbol{x}_U$. Following this coarsening, we compute the Hamiltonian that defines the probability distribution of the coarse variables.

For example, under a renormalization rule one may decrease the size of the lattice by a factor $b = 2$ through the creation of 2×2 blocks of variables, and assigning a group variable to each block. One obtains a new set of variables $\boldsymbol{x}_U$ that live on a lattice U coarser by a linear factor of two than the lattice V occupied by $\boldsymbol{x}_V$. The variables $\boldsymbol{x}_U$ are then related to $\boldsymbol{x}_V$ through a renormalization rule (e.g., decimation and majority rule defined in detail in Section 6.6) which gives the conditional probability $P(\boldsymbol{x}_U \mid \boldsymbol{x}_V)$ of a state $\boldsymbol{x}_U$ given a configuration $\boldsymbol{x}_V$. The renormalization rule describes the connection between the original variables $\boldsymbol{x}_V$, the renormalized variables $\boldsymbol{x}_U$ and their probabilities; the joint distribution of the two sets of variables is given by the Bayes' rule

$$P(\boldsymbol{x}_U, \boldsymbol{x}_V) = P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V).$$

Our interest lies in the probability density of the renormalized system $\boldsymbol{x}_U$,

$$P(\boldsymbol{x}_U) = \int P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) d\boldsymbol{x}_V.$$

We assume that the renormalized probability distribution $P(\boldsymbol{x}_U)$ can be written as

$$P(\boldsymbol{x}_U) = \exp(W(\boldsymbol{x}_U)) \Big/ Z_U.$$

$W(\boldsymbol{x}_U)$ is the renormalized Hamiltonian, which takes the general form

$$W(\boldsymbol{x}_U) = \sum_i c_i' \Phi_i(\boldsymbol{x}_U). \tag{1.2}$$

While the interactions Φ_i are frequently of the same functional form on both the original and renormalized lattices, they need not be. Even when they are of the same form, typically the renormalized Hamiltonian has more non-zero coupling constants $\boldsymbol{c}'$ than the original one: e.g., a Hamiltonian $W(\boldsymbol{x}_V)$ consisting of only the nearest-neighbor interaction may produce a renormalized Hamiltonian $W(\boldsymbol{x}_U)$ that includes both the nearest neighbor interaction, interactions with second-nearest neighbors, and the plaquette interaction.

Numerous quantities of physical interest can be computed using renormalization techniques. The most basic of those are the renormalized coupling coefficients $\boldsymbol{c}'$. The behavior of these coefficients under renormalization indicates the general behavior of the system; for example, when the coupling coefficients do not change under renormalization, we are dealing with a *fixed point* of the renormalization. A system described by the fixed point coefficients does not change under renormalization, which means that it behaves in the same way at every observable scale. Finding the fixed points is of great interest as they may be related to phase transitions and can be used to study them.

In principle, performing a marginalization requires the integration of the joint probability distribution function over an enormous number of variables, a task impossible to perform exactly in case of statistical models of reasonable size. Instead, approximate methods must be used.

The foundations of renormalization methods were laid by Kadanoff (1966), who proposed to divide the spins of the Ising model into subsets (cells) and study the interactions between the cells rather than simply between the individual spins. He used this model to study the Ising model around the phase transition by calculating the free energy of the system; however, Kadanoff (1966) did not attempt to compute the renormalized coefficients $\boldsymbol{c}'$. In later papers, Kadanoff (1975) and Kadanoff and Houghton (1975) define the renormalized coupling coefficients and a renormalization transformation $R : \boldsymbol{c} \to \boldsymbol{c}'$ connecting the two, and use them to

study the behavior of the Ising model and the renormalized coefficients with the help of perturbation theory.

Many methods have since been devised to compute the renormalized coefficients or to directly find fixed points of the renormalization transformation. Wilson (1971a,b, 1980) attempts to find a set of renormalized coefficients $\boldsymbol{c}'$ such that the so called correlation functions

$$\langle \Phi_i(\boldsymbol{x}_U) - \Phi_i(\boldsymbol{x}_V) \rangle = 0$$

are zero for the two systems, the fine system $\boldsymbol{x}_V$ and the renormalized system $\boldsymbol{x}_U$ described by couplings $\boldsymbol{c}$ and $\boldsymbol{c}'$, respectively. Unfortunately, such a method is not very efficient and thus limited to special cases (Swendsen, 1984b).

Following Kadanoff (1975) and Kadanoff and Houghton (1975), the renormalization process can be seen as a mapping $R : \boldsymbol{c} \to \boldsymbol{c}'$ of the original set of coefficients $\boldsymbol{c}$ to the renormalized set $\boldsymbol{c}'$ (Nauenberg and Nienhuis, 1974b). At the fixed point $\boldsymbol{c}^*$ we have $\boldsymbol{c}^* = R(\boldsymbol{c}^*)$, while in the vicinity of the fixed point $\boldsymbol{c}^*$ the renormalization transformation R may be expanded in a Taylor series as

$$\boldsymbol{c}' = \boldsymbol{c}^* + A(\boldsymbol{c} - \boldsymbol{c}^*) + \mathcal{O}(|\boldsymbol{c} - \boldsymbol{c}^*|^2).$$

with A being the Jacobian of the renormalization mapping R evaluated at the fixed point. Nauenberg and Nienhuis (1974a,b) found the location of the fixed point using a 4×4 lattice, finding it to be located at $c_1 = 0.307$, $c_2 = 0.084$, $c_3 = -0.004$ (Nauenberg and Nienhuis, 1974b) or $c_1 = 0.300$, $c_2 = 0.0871$, $c_3 = -0.00126$ (Nauenberg and Nienhuis, 1974a). However, the authors do not explain how these values were found; a likely different method is discussed by Binney et al. (1992), who quote the fixed point location found by Nauenberg and Nienhuis (1974b).

Ma (1976) took a different approach. He simulated the original lattice $\boldsymbol{x}_V$ using the Ising probability distribution $P(\boldsymbol{x}_V)$ and from each state $\boldsymbol{x}_V$ he generated samples of the group spins $\boldsymbol{x}_U$ using the known conditional probability $P(\boldsymbol{x}_U \mid \boldsymbol{x}_V)$. The renormalized parameters $\boldsymbol{c}'$ could then obtained by observing the probabilities with which the group spins flip under different circumstances. The method of Ma (1976) takes the detailed balance equations for the renormalized lattice

$$\begin{aligned}\frac{P(x_u \to -x_u \mid \boldsymbol{x}_{U\setminus u})}{P(-x_u \to x_u \mid \boldsymbol{x}_{U\setminus u})} &= \frac{\exp(W(-x_u, \boldsymbol{x}_{U\setminus u}) - W(x_u, \boldsymbol{x}_{U\setminus u}))}{\exp(W(x_u, \boldsymbol{x}_{U\setminus u}) - W(-x_u, \boldsymbol{x}_{U\setminus u}))} \\ &= \exp(2W(-x_u, \boldsymbol{x}_{U\setminus u}) - 2W(x_u, \boldsymbol{x}_{U\setminus u}))\end{aligned}$$

that can be written for every configuration of the neighboring spins, and attempts to solve them to obtain the renormalized coupling parameters $\boldsymbol{c}'$. The quantity on the left hand side is estimated using a Monte Carlo simulation of the renormalized lattice, with samples generated using the original probability distribution $P(\boldsymbol{x}_V)$ and the conditional $P(\boldsymbol{x}_U \mid \boldsymbol{x}_V)$. The right hand side is of known form, with $W(\boldsymbol{x}_U)$ given by Equation 1.2, but with unknown coupling coefficients $\boldsymbol{c}'$. The equation can be rewritten as

$$\ln\left(\frac{P(x_u \to -x_u \mid \boldsymbol{x}_{U\setminus u})}{P(-x_u \to x_u \mid \boldsymbol{x}_{U\setminus u})}\right) = 2W(-x_u, \boldsymbol{x}_{U\setminus u}) - 2W(x_u, \boldsymbol{x}_{U\setminus u}),$$

where the right hand side is typically linear in the coupling coefficients. For example, in the case of only one coupling coefficient related to the nearest neighbors, the right hand side becomes

$$2W(-x_u, \boldsymbol{x}_{U\setminus u}) - 2W(x_u, \boldsymbol{x}_{U\setminus u}) = -2c_1 \sum_{N(u)} x_v;$$

however, with four spins in the nearest neighbor set $N(u)$ one can write ten equations: there are two possible values of x_u and five possible values of $\sum_{N(u)} x_v$. Therefore, the resulting linear constraints on c_1 must be solved approximately in the least squares sense due to the inevitable stochastic errors involved in the estimation of the left hand side through Monte Carlo simulation.

Using the same linearization of the renormalization mapping R as Nauenberg and Nienhuis (1974a,b), Swendsen (1979a) introduced the Monte Carlo Renormalization Group (MCRG) method for studying the critical exponents of the Ising model. Swendsen found formulas for the matrix elements of the Jacobian A through the use of chain rule applied to derivatives of the expected interaction strengths $\langle \Phi_i \rangle$, where one obtains

$$\begin{aligned}\sum_i \left(\langle \Phi_j(\boldsymbol{x}_V)\Phi_i(\boldsymbol{x}_V)\rangle - \langle \Phi_j(\boldsymbol{x}_V)\rangle\langle \Phi_i(\boldsymbol{x}_V)\rangle\right) A_{ik} \\ = \langle \Phi_j(\boldsymbol{x}_V)\Phi_k(\boldsymbol{x}_U)\rangle - \langle \Phi_j(\boldsymbol{x}_V)\rangle\langle \Phi_k(\boldsymbol{x}_U)\rangle.\end{aligned}$$

The renormalized system $\boldsymbol{x}_U$ is simulated using Monte Carlo in the same manner as in Ma (1976). The eigenvalues of the matrix $A = (A_{ik})$ provide the critical exponents. While the renormalization methods discussed thus far are frequently termed *real-space* renormalization, Swendsen (1981) describes a related method using the momentum-space representation of the

Hamiltonian, where the variables are the coefficients of Fourier modes on the lattice; the two formulations are formally equivalent.

Swendsen (1979b) performed a study of the two-dimensional Ising model with different numbers of coupling coefficients, obtaining $\nu = 0.998$, $\alpha = 0.004$, $\beta = 0.1259$ and $\gamma = 1.744$, in good agreement with the exact values of $\nu = 1$, $\alpha = 0$, $\beta = 0.125$ and $\gamma = 1.750$. He also briefly analyzed the eigenvalues of A coming from the three-state Potts model in two dimensions. This calculation was then extended in Swendsen and Berker (1983), where they computed the renormalized coupling coefficients and critical exponents for the three-state Potts model in two dimensions.

Swendsen and Wang (1987) then applies the MCRG method to study the critical exponents and the location of the phase transition in a $\pm J$ spin glass model in dimensions two, three and four. The critical coupling J_c is determined as the crossing point of the renormalization group scaling exponent $y_H(n, J)$.

Swendsen (1984a,b,c) uses methodology related to the Monte Carlo Renormalization Group to study the behavior of coupling coefficients under renormalization. Assume one has the ability to sample the renormalized variables exactly, for example using the method used earlier by Ma (1976) and Swendsen (1979a,b). Let the our current guess for the renormalized coefficients be $\tilde{\boldsymbol{c}}'$ and denote the *local interactions* around a variable x_u as $\hat{\Phi}_{i,u}$; for example, the nearest neighbor term becomes

$$\hat{\Phi}_{i,u} = \sum_{N(u)} x_v.$$

With m_i being the number of variables showing up in the interaction Φ_i, we define

$$\langle \tilde{\Phi}_i \rangle = m_i^{-1} \sum_u \left\langle \hat{\Phi}_{i,u} \tanh \left(\sum_j \tilde{c}'_j \hat{\Phi}_{j,u} \right) \right\rangle .$$

Swendsen (1984a) notes that the equality $\langle \tilde{\Phi}_i \rangle = \langle \Phi_i \rangle$ would hold only if $\tilde{\boldsymbol{c}}' = \boldsymbol{c}'$, i.e., the guess couplings were exact. Otherwise, in the vicinity of the exact coupling $\boldsymbol{c}'$ the difference becomes

$$\langle \tilde{\Phi}_i \rangle - \langle \Phi_i \rangle = \sum_j \frac{\partial \langle \tilde{\Phi}_i \rangle}{\partial \tilde{c}'_j} (\tilde{c}'_j - c'_j),$$

which can be solved by matrix inversion for the exact coefficients $\boldsymbol{c}'$. Two steps of such iteration are sufficient to obtain good estimates of $\boldsymbol{c}'$. Swendsen (1984b) uses this method to compute renormalized coupling coefficients for the critical ($\mu = 0.440687$) two-dimensional Ising model on a 32×32 lattice, using both decimation and majority rule renormalization methods. Using the same method, Swendsen (1984c) computed coupling coefficients for the critical ($\mu = 0.22166$) three-dimensional Ising model on a $32 \times 32 \times 32$ lattice under majority rule renormalization. Finally, Swendsen (1984a) studied the flow of coupling coefficients in the two-dimensional Ising model on 32×32 lattice under majority rule with up to seven interactions and briefly the three-dimensional Ising model on $32 \times 32 \times 32$ lattice with seventeen interactions.

Using a method similar to that of Swendsen (1984a), Gupta and Cordery (1984) computed the coupling coefficients of the critical two-dimensional Ising model under majority rule renormalization. In their method, states $\boldsymbol{x}_U$ are sampled from the probability

$$\boldsymbol{x}_U \sim P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) \times \exp\left(-\tilde{W}(\boldsymbol{x}_U)\right) / Z_U,$$

where $\tilde{W}(\boldsymbol{x}_U)$ is the current guess for the renormalized Hamiltonian, determined by the current guess for the renormalized coupling coefficients $\tilde{\boldsymbol{c}}'$. When $\tilde{\boldsymbol{c}}' = \boldsymbol{c}'$, we obtain

$$\int P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) d\boldsymbol{x}_V = \frac{\exp\left(-\tilde{W}(\boldsymbol{x}_U)\right)}{Z_U}.$$

Since the group variables $\boldsymbol{x}_U$ are independent of each other given $\boldsymbol{x}_V$, it follows that

$$\frac{1}{Z_U} \int \Phi_i(\boldsymbol{x}_U) P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) \exp\left(-\tilde{W}(\boldsymbol{x}_U)\right) d\boldsymbol{x}_U = \Phi_i(\boldsymbol{x}_U).$$

For $\Phi_i(\boldsymbol{x}_U)$ being even polynomials of the spin variables averaging, this quantity over $\boldsymbol{x}_U$ yields

$$\langle \Phi_i(\boldsymbol{x}_U) \rangle = 0 \qquad \text{and} \qquad \langle \Phi_i(\boldsymbol{x}_U) \Phi_j(\boldsymbol{x}_U) \rangle = 2^{|U|} \delta_{ij},$$

where $|U|$ is the number of lattice sites on the renormalized lattice U. The latter equation shows that the lattice polynomials are in fact orthogonal

with respect to the uniform inner product. When $\tilde{\boldsymbol{c}}' \neq \boldsymbol{c}'$, in the vicinity of $\boldsymbol{c}'$ one may use a linear expansion and obtain

$$\langle \Phi_i(\boldsymbol{x}_U) \rangle = \sum_j \langle \Phi_i(\boldsymbol{x}_U) \Phi_j(\boldsymbol{x}_U) \rangle (\tilde{c}'_j - c'_j),$$

a linear equation allowing one to solve for $\boldsymbol{c}'$ by matrix inversion and improve the guess $\tilde{\boldsymbol{c}}'$. Repeated iteration quickly leads to the a good approximation of the sought coefficient values. In comparison with the method with that of Swendsen (1984a), this method is less complicated but leads to similar coupling coefficient values.

Binney et al. (1992) describe a variation on the method of computing the renormalized coefficients that requires computing the exact marginal. They define the marginal Hamiltonian as

$$W(\boldsymbol{x}_U) = \ln \left[\int P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) d\boldsymbol{x}_V \right]$$

and directly project $W(\boldsymbol{x}_U)$ onto a basis of interactions $\{\Phi_i\}$, requiring one to compute

$$c_i = \frac{1}{2^{|U|}} \int \Phi_i(\boldsymbol{x}_U) \ln \left[\int P(\boldsymbol{x}_U \mid \boldsymbol{x}_V) P(\boldsymbol{x}_V) d\boldsymbol{x}_V \right] d\boldsymbol{x}_U$$

exactly in order to compute the renormalized coefficients. Their method works because the basis functions used (polynomials of the lattice variables) are orthogonal under the uniform inner product. However, Binney et al. (1992) commits an error by applying it to a 4×4 lattice renormalized to a 2×2 lattice and not adjusting the resulting coefficients for double-counting due to periodicity. A corrected version of this approach is described and used to study parameter flow in Chapter 7.

Further developments are due to Brandt and Ron (2001a,b), who computed a representation of the renormalized Hamiltonian $W(\boldsymbol{x}_U)$ in a different, but related way. Their approach was to build a table $P_+(\boldsymbol{x}_{N(u)})$ that assigns to each possible state of the neighborhood $\boldsymbol{x}_{N(u)}$ of the variable x_u the probability that $x_u = 1$. These probabilities are obtained by sampling $\boldsymbol{x}_V$ using the original probability density and subsequently sampling $\boldsymbol{x}_U$ using the conditional probability, as in Ma (1976), Swendsen (1979a,b, 1984a,b,c) and Gupta and Cordery (1984). The table $P_+(\boldsymbol{x}_{N(u)})$ is computed by counting the fraction of times the neighborhood was in the state $\boldsymbol{x}_{N(u)}$ and $x_u = 1$. Brandt and Ron (2001a,b) curb the rapid growth of the table using lattice symmetries and by considering only neighbor-

hood states whose P_+ value varies significantly from $1/2$; however, this results in a complicated data structure.

Although the method of Brandt and Ron (2001a,b) at the outset appears very different from that of the remaining authors, it is in fact closely related to the method discussed by Binney et al. (1992). Consider the fact that the tabulated representation may be seen as a series expansion

$$P(x_u = 1 \mid \boldsymbol{x}_{N(u)}) = \sum_i c_i \Phi_i(\boldsymbol{x}_{N(u)}) = \sum_i c_i \delta(\boldsymbol{x}_{N(u)} - \boldsymbol{x}^i_{N(u)}),$$

where the basis functions are the discrete delta functions, defined as

$$\delta(x - y) = \begin{cases} 1 & \text{for } x = y \\ 0 & \text{otherwise.} \end{cases}$$

The coefficient c_i is the table entry for $P_+(\boldsymbol{x}^i_{N(u)})$ corresponding to the state $x^i_{N(u)}$. The apparent absence of a linear projection or solution of a linear system is due to the fact that the δ basis functions are orthonormal under every inner product, thus only need a normalizing constant: the number of times the state $\boldsymbol{x}^i_{N(u)}$ was observed. However, the cost paid by using this approach is the inability to handle continuous variables: an approximate method using a discretization of the continuous variables was pursued by Shmulyian (1999) and mentioned by Brandt and Ron (2001b), but was not continued.

Finally, Chorin (2003, 2008) and Okunev (2005) describe a novel method of computing the renormalized coefficients using an approximate projection related to that described by Binney et al. (1992); however, their method is only applicable to decimation because of the required assumption that $U \subseteq V$. After defining the marginal Hamiltonian, Chorin (2003, 2008) and Okunev (2005) differentiate it with respect to x_u for $u \in U \subseteq V$, obtaining the fast marginalization equation

$$\frac{\partial W(\boldsymbol{x}_U)}{\partial x_u} = \mathbb{E}\left[\frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} \,\middle|\, \boldsymbol{x}_U \right].$$

Projecting the partial derivative $\partial W(\boldsymbol{x}_U)/\partial x_u$ onto a basis ϕ using least squares produces a linear system with

$$A_{ij} = \mathbb{E}\left[\phi_i \phi_j\right] \quad \text{and} \quad b_i = \mathbb{E}\left[\phi_i \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u}\right],$$

which satisfies $A\boldsymbol{c} = \boldsymbol{b}$. The method is applied to the Ising model by letting the variable x_u be continuous and taking a derivative of the smooth

Hamiltonian; the discrete renormalized Hamiltonian is recovered by integration of the approximate partial derivative. However, the references contain two major errors. The partial derivative $\partial W(\boldsymbol{x}_U)/\partial x_u$ is explicitly required by Chorin (2008) to be a constant function of x_u, while one may in fact show that it is highly non-linear even in the simplest, one-dimensional Ising model. Additionally, the expectation values used to construct the linear system ignore the entire interior of the interval $x_u \in [-1, 1]$; as a result, the non-linearity of the derivative is not captured.

Okunev (2005) proposed an exponential projection method that, although similar in spirit to that of Chorin (2003, 2008), was applicable to discrete variables of the Ising model. He noticed that the function

$$\exp\left(\Delta_u W(\boldsymbol{x}_{U\setminus u}; x_u, -x_u)\right) = \exp\left(W(-x_u, \boldsymbol{x}_{U\setminus u}) - W(x_u, \boldsymbol{x}_{U\setminus u})\right)$$

may be written as a conditional expectation given $\boldsymbol{x}_U$, allowing for the use of the remainder of the fast marginalization methodology in unchanged form. Because this method does not require making x_u continuous, it does not suffer from the difficulties faced by the method used by Chorin (2003, 2008). Unfortunately, the exponential projection is much more complicated in practice, because it requires one to approximate an exponential, which has to be positive. Therefore, finding an approximation using least squares or other methods is extremely challenging.

1.2 MULTI-SCALE SAMPLING

Although renormalization was long used to compute coefficients of the marginal probability distributions, the goal of the computation was to obtain quantities of physical interest — such as critical exponents — via scaling arguments. The possibility of using the marginal probability distributions to sample from the original statistical model was not pursued. One of the first methods of attempting to use a multi-scale approach to sampling lattice models was constructed by Goodman and Sokal (1989). They propose a Monte Carlo method formulated in the language of the multigrid method (Briggs, Henson, and McCormick, 2000), where the coarse lattices are composed of blocks of the original, fine variables. In their method the variables are then updated as blocks, performing large-scale moves on the very coarse lattices and finer-scaled moves on lattices closer to the original. Although their method does indeed improve the convergence of the Markov Chain Monte Carlo (MCMC) method, it requires that the magnitude of the proposed changes be small on the coarse lattice in order to

obtain reasonable acceptance probabilities: the proposal probability distribution is in fact unrelated to any marginal distribution. Although this requirement does not preclude application of their method to discrete systems, Goodman and Sokal suggest that such applications are impractical.

The next approach to sampling lattices is that of Brandt and Ron (2001a,b), who use their P_+ table representation of the marginal probability density to sample states of the two-dimensional Ising model using the top-down approach. They first compute the P_+ tables numerically by sampling the original lattice $V_0 = V$ and computing the P_+^1 table describing the marginal probability density $P(\boldsymbol{x}_{V_1})$, where V_1 is a lattice of majority rule block variables with block size of 2×2. Through a finite recursion, Brandt and Ron (2001a,b) construct a sequence of lattices V_i and tables P_+^i by sampling the lattice V_i using P_+^i and computing the table P_+^{i+1}. Their approach has the consequence that the coefficients slowly drift away from the true values, because the subsequent tables are computed using the already approximate probability distributions; however, their tests suggest that the errors may be small.

Following the determination of the P_+ tables, Brandt and Ron (2001a,b) sample in the reverse direction: first, the coarsest (top) lattice V_m is randomly initialized and sampled using MCMC with the help of the table P_+^m, producing a state $\boldsymbol{x}_{V_m}$. Assuming the lattice V_{i+1} has already been sampled, the lattice V_i is sampled by assigning the spins $\boldsymbol{x}_{V_i}$ random values that are consistent with the state $\boldsymbol{x}_{V_{i+1}}$, i.e., such that the coarsening rule $P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i}) > 0$. Subsequently, MCMC is employed to sample from the joint distribution $P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}})$ while holding $\boldsymbol{x}_{V_{i+1}}$ constant, thus having the effect of sampling from the conditional probability $P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i+1}})$: see Chapter 6 for an in-depth discussion. Because the table P_+^{i+1} is only an approximation of the true marginal of P_+^i, Brandt and Ron (2001a,b) employ a *post-relaxation* step where the value $\boldsymbol{x}_{V_{i+1}}$ is discarded and p iterations of unconstrained MCMC on the lattice V_i are employed to bring the probability density of $\boldsymbol{x}_{V_i}$ closer to the target. They show that very few post-relaxation sweeps are necessary.

While the sampling method of Brandt and Ron (2001a,b) is shown to work well with the two-dimensional Ising model, the validation comes from observed quantities such as the two-point correlation function between spins located at a distance $\sqrt{2}$ on the lattice, which is compared to values computed using long simulations using the Wolff cluster algorithm (Wolff, 1989). Therefore, this sampling method is unable to provide information about sample quality for previously unstudied statistical models, cf. Chapter 6.

Ron, Swendsen, and Brandt (2002) describe a variation of their sampling method that is limited to sampling from the critical point of the renormalization transformation. Let the original model with coupling coefficients $\boldsymbol{c}$ undergo a series of renormalization steps. If the repeated application of the renormalization transformation R maps the initial coupling coefficients $\boldsymbol{c}$ to the fixed point $\boldsymbol{c}^*$, Ron, Swendsen, and Brandt (2002) propose to reverse this operation for sampling. They compute a large basis approximation of the fixed point $\boldsymbol{c}^*$ and store it as the table P_+^*. Subsequently, they sample a sequence of lattices $V_m, V_{m-1}, \dots, V_0$ in a top-down fashion using the same table P_+^* on each lattice. The rationale behind such a move is that if repeated marginalization brings the coefficients towards $\boldsymbol{c}^*$ and the P_+ table toward P_+^*, sampling using P_+^* at each lattice implies that a virtual original lattice was extremely large, to the point that the intermediate coefficients between $\boldsymbol{c}$ and $\boldsymbol{c}^*$ were eliminated. This way one may generate high quality states for very large lattices, assuming that the fixed point coefficients can be approximated accurately and that the coefficients $\boldsymbol{c}^*$ do not depend significantly on the size of the original lattice used to compute them. However, since renormalization flows from multiple points converge onto the fixed point $\boldsymbol{c}^*$, it is unclear what is the relationship between the samples obtained using this method and the samples of the original model at criticality.

Weare (2007) constructs an MCMC method utilizing multiple lattices, but without a top-down approach. Starting with the fine lattice V_0 and probability distribution function $P(\boldsymbol{x}_{V_0})$, he constructs a sequence of subsequently coarser lattices $V_0 \supset V_1 \supset V_2 \supset \dots \supset V_m$ with probability distributions functions $P(\boldsymbol{x}_{V_i})$, $0 < i \leq m$, defined as the approximate marginals

$$P(\boldsymbol{x}_{V_i}) \approx \int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i}.$$

Each lattice hosts a Markov chain Y_i^n with transition probability $T_i(\boldsymbol{x}_{V_i} \to \boldsymbol{y}_{V_i})$ that leaves $P(\boldsymbol{x}_{V_i})$ invariant. The Markov chains on the coarser lattices equilibrate much more quickly than those on the fine lattices; therefore, to speed-up the convergence of the fine chains, Weare introduces a swap move where the variables $\boldsymbol{x}_{V_i}$ are swapped with the corresponding variables in $\boldsymbol{x}_{V_{i+1}}$. Since the invariant probability distribution $P(\boldsymbol{x}_{V_{i+1}})$ of the chain Y_{i+1}^n is not an exact marginal of $P(\boldsymbol{x}_{V_i})$, unconditional swaps would not preserve the invariant distributions of the chains. In order to leave these distributions invariant, Weare introduces a swap move acceptance probability that corrects for the approximate nature of the probabilities $P(\boldsymbol{x}_{V_i})$. He then continues

to simplify the calculation of the acceptance probability to eliminate the need of computing $P(\boldsymbol{x}_{V_i})$ entirely, and applies the resulting method to the problems of bridge path sampling and non-linear filtering.

Lastly, Chorin (2008) and Okunev (2005) use their approximate renormalized coupling coefficients to sample spins of the two-dimensional Ising model and the three dimensional Edwards-Anderson spin glass using a top-down approach. Their Chainless Monte Carlo (ChMC) method does not use the MCMC approach at all; instead, the samples are constructed using a conditional sampling process: the variables are determined individually using a probability conditional on the already sampled spins. In contrast to Brandt and Ron (2001a,b), the ChMC produces a proposal density of each generated sample and thus the approximate nature of the coefficients may be corrected using importance sampling by computing weights. Furthermore, it can be shown that the ChMC sampling approach is equivalent to Sequential Importance Sampling (SIS) (cf. Sections 5.1.1 and 5.2.1), paving the way to the use of advanced particle filtering techniques for improving the quality of the generated samples.

1.3 GRAPHICAL MODELS

Graphical models provide a natural framework for the marginalization of lattice models (Airoldi, 2007; Koller and Friedman, 2009). The information encoded by the graph provides a natural way to describe sparsity present in many probability distribution functions, making it possible to optimally compute exact marginals. For an introduction to graphical models and their typical applications consult Jordan (2004), Koller and Friedman (2009), and Wainwright and Jordan (2008).

We first define graphical models. The conditional independence of random variables is a crucial part of this definition.

Definition (Conditional independence). *The random variables x_u and x_v defined on a lattice of variables V, $u, v \in V$, are said to be* conditionally independent *given all remaining variables $\boldsymbol{x}_{V\setminus\{u,v\}}$, denoted as $x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V\setminus\{u,v\}}$, if and only if*

$$P\left(x_i, x_j \mid \boldsymbol{x}_{V\setminus\{u,v\}}\right) = P\left(x_i \mid \boldsymbol{x}_{V\setminus\{u,v\}}\right) P\left(x_j \mid \boldsymbol{x}_{V\setminus\{u,v\}}\right),$$

i.e., the joint conditional probability distribution given all other variables factors into two functions dependent on x_u and x_v, respectively. If random variables are not conditionally independent they are said to be dependent.

Intuitively, the above definition captures the fact that variables interact directly only with a limited set of other variables. Similar behavior is

present in other areas of applied mathematics. A natural example is the heat equation $\nabla^2 u(\boldsymbol{x}) = 0$, where knowing the values of the solution $u(\boldsymbol{x})$ on a sphere $\partial B(\boldsymbol{x}_0, r)$ around a certain point $\boldsymbol{x}_0$ uniquely determines $u(\boldsymbol{x}_0)$ (Evans, 1998). In case of the heat equation, the points on the sphere separate $\boldsymbol{x}_0$ from the remainder of the space. Similar separation result can be found in the case of graphical models, where the interactions between variables occur through the graph rather than through space, leading to the following definition.

Definition (Graphical model). *The probability distribution $P(\boldsymbol{x}_V)$ is said to induce a dependency graph $G = (V, E)$. Each variable x_u is assigned a node u in the set of nodes V. Two nodes $u, v \in V$ are* not *connected by an edge if and only if the variables x_u and x_v are conditionally independent, i.e., $(u, v) \notin E$ iff $x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V\setminus\{u,v\}}$. The probability distribution together with its induced undirected dependency graph is called an undirected graphical model.*

Using the Partial Differential Equation (PDE) analogy again, the dependency graph can be thought of as the graph of the matrix discretization of a differential operator. For example, the heat equation has operator $\mathcal{L} = \nabla^2$, whose classical discretization in two dimensions

$$\nabla^2 u(x, y) = \frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} + \mathcal{O}(h^2)$$

has non-zero matrix elements at positions corresponding to the couplings between the variable u_{ij} and its neighbors on the Cartesian mesh. The graph of the resulting matrix is incidentally the same as that of the Ising model.

With the above definition, every probability distribution is also a graphical model. However, the machinery of graphical model theory is only useful when the resulting dependency graph is sparse. Many physically motivated statistical models lead to sparse dependency graphs in the same way that many PDE can be discretized using sparse matrices.

The graphical models will be used throughout the present thesis as the language used to describe our methodology. Graphical models will be used mainly in Chapter 3 to show how marginalization of variables alters the conditional independence relations between the remaining variables, and in Chapter 4 to motivate the choice of consistent basis functions. Finally, the dependency graph will define the order of sampling variables discussed in Chapter 5.

1.4 RELATED AREAS

There are several areas of current interest that are tangentially related to the concepts or approaches used within the present thesis. The renormalization of a graphical model is related to the concept of deep architectures used in the computer learning community, where a seemingly complex probability distribution is described using a sparse multi-level structure involving hidden variables (Poon and Domingos, 2011). The construction of the acyclic Monte Carlo method, especially using the framework for general coarsening rules, can indeed be seen as adding auxiliary hidden variables in order to obtain a new representation for the probability distribution function.

Because of the similarity between the dependency graph induced by the probability distribution $P(\boldsymbol{x}_V)$ and the graph of a matrix A, the marginalization of a probability distribution has links to the process of Gaussian elimination and LU decomposition (Demmel, 1997). As such, the developments in algorithms for finding sparsity-preserving variable orderings are of natural interest (Davis et al., 2004a,b).

Finally, the acyclic Monte Carlo attempts to construct an acyclic representation of a given probability distribution function $P(\boldsymbol{x}_V)$, which allows for efficient sampling. Such approximate representations are known in the fields of genetics and automated learning as Conditional Sampling Distributions (CSDs) or Products of Approximate Conditionals (PACs); the standard references in the field are Chow and Liu (1968) and Paul and Song (2010).

1.5 STRUCTURE OF THE PRESENT THESIS

The methodology described within this thesis is named acyclic Monte Carlo, encompassing both the method for coarsening a graphical model, the computation of approximate renormalized coupling coefficients, and the subsequent sampling techniques. The fact that our method transforms a graphical model with circular dependencies between variables into an acyclic model is at the intersection of these methods and makes them efficient. Therefore, it is only fitting that the methodology herein described bear that name.

The present thesis is structured in the following way. We begin with the description of the prior work of Chorin (2003, 2008) and Okunev (2005), discuss a straightforward method for computing the renormalized coefficients, and use them for sampling the original model in Chapter 2. While this straightforward sampler is not intended for practical use, it serves

the purpose of illustrating the major elements of the methodology without unnecessary complexity.

In Part I we present the contributions of this thesis. Chapter 3 opens the discussion, describing renormalization methodology using the framework of graphical models. We detail the transformation of the undirected dependency graph induced by a probability distribution into an approximate directed acyclic graph and present related algorithms. The discussion of graphical methods is done separately from the computation of the resulting coefficients.

The computation of renormalized coupling coefficients forms Chapters 4 and 6, where we describe the fast marginalization algorithm and its generalized version, respectively. Chapter 4 is concerned with the computation of approximate renormalized coefficients describing a marginal probability distribution $P(\boldsymbol{x}_U)$, where $U \subseteq V$. Therefore, it does not depend on the fact that the set U forms a part of a hierarchy of increasingly coarse lattices; however, it is intended that U be thought of as one of the lattices V_i described in Chapter 3.

The sampling methods using the acyclic form of the probability distribution are discussed in Chapter 5. We discuss the sequential importance sampler used by Chorin (2008) and Okunev (2005) and improve upon it, using techniques from particle filtering. We touch on the topic of using the acyclic form to construct MCMC sampling schemes. Chapter 6 discusses the sampling techniques compatible with arbitrary coarsening rules.

Part II discusses the results obtained using the acyclic Monte Carlo. In Chapter 7 we describe the application of the fast marginalization algorithm and its generalized version to calculate the parameter flow of the two-dimensional Ising model, showing the lack of a critical point in case of decimation. Finally, Chapter 8 benchmarks the performance of the acyclic Monte Carlo on the two-dimensional Ising model.

2

A SIMPLE MARKOV FIELD SAMPLER

The method of sampling Markov fields described in this thesis is composed of multiple parts. In order to bring together the seemingly disconnected components, we present here a description of a simplified version of the more general sampler discussed in Chapters 3 and 5. This sampler contains all the parts required by the advanced method and is based on the earlier work of Okunev (2005) and Chorin (2003, 2008), with several improvements. As a pedagogical example, we will apply the simple sampler to the Ising model on a Cartesian lattice in one and two dimensions (Ising, 1925) .

The current chapter is organized as follows. For both the one- and two-dimensional Ising model, we first describe the components of the algorithm: (i) coarsening of the lattice, (ii) computation of marginal probability densities and (iii) a sampling scheme using the products of the prior two parts. Following the description is our analysis and commentary on the presented material, discussion of the choices made and interpretation of the method from different vantage points. We hope that the concise description of the algorithm will allow the reader to learn about the algorithms presented, while the subsequent analysis will provide the necessary discussion and lead the reader toward the main parts of the thesis contained in Chapters 3, 4 and 5.

We begin by describing decimation of the one-dimensional Ising model, following the standard approach of Kadanoff (1966, 2002) (cf. Binney et al., 1992; Migdal, 1975; Мигдал, 1975). We describe the decimation process in the language of graphical models, leading to the concept of *graph coarsening*. We motivate and analyze the choices made during Kadanoff renormalization, and finally use the analysis to hint at a possible generalization of the decimation algorithm to complex graphical models. The renormalization (graph coarsening) produces a ladder of increasingly coarse graphical models, eventually reducing the original Ising lattice to only one variable. We show that this ladder structure can be used to efficiently sample the Ising model and describe the resulting algorithm. Analyzing the sampling algorithm, we show that the ladder structure produces an alternative graphical model representation of the Ising model: while the original graphical model was *undirected* and contained cyclical dependencies, the ladder structure produces an equivalent *directed* and *acyclic* graphical

model reproducing the same probability density. This directed acyclic representation of the probability density is a Bayesian network and therefore can be sampled efficiently.

The simplified method consists of two steps, (i) the decimation of the original Cartesian lattice to a series of coarser lattices through Kadanoff renormalization (e.g. Binney et al., 1992; Kadanoff, 1966, 2002) and (ii) the sampling of the resulting hierarchy of lattices in the reverse, coarse-to-fine, direction. The decimation step attempts to construct a sequence of smaller Ising-type lattices by marginalizing (integrating out) a fraction of random variables at a time. These coarser lattices may be easier to sample due to their reduced size, albeit at a cost of computing their probability densities through marginalization. The sampling step uses the coarser lattices and the relationship between their respective probability densities. Starting at the coarsest lattice, which is assumed to be easy to sample using an alternative method, such as MCMC or even direct sampling (Liu, 2001) , the lattices are sampled iteratively: the immediately coarser lattice is sampled using the conditional probabilities $P(\text{fine} \mid \text{coarse})$, which can be obtained efficiently in a number of ways. We will first discuss the Ising model in one dimension, where the above procedure may be performed exactly, and use it to motivate the developments of Chapter 3.

Direct sampling means computing the probabilities of all possible states and choosing one of them at random.

2.1 ISING MODEL IN ONE DIMENSION

We begin with the Ising model defined on a periodic chain of length $n = 2^m$, with $m \geq 2$ (see Figure 2.1). The probability distribution defined over the spins $\boldsymbol{x}_V = (x_1, x_2, \dots, x_n)$ is

$$P(\boldsymbol{x}_V) = \frac{1}{Z(\mu)} \exp\left[\frac{\mu}{2} \sum_{i=1}^{n} x_i \left(x_{i-1} + x_{i+1}\right)\right],$$

where $\mu = J/T$ is the coupling strength and $Z(\mu)$ the partition function. The insight of Kadanoff (1966, 2002) was to notice that changing the value of a spin x_i, while keeping all the other spins fixed, will have a very limited influence on the probabilities of the remaining spins. In fact, the conditional probabilities of only two spins will be changed, precisely the nearest neighbors of x_i in the lattice: x_{i-1} and x_{i+1}. These spins are said to be *dependent* on x_i, while all other spins are *conditionally independent* of x_i given x_{i-1} and x_{i+1}. The conditional independence gives rise to the graphical structure depicted on Figure 2.1.

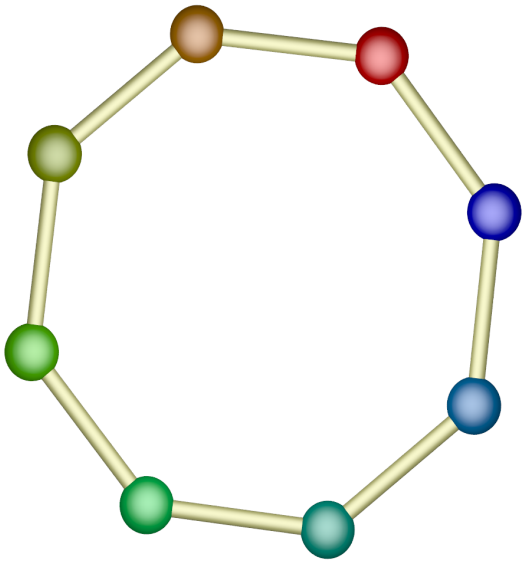


Figure 2.1: Periodic lattice of a one dimensional Ising model of size 8. The nodes are color coded using the index of the node: node 1 is deep blue, node 5 is green, while node 8 is red.

2.1.1 *Coarsening*

Kadanoff renormalization or decimation attempts to eliminate, or decimate, random variables to produce a coarser random model. To describe the renormalization process we introduce the following notation. The components of the random vector $\boldsymbol{x}_V$ correspond to nodes of the graph $G = (V, E)$; thus for every node $u \in V$ we identify x_u as the corresponding random variable. Given a subset $U \subset V$, $\boldsymbol{x}_U$ is a vector of dimension $|U|$ made of components x_u of the original vector $\boldsymbol{x}_V$ for all $u \in U$. For simplicity of notation we will frequently write $U \setminus u$ to mean $U \setminus \{u\}$. To avoid clutter, the probability distribution over the variables $P(\boldsymbol{x}_U)$ uses the same symbol as that of $P(\boldsymbol{x}_V)$; therefore, the precise distribution is identified by the variables it depends on.

Following the standard approach (Binney et al., 1992; Kadanoff, 1966, 2002; Migdal, 1975; Мигдал, 1975), we eliminate half of the variables by

marginalizing them. Let $V_0 = V$ and decompose V_0 into two into two subsets,

$$V_1 = \{1, 3, 5, \dots, n-1\} \quad \text{and} \quad V_0 \setminus V_1 = \{2, 4, 6, \dots, n\},$$

which separate the variables in $\boldsymbol{x}_V$ into two non-overlapping parts

$$\boldsymbol{x}_{V_1} = \{x_1, x_3, x_5, \dots, x_{n-1}\} \quad \text{and} \quad \boldsymbol{x}_{V_0 \setminus V_1} = \{x_2, x_4, x_6, \dots, x_n\}.$$

The set V_1 will represent the renormalized, or coarse, variables. The marginal probability of $\boldsymbol{x}_{V_1}$ is the integral of the joint probability $P(\boldsymbol{x}_{V_1}, \boldsymbol{x}_{V_0 \setminus V_1})$ over $\boldsymbol{x}_{V_0 \setminus V_1}$,

An integral is with respect to a discrete measure, therefore the integral becomes a sum.

$$P(\boldsymbol{x}_{V_1}) = \int P(\boldsymbol{x}_{V_1}, \boldsymbol{x}_{V_0 \setminus V_1}) d\boldsymbol{x}_{V_0 \setminus V_1}.$$

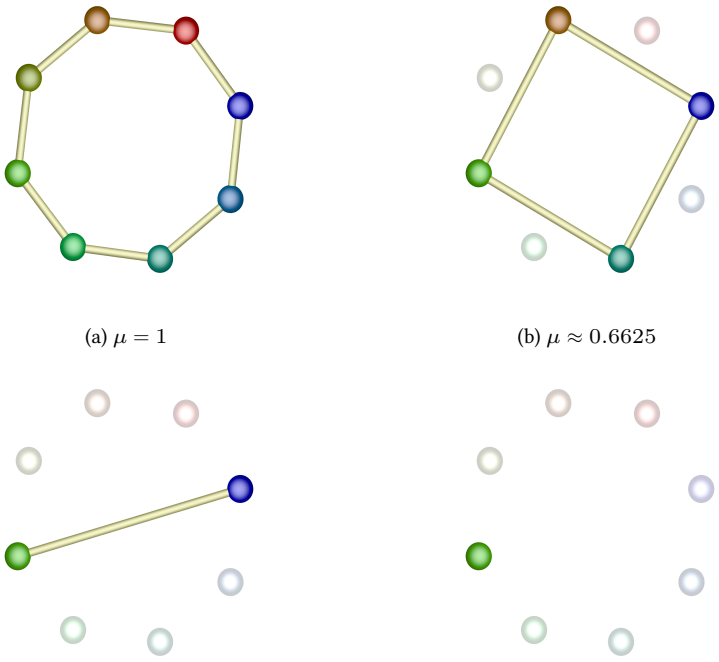
The resulting probability density is defined for a half of the variables of the original Ising model, yet a straightforward calculation shows that $P(\boldsymbol{x}_{V_1})$ can be written in the same form as the original Ising model (see Appendix A):

$$P(\boldsymbol{x}_{V_1}) = \frac{1}{\hat{Z}} \exp\left[\frac{\mu_1}{2} \sum_{i=1}^{n/2} x_{2i-1} \left(x_{2i-3} + x_{2i+1}\right)\right], \tag{2.1}$$

where $\mu_1 = {}^1/_2 \ln \cosh(2\mu_0)$ (see Example 2.2). This exact result shows the effect of eliminating a variable on the graphical structure of the problem. Indeed, if we look at the graph $G_1 = (V_1, E_1)$ induced by $P(\boldsymbol{x}_{V_1})$ we will again find a circular chain, i.e., the structure of the graph has not changed. The effect of marginalizing (integrating out) a variable can be seen purely in terms of graphs (Section 3.2.1 or Chapter 9 of Koller and Friedman (2009)): eliminating a node u in a graph requires connecting by edges all of the neighboring nodes $N(u)$.

This is precisely the reason why renormalization can be performed exactly in case of one dimensional Ising model but fails in two or more dimensions.

The hierarchy of graphical models constructed this way from the original Ising model is shown on Figure 2.2. The top-most lattice is composed of only one variable and therefore has a different graphical structure than the finer lattices; as such, the general formula shown on Equation 2.1 cannot be used to describe the probability distribution $P(\boldsymbol{x}_{V_m})$. Instead, we notice that $P(\boldsymbol{x}_{V_m})$ is the probability that a single variable in $\boldsymbol{x}_V$ is positive or negative. Using a symmetry argument it becomes clear that $P(\boldsymbol{x}_{V_m} = 1) = P(\boldsymbol{x}_{V_m} = -1) = {}^1/_2$, which completes the renormalization process.



(a) $\mu = 1$

(b) $\mu \approx 0.6625$

(c) $\mu \approx 0.3501$

(d) $\mu \approx 0.1137$

Figure 2.2: Hierarchy of graphical models produced by the Kadanoff-Midgal renormalization. The probability distribution of each model is of Ising-type, with $n_i = {}^{n}\!/_{2^i}$ nodes and inverse temperature $\mu_i = {}^{1}\!/_{2} \ln \cosh(2\mu_{i-1})$. The quoted numbers show approximately the decay of the inverse temperature with renormalization, starting from an arbitrary $\mu = 1$.

EXAMPLE 2.1. In the circular graph of Figure 2.1, each variable is connected to only two other variables; thus, removing node 2 connects the neighboring nodes, i.e., nodes 1 and 3, while removing node 4 connects nodes 3 and 5. Initially, node 3 was connected to nodes 2 and 4, but after eliminating them it is connected to nodes 1 and 5. Removing all even numbered nodes we obtain a graph with nodes 1, 3, 5, and 7 with edges $(1, 3)$, $(3, 5)$, $(5, 7)$ and $(7, 1)$. The resulting decimated graph is shown on Figure 2.3. ■

Since the result of a Kadanoff renormalization is again a graphical model, let us consider repeating the procedure. Let $M = (\boldsymbol{x}_V, P(\boldsymbol{x}_V), G = (V, E))$ be a graphical model. The Kadanoff

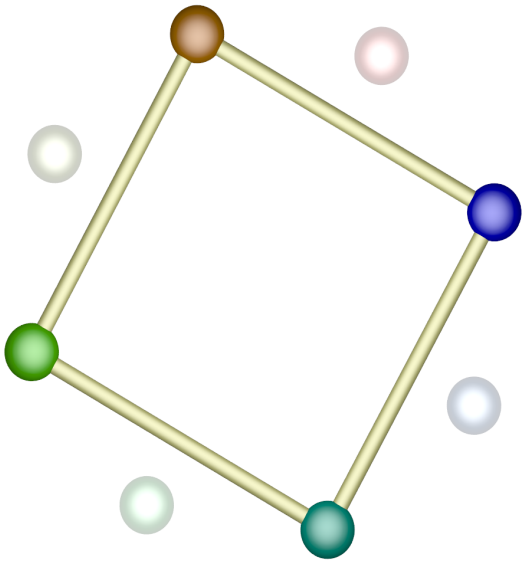


Figure 2.3: The graph induced by the probability density $P(\boldsymbol{x}_{V_1})$.

renormalization (cf. Migdal, 1975; Мигдал, 1975) can be though of as a mapping R

$$M' = R(M)$$

between the original graphical model and the renormalized model, specified using the process described above. Beginning with the original graphical model $M_0 = M$, we may define iteratively a hierarchy of graphical models $M_i = \left(\boldsymbol{x}_{V_i}, P(\boldsymbol{x}_{V_i}), G_i = (V_i, E_i)\right)$ through

$$M_{i+1} = R(M_i),$$

where the resulting model M_{i+1} has $n_{i+1} = {n_i}/{2} = {n}/{2^{i+1}}$ variables. Because the probability distribution can be described by a single parameter μ_i, we can think of the renormalization as a mapping between the original and renormalized coupling parameters,

$$\mu_{i+1} = R(\mu_i).$$

The plot of $R(\mu_i)$ is shown on Figure 2.4. Visual analysis of the curve shows that there exists only one fixed point of the mapping, i.e., the zero-coupling (infinite temperature) fixed point $\mu = 0$. Additionally, for $\mu > 0$ the renormalized value $R(\mu) < \mu$; therefore, the variables on coarse lattices become increasingly decorrelated. These two observations are precisely the reason for lack of a phase transition in the one-dimensional Ising model: at large distances the Ising spins become decorrelated, while at the phase transition correlation length grows infinitely large.

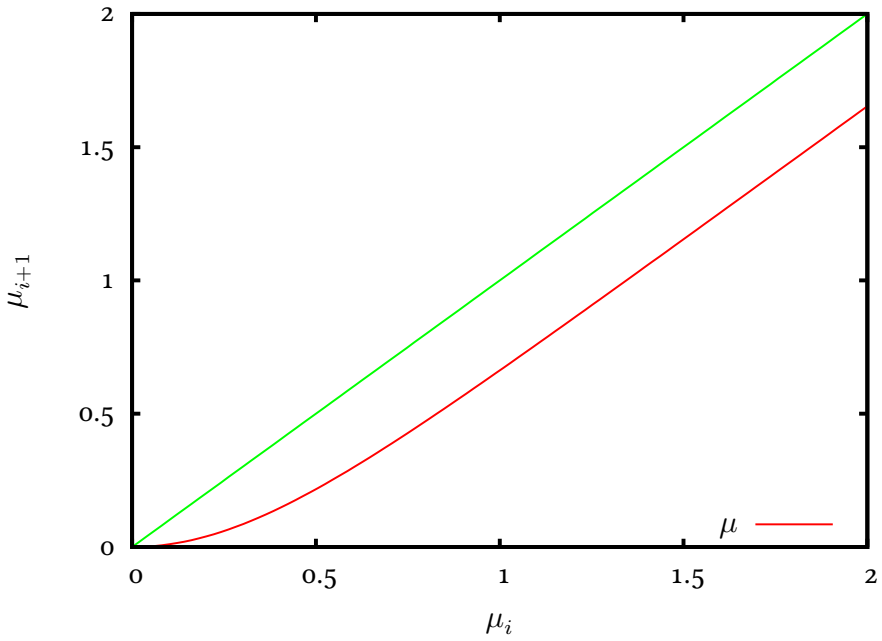


Figure 2.4: Mapping of the Ising model in one dimension.

2.1.2 *Sampling*

The sampling algorithm uses the ladder of lattices in the coarse-to-fine direction, opposite to that of the coarsening procedure. The sampling procedure will iterate, filling the variables one level at a time. At each iteration we will assume that variables $\boldsymbol{x}_{V_{i+1}}$ are known and those in $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ need to be sampled. Therefore, we will begin by handling the case of sampling the top lattice V_m with variables $\boldsymbol{x}_{V_m}$ separately and follow with a general coarse-to-fine iteration.

The top lattice V_m is composed of only one variable, thus must be treated separately in a special manner. Since both values $\boldsymbol{x}_{V_m} = 1$ and $\boldsymbol{x}_{V_m} = -1$ are equiprobable, we simply choose one at random. Having

Algorithm 2.1 Sampling a one-dimensional Ising model of of $n = 2^m$ spins using the coarsened lattice structure.

The algorithm generates a single state of random variables $\boldsymbol{x}_V$, given the lattices $G = G_0, G_1, \dots, G_m$ and the renormalized coupling constants $\mu_0, \mu_1, \dots, \mu_{m-1}$. The top lattice V_m has only one spin, therefore by symmetry both values are equally likely. The function SAMPLESPIN returns a random spin with the prescribed unnormalized probabilities.

procedure SAMPLEISING1D(m, G_i, μ_i)
  $\boldsymbol{x}_{V_m} \leftarrow$ SAMPLESPIN$(1, 1)$
  for $i = m - 1 \to 0$ **do**
    for all $u \in V_i \setminus V_{i+1}$ **do**
      $W_u \leftarrow \mu_i \left(x_{u+2^i} + x_{u-2^i}\right)$
      $x_u \leftarrow$ SAMPLESPIN$\left(\mathrm{e}^{-W_u}, \mathrm{e}^{W_u}\right)$
    end for
  end for
end procedure

function SAMPLESPIN(p_-, p_+)
  $p \leftarrow \mathrm{U}[0, 1]$
  if $p < \frac{p_-}{p_- + p_+}$ **then**
    return -1
  else
    return 1
  end if
end function

sampled the top lattice V_m, we may assume that the variables in $\boldsymbol{x}_{V_{i+1}}$ have been sampled. Because $V_{i+1} \subset V_i$, in order to complete the sampling of variables $\boldsymbol{x}_{V_i}$ we must sample the variables $\boldsymbol{x}_{V_i \setminus V_{i+1}}$. We make the important observation that for any $u, v \in V_i \setminus V_{i+1}$ the variables x_u and x_v are conditionally independent given $\boldsymbol{x}_{V_{i+1}}$,

$$x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V_{i+1}}.$$

Therefore, the variables to be sampled at level i are conditionally independent of each other and can be sampled individually. We contrast this with sampling the original lattice, which requires determining all variables simultaneously, typically using an iterative process such as the Markov Chain Monte Carlo.

To sample a spin $x_u \in V_i \setminus V_{i+1}$, we require the probabilities of both states, $x_u = -1$ and $x_u = 1$. Starting with the joint probability of $P(\boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{V_{i+1}}, \boldsymbol{x}_{V_i \setminus V_{i+1}})$, we obtain

$$\begin{aligned} P(\boldsymbol{x}_{V_{i+1}}, \boldsymbol{x}_{V_i \setminus V_{i+1}}) &= \frac{1}{Z_i} \exp\left[\frac{\mu_i}{2} \sum_{V_i} x_u \left(x_{u-2^i} + x_{u+2^i}\right)\right] \\ &= \frac{1}{Z_i} \prod_{V_i \setminus V_{i+1}} \exp\left[\mu_i x_u \left(x_{u-2^i} + x_{u+2^i}\right)\right] \\ &= \prod_{V_i \setminus V_{i+1}} P\left(x_u \mid \boldsymbol{x}_{V_{i+1}}\right), \end{aligned}$$

where Z_i is a product of normalization constants for the individual exponents. Defining

$$W_u = \mu_i \left(x_{u-2^i} + x_{u+2^i}\right),$$

each x_u for $u \in V_i \setminus V_{i+1}$ can be sampled using the probability distribution

$$\begin{aligned} P\left(x_u = -1\right) &= \frac{e^{-W_u}}{e^{-W_u} + e^{W_u}}, \\ P\left(x_u = 1\right) &= \frac{e^{W_u}}{e^{-W_u} + e^{W_u}}. \end{aligned}$$

The complete sampling procedure is described in Algorithm 2.1. We make the observation that we can write down the probability of the state $\boldsymbol{x}_V$ generated by Algorithm 2.1, as detailed in the Example 2.2 below.

Example 2.2. Consider the Ising model with $n = 8$ spins described in Example 2.1. Given the original coupling coefficient $\mu = \mu_0$ and the renormalized

$$\mu_1 = {}^1\!/_2 \ln\cosh 2\mu_0 \quad \text{and} \quad \mu_2 = {}^1\!/_2 \ln\cosh 2\mu_1,$$

the complete probability distribution may be written as

$$\begin{aligned} P(\boldsymbol{x}_V) = \frac{1}{2} &\times \frac{\mathrm{e}^{2\mu_2 x_1 x_5}}{\mathrm{e}^{-2\mu_2 x_1} + \mathrm{e}^{2\mu_2 x_1}} \\ &\times \frac{\mathrm{e}^{\mu_1 x_3 (x_1 + x_5)}}{\mathrm{e}^{-\mu_1 (x_1 + x_5)} + \mathrm{e}^{\mu_1 (x_1 + x_5)}} \times \frac{\mathrm{e}^{-\mu_1 x_7 (x_1 + x_5)}}{\mathrm{e}^{-\mu_1 (x_1 + x_5)} + \mathrm{e}^{\mu_1 (x_1 + x_5)}} \\ &\times \frac{\mathrm{e}^{\mu_0 x_2 (x_1 + x_3)}}{\mathrm{e}^{-\mu_0 (x_1 + x_3)} + \mathrm{e}^{\mu_0 (x_1 + x_3)}} \times \frac{\mathrm{e}^{-\mu_0 x_4 (x_3 + x_5)}}{\mathrm{e}^{-\mu_0 (x_3 + x_5)} + \mathrm{e}^{\mu_0 (x_3 + x_5)}} \end{aligned}$$

Algorithm 2.2 A classical greedy algorithm for computing the Maximum Independent Set (MIS).

function MISGREEDY($G = (V, E)$)
  $S \leftarrow \emptyset$
  $Q \leftarrow V$
  repeat
    $u \leftarrow \arg\max_u$ NODEORDER(u) for $u \in Q$
    $Q \leftarrow Q \setminus u$
    $S \leftarrow S \cup u$.
    for all v s.t. $(u, v) \in E$ **do**
      $Q \leftarrow Q \setminus v$
    end for
  until Q is empty
  return S
end function

$$\times \frac{\mathrm{e}^{\mu_0 x_6 (x_5 + x_7)}}{\mathrm{e}^{-\mu_0 (x_5 + x_7)} + \mathrm{e}^{\mu_0 (x_5 + x_7)}} \times \frac{\mathrm{e}^{-\mu_0 x_8 (x_1 + x_7)}}{\mathrm{e}^{-\mu_0 (x_1 + x_7)} + \mathrm{e}^{\mu_0 (x_1 + x_7)}}$$

Note the special form of the term with μ_2: since lattice 2 has only two spins, the coupling between them must be counted twice, hence the extra factor of two. ■

2.1.3 *Analysis*

Having described the standard renormalization of the one dimensional Ising model and the sampling Algorithm 2.1 that uses the renormalized lattices, we can analyze the renormalization (coarsening) and subsequent sampling to understand how the two work and interconnect.

2.1.3.1 *Graph coarsening*

We begin with the splitting of variables $\boldsymbol{x}_{V_i}$ into those that are to be kept on the coarse lattice $\boldsymbol{x}_{V_{i+1}}$ and the remainder of variables that are to be marginalized $\boldsymbol{x}_{V_i \setminus V_{i+1}}$. From the graphical point of view, the splitting decides which nodes of the graph $G_i = (V_i, E_i)$ are to be kept in the coarsened graph $G_{i+1} = (V_{i+1}, E_{i+1})$. The standard approach removes every other variable and can be motivated in multiple ways, however the reasoning we employ comes from the requirements of the subsequent sam-

Algorithm 2.3 An improved greedy algorithm for computing the Maximum Independent Set (MIS) due to Prof. Richard M. Karp (private communication). The function FRONTIER produces a set that does not belong to U, but forms the boundary (frontier) of U in the graph G.

function MISGREEDYKARP($G = (V, E)$)
  $S \leftarrow \emptyset$
  $Q \leftarrow V$
  repeat
    $u \leftarrow \arg\min_u \mid \textsc{Frontier}(G, S \cup u)$ for $u \subset Q$
    $Q \leftarrow Q \setminus u$
    $S \leftarrow S \cup u$.
    for all v s.t. $(u, v) \in E$ **do**
      $Q \leftarrow Q \setminus v$
    end for
  until Q is empty
  return S
end function

function FRONTIER($G = (V, E), U$)
  $S \leftarrow \emptyset$
  for all $v \in V$ **do**
    if $v \notin U$ and $(v, u) \in E$ s.t. $u \in U$ **then**
      $S \leftarrow S \cup v$
    end if
  end for
  return S
end function

pling step. As we saw, the spins $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ defined above are conditionally independent given the values of spins $\boldsymbol{x}_{V_{i+1}}$,

$$x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V_{i+1}} \quad \text{for all} \quad u, v \in V_i \setminus V_{i+1}, \tag{2.2}$$

which allows the sampling of the components of $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ independently of each other. Therefore, the variable partition must satisfy Equation 2.2. Graphically, this condition requires that no two nodes $u, v \in V_i \setminus V_{i+1}$ can be connected by an edge,

$$(u, v) \notin E_i \quad \text{for all} \quad u, v \in V_i \setminus V_{i+1}. \tag{2.3}$$

A set of nodes $U_{IS} \subseteq V_i$ in a graph $G_i = (V_i, E_i)$, such that no two nodes are connected by an edge, is called an Independent Set (IS); conversely, a set of nodes $U_{VC} \subseteq V_i$, such that the complement $V_i \setminus U_{VC}$ is an independent set, is known as a Vertex Cover (VC).

Using the independence requirement 2.2 we see that the set of variables that are to be marginalized, $\boldsymbol{x}_{V_i \setminus V_{i+1}}$, must form an Independent Set (IS) U_{IS}, while the remaining variables $\boldsymbol{x}_{V_{i+1}}$ that are to be kept for the coarse lattice must be the matching Vertex Cover (VC) $U_{VC} = V_i \setminus U_{IS}$. Because we wish to remove as many variables from $\boldsymbol{x}_{V_i}$ as possible, it is reasonable to require that U_{IS} be a Maximum Independent Set (MIS): an independent set of the largest possible size in the graph G_i. The complementary task of finding the smallest vertex cover is known as the Minimum Vertex Cover (MVC) problem. The graph associated with the one dimensional Ising model with even number of variables has a special structure: the graph is bipartite, i.e., the nodes V_i may be divided into two subsets such that there are no edges connecting the nodes within each subset. These two subsets are the exact solutions to the MIS and MVC problems and may be computed efficiently through graph coloring. Up to equivalence, the partition of the set V_i into $U_{VC} = V_{i+1}$ and $U_{IS} = V_i \setminus V_{i+1}$ is

$$U_{VC} = \{1, 3, 5, \dots, n-1\}, \qquad U_{IS} = \{2, 4, 6, \dots, n\}$$

for even n, the same as the partition we chose previously without explanation. Therefore, the division of variables may be generalized by requiring variables in $\boldsymbol{x}_{V_{i+1}}$ to form a Minimum Vertex Cover (MVC), while $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ form the complementary Maximum Independent Set (MIS). Finding a solution to the MVC or the MIS problem is prohibitively expensive; in fact, both graphical problems are part of the original twenty-one NP-complete problems compiled by Karp (1972), thus there are no known polynomial time algorithms for solving them. Instead, the requirement on $\boldsymbol{x}_{V_{i+1}}$ and $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ must be relaxed. We require that V_i be a Minimal Vertex Cover of G_i, i.e., a Vertex Cover that cannot be made smaller by removing a node, making it a locally optimal solution. Similarly, $V_i \setminus V_{i+1}$ be the complementary Maximal Independent Set, an Independent Set which cannot be made larger by adding a node. Both the Minimal Vertex Cover and Maximal Independent Set can be found quickly using greedy algorithms, such as Algorithm 2.2 or 2.3.

2.1.3.2 *Marginalization*

In the above example of the one-dimensional Ising model, the marginal probability distribution $P(\boldsymbol{x}_{V_{i+1}})$ was computed exactly from the definition

$$P(\boldsymbol{x}_{V_{i+1}}) = \int P(\boldsymbol{x}_{V_{i+1}}, \boldsymbol{x}_{V_i \setminus V_{i+1}}) d\boldsymbol{x}_{V_i \setminus V_{i+1}},$$

which in that case is analytically tractable (see Appendix A). While the computation of the value of the renormalized coupling coefficient μ_{i+1} is complicated, the form of the renormalized Probability Distribution Function (PDF) can be found easily using arguments from graphical model theory.

We return to the Example 2.2. Consider five consecutive nodes in the Ising model, x_1, x_2, x_3, x_4 and x_5, where we wish to marginalize x_2 and x_4. Per previous analysis, removing a node induces dependencies between all of its neighbors; therefore, removing node x_2 induces a dependency between x_1 and x_3, while removing x_4 induces another one between x_3 and x_5. The conditional probability of x_3 given all remaining spins,

$$P(x_3 \mid \boldsymbol{x}_{\{1,5,7\}}) = P(x_3 \mid x_1, x_5)$$

simplifies to a function of only x_3 and the two dependent variables x_1 and x_5. Since the components of $\boldsymbol{x}_V$ are binary and can only assume values of -1 or 1, the probability of x_3 can be expressed exactly using all the possible monomial terms involving x_1, x_3 and x_5:

$$\begin{aligned} P(x_3 \mid x_1, x_5) = \\ \exp\Big(a_1 + a_2 x_1 + a_3 x_3 + a_4 x_5 \\ + a_5 x_1 x_3 + a_6 x_1 x_5 + a_7 x_3 x_5 + a_8 x_1 x_3 x_5\Big) \Big/ Z. \end{aligned}$$

The terms a_1, $a_2 x_1$, $a_4 x_5$ and $a_6 x_1 x_5$ are constant in x_3 and thus can be arbitrarily set to zero. Due to the symmetries of the Ising lattice, the spins x_1 and x_5 are identical from the perspective of x_3 and we immediately obtain that $a_5 = a_7$. The resulting formula is then written as

$$P(x_3 \mid x_1, x_5) = \exp\Big(a_3 x_3 + a_5 x_3 (x_1 + x_5) + a_8 x_1 x_3 x_5\Big) \Big/ Z.$$

Due to the symmetry $P(\boldsymbol{x}_V) = P(-\boldsymbol{x}_V)$, only terms that are even in components of $\boldsymbol{x}_V$ may have non-zero coefficients, requiring $a_3 = a_8 = 0$. Therefore, the conditional probability of x_3 is written as

$$P(x_3 \mid x_1, x_5) = \exp\Big(\mu_1 x_3 \left(x_1 + x_5\right)\Big) \Big/ Z,$$

where μ_1 was substituted for a_5. The complete marginal probability $P(\boldsymbol{x}_{V_1})$ of Equation 2.1 can be deduced from the conditional probabilities of individual spins.

Unfortunately, in case of models of interest computing the marginal is impossible due to the complicated nature of the probability distribution and the size of the space to be integrated over. Instead, approximate methods must be used. In anticipation of the more advanced method described in later chapters, we describe a variant of the method of Chorin (2008) applied to the two-dimensional Ising model in Section 2.2.

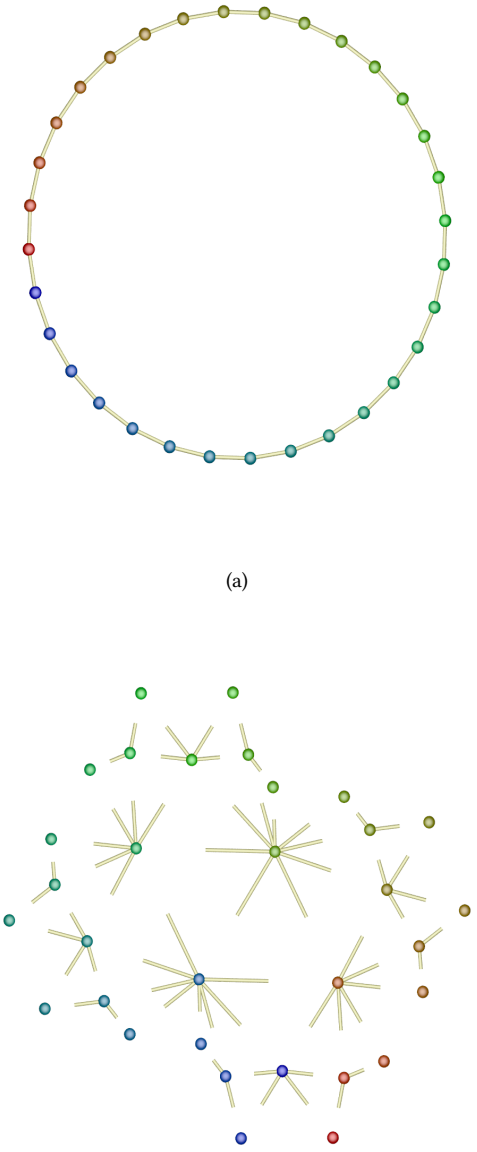
2.1.3.3 *Sampling*

The choice of variables that were to be marginalized was dictated by the goal of using the marginal probability densities for efficient sampling. Indeed, as Example 2.2 shows, the original probability $P(\boldsymbol{x}_V)$ can be rewritten in its *acyclic form*. Following the nomenclature of that example, the probability $P(\boldsymbol{x}_V)$ may be written as

$$\begin{aligned} P(\boldsymbol{x}_V) = P(x_1) &\times P(x_5 \mid x_1) \\ &\times P(x_3 \mid x_1, x_5) \times P(x_7 \mid x_1, x_5) \\ &\times P(x_2 \mid x_1, x_3) \times P(x_4 \mid x_3, x_5) \\ &\times P(x_6 \mid x_5, x_7) \times P(x_8 \mid x_1, x_7). \end{aligned}$$

The striking feature is that the original distribution required determining values of all the variables at once due to cyclical dependencies, yet the acyclic form does not: a value for x_1 can be determined directly by sampling from its marginal distribution, $x_1 \sim P(x_1)$. The remaining spins are then sampled from marginal probabilities conditional on the variables that have already been determined. In graphical terms, the graphs of the two representations of $P(\boldsymbol{x}_V)$ are different: the original is a circular chain, while the acyclic representation is a Directed Acyclic Graph (DAG). Visualization of these two graphs for a 32-spin Ising model is shown on Figure 2.5.

Every Directed Acyclic Graph (DAG) induces a partial order on the nodes of the graph, known as topological order: for every two nodes $u, v \in V$



(a)

(b)

Figure 2.5: Graphs of the (a) original graphical model and (b) its acyclic form.

we have $u \leq v$ if there exists a directed path from u to v. Nodes u, v for which a directed path either from u to v or from v to u, but not both, exist are called *comparable*, because one can write either $u \leq v$ or $u \geq v$;

otherwise, the nodes are *incomparable*. Sampling can be accomplished by choosing values for variables in the order dictated by the partial order. Following Example 2.2, the partial order is

$$x_1,\ x_5,\ \underline{x_3,\ x_7},\ \underline{x_2,\ x_4,\ x_6,\ x_8},$$

where the underlined parts of the order are composed of incomparable variables. The incomparability of these variables, e.g. x_3 and x_7, arises due to the fact that they are conditionally independent of each other; therefore, given that the variables preceding them in the ordering, i.e. x_1 and x_5, have been sampled, the variables x_3 and x_7 may be sampled in an arbitrary order. This situation arises because the factorization

$$P(x_3, x_7 \mid x_1, x_5) = P(x_3 \mid x_1, x_5) \times P(x_7 \mid x_1, x_5)$$

implies that the values for x_3 and x_7 can be determined independently of each other and in arbitrary order. As we will see in later chapters, this independence is a crucial part of the proposed method and decisive for its efficiency: the acyclic form of the probability distribution turns the simultaneous sampling of a large number of random variables into a sequential sampling of individual variables.

2.2 ISING MODEL IN TWO DIMENSIONS

The two-dimensional Ising model presents a bigger challenge than the one-dimensional case because exact renormalization cannot be performed. This statement is frequently mentioned while discussing Kadanoff renormalization, but should be quantified: exact renormalization of the two- and three-dimensional Ising model changes the form of its dependency graph, which becomes increasingly dense. The renormalized probability distributions are described by a rapidly increasing number of interactions and finding the required coupling strengths becomes analytically intractable. Computationally, though, it is possible to perform renormalization *approximately* or even exactly in certain cases.

In the following we will describe a modified version of the approximate algorithm of Chorin (2008) as applied to the two-dimensional Ising model.

Algorithm 2.4 Coarsening procedure for a two-dimensional Cartesian lattice graph.

The algorithm may be applied to arbitrary graphs as long as an appropriate metric $\rho(u, v)$ is supplied. For example, one may define the distance between two nodes to be the length of the shortest path in the graph. However, the *natural* choice of a metric should be used whenever possible.

procedure GRAPHCOARSENINGCARTESIAN2D($G_i = (V_i, E_i)$)
  $U_{IS} \leftarrow$ MISGREEDYKARP(G_i)
  $V_{i+1} \leftarrow V_i \setminus U_{IS}$
  $E_{i+1} \leftarrow \emptyset$
  for all $u \in V_{i+1}$ **do**
    $d \leftarrow \min_{u \neq v} \rho(u, v)$
    for all $v \in V_{i+1}$ s.t. $\rho(u, v) = q$ **do**
      $E_{i+1} \leftarrow E_{i+1} \bigcup (u, v)$
    end for
  end for
end procedure

2.2.1 *Coarsening*

We begin with the double-periodic Cartesian lattice $G_0 = G$ of size $n \times n$, restricting our consideration to cases where n is a power of two. Exact renormalization of such a graph, showed on Figure 2.6, requires that when a node $u \in V$ is removed from the graph all its neighbors $v \in N(u)$ be connected; unfortunately, in two and three dimensions this process rapidly produces a clique, a graph where every node is connected to every other node. At this point, exact computation of the marginal probability distribution becomes impossible for problems of interest.

Instead, we forgo exact coarsening and from the beginning assume that the dependency graphs and marginal probability distributions are approximate. In the crudest approximation, we will assume that the graphs G_i keep their structure and are all Cartesian lattices. Given a graph $G_i = (V_i, E_i)$, we construct $G_{i+1} = (V_{i+1}, E_{i+1})$ by removing nodes of V_i in a checkerboard-pattern and connecting the remaining nodes to their four nearest neighbors, thus preserving the lattice structure. Algorithm 2.4 describes the procedure in detail. We initially find an independent set $U_{IS} \subset V_i$ that should be removed from V_i, therefore, V_{i+1} becomes the vertex set separating the nodes of U_{IS}. Since G_i was a Cartesian lattice,

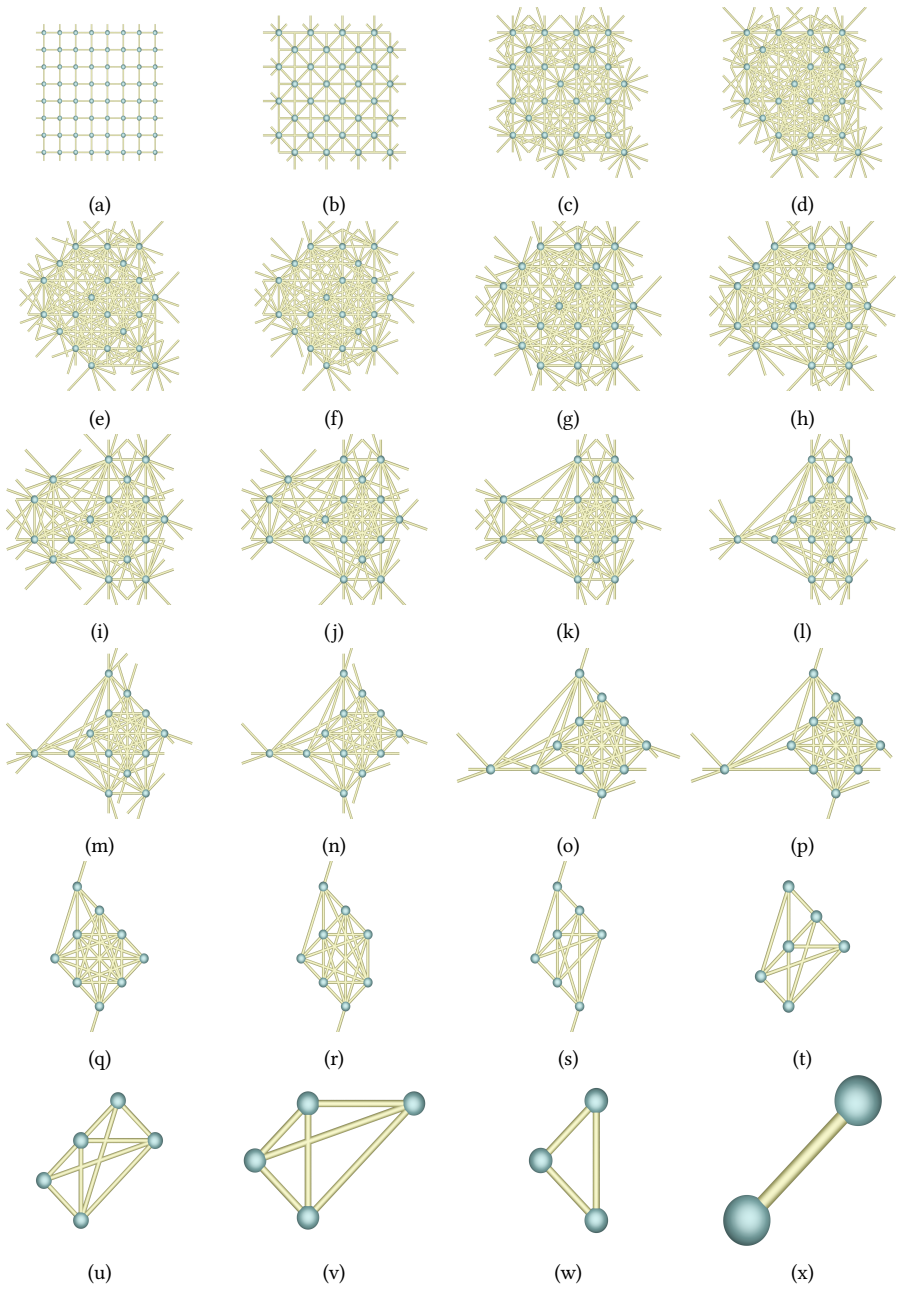


Figure 2.6: Exact coarsening of a 8×8 Cartesian lattice. At each coarsening the algorithm removes a Minimal Independent Set from the set of nodes.

both the independent set and the vertex cover form a checkerboard pattern. The coarse edge set E_{i+1} is constructed by connecting every node $u \in V_{i+1}$ to the four closest neighbors among other nodes in V_{i+1}. An example with a 8×8 initial lattice is shown on Figure 2.7.

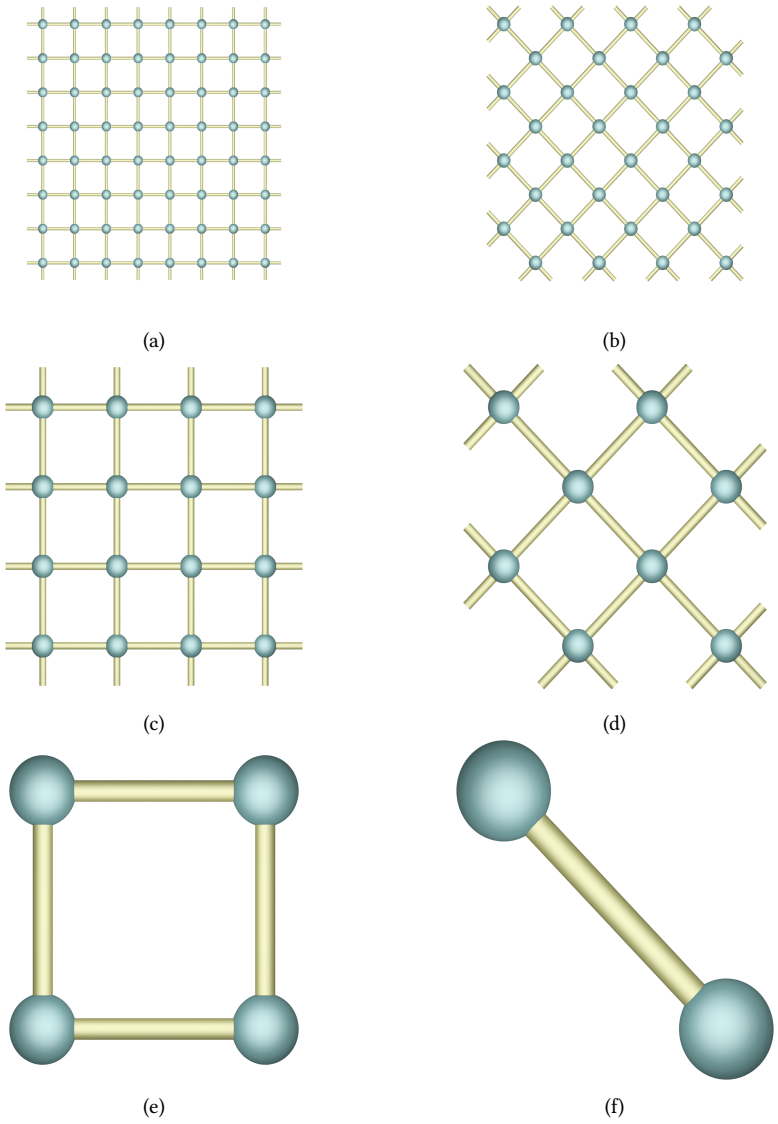


Figure 2.7: Approximate coarsening of a 8×8 Cartesian lattice computed using Algorithm 2.4 with $p = 2$ metric.

2.2.1.1 *Approximate marginalization*

Given the coarsened graphs

$$G_0 = (V_0, E_0), G_1 = (V_1, E_1), \dots, G_m = (V_m, E_m),$$

for $i = 0, 1, \dots, m$ the set of nodes V_i is a subset of the original set of nodes $V = V_0$. Defining $\boldsymbol{x}_{V \setminus V_i}$ as the set of all variables that belong to

the original set of variables V but are not present in the coarse set V_i, our aim is to obtain for each graph the marginal probability density $P(\boldsymbol{x}_{V_i})$ defined through

$$P(\boldsymbol{x}_{V_i}) = \int P(\boldsymbol{x}_{V_{i-1}}) d\boldsymbol{x}_{V_{i-1}\setminus V_i} = \int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V\setminus V_i}.$$

As already remarked, for any practical size it is impossible to compute the marginal probabilities directly from the definition, therefore we will use the fast marginalization algorithm of Chorin (2003, 2008) and Okunev (2005).

fast marginalization is based on the observation that for each $u \in V_i$ the logarithmic derivative

$$\begin{aligned}\frac{\partial \ln P(\boldsymbol{x}_{V_i})}{\partial x_u} &= \int \frac{\partial \ln P(\boldsymbol{x}_{V_i})}{\partial x_u} P(\boldsymbol{x}_V) d\boldsymbol{x}_{V\setminus V_i} \Big/ \int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V\setminus V_i} \\ &= \mathbb{E}\left[\frac{\partial \ln P(\boldsymbol{x}_V)}{\partial x_u} \,\middle|\, \boldsymbol{x}_{V_i} \right]\end{aligned}$$

is equal to the expected value of the logarithmic derivative of the original probability distribution, a quantity that can be readily computed through Monte Carlo simulation. We shall quietly assume that it is possible to differentiate a function of discrete variables, leaving the associated technical difficulty for later.

Assume that for each V_i we have $P(\boldsymbol{x}_{V_i}) > 0$, that is the original and marginal probability distributions are strictly positive. Then

$$P(\boldsymbol{x}_{V_i}) = \exp\left(W(\boldsymbol{x}_{V_i})\right)/Z_{V_i}$$

and the logarithmic derivatives become simply

$$\frac{\partial \ln P(\boldsymbol{x}_{V_i})}{\partial x_u} = \frac{\partial W(\boldsymbol{x}_{V_i})}{\partial x_u} \quad \text{and} \quad \frac{\partial \ln P(\boldsymbol{x}_V)}{\partial x_u} = \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u}.$$

For each $u \in V_i$ we define a function $\mathcal{F}(\boldsymbol{x}_{V_i})$ through

$$\mathcal{F}(\boldsymbol{x}_{V_i}) = \frac{\partial \ln P(\boldsymbol{x}_{V_i})}{\partial x_u} = \frac{\partial W(\boldsymbol{x}_{V_i})}{\partial x_u} = \mathbb{E}\left[\frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} \,\middle|\, \boldsymbol{x}_{V_i} \right].$$

This function can be subsequently approximated by projecting it onto a subspace spanned by a basis ϕ consistent with the graph $G_i = (V_i, E_i)$, that is if ϕ contains only functions of variables x_u and $\boldsymbol{x}_{N(u)}$ that correspond to cliques of the graph G_i. The definition of consistency of a basis

will be further discussed in Section 4.1.3.4, because symmetries of the lattice and the translation invariance put further constraints on the allowed basis functions.

Write the basis abstractly as

$$\phi = \left\{ \phi_1, \phi_2, \dots, \phi_K \right\}$$

so that an approximation $\widehat{\mathcal{F}}(\boldsymbol{x}_{V_i})$ of $\mathcal{F}(\boldsymbol{x}_{V_i})$ within the space spanned by functions of ϕ becomes

$$\widehat{\mathcal{F}}(\boldsymbol{x}_{V_i}) = \sum_{i=1}^{K} c_i \phi_i(x_u, \boldsymbol{x}_{N(u)}).$$

The coefficients of the approximation are obtained through linear projection in an inner product space. We associate with the space spanned by the basis ϕ the inner product

$$\langle f, g \rangle = \int f(\boldsymbol{x}_{V_i}) g(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i} = \mathbb{E}\left[fg \mid x_{V_i} \right],$$

turning the space into an inner product space. Finding the set of coefficients $\boldsymbol{c}$ that is optimal with respect to the above inner product involves solving a least squares linear problem $A\boldsymbol{c} = \boldsymbol{b}$ showed in detail in Figure 2.8. The elements of matrix A, known as the Gram matrix, and vector $\boldsymbol{b}$ are

$$A_{kl} = \langle \phi_k, \phi_l \rangle \qquad \text{and} \qquad b_k = \langle \phi_k, \mathcal{F} \rangle.$$

The marginal probability density $P(\boldsymbol{x}_{V_i})$ appearing in the inner product is not known. However, the special form of the inner product allows these equations to be written in terms of the expected value with respect to the original probability density $P(\boldsymbol{x}_V)$ as

$$\begin{aligned} A_{kl} = \langle \phi_k, \phi_l \rangle &= \int \phi_k(\boldsymbol{x}_{V_i}) \phi_l(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i} \\ &= \int \phi_k(\boldsymbol{x}_{V_i}) \left(\int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i} \right) d\boldsymbol{x}_{V_i} \\ &= \int \phi_k(\boldsymbol{x}_{V_i}) \phi_l(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_V) d\boldsymbol{x}_V \\ &= \mathbb{E}\Big[\phi_k \phi_l \Big] \end{aligned} \tag{2.4}$$

and

$$
\begin{aligned}
b_k = \langle \phi_k, \mathcal{F} \rangle &= \int \phi_k(\boldsymbol{x}_{V_i}) \mathcal{F}(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i} \\
&= \int \phi_k(\boldsymbol{x}_{V_i}) \\
&\qquad \times \left(\int \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i} \Big/ \int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i} \right) \\
&\qquad\qquad \times \left(\int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i} \right) d\boldsymbol{x}_{V_i} \\
&= \int \phi_k(\boldsymbol{x}_{V_i}) \left(\int \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} P(\boldsymbol{x}_V) d\boldsymbol{x}_{V \setminus V_i} \right) d\boldsymbol{x}_{V_i} \\
&= \int \phi_k(\boldsymbol{x}_{V_i}) \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} P(\boldsymbol{x}_V) d\boldsymbol{x}_V \\
&= \mathbb{E}\left[\phi_k \frac{\partial W(\boldsymbol{x}_V)}{\partial x_u} \right]. \qquad (2.5)
\end{aligned}
$$

The variable x_u was implicitly made continuous by taking the derivative $\partial W \big/ \partial x_u$ in the definition of

$$\mathcal{F}(\boldsymbol{x}_{V_i}) = \frac{\partial W}{\partial x_u}.$$

Therefore, the function $\mathcal{F}(\boldsymbol{x}_{V_i})$ is continuous in x_u and the basis ϕ onto which we project $\mathcal{F}(\boldsymbol{x}_{V_i})$ must contain continuous functions of x_u in addition to discrete functions of $\boldsymbol{x}_{N(u)}$. Our choice for the basis ϕ is the *outer product* of a polynomial basis ϕ_c of functions of the continuous variable x_u and a polynomial basis ϕ_d of functions of the discrete variables $\boldsymbol{x}_{N(u)}$. The basis ϕ_c is simply

$$\phi_c = \Big\{ 1,\ x_u \Big\}$$

due to the fact that the expectation value $\mathbb{E}\big[\,\cdot\,\big]$ samples only two values of x_u, thus allowing two degrees of freedom. However, the basis ϕ_c could be extended to higher powers of x_u as shown in later chapters. The basis ϕ_d on the other hand is restricted by constraints and the only possible choice is

$$\phi_d = \Big\{ 1, \sum_{N(u)} x_v \Big\},$$

due to (i) lattice symmetries, (ii) shift invariance and (iii) consistency with independence graph G_i, which will be discussed in detail in Section 4.1.3. Therefore, the example basis takes the form

$$\begin{aligned}\phi &= \phi_c \times \phi_d \\ &= \left\{1,\ \sum_{N(u)} x_v,\ x_u,\ x_u \sum_{N(u)} x_v\right\}\end{aligned}$$

and the full linear system satisfied by the optimal expansion coefficients is shown on Figure 2.8. The projection is performed once for each graph $G_i = (V_i, E_i)$; however, the projection matrices for all lattices are typically accumulated simultaneously during a simulation. Additionally, due to shift invariance of the Ising model it is possible to accumulate the projection matrices by averaging over all spins in $\boldsymbol{x}_{V_i}$, as explained in Algorithm 2.5. The expectation values required by Equation 2.8 can be obtained by sampling the original probability distribution $P(\boldsymbol{x}_V)$.

2.2.1.2 *Probability reconstruction*

The coefficients $\boldsymbol{c}_i$ obtained with Algorithm 2.5 describe the best approximation of the logarithmic derivative of $P(\boldsymbol{x}_{V_i})$, $\partial W \big/ \partial x_u$ with respect to the norm induced by the inner product used. The knowledge of the logarithmic partial derivatives of $P(\boldsymbol{x}_{V_i})$ uniquely determines the probability $P(\boldsymbol{x}_{V_i})$, which we will now show by demonstrating the reconstruction of the probability $P(\boldsymbol{x}_{V_i})$ from the logarithmic partial derivatives.

To determine the probability distribution $P(\boldsymbol{x}_{V_i})$ uniquely one only needs to know a function proportional to it; that is, knowing a function $\bar{P}(\boldsymbol{x}_{V_i}) = C \times P(\boldsymbol{x}_{V_i})$ for some constant $C > 0$ allows one to compute

$$\frac{\bar{P}(\boldsymbol{x}_{V_i})}{\int \bar{P}(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i}} = \frac{P(\boldsymbol{x}_{V_i})}{\int P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i}} = P(\boldsymbol{x}_{V_i}),$$

where the constant and integral disappear since

$$\int P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i} = 1,$$

by definition. Therefore, the probability distribution $P(\boldsymbol{x}_{V_i})$ must be defined up to a multiplicative constant, or equivalently, its logarithm $W(\boldsymbol{x}_{V_i})$ must be known up to an additive constant. Our approach to defining $P(\boldsymbol{x}_{V_i})$ will be to construct a function $\bar{P}(\boldsymbol{x}_{V_i})$ proportional to $P(\boldsymbol{x}_{V_i})$.

Algorithm 2.5 Algorithm for computing expansion coefficients $\boldsymbol{c}_i$ using a weighted sampling scheme GETSAMPLE

procedure PROJECTION(m, $G_i = (V_i, E_i)$, ϕ_i)
  for $i = 1, 2, \ldots, m$ **do**
    $A_i \leftarrow$ EMPTYMATRIX($|\phi_i|, |\phi_i|$)
    $\boldsymbol{b}_i \leftarrow$ EMPTYVECTOR($|\phi_i|$)
  end for

  for all samples **do**
    $\boldsymbol{x}_V, w \leftarrow$ GETSAMPLE
    for $i = 1, 2, \ldots, m$ **do**
      for all $u \in V_i$ **do**
        $f \leftarrow \partial W \Big/ \partial x_u$
        $\boldsymbol{v} \leftarrow$ EVALUATEBASIS($\phi_i, u, \boldsymbol{x}_{V_i}$)
        $\boldsymbol{b}_i \leftarrow \boldsymbol{b}_i + wf\boldsymbol{v}$
        $A_i \leftarrow A_i + w\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$
      end for
    end for
  end for

  for $i = 1, 2, \ldots, m$ **do**
    $\boldsymbol{c}_i \leftarrow A_i^{-1}\boldsymbol{b}_i$
  end for
end procedure

We will choose a fixed state $\boldsymbol{y}_{V_i}$ such that $P(\boldsymbol{y}_{V_i}) > 0$. Then, we will produce a function $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ defined as

$$\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) = \frac{P(\boldsymbol{x}_{V_i})}{P(\boldsymbol{y}_{V_i})}.$$

For constant $\boldsymbol{y}_{V_i}$ we have

$$P(\boldsymbol{y}_{V_i}) \times \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) = P(\boldsymbol{x}_{V_i}),$$

thus $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ is proportional to $P(\boldsymbol{x}_{V_i})$, as required. We will first construct $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ for two states differing in only one component and then generalize it to two arbitrary states $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$, completing the reconstruction process.

The probability of every state is specified uniquely given a procedure for computing the ratio $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ of probabilities of two states $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$,

$$\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) = \frac{P(\boldsymbol{x}_{V_i})}{P(\boldsymbol{y}_{V_i})}.$$

Because probabilities of different states can vary greatly in magnitude, we would be better off to compute the logarithm of $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ instead, obtaining

$$\ln \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) = \ln \left(\frac{P(\boldsymbol{x}_{V_i})}{P(\boldsymbol{y}_{V_i})} \right) = W(\boldsymbol{x}_{V_i}) - W(\boldsymbol{y}_{V_i}).$$

Consider for a moment the simpler case of $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$ differing by only one component $u \in V_i$, i.e., for any $v \in V_i$ such that $v \neq u$ the equality $x_v = y_v$ holds. In this simple case we have

$$\begin{aligned} \ln \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) &= W(\boldsymbol{x}_{V_i}) - W(\boldsymbol{y}_{V_i}) \\ &= \int_{y_u}^{x_u} \left. \frac{\partial W}{\partial x_u} \right|_t dt \\ &= \int_{y_u}^{x_u} \sum_{j=1}^{K} c_{ij} \phi_j \left(t, \boldsymbol{x}_{N(u)} \right) dt \\ &= \sum_{j=1}^{K} c_{ij} \int_{y_u}^{x_u} \phi_j \left(t, \boldsymbol{x}_{N(u)} \right) dt, \end{aligned}$$

The notation $\left. \frac{\partial W}{\partial x_u} \right|_t$ implies that we take the derivative of $W(\boldsymbol{x}_{V_i})$ with respect to x_u and then evaluate it at $x_u = t$, renaming it so to avoid confusion with the value x_u appearing in the integration limit.

a quantity known given the coefficients $\boldsymbol{c}_i$.

The general problem of computing $\ln \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ for arbitrary vectors $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$ can be dealt with using the single-component version. Let $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$ differ in multiple components. Construct a path of $k+1$ states

$$\left\{ \boldsymbol{z}_0,\ \boldsymbol{z}_1,\ \ldots,\ \boldsymbol{z}_{k-1},\ \boldsymbol{z}_k \right\}$$

such that $\boldsymbol{z}_0 = \boldsymbol{y}_{V_i}, \boldsymbol{z}_k = \boldsymbol{x}_{V_i}$ and any two consecutive vectors $\boldsymbol{z}_j$ and $\boldsymbol{z}_{j+1}$ differ by only one component. Additionally, we require that $P(\boldsymbol{z}_j) > 0$ for $0 \leq j \leq k$, to ensure that the logarithm is finite. Then, we may write

$$\ln \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) = \ln \bar{P}(\boldsymbol{z}_0, \boldsymbol{z}_k) = \sum_{j=0}^{k-1} \ln \bar{P}(\boldsymbol{z}_j, \boldsymbol{z}_{j+1}),$$

essentially adding up the logarithmic quotients along the path. The case of strictly positive probability, i.e. $P(\boldsymbol{x}_{V_i}) > 0$, allows any path between $\boldsymbol{x}_{V_i}$ and $\boldsymbol{y}_{V_i}$ that satisfies the *single-component differece* constraint, as the path will automatically satisfy the positivity constraint. In particular, the path constructed by changing each differing component exactly once satisfies these constraints and is used by the Algorithm 2.6.

The probability of a state $\boldsymbol{x}_{V_i}$ is then defined through

$$P(\boldsymbol{x}_{V_i}) = P(\boldsymbol{y}_{V_i}) \times \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}),$$

with $\bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i})$ computed as above by following a path of states connecting $\boldsymbol{x}_{V_i}$ with $\boldsymbol{y}_{V_i}$. The probability of the fixed state $\boldsymbol{y}_{V_i}$ acts here as a normalization constant, ensuring $P(\boldsymbol{x}_{V_i})$ is properly normalized. While $P(\boldsymbol{y}_{V_i})$ may be computed through

$$P(\boldsymbol{y}_{V_i}) = \frac{1}{\int \bar{P}(\boldsymbol{x}_{V_i}, \boldsymbol{y}_{V_i}) d\boldsymbol{x}_{V_i}},$$

in most actual computations the knowledge of the normalization constant is not necessary.

2.2.2 *Sampling*

In order to discuss the sampling procedure it is important to describe the independence structure of the graphs. As we have seen in prior sections, the graphs were constructed so that for any $u, v \in V_i \setminus V_{i+1}$ the variables x_u and x_v are conditionally independent given the variables in $\boldsymbol{x}_{V_{i+1}}$, written

$$x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V_{i+1}},$$

with the conditional independence being with respect to the probability $P(\boldsymbol{x}_{V_i})$. That is, assuming that the variables $\boldsymbol{x}_{V_i}$ are distributed according to $P(\boldsymbol{x}_{V_i})$, the variables x_u and x_v for $u, v \in V_i \setminus V_{i+1}$ are conditionally independent given $\boldsymbol{x}_{V_{i+1}}$. Therefore, knowing the values of the variables $\boldsymbol{x}_{V_{i+1}}$ allows to fill-in the remaining variables $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ individually as they are independent of each other.

The sampling algorithm proceeds as follows. The top graph $G_m = (V_m, E_m)$ with probability distribution $P(\boldsymbol{x}_{V_m})$ has to be sampled using

Algorithm 2.6 Algorithm for computing the logarithmic ratio of marginal probability densities for two states $\boldsymbol{x}_V$ and $\boldsymbol{y}_V$.

function LOGQUOTIENT($\boldsymbol{x}_V$, $\boldsymbol{y}_V$, V, $\boldsymbol{c}$, $\boldsymbol{\phi}$)
  $\boldsymbol{z}_V \leftarrow y_V$
  $W \leftarrow 0$
  for all $u \in V$ **do**
    if $z_u \neq y_u$ **then**
      $\Delta W \leftarrow$ LOGLOCALQUOTIENT$(u, \boldsymbol{z}, \boldsymbol{c}, \boldsymbol{\phi})$
      if $z_u = -1$ **then**
        $W \leftarrow W + \Delta W$
      else
        $W \leftarrow W - \Delta W$
      end if
      $z_u \leftarrow y_u$
    end if
  end for
  return W
end function

function LOGLOCALQUOTIENT($u \in V$, $\boldsymbol{z}$, $\boldsymbol{c}$, $\boldsymbol{\phi}$)
  return $\sum_{j=1}^{m} c_j \int_{-1}^{1} \phi_j\left(z_u, \boldsymbol{z}_{N(u)}\right) dz_u$
end function

some alternative algorithm: Markov Chain Monte Carlo or even direct sampling. Thus, we obtain a sample

$$\boldsymbol{x}_{V_m} \sim P(\boldsymbol{x}_{V_m})$$

and begin sampling finer lattices given values of the variables on the immediately coarser lattice. Assume $\boldsymbol{x}_{V_{i+1}}$ has been successfully sampled and we wish to sample the remaining variables on the immediately finer lattice V_i, i.e., sample the variables in $\boldsymbol{x}_{V_i \setminus V_{i+1}}$. We obtain

$$\begin{aligned} P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) &= \prod_{u \in V_i \setminus V_{i+1}} P(x_u \mid \boldsymbol{x}_{V_{i+1}}) \\ &= \prod_{u \in V_i \setminus V_{i+1}} P(x_u \mid \boldsymbol{x}_{N(u)}) \end{aligned}$$

due to conditional independence. By construction, $N(u) \subset V_{i+1}$ and the conditional probability $P(x_u \mid x_{N(u)})$ depends only on one unknown

value, x_u, making it straightforward to sample. Defining ΔW_u to be the logarithmic quotient at node $u \in V_i \setminus V_{i+1}$ (cf. Algorithm 2.6)

$$\begin{aligned}\Delta W_u &= \frac{P(x_u = 1 \mid \boldsymbol{x}_{N(u)})}{P(x_u = -1 \mid \boldsymbol{x}_{N(u)})} \\ &= \sum_{j=1}^{K} c_{ij} \int_{-1}^{1} \phi_j \left(x_u, \boldsymbol{x}_{N(u)}\right) dx_u\end{aligned}$$

we can further write

$$\begin{aligned}P(x_u \mid \boldsymbol{x}_{V_{i+1}}) &= P(x_u \mid \boldsymbol{x}_{N(u)}) \\ &= \begin{cases} \dfrac{\mathrm{e}^{\Delta W_u}}{\mathrm{e}^{\Delta W_u} + \mathrm{e}^{-\Delta W_u}} & \text{if } x_u = 1, \\ \dfrac{\mathrm{e}^{-\Delta W_u}}{\mathrm{e}^{\Delta W_u} + \mathrm{e}^{-\Delta W_u}} & \text{if } x_u = -1. \end{cases}\end{aligned} \tag{2.6}$$

Thus, sampling the variable x_u involves choosing at random between $x_u = -1$ and $x_u = 1$ with probabilities

$$\frac{\mathrm{e}^{-\Delta W_u}}{\mathrm{e}^{\Delta W_u} + \mathrm{e}^{-\Delta W_u}} \quad \text{and} \quad \frac{\mathrm{e}^{\Delta W_u}}{\mathrm{e}^{\Delta W_u} + \mathrm{e}^{-\Delta W_u}},$$

respectively. Sampling all of the variables in $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ completes the missing variables, making all of the variables in $\boldsymbol{x}_{V_i}$ known. The top-down approach ends when all of the original variables in $\boldsymbol{x}_V = \boldsymbol{x}_{V_0}$ are sampled.

The above sampling algorithm defines a probability distribution

$$\begin{aligned}P_{\approx}(\boldsymbol{x}_V) = \hat{P}(\boldsymbol{x}_{V_m}) \times \hat{P}(\boldsymbol{x}_{V_{m-1} \setminus V_m} \mid \boldsymbol{x}_{V_m}) \times \ldots \times \\ \hat{P}(\boldsymbol{x}_{V_{i-1} \setminus V_i} \mid \boldsymbol{x}_{V_i}) \times \ldots \times P(\boldsymbol{x}_{V_0} \mid \boldsymbol{x}_{V_0 \setminus V_1}),\end{aligned} \tag{2.7}$$

which we will refer to as the trial probability distribution in the importance sampling framework. In the discussion that follows, we denote with a hat all approximate quantities; that is, $\widehat{A}$ is always an approximation of A. Were the conditional probabilities exact, $P_{\approx}(\boldsymbol{x}_V)$ would be equal to $P(\boldsymbol{x}_V)$ and the sampling algorithm would be complete, as was the case in one dimension. However, due to the necessity of using approximate marginal distributions, the conditional probabilities are also approximate

Algorithm 2.7 Algorithm for generating a weighted sample $\boldsymbol{x}_V \sim P(\boldsymbol{x}_V)$ along with logarithm of its weight $\ln w$. The conditional probability $P(x_u \mid \boldsymbol{x}_{N(u)})$ is defined through Equation 2.6.

function GETSAMPLE($G_i = (V_i, E_i)$, $\boldsymbol{c}_i$, $\boldsymbol{\phi}_i$)
  $\boldsymbol{x}_{V_m} \sim P(\boldsymbol{x}_{V_m})$
  $\ln w \leftarrow \ln \hat{P}(\boldsymbol{x}_{V_m})$
  for $i = m - 1$ down to 0 **do**
    for all $u \in V_i \setminus V_{i+1}$ **do**
      $x_u \sim \hat{P}(x_u \mid \boldsymbol{x}_{N(u)})$
      $\ln w \leftarrow \ln w + \ln \hat{P}(x_u \mid \boldsymbol{x}_{N(u)})$
    end for
  end for
  $\ln w \leftarrow \ln P(\boldsymbol{x}_V) - \ln w$
  return $\ln w$, $\boldsymbol{x}_V$
end function

and in general $P_{\approx}(\boldsymbol{x}_V) \neq P(\boldsymbol{x}_V)$. To correct for the approximation, we attach a weight

$$w = \frac{P(\boldsymbol{x}_V)}{P_{\approx}(\boldsymbol{x}_V)}$$

to each generated sample $\boldsymbol{x}_V$. Therefore, the expected value of a function $f(\boldsymbol{x}_V)$ is written as

$$\mathbb{E}\Big[f(\boldsymbol{x}_V)\Big] = \frac{\sum_{i=1}^{N} w_i f(\boldsymbol{x}_i)}{\sum_{i=1}^{N} w_i},$$

a minimal change to the standard Monte Carlo expression.

2.2.3 *Iterative improvement*

The algorithm for sampling the lattice ladder requires prior knowledge of the approximate expansion coefficients $\boldsymbol{c}_i$ for each lattice. While the fast marginalization algorithm used for computing the expansion coefficients can utilize any sampling algorithm, the use of an algorithm different than the one described above defeats its purpose. Instead, following (Chorin,

2008) we make the sampling algorithm iteratively improve itself through a fixed point iteration.

Consider the set of expansion coefficients for each lattice, $\boldsymbol{c}_i$. We begin by choosing a reasonable initial guess for the values of all expansion coefficients for all the lattices. The known values are used to compute expected values required by the fast marginalization, producing an updated set of coefficients. Assuming convergence, the process quickly leads to a stable set of coefficients.

Due to the stochastic nature of the fixed point function it is difficult to perform a rigorous analysis of conditions leading to convergence of the iteration even for the simplest models. However, in practice it is observed that the iteration converges in at most three iterations under most conditions. The only failure was recorded for very strong couplings, a case where the fast marginalization algorithm fails due to low variability in observed states leading to singular projection matrices.

2.2.4 *Analysis*

The two-dimensional Ising model brought important differences from the one-dimensional case. The graph coarsening cannot be performed exactly, marginal probabilities must be computed approximately using the fast marginalization algorithm of Chorin (2003, 2008), and an iterative process has to be used to determine the expansion coefficients in order to make the overall scheme an effective sampling algorithm. We analyze these new elements below.

2.2.4.1 *Approximate graph coarsening*

As it was the case in one-dimension, at each stage of the coarsening algorithm a graph $G_i = (V_i, E_i)$ with random variables $\boldsymbol{x}_{V_i}$ described by marginal probability density $P(\boldsymbol{x}_{V_i})$ are given. Importantly, the marginal probability distribution is not known explicitly but only through its definition. The required output is a coarser graph $G_{i+1} = (V_{i+1}, E_{i+1})$ such that $V_{i+1} \subset V_i$ and for any $u, v \in V_i \setminus V_{i+1}$ the variables x_u and x_v are independent given $\boldsymbol{x}_{V_{i+1}}$. As before, the graph describes the independency structure, translating the conditional independency into the requirement that $V_{i+1} \subset V_i$ be a vertex cover of the graph G_i.

However, because the marginal probability density $P(\boldsymbol{x}_{V_i})$ is assumed to be only an approximation of the true marginal, the graph G_i is also only approximate. As a result, the spins in $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ are not independent of each other as assumed by the sampling Algorithm 2.7.

$$\begin{pmatrix} \mathbb{E}\left[1\right] & \mathbb{E}\left[\sum_{N(u)} x_v\right] & \mathbb{E}\left[x_u\right] & \mathbb{E}\left[x_u \sum_{N(u)} x_v\right] \\ \mathbb{E}\left[\sum_{N(u)} x_v\right] & \mathbb{E}\left[\left(\sum_{N(u)} x_v\right)^2\right] & \mathbb{E}\left[x_u \sum_{N(u)} x_v\right] & \mathbb{E}\left[x_u \left(\sum_{N(u)} x_v\right)^2\right] \\ \mathbb{E}\left[x_u\right] & \mathbb{E}\left[x_u \sum_{N(u)} x_v\right] & \mathbb{E}\left[x_u^2\right] & \mathbb{E}\left[x_u^2 \sum_{N(u)} x_v\right] \\ \mathbb{E}\left[x_u \sum_{N(u)} x_v\right] & \mathbb{E}\left[x_u \left(\sum_{N(u)} x_v\right)^2\right] & \mathbb{E}\left[x_u^2 \sum_{N(u)} x_v\right] & \mathbb{E}\left[x_u^2 \left(\sum_{N(u)} x_v\right)^2\right] \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} \mathbb{E}\left[\frac{\partial W}{\partial x_u}\right] \\ \mathbb{E}\left[\frac{\partial W}{\partial x_u} \sum_{N(u)} x_v\right] \\ \mathbb{E}\left[\frac{\partial W}{\partial x_u} x_u\right] \\ \mathbb{E}\left[\frac{\partial W}{\partial x_u} x_u \sum_{N(u)} x_v\right] \end{pmatrix} \tag{2.8}$$

Figure 2.8: Complete linear system used to compute an example set of approximate coefficients $\boldsymbol{c} = (c_1, c_2, c_3, c_4)$ using the fast marginalization algorithm.

2.2.4.2 *fast marginalization*

The fast marginalization algorithm is a necessary step for computing the approximate marginal densities. While other methods exist for computing expansion coefficients, they are either applicable to a very limited number of problems (Brandt and Ron, 2001b) or do not actually compute the marginal probabilities (Swendsen, 1984b). Therefore, at present there is no other algorithm capable of performing the computation and it is imperative to study it on its own.

fast marginalization computes the best approximation of the derivative $\partial W / \partial x_u$ in the norm induced by the inner product used. The particular choice of the inner product, namely

$$\langle f, g \rangle = \int f(\boldsymbol{x}_{V_i}) g(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i},$$

is a crucial step necessary to make the algorithm work. Comparing the algorithm with the definition of marginalization (cf. Binney et al. 1992), the sole difference is the presence of the weight $P(\boldsymbol{x}_{V_i})$. As a result, the inner product does not attach equal importance to all the possible states, reducing the pool of important (high weight) states down to a manageable size.

The appropriate weight performs an additional function. Inspecting the Equations 2.4 and 2.5 shows that the weight can be used to remove the unknown marginal probability $P(\boldsymbol{x}_{V_i})$ from the inner products through careful algebraic manipulation. The result is a series of expected values with respect to the original, rather than marginal, probability distribution that can be computed effectively using the same set of samples (cf. Equation 2.8).

The use of a weighted inner product has a negative side, however, especially important in the case of approximating the probability distribution. The use of $P(\boldsymbol{x}_{V_i})$ as weight skews the approximation toward states of high probability. While difficult to show rigorously, in practice the *high-probability bias* leads to incorrect estimation of the ratios of probabilities of high- and low-probability states. Due to the importance of the particular choice of weight, tackling this issue is difficult and will be discussed in later chapters.

In principle, given a very large basis and unlimited samples, the fast marginalization algorithm would compute expansion coefficients as close to their true values as desired. However, the fast marginalization algo-

rithm as stated here and in Chorin (2008) would not produce exact results due to a technicality. The function

$$\mathcal{F}(\boldsymbol{x}_{V_i}) = \frac{\partial W(\boldsymbol{x}_{V_i})}{\partial x_u}$$

is defined over a variable x_u, which was implicitly made continuous. However, the algorithm samples that variable only at two points, $x_u = -1$ or 1, ignoring the behavior of $\mathcal{F}(\boldsymbol{x}_{V_i})$ over the rest of the interval $[-1, 1]$. The exact computation carried out in Appendix A can be used to see the exact behavior of $\mathcal{F}(\boldsymbol{x}_{V_i})$. Indeed, beginning with Equation A.10, we obtain

$$\begin{aligned} W(\boldsymbol{x}_{V_i}) &= \ln\left(\frac{2^{n/2}}{Z(\mu)}\prod_{E_i}\cosh\left[\mu\left(x_u + x_v\right)\right]\right) \\ &= \frac{n}{2}\ln 2 - \ln Z(\mu) + \sum_{E_i}\ln\cosh\left[\mu(x_u + x_v)\right] \end{aligned}$$

where the products and sums are over all edges $(u, v) \in E_i$. Therefore, the derivative becomes

$$\mathcal{F}(\boldsymbol{x}_{V_i}) = \frac{\partial W(\boldsymbol{x}_{V_i})}{\partial x_u} = \mu\sum_{N(u)}\tanh\left[\mu(x_u + x_v)\right],$$

a function plotted on Figure 2.9, which shows clearly that the function $\mathcal{F}(\boldsymbol{x}_{V_i})$ is far from linear in x_u, which was the assumption used in constructing the basis ϕ. Additionally, due to the weighted inner product, the linear approximation obtained by the fast marginalization algorithm will tend to favor the areas of high probability: $x_u = 1$ in the positive and $x_u = -1$ in the negative case. While the addition of basis functions of higher powers in x_u would allow for a better fit, it would require sampling values of x_u at more points than just the endpoints. The solution to this and other problems will be discussed in the later chapters.

2.2.4.3 *Sampling*

The sampling Algorithm 2.7 proposed by Chorin (2008) is very straightforward thanks to the careful construction of the lattice ladder during the graph coarsening stage. While the computation of approximate marginal probability densities moves bottom-to-top, or from fine to coarse, the sampling moves in the opposite direction, beginning with sampling the coarsest lattice and systematically filling up the finer lattices. There are two

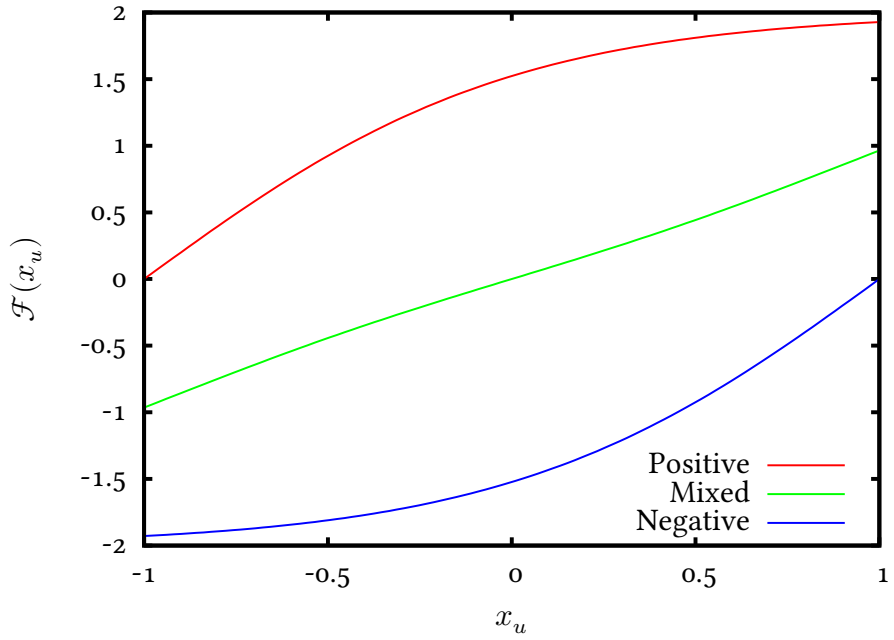


Figure 2.9: Plot of the exact derivative $\partial W / \partial x_u$ computed for the one-dimensional Ising model. The curves represent dependence of the derivative on the continuous variable x_u for the three distinguishable combinations of the neighboring discrete variables: positive $\sum_{N(u)} x_v > 0$, mixed $\sum_{N(u)} x_v = 0$, and negative $\sum_{N(u)} x_v < 0$.

equally valid ways of looking at the sampling algorithm and I will briefly describe both.

We begin with the more straightforward. As was shown in Equation 2.7, the sample $\boldsymbol{x}_V$ generated by the sampling process can be described as coming from the probability distribution $P_{\approx}(\boldsymbol{x}_V)$ defined through

$$P_{\approx}(\boldsymbol{x}_V) = \hat{P}(\boldsymbol{x}_{V_m}) \times \hat{P}(\boldsymbol{x}_{V_{m-1} \setminus V_m} \mid \boldsymbol{x}_{V_m}) \times \ldots \times \hat{P}(\boldsymbol{x}_{V_{i-1} \setminus V_i} \mid \boldsymbol{x}_{V_i}) \times \ldots \times P(\boldsymbol{x}_{V_0} \mid \boldsymbol{x}_{V_0 \setminus V_1}).$$

Therefore, the set of approximate marginal probability densities simply defines a trial proposal density, which one hopes approximates well the target density $P(\boldsymbol{x}_V)$. At the final stage, once the entire sample has been selected, the target probability $P(\boldsymbol{x}_V)$ may be evaluated and the discrepancy between the two corrected by assigning the sample a weight $w = \hat{P}(\boldsymbol{x}_V)/P(\boldsymbol{x}_V)$. Assuming that the proposal density $\hat{P}(\boldsymbol{x}_V)$ is at least as broad as the target distribution, i.e., that $P(\boldsymbol{x}_V) > 0$ implies $P_{\approx}(\boldsymbol{x}_V) > 0$, the weights exactly correct for the mismatch between the

two distributions. However, for reasons of efficiency one would hope that the range of weights is small, as the error in the estimation of expected value grows with the range of weights.

However, the algorithm could also be analyzed in more detail. Assume that $\boldsymbol{x}_{V_{i+1}}$ has been sampled according to $\hat{P}(\boldsymbol{x}_{V_{i+1}})$ and we wish to fill out the remaining values in $\boldsymbol{x}_{V_u}$ in such a way that the complete sample $\boldsymbol{x}_{V_i}$ is distributed according to $\hat{P}(\boldsymbol{x}_{V_i})$. Because the marginalization was approximate, there is a mismatch between the marginal densities $\hat{P}(\boldsymbol{x}_{V_i})$ and $\hat{P}(\boldsymbol{x}_{V_{i+1}})$ in that the latter is not an exact marginal of the former, but only an approximate one. Therefore, an error is accumulated not only due to the mismatch between $\hat{P}(\boldsymbol{x}_V)$ and $P(\boldsymbol{x}_V)$, but also due to approximations at intermediate lattices.

For the moment, assume that the probability $\hat{P}(\boldsymbol{x}_{V_i})$ is exact, writing it $P(\boldsymbol{x}_{V_i})$. Using Bayes' rule, the exact conditional probability would be

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) = \frac{P(\boldsymbol{x}_{V_i})}{P(\boldsymbol{x}_{V_{i+1}})},$$

however the exact marginal $P(\boldsymbol{x}_{V_{i+1}})$ is not known. Therefore, the sample $\boldsymbol{x}_{V_i}$ is generated from the probability distribution

$$\boldsymbol{x}_{V_i} \sim \hat{P}(\boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) \hat{P}(\boldsymbol{x}_{V_{i+1}}).$$

Substituting for the conditional, we find that the sample was generated from the probability

$$\boldsymbol{x}_{V_i} \sim P(\boldsymbol{x}_{V_i}) \times \frac{\hat{P}(\boldsymbol{x}_{V_{i+1}})}{P(\boldsymbol{x}_{V_{i+1}})},$$

thus requiring a correction using weight

$$w_{i+1} = \frac{P(\boldsymbol{x}_{V_{i+1}})}{\hat{P}(\boldsymbol{x}_{V_{i+1}})}$$

in order for the sample to be distributed according to $P(\boldsymbol{x}_{V_i})$. Here, $P(\boldsymbol{x}_{V_{i+1}})$ is the exact marginal (assuming the probability at level i is exact), while $\hat{P}(\boldsymbol{x}_{V_{i+1}})$ is the approximate marginal; the weight w_i simply corrects for this local mismatch between the consecutive lattices. The fi-

nal weight w is then simply the product of these local weights (cf. Section 5.1.1,

$$w = w_1 \times w_2 \times \ldots \times w_{m-1} \times w_m = \prod_{i=1}^{m} w_i.$$

These weights can only be computed because the expression

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) = \frac{P(\boldsymbol{x}_{V_i})}{P(\boldsymbol{x}_{V_{i+1}})}$$

can be computed; in fact, this conditional probability is given in closed form as

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) = \prod_{V_i \setminus V_{i+1}} P(x_u \mid \boldsymbol{x}_{V_{i+1}}),$$

which rests on the fact that variables in $V_i \setminus V_{i+1}$ are conditionally independent given those in V_{i+1}.

The sampling process can thus be seen also as a series of small steps between lattices, where an error is committed at each level and the corrections are accumulated as weights. This point of view is very helpful as it shows how critical is the conditional independence assumption, but also because it shows a way of reducing weights. Because the final weight w is a product of lattice-to-lattice weights, if it were possible to correct the sample at an earlier stage, the final weight could be reduced.

In fact, this and other weight-reducing strategies are necessary to achieve good results. Numerical experiments show that the range of weights in the two-dimensional Ising model grows exponentially with the number of variables in the model, i.e., at a rate of roughly $\mathcal{O}(\mathrm{e}^{kn^2})$, making it virtually impossible to sample even modest lattices of size 32×32 using the method of Chorin (2008). These and other modifications will be discussed in the subsequent chapter.

Part I

AN ADVANCED MARKOV FIELD SAMPLER

3

GRAPH COARSENING

In the previous chapter we saw how a ladder of marginal probability densities may be used to efficiently sample from a given probability $P(\boldsymbol{x}_V)$. This was done in effect by sampling from the acyclic form of $P(\boldsymbol{x}_V)$, written as

$$P(\boldsymbol{x}_V) = P(x_1)P(x_2 \mid x_1) \dots P(x_{|V|} \mid x_1, x_2, \dots, x_{|V|-1}).$$

Even when this probability was only approximate, we found that we may still use it as a trial density through importance sampling. In the present chapter we will discuss the circumstances that led us earlier to coarsen the Ising lattice using the checkerboard coarsening pattern and how this method may be generalized to more complex situations.

3.1 THE SAMPLING MOTIVATION

Suppose we split the variables $\boldsymbol{x}_V$ into two subsets, $\boldsymbol{x}_U$ and the remainder $x_{V\setminus U}$. Further assume that we know the values of $\boldsymbol{x}_U$, but may compute the probability density $P(\boldsymbol{x}_V)$ only up to a multiplicative constant; under what circumstances can we sample the variables $x_{V\setminus U}$ given the known values $\boldsymbol{x}_U$?

The prior chapter showed us that when variables of $x_{V\setminus U}$ are conditionally independent of each other given $\boldsymbol{x}_U$ we may indeed do so. The reason for this situation is rather surprising; let

$$P(\boldsymbol{x}_V) = \frac{F(\boldsymbol{x}_V)}{Z_V}, \qquad \text{where} \qquad Z_V = \int F(\boldsymbol{x}_V) d\boldsymbol{x}_V$$

is the unknown normalizing factor and $F(\boldsymbol{x}_V)$ is the known unnormalized probability density. When two variables x_u and x_v are conditionally independent given $\boldsymbol{x}_U$ for any $u, v \in V \setminus U$, written formally as $x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_U$, the probability density function factors into two parts, one depending on x_u but not on x_v and another dependent only on x_v but not on x_u,

$$P(\boldsymbol{x}_V) = \frac{F(\boldsymbol{x}_V)}{Z_V} = \frac{F_u(\boldsymbol{x}_u, \boldsymbol{x}_U) F_v(\boldsymbol{x}_v, \boldsymbol{x}_U)}{Z_V}.$$

The conditional independence of all variables in $V \setminus U$ then implies that the probability factors as

$$P(\boldsymbol{x}_V) = \frac{1}{Z_V} F_V(\boldsymbol{x}_V) \prod_{u \in V \setminus U} F_u(\boldsymbol{x}_u, \boldsymbol{x}_U).$$

As a result, the conditional probability $P(\boldsymbol{x}_{V\setminus U} \mid \boldsymbol{x}_U)$ becomes

$$\begin{aligned} P(\boldsymbol{x}_{V\setminus U}) \mid \boldsymbol{x}_U) &= \frac{P(\boldsymbol{x}_{V\setminus U}, \boldsymbol{x}_U)}{P(\boldsymbol{x}_U)} \\ &= \frac{F(\boldsymbol{x}_{V\setminus U}, \boldsymbol{x}_U)}{\int F(\boldsymbol{x}_{V\setminus U}, \boldsymbol{x}_U) d\boldsymbol{x}_{V\setminus U}} \\ &= \frac{\prod_{u \in V\setminus U} F_u(\boldsymbol{x}_u, \boldsymbol{x}_U)}{\prod_{u \in V\setminus U} \int F_u(\boldsymbol{x}_u, \boldsymbol{x}_U) dx_u} \\ &= \prod_{u \in V\setminus U} \frac{F_u(\boldsymbol{x}_u, \boldsymbol{x}_U)}{\int F_u(\boldsymbol{x}_u, \boldsymbol{x}_U) dx_u}, \end{aligned} \tag{3.1}$$

a product of terms that may be easily computed. The multidimensional integral over $\boldsymbol{x}_{V\setminus U}$ disappeared as did the problematic normalization constant Z_V, instead requiring the computation of a series of one-dimensional integrals. Therefore, if the variables in $\boldsymbol{x}_{V\setminus U}$ are conditionally independent given $\boldsymbol{x}_U$ we may compute their properly normalized conditional probability and thus efficiently sample from it.

If $\boldsymbol{x}_U$ were sampled according to the exact marginal density $P(\boldsymbol{x}_U)$, the sampling algorithm would be complete, since the complete state $\boldsymbol{x}_V$ would be sampled according to

$$P(\boldsymbol{x}_U) P(\boldsymbol{x}_{V\setminus U} \mid \boldsymbol{x}_U) = P(\boldsymbol{x}_{V\setminus U}, \boldsymbol{x}_U) = P(\boldsymbol{x}_V),$$

as we would hope. Since we were successful in splitting $\boldsymbol{x}_V$ into $\boldsymbol{x}_{V\setminus U}$ and $\boldsymbol{x}_U$, perhaps we could repeat this procedure and split $\boldsymbol{x}_U$ into yet smaller subsets? Unfortunately, we do not know anything about the conditional independencies present in the marginal density $P(\boldsymbol{x}_U)$ and thus do not know how $\boldsymbol{x}_U$ could be split. To gather this information we will require posing the problem using the formulation of graphical models, seeking to tackle it using graph manipulation rather than algebra.

3.2 GRAPHICAL MODEL REPRESENTATION

The conditional independence between variables is at the core of the framework of graphical models. While posing a probability distribution as a graphical model does not add any new information about the probability distribution, it allows for very convenient ways of manipulating the existing information. As a result, we see this framework solely as a useful extension or way of thinking.

For any given probability density $P(\boldsymbol{x}_V)$ we may define a dependence graph G, composed of a set of nodes V and a set of edges E, denoted $G = (V, E)$. Each variable x_u is assigned a unique node $u \in V$. The undirected edges (u, v) forming the set E are used to represent the conditional independence relations between the variables of $\boldsymbol{x}_V$ in the following manner. For $u, v \in V$, if an edge (u, v) in E does not exist then the variables x_u and x_v are conditionally independent given all other variables, $x_{V\setminus\{u,v\}}$. Formally,

$$(u, v) \notin E_U \implies x_u \perp\!\!\!\perp x_v \mid x_{V\setminus\{u,v\}}.$$

In case of the Ising model the edges signified direct couplings between the random variables and we may informally think of edges in the dependency graph G to imply such couplings.

3.2.1 *Marginalization on a graph*

The graphical framework allows us to perform operations involving changes in the conditional independence structure simply by operating on the graph, making algebraic operations on the probability distribution more visual and easier to grasp. In particular, the marginalization of a variable is a straightforward operation on the dependency graph.

Consider a node $u \in V$ with a set of neighbors $N(u)$, i.e., a set of nodes $v \in V$ such that an edge $(u, v) \in E$ exists. Marginalizing the variable x_u means, algebraically, the computation of

$$P(\boldsymbol{x}_{V\setminus u}) = \int P(\boldsymbol{x}_V) dx_u.$$

Since we know that the only variables that are not conditionally independent of x_u given the remaining nodes are those of $\boldsymbol{x}_{N(u)}$, we obtain that $P(\boldsymbol{x}_V)$ factors as

$$P(\boldsymbol{x}_V) = F_\alpha(\boldsymbol{x}_{V\setminus u}) F_u(x_u, \boldsymbol{x}_{N(u)}) / Z_V.$$

This result is a manifestation of conditional independence, the fact that once $\boldsymbol{x}_{N(u)}$ are known the values of the remaining variables $\boldsymbol{x}_{V\setminus\bar{N}(u)}$ have no impact on the conditional probability of x_u.

Substituting this factorization into the integral we obtain

$$\begin{aligned} P(\boldsymbol{x}_{V\setminus u}) &= \int P(\boldsymbol{x}_V)dx_u \\ &= \int F_\alpha(\boldsymbol{x}_{V\setminus u})F_u(x_u,\boldsymbol{x}_{N(u)})dx_u \Big/ Z_V \\ &= F_\alpha(\boldsymbol{x}_{V\setminus u})\int F_u(x_u,\boldsymbol{x}_{N(u)})dx_u \Big/ Z_V. \end{aligned}$$

The first term remains unchanged, appearing the same in $P(\boldsymbol{x}_V)$, showing that all independence relations between variables of $\boldsymbol{x}_{V\setminus\bar{N}(u)}$ remain unchanged. However, we are unable to say anything about the term

$$F_\beta(\boldsymbol{x}_{N(u)}) = \int F_u(x_u,\boldsymbol{x}_{N(u)})dx_u,$$

which as a function of only $\boldsymbol{x}_{N(u)}$ suggests that the variables $\boldsymbol{x}_{N(u)}$ may no longer be conditionally independent.

The above rather involved derivation may be encoded more simply using the graphical approach. When a node $u \in V$ is marginalized, both the node and all edges incident upon it are removed from the graph. However, all nodes $v \in N(u)$ are then connected to each another, forming a clique. This approach represents the possible loss of conditional independence among nodes of $N(u)$, thus encoding the worst possible scenario.

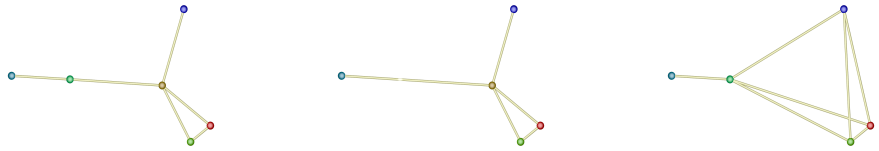
EXAMPLE 3.1. Consider a PDF $P(\boldsymbol{x}_V) = P(x_1,x_2,x_3,x_4,x_5,x_6)$ that factors as

$$\begin{aligned} P(\boldsymbol{x}_V) = \frac{1}{Z_V}&F_{12}(x_1,x_2)F_{13}(x_1,x_3)F_{23}(x_2,x_3) \\ &\times F_{24}(x_2,x_4)F_{26}(x_2,x_6)F_{45}(x_4,x_5). \end{aligned}$$

Because of this factorization, we see that the dependency graph $G = (V,E)$ must have a set of edges

$$E = \{(1,2),(1,3),(2,3),(2,4),(2,6),(4,5)\},$$

as shown on Figure 3.10a. When variable x_4 is marginalized, we must remove it from the graph along with edges $(2,4)$ and $(4,5)$; instead, the nodes of $N(4) = \{2,5\}$ are connected to form a clique, requiring only



(a) Original graph (b) Marginalized x_4 (c) Marginalized x_2

Figure 3.10: Graph from Example 3.1, both original and after marginalizing variables x_2 or x_4.

one new edge $(2, 5)$. The resulting graph shown of Figure 3.10b is not significantly denser than the original.

Instead of x_4, consider marginalizing variable x_2. This time we remove node 2 and edges containing it, noticing that the set of nodes that must be reconnected has grown significantly and is $N(2) = \{1, 3, 4, 6\}$. Reconnecting the nodes requires $\binom{4}{2} = 6$ edges, increasing the density of the resulting graph shown on Figure 3.10c. ■

The graph $G = (V, E)$ encodes not only the conditional independence relations of $P(\boldsymbol{x}_V)$, but also the dependencies of the conditional probability of every variable $u \in V$,

$$P(x_u \mid \boldsymbol{x}_{V\setminus u}) = \frac{P(\boldsymbol{x}_{V\setminus u}, x_u)}{P(\boldsymbol{x}_{V\setminus u})}.$$

The variable x_u is dependent only on the variables $\boldsymbol{x}_{N(u)}$, allowing us to write the factorization of $P(\boldsymbol{x}_V)$ as

$$P(\boldsymbol{x}_V) = \frac{1}{Z_V} F_\alpha(\boldsymbol{x}_{V\setminus u}) F_\beta(\boldsymbol{x}_{N(u)}, x_u).$$

Inserting it into the conditional probability formula we obtain

$$\begin{aligned} P(x_u \mid \boldsymbol{x}_{V\setminus u}) &= \frac{P(\boldsymbol{x}_{V\setminus u}, x_u)}{P(\boldsymbol{x}_{V\setminus u})} \\ &= \frac{F_\alpha(\boldsymbol{x}_{V\setminus u}) F_\beta(\boldsymbol{x}_{N(u)}, x_u)/Z_V}{\int F_\alpha(\boldsymbol{x}_{V\setminus u}) F_\beta(\boldsymbol{x}_{N(u)}, x_u) dx_u / Z_V} \\ &= \frac{F_\beta(\boldsymbol{x}_{N(u)}, x_u)}{\int F_\beta(\boldsymbol{x}_{N(u)}, x_u) dx_u}, \end{aligned}$$

showing that the conditional probability density of x_u given all the remaining variables $\boldsymbol{x}_{V\setminus u}$ depends only on the variables $\boldsymbol{x}_{N(u)}$ that neigh-

bor the node u on the graph $G = (V, E)$. Therefore, sampling of x_u may be accomplished when the variables $\boldsymbol{x}_{N(u)}$ are known, a fact we shall need in future sections.

The extension of the conditional formula to a set of variables $\boldsymbol{x}_U$ is given by Equation 3.1.

Using the graphical framework we may now say something about the conditional independence structure of the marginal density $P(\boldsymbol{x}_U)$ without the need to know the values of $P(\boldsymbol{x}_U)$; instead, we compute symbolically its worst-case factorization structure. However, we do not know how to use the graphical description of the independence structure to split the variables.

3.2.2 *Conditional independence of a set of variables*

The question arises whether the set $V \setminus U$ had any special properties on the graph $G = (V, E)$ that allowed us to easily compute the conditional probability $P(\boldsymbol{x}_{V\setminus U} \mid \boldsymbol{x}_U)$. We have in fact mentioned this in Section 2.1.3.1, namely that the set $V \setminus U$ forms a so-called *independent set* within G. The fact that all variables in $\boldsymbol{x}_{V\setminus U}$ are conditionally independent given $\boldsymbol{x}_U$ implies that there are no edges between the nodes of $V \setminus U$, which is the definition of an independent set.

The remaining nodes U form the matching *vertex cover*, defined as a set of nodes whose removal from a graph leaves a totally disconnected graph, i.e., a graph with no edges. A vertex cover matches a particular independent set, because — by definition — the complement of an independent set is a vertex cover and *vice versa*.

The property that U forms a vertex cover within the graph while $V\setminus U$ is the matching independent set solves our conundrum. Given the probability $P(\boldsymbol{x}_V)$ we constructed a graph $G = (V, E)$ encoding its conditional independence structure and chose to split V into two subsets, $U \subset V$ forming a vertex cover and the complementary independent set $V\setminus U$. The variables $\boldsymbol{x}_{V\setminus U}$ are conditionally independent given $\boldsymbol{x}_U$, therefore allowing us to efficiently compute the conditional probability $P(\boldsymbol{x}_{V\setminus U} \mid \boldsymbol{x}_U)$; therefore, the variables $\boldsymbol{x}_{V\setminus U}$ may be marginalized, leaving $\boldsymbol{x}_U$ with probability density $P(\boldsymbol{x}_U)$. The conditional independence structure may be obtained from the graph G by rules of marginalization, giving a modified graph $G_U = (U, E_U)$ describing the worst-case conditional independence structure of $P(\boldsymbol{x}_U)$, allowing us to repeat the procedure.

3.3 EXACT COARSENING

The exact coarsening algorithm is based on repeated splitting of the set of nodes into an independent set and the matching vertex cover, followed

by marginalization of the variables contained in the independent set and repeating the algorithm on the remaining variables. Given the original set of variables V and the original dependence graph $G = (V, E)$ we define $V_0 = V$ and $G_0 = G$, then recursively generate a ladder of increasingly coarse graphs.

At step i we have the set of variables V_i and graph $G_i = (V_i, E_i)$. We find a vertex cover $U_i \subset V_i$ that is a proper subset of V_i and define $V_{i+1} = U_i$. The edges of E_{i+1} are defined by stating that two nodes $u, v \in V_{i+1}$ are connected by an edge $(u, v) \in E_{i+1}$ if (i) they were connected by an edge $(u, v) \in E_i$ on the graph G_i or (ii) there exists a removed node $w \in V_i \setminus V_{i+1}$ such that both $(u, w) \in E_i$ and $(v, w) \in E_i$, that is if the nodes u, v shared a common neighbor on the graph G_i. Finally, we let $G_{i+1} = (V_{i+1}, E_{i+1})$, completing the recursion.

Since the original graph G is assumed to be finite and V_{i+1} is always a proper subset of V_i, after a finite number of coarsening steps we will obtain a ladder of lattices

$$V = V_0 \supset V_1 \supset V_2 \supset \ldots \supset V_m$$

with the property that it is computationally feasible to compute the conditional probability $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$ given that the unnormalized marginal probability density $P(\boldsymbol{x}_{V_i})$ is known. This results in Algorithm 3.1.

3.3.1 *Relation of exact coarsening to the LU decomposition*

Algorithm 3.1 shares its graphical structure with another widely known algorithm from linear algebra, the LU factorization of an invertible, symmetrically patterned matrix A. The random variables correspond to dimensions of the vector space, with nodes representing each. If the entry $A_{ij} \neq 0$ then the two nodes $1 \leq i, j \leq n$ are connected by an edge. Performing a step of the LU decomposition corresponds to a marginalization of a variable, completing the similarities.

In the LU algorithm the L and U factors are obtained by Gaussian elimination, with L storing the weights used to eliminate sub-diagonal nonzero entries and U being the upper triangular matrix leftover after performing Gaussian elimination. Therefore, to show the symbolic equivalence of the LU decomposition to marginalization of random variables in a graphical model it suffices to show that a single step of Gaussian elimination is equivalent to the marginalization of a single random variable.

Algorithm 3.1 Exact coarsening algorithm.

function EXACTCOARSENING($G_i = (V_i, E_i)$)
  $V_{i+1} \leftarrow \text{MIS}(G_i)$
  $E_{i+1} \leftarrow \emptyset$
  for all $u \in V_{i+1}$ **do**
    for all $v \in N_{G_i}(u)$ **do**
      if $v \in V_{i+1}$ **then**
        $E_{i+1} \leftarrow E_{i+1} \cup (u, v)$
      end if
      for all $w \in N_{G_i}(v)$ **do**
        if $w \in V_{i+1}$ and $w \neq u$ **then**
          $E_{i+1} \leftarrow E_{i+1} \cup (u, w)$
        end if
      end for
    end for
  end for
  return $G_{i+1} \leftarrow (V_{i+1}, E_{i+1})$
end function

At step i, the matrix A has zeros below the diagonal in all columns to the left of column i. Therefore, by symmetry of the non-zero pattern of A, the variable i is connected to a set of variables $N(i)$, such that for $i < j$ for $j \in N(i)$. Therefore, the only sub-diagonal non-zero entries in column i are A_{ji} for $j \in N(i)$. Consider eliminating one of these non-zeros; denoting the i^{th} row of the matrix A as $\boldsymbol{a}_i$ we obtain an update equation

$$\boldsymbol{a}'_j = \boldsymbol{a}_j - \frac{A_{ji}}{A_{ii}} \boldsymbol{a}_i.$$

As a result, the updated j^{th} row may contain non-zero entries only when either $A_{jk} \neq 0$ or $A_{ik} \neq 0$; therefore, the updated matrix A' will have the variable j connected with all of its original neighbors ($A_{jk} \neq 0$) and all the neighbors of the variable i ($A_{ik} \neq 0$), sans the variable i because by construction the entry $A'_{ji} = 0$. Since all variables $j \in N(i)$ undergo such operation, the variable i is effectively removed from the graph while all variables in $N(i)$ are connected to each another, precisely as was the case with marginalization of a random variable.

It is rather well-known that the LU algorithm applied to sparse matrices causes fill-in, that is the number of non-zero entries in the factors L and U is generally higher than that of the original matrix A. Because the

number and location of the non-zero entries is determined entirely by the graphical structure of the matrix A, the same phenomenon will occur while performing coarsening of a graph describing the conditional independence relations of a probability distribution. As the increased density of the LU factors leads to increased storage needs and slower computations involving the factors, many approaches were developed to combat this phenomenon. Among the most widely known are sparsity-preserving variable permutations (Davis et al., 2004a) and incomplete LU factorizations (Chan and van der Vorst, 1997; Saad, 2003).

Incomplete factorizations make use of the fact that not all non-zero entries are equally important, hoping that ignoring some of the additional entries will have small effect on the LU factors. Multiple strategies were devised (Kershaw, 1978; Meijerink and van der Vorst, 1977; Watts, 1981), including the ILU(k) technique of only allowing fill-in to propagate k times before being rejected (Saad, 2003, p. 296); for example, ILU(0) only allows fill-in at positions of non-zero entries of the original matrix A (Saad, 2003, p. 293), while ILU(1) allows fill-in due to elimination of a non-zero entry that existed in the original matrix A, but not due to elimination of a non-zero entry that was subsequently added as fill-in. Other techniques attempt to ignore certain entries based on magnitude, treating A_{ij} as if it were zero if the magnitude $|A_{ij}| < T$ is less than a threshold (Munksgaard, 1980; Zlatev, Wasniewski, and Schaumburg, 1982).

Sparsity-preserving variable permutations use a different feature of the problem. It was found that the order in which variables are eliminated has a significant effect on the amount of fill-in generated by factorization, leading to algorithms such as COLAMD (Column Approximate Minimum Degree ordering, Davis et al., 2004b). Unfortunately, the task of finding the optimal ordering for a general matrix A is an NP-complete problem (Heggernes et al., 2001), requiring the use of heuristics and approximate algorithms based on simplified assumptions.

While these techniques allow for handling exact LU factorizations of matrices with tens of thousands of variables on desktop computers (Davis, 2004), the costs of computing the marginal distributions respecting the exact dependency graphs generated by equivalent algorithms are far beyond the available computational capabilities. Therefore, it becomes necessary for any practical coarsening algorithm to instead produce approximate dependency graphs, where the number of edges incident upon each node is kept under control. In the following sections we discuss a class of such approximate algorithms and possible ways of obtaining the highest quality approximation possible.

3.4 APPROXIMATE COARSENING

The strict requirement from the exact coarsening was that the nodes V_i must be split into a vertex cover U_i and a matching independent set $V_i \setminus U_i$. The independent set $V_i \setminus U_i$ was subsequently marginalized, while the vertex cover became the next node set in the ladder, $V_{i+1} = U_i$. However, the step involving forming the set of edges E_{i+1} among the nodes of V_{i+1} did not have any strict requirements and may be modified, though at a cost of leading to an approximate conditional independence structure.

Any coarsening algorithm will thus be composed of two parts:

- the choice of a suitable splitting of V_i into a vertex cover and a matching independent set,
- the re-creation of edges between the kept variables V_{i+1}.

The splitting phase must choose a particular pair of an independent set and a matching vertex cover out of many that typically exist. While many optimality criteria may be chosen to make such choice unique, it appears natural to attempt to marginalize as many variables as possible, leading to the requirement that the independent set be a Maximum Independent Set (MIS). The complementary vertex cover becomes then the smallest possible vertex cover, or Minimum Vertex Cover (MVC). The maximum independent set on a Cartesian lattice divides the graph into a checkerboard pattern, recovering the coarsening pattern used by Chorin (2008) while also generalizing naturally to arbitrary graphs.

The latter stage involving reconnecting nodes of V_{i+1} is unfortunately a far less studied and more problem-dependent part of the algorithm. While the exact choice is known, it is unfortunately generally unfeasible and instead we recommend that nodes of V_{i+1} be connected only if they pass a distance criterion. Assuming a metric $\rho : V_{i+1} \times V_{i+1} \to \mathbb{R}$ is defined, we only allow edges to be formed between nodes $u, v \in V_{i+1}$ such that $\rho(u, v) < T_{i+1}$, where T_{i+1} is a user-provided threshold.

The complete approximate graph coarsening algorithm is provided as Algorithm 3.2, whose possible sub-components are discussed in what follows.

3.4.1 *Optimality condition*

The computation of a MIS is one of the most difficult problems of theoretical computer science. Just as the problem of finding the optimal variable ordering, the MIS problem is NP-complete, thus any algorithm solving it in

Algorithm 3.2 Approximate coarsening.

function APPROXIMATECOARSENING($G_i = (V_i, E_i)$, C)
  $V_{i+1} \leftarrow \text{MIS}(G_i)$
  $\rho_{i+1} \leftarrow \min_{u \neq v} \rho(u, v)$ for $u, v \in V_{i+1}$
  $E_{i+1} \leftarrow \emptyset$
  for all $u \in V_{i+1}$ **do**
    for all $v \in V_{i+1}$ **do**
      if $u \neq v$ and $\rho(u, v) \leq C\rho_{i+1}$ **then**
        $E_{i+1} \leftarrow E_{i+1} \cup (u, v)$
      end if
    end for
  end for
  return $G_{i+1} \leftarrow (V_{i+1}, E_{i+1})$
end function

polynomial time will also be able to solve every other NP-complete problem in polynomial time. In fact, the MIS problem is one of the original 21 NP-complete problems of Karp (1972) and there are no known polynomial time algorithms for solving them.

This difficulty necessitates relaxing the optimality condition. Instead of requiring the maximum independent set, we ask for a maximal independent set, that is an independent set that cannot be made larger by adding other nodes. Multiple heuristic algorithms exist and produce maximal independent sets. We recommend the algorithm suggested by Prof. Richard M. Karp in a personal communication, listed as Algorithm 2.3, due to the fact that on Cartesian lattice graphs it closely reproduces the expected checkerboard pattern yet is applicable to arbitrary graphs.

3.4.2 *Reconnecting V_{i+1}*

Consider a metric $\rho : V_{i+1} \times V_{i+1} \to \mathbb{R}$. The natural choice of a reconnecting algorithm is to form an edge (u, v) for every pair of u and v such that

$$\rho(u, v) \leq T_{i+1},$$

where T_{i+1} is a user-prescribed threshold. The threshold should ideally scale with the distance between nodes on the lattice and be automatically adjusted. A good choice is to let T_{i+1} be a multiple of the smallest distance between two nodes in V_{i+1}.

Alternatively, if the dependency graph is highly inhomogeneous the threshold must be adjusted on a per-node basis, requiring an automated determination of a suitable threshold, making it highly problem-dependent.

Note that the reconnecting algorithm must always keep the edges of E_{i+1} being undirected; in other words, if a node u is connected to a node v, then the node v must also be connected to the node u. Failure to do so would lead to significant problems that make it impossible to compute a properly defined marginal density $P(\boldsymbol{x}_{V_{i+1}})$.

3.4.3 *Choice of metric*

Frequently the specification of the original probability density suggests a choice of a natural metric $\rho : V \times V \to \mathbb{R}$, which in turns defines a metric on all coarser lattices through restriction. As an example, the Ising model variables form a Cartesian lattice, thus a metric based on the p-norm may be considered a natural choice.

In situations where no such metric exists one may use either the graphical structure of the problem or quantities obtained directly from the original probability density $P(\boldsymbol{x}_V)$. If the graph $G = (V, E)$ is to be used, one may define $\rho(u, v)$ to be the length of the shortest path connecting the nodes u and v. Efficient algorithms exist and may be used to compute the shortest path distance between individual pairs of nodes (Dijkstra, 1959) or between all pairs at once (Cormen et al., 1990, p. 643–700). The edges forming paths between nodes might be either weighted or unweighted in case of homogeneous distributions. When weights are used, they should be inversely proportional to the coupling strength between the nodes, ensuring that the distance between two strongly interacting variables is smaller than between weekly connected ones.

Alternatively one may be able to extract metric information from the probability distribution by computing correlations between variables, defining

$$\rho(u, v) = 1 \Big/ \operatorname{corr}(x_u, x_v) = \frac{\sigma_{x_u} \sigma_{x_v}}{\mathbb{E}\left[(x_u - \mu_{x_u})(x_v - \mu_{x_u})\right]},$$

where μ_x and σ_x are the average and standard deviation of the variable x, respectively. This approach is especially useful in case of highly inhomogeneous models, where heuristic approaches based on the graph structure alone may not suffice.

EXAMPLE 3.2. Consider a regular Cartesian lattice if size 32×32. We define a natural metric $\rho(u, v)$ as the 2-norm distance between the nodes on the original lattice, assuming the distance between the nearest neighbors is equal to one.

The set of nodes at each step is split by finding an independent set and the matching vertex cover using Algorithm 2.3. The reconnecting algorithm connects together all node pairs such that

$$\rho(u, v) \leq C\rho_{i+1},$$

where $\rho_{i+1} = \min_{u \neq v} \rho(u, v)$ is the smallest distance between distinct nodes on lattice V_{i+1}. The resulting lattice ladders are shown on Figure 3.11 for different values of the distance parameter C. ■

3.5 LATERAL DEPENDENCY GRAPH DENSENING

The results of Figure 3.11 shows clearly that increasing the number of reconnected nodes quickly leads to dense graphs, making the computation of marginal probabilities a costly enterprise. Additionally, the resulting lattices lose the regularity and symmetries existing in the original model, qualities we may wish to preserve during the coarsening process. The least dense variant is therefore frequently an attractive option, even though the significant number of missed dependencies between variables results in low quality of the resulting approximate marginal probabilities. In what follows we will describe a method for selectively densening an initially sparse dependency graph, allowing for performing a sparse coarsening that ensures large cliques are not formed yet attaining good quality.

Consider lattice V_i shown on Figure 3.12a obtained during coarsening of a larger Cartesian lattice and assume that V_i is also a Cartesian lattice, but of size 8×8. The reconnected graph $G_i = (V_i, E_i)$ is very sparse, with edges joining only the nearest neighbors on the lattice. We continue coarsening it, finding a vertex cover $V_{i+1} \subset V_i$ (red nodes) that paints the familiar checkerboard pattern on the lattice G_i. Assume all variables $\boldsymbol{x}_{V_{i+1}}$ are known and we are attempting to sample $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ (yellow nodes) from the conditional probability $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$. Equation 3.1 shows that the conditional probability factors into a product

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}}) \mid \boldsymbol{x}_{V_{i+1}}) = \prod_{V_i \setminus V_{i+1}} \frac{F_u(\boldsymbol{x}_u, \boldsymbol{x}_{V_{i+1}})}{\int F_u(\boldsymbol{x}_u, \boldsymbol{x}_{V_{i+1}}) dx_u}$$

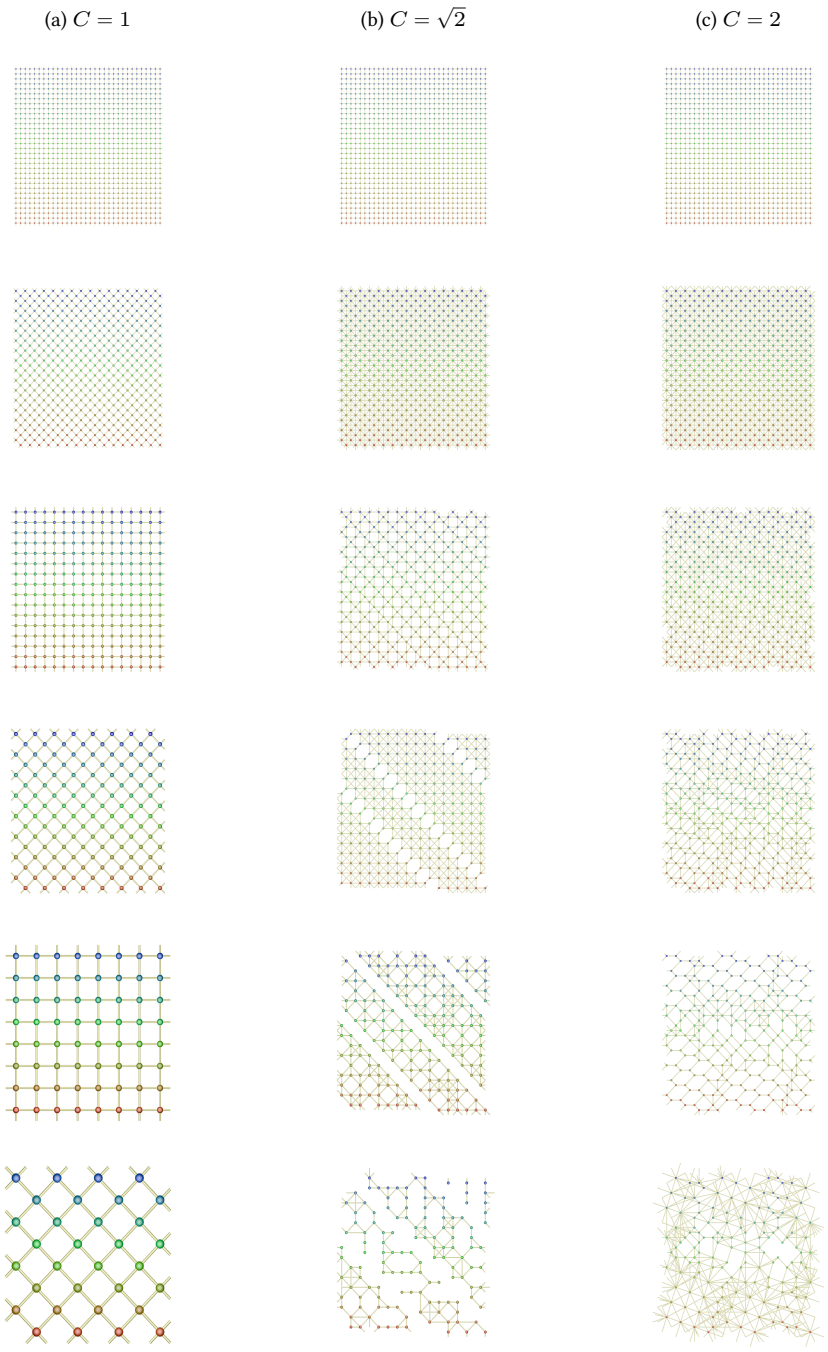


Figure 3.11: Comparison of different distance multiples C used in the reconnecting algorithm. Although higher values of C initially lead to higher density graphs, they inadvertently form very irregular graphs that may become disconnected. This is caused by two neighboring nodes that are very close on the lattice, although the general inter-node distances are much larger.

$$= \prod_{V_i \setminus V_{i+1}} P(x_u \mid \boldsymbol{x}_{V_{i+1}}).$$

Furthermore, using the fact that each x_u is conditionally independent of all other variables given the values of the neighboring variables $\boldsymbol{x}_{N(u)}$,

$$x_u \perp\!\!\!\perp \boldsymbol{x}_{V_i \setminus \bar{N}(u)} \mid \boldsymbol{x}_{N(u)},$$

we see that the conditional probability of x_u depends only on $\boldsymbol{x}_{N(u)}$ rather than on all of $\boldsymbol{x}_{V_{i+1}}$,

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}}) \mid \boldsymbol{x}_{V_{i+1}}) = \prod_{V_i \setminus V_{i+1}} P(x_u \mid \boldsymbol{x}_{N(u)}).$$

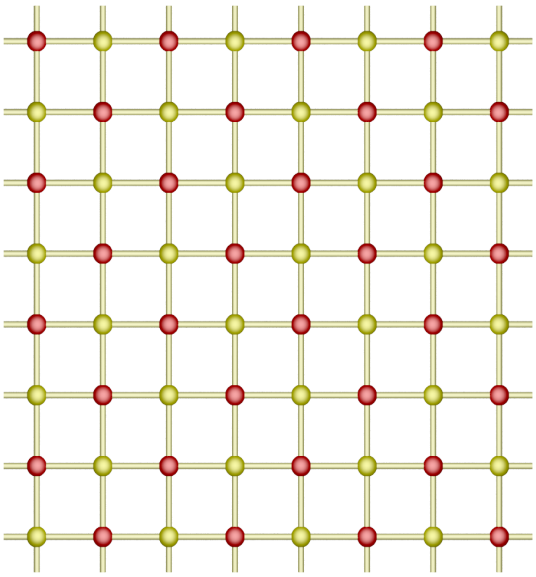
This implicitly assumes that for any $u \in V_i \setminus V_{i+1}$ the set $N(u) \subseteq V_{i+1}$, which is clearly true because V_{i+1} forms a vertex cover in $G_i = (V_i, E_i)$. At least one node of every edge in E_i belongs to V_{i+1}; since for $(u, v) \in E_i$ the node $u \notin V_{i+1}$, it follows that $v \in N(u)$ must belong to V_{i+1}, proving $N(u) \subseteq V_{i+1}$.

3.5.1 *Motivating example*

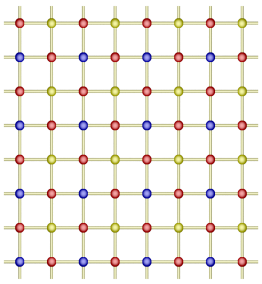
The selective increase in the density of the dependency graph is illustrated by Figure 3.12. The panel 3.12a shows the lattice $G_i = (V_i, E_i)$ with nodes V_{i+1} marked red. Determining the remaining variables $x_{V_i \setminus V_{i+1}}$, marked yellow, requires only knowing the red variables $x_{V_{i+1}}$. Moving to Figure 3.12b, notice that the variables corresponding to nodes marked blue are far enough from each other that they could safely use both the values of the red and yellow variables. In other words, we may add diagonal edges connecting the blue variables and the neighboring yellow variables, forming an increased density graph $G_i' = (V_i, E_i')$, shown on the final panel 3.12c: assuming both the red and yellow variables are known, the blue variables may be sampled using a marginal probability density respecting the denser graph G_i'.

The introduction of the increased density graph requires that we compute a marginal probability density

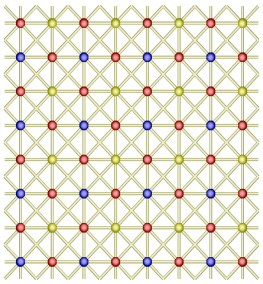
$$P'(\boldsymbol{x}_{V_i}) = F'(\boldsymbol{x}_{V_i}) \Big/ Z'_{V_i}$$



(a)



(b)



(c)

Figure 3.12: Motivation behind selective coarsening: (a) original graph with known variables marked red and the variables to be sampled using yellow, (b) original graph with nodes that remain independent after increasing the edge density marked blue, (c) denser graph used to sample the blue nodes given values of the red and yellow ones.

that respects the dependence graph G'_i, in addition to the computation of the regular marginal

$$P(\boldsymbol{x}_{V_i}) = F(\boldsymbol{x}_{V_i}) \Big/ Z_{V_i}$$

respecting the dependence graph G_i. Denoting the blue nodes $U_i^B \subset V_i \setminus V_{i+1}$ and the yellow variables $U_i^Y \subset V_i \setminus V_{i+1}$, we may write that an improved conditional probability density $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$ may be written using a two-stage formula

In the following, note that $N(u) \subseteq V_{i+1}$, while $N'(v) \subseteq V_{i+1} \cup U_i^Y$ because the blue nodes U_i^B depend on the yellow nodes U_i^Y.

$$\begin{aligned}
&P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) \\
&\quad = P(\boldsymbol{x}_{U_i^Y}, \boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{V_{i+1}}) \\
&\quad = P(\boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{U_i^Y}, \boldsymbol{x}_{V_{i+1}}) P(\boldsymbol{x}_{U_i^Y} \mid \boldsymbol{x}_{V_{i+1}}) \\
&\quad = P(\boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{U_i^Y}, \boldsymbol{x}_{V_{i+1}}) \prod_{u \in U_i^Y} \frac{F_{iu}(x_u, \boldsymbol{x}_{N(u)})}{\int F_{iu}(x_u, \boldsymbol{x}_{N(u)}) dx_u} \\
&\quad = \prod_{v \in U_i^B} \frac{F'_{iv}(x_v, \boldsymbol{x}_{N'(v)})}{\int F'_{iv}(x_v, \boldsymbol{x}_{N'(v)}) dx_v} \prod_{u \in U_i^Y} \frac{F_{iu}(x_u, \boldsymbol{x}_{N(u)})}{\int F_{iu}(x_u, \boldsymbol{x}_{N(u)}) dx_u},
\end{aligned}$$

where the yellow variables $\boldsymbol{x}_{U_i^Y}$ are sampled first using the coarse marginal $P(\boldsymbol{x}_{V_i})$, allowing for sampling of the blue variables $\boldsymbol{x}_{U_i^B}$ using the denser marginal $P'(\boldsymbol{x}_{V_i})$. The formula for $P(\boldsymbol{x}_{U_i^Y} \mid \boldsymbol{x}_{V_{i+1}})$ is obtained trivially because $\boldsymbol{x}_{U_i^Y} \perp\!\!\!\perp \boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{V_{i+1}}$, thus

$$P(\boldsymbol{x}_{U_i^Y}, \boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{V_{i+1}}) = P(\boldsymbol{x}_{U_i^Y} \mid \boldsymbol{x}_{V_{i+1}}) P(\boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{V_{i+1}}),$$

by definition. In the two-stage formula we simply replace $P(\boldsymbol{x}_{U_i^B} \mid \boldsymbol{x}_{V_{i+1}})$ with a more accurate approximation that uses a denser dependency graph G'_i.

3.5.2 *General algorithm*

While the two-step algorithm already improves upon the sparse dependency graph $G_i = (V_i, E_i)$, the above algorithm may be repeated recursively, leading to a lateral sequence of increasingly dense dependency graphs. Therefore, we leave the color-coded notation and instead introduce a top index ^{j} to denote the depth of the recursion.

We begin with the sparse dependency graph $G_i = (V_i, E_i)$, defining $G_i^0 = (V_i, E_i^j) = (V_i, E_i)$. Note that the set of nodes does not change during the lateral recursion, therefore we define a second set of nodes $U_i^0 = V_i \setminus V_{i+1}$; in general, the set U_i^j will contain an independent set under the graph G_i^j, thus the variables in $U_i^j \setminus U_i^{j+1}$ shall be sampled from a conditional probability computed using the marginal density $P^j(\boldsymbol{x}_{V_i})$ respecting the dependency graph G_i^j.

Assuming the graph $G_i^j = (V_i, E_i^j)$ and set U_i^j are known, we seek to define the successors G_i^{j+1} and U_i^{j+1} in terms of G_i^j and U_i^j. The set of nodes of G_i^{j+1} remains unchanged from G_i^j, however the set $E_i^j \subset E_i^{j+1}$ should be enriched by adding edges between more distant variables; this may be accomplished by running the reconnecting algorithm of Section 3.4.2 with varying distance thresholds C_j, such that $C_j < C_{j+1}$. Specification of E_i^{j+1} thus completes the definition of $G_i^j = (V_i, E_i^j)$.

Define H_i^{j+1} as a subgraph of G_i^{j+1}, denoted $G_i^{j+1}|_{U_i^j}$, obtained by restricting the set of nodes to U_i^j and keeping only edges of E_i^{j+1} between the nodes of the restricted set. The set U_i^{j+1} is then defined as a maximum independent set within the graph H_i^{j+1},

$$U_i^{j+1} = \textsc{MaximumIndependentSet}(H_i^{j+1}).$$

In a practical algorithm the optimality condition is relaxed, requiring only a maximal rather than maximum independent set, a weaker condition (cf. Section 3.4.1). Due to the finite size of V_i, the repeated application of this recurrence will generate a lateral sequence of lattices G_i^j together with node sets

$$V_i \setminus V_{i+1} = U_i^0 \supset U_i^1 \supset U_i^2 \supset \ldots \supset U_i^m$$

such that for any $u, v \in U_i^j$ the variables x_u and x_v are conditionally independent within the marginal density $P^j(\boldsymbol{x}_{V_i})$ given $\boldsymbol{x}_{V_i \setminus U_i^j}$.

The nested nature of this lateral sequence implies that all variables $\boldsymbol{x}_{U_i^j}$ could be sampled with the use of conditional probability derived from $P^j(\boldsymbol{x}_{V_i})$, however the variables $\boldsymbol{x}_{U_i^{j+1}}$ could be sampled using a still denser, more accurate probability $P^{j+1}(\boldsymbol{x}_{V_i})$. Therefore, the possibly most accurate sampling order is to sample variables corresponding to nodes $U_i^0 \setminus U_i^1$ using the conditional probability derived from $P^0(\boldsymbol{x}_{V_i})$, followed by the sampling of variables corresponding to nodes $U_i^1 \setminus U_i^2$, *et cetera*, until the last set of nodes U_i^m.

The complete description of the lateral graph densening technique is provided as Algorithm 3.3 together with the brief sampling Algorithm 3.4.

Example 3.3. We will illustrate Algorithm 3.3 using the previously described example of an 8×8 Cartesian lattice $G_i = (V_i, E_i)$, containing only edges between nearest neighbors. The set V_{i+1} takes the form of a checkerboard pattern, denoted on Figure 3.13 using blue nodes. Progressing from Figure 3.13a toward 3.13f we repeatedly increase the density of edges by connecting all pairs of nodes separated by at most $C_0 = 1$,

Algorithm 3.3 Lateral densening.

procedure LATERALDENSENING($G_i = (V_i, E_i)$, V_{i+1}, C_j)
  $U_i^0 \leftarrow V_i \setminus V_{i+1}$
  $G_i^0 \leftarrow G_i$
  $j \leftarrow 0$
  while $|U_i^j| > 1$ **do**
    $E_i^{j+1} \leftarrow$ RECONNECT(V_i, C_{j+1})
    $G_i^{j+1} \leftarrow (V_i, E_i^{j+1})$
    $H_i^{j+1} \leftarrow$ SUBGRAPH(G_i^{j+1}, U_i^j)
    $U_i^{j+1} \leftarrow$ MAXIMUMINDEPENDENTSET(H_i^{j+1})
    $j \leftarrow j + 1$
  end while
  $m \leftarrow j$
end procedure

Algorithm 3.4 Sampling the variables $\boldsymbol{x}_{V_i}$ given $\boldsymbol{x}_{V_{i+1}}$ under lateral densening.

function LATERALDENSENINGSAMPLING($\boldsymbol{x}_{V_{i+1}}$, U_i^j, $P^j(\boldsymbol{x}_{V_i})$, m)
  for $i = 0 \rightarrow m - 1$ **do**

$$x_{U_i^j \setminus U_i^{j+1}} \sim \frac{P^j(\boldsymbol{x}_{V_i})}{\int P^j(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i \setminus V_{i+1}}}$$

  end for

$$x_{U_i^m} \sim \frac{P^m(\boldsymbol{x}_{V_i})}{\int P^m(\boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i \setminus V_{i+1}}}$$

  return $x_{V_i} \leftarrow \boldsymbol{x}_{V_{i+1} \bigcup_{j=0}^{m} U_i^j}$
end function

$C_1 = \sqrt{2}$, $C_2 = 2$, $C_3 = \sqrt{5}$, $C_4 = \sqrt{8}$ and $C_5 = 3$. The sets V_{i+1}, $U_i^0 = V_i \setminus V_{i+1}$, U_i^2, U_i^3 and U_i^4 are marked with colors ranging from red, through yellow to blue.

Figure 3.13a shows the first densening step, where we produce a denser set of edges E_i^1 containing both nearest neighbor and diagonal connections. A maximum independent set is found within the graph G_i^1 restricted to nodes of $U_i^0 = V_i \setminus V_{i+1}$ and marked with a different color, becoming the set U_i^1. The process repeats itself, producing sets U_i^j of decreasing size.

Note the special transition between U_i^2 and U_i^3 shown on Figure 3.13c: we find that $U_i^3 = U_i^2$. This takes place because the set of edges connect-

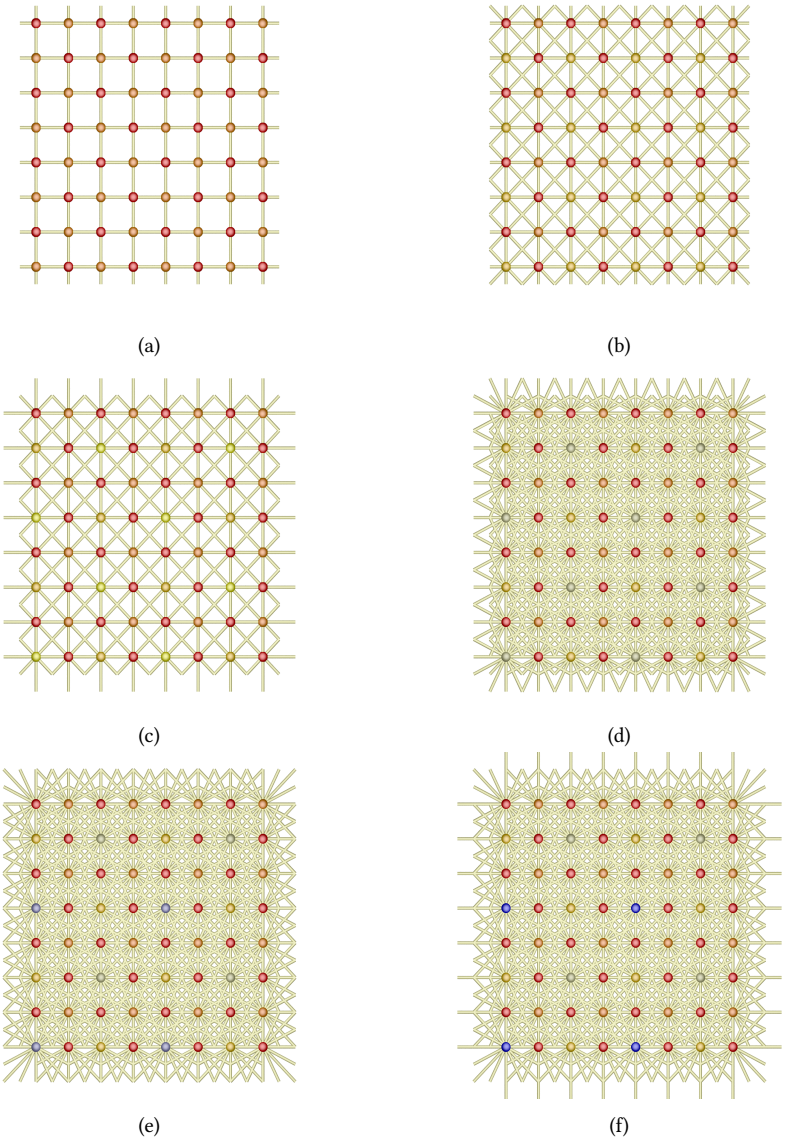


Figure 3.13: Results of the lateral densening algorithm. Figure (a) shows the original, sparse graph, where the yellow variables are sampled given the values of the red variables. We increase the density of edges, i.e., the width of the allowed interactions between the variables, creating subsequently denser graphs in Figures (b) through (f). The colors show order of sampling, beginning with red (variables sampled on prior lattices) through yellow and blue.

ing all nodes within distance $C_3 = \sqrt{5}$ of each other is no more restrictive than that with $C_2 = 2$. As a result one does not need to worry about un-

used stages in the algorithm, because if the increase in edge density is found to be too small the algorithm will automatically skip the unnecessary increase. Similarly, the set $U_i^5 = U_i^4$, because the increase from $C_4 = \sqrt{8}$ to $C_5 = 3$ does not create edges between the variables $\boldsymbol{x}_{U_i^4}$.

While each step of the algorithm improves the quality of the approximation used to sample variables, the set of affected variables is steadily decreasing. Initially we face the task of sampling 32 variables using the sparse probability density $P^0(\boldsymbol{x}_{V_i})$, however by performing the first step of the algorithm we produce an improved probability density that might be used to sample 16 of those variables. In the later steps, an improvement is brought upon 8 and 4 variables, respectively, reaching a point of diminishing returns.

When sampling according to Algorithm 3.4, the variables will be sampled in the order suggested by node coloring, with the red nodes being sampled first (assumed known when starting the algorithm), continuing through yellow nodes and finishing with blue nodes. ■

3.6 ACYCLIC STRUCTURE

The initial graph $G = (V, E)$ that encodes the conditional independence structure of $P(\boldsymbol{x}_V)$ is an undirected graph, where each edge can be traversed in both directions. Therefore, it contains cycles: paths of dependence between random variables that preclude efficient sampling due to circular dependencies. Consider the two-dimensional Ising model. The spin $x_{i,j}$ to be sampled requires the knowledge of the value of the neighboring spins, including the spin $x_{i+1,j}$; since its value is unknown by proxy it requires the knowledge of the value of the neighbors of $x_{i+1,j}$, including $x_{i+1,j+1}$. Proceeding further, sampling the unknown value of $x_{i+1,j+1}$ requires the knowledge of the value of $x_{i,j+1}$, which then finally requires the value of $x_{i,j}$. The resulting unbreakable chain of dependencies makes it impossible to sample one of the variables without sampling all others simultaneously. However, when the probability density $P(\boldsymbol{x}_V)$ is written in the acyclic form as a product of conditional probabilities

$$\begin{aligned} P(\boldsymbol{x}_V) = P(\boldsymbol{x}_{V_m}) P(\boldsymbol{x}_{V_{m-1} \setminus V_m} \mid \boldsymbol{x}_{V_m}) \\ \times P(\boldsymbol{x}_{V_{m-2} \setminus V_{m-1}} \mid \boldsymbol{x}_{V_{m-1}}) \dots P(\boldsymbol{x}_{V_0 \setminus V_1} \mid \boldsymbol{x}_1), \end{aligned}$$

those circular dependencies disappear. Unfortunately, this structure cannot be represented by an undirected graph due to the fact that undirected edges lack directionality. Instead, we introduce a directed graph to encode the conditional probabilities. If the PDF $P(\boldsymbol{x}_V)$ contains a term

$P(x_v \mid \boldsymbol{x}_U)$ with $u \subset U$, then we say that there exists a directed edge, or arc, $(\overrightarrow{u,v}) \in A$, the set of arcs. Together, the node set V and the directed edge/arc set A form a directed graph $D = (V, A)$, also called a *digraph*. As the notion of neighbors $N(v)$ of a node v is imprecise in the context of directed edges, we instead introduce the sets of direct predecessors $N_p(v)$ and direct successors $N_s(v)$; in the example of an edge $(\overrightarrow{u,v})$, the node u is the direct predecessor of v, while v is the direct successor.

Using this definition we find that the conditional probabilities $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$ may be encoded in a straightforward manner. Since the conditional factorizes according to

$$P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}}) = \prod_{V_i \setminus V_{i+1}} P(x_v \mid \boldsymbol{x}_{N(v)}),$$

with the neighborhood understood in the sense of the undirected graph $G_i = (V_i, E_i)$, we see that the directed graph D_i encoding this conditional probability is made of the nodes of V_i and directed edges pointing from the neighbors $N(v)$ toward v; therefore, $N_p(v) = N(v) \subset V_{i+1}$ and the conditional probabilities may be written as $P(x_v \mid \boldsymbol{x}_{N_p(v)})$. For visualizations of example directed graphs see Figure 3.14.

When multiple conditional probabilities are included we obtain a the complete directed graph $D = (V, A)$, defined as the union of the individual graphs D_i with

$$V = V_m \cup \bigcup_{i=0}^{m-1} V_i \setminus V_{i+1} \text{ and } A = \bigcup_{i=0}^{m-1} A_i,$$

that encodes the acyclic form of $P(\boldsymbol{x}_V)$. The word acyclic in the name of the acyclic Monte Carlo comes from the structure of the digraph D, namely the fact that it does not contain cycles. Graphs of this kind are known as Directed Acyclic Graphs (DAGs) and occur frequently in applications that involve the notion of dependence, e.g., scheduling of interdependent tasks, software dependency graphs, parallelization of algorithms.

A feature of DAGs that will be useful to us is the fact that each DAG induces a partial order on its nodes, known in computer science as the topological order. For two nodes $u, v \in V$ we write $u \leq v$ if and only if there exists a directed path from u to v, i.e., if u is a predecessor of v. If neither $u \leq v$ nor $v \leq u$ the nodes u, v are said to be *incomparable*, because the ordering cannot distinguish between them yet they may not be equal.

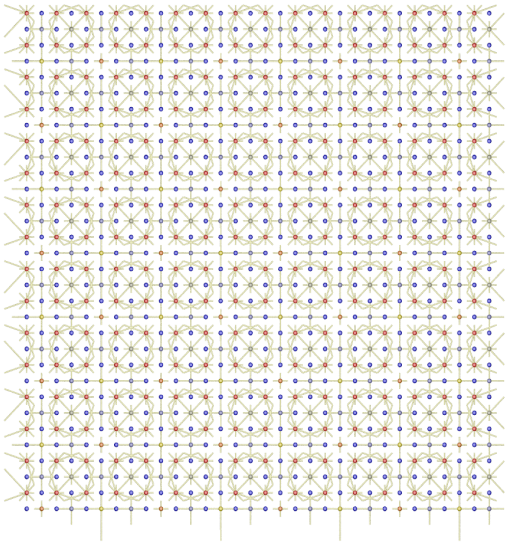


Figure 3.14: Example of a directed graph $D_i = (V_i, A_i)$ encoding the conditional probability $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$. Directed edges are visualized by painting a line emanating from $(\overrightarrow{u,v})$ the predecessor $u \in N_p(v)$ toward the successor node v and reaching half the distance between the two nodes.

Applying topological order to the DAG D defined earlier we obtain

$$V_m \leq V_{m-1} \setminus V_m \leq V_{m-2} \setminus V_{m-1} \leq \ldots \leq V_0 \setminus V_1,$$

which is the order in which we may sample the variables using the conditional probabilities. Additionally, for $u, v \in V_i \setminus V_{i+1}$ neither $u \leq v$ nor $v \leq u$, therefore the nodes of $V_i \setminus V_{i+1}$ are incomparable and cannot be ordered. This represents the fact that all variables in $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ are conditionally independent of each other given the variables $\boldsymbol{x}_{V_{i+1}}$ and may be sampled independently of each other in any order, as long as the values of $\boldsymbol{x}_{V_{i+1}}$ are already known.

Similarly to the way the conditional probabilities $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$ are derived from the marginals $P(\boldsymbol{x}_{V_i})$, the digraphs $D_i = (V_i, A_i)$ and their union $D = (V, A)$ are constructed from the undirected graphs G_i. More precisely, the directed edges $(\overrightarrow{u,v}) \in A_i$ of the digraph D_i are defined using the undirected edges $(u, v) \in E_i$ of G_i, with the direction pointing from $u \in V_{i+1}$ toward $v \in V_i \setminus V_{i+1}$. Therefore the digraph $D =$

(V, A) contains all the conditional independence information contained by the collection of graphs G_i and additionally encodes the precedence among its nodes.

This fact makes the digraph D very useful both from a theoretical point of view and as an element of a practical software implementation. From the theoretical standpoint, the digraph is the glue connecting the seemingly disconnected lattices G_i and explaining the benefits of the entire methodology. Equally importantly the fact that the dependency information is encoded in a single structure of relative simplicity is of tremendous help, because independently of how complex the coarsening algorithm might be, its final product is the digraph D.

3.7 DISCUSSION

We have described how an arbitrary probability density $P(\boldsymbol{x}_V)$ may be described using the language of graphical models. We used this framework to study the conditional independence relations between the variables of $\boldsymbol{x}_V$ by encoding them using a dependency graph $G = (V, E)$. We showed how marginalization affects the dependencies between variables and used the dependency graph to follow those changes without the need of computing the marginal probability distributions.

We used these developments to construct a class of algorithms for computing symbolically a ladder of nested subsets V_i and the related dependency graphs $G_i = (V_i, E_i)$, such that the knowledge of the marginal density $P(\boldsymbol{x}_{V_i})$ allows for efficient computation of the conditional probability density $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$. The variables may therefore be sampled using the acyclic form of $P(\boldsymbol{x}_V)$. Unfortunately, the exact algorithm for computing the dependency graph is not feasible due to the extreme computational cost of computing exact marginal distributions. Instead, the described class of algorithms performs an approximate calculation, requiring the tuning of various components to suit the problem at hand. To simplify the at times complex description, we provide below a suggested algorithm that should be used as an initial choice.

3.7.1 *Recommended algorithm*

There are four parts that need to be combined in order to construct a working coarsening algorithm:

- independent set algorithm,
- reconnecting algorithm,

- stopping criterion,
- lateral densening,

with the last one being optional.

The choice of the algorithm for finding an independent set is fairly clear, with the Algorithm 2.3 due to Prof. Richard M. Karp being both fast, relatively straightforward and able to reproduce the expected results in the case of Cartesian lattices.

The reconnecting algorithm is used to decide which nodes of the newly found set of nodes V_{i+1} should be connected by edges. In other words, it attempts to decide which random variables in $\boldsymbol{x}_{V_{i+1}}$ should affect each other. While the vast literature on renormalization did not consider the problem from the point of view of graphical models (Brandt and Ron, 2001b; Chorin, 2008), all published papers unwittingly chose to connect nodes based on a distance criterion scaled by the smallest distance between nodes, with edges formed between nodes $u, v \in V_{i+1}$ such that

$$\rho(u, v) \leq C_{i+1} \min_{V_{i+1}} \rho(a, b).$$

Therefore, we recommend that the same condition be used, with $\rho : V \times V \to \mathbb{R}$ being the *natural metric* typically associated with the problem at hand, such as a p-metric for Cartesian latices. In cases where no such metric exists, we recommend using the shortest path distance between nodes u and v as computed on the original graph $G = (V, E)$.

The stopping criterion refers to the choice of the final lattice V_m, at which point the coarsening procedure ends with a marginal density $P(\boldsymbol{x}_{V_m})$. Due to the fact that the graph $G_m = (V_m, E_m)$ is not acyclic, we recommend to end coarsening at a point when only a single variable remains. This particular choice greatly simplifies the algorithm because the state $\boldsymbol{x}_{V_m}$ may be determined by sampling a single variable and the computation of the marginal distribution $P(\boldsymbol{x}_{V_m})$ is frequently particularly straightforward.

Finally, the lateral densening may or may not be included. While it does complicate the code due to the extra lateral recursion, in addition to the already existing vertical one, we recommend that it is included in any serious implementation. The costs associated with the need to compute multiple marginal densities $P^j(\boldsymbol{x}_{V_i})$ may be reduced to those of computing only one such density, as explained in the later chapters. Additionally, there exists a synergy with the particle filtering algorithm described in Chapter 5, allowing one to reuse these marginal densities to further improve sampling quality. Finally, the implementation difficulties may be

solved by requesting that the entire coarsening algorithm produce a Directed Acyclic Graph (DAG) $D = (V, A)$ rather than a lattice of graphs $G_i = (V_i, E_i)$, thus hiding the complexity of the coarsening procedure. We recommend that one or two passes of the lateral densening be performed, as further improvements are limited to affect no more than $1/16$ of the variables in $\boldsymbol{x}_{V_i}$.

4

MARGINALIZATION

In the previous chapter, we studied the marginal density

$$P(\boldsymbol{x}_U) = \int P(\boldsymbol{x}_U, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U} \tag{4.1}$$

from the point of view of its graphical structure. We have found that the conditional independence structure, thus also the factorization, of the marginal probability density can be read from the dependency graph $G_U = (U, E_U)$ obtained from the dependency graph $G = (V, E)$ describing the original probability distribution $P(\boldsymbol{x}_V)$. For $u, v \in U$, the fact that there is no edge connecting the two nodes directly, i.e. $(u, v) \notin E_U$, implies that the variables x_u and x_v are conditionally independent given the remaining variables $\boldsymbol{x}_{U\setminus\{u,v\}}$, and can be written as

$$(u, v) \notin E_U \Longrightarrow x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{U\setminus\{u,v\}}.$$

This is equivalent to the fact that the probability density $P(\boldsymbol{x}_U)$ factorizes according to

$$P(\boldsymbol{x}_U) = \frac{1}{Z_U} F_u\left(x_u, \boldsymbol{x}_{U\setminus\{u,v\}}\right) F_v\left(x_v, \boldsymbol{x}_{U\setminus\{u,v\}}\right).$$

We will find useful a corollary of the above, namely that for any $u \in U$, the set of neighbors $N(u)$ in the graph $G_U = (U, E_U)$ shields the variable x_u from the influence of the remaining variables. More rigorously, we may write

$$P(\boldsymbol{x}_U) = \frac{1}{Z_U} F_u(x_u, \boldsymbol{x}_{N(u)}) F_\alpha(\boldsymbol{x}_{U\setminus u}) \tag{4.2}$$

splitting the probability into two parts, with only $F_u(x_u, \boldsymbol{x}_{N(u)})$ dependent on x_u. Armed with this observation, we move on to the main topic of this chapter: the computation of an approximation to the marginal density $P(\boldsymbol{x}_U)$ for any $U \subseteq V$.

Computing the marginal probability $P(\boldsymbol{x}_U)$ from the definition (Equation 4.1) is a futile enterprise in most situations, because the integral involved is of very high dimensionality and the integrand is extremely com-

plicated. Additionally, it would require re-computing the expensive integral for every value of $\boldsymbol{x}_U$, further increasing the computational costs.

An entirely different approach is needed and comes from the work of Chorin, Hald, and Kupferman (2000) on optimal prediction in the face of unknown data (Chorin, 2003; Chorin, Hald, and Kupferman, 2000, 2002; Chorin and Stinis, 2005). They noticed that the logarithmic derivative of a probability density becomes an expected value, a quantity much easier to compute approximately than the marginal probability density. Chorin (2003) expanded upon this observation, creating the fast marginalization method (Chorin, 2008; Okunev, 2005).

Because of the differences between the cases of distributions defined over continuous and discrete variables, we will discuss them separately, beginning with the continuous case.

4.1 THE CASE OF CONTINUOUS VARIABLES

Consider a variable x_u for $u \in U \subseteq V$. By definition, the marginal probability density $P(\boldsymbol{x}_U)$ is

$$P(\boldsymbol{x}_U) = P(x_u, \boldsymbol{x}_{U\setminus u}) = \int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}.$$

Make the mild assumption that $P(\boldsymbol{x}_U) > 0$. This allows us to define the Hamiltonian $W(\boldsymbol{x}_U)$ associated with the probability distribution $P(\boldsymbol{x}_U)$ by

$$P(\boldsymbol{x}_U) = \exp(W(\boldsymbol{x}_U)) \Big/ Z_U.$$

Substituting it into the definition of the marginal density $P(\boldsymbol{x}_U)$ yields

$$\exp(W(\boldsymbol{x}_U)) \Big/ Z_U = \int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}.$$

We apply the logarithmic derivative $\left(\partial \Big/ \partial x_u\right) \ln$ to both sides,

$$\begin{aligned} &\frac{\partial}{\partial x_u}\Big[W(x_u, \boldsymbol{x}_{U\setminus u}) - \ln Z_U\Big] \\ &\qquad = \frac{\partial}{\partial x_u}\left[\ln\left(\int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}\right)\right], \end{aligned}$$

obtaining

$$\frac{\partial W(x_u, \boldsymbol{x}_{U\setminus u})}{\partial x_u} = \frac{\dfrac{\partial}{\partial x_u}\displaystyle\int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}}{\displaystyle\int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}}.$$

Simplifying further using Equation 4.2, we have

$$\begin{aligned}
\frac{\partial W(x_u, \boldsymbol{x}_{U\setminus u})}{\partial x_u} &= \frac{\dfrac{1}{Z_V}\displaystyle\int \frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} F_\alpha(\boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}}{\displaystyle\int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}} \\
&= \frac{\displaystyle\int \frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} \frac{P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})}{F_u(x_u, \boldsymbol{x}_{N(u)})} d\boldsymbol{x}_{V\setminus U}}{\displaystyle\int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}} \\
&= \mathbb{E}\left[\frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} \Big/ F_u(x_u, \boldsymbol{x}_{N(u)}) \,\middle|\, \boldsymbol{x}_U \right],
\end{aligned}$$

which is a generalization of the primary result of Chorin (2003, 2008) and Okunev (2005).

Henceforth we always use the Lagrange notation $f'(x_u)$ to imply a derivative with respect to x_u. All other derivatives will be using the Leibnitz notation $\partial f(x_u)/\partial x_u$ to avoid confusion. The graphical structure of the marginal probability distribution $P(\boldsymbol{x}_U)$ discussed in Chapter 3 states which variables among $\boldsymbol{x}_U$ the derivative $W'(x_u, \boldsymbol{x}_{U\setminus u})$ may depend on. We obtain the equivalent results here using an algebraic argument. Compute the derivative $W'(x_u, \boldsymbol{x}_{U\setminus u})$ by taking a logarithmic derivative of the marginal probability distribution $P(\boldsymbol{x}_U)$. Using a splitting analogous to that of Equation 4.2, we obtain

$$\begin{aligned}
W'(x_u, \boldsymbol{x}_{U\setminus u}) &= \frac{\partial}{\partial x_u} \ln P(\boldsymbol{x}_U) \\
&= \frac{\partial}{\partial x_u} \ln \left[\frac{1}{Z_U} F_u^U(x_u, \boldsymbol{x}_{N(u)}) F_\alpha^U(\boldsymbol{x}_{N(u)}, \boldsymbol{x}_{U\setminus \bar{N}(u)}) \right] \\
&= \frac{\partial}{\partial x_u} \ln F_u^U(x_u, \boldsymbol{x}_{N(u)})
\end{aligned}$$

Therefore, $W'(x_u, \boldsymbol{x}_{U\setminus u})$ is a function of only x_u and the variables $\boldsymbol{x}_{N(u)}$ that are in the neighborhood of the node u on the dependency

graph $G_U = (U, E_U)$, a result consistent with the graphical arguments of the previous chapter. Thus, we may write $W'(x_u, \boldsymbol{x}_{N(u)})$ instead of $W'(x_u, \boldsymbol{x}_{U\setminus u})$.

This observation affects the choice of basis described in Section 4.1.3. If $W'(\boldsymbol{x}_U)$ were a function of variables beyond $\boldsymbol{x}_{\bar{N}(u)}$, the probability distribution $P(\boldsymbol{x}_U)$ would not have a conditional independence structure consistent with the graph $G_U = (U, E_U)$. Therefore, the approximation of $W'(\boldsymbol{x}_U)$ must respect these constraints.

4.1.1 *Projection*

Let the vector spaces X_V and X_U be the spaces of functions of $\boldsymbol{x}_V$ and $\boldsymbol{x}_U$, respectively. We wish to find an approximation of $W'(\boldsymbol{x}_{\bar{N}(u)}) \in X_U$ within the subspace $X_\phi \leq X_U$ of functions spanned by a basis ϕ,

$$f(\boldsymbol{x}_U) \in X_\phi \quad \Rightarrow \quad f(\boldsymbol{x}_U) = \sum_{i=1}^{K} c_i \phi_i(\boldsymbol{x}_U),$$

where $K = \dim X_\phi$ is the size of the basis and the dimension of the subspace X_ϕ. We leave the discussion of the particular choice of ϕ until Section 4.1.3.

We want to find the best approximation in the least squares sense. We define an inner product through

$$\langle f, g \rangle = \int f(\boldsymbol{x}_U) g(\boldsymbol{x}_U) \frac{P(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} d\boldsymbol{x}_U, \tag{4.3}$$

where $Q(\boldsymbol{x}_U) > 0$ is a weight discussed in Section 4.1.2. We define the distance between $W'(\boldsymbol{x}_{\bar{N}(u)}) \in X_U$ and its approximation $\hat{W}'(\boldsymbol{x}_{\bar{N}(u)}) \in X_\phi$ to be

$$\begin{aligned}
\rho(W', \hat{W}') &= ||W' - \hat{W}'||^2 \\
&= \left\langle W' - \hat{W}', W' - \hat{W}' \right\rangle^2 \\
&= \left\langle W' - \sum_{i=1}^{K} c_i \phi_i, W' - \sum_{i=1}^{K} c_i \phi_i \right\rangle^2 .
\end{aligned}$$

This is minimal when

$$\frac{\partial \rho(W', \hat{W}')}{\partial c_i} = \frac{\partial}{\partial c_i} \left\langle W' - \sum_{j=1}^{K} c_j \phi_j, W' - \sum_{j=1}^{K} c_j \phi_j \right\rangle^2$$

$$= -2 \left\langle W' - \sum_{j=1}^{K} c_j \phi_j, \phi_i \right\rangle$$
$$= -2 \left[\langle W', \phi_i \rangle - \sum_{j=1}^{K} c_j \left\langle \phi_j, \phi_i \right\rangle \right],$$

are equal to zero for each i; thus, we obtain a set of K linear equations

$$\sum_{j=1}^{K} c_j \langle \phi_j, \phi_i \rangle = \langle W', \phi_i \rangle.$$

Writing $\boldsymbol{c} = (c_1, c_2, \dots, c_K)$ and $\boldsymbol{b} = (b_1, b_2, \dots, b_K)$ we obtain a linear system $A\boldsymbol{c} = \boldsymbol{b}$, where $A = (A_{ij})$, $A_{ij} = \langle \phi_i, \phi_j \rangle$, is the $K \times K$ symmetric positive definite Gram matrix for the basis ϕ and $b_i = \langle W', \phi_i \rangle$.

We compute the terms A_{ij} and b_i from the definition of the inner product. The simpler term of the two, A_{ij} becomes

$$\begin{aligned} A_{ij} &= \langle \phi_i, \phi_j \rangle \\ &= \int \phi_i(\boldsymbol{x}_U) \phi_j(\boldsymbol{x}_U) \frac{P(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} d\boldsymbol{x}_U \\ &= \int \frac{\phi_i(\boldsymbol{x}_U) \phi_j(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} P(\boldsymbol{x}_V) d\boldsymbol{x}_V \\ &= \mathbb{E}\left[\frac{\phi_i(\boldsymbol{x}_U) \phi_j(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} \right]. \end{aligned}$$

The crucial step in the derivation is the change from an expected value with respect to $P(\boldsymbol{x}_U)$ to an expected value with respect to the original probability distribution $P(\boldsymbol{x}_V)$. This allows us to approximate the inner product without the knowledge of the marginal density $P(\boldsymbol{x}_U)$, but simply through sampling states $\boldsymbol{x}_V$ from the original probability distribution $P(\boldsymbol{x}_V)$.

The projection vector $\boldsymbol{b}$ presents us with a bit more difficulty.

$$\begin{aligned} b_i &= \langle W', \phi_i \rangle \\ &= \int W'(\boldsymbol{x}_{\bar{N}(u)}) \phi_i(\boldsymbol{x}_U) \frac{P(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} d\boldsymbol{x}_U \end{aligned}$$

$$= \int \phi_i(\boldsymbol{x}_U) \frac{\displaystyle\int \frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} \frac{P(\boldsymbol{x}_V)}{F_u(x_u, \boldsymbol{x}_{N(u)})} d\boldsymbol{x}_{V\setminus U}}{\displaystyle\int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V\setminus U}} \frac{P(\boldsymbol{x}_U)}{Q(\boldsymbol{x}_U)} d\boldsymbol{x}_U.$$

The denominator is the marginal density of $\boldsymbol{x}_U$,

$$\int P(\boldsymbol{x}_V) d\boldsymbol{x}_{V\setminus U} = P(\boldsymbol{x}_U),$$

so that

$$\begin{aligned} b_i &= \int \frac{\phi_i(\boldsymbol{x}_U)}{F_u(x_u, \boldsymbol{x}_{N(u)}) Q(\boldsymbol{x}_U)} \frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} P(\boldsymbol{x}_V) d\boldsymbol{x}_V \\ &= \mathbb{E}\left[\frac{\phi_i(\boldsymbol{x}_U)}{F_u(x_u, \boldsymbol{x}_{N(u)}) Q(\boldsymbol{x}_U)} \frac{\partial F_u(x_u, \boldsymbol{x}_{N(u)})}{\partial x_u} \right]. \end{aligned}$$

Since this is an expected value with respect to the original probability density $P(\boldsymbol{x}_V)$, sampling from the original model allows us to compute the terms of the least squares equation.

The resulting linear system $A\boldsymbol{c} = \boldsymbol{b}$ can be solved using the Cholesky decomposition $A = LL^{\mathrm{T}}$, the QR decomposition $A = QR$ or the Singular Value Decomposition $A = U\Sigma V^{\mathrm{T}}$. Practice shows that for strongly coupled systems, such as the Ising model with large μ, the matrix A is frequently numerically singular. The QR or SV decompositions help handle these degenerate cases gracefully. Using the solution vector $\boldsymbol{c}$ we obtain a series

$$\hat{W}'(\boldsymbol{x}_{\bar{N}(u)}) = \sum_{i=1}^{K} c_i \phi_i(\boldsymbol{x}_{\bar{N}(u)}),$$

which can be integrated to obtain an approximation of the Hamiltonian $W(\boldsymbol{x}_U)$, and hence of the marginal probability density $P(\boldsymbol{x}_U)$.

4.1.2 *The weight factor $Q(\boldsymbol{x}_U)$*

Throughout the derivations we have kept track of the factor $1/Q(\boldsymbol{x}_U)$ that was present in the inner product definition. The motivation for introducing $Q(\boldsymbol{x}_U)$ comes from the observation that the inner product used in the least squares approximation above is inherently biased. The weight is pro-

portional to $P(\boldsymbol{x}_U)$ and typically $W'(\boldsymbol{x}_U)$ is small around the maximum of $P(\boldsymbol{x}_U)$, causing the bias in the approximation.

The bias is not present, however, when the basis ϕ is orthogonal with respect to the inner product. The use of an orthogonal basis has additional advantages, because projection onto an orthogonal basis ϕ would be more stable numerically, cheaper computationally, and would make the expansion coefficients $\boldsymbol{c}$ independent of the size of the basis. Unfortunately, due to the weight factor $P(\boldsymbol{x}_U)$, the orthogonal basis is problem dependent, and in general, would have to be computed numerically. The orthogonalization of the basis ϕ would necessarily reduce to the QR decomposition (Francis, 1961, 1962; Kublanovskaya, 1962; Кублановская, 1961) of the Gram matrix A_ϕ, leading to an orthogonal basis ϕ'. Computing an updated Gram matrix $A_{\phi'}$ using the same random samples as those used to compute $A_{\phi'}$ indeed leads to a diagonal matrix, but it does not affect the errors due to the use of a truncated basis. That is, although the functions included in the basis ϕ' are orthogonal, the remaining functions that are beyond the basis are not orthogonal to those in ϕ'; therefore, their biasing influence remains.

An improvement to the above is the use of a weight factor $Q(\boldsymbol{x}_U)$ that partially eliminates the weight $P(\boldsymbol{x}_U)$, at the cost of a broader weight distribution. Using $Q(\boldsymbol{x}_U) = P(\boldsymbol{x}_U)$ would make the inner product uniform and allow for the use of an orthogonal basis (Binney et al., 1992), however the resulting weights would span an enormous range. Instead, we note that using $Q(\boldsymbol{x}_U) = P(\boldsymbol{x}_{\bar{N}(u)})$ also allows achieving a uniform inner product. Substituting this particular choice into the definition of the inner product from Equation 4.3 leads to

$$\begin{aligned}
\langle f, g \rangle &= \int f(\boldsymbol{x}_{\bar{N}(u)}) g(\boldsymbol{x}_{\bar{N}(u)}) \frac{P(\boldsymbol{x}_U)}{P(\boldsymbol{x}_{\bar{N}(u)})} d\boldsymbol{x}_U \\
&= \int \frac{f(\boldsymbol{x}_{\bar{N}(u)}) g(\boldsymbol{x}_{\bar{N}(u)})}{P(\boldsymbol{x}_{\bar{N}(u)})} \left(\int P(\boldsymbol{x}_U) d\boldsymbol{x}_{U \setminus \bar{N}(u)} \right) d\boldsymbol{x}_{\bar{N}(u)} \\
&= \int f(\boldsymbol{x}_{\bar{N}(u)}) g(\boldsymbol{x}_{\bar{N}(u)}) d\boldsymbol{x}_{\bar{N}(u)},
\end{aligned}$$

turning the weighted inner product into a uniform inner product for functions of $\boldsymbol{x}_{\bar{N}(u)}$. This choice of $Q(\boldsymbol{x}_U)$ makes it possible to use an orthogonal basis ϕ that is not specific to the statistical model under study.

The use of the weight factor $Q(\boldsymbol{x}_U)$ works differently from choosing a numerically computed orthogonal basis using the QR decomposition. The QR decomposition does not achieve more than simply solving the original

linear system $A_\phi \boldsymbol{c} = \boldsymbol{b}$. On the other hand, the application of the factor $Q(\boldsymbol{x}_U)$ modifies the inner product used and partially removes its bias.

The weight factor $Q(\boldsymbol{x}_U)$ does not have to be exact; in fact, using an approximate $Q(\boldsymbol{x}_U) = \hat{P}(\boldsymbol{x}_{\bar{N}(u)})$, obtained e.g. using the fast marginalization method, also leads to a marked improvement. However, the benefits decay with the size of the neighborhood $\bar{N}(u)$. Computing the marginal of the set of nearest neighbors of the node u is often sufficient.

The computation of the $Q(\boldsymbol{x}_U)$ factor is frequently costly and adds complexity to the fast marginalization method. For example, if the nearest neighborhood $\bar{N}(u)$ consisting of five nodes is used to compute the factor $Q(\boldsymbol{x}_U)$, the number of linear projections to be performed grows by a factor of five as well. However, we find that the use of this correction is necessary to overcome the bias caused by the inner product.

4.1.3 *Choice of a basis*

Thus far we have assumed that the basis ϕ is given. The choice of ϕ is generally very straightforward, but with a few important caveats. We open with the description of a polynomial basis and move on to discuss requirements for the basis terms that may be used. Finally, we discuss an algorithm that constructs a basis given the information obtained by the graph coarsening algorithm described in Chapter 3: the dependency digraph $D = (V, A)$ and the collection of subgraphs $G_u = (\bar{N}(u), E_u)$ for each $u \in V$.

4.1.3.1 *Basis functions*

We use here basis functions which are monomials in $\boldsymbol{x}_{\bar{N}(u)}$, e.g. $x_u^{k_u} x_v^{k_v} x_w^{k_w}$. The number of terms of order n in m variables is given by

$$\#_{n,m} = \binom{n+m-1}{m-1},$$

growing very quickly with the number of variables and the order of the terms. The basis functions must be limited to low order monomials.

4.1.3.2 *Integrability condition*

The approximation $\hat{W}'(\boldsymbol{x}_{\bar{N}(u)})$ of the derivative of the Hamiltonian $W'(\boldsymbol{x}_{\bar{N}(u)})$ must satisfy an important consistency condition to ensure that a unique function $\hat{W}(\boldsymbol{x}_U)$ exists, such that

$$\frac{\partial \hat{W}(\boldsymbol{x}_U)}{\partial x_u} = \hat{W}'(\boldsymbol{x}_{\bar{N}(u)})$$

for every $u \in U$. This integrability condition is equivalent to requiring that the Hessian of $\hat{W}(\boldsymbol{x}_U)$ be symmetric, that is,

$$\frac{\partial}{\partial x_v}\frac{\partial \hat{W}(\boldsymbol{x}_{\bar{N}(u)})}{\partial x_u} = \frac{\partial}{\partial x_u}\frac{\partial \hat{W}(\boldsymbol{x}_{\bar{N}(v)})}{\partial x_v}$$

for any $u, v \in U$. This is not immediately satisfied by the approximation $\hat{W}'$ because the derivatives are obtained independently using a stochastic algorithm.

With our choice of basis, the integrability condition translates into the following two requirements. Denote the basis at nodes $u, v \in U$ as ϕ_u and ϕ_v, respectively; similarly, let $\boldsymbol{c}^u$ and $\boldsymbol{c}^v$ be the expansion coefficients for the partial derivatives of $W(\boldsymbol{x}_U)$ with respect to the variables x_u and x_v. The first requirement is that if the basis ϕ_u contains a function $\phi_i \propto x_u^{p-1}x_v^q$, then the basis ϕ_v must contain the function $\phi_j \propto x_u^p x_v^{q-1}$. Secondly, the coefficients c_i^u and c_j^v must satisfy

$$\frac{c_i^u}{p} = \frac{c_j^v}{q}.$$

These relations are a consequence of the theorem of Hammersley and Clifford (1971):

Theorem (Hammersley-Clifford). *A probability distribution $P(\boldsymbol{x}_V)$ is both (i) strictly positive and (ii) respects the conditional independence structure encoded by the graph $G = (V, E)$ if and only if it factors over the cliques of G.*

The requirement about the conditional independence structure can be written as $x_u \perp\!\!\!\perp x_v \mid x_{V\setminus\{u,v\}}$ if and only if $(u, v) \notin E$, while the factorization means that one may write

$$P(\boldsymbol{x}_V) = \prod_C F_C(\boldsymbol{x}_C)$$

for all subsets $C \subset V$ for which the subgraph $G|_C$ is a clique. As a corollary, the Hamiltonian $W(\boldsymbol{x}_V)$ factors as

$$W(\boldsymbol{x}_V) = \sum_C \ln F_C(\boldsymbol{x}_C).$$

Since $P(\boldsymbol{x}_U) > 0$, by the Hammersley-Clifford theorem, we may write that if $(u, v) \in E_U$, it follows that there exist terms of the form $cx_u^p x_v^q$ in the expansion of $W(\boldsymbol{x}_U)$ in the basis Φ, where $c = c(\boldsymbol{x}_{U\setminus\{u,v\}})$ is independent of x_u and x_v. Take such a term and call it $\Phi_k = cx_u^p x_v^q$. The terms $\partial\Phi_k/\partial x_u = cpx_u^{p-1}x_v^q$ and $\partial\Phi_k/\partial x_v = cqx_u^p x_v^{q-1}$ must therefore appear in the expansions of the partial derivatives of $W(\boldsymbol{x}_U)$ with respect to x_u and x_v, respectively. Letting $\phi_i^u = x_u^{p-1}x_v^q$ and $\phi_j^v = x_u^p x_v^{q-1}$, we see that they represent the same term in the expansion of $W(\boldsymbol{x}_U)$, and thus the coefficients $c_i^u = cp$ and $c_j^v = cq$ must satisfy the relationship above.

4.1.3.3 *Reduction by symmetry*

Probabilistic models defined on regular lattices frequently involve symmetries. For example, the Ising model on a square lattice inherits the complete set of symmetries of the underlying Cartesian lattice: rotation by $\pi/2$, π and $3\pi/2$, reflections about the major, $\pi/4$ and $3\pi/4$ axes, translation, and their arbitrary compositions. Many basis functions can thus be seen as images of each other under symmetry transformations.

Choose two cliques $C, C' \subset U$ and construct two basis functions Φ_C and $\Phi_{C'}$ of the same functional form. If the clique C' is an image of C under the symmetry transformation γ, written as $C' = \gamma(C)$, then the two basis functions must be equivalent, that is, their coefficients in the expansion of $W(\boldsymbol{x}_U)$ in the basis Φ must be equal. For example, in the Ising model the only cliques are edges and all edges have the same coupling coefficient μ, because each edge can be mapped onto any other edge by the symmetry transformations. This observation allows one to link certain basis functions, because their expansion coefficients must be the same, thus reducing the size of the needed basis.

Example 4.1. In the Ising model example, consider the linear basis functions $x_{i+1,j}, x_{i-1,j}, x_{i,j+1}$ and $x_{i,j-1}$ appearing in the basis ϕ for the expansion of $\partial W(\boldsymbol{x}_U)\big/\partial x_{ij}$. By the Hammersley-Clifford theorem, these basis functions correspond to the terms $x_{ij}x_{i+1,j}, x_{ij}x_{i-1,j}, x_{ij}x_{i,j+1}$ and $x_{ij}x_{i,j-1}$ appearing in the expansion of $W(\boldsymbol{x}_U)$. The subsets of nodes U these terms correspond to are clearly equivalent under symmetry transformations and their basis coefficients must be equal. We combine them into a single basis function $x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}$ that corre-

Algorithm 4.1 Basis reduction algorithm using the equivalences between basis functions due to symmetry transformations γ.

```
function SymmetryReduction(φ, γ)
    φ' = ∅
    for all φ_i do
        ψ ← φ_i
        φ ← φ \ φ_i
        for all γ_j do
            φ_i^j ← γ_j(φ_i)
            for all φ_k do
                if φ_i^j = φ_k then
                    ψ ← ψ + φ_k
                end if
            end for
        end for
        φ' ← φ' ∪ ψ
    end for
    return φ'
end function
```

sponds to the term $x_{ij}(x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1})$ in the expansion of $W(\boldsymbol{x}_U)$. We have reduced the basis by three functions without reducing the size of the subspace X_ϕ. This reduction is not possible if the graph $G_U = (U, E_U)$ is not symmetric, even if the original graph $G = (V, E)$ was. ■

4.1.3.4 *Consistency and dependencies*

The fact that the images of basis functions under symmetry transformations $\phi_i^j = \gamma_j(\phi_i)$, defined in Algorithm 4.1, may be functions of variables beyond those of $\boldsymbol{x}_{\bar{N}(u)}$ is of crucial importance and is mentioned by Ron and Swendsen (2002). Following their example, we make the point in a simple context. Consider the Ising model and an approximate renormalized graph $G_U = (U, E_U)$ where the nodes U form a regular lattice and edges are formed only between the nearest neighbors. Because of the assumption made in the graph G_U that the variable x_{ij} depends only on the variables $x_{i+1,j}, x_{i-1,j}, x_{i,j+1}$ and $x_{i,j-1}$, a naive algorithm for constructing the basis ϕ would produce a basis containing all the polynomials in the neighboring variables,

$$1, \quad x_{i\pm1,j}, \quad x_{i,j\pm1}, \quad x_{i\pm1,j}x_{i,j+1},$$

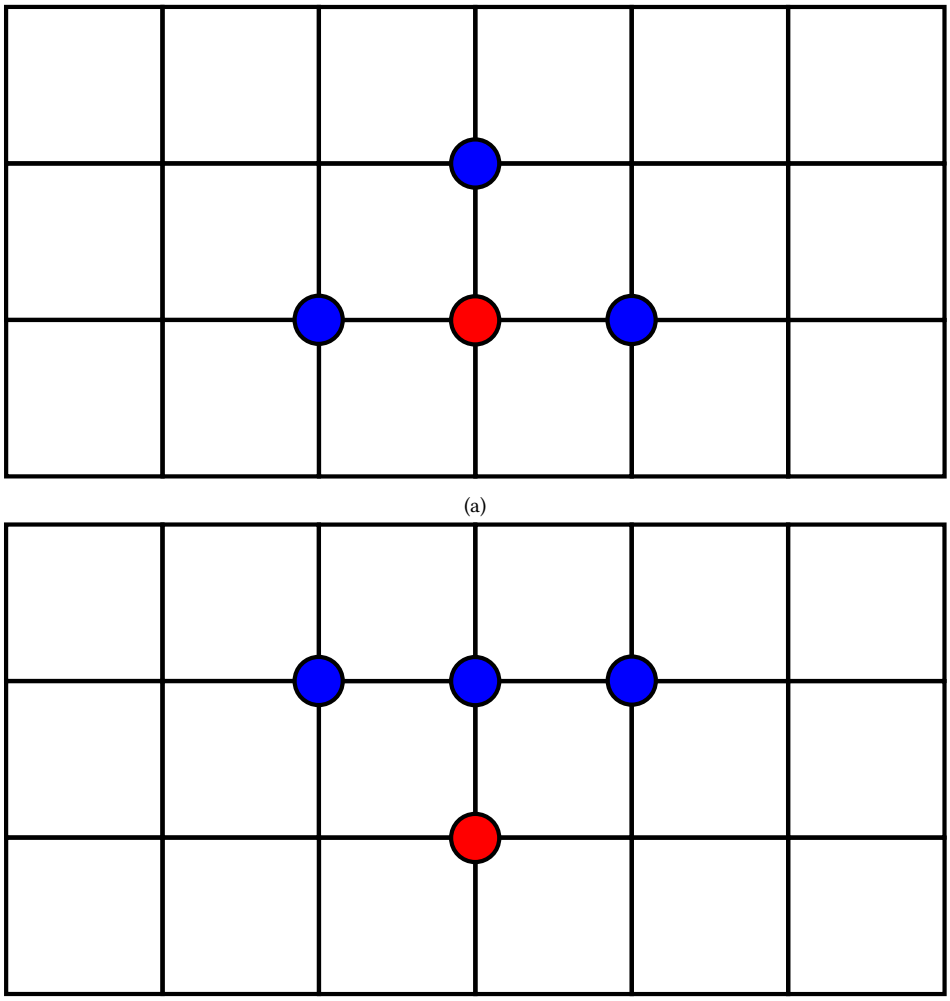
$$x_{i\pm1,j}x_{i,j-1}, \quad x_{i+1,j}x_{i-1,j}, \quad x_{i,j+1}x_{i,j-1},$$
$$x_{i+1,j}x_{i-1,j}x_{i,j\pm1}, \quad x_{i\pm1,j}x_{i,j-1}x_{i,j+1},$$

where the absence of the variable x_{ij} is explained in later sections. Consider the cubic function $\phi_k = x_{i+1,j}x_{i,j+1}x_{i-1,j}$ shown on Figure 4.15. Because of the consistency requirement, the function $\phi'_k = x_{i+1,j}x_{i,j+1}x_{i-1,j}$ must belong to the basis of the node $x_{i,j+1}$. However, the graph G_U assumed that the variable $x_{i,j+1}$ depends only on its nearest neighbors $x_{i-1,j+1}, x_{i+1,j+1}, x_{i,j+1}$ and x_{ij}. Due to the consistency requirements, the choice of a basis function $x_{i+1,j}x_{i,j+1}x_{i-1,j}$ at the node x_{ij} forces additional dependency relations between the nodes $x_{i,j+1}$, $x_{i+1,j}$ and $x_{i-1,j}$ that are inconsistent with G_U, because the edges $\Big((i,j+1),(i+1,j)\Big)$, $\Big((i,j+1),(i-1,j)\Big)$ and $\Big((i-1,j),(i+1,j)\Big)$ do not belong to E_U. Therefore, the highest order basis function consistent with G_U is the linear term. Inclusion of higher order terms leads to breaking the consistency requirement and thus lack of a Hamiltonian $\hat{W}(\boldsymbol{x}_U)$ consistent with the partial derivatives $\hat{W}'(\boldsymbol{x}_{\bar{N}(u)})$, while expansion of the basis to include the additional basis functions required by the consistency requirement causes a break in the dependency graph G_U. This is a fundamental failure, because the renormalized coupling coefficients obtained using an inconsistent basis cannot be used to reliably approximate the marginal probability $P(\boldsymbol{x}_U)$. In practice it is found that an inconsistent basis may bias the probability distribution of the Ising model so that only states of positive magnetization have significant probability densities.

The solution to this troubling development is to respect the dependency graph G_U and remove the basis functions that would cause consistency issues. Take the subgraph $G_U|_{\bar{N}(u)}$ consisting of the node u and its neighborhood. To ensure that the basis is consistent with the dependency graph, we allow only basis functions

$$\phi_k = \prod_{v\in C} x_v^{k_v}$$

such that the set C is a clique of the subgraph $G_U|_{\bar{N}(u)}$. This requirement guarantees that adding the above basis function ϕ_k does not form additional edges in the dependency graph G_U, because all possible edges between the variables $v \in C$ already exist. Therefore, the graph G_U remains unchanged and the basis ϕ is consistent.



(a)

(b)

Figure 4.15: The cubic basis function (a) $x_{i+1,j}x_{i,j+1}x_{i-1,j}$ and its equivalent function, (b) $x_{i+1,j+1}x_{i,j+1}x_{i-1,j+1}$. The node x_{ij} is marked red, while the remaining nodes of the particular basis function are colored blue.

4.1.3.5 *Practical basis construction algorithm*

The above restrictions placed on the possible basis choices are combined to create an algorithm for constructing a basis, described in Algorithm 4.2. The main feature of the algorithm is the outer loop. We loop over the cliques of G_u and consider each such clique C separately. Within the clique C we are free to consider all combinations of variables x_v, $v \in C$. We choose to form all possible monomials of variables $\boldsymbol{x}_C$ that have order smaller or equal to m, with the monomial defined by the powers

Algorithm 4.2 Algorithm for constructing a basis ϕ for $W'(\boldsymbol{x}_U)$ at node $u \in U$.

function BASISCONSTRUCTION($G_u = G_U|_{\bar{N}(u)}$, n, m, T_i, γ)
  $\phi \leftarrow \emptyset$
  for all cliques $C \in G_u$, $|C| \leq n$ **do**
    for $i = 0$ to m **do**
      for all $k_1, k_2, \ldots, k_{|C|}$ with $\sum_{j=1}^{|C|} k_j = i$ **do**
        $C' = \{v_j \mid v_j \in C \text{ and } k_j > 0\}$
        if $r(C') < T_i$ **then**
          $\phi \leftarrow \phi \cup \prod_{v_j \in C'} x_{v_j}^{k_j}$
        end if
      end for
    end for
  end for
  $\phi \leftarrow$ ELIMINATEDUPLICATES(ϕ)
  $\phi \leftarrow$ SYMMETRYREDUCTION(ϕ, γ)
  return ϕ
end function

$k_1, k_2, \ldots, k_{|C|}$ as $x_{v_1}^{k_1} x_{v_2}^{k_2} \ldots x_{v_{|C|}}^{k_{|C|}}$. Because the number of such monomials may be very large, we allow only those that involve nodes forming a sub-clique C' of radius $r(C') < T_i$, where T_i is the maximum radius for monomials of order i.

The Algorithm 4.2 will construct certain functions multiple times. Duplications will occur when two cliques C and C' have an intersection, with the trivial intersection $C \cap C' = \emptyset$ leading to the generation of a constant function once for each clique. The duplicate functions are eliminated once the entire basis is formed. Reduction of the basis using symmetries is applied as the last step of Algorithm 4.1.

4.1.4 *Representation of the marginal probability*

The fast marginalization method does not produce an approximation of the marginal probability density $P(\boldsymbol{x}_U)$ directly. Instead, it gives us an approximation to the partial derivative of the Hamiltonian, $\widehat{W}'(\boldsymbol{x}_{\bar{N}(u)})$. This approximation may be used to compute various quantities of interest, with different levels of difficulty. We discuss the three most important: (i) the energy difference between two states $\boldsymbol{x}_U$ and $\boldsymbol{y}_U$, (ii) the conditional

probability of x_u given the remaining variables $\boldsymbol{x}_{U\setminus u}$, and (iii) the unnormalized marginal probability density $P(\boldsymbol{x}_U)$.

4.1.4.1 *Energy difference*

The energy difference between the states $\boldsymbol{x}_U$ and $\boldsymbol{y}_U$ is the difference in the value of the Hamiltonian W evaluated at the two states,

$$\Delta W(\boldsymbol{x}_U, \boldsymbol{y}_U) = W(\boldsymbol{y}_U) - W(\boldsymbol{x}_U).$$

This quantity is of interest, because it is the logarithm of the ratio of probabilities of the states $\boldsymbol{x}_U$ and $\boldsymbol{y}_U$,

$$\begin{aligned}\ln\left(\frac{P(\boldsymbol{y}_U)}{P(\boldsymbol{x}_U)}\right) &= \ln P(\boldsymbol{y}_U) - \ln P(\boldsymbol{x}_U)\\ &= W(\boldsymbol{y}_U) - W(\boldsymbol{x}_U)\\ &= \Delta W(\boldsymbol{x}_U, \boldsymbol{y}_U).\end{aligned}$$

The ratio $P(\boldsymbol{y}_U)/P(\boldsymbol{x}_U)$ appears in the Metropolis-Hastings probability of accepting a proposed move $\boldsymbol{x}_U \to \boldsymbol{y}_U$

$$\alpha(\boldsymbol{x}_U, \boldsymbol{y}_U) = \frac{P(\boldsymbol{y}_U)P(\boldsymbol{y}_U \to \boldsymbol{x}_U)}{P(\boldsymbol{x}_U)P(\boldsymbol{x}_U \to \boldsymbol{y}_U)},$$

where $P(\boldsymbol{x}_U \to \boldsymbol{y}_U)$ is the proposal probability (Liu, 2001; Metropolis et al., 1953; Robert and Casella, 2004). Our representation allows for an efficient computation of $\Delta W(\boldsymbol{x}_U, \boldsymbol{y}_U)$ when the change between $\boldsymbol{x}_U$ and $\boldsymbol{y}_U$ involves a single variable, that is, when

$$x_{U\setminus u} = y_{U\setminus u} \quad \text{but} \quad x_u \neq y_u.$$

The single-variable energy difference $\Delta_u W(\boldsymbol{x}_{U\setminus u}; x_u, y_u)$, representing the difference in the value of the Hamiltonian $\Delta W(\boldsymbol{x}_{U\setminus u}, x_u; \boldsymbol{x}_{U\setminus u}, y_u) = W(\boldsymbol{x}_{U\setminus u}, y_u) - W(\boldsymbol{x}_{U\setminus u}, x_u)$, may be approximated by

$$\begin{aligned}\Delta_u \hat{W}(\boldsymbol{x}_{U\setminus u}; x_u, y_u) &= \int_{x_u}^{y_u} \hat{W}'(\boldsymbol{x}_{N(u)}, s)ds\\ &= \int_{x_u}^{y_u} \sum_{i=1}^{K} c_i \phi_i(\boldsymbol{x}_{N(u)}, s)ds\\ &= \sum_{i=1}^{K} c_i \int_{x_u}^{y_u} \phi_i(\boldsymbol{x}_{N(u)}, s)ds.\end{aligned}$$

The general energy difference $\Delta W(\boldsymbol{x}_U, \boldsymbol{y}_U)$ may be decomposed into a sequence of single-variable energy differences and approximated by their sum, a technique used later in this chapter to reconstruct the marginal probability density $P(\boldsymbol{x}_U)$.

4.1.4.2 *Conditional probability*

We are interested in a slightly modified form of the above energy difference. The Chapter 3 ended by constructing a dependency digraph $D = (V, A)$, which may be used to write the probability density $P(\boldsymbol{x}_V)$ in the acyclic form

$$P(\boldsymbol{x}_V) = P(\boldsymbol{x}_{V_m}) \times P(x_{u_{|V_m|+1}} \mid \boldsymbol{x}_{N_p(u_{|V_m|+1})}) \times \\ \times \ldots P(x_{u_i} \mid \boldsymbol{x}_{N_p(u_{i-1})}) \ldots \times P(x_{u_{|V|}} \mid \boldsymbol{x}_{N_P(u_{|V|-1})}).$$

We would like to compute the conditional probability of x_u given its direct predecessor variables $\boldsymbol{x}_{N_p(u)}$, $P(x_u \mid \boldsymbol{x}_{N_p(u)})$. Assuming that $u \in U$, $N_p(u) \subset U$ and that $N(u) \subseteq N_p(u)$, where $N(u)$ is the set of neighbors within the graph $G_U = (U, E_U)$ while $N_p(u)$ is the set of direct predecessor nodes of u within the digraph $D = (V, A)$; we may compute this conditional probability very efficiently from the definition

$$\begin{aligned} P(x_u \mid \boldsymbol{x}_{N_p(u)}) &= \frac{P(x_u, \boldsymbol{x}_{U\setminus u})}{\int P(t, \boldsymbol{x}_{U\setminus u}) dt} \\ &= \frac{F_u(x_u, \boldsymbol{x}_{N(u)}) F_\alpha(\boldsymbol{x}_{N(u)}, \boldsymbol{x}_{U\setminus \bar{N}(u)}) \Big/ Z_U}{\int \left(F_u(t, \boldsymbol{x}_{N(u)}) F_\alpha(\boldsymbol{x}_{N(u)}, \boldsymbol{x}_{U\setminus \bar{N}(u)}) \Big/ Z_U \right) dt} \\ &= \frac{F_u(x_u, \boldsymbol{x}_{N(u)})}{\int F_u(t, \boldsymbol{x}_{N(u)}) dt}. \end{aligned}$$

Note that in this context we may use $N_p(u)$ and $N(u)$ interchangeably. The function $F_u(x_u, \boldsymbol{x}_{N_p(u)})$, appearing earlier in Equation 4.2, is the exponential of a local part $W_u(x_u, \boldsymbol{x}_{N(u)})$ of the Hamiltonian $W(\boldsymbol{x}_U)$, defined below. Because the probability $P(\boldsymbol{x}_U)$ factorizes as in Equation 4.2, so does its logarithm, allowing us to write

$$W(\boldsymbol{x}_U) = W_u(x_u, \boldsymbol{x}_{N(u)}) + W_\alpha(\boldsymbol{x}_{N(u)}, \boldsymbol{x}_{U\setminus \bar{N}(u)}). \tag{4.4}$$

It follows that $W'(\boldsymbol{x}_U) = W'_u(x_u, \boldsymbol{x}_{N(u)})$. We can therefore compute $W_u(x_u, \boldsymbol{x}_{N(u)})$ up to an additive constant,

$$W_u(x_u, \boldsymbol{x}_{N(u)}) = \int W'_u(x_u, \boldsymbol{x}_{N(u)}) dx_u + C(\boldsymbol{x}_{U\setminus u}).$$

We fix the constant by selecting a reference value of $x_u = x^*$. Then,

$$\begin{aligned} P(x_u \mid \boldsymbol{x}_{N_p(u)}) &= \frac{F_u(x_u, \boldsymbol{x}_{N_p(u)})}{\int F_u(t, \boldsymbol{x}_{N_p(u)}) dt} \frac{F_u(x^*, \boldsymbol{x}_{N_p(u)})}{F_u(x^*, \boldsymbol{x}_{N_p(u)})} \\ &= \frac{F_u(x_u, \boldsymbol{x}_{N_p(u)}) \Big/ F_u(x^*, \boldsymbol{x}_{N_p(u)})}{\int \left(F_u(t, \boldsymbol{x}_{N_p(u)}) \Big/ F_u(x^*, \boldsymbol{x}_{N_p(u)}) \right) dt} \\ &= \frac{\exp\left(\Delta_u W(\boldsymbol{x}_{U\setminus u}; x^*, x_u) \right)}{\int \exp\left(\Delta_u W(\boldsymbol{x}_{U\setminus u}; x^*, t) \right) dt}, \end{aligned}$$

where $\Delta_u W(\boldsymbol{x}_{U\setminus u}; x^*, x_u)$ is the single-variable energy difference defined previously. Finally, we may write

$$\hat{P}(x_u \mid \boldsymbol{x}_{N_p(u)}) = \frac{\exp\left(\sum_{i=1}^{K} c_i \int_{x^*}^{x_u} \phi_i(\boldsymbol{x}_{N_p(u)}, s) ds \right)}{\int \exp\left(\sum_{i=1}^{K} c_i \int_{x^*}^{t} \phi_i(\boldsymbol{x}_{N_p(u)}, s) ds \right) dt},$$

which is a self-contained formula for an approximate conditional probability of x_u given the neighboring variables $\boldsymbol{x}_{N_p(u)}$.

4.1.4.3 *Marginal probability*

The reconstruction of the marginal probability $P(\boldsymbol{x}_U)$ from the approximation of the derivative $W'(\boldsymbol{x}_U)$ can be accomplished using the following algorithm. Pick a particular state $\boldsymbol{x}^*_U$ such that $P(\boldsymbol{x}^*_U) > 0$ and specify that the Hamiltonian attains zero at $\boldsymbol{x}^*_U$, that is $\hat{W}(\boldsymbol{x}^*_U) \equiv 0$. While any state $\boldsymbol{x}^*_U$ satisfying the above positivity constraint $P(\boldsymbol{x}^*_U) > 0$ is allowed,

numerically it is preferable that a state of *average probability* be selected, defined as a state $\boldsymbol{x}_U^*$ such that

$$\boldsymbol{x}_U^* = \arg\min_{\boldsymbol{x}_U} \left(\int W(\boldsymbol{y}_U) - W(\boldsymbol{x}_U) d\boldsymbol{y}_U \right)^2 .$$

This choice ensures that the values of the exponential

$$\exp\left(\hat{W}(\boldsymbol{x}_U)\right) \approx \exp\left(W(\boldsymbol{x}_U) - W(\boldsymbol{x}_U^*)\right)$$

can be computed with minimum round-off error.

Given a choice of $\boldsymbol{x}_U^*$, we construct a sequence of states $\{\boldsymbol{y}_U^i\}$ with $\boldsymbol{y}_U^0 = \boldsymbol{x}_U^*$ and $\boldsymbol{y}_U^j = \boldsymbol{x}_U$. The sequence $\{\boldsymbol{y}_U^i\}$ must have the property that any two states $\boldsymbol{y}_U^i$ and $\boldsymbol{y}_U^{i+1}$ differ in the value of only a single variable x_{v_i}. Therefore,

$$\Delta W(\boldsymbol{y}_U^i, \boldsymbol{y}_U^{i+1}) = \Delta_{v_i} W(\boldsymbol{y}_U^i; y_{v_i}^i, y_{v_i}^{i+1}).$$

We may express the energy difference $\Delta W(\boldsymbol{x}_U^*, \boldsymbol{x}_U)$ as

$$\Delta W(\boldsymbol{x}_U^*, \boldsymbol{x}_U) = \sum_{i=0}^{j-1} \Delta_{v_i} W(\boldsymbol{y}_U^i; y_{v_i}^i, y_{v_i}^{i+1}),$$

giving a natural definition of $\hat{W}(\boldsymbol{x}_U)$ as

$$\begin{aligned} \hat{W}(\boldsymbol{x}_U) &= \hat{W}(\boldsymbol{x}_U^*) + \Delta\hat{W}(\boldsymbol{x}_U^*, \boldsymbol{x}_U) \\ &= \sum_{i=0}^{j-1} \Delta_{v_i} \hat{W}(\boldsymbol{y}_U^i; y_{v_i}^i, y_{v_i}^{i+1}), \end{aligned}$$

since $\hat{W}(\boldsymbol{x}_U^*) \equiv 0$. Finally, the approximate marginal probability $\hat{P}(\boldsymbol{x}_U)$ is defined to be

$$\hat{P}(\boldsymbol{x}_U) \equiv \exp\left(\hat{W}(\boldsymbol{x}_U)\right) \Big/ \hat{Z}_U(\boldsymbol{x}_U^*),$$

completing the reconstruction. While we do not know the value of the normalizing constant $\hat{Z}_U(\boldsymbol{x}_U^*)$, its value is determined uniquely by the choice of the state $\boldsymbol{x}_U^*$ and the approximation $\hat{W}(\boldsymbol{x}_U)$.

In models considered in this thesis the above approach always produces a well-defined probability density $\hat{P}(\boldsymbol{x}_U)$. The process consists of a finite number of steps, thus we do not need to worry about convergence of the sum. More importantly, the assumption $P(\boldsymbol{x}_V) > 0$ implies that the

Problems could appear if $P(\boldsymbol{x}_U)$ were zero for some states $\boldsymbol{x}_U$, because then the Hamiltonian $W(\boldsymbol{x}_U)$ would be infinite.

Hamiltonian is always finite; therefore, independently of the choice of the sequence $\{\boldsymbol{y}_U^i\}$, the algorithm will never involve performing undefined operations, such as $\infty - \infty$. Because of this, the simplest path involving changing components of $\boldsymbol{x}_U$ in a pre-decided ordering will lead to a correct reconstruction algorithm.

4.2 THE CASE OF DISCRETE VARIABLES

The methods described earlier are mostly compatible with probability distributions defined over discrete variables, as the algorithms necessary for graph coarsening and sampling are compatible with both discrete and continuous variables. However, the fast marginalization algorithm seemingly hits a wall when we need to take a derivative with respect to x_u. In this section we solve this difficulty by using differentiable extensions.

4.2.1 *Projection*

We want to note that the issues involved in the application of the fast marginalization method to discrete variables were observed earlier by Okunev (2005), who suggested that the variables may be made continuous one at a time, producing a hypercube with edges corresponding to the continuous variables. However, both Okunev (2005) and Chorin (2008) assumed that the function $W_u'(x_u, \boldsymbol{x}_{N(u)})$ obtained using fast marginalization is constant in the continuous variable x_u, which was not the case.

4.2.1.1 *Derivative projection*

We modify the fast marginalization method through the introduction of a differentiable extension of $P(\boldsymbol{x}_V)$, allowing for differentiation to take place. Throughout this section we denote the differentiable extensions of otherwise discrete functions using the tilde, e.g. the differentiable extension of $P(\boldsymbol{x}_V)$ becomes $\tilde{P}(\boldsymbol{x}_V)$.

Assume the probability distribution function $P(\boldsymbol{x}_V)$ and its marginal $P(\boldsymbol{x}_U)$ are defined over discrete variables. Integrals are to be understood as summation over the relevant variables. For $u \in U$, extend the variable x_u to the real line, changing notation from x_u to χ_u to reflect this change. We define a differentiable extension $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$ of the original probability density $P(\boldsymbol{x}_V)$ such that

$$\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) = P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}),$$

that is the differentiable extension equals $P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$ whenever the continuous variable χ_u takes one of the original, discrete values.

Having chosen the interpolant $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$ defining the differentiable extension, we obtain the differentiable extension of the marginal probability $\tilde{P}(\boldsymbol{x}_U)$ through

$$\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}) = \int \tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U}.$$

The differentiable extension $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$ uniquely determines the differentiable extension of the marginal probability distribution, which in general is a non-linear function of the continuous variable χ_u. We notice that $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u})$ is an interpolant of $P(x_u, \boldsymbol{x}_{U\setminus u})$, since for a discrete χ_u taking one of the original values we have

$$\begin{aligned}
\tilde{P}(x_u, \boldsymbol{x}_{U\setminus u}) &= \int \tilde{P}(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U} \\
&= \int P(x_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U}) d\boldsymbol{x}_{V\setminus U} \\
&= P(x_u, \boldsymbol{x}_{U\setminus u}).
\end{aligned}$$

However, while the interpolant used to define $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$ may be a low-order function of χ_u, the marginal distribution $\tilde{P}(x_u, \boldsymbol{x}_{U\setminus u})$ will typically be a highly non-linear function of χ_u.

The derivation of the fast marginalization equation follows directly the steps discussed in Section 4.1, where we use the differentiable extension in place of the probability distribution function. Denoting the differentiable extension of $F_u(x_u, \boldsymbol{x}_{N(u)})$ of Equation 4.2 as $\tilde{F}_u(x_u, \boldsymbol{x}_{N(u)})$, the final result becomes

$$\frac{\partial \tilde{W}(\chi_u, \boldsymbol{x}_{U\setminus u})}{\partial \chi_u} = \mathbb{E}\left[\frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{V\setminus u})}{\partial \chi_u} \Big/ \tilde{F}_u(\chi_u, \boldsymbol{x}_{V\setminus u}) \middle| \boldsymbol{x}_U \right],$$

which is precisely equivalent to that obtained for the continuous variables. Therefore, we may approximate $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$ through fast marginalization.

In a step absent from the continuous case, we recover the discrete $W(x_u, \boldsymbol{x}_{U\setminus u})$ by integrating $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$ between the original discrete values, obtaining

$$\begin{aligned}\Delta_u W(\boldsymbol{x}_{U\setminus u}; a, b) &= \int_a^b \tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u}) d\chi_u \\ &= \tilde{W}(b, \boldsymbol{x}_{U\setminus u}) - \tilde{W}(a, \boldsymbol{x}_{U\setminus u}) \\ &= W(b, \boldsymbol{x}_{U\setminus u}) - W(a, \boldsymbol{x}_{U\setminus u})\end{aligned}$$

for two discrete values a and b of x_u. Because the value of the integral is fixed, the method does not depend on the particular form of the differentiable extension (cf. Section 6.6.5). Although the function $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$ will depend on the choice of the differentiable extension $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u}, \boldsymbol{x}_{V\setminus U})$, the integral will not. However, we stress that the values of the function $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$ are *not* independent of the choice of interpolant, which has a major influence on the performance of the numerical method.

4.2.1.2 *Natural interpolants*

The probability distribution $P(\boldsymbol{x}_V)$ may be interpolated in multiple ways. If the variable x_u takes only two values, $x_u \in \{a, b\}$, the probability distribution may be interpolated linearly as

$$\tilde{P}(\chi_u, \boldsymbol{x}_{V\setminus U}) = \left(1 - \frac{\chi_u - a}{b - a}\right) P(a, \boldsymbol{x}_{V\setminus U}) + \frac{\chi_u - a}{b - a} P(b, \boldsymbol{x}_{V\setminus U}),$$

which can be differentiated with respect to χ_u, yielding

$$\frac{\partial \tilde{P}(\chi_u, \boldsymbol{x}_{V\setminus U})}{\partial \chi_u} = \frac{1}{b - a} \left(P(b, \boldsymbol{x}_{V\setminus U}) - P(a, \boldsymbol{x}_{V\setminus U})\right).$$

This formula may then be used directly in the above fast marginalization equation. However, frequently there exists a natural interpolant, since the probability distribution $P(\boldsymbol{x}_V)$ is defined through a formula that may be extended to continuous variables; therefore, the formula acts as the interpolant. For example, in case of the Ising model we have

$$P(\boldsymbol{x}_{\tilde{N}(u)}) \propto \exp\left(\mu x_u \sum_{v \in N(u)} x_v\right),$$

which may be trivially extended to continuous values by replacing x_u with χ_u, giving

$$\frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{V\setminus u})}{\partial \chi_u} \Big/ \tilde{F}_u(\chi_u, \boldsymbol{x}_{V\setminus u}) = \mu \sum_{v \in N(u)} x_v.$$

This choice reduces the method above to that of Chorin (2003, 2008) and Okunev (2005).

4.2.2 *Choice of a basis*

Having explained the machinery used to taking a derivative $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$, we return to the question of how $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$ should be represented and introduce the mixed continuous-discrete representation.

4.2.3 *Mixed projection*

While the variables $\boldsymbol{x}_{N(u)}$ remain discrete, the variable χ_u is continuous. It does not have the same limits in terms of indistinguishableness of polynomials, therefore the complete basis for the functions of χ_u and $\boldsymbol{x}_{N(u)}$ is the outer product

$$\begin{aligned}\phi = \{1, \chi_u, \chi_u^2, \chi_u^3, \dots, \chi_u^m\} \otimes \{1, x_{v_1}\} \\ \otimes \{1, x_{v_2}\} \otimes \dots \otimes \{1, x_{v_{|N(u)|}}\}\end{aligned}$$

for $v_1, v_2, \dots, v_{|N(u)|} \in N(u)$. However, after computing the expansion, we are only interested in the energy difference

$$\begin{aligned}\Delta_u W(\boldsymbol{x}_{U\setminus u}; a, b) &= W(x_u = b, \boldsymbol{x}_{U\setminus u}) - W(x_u = a, \boldsymbol{x}_{U\setminus u}) \\ &= \int_a^b \tilde{W}(\chi_u, \boldsymbol{x}_{N(u)}) d\chi_u.\end{aligned}$$

Therefore, the basis functions $\{1, \chi_u, \chi_u^2, \chi_u^3, \dots, \chi_u^m\}$ are immediately integrated out, suggesting that a more efficient approach may be employed.

4.2.3.1 *Mixed representation*

Numerical integration of a function requires us that we know its value at a set of quadrature modes. Therefore, instead of representing $\tilde{W}(\chi_u, \boldsymbol{x}_{N(u)})$ continuously at all possible values of χ_u we find a set of approximations

for χ_u taking values from the set of quadrature nodes $\{t_1, t_2, \dots, t_n\}$. In other words, instead of the series

$$\hat{W}'(\chi_u, \boldsymbol{x}_{N(u)}) = \sum_i c_i \phi_i(\chi_u, \boldsymbol{x}_{N(u)})$$

we will expand $\tilde{W}(\chi_u, \boldsymbol{x}_{N(u)})$ in a series

$$\hat{W}'(\chi_u, \boldsymbol{x}_{N(u)}) = \sum_i c_i(\chi_u) \phi_i(\boldsymbol{x}_{N(u)}),$$

capturing the continuity of the variable χ_u in the expansion coefficients. Thus, the basis ϕ is composed of functions of the discrete variables $\boldsymbol{x}_{N(u)}$ only.

At each integration node we find a set of coefficients $\boldsymbol{c}(t_j)$ that represents the closest match to $\tilde{W}'(t_j, \boldsymbol{x}_{N(u)})$. Then, the discrete approximation to the difference $\Delta_u W(\boldsymbol{x}_{U\setminus u}; a, b)$ becomes

$$\begin{aligned}
\Delta_u W(\boldsymbol{x}_{U\setminus u}; a, b) &= W(x_u = b, \boldsymbol{x}_{U\setminus u}) - W(x_u = a, \boldsymbol{x}_{U\setminus u}) \\
&= \int_a^b \tilde{W}(\chi_u, \boldsymbol{x}_{N(u)}) d\chi_u \\
&= \int_a^b \sum_i c_i(\chi_u) \phi_i(\boldsymbol{x}_{N(u)}) d\chi_u \\
&= \sum_i \phi_i(\boldsymbol{x}_{N(u)}) \int_a^b c_i(\chi_u) d\chi_u \\
&\approx \sum_i \phi_i(\boldsymbol{x}_{N(u)}) \sum_j c_i(t_j) w_j
\end{aligned}$$

where the integrals of the coefficients $c_i(\chi_u)$ are approximated using an appropriate quadrature rule composed of the integration nodes t_j and weights w_j. Therefore, we eliminate the continuous variable χ_u, recovering a discrete approximation of the Hamiltonian $W(\boldsymbol{x}_U)$.

4.2.3.2 *Node-wise approximation via fast marginalization*

The fast marginalization performs a linear least squares projection of the function

$$\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) = \frac{\partial \tilde{W}(\chi_u, \boldsymbol{x}_{U\setminus u})}{\partial \chi_u}$$

$$= \mathbb{E}\left[\frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)}}{\partial \chi_u} \Big/ \tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)}) \middle| \boldsymbol{x}_U\right]$$

onto a basis ϕ. As with continuous variables, we must develop a suitable inner product that will allow us to perform this projection node-wise, i.e., separately for each value $\chi_u = t_j$. We want to turn the conditional expectation above into expected values with respect to the original probability distribution $P(\boldsymbol{x}_V)$; however, in the case of discrete variables we face the additional difficulty that the variable x_u has been made continuous, requiring sampling from the differentiable extension $\tilde{P}(\chi_u, \boldsymbol{x}_{V\setminus u})$. We solve this issue below using an approach akin to importance sampling.

Consider finding the optimal approximation of the function $\mathcal{F}(\chi_u, \boldsymbol{x}_{U\setminus u})$ in the least squares sense for a fixed value $\chi_u = t_j$. We project it on a basis ϕ of functions of $\boldsymbol{x}_{N(u)}$, thus requiring the inner product to be

$$\begin{aligned}\langle f, g\rangle_{\chi_u} &= \int f(s, \boldsymbol{x}_{U\setminus u}) g(s, \boldsymbol{x}_{U\setminus u}) \frac{\tilde{P}(s, \boldsymbol{x}_{U\setminus u})\delta(s - \chi_u)}{Q(s, \boldsymbol{x}_{U\setminus u})} ds d\boldsymbol{x}_{U\setminus u} \\ &= \int f(\chi_u, \boldsymbol{x}_{U\setminus u}) g(\chi_u, \boldsymbol{x}_{U\setminus u}) \frac{\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} d\boldsymbol{x}_{U\setminus u}.\end{aligned}$$

Following the derivation of the projection equation from Section 4.1.1, we compute the Gram matrix $A(\chi_u)$ and the right hand side projection vector $\boldsymbol{b}(\chi_u)$, which are now functions of the continuous variable χ_u. The Gram matrix becomes

$$A_{ij}(\chi_u) = \mathbb{E}\left[\frac{\phi_i(\boldsymbol{x}_{N(u)})\phi_j(\boldsymbol{x}_{N(u)})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} \frac{\tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)})}{\int F_u(x_u, \boldsymbol{x}_{N(u)}) dx_u}\right].$$

Similarly, the formula for the right hand side projection vector $\boldsymbol{b}(\chi_u)$ is found to be

$$b_i(\chi_u) = \mathbb{E}\left[\frac{\phi_i(\boldsymbol{x}_{N(u)})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} \frac{\frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)})}{\partial \chi_u}}{\int F_u(x_u, \boldsymbol{x}_{N(u)}) dx_u}\right].$$

These equations show a remarkable feature. At the cost of an additional factor of $\left(\int F_u(x_u, \boldsymbol{x}_{N(u)}) dx_u\right)^{-1}$ we turned the equations requiring sam-

Algorithm 4.3 Algorithm for computing the fast marginalization approximation of the energy difference $W(b, \boldsymbol{x}_{U\setminus u}) - W(a, \boldsymbol{x}_{U\setminus u})$ using mixed projection. We assume that all variables $\boldsymbol{x}_U$ have specified bases ϕ_u of size $K_u = |\phi_u|$.

procedure MIXEDPROJECTION(ϕ_u, t_j)
  for all variables and quadrature nodes **do**
    $A_u(t_j) \leftarrow$ EMPTYMATRIX(K_u, K_u)
    $\boldsymbol{b}_u(t_j) \leftarrow$ EMPTYVECTOR(K_u)
  end for

  for all samples **do**
    $\boldsymbol{x}_V, w \leftarrow$ GETSAMPLE
    for all u and t_j **do**
      $w' \leftarrow w \Big/ \int F(\boldsymbol{x}_{V\setminus u}, x_u) dx_u$
      $\boldsymbol{v} \leftarrow$ EVALUATEBASIS$(\phi_u, \boldsymbol{x}_{N(u)})$
      $\boldsymbol{b}_u(t_j) \leftarrow \boldsymbol{b}_u(t_j) + w' \tilde{F}'(\boldsymbol{x}_{V\setminus u}, t_j)\boldsymbol{v}$
      $A_u(t_j) \leftarrow A_u(t_j) + w' \tilde{F}(\boldsymbol{x}_{V\setminus u}, t_j)\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}}$
    end for
  end for

  for all u and t_j **do**
    $\boldsymbol{c}_u(t_j) \leftarrow A_u^{-1}(t_j)\boldsymbol{b}_u(t_j)$
  end for

  for all u and t_j **do**
    $\boldsymbol{c}_u \leftarrow \sum_j w_j c_u(t_j)$
  end for
end procedure

pling from an extended model with continuous variable χ_u into the above, requiring only samples from the original distribution $P(\boldsymbol{x}_V)$. Therefore, the functions $A(\chi_u)$ and $\boldsymbol{b}(\chi_u)$ may be determined simultaneously using the same set of random samples, differing only in the χ_u dependent weights. The resulting procedure is summarized as Algorithm 4.3.

4.2.4 *Symmetrization*

Frequently, the probabilistic model has a great deal of symmetries due to various physical properties. These physical symmetries manifest them-

selves as symmetries of the Hamiltonian $W(\boldsymbol{x}_V)$. For example, the Ising Hamiltonian in the absence of an external magnetic field,

$$W_{\text{Ising}}(\boldsymbol{x}_V) = \frac{\mu}{2} \sum_{u \in V} x_u \sum_{v \in N(u)} x_v,$$

has even symmetry in $\boldsymbol{x}_V$, since $W_{\text{Ising}}(\boldsymbol{x}_V) = W_{\text{Ising}}(-\boldsymbol{x}_V)$. Thus, odd functions — such as linear, cubic or quintic polynomials — cannot appear in the expansion of $W_{\text{Ising}}(\boldsymbol{x}_V)$. Similarly, the marginal Hamiltonian $W(\boldsymbol{x}_U)$ defined for the Ising model

The lack of the minus sign is due to the change in integration limits.

$$\begin{aligned} W(\boldsymbol{x}_U) &= \ln \left(\int \mathrm{e}^{W_{\text{Ising}}(\boldsymbol{x}_U, \boldsymbol{x}_{V \setminus U})} d\boldsymbol{x}_{V \setminus U} \right) + \ln Z_U \\ &= \ln \left(\int \mathrm{e}^{W_{\text{Ising}}(-\boldsymbol{x}_U, -\boldsymbol{x}_{V \setminus U})} d\boldsymbol{x}_{V \setminus U} \right) + \ln Z_U \\ &= \ln \left(\int \mathrm{e}^{W_{\text{Ising}}(-\boldsymbol{x}_U, \boldsymbol{y}_{V \setminus U})} d\boldsymbol{y}_{V \setminus U} \right) + \ln Z_U \\ &= W(-\boldsymbol{x}_U) \end{aligned}$$

is also even. The differentiable extension $\tilde{W}(\chi_u, \boldsymbol{x}_{U \setminus u})$ similarly is even, making its derivative odd,

$$\begin{aligned} \frac{\partial \tilde{W}(-\chi_u, \boldsymbol{x}_{U \setminus u})}{\partial \chi_u} &= \frac{\partial(-\chi_u)}{\partial \chi_u} \frac{\partial \tilde{W}(-\chi_u, \boldsymbol{x}_{U \setminus u})}{\partial(-\chi_u)} \\ &= -\frac{\partial \tilde{W}(-\chi_u, \boldsymbol{x}_{U \setminus u})}{\partial(-\chi_u)} \\ &= -\frac{\partial \tilde{W}(\chi_u, \boldsymbol{x}_{U \setminus u})}{\partial \chi_u}. \end{aligned}$$

Therefore, the expansion of $\tilde{W}'(\chi_u, \boldsymbol{x}_{U \setminus u})$ may only consist of odd functions.

In the case of continuous variables we could simply remove the irrelevant, even functions from the basis ϕ; however, in the mixed projection approach we break this symmetry by fixing the value of χ_u. Thus the expansion of $\tilde{W}'(\chi_u, \boldsymbol{x}_{U \setminus u})$ in functions of $\boldsymbol{x}_{U \setminus u}$ is asymmetric. Consider the expansion

$$\tilde{W}'(\chi_u, \boldsymbol{x}_{U \setminus u}) = \sum_i c_i^{\text{e}}(\chi_u) \phi_i^{\text{e}}(\boldsymbol{x}_{U \setminus u}) + \sum_i c_i^{\text{o}}(\chi_u) \phi_i^{\text{o}}(\boldsymbol{x}_{U \setminus u}),$$

where we split the basis into even and odd parts ϕ_i^{e} and ϕ_i^{o}, respectively. Using the odd symmetry of $\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u})$,

$$\tilde{W}'(\chi_u, \boldsymbol{x}_{U\setminus u}) = -\tilde{W}'(-\chi_u, -\boldsymbol{x}_{U\setminus u}),$$

we obtain

$$\begin{aligned}\sum_i c_i^{\mathrm{e}}(\chi_u)\phi_i^{\mathrm{e}}(\boldsymbol{x}_{U\setminus u}) + \sum_i c_i^{\mathrm{o}}(\chi_u)\phi_i^{\mathrm{o}}(\boldsymbol{x}_{U\setminus u}) \\ = -\sum_i c_i^{\mathrm{e}}(-\chi_u)\phi_i^{\mathrm{e}}(-\boldsymbol{x}_{U\setminus u}) - \sum_i c_i^{\mathrm{o}}(-\chi_u)\phi_i^{\mathrm{o}}(-\boldsymbol{x}_{U\setminus u}).\end{aligned}$$

We eliminate $-\boldsymbol{x}_{U\setminus u}$ with the help of the even and odd symmetries of ϕ_i^{e} and ϕ_i^{o}, respectively, and equate terms to find

$$\begin{aligned}\phi_i^{\mathrm{e}}(\boldsymbol{x}_{U\setminus u}) : \quad & c_i^{\mathrm{e}}(\chi_u) = -c_i^{\mathrm{e}}(-\chi_u), \\ \phi_i^{\mathrm{o}}(\boldsymbol{x}_{U\setminus u}) : \quad & c_i^{\mathrm{o}}(\chi_u) = c_i^{\mathrm{o}}(-\chi_u).\end{aligned}$$

Therefore, the even basis function coefficients $c_i^{\mathrm{o}}(\chi_u)$ must be odd functions of χ_u, while the odd basis function coefficients $c_i^{\mathrm{o}}(\chi_u)$ must be even. Although the coefficients for the irrelevant even basis functions are nonzero, they integrate out to zero when the discrete Hamiltonian $W(\boldsymbol{x}_U)$ is reconstructed. They are important and must be included in the basis, yet their computation is an ultimately lost effort. In this section we discuss a way of symmetrizing the inner product and the projected function $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ to ensure that the irrelevant part is discarded and the odd symmetric part remains conserved.

The method described here amounts to splitting the function $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ into parts that are even and odd symmetric in χ_u,

$$\begin{aligned}\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) = \frac{1}{2}\left(\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) + \mathcal{F}(-\chi_u, \boldsymbol{x}_{N(u)})\right) \\ + \frac{1}{2}\left(\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) - \mathcal{F}(-\chi_u, \boldsymbol{x}_{N(u)})\right),\end{aligned}$$

then projecting the former, since we are interested in the coefficients $c_i^{\mathrm{e}}(\chi_u)$ that are even in χ_u. However, both could be projected and the resulting method remains applicable to general models, where it may be used to lower the computational cost of this method by a half. Presently we restrict ourselves to the case when the relevant part of $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ has an odd symmetry.

4.2.4.1 *Partial symmetrization*

The main part of the algorithm that we have assumed to be set in stone is the projected function. In all cases, we considered the approximation of

$$\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) = \frac{\partial \tilde{W}(\chi_u, \boldsymbol{x}_{U\setminus u})}{\partial \chi_u}$$

or the equivalent formula in the case of continuous variables. However, since the relevant part of $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ is even in χ_u, we may project only its even part,

$$\mathcal{F}^{\mathrm{e}}(\chi_u, \boldsymbol{x}_{N(u)}) = \frac{1}{2}\left(\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)}) + \mathcal{F}(-\chi_u, \boldsymbol{x}_{N(u)})\right).$$

The result is that the basis ϕ may be limited to include only odd functions of $\boldsymbol{x}_{N(u)}$, reducing the size of the matrix $A(\chi_u)$ by a factor of approximately four.

Define a correction factor $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$ through

$$\begin{aligned}\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)}) &= \frac{\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u})}{\tilde{P}(-\chi_u, \boldsymbol{x}_{U\setminus u})} \\ &= \exp\left(\int_{-\chi_u}^{\chi_u} W'(s, \boldsymbol{x}_{N(u)})ds\right).\end{aligned}$$

Substituting into the formula for the right hand side vector $\boldsymbol{b}(\chi_u)$, we obtain

$$\begin{aligned}b_i(\chi_u) = \frac{1}{2}\mathbb{E}\Bigg[&\frac{\phi_i(\boldsymbol{x}_{N(u)})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})}\frac{1}{\int F_u(x_u, \boldsymbol{x}_{N(u)})dx_u} \\ &\times\left(\frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)})}{\partial \chi_u}\right. \\ &\left.+\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})\frac{\partial \tilde{F}_u(-\chi_u, \boldsymbol{x}_{N(u)})}{\partial \chi_u}\right)\Bigg].\end{aligned} \tag{4.6}$$

Therefore, at the cost of computing a weight function of χ_u and the neighboring variables $\boldsymbol{x}_{N(u)}$ we may now project the odd symmetric part of $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$. This allows us to use only the odd polynomials as the basis, reducing the size of the basis by a factor of approximately two and, more

importantly, the size of the matrix $A(\chi_u)$ by a factor of approximately four.

4.2.4.2 *Full symmetrization*

We notice that the Equation 4.6 detailing the partial symmetrization of the projected function $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ is asymmetric in the correction. The symmetrizing term $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$ is only applied to the $\mathcal{F}(-\chi_u, \boldsymbol{x}_{N(u)})$ part of the formula, possibly leading to a bias due to the approximations used. In the present section we develop a fully symmetrized projection, where both the projected function $\mathcal{F}(\chi_u, \boldsymbol{x}_{N(u)})$ and the weight $\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u})$ are symmetrized.

The lack of symmetry in Equation 4.6 arises because the weight coming from the inner product contained a term $P(\chi_u, \boldsymbol{x}_{N(u)})$, i.e., was positive in χ_u. Therefore, let us employ an inner product even in χ_u, defined as

$$\begin{aligned}
\langle f, g \rangle_{\chi_u} &= \int f(s, \boldsymbol{x}_{U\setminus u}) g(s, \boldsymbol{x}_{U\setminus u}) \\
&\qquad \times \frac{\tilde{P}(s, \boldsymbol{x}_{U\setminus u})}{Q(s, \boldsymbol{x}_{U\setminus u})} \frac{\delta(s - \chi_u) + \delta(s + \chi_u)}{2} ds d\boldsymbol{x}_{U\setminus u} \\
&= \frac{1}{2} \int f(\chi_u, \boldsymbol{x}_{U\setminus u}) g(\chi_u, \boldsymbol{x}_{U\setminus u}) \\
&\qquad \times \left(\frac{\tilde{P}(\chi_u, \boldsymbol{x}_{U\setminus u})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} + \frac{\tilde{P}(-\chi_u, \boldsymbol{x}_{U\setminus u})}{Q(-\chi_u, \boldsymbol{x}_{U\setminus u})} \right) d\boldsymbol{x}_{U\setminus u}.
\end{aligned}$$

Because of the change in the inner product, the full symmetrization affects both the formulae for $A(\chi_u)$ and $\boldsymbol{b}(\chi_u)$. Beginning with $A(\chi_u)$, we obtain

$$\begin{aligned}
A_{ij}(\chi_u) = \frac{1}{2} \mathbb{E} \Bigg[& \frac{\phi_i(\boldsymbol{x}_{N(u)}) \phi_j(\boldsymbol{x}_{N(u)})}{\int F_u(x_u, \boldsymbol{x}_{N(u)}) dx_u} \\
& \left(\frac{\tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} + \frac{\tilde{F}_u(-\chi_u, \boldsymbol{x}_{N(u)})}{Q(-\chi_u, \boldsymbol{x}_{U\setminus u})} \right) \Bigg].
\end{aligned}$$

Similarly, we obtain the fully-symmetrized vector $\boldsymbol{b}(\chi_u)$ as

$$\begin{aligned}
b_i(\chi_u) = \frac{1}{4} \mathbb{E} \Bigg[& \frac{\phi_i(\boldsymbol{x}_{N(u)})}{\int F_u(x_u, \boldsymbol{x}_{N(u)}) dx_u} \\
& \times \left(\left(\frac{\mathcal{R}(-\chi_u, \boldsymbol{x}_{N(u)})}{Q(-\chi_u, \boldsymbol{x}_{U\setminus u})} + \frac{1}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} \right) \times \frac{\partial \tilde{F}_u(\chi_u, \boldsymbol{x}_{N(u)})}{\partial \chi_u} \right.
\end{aligned}$$

Table 4.1: Values of the renormalized coefficients obtained using no symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$	$c_4(\chi_u)$	$c_5(\chi_u)$
-0.949	0.250885	0.090069	-0.011387	-0.678121	0.031801
-0.742	0.269774	0.093073	-0.020850	-0.539799	0.026770
-0.406	0.294252	0.095408	-0.035109	-0.301255	0.015796
0.000	0.306469	0.095871	-0.042884	-0.000279	0.000014
0.406	0.294488	0.095376	-0.035348	0.300623	-0.015759
0.742	0.270104	0.093048	-0.021208	0.539073	-0.026720
0.949	0.251230	0.090058	-0.011773	0.677363	-0.031747
c_i	0.284457	0.094150	-0.029794	-0.000330	0.000020

$$+ \left(\frac{\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})}{Q(\chi_u, \boldsymbol{x}_{U\setminus u})} + \frac{1}{Q(-\chi_u, \boldsymbol{x}_{U\setminus u})} \right) \frac{\partial \tilde{F}_u(-\chi_u, \boldsymbol{x}_{N(u)})}{\partial \chi_u} \Bigg) \Bigg]. \tag{4.7}$$

Note that, by definition,

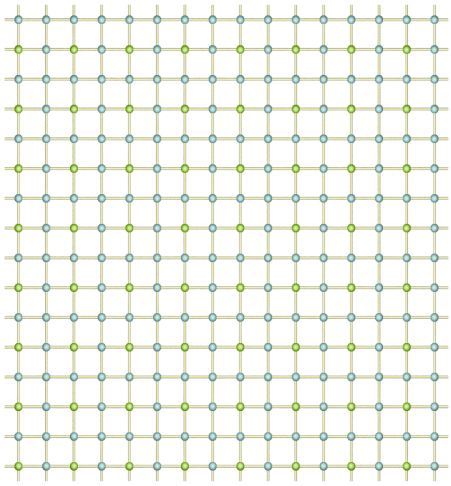
$$\mathcal{R}(-\chi_u, \boldsymbol{x}_{N(u)}) = \frac{1}{\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})},$$

requiring the computation of $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$ only once.

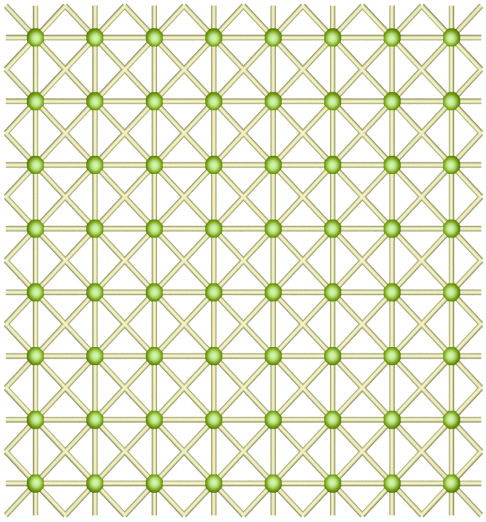
Unfortunately, there is a hidden cost to both partial and full symmetrization. The fast marginalization equation becomes implicit, because computing $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$ requires the knowledge of the solution, that is the expansion of $\hat{W}'(\chi_u, \boldsymbol{x}_{N(u)})$. We solve the resulting equation using a fixed-point iteration, repeating the projection with $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$ computed using the current guess of the expansion coefficients $\boldsymbol{c}(\chi_u)$.

4.2.4.3 *Performance of symmetrization*

We close the description of the symmetrization approaches by considering an example computation of the expansion coefficients. Consider the Ising model on a 16×16 lattice V at critical coupling $\mu_c = \ln\left(1+\sqrt{2}\right)/2$ and let the sub-lattice $U \subset V$ be the 8×8 lattice obtained by choosing nodes whose both coordinates are divisible by 2; both lattices are shown on Figure 4.16. We perform twelve steps of fixed-point iteration with



(a)



(b)

Figure 4.16: Visualization of the arrangement of nodes, showing (a) the original lattice V and (b) the sublattice U. The nodes $U \subset V$ are marked green on both images.

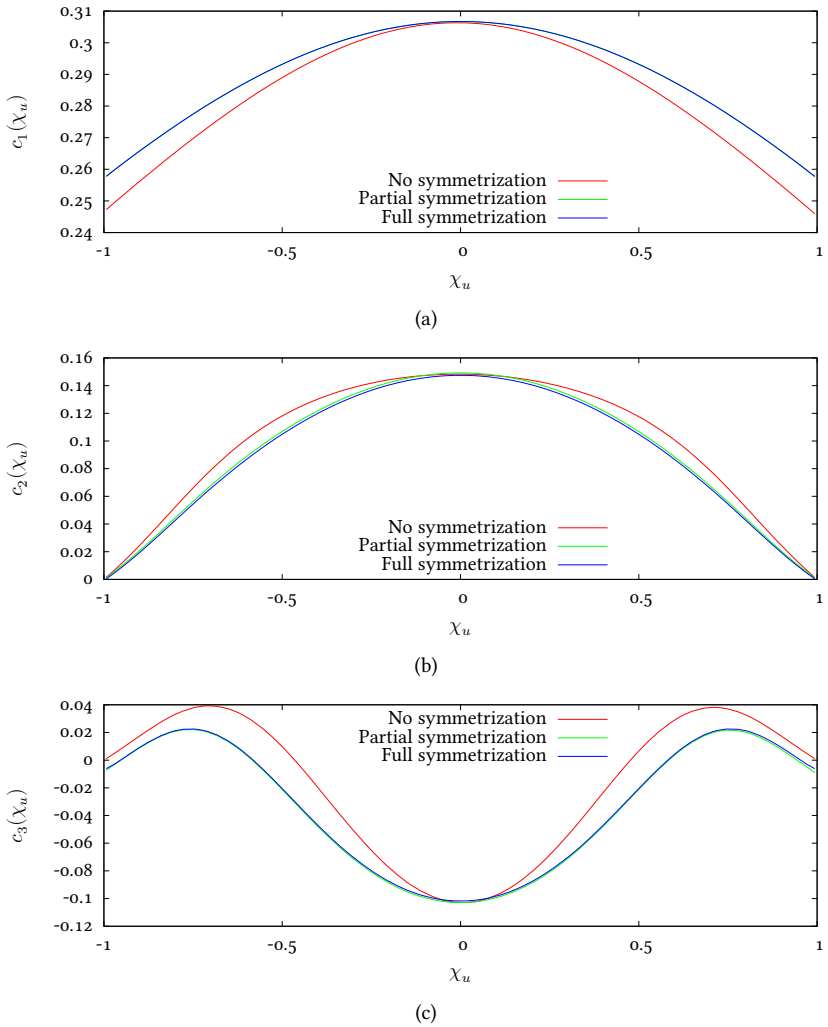


Figure 4.17: The χ_u dependence of selected basis function coefficients $c_i(\chi_u)$ under (a) no symmetrization, (b) partial symmetrization and (c) full symmetrization.

Robbins-Monro smoothing, with the results reported after the twelfth step.

We use the mixed projection method using (i) no symmetrization, (ii) partial symmetrization and (iii) full symmetrization, depending on the example. We compute the expansion coefficients $\boldsymbol{c}(\chi_u)$ at 84 integration nodes obtained by dividing the interval $\chi_u \in [-1, 1]$ into 21 subintervals and placing four Gaussian quadrature nodes within each, showing the shapes of the coefficients $\boldsymbol{c}(\chi_u)$. Additionally, we compute them sep-

Table 4.2: Values of the renormalized coefficients obtained using partial symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.261661	0.091275	-0.019786
-0.742	0.278050	0.093267	-0.027802
-0.406	0.297655	0.095232	-0.038125
0.000	0.306621	0.095976	-0.043125
0.406	0.297612	0.095249	-0.038100
0.742	0.277859	0.093342	-0.027709
0.949	0.261329	0.091409	-0.019640
c_i	0.289328	0.094348	-0.033881

Table 4.3: Values of the renormalized coefficients obtained using full symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.261325	0.091449	-0.019641
-0.742	0.277813	0.093390	-0.027706
-0.406	0.297516	0.095304	-0.038084
0.000	0.306509	0.096032	-0.043104
0.406	0.297516	0.095304	-0.038084
0.742	0.277813	0.093390	-0.027706
0.949	0.261325	0.091449	-0.019641
c_i	0.289198	0.094421	-0.033842

arately at seven Gaussian quadrature nodes, with values presented in Tables 4.1, 4.2 and 4.3.

Figure 4.17 clearly shows that the projected coefficients $\boldsymbol{c}(\chi_u)$ are symmetric. In particular, the irrelevant coefficients corresponding to even basis functions have odd symmetry, therefore integrate out to zero. However, in case of the partially and fully symmetrized schemes the irrelevant coefficients are eliminated completely. This is extremely important in face of the fact that the magnitude of the relevant coefficients is fre-

quently dwarfed by that of the irrelevant coefficients $c_4(\chi_u)$ and $c_5(\chi_u)$, cf. Table 4.1.

5

SAMPLING

Thus far we considered the task of computing the marginal probability density of a subset $\boldsymbol{x}_U$ of the original variables $\boldsymbol{x}_V$. We discussed the graphical underpinnings of that process and a numerical procedure for the efficient approximate computation of the marginal probability density $P(\boldsymbol{x}_U)$. We now turn to the task of using the marginal probabilities defining the approximate acyclic representation of the original probability density $P(\boldsymbol{x}_V)$ to generate random samples distributed according to $P(\boldsymbol{x}_V)$.

The acyclic representation is a tool that can be used to generate samples in numerous ways. Instead of trying to describe all possibilities, we will instead explain in detail a method most closely resembling that of Chorin (2008) and suggest variations that enhance it in various ways. However, the importance sampling approach remains the core of nearly all methods considered in this chapter.

The chapter is structured as follows. We begin with the importance sampling method of Chorin (2008), analyzing the method in detail in Section 5.1. Chorin's method is as an example of Sequential Importance Sampling (SIS) and we describe a method of improving the weight distribution by particle filtering in Section 5.2, closing with a description of a practical Partial Rejection Control (PRC) algorithm in Section 5.2.5. In Section 5.3 we discuss a widely different approach, sampling the acyclic representation of the probability distribution $P(\boldsymbol{x}_V)$ using Markov Chain Monte Carlo (MCMC). We begin with a straightforward transformation of the importance sampling scheme into an MCMC algorithm in 5.3.1 and continue to describe a generalized Gibbs sampler in 5.3.2. Finally, we close with a discussion of the described sampling methods.

5.1 IMPORTANCE SAMPLING

We begin with the Chainless Monte Carlo (ChMC) method of Chorin (2003, 2008) and Okunev (2005). Their method relies on the fact that the variables $\boldsymbol{x}_{V_i \setminus V_{i+1}}$ are conditionally independent given $\boldsymbol{x}_{V_{i+1}}$. The variables may be separated and sampled individually, making this method potentially very efficient. The ChMC method consists of two parts: the ascent involving computation of the approximate marginal densities, followed by the de-

scent phase where the individual variables are sampled, starting with the top of the lattice ladder.

The goal of the sampling routine is to generate a vector $\boldsymbol{x}_V$ with probability $P(\boldsymbol{x}_V)$. The approach we take is that of importance sampling (Liu, 2001; Robert and Casella, 2004): we will produce a state $\boldsymbol{x}_V$ with probability $P_{\approx}(\boldsymbol{x}_V)$, which hopefully is close to the original probability $P(\boldsymbol{x}_V)$ in the sense that

$$w(\boldsymbol{x}_V) = P(\boldsymbol{x}_V) \Big/ P_{\approx}(\boldsymbol{x}_V) \approx 1.$$

The ratio $w(\boldsymbol{x}_V)$ is called the weight of the sample $\boldsymbol{x}_V$ and represents the amount of mismatch between the target distribution $P(\boldsymbol{x}_V)$ and the trial distribution $P_{\approx}(\boldsymbol{x}_V)$.

Starting from the top lattice V_m, we generate a random sample

$$\boldsymbol{x}_{V_m} \sim \hat{P}(\boldsymbol{x}_{V_m})$$

using a method of our choice. The approximate marginal distribution $\hat{P}(\boldsymbol{x}_{V_m})$ is known, having been computed during the ascent phase, and the size of the final lattice $|V_m|$ is expected to be small compared to that of the original $|V|$. Therefore, even though the probability distribution $\hat{P}(\boldsymbol{x}_{V_m})$ contains circular dependencies between variables, it may be sampled efficiently. Depending on the size of the final lattice, we may use one of two approaches:

- direct sampling: list all possible states of the variables $\boldsymbol{x}_{V_m}$ and compute their normalized probabilities $\hat{P}(\boldsymbol{x}_{V_m})$; then, choose one of those states at random with the appropriate probability;
- Monte Carlo: sample states $\boldsymbol{x}_{V_m}$ using Markov Chain Monte Carlo (MCMC) and stop the updates at regular intervals; for each generated state $\boldsymbol{x}_{V_m}$ sample the remainder of the variables $\boldsymbol{x}_{V \setminus V_m}$ using the conditional probabilities.

The choice between the two depends on the size of the final lattice. As the number of possible states grows rapidly with the number of dimensions, the direct sampling method quickly becomes infeasible; it is also difficult to use with continuous variables. Markov Chain Monte Carlo (MCMC), on the other hand, is very efficient for small systems and handles both continuous and discrete systems. MCMC in the simplest case requires only the ability to compute

$$\alpha(\boldsymbol{x}_{V_m}, \boldsymbol{y}_{V_m}) = \hat{P}(\boldsymbol{y}_{V_m}) \Big/ \hat{P}(\boldsymbol{x}_{V_m}) = \exp\left(\Delta W(\boldsymbol{x}_{V_m}, \boldsymbol{y}_{V_m})\right),$$

where $\boldsymbol{x}_{V_m}$ and $\boldsymbol{y}_{V_m}$ differ by a single component $\boldsymbol{x}_u$, $u \in V_m$; therefore, MCMC can directly use the representation of the marginal distribution $P(\boldsymbol{x}_{V_m})$ produced by the fast marginalization algorithm described in Chapter 4.

We obtain a state $\boldsymbol{x}_{V_m}$ together with the probability of generating it, $P_{\approx}^{|V_m|}(\boldsymbol{x}_{V_m}) = \hat{P}(\boldsymbol{x}_{V_m})$, or an approximation of the latter if MCMC is used. From there we begin an iterative descent down the Directed Acyclic Graph (DAG) $D = (V, A)$, with variables sampled in the partial order implied by D. Let a particular topological ordering be an ordered sequence of nodes $u_i \in V$, $T = (u_1, u_2, \dots, u_{|V|})$, and denote subsets of the first i nodes in the ordering as $T_i = (u_1, u_2, \dots, u_i)$. For $i = |V_m| + 1, |V_m| + 2, \dots, |V|$, we sample the random variable x_{u_i} from

$$x_{u_i} \sim \hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) \tag{5.1}$$

where $N_p(u_i)$ is the set of direct predecessors of the node u_i in the directed graph $D = (V, A)$, with the property that $u_j \in N_p(u_i)$ implies $u_j \leq u_i$. Simultaneously we update $P_{\approx}^i(\boldsymbol{x}_{T_i})$ via

$$P_{\approx}^i(\boldsymbol{x}_{T_i}) = \hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) P_{\approx}^{i-1}(\boldsymbol{x}_{T_{i-1}}),$$

where the conditional probability $\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)})$ is an approximation of the true conditional probability of x_{u_i} given all the variables that came before it in the topological ordering T. After sampling the entire set of variables we obtain the probability of generating the state $\boldsymbol{x}_V$ using our sampling method, with the trial probability

$$\begin{aligned} P_{\approx}(\boldsymbol{x}_V) &= P_{\approx}^{|V|}(\boldsymbol{x}_V) \\ &= \hat{P}(\boldsymbol{x}_{V_m}) \prod_{i=|V_m|+1}^{|V|} \hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}). \end{aligned}$$

The values $P_{\approx}(\boldsymbol{x}_V)$ are unnormalized, with the unknown normalizing constant $Z_{\approx}$ equal to the normalizing constant of the unnormalized density $\hat{P}(\boldsymbol{x}_{V_m})$. This is due to the fact that the conditional densities used to sample the variables u_i are properly normalized, thus the normalization constant $\hat{Z}_{V_m}$ of the final marginal density $\hat{P}(\boldsymbol{x}_{V_m})$ is propagated down to the fine lattice. Therefore, as long as the method of computing $\hat{P}(\boldsymbol{x}_{V_m})$ produces values normalized to the same normalization constant $\hat{Z}_{V_m}$ independent of $\boldsymbol{x}_{V_m}$ — that is if the values of $\hat{P}(\boldsymbol{x}_{V_m})$ are consistent — the

Algorithm 5.1 Sequential importance sampling algorithm.

function SISAMPLING($D = (V, A),\ P(\boldsymbol{x}_{V_m}),\ \hat{P}(x_u \mid \boldsymbol{x}_{N_p(u)})$)
  $\boldsymbol{x}_{V_m} \sim \hat{P}(\boldsymbol{x}_{V_m})$
  $P_{\approx}^{|V_m|} \leftarrow \hat{P}(\boldsymbol{x}_{V_m})$
  $T = \left(u_{|V_m|+1}, u_{|V_m|+2}, \dots, u_{|V|}\right) \leftarrow$ TOPOLOGICALORDER(D)
  for $i = |V_m| + 1, |V_m| + 2, \dots, |V|$ **do**
    $x_{u_i} \sim \hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)})$
    $P_{\approx}^{i} \leftarrow P_{\approx}^{i-1} \hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)})$
  end for
  $w(\boldsymbol{x}_V) \leftarrow \dfrac{P(\boldsymbol{x}_V)}{P_{\approx}^{|V|}}$
  return $(\boldsymbol{x}_V, w(\boldsymbol{x}_V))$
end function

resulting trial probability will also be consistent, so that they can be used to compute weights in the importance sampling scheme.

We stress at this point the importance of the fact that the variables are conditionally independent of each other, and thus can be sampled individually. If they were not, as indeed they are not when one samples via more general renormalization methods such as the majority rule used by Brandt and Ron (2001b) and Ron and Swendsen (2001), the computation of the conditional probability $\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)})$ would not be possible, as the normalization factor would be computationally intractable.

Once all variables are sampled, we proceed with the correction of the trial probability density. The weight

$$w(\boldsymbol{x}_V) = \frac{P(\boldsymbol{x}_V)}{P_{\approx}(\boldsymbol{x}_V)}$$

is computed and used to correct the expected value. For a sequence of random states $\boldsymbol{x}_V^i$ is generated with weights $w(\boldsymbol{x}_V^i)$, the expected value of a function $f(\boldsymbol{x}_V)$ becomes

$$\mathbb{E}\left[f\right] = \frac{\sum_i f(\boldsymbol{x}_V^i) w(\boldsymbol{x}_V^i)}{\sum_i w(\boldsymbol{x}_V^i)},$$

where the sum of the weights acts to counter the fact that neither the trial nor the target probabilities are properly normalized.

5.1.1 *Analysis of the weights*

In preparation for the coming Section 5.2 that describes a method for improving the basic method outlined above, let us pause and analyze the process of generating the state $\boldsymbol{x}_V$, and especially its trial probability. This is interesting because the state $\boldsymbol{x}_V$ is constructed by exchanging information between a collection of unrelated probability densities.

The quantity $P^i_{\approx}(\boldsymbol{x}_{T_i})$ that is computed together with the variables is the probability of sampling the partially complete state $\boldsymbol{x}_{T_i}$. It is an approximation of the marginal probability $P(\boldsymbol{x}_{T_i})$ of the variables that precede u_i in the ordering T and x_{u_i}, however one's inability to compute the marginal exactly even for known values of $\boldsymbol{x}_{T_i}$ makes it impossible to judge the quality of the approximation. Only the complete sample $\boldsymbol{x}_V$ has a computable weight $w(\boldsymbol{x}_V)$.

Although it might look like the weight appears suddenly at the end of the sampling process, this is not true. Let us look in more detail at the process of updating the trial probability and the errors committed at each step. In terms of the exact marginal probabilities,

$$P(\boldsymbol{x}_{T_i}) = P(x_{u_i} \mid \boldsymbol{x}_{T_{i-1}}) P(\boldsymbol{x}_{T_{i-1}}), \tag{5.2}$$

while the approximate version used by the update of the trial density takes the form

$$P^i_{\approx}(\boldsymbol{x}_{T_i}) = \hat{P}(x_{u_i} \mid \boldsymbol{x}_{P(u_i)}) P^{i-1}_{\approx}(\boldsymbol{x}_{T_i}). \tag{5.3}$$

Write the approximations to the marginal densities $P(\boldsymbol{x}_{T_i})$ and $P(\boldsymbol{x}_{T_{i-1}})$ as $\hat{P}(\boldsymbol{x}_{T_i})$ and $\hat{P}(\boldsymbol{x}_{T_{i-1}})$, respectively. Let $P^{i-1}_{\approx}(\boldsymbol{x}_{T_i}) = \hat{P}(\boldsymbol{x}_{T_{i-1}})$, meaning that the state $\boldsymbol{x}_{T_{i-1}}$ was sampled from $\hat{P}(\boldsymbol{x}_{T_{i-1}})$. The only discrepancy between $\hat{P}(\boldsymbol{x}_{T_i})$ and $P^i_{\approx}(\boldsymbol{x}_{T_i})$ will thus come from the error due to a mismatch between $\hat{P}(\boldsymbol{x}_{T_i})$ and $\hat{P}(\boldsymbol{x}_{T_{i-1}})$.

Consider computing a weight $w_i(\boldsymbol{x}_{T_i})$ that measures how well $P^i_{\approx}(\boldsymbol{x}_{T_i})$ approximates $\hat{P}(\boldsymbol{x}_{T_i})$, or equivalently, what correction factor must be applied to the state due to the mismatch between the two approximations $\hat{P}(\boldsymbol{x}_{T_i})$ and $\hat{P}(\boldsymbol{x}_{T_{i-1}})$. Using the definition of $P^i_{\approx}(\boldsymbol{x}_{T_i})$, we obtain

$$w_i(\boldsymbol{x}_{T_i}) = \frac{\hat{P}(\boldsymbol{x}_{T_{i-1}})}{P^i_{\approx}(\boldsymbol{x}_{T_i})} = \frac{\hat{P}(\boldsymbol{x}_{T_i})}{\hat{P}(x_{u_i} \mid \boldsymbol{x}_{P(u_i)}) P^{i-1}_{\approx}(\boldsymbol{x}_{T_i})}. \tag{5.4}$$

Because the conditional probability is computed using the approximation $\hat{P}(\boldsymbol{x}_{T_i})$, we obtain through Bayes' formula that

$$\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) = \frac{\hat{P}(\boldsymbol{x}_{T_i})}{\int \hat{P}(\boldsymbol{x}_{T_i}) dx_{u_i}}, \tag{5.5}$$

the integral term representing the exact marginal of $\hat{P}(\boldsymbol{x}_{T_i})$. Substituting back to Equation 5.4 we find

$$\begin{aligned} w_i(\boldsymbol{x}_{T_i}) &= \frac{\hat{P}(\boldsymbol{x}_{T_i})}{\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) P_{\approx}^{i-1}(\boldsymbol{x}_{T_i})} \\ &= \frac{\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) \int \hat{P}(\boldsymbol{x}_{T_i}) dx_{u_i}}{\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)}) P_{\approx}^{i-1}(\boldsymbol{x}_{T_i})} \\ &= \frac{\int \hat{P}(\boldsymbol{x}_{T_i}) dx_{u_i}}{P_{\approx}^{i-1}(\boldsymbol{x}_{T_i})}. \end{aligned}$$

Therefore, the weight $w_i(\boldsymbol{x}_{T_i})$ that has to be applied due to the mismatch between the successful marginals is the ratio of the exact marginal of the approximation $\hat{P}(\boldsymbol{x}_{T_i})$ and the separate approximation $\hat{P}(\boldsymbol{x}_{T_{i-1}})$. In fact, looking slightly differently at the trial density update formula

$$\begin{aligned} P_{\approx}^{i}(\boldsymbol{x}_{T_i}) &= \hat{P}(x_{u_i} \mid \boldsymbol{x}_{P(u_i)}) P_{\approx}^{i-1}(\boldsymbol{x}_{T_i}) \\ &= \left(\hat{P}(\boldsymbol{x}_{T_i}) \Big/ \int \hat{P}(\boldsymbol{x}_{T_i}) dx_{u_i} \right) P_{\approx}^{i-1}(\boldsymbol{x}_{T_i}) \\ &= \hat{P}(\boldsymbol{x}_{T_i}) \left(P_{\approx}^{i-1}(\boldsymbol{x}_{T_i}) \Big/ \int \hat{P}(\boldsymbol{x}_{T_i}) dx_{u_i} \right) \\ &= \hat{P}(\boldsymbol{x}_{T_i}) / w_i(\boldsymbol{x}_{T_i}), \end{aligned}$$

we see that the update formula recognizes the fact that the two approximations do not agree and applies a correcting weight, which is recovered in Equation 5.5. Extending this equation to the full sample $\boldsymbol{x}_V$ we find

$$P_{\approx}(\boldsymbol{x}_V) = P(\boldsymbol{x}_V) \left(\prod_i w_i(\boldsymbol{x}_{T_i}) \right)^{-1},$$

here the trial density becomes a product of the corrective weights collected during the sampling. Indeed, the product of these becomes the final weight

$$w(\boldsymbol{x}_V) = \frac{P(\boldsymbol{x}_V)}{P_{\approx}(\boldsymbol{x}_V)} = \prod_i w_i(\boldsymbol{x}_{T_i}).$$

Because the errors are multiplicative, it is easy to see that the total weight grows exponentially with the number of variables. Therefore, the errors must be caught early and not allowed to plague the later computation.

5.2 PARTICLE FILTERING

The sampling described in Section 5.1 can be seen as an example of Sequential Importance Sampling (SIS), with the ordering index i serving as discrete time. We may therefore use the particle filtering methods to improve the weights as they appear using one of the many algorithms developed in that field (Doucet, de Freitas, and Gordon, 2001). In this section we set up a common framework for performing particle filtering and describe two particular algorithms, the Sequential Importance Resampling (SIR) and Partial Rejection Control (PRC).

5.2.1 *Sequential Importance Sampling*

Consider the partial sample $\boldsymbol{x}_{T_i}$ and assume that we have computed a probability $\hat{P}_*(\boldsymbol{x}_{T_i})$ that approximates the exact marginal density $P(\boldsymbol{x}_{T_i})$. We could compute a weight

The precise choice of $\hat{P}_*$ will be left for later, but the goal is to use a function more accurate than $P_{\approx}$.

$$w_*(\boldsymbol{x}_{T_i}) = \frac{\hat{P}_*(\boldsymbol{x}_{T_i})}{P_{\approx}(\boldsymbol{x}_{T_i})},$$

obtaining a measure of the correction that needs to be applied to the state $\boldsymbol{x}_{T_i}$. We only assume that the weight can be computed for a specific value of $\boldsymbol{x}_{T_i}$.

Let each sample we generate by the implicit sampling be a particle, indexed by a parameter $j = 1, 2, \dots, M$ as $\boldsymbol{x}^j_{T_i}$. Thus far we have been generating these particles separately, effectively using $M = 1$, but consider executing the sampling algorithm in an almost unchanged form, simultaneously for $M \geq 1$ at a time. At the coarsest level we generate M samples

$$\boldsymbol{x}^j_{V_m} \sim \hat{P}(\boldsymbol{x}^j_{V_m})$$

and perform the conditional sampling separately, using Equation 5.1 to reach a set of particles $\boldsymbol{x}_{T_i}^j \sim P_{\approx}(\boldsymbol{x}_{T_i}^j)$. Because we know an approximation $\hat{P}_*(\boldsymbol{x}_{T_i}^j)$ of the true marginal $P(\boldsymbol{x}_{T_i}^j)$, we may compute weights for each particle

$$w_*(\boldsymbol{x}_{T_i}^j) = \frac{\hat{P}_*(\boldsymbol{x}_{T_i}^j)}{P_{\approx}(\boldsymbol{x}_{T_i}^j)}.$$

Were the trial $P_{\approx}(\boldsymbol{x}_{T_i}^j)$ and corrective distributions $\hat{P}_*(\boldsymbol{x}_{T_i}^j)$ exact, the weights would be all equal to one another. In practice, the weights will be spanning a wide range of values, with particles having high weights being under-sampled by the trial distribution $P_{\approx}(\boldsymbol{x}_{T_i}^j)$ and those with low weights being over-sampled. To correct this imbalance we will perform resampling, that is we will remove particles that were oversampled and, in their stead, place copies of those particles that were under-sampled.

The resampling procedure should be seen as sampling from an approximation of $\hat{P}_*(\boldsymbol{x}_{T_i})$. The particles and their attached weights define a discrete approximation $\bar{P}_*(\boldsymbol{x}_{T_i})$ of the target distribution $\hat{P}_*(\boldsymbol{x}_{T_i})$,

$$\bar{P}_*(\boldsymbol{x}_{T_i}) = \sum_{j=1}^{M} w_*(\boldsymbol{x}_{T_i}^j) P_{\approx}(\boldsymbol{x}_{T_i}^j) \delta(\boldsymbol{x}_{T_i} - \boldsymbol{x}_{T_i}^j).$$

Note that this approximation is non-zero only for $\boldsymbol{x}_{T_i}$ equal to one of the particles $\boldsymbol{x}_{T_i}^j$. Through resampling, we obtain an updated collection of particles that follow $\bar{P}_*(\boldsymbol{x}_{T_i})$, which in the limit of $M \to \infty$ is equivalent to $\hat{P}_*(\boldsymbol{x}_{T_i})$.

The discrete nature of the approximate distribution $\bar{P}_*(\boldsymbol{x}_{T_i})$ means that the resampling step will lead to a less diverse set of particles, because states $\boldsymbol{x}_{T_i}$ that are not among those represented by the particles $\boldsymbol{x}_{T_i}^j$ will not be sampled at all. We still benefit from the resampling, however, because the initially exact copies will differentiate during the subsequent sampling steps when the remaining variables $\boldsymbol{x}_{T \setminus T_i}$ are determined.

There are multiple algorithms for performing resampling, but they all attempt to do the same thing, namely to reduce the variance of weights by sampling from the discrete approximation of the target distribution (Doucet, de Freitas, and Gordon, 2001). We summarize two of those algorithms below, beginning with the SIR algorithm.

5.2.2 *Sequential Importance Resampling*

The Sequential Importance Resampling (SIR) algorithm adds a resampling stage to the SIS, which partially corrects the weights by sampling from a discrete approximation of the target distribution. Following Section 11.3.1 of Liu, Chen, and Logvinenko (2001), we choose the transformed weight $a_*(\boldsymbol{x}_{T_i}^j)$ to be a monotone increasing function of $w_*(\boldsymbol{x}_{T_i}^j)$. This flexibility allows for balancing the needs of reducing weight variance and keeping a diverse set of particles; thus, a generic choice is the square root transform $a_*(\boldsymbol{x}_{T_i}^j) = \sqrt{w_*(\boldsymbol{x}_{T_i}^j)}$. The resampling stage replaces the original collection of particles $\boldsymbol{x}_{T_i}^j$ with $\boldsymbol{y}_{T_i}^k$ by selecting M particles from among the original particles $\boldsymbol{x}_{T_i}^j$ with probability dependent on the transformed weights.

5.2.2.1 *Reallocation*

One method is reallocation, where the resampled particles $\boldsymbol{y}_{T_i}^k$ are formed by choosing them at random from among the $\boldsymbol{x}_{T_i}^j$. For each particle we compute an effective fraction q_j, that is what fraction the M particles it is worth, given by

$$q_j = \frac{a_*(\boldsymbol{x}_{T_i}^j)}{\sum_{j=1}^{M} a_*(\boldsymbol{x}_{T_i}^j)}.$$

A particle j is intuitively worth Mq_j particles, but it is important to handle correctly the cases when $Mq_j < 1$.

Following Liu, Chen, and Logvinenko (2001), for $Mq_j \geq 1$ we keep $\lfloor Mq_j \rfloor$ copies of the particle $\boldsymbol{x}_{T_i}^j$ and assign an updated weight $w_*(\boldsymbol{x}_{T_i}^j)/\lfloor Mq_j \rfloor$ to each copy. In case $Mq_j < 1$, we keep the particle with probability Mq_j; if the particle survives, we assign it an updated weight $w_*(\boldsymbol{x}_{T_i}^j)/Mq_j$.

5.2.2.2 *Low variance resampling*

The downside of the previous method is that the number of particles fluctuates after resampling, complicating the code by requiring a form of control to ensure that a similar number is generated each time.

Instead, we will choose M particles randomly from among the existing ones. Each particle will have a probability of being selected equal to q_j

Algorithm 5.2 Reallocation particle filtering algorithm for improving the quality of the batch of particles X_{T_i}.

function REALLOCATION($X_{T_i} = \left\{(\boldsymbol{x}^j_{T_i}, P^i_{\approx}(\boldsymbol{x}^j_{T_i})\right\}$)
  $Y_{T_i} \leftarrow \emptyset$
  for all j **do**
    $w_*(\boldsymbol{x}^j_{T_i}) \leftarrow \dfrac{\widehat{P}_*(\boldsymbol{x}^j_{T_i})}{P_{\approx}(\boldsymbol{x}^j_{T_i})}$
    $a_*(\boldsymbol{x}^j_{T_i}) \leftarrow \sqrt{w_*(\boldsymbol{x}^j_{T_i})}$
  end for

  for all j **do**
    $q_j \leftarrow \dfrac{a_*(\boldsymbol{x}^j_{T_i})}{\sum_{j=1}^{M} a_*(\boldsymbol{x}^j_{T_i})}$
    if $q_j \geq 1$ **then**
      for $k = 1, 2, \ldots, \lfloor q_j \rfloor$ **do**
        $Y_{T_i} \leftarrow Y_{T_i} \cup \left(x^j_{T_i}, w_*(\boldsymbol{x}^j_{T_i})/\lfloor q_j \rfloor\right)$
      end for
    else
      $p \sim U[0,1]$
      if $p < q_j$ **then**
        $Y_{T_i} \leftarrow Y_{T_i} \cup \left(x^j_{T_i}, w_*(\boldsymbol{x}^j_{T_i})/q_j\right)$
      end if
    end if
  end for

  return Y_{T_i}
end function

defined as above. However, choosing the resampled particles entirely randomly does not always work well. Consider for example the case where all weights are equal to each other. Nothing should change because $q_j = 1/M$ and each particle should be sampled once, but due to the stochastic nature of the process some particles will be invariably selected twice while some others won't be selected at all.

To combat these potential issues we will select the M new particles in the following way. The q_j's specify the probabilities of the M particles and we put them together in a box. We select M particles by choosing a

random number $p \in U[0, 1/M]$ and selecting particles whose parts contain the numbers

$$r,\ r + 1/M,\ r + 2/M, \dots,\ r + k/M, \dots, ,\ r + (M-1)/M.$$

As a result, each particle is selected with the appropriate probability yet the algorithm behaves correctly in the limit of all equal weights. The weights of the resampled particles are updated by dividing the weight of the original sample $w_*(\boldsymbol{x}^j_{T_i})$ by the probability of choosing it Mq_j. If the resampled particle $\boldsymbol{y}^k_{T_i}$ corresponds to an original particle $\boldsymbol{x}^j_{T_i}$, we obtain

$$w_*(\boldsymbol{y}^k_{T_i}) = \frac{w_*(\boldsymbol{x}^j_{T_i})}{a_*(\boldsymbol{x}^j_{T_i})}.$$

5.2.2.3 *Trial density update*

Once the resampled particles are selected we need to compute the probability of generating the resampled particles using the algorithm, $P^i_{\approx}(\boldsymbol{y}^k_{T_i})$. Consider a resampled particle $\boldsymbol{y}^k_{T_i}$ which is a copy of the particle $\boldsymbol{x}^j_{T_i}$.

In the reallocation algorithm, for $Mq_j \geq 1$ we obtain $\lfloor Mq_j \rfloor$ copies of the particle, thus the probability of generating the state $\boldsymbol{y}^k_{T_i} = \boldsymbol{x}^j_{T_i}$ increased $\lfloor Mq_j \rfloor$-fold:

$$P^i_{\approx}(\boldsymbol{y}^k_{T_i}) = \lfloor Mq_j \rfloor P^i_{\approx}(\boldsymbol{x}^j_{T_i}).$$

On the other hand, when $Mq_j < 1$, the particle survives with probability Mq_j, reducing the trial probability

$$P^i_{\approx}(\boldsymbol{y}^k_{T_i}) = Mq_j P^i_{\approx}(\boldsymbol{x}^j_{T_i}).$$

Both formulas are very similar and reflect the changed weight of the particle. Therefore, in the case of low variance resampling, we simply obtain

$$P^i_{\approx}(\boldsymbol{y}^k_{T_i}) = Mq_j P^i_{\approx}(\boldsymbol{x}^j_{T_i}),$$

reflecting the fact that the expected number of copies of the state $\boldsymbol{x}^j_{T_i}$ is Mq_j.

5.2.3 *Rejection control*

The reallocation and resampling algorithms invariably suffer from particle impoverishment, because the resampling stage reduces the diversity of generated samples. This process is referred to as particle degeneracy and in extreme situations causes the batch to collapse onto a single particle (Berzuini et al., 1997; Gordon, Salmond, and Ewing, 1995). The above methods address this issue using the transformed weights $a_*(\boldsymbol{x}^j_{T_i})$, which reduce the weight variance less aggressively than the weights $w_*(\boldsymbol{x}^j_{T_i})$; however, this change does not eliminate the particle impoverishment problem completely.

Here we describe two related algorithms for eliminating poor samples that do not produce multiple copies of existing states. Instead, they conditionally reject particles deemed to be of low weight and replace them with new ones. The most common variants, the Full Rejection Control (FRC) and Partial Rejection Control (PRC), differ in the way the new particles are created: the FRC recreates the lost particles from scratch, while PRC from the most recent check-point. Because the FRC algorithm is very costly, we describe here the more moderate PRC (Liu, Chen, and Logvinenko, 2001, p. 233).

5.2.3.1 *Partial rejection control*

Similarly to the Sequential Importance Resampling (SIR) described above, consider the particles $\boldsymbol{x}^j_{T_i}$ at a check-point where the weights

$$w_*(\boldsymbol{x}^j_{T_i}) = \frac{\hat{P}_*(\boldsymbol{x}^j_{T_i})}{P^i_{\approx}(\boldsymbol{x}^j_{T_i})}$$

may be computed. Select a weight threshold c_i and accept particles with probability

$$q_j = \min\left(1, w_*(\boldsymbol{x}^j_{T_i}) \Big/ c_i\right).$$

The accepted particles are assigned an updated weight

$$w'_*(\boldsymbol{x}^j_{T_i}) = \max\left(w_*(\boldsymbol{x}^j_{T_i}), c_i\right).$$

Any rejected particle $\boldsymbol{x}^j_{T_i}$ is then replaced with a regenerated particle $\boldsymbol{x}'_{T_i}$. We begin by selecting a partial particle $\boldsymbol{x}^k_{T_l}$ from a previous check-point l at random, with probability proportional to its weight after that

checkpoint $w_*(\boldsymbol{x}^k_{T_l})$. This partial regenerated particle is assigned the trial probability

$$P^l_{\approx}(\boldsymbol{x}'_{T_l}) = \frac{M}{\sum_{m=1}^{M} w_*(\boldsymbol{x}^m_{T_l})} P^l_{\approx}(\boldsymbol{x}^k_{T_l}).$$

The missing variables $\boldsymbol{x}^j_{T_i \setminus T_l}$ are sampled, producing a fully regenerated particle $\boldsymbol{x}'_{T_i}$ with trial probability $P^i_{\approx}(\boldsymbol{x}'_{T_i})$. The regenerated particle replaces the rejected particle $\boldsymbol{x}^j_{T_i}$ and undergoes rejection control at the check-point T_i. The regeneration process is repeated until a particle is accepted, thus keeping a constant batch size M.

The PRC algorithm improves upon the simple resampling algorithm, because it goes back to previously accepted particles to construct the resampled particle set instead of using particles present at the check-point. At the cost of significantly increased computation time, the FRC, which regenerates the rejected particles from scratch, produces nearly independent, non-degenerate particles.

5.2.4 *Dense marginal probabilities*

The resampling stage described above attempts to correct the trial probability density $P^i_{\approx}(\boldsymbol{x}_{T_i})$ so that it is closer to the density $\widehat{P}_*(\boldsymbol{x}_{T_i})$. It is imperative that the corrective density $\widehat{P}_*(\boldsymbol{x}_{T_i})$ be closer to the true marginal density $P(\boldsymbol{x}_{T_i})$ than the trial density, as otherwise the resampling step will actually make the matters worse.

The quality of an approximation to the marginal density is directly related to the number of basis functions used by the approximation. Typically, the bigger the basis the better the approximation may potentially be, but the size of the basis is severely limited by the need to keep the dependency graph as sparse as possible. Since sparse dependency graphs are necessary to prevent the dependency graph from quickly becoming a clique, forming a denser proposal density is very difficult. However, the approximation $\widehat{P}_*(\boldsymbol{x}_{T_i})$ is only to be evaluated, rather than used to sample using conditional probabilities, thus the dependency graph of $\widehat{P}_*(\boldsymbol{x}_{T_i})$ may be arbitrarily dense and thus more accurate than $P^i_{\approx}(\boldsymbol{x}_{T_i})$.

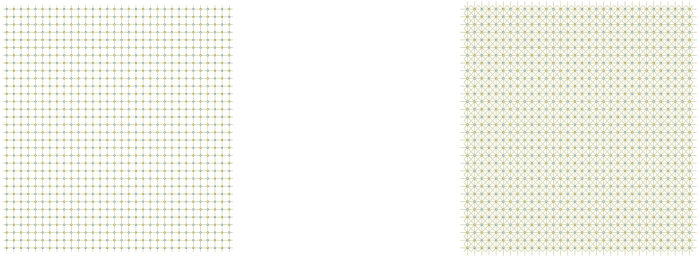
Consider the simple case of a two-dimensional Cartesian lattice and let the blue variables in Figure 5.18 be already sampled. When sampling the yellow variables, the densest dependency graph that still respects the conditional independence between yellow variables given the blue variables

Algorithm 5.3 Partial rejection control algorithm for improving the quality of a batch of particles X_{T_i}, regenerating the particles from an earlier checkpoint T_l, where the batch of particles was X_{T_l}.

function PARTIALREJECTIONCONTROL(X_{T_i}, X_{T_l})
  $Y_{T_i} \leftarrow \emptyset$
  for all j **do**
    repeat
      $w_*(\boldsymbol{x}^j_{T_i}) \leftarrow \dfrac{\hat{P}_*(\boldsymbol{x}^j_{T_i})}{P^i_{\approx}(\boldsymbol{x}^j_{T_i})}$
      $q_j \leftarrow \min\left(1, w_*(\boldsymbol{x}^j_{T_i}) \big/ c_i\right)$
      $p \sim U[0,1]$
      if $p < q_j$ **then**
        $Y_{T_i} \leftarrow Y_{T_i} \cup \left(x^j_{T_i}, \max\left(w_*(\boldsymbol{x}^j_{T_i}), c_i\right)\right)$
      else
        $k \sim w_*(\boldsymbol{x}^k_{T_l})$
        $\boldsymbol{x}'_{T_l} \leftarrow \boldsymbol{x}^k_{T_l}$
        $P^l_{\approx}\left(\boldsymbol{x}'_{T_l}\right) \leftarrow \dfrac{M}{\sum_{m=1}^{M} w_*(\boldsymbol{x}^m_{T_l})} P^l_{\approx}(\boldsymbol{x}^k_{T_l})$
        $\boldsymbol{x}'_{T_i \setminus T_l} \sim \hat{P}\left(\boldsymbol{x}'_{T_i \setminus T_l} \mid \boldsymbol{x}'_{T_l}\right)$
        $P^i_{\approx}\left(\boldsymbol{x}'_{T_i}\right) \leftarrow \hat{P}\left(\boldsymbol{x}'_{T_i \setminus T_l} \mid \boldsymbol{x}'_{T_l}\right) P^i_{\approx}\left(\boldsymbol{x}'_{T_l}\right)$
        $\boldsymbol{x}^j_{T_i} \leftarrow \boldsymbol{x}'_{T_i}$
        $\boldsymbol{P}^i_{\approx}\left(x^j_{T_i}\right) \leftarrow \boldsymbol{P}^i_{\approx}\left(\boldsymbol{x}'_{T_i}\right)$
      end if
    until $p < q_j$
  end for
  return Y_{T_i}
end function

is the graph on Figure 5.18a, which includes only nearest neighbor interactions. Because of this severely limiting choice, the resulting approximation may be rather poor.

Adding the second- and third-nearest neighbor interaction increases the complexity of the approximation, producing the graph on Figure 5.18b. Because yellow variables are now dependent on each other, an approximation of the marginal density of these variables cannot be used to sample the yellow variables given the blue ones. It may, however, be used to



(a) Nearest neighbor interaction (b) Nearest, second- and third-nearest interactions

Figure 5.18: Two dependency graphs respected by (a) the approximation $P^i_{\approx}(\boldsymbol{x}_{T_i})$ and (b) the approximation $\widehat{P}_*(\boldsymbol{x}_{T_i})$.

approximately evaluate the marginal density given the values of all variables $\boldsymbol{x}_{T_i}$, making the dense approximation $\widehat{P}_*(\boldsymbol{x}_{T_i})$ useful in computing weights.

Denote the probability, whose conditional independence structure respects the graph on Figure 5.18a by $P^i_{\approx}(\boldsymbol{x}_{T_i})$, while that respecting the denser graph on Figure 5.18b by $\widehat{P}_*(\boldsymbol{x}_{T_i})$. Using the first one for sampling, while the latter is used solely for later resampling, allows for approximately sampling from the denser probability while keeping the benefits of the sparser dependency graphs used to produce the sampling density $P^i_{\approx}(\boldsymbol{x}_{T_i})$. Of course the dense approximation may include further interactions, such as the interaction with third-nearest neighbors, being limited only by the available computing power and the fact that a very good resampling algorithm will still fail to correct a poor trial probability.

5.2.5 *Practical particle filtering algorithm*

The single-stage resampling algorithms described above are not robust enough to lead to much improvement. Here we describe a practical, multi-stage algorithm.

We define a set of check-points, defined as a set of indexes $R = \{R_1, R_2, \dots, R_L\}$ of topological ordering T. Therefore, as soon as the variable associated with node u_{R_i} has been sampled, the i^{th} check-point has been reached. The i^{th} location R_i attempts to improve all variables sampled thus far, that is the variables $\boldsymbol{x}_{T_{R_i}}$. A set of dense approximations $\widehat{P}_*(\boldsymbol{x}_{T_{R_i}})$ must be computed using the fast marginalization algorithm of Chapter 4 in order to do so. The final node of each lattice V_i is a recom-

mended candidate for the check-point R_i. Note that the approximation used in lateral densening might be shared and used as $\widehat{P}_*(\boldsymbol{x}_{T_{R_i}})$.

The samples are generated in batches of M samples (particles) at once, with separate batches being independent of each other. As the particles in a batch reach a check-point R_i, the weights $w_*(\boldsymbol{x}^j_{T_{R_i}})$ are computed for each particle. Then, the normalized weights are computed via

$$\widehat{w}_*(\boldsymbol{x}^j_{T_{R_i}}) = \frac{w_*(\boldsymbol{x}^j_{T_{R_i}})}{\sum_{j=1}^{M} w_*(\boldsymbol{x}^j_{T_{R_i}})},$$

allowing for the computation of an effective number of particles (Liu, Chen, and Logvinenko, 2001, p. 232),

$$M_{\text{eff}} = \frac{1}{\sum_{j=1}^{M} \widehat{w}_*(\boldsymbol{x}^j_{T_{R_i}})}.$$

M_{eff} can be intuitively understood as the equivalent number of independently distributed particles. In case the effective number M_{eff} is smaller than a pre-defined threshold M_{min}, the particles undergo a Partial Rejection Control (PRC) algorithm from Section 5.2.3.1. Whenever a resampling stage occurred or not, the particles continue being sampled until the next stopping point R_{i+1} or until all variables are sampled.

The weight thresholds c_{R_i} are determined in a short sampling run ahead of the main computation. A single large batch of particles is generated. When the particles reach the check-point R_i, the threshold c_{R_i} is selected based on the distribution of the weights $w_*(\boldsymbol{x}^j_{T_{R_i}})$. A practical formula suggested by Liu, Chen, and Wong (1998) is the weighted average

$$\begin{aligned} c_{R_i} = p_{\text{min}} \min_j \left(w_*(\boldsymbol{x}^j_{T_{R_i}}) \right) + p_{\text{mean}} \operatorname*{mean}_j \left(w_*(\boldsymbol{x}^j_{T_{R_i}}) \right) \\ + p_{\text{max}} \max_j \left(w_*(\boldsymbol{x}^j_{T_{R_i}}) \right) \end{aligned}$$

with $p_{\text{min}} + p_{\text{mean}} + p_{\text{max}} = 1$. Once the threshold is selected, the particles undergo rejection and the process continues. The set of thresholds c_{R_i} is then fixed and does not change during the main computation, thus ensuring that different batches are properly normalized.

5.3 WEIGHT-BASED MONTE CARLO ALGORITHMS

The Markov Chain Monte Carlo (MCMC) algorithm resembles importance sampling: states that are visited less frequently by the trial distribution than demanded by the target distribution are more difficult to leave, effectively multiplying the number of times the particular state will appear. It comes then as no surprise that the importance sampling scheme may be quickly turned into an MCMC algorithm. We present here two developments: the straightforward MCMC algorithm and a more advanced Generalized Gibbs sampler.

5.3.1 *Markov Chain Monte Carlo*

Consider a complete state $\boldsymbol{x}_V$ together with the attached weight $\boldsymbol{w}(\boldsymbol{x}_V) = P(\boldsymbol{x}_V)/P_{\approx}(\boldsymbol{x}_V)$ generated by the importance sampler described above. Generate a second state $\boldsymbol{y}_V$ with weight $\boldsymbol{w}(\boldsymbol{y}_V)$. The two states were generated from the proposal density $P_{\approx}$, therefore the classical Metropolis-Hastings algorithm (Liu, 2001; Metropolis et al., 1953; Robert and Casella, 2004) conditionally accepts the move from $\boldsymbol{x}_V$ to $\boldsymbol{y}_V$ with probability

$$\begin{aligned}\alpha(\boldsymbol{x}_V, \boldsymbol{y}_V) &= \frac{P_{\approx}(\boldsymbol{x}_V)P(\boldsymbol{y}_V)}{P_{\approx}(\boldsymbol{y}_V)P(\boldsymbol{x}_V)} \\ &= \frac{P(\boldsymbol{y}_V)}{P_{\approx}(\boldsymbol{y}_V)}\frac{P_{\approx}(\boldsymbol{x}_V)}{P(\boldsymbol{x}_V)} \\ &= \frac{w(\boldsymbol{y}_V)}{w(\boldsymbol{x}_V)}.\end{aligned}$$

The resulting equation has a straightforward interpretation, since the transition from a state with a lower weight into a state with a larger weight should be easier than the opposite.

Since the MCMC scheme does not do anything beyond the eliminating weights by rejection sampling, the performance of the original importance sampling and the MCMC algorithms is expected to be very similar.

5.3.2 *Generalized Gibbs sampler*

The Gibbs sampler or heat-bath MCMC splits the nodes of a lattice $V = U \cup V \setminus U$ and samples the variables $\boldsymbol{x}_U$ using the exact conditional

probability. Starting with the state $\boldsymbol{x}_V$ the algorithm creates a proposal state $\boldsymbol{y}_V$ given by

$$\begin{aligned}\boldsymbol{y}_{V\setminus U} &= \boldsymbol{x}_{V\setminus U} \\ \boldsymbol{y}_U &\sim P(\boldsymbol{y}_U|\boldsymbol{y}_{V\setminus U}) = P(\boldsymbol{y}_U, \boldsymbol{y}_{V\setminus U}) \Big/ \int P(\boldsymbol{y}_U, \boldsymbol{y}_{V\setminus U}) d\boldsymbol{y}_U,\end{aligned}$$

where the subset of variables $\boldsymbol{y}_U$ is resampled using the conditional probability. The subset U must be small in order for the computation of the exact marginal density in the denumerator to be feasible. Because the proposal density is the exact conditional probability, the acceptance probability is equal to one and the proposed move is always accepted. In order to sample the entire space, the choice of the set U is changed at random between MCMC moves.

For a set U consisting of a single variable we recover the slice sampler, whose name comes from the fact that it updates the state $\boldsymbol{x}_V$ one dimension (slice) at a time. Larger sizes of U are also possible and allow for larger moves and quicker mixing rates, but the added performance comes at the cost of increased computational cost, growing exponentially in the number of variables $\boldsymbol{x}_U$.

Because the variables in $\boldsymbol{x}_U$ are sampled using their conditional probability given the remaining variables, we see that the method is related to the acyclic representation of $P(\boldsymbol{x}_V)$ we have constructed. In fact, the splitting $V = U \cup V \setminus U$ is similar to the first step of the coarsening algorithm. Indeed, the variables $\boldsymbol{x}_{V_0\setminus V_1}$ are sampled using conditional probabilities computed using the exact density $P(\boldsymbol{x}_{V_0}) = P(\boldsymbol{x}_V)$, therefore those variables are indeed sampled using the one-dimensional Gibbs sampler. However, for variables appearing earlier in the dependency digraph $D = (V, A)$ the connection is less clear.

5.3.2.1 *Overview*

We proceed in the following way. Starting with a random node $u \in V$ we build a proposal sample using the approximate conditional probability $P_{\approx}(\boldsymbol{x}_V)$. Having at the node u, we subsequently sample the nodes whose approximate conditional probabilities depend on the node u, iteratively building the updated set U. Because the proposal density of the new state is not exact, we recover $P(\boldsymbol{x}_V)$ as the stationary distribution through the use of a rejection step, described below, producing a Markov Chain Monte Carlo method.

5.3.2.2 *Constructing the proposal state*

Select an arbitrary node $u \in V$, referred to thereafter as the head node, and consider the set of nodes whose approximate conditional probabilities depend on the value of the variable associated with the head node u. Given the dependency digraph $D = (V, A)$ we see that this set is simply the set of direct successors (children) of the head node u, written $N_s(u)$. When the variables associated with the nodes in $N_s(u)$ are sampled to reflect the change in their conditional distributions, the set of variables affected by the change in the variable associated with u grows; if the process is repeated recursively, the set of nodes affected by the change in the variable associated with the head node u becomes the set of successors $S(u)$ defined as

$$S(u) = \Big\{ v \in V \mid u \leq v \Big\},$$

i.e. the set of nodes $v \in V$ such that there exists a directed path from u to v. Given the DAG $D = (V, A)$, the set $S(u)$ may be found quickly by recursively following the directed edges emanating from the head node u and collecting all visited nodes.

Call the initial state $\boldsymbol{x}_V$ and assume we have computed a list of approximate conditional probabilities $\hat{P}(x_{u_i} \mid \boldsymbol{x}_{N_p(u_i)})$ from Equation 5.3, where i is the index of the topological ordering T. We choose a head node $u \in V$ and produce a proposal state $\boldsymbol{y}_V$ defined through

$$\begin{aligned} \boldsymbol{y}_{V \setminus S(u)} &= \boldsymbol{x}_{V \setminus S(u)} \\ \boldsymbol{y}_{S(u)} &\sim \hat{P}(\boldsymbol{y}_{S(u)} | \boldsymbol{y}_{V \setminus S(u)}), \end{aligned}$$

where the conditional probability of $\boldsymbol{y}_{S(u)}$ given the remaining variables is given by

$$\hat{P}(\boldsymbol{y}_{S(u)} | \boldsymbol{y}_{V \setminus S(u)}) = \prod_{v \in S(u)} \hat{P}(y_v \mid \boldsymbol{y}_{N_P(v)}).$$

Additionally, produce an updated list of approximate conditional probabilities, keeping the existing $\boldsymbol{x}_V$ values for nodes $V \setminus S(u)$ and recomputing the conditionals for the nodes in $S(u)$ using the newly sampled values $\boldsymbol{y}_V$.

5.3.2.3 *Rejection*

Finally, we perform the rejection step and accept the proposed state $\boldsymbol{y}_V$ with acceptance probability

$$\begin{aligned}\alpha(\boldsymbol{x}_V, \boldsymbol{y}_V) &= \frac{P_{\approx}(\boldsymbol{x}_{S(u)}|\boldsymbol{x}_{V\setminus S(u)})P_{\approx}(\boldsymbol{x}_{V\setminus S(u)})P(\boldsymbol{y}_V)}{P_{\approx}(\boldsymbol{y}_{S(u)}|\boldsymbol{y}_{V\setminus S(u)})P_{\approx}(\boldsymbol{y}_{V\setminus S(u)})P(\boldsymbol{x}_V)} \\ &= \frac{P_{\approx}(\boldsymbol{x}_{S(u)}|\boldsymbol{x}_{V\setminus S(u)})P(\boldsymbol{y}_V)}{P_{\approx}(\boldsymbol{y}_{S(u)}|\boldsymbol{y}_{V\setminus S(u)})P(\boldsymbol{x}_V)} \\ &= \frac{P_{\approx}(\boldsymbol{x}_{S(u)}|\boldsymbol{x}_{V\setminus S(u)})}{P(\boldsymbol{x}_V)} \frac{P(\boldsymbol{y}_V)}{P_{\approx}(\boldsymbol{y}_{S(u)}|\boldsymbol{y}_{V\setminus S(u)})} \\ &= \frac{w(\boldsymbol{y}_V)}{w(\boldsymbol{x}_V)},\end{aligned}$$

which is the same weight ratio as in the MCMC method above. However, since the change due to the proposed move is limited to a fraction of the set of variables, there will be a significant cancellation between the two weights. Using the decomposition of the total weight $w(\boldsymbol{x}_V)$ into per-variable weights $w_i(\boldsymbol{x}_V)$ defined in Equation 5.4, we obtain

$$w(\boldsymbol{x}_V) = \prod_{v_i \in V} w_i(\boldsymbol{x}_V) = \prod_{v_i \in S(u)} w_i(\boldsymbol{x}_V) \prod_{v_i \in V\setminus S(u)} w_i(\boldsymbol{x}_V).$$

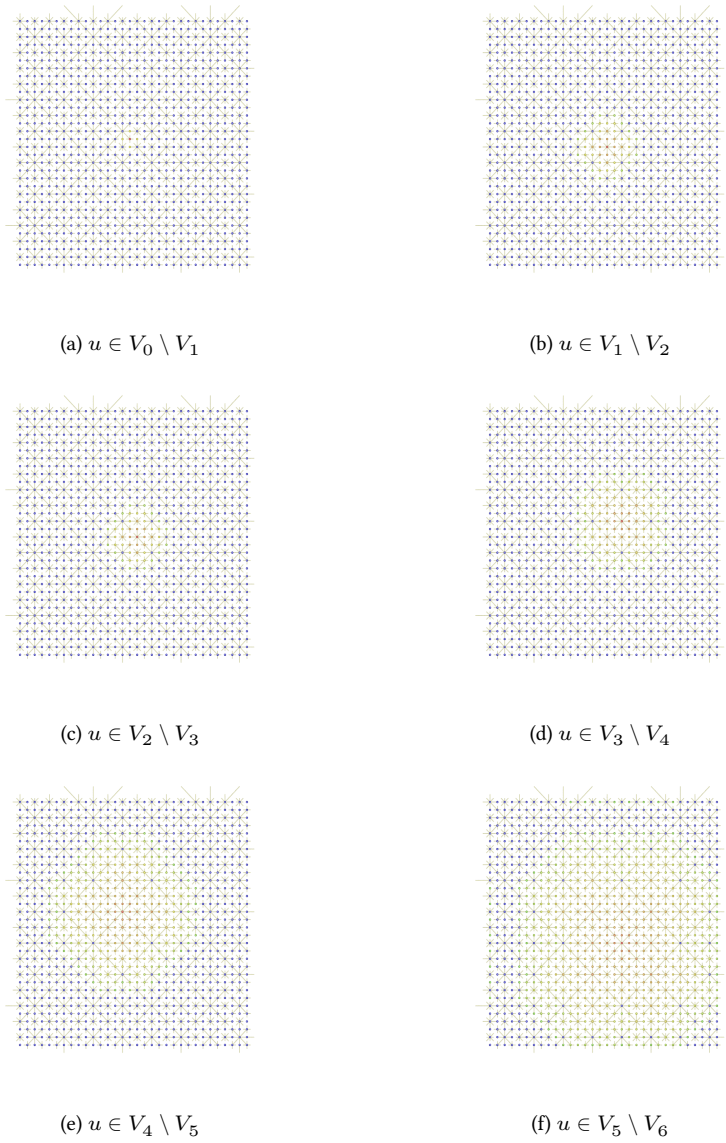
Since the variables $\boldsymbol{x}_{V\setminus S(u)}$ and $\boldsymbol{y}_{V\setminus S(u)}$ are by definition the same, so must be their trial probabilities and partial weights. Through cancellation we finally have

$$\begin{aligned}\alpha(\boldsymbol{x}_V, \boldsymbol{y}_V) &= \frac{\left(\prod_{v_i \in S(u)} w_i(\boldsymbol{y}_V)\right)\left(\prod_{v_i \in V\setminus S(u)} w_i(\boldsymbol{y}_V)\right)}{\left(\prod_{v_i \in S(u)} w_i(\boldsymbol{x}_V)\right)\left(\prod_{v_i \in V\setminus S(u)} w_i(\boldsymbol{x}_V)\right)} \\ &= \prod_{v_i \in S(u)} \frac{w_i(\boldsymbol{y}_V)}{w_i(\boldsymbol{x}_V)}.\end{aligned}$$

When $S(u)$ is small, the proposed move will be small as well, resulting in a large probability of acceptance.

5.3.2.4 *Head node selection*

The remaining question regards the particular manner in which the head nodes u defining the heat baths should be selected. Figure 5.19 shows the expected size of a move given that $u \in V_i \setminus V_{i+1}$ for different values of i. In the limit $i = 0$ we recover the exact proposal density and the slice



(a) $u \in V_0 \setminus V_1$

(b) $u \in V_1 \setminus V_2$

(c) $u \in V_2 \setminus V_3$

(d) $u \in V_3 \setminus V_4$

(e) $u \in V_4 \setminus V_5$

(f) $u \in V_5 \setminus V_6$

Figure 5.19: Visualizations of the set of successors $S(u)$ for head nodes u lying on different lattices obtained by a checkerboard coarsening of a 32×32 Cartesian lattice. Nodes are colored by the number of links from the initiating node, marked red, through yellow to green. The nodes which are not accessible from the head node u are marked blue.

sampler, achieving acceptance probability $\alpha(\boldsymbol{x}_V, \boldsymbol{y}_V) = 1$ but very small move size. In the opposite limit $i = m$, we find the reverse situation of a very large move, but at the cost of a very low acceptance probability. At a

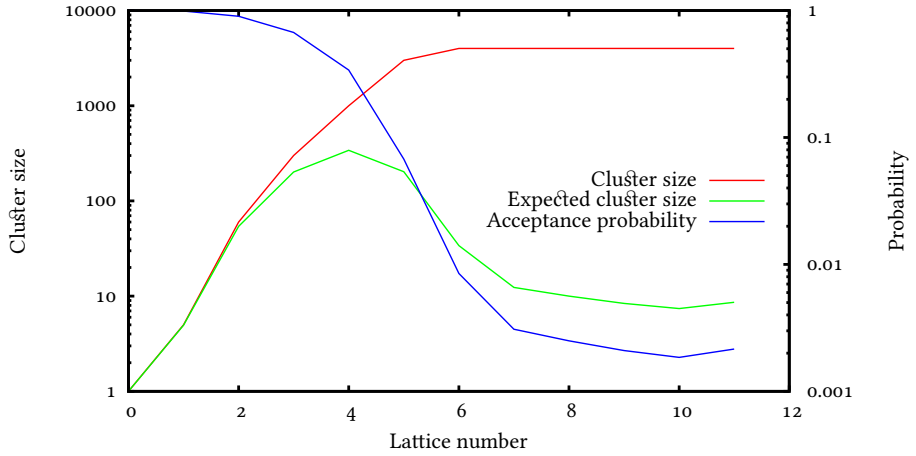


Figure 5.20: Size of the affected subset $S(u)$, proposed move acceptance probability and the resulting average move size in the case of a 64×64 Ising model at critical coupling.

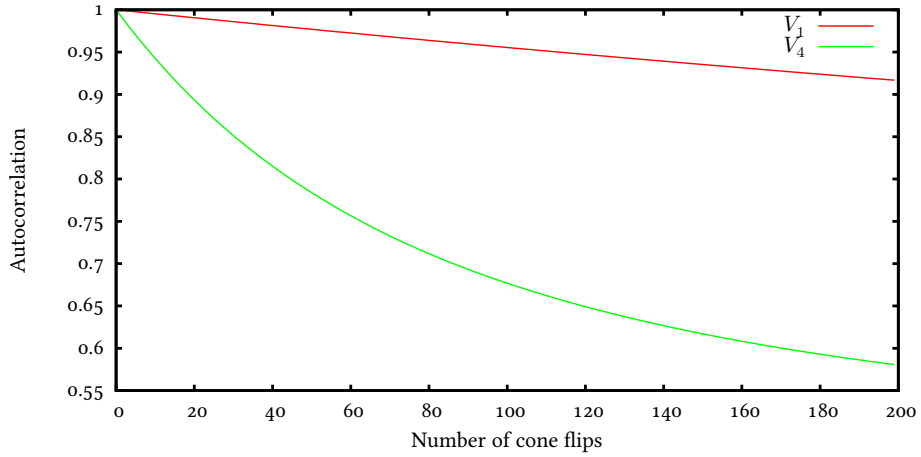


Figure 5.21: Autocorrelation in the case of the 32×32 Ising model at critical coupling. The lines show the initial lattice used to choose head nodes, including both the first renormalized lattice V_1 and the optimal lattice V_4. Although the number of variables where a change is attempted is quite large in the case of V_4, most of the variables remain unaltered due to the very strong pull of the unchanged variables $\boldsymbol{x}_{V \setminus S(u)}$, resulting in a long autocorrelation time.

point in between the two limits we usually find the optimum lattice that maximizes the expected move size.

Since the union of the sets of successors of all nodes on any V_i covers the whole set of variables,

$$\bigcup_{u \in V_i} S(u) = V,$$

we may select the head nodes u only from nodes of the optimal lattice. The resulting sampling will nonetheless have the opportunity to alter the values of all variables and the resulting Markov chain will sample the entire state space with the optimal mixing speed.

5.3.2.5 *Performance*

While the algorithm is very promising as it specifies a general family of algorithms and selects the optimum, the influence of the unchanged variables $\boldsymbol{x}_{V \setminus S(u)}$ is frequently very large, resulting in only very few variables being changed during the partial sampling algorithm. Figure 5.21 clearly shows that the autocorrelation remain high, even for the optimal method. Experimental evidence that the strong pull of the unchanged variables could only be broken by very large moves starting from a very coarse lattice, a process leading to low acceptance probabilities.

5.4 DISCUSSION

The chainless sampling described here has many interesting features. First of all, in its purest form — when the top lattice consists of a single variable and no particle filtering is used — the method involves no Markov chains, generating truly independent samples. This may be unfortunately be at the cost of a very wide range of corrective weights, especially for high-dimensional problems.

The weights of the generated samples may be improved through the sacrifice of the complete sample independence. In general the state of the art implementation of the method should follow the following steps.

- Optimize the choice of the top lattice, i.e., the stopping criterion of the graph coarsening algorithm from Chapter 3. The choice of the stopping point allows one to optimize the trade-off between the difficulty of sampling the top lattice and the number and size of errors committed by sampling individual variables using the conditional probabilities. At the possible end-points of the spectrum one may use a pure Markov Chain Monte Carlo (MCMC) method when no coarsening is performed, or the pure chain-less Monte Carlo when only a single variable remains on the top lattice; the optimal method

is usually somewhere in between, with the precise location being problem-dependent.

- Use a sound method of resampling that utilizes dense approximations of the marginal distributions to compute weights.
- Utilize lateral densening described in Section 3.5 to produce the most accurate trial probability possible.

While the Monte Carlo methods based on the above may be potentially useful, they do not represent any advantages over the weight-based algorithms and should not be used unless necessary.

The complete acyclic Monte Carlo method, including the graph coarsening described in Chapter 3, the computation of the approximate marginal densities of Chapter 4 and the importance sampling/particle filtering method above of generating random samples will be benchmarked in the chapters making up Part II of this thesis. We apply the method to the Ising model in Chapter 8 to compare the performance of the method with the existing literature.

6

GENERALIZED ACYCLIC MONTE CARLO

The method we have completely described in the prior Chapters 3, 4 and 5 is in some sense unsatisfying, because the coarser graphs V_i are composed of subsets of the fine variables $V_0 = V$. As a result, as the sampling proceeds, the variables already determined are frozen and cannot be changed, even if this could lead to an improvement. This is because changing their values would invalidate the entire acyclic structure and the resulting trial probability $P_{\approx}(\boldsymbol{x}_V)$.

In the setting of other existing methods utilizing the multi-level paradigm — the most well known being the multigrid method for the iterative solution if linear equations (Briggs, Henson, and McCormick, 2000) — this choice of coarse variables is indeed uncommon. In multigrid, the coarse variables are actually distinct from the fine variables and linked together by a certain rule, the prolongation operator. Frequently called the interpolation operator, the prolongation operator attempts to smooth the errors in the solution, although it does not necessarily reflect geometrical smoothness. There are many choices for these operators and it would strike one as unusual that in our method there is no such freedom.

The second reason for our uneasiness is the fact that the literature on renormalization methods is thus far mostly incompatible with the acyclic Monte Carlo method, because the methodology described in the earlier chapters depends on the fact that coarse variables are subsets of the original, fine variables. Thus, there is no possibility of introducing coarse variables defined using interpolation rules, such as the majority rule for spin systems.

In this chapter we develop a more general theory of renormalization based on conditional probability distributions joining subsequent levels of variables, much as the prolongation operator joins together levels of the multigrid method.

6.1 THE TWO-LATTICE METHOD

We begin with a two-lattice method, which may be extended in a straightforward manner to a full multi-lattice approach, similarly to how a two-grid method can be turned into a multigrid method. Let $G = (V, E)$ be

the undirected graph encoding the independence structure of the original probability distribution $P(\boldsymbol{x}_V)$, with each node of V corresponding to a component of $\boldsymbol{x}_V$.

Consider a method of dividing V into a collection S of disjoint subsets $\{U_1, U_2, \dots, U_n\}$ that cover V, that is

$$U_i \cap U_j = \emptyset \text{ for } i \neq j \text{ and } \bigcup_{i=1}^{n} U_i = V.$$

Therefore, the subsets $\{U_1, U_2, \dots, U_n\}$ form a partition of V and are typically selected so that nodes $v \in U_i$ are close in some respect, but this is not strictly necessary. To each subset $s \in S$ we assign a new auxiliary variable, x_s, which did not exist among the original variables $\boldsymbol{x}_V$. Using the one-dimensional Ising model as an example, the graphical structure of this situation is shown on Figure 6.22a.

Thus far there is no relation between the variables of $\boldsymbol{x}_V$ and those of $\boldsymbol{x}_S$, apart from our mental association. We formalize the connection between $\boldsymbol{x}_V$ and $\boldsymbol{x}_S$ by defining a probability distribution over $\boldsymbol{x}_S$ conditional on $\boldsymbol{x}_V$, giving us a joint probability

$$P(\boldsymbol{x}_V, \boldsymbol{x}_S) = P(\boldsymbol{x}_V) P(\boldsymbol{x}_S \mid \boldsymbol{x}_V).$$

The conditional probability $P(\boldsymbol{x}_S \mid \boldsymbol{x}_V)$ specifies a probabilistic rule assigning values to variables in $\boldsymbol{x}_S$ based on the values of $\boldsymbol{x}_V$, similarly to the prolongation operator of the multigrid method.

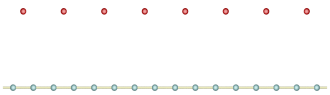
We will assume that the conditional probability factorizes as

$$P(\boldsymbol{x}_S \mid \boldsymbol{x}_V) = \prod_{s \in S} P(x_s \mid \boldsymbol{x}_{U_s}),$$

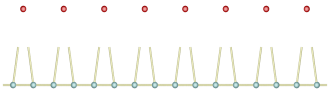
where U_s is the subset of V to which the variable x_s was assigned. Due to this factorization, the graphical structure becomes that of Figure 6.22b, with the lines connecting the new variables representing formal dependence: the variables $x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{U_u}$ and $x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{U_v}$ for $u \neq v$ due to the factorization. The choice of the coarsening rule that specifies the conditional probability $P(x_s \mid \boldsymbol{x}_{U_s})$ is nearly arbitrary, but common choices include decimation and majority rule will be mentioned later on.

The subsequent step in the construction is the computation of a marginal probability density, which we obtain from the joint distribution $P(\boldsymbol{x}_V, \boldsymbol{x}_S)$ using the definition

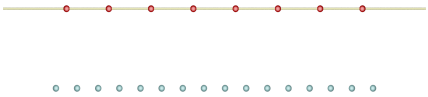
$$P(\boldsymbol{x}_S) = \int P(\boldsymbol{x}_V, \boldsymbol{x}_S) d\boldsymbol{x}_V = \int P(\boldsymbol{x}_V) P(\boldsymbol{x}_S \mid \boldsymbol{x}_V) d\boldsymbol{x}_V.$$



(a) The graph just after the auxiliary variables are introduced; notice the lack of connections between the original and auxiliary variables.



(b) Edges are added between the original and auxiliary variables to reflect the conditional probability $P(\boldsymbol{x}_S \mid \boldsymbol{x}_V)$.



(c) The coarse graph with exactly reconnected coarse nodes after the original variables $\boldsymbol{x}_V$ were marginalized.

Figure 6.22: Variables and the dependence graph $G = (V, E)$ of the one-dimensional Ising model (turquoise nodes), together with the auxiliary variables $\boldsymbol{x}_S$ (red nodes).

As before, performing the integration is computationally intractable in most cases and we will return to this problem while discussing the generalized fast marginalization algorithm. Presently we are interested in the

graphical structure implied by this definition. We know from Chapter 3 that removing a node in the graph by integrating out the corresponding variable induces edges between the neighboring variables.

The resulting dependency graph describing the marginal density $P(\boldsymbol{x}_S)$, shown on Figure 6.22c, is a clique, a completely connected graph where each node is directly linked to every other node. Even if the original graph $G = (V, E)$ was sparse, the immediately coarser graph becomes fully connected, making it computationally intractable to compute exact marginal densities for even the simplest models, such as the one-dimensional Ising model shown here.

Assume for now that it is possible to generate a sample $\boldsymbol{x}_S \sim P(\boldsymbol{x}_S)$. Because $\boldsymbol{x}_V$ are no longer conditionally independent given $\boldsymbol{x}_S$, it is no longer possible to sample the variables of $\boldsymbol{x}_V$ individually. Instead, all variables in $\boldsymbol{x}_V$ mus be sampled simultaneously, making MCMC the method of choice. However, the use of MCMC makes it impossible to use advanced techniques such as importance sampling or particle filtering, because the exact trial probability cannot be computed.

While the prospects of this approach appear to be bleak, it turns out that they represent the worst case scenario. Indeed, we will show that the acyclic Monte Carlo is a special case achieved through the choice of decimation as the coarsening rule.

We will describe the generalized acyclic Monte Carlo method using the example of the Ising model undergoing coarsening under the most commonly used coarsening rule, the majority rule. We will describe how to construct a ladder of coarse graphical models, then explain the computation of approximate marginal densities using the generalized fast marginalization approach. We will finish with a description of how the structure obtained may be used to sample the original probability distribution $P(\boldsymbol{x}_V)$.

6.2 COARSENING

We begin with the initial set of variables and graph $G_0 = (V_0, E_0) = (V, E)$ encoding the independence structure of the original probability distribution $P(\boldsymbol{x}_{V_0})$. Assume the set of variables V_i is known. We divide it into a collection S_i of disjoint subsets $\{U_j\}$ of V_i that form a partition of V_i, that is

$$U_j \cap U_k = \emptyset \quad \text{for} \quad j \neq k \quad \text{and} \quad \bigcup_j U_j = V_i.$$

We define the joint probability of $\boldsymbol{x}_{V_i}$ and $\boldsymbol{x}_{S_i}$ through

$$P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{S_i}) = P(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{S_i} \mid \boldsymbol{x}_{V_i}),$$

leading to the natural definition of the marginal density

$$P(\boldsymbol{x}_{S_i}) = \int P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{S_i}) d\boldsymbol{x}_{V_i} = \int P(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{S_i} \mid \boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i}.$$

We rename S_i by $V_{i+1} = S_i$, so that the marginal becomes

$$P(\boldsymbol{x}_{S_i}) = P(\boldsymbol{x}_{V_{i+1}}) = \int P(\boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_i}.$$

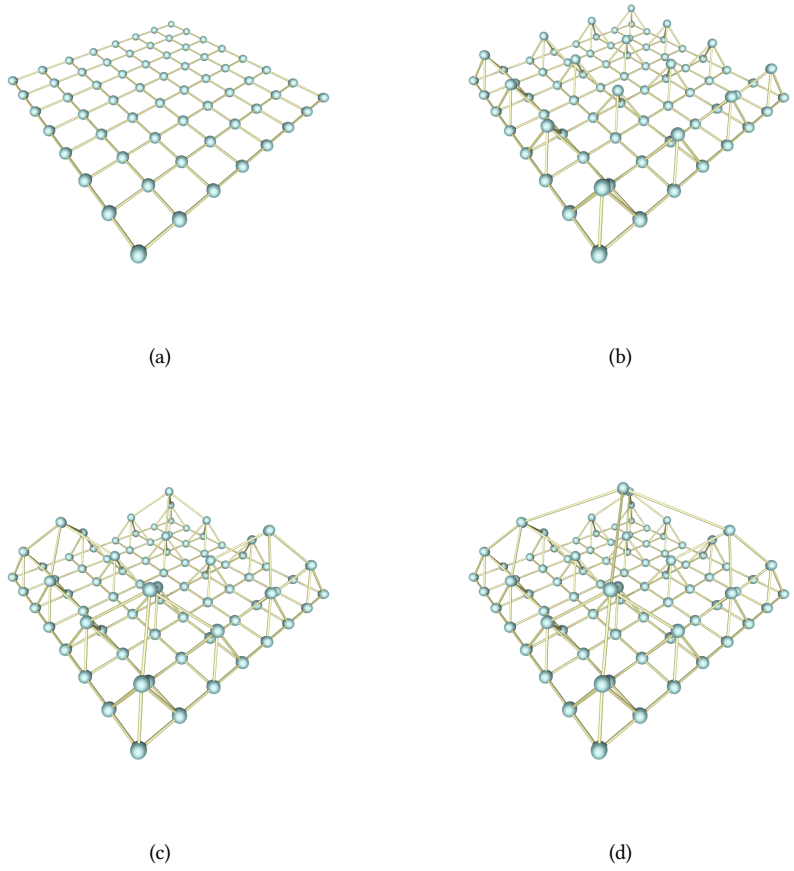
Because the conditional independence structure induced by $P(\boldsymbol{x}_{V_{i+1}})$ is that of a fully connected graph, we must immediately approximate it. Therefore, the set of edges E_{i+1} is reconstructed by reconnecting nodes that are nearby in the sense of a metric of choice, see Chapter 3 for a more detailed discussion. In the case of the Ising model, the original lattice is endowed with a set of Cartesian coordinates, which may be passed on to the coarser levels by computing the average position of the nodes within the subset U_j. The distance between the nodes provides a natural metric in this case.

For each $u, v \in V_{i+1}$ such that the metric $\rho(u, v) \leq T_{i+1}$ we form an edge $(u, v) \in E_{i+1}$, thus completing the specification of the graph $G_{i+1} = (V_{i+1}, E_{i+1})$.

The above iterative construction builds a ladder of successively coarser lattices with node sets $V = V_0, V_1, V_2, \ldots, V_m$, which stops when the set of variables V_m is deemed sufficiently small.

The complete coarsening process is shown on Figure 6.23. We begin with a regular two-dimensional Cartesian lattice of the Ising model and divide it into subsets of 2×2 nodes. Larger or irregular subsets are allowed when needed. The process continues until a small enough lattice is achieved.

The connections are added up to a distance of the third nearest neighbor, resulting in a dense graph. However, this is not a difficulty because the variables will be sampled together using MCMC rather than individually using conditional probabilities.



(a) (b)

(c) (d)

Figure 6.23: Complete coarsening process of a two-dimensional Cartesian lattice of initial size 8×8. The nodes of the lattice are divided into subsets of size 2×2, decreasing the number of nodes by a factor of 4 during each coarsening step. Although the lattices are two-dimensional, height was used to represent the coarsening level, with nodes higher up belonging to the coarser lattices.

6.3 GENERALIZED FAST MARGINALIZATION

Consider the marginal probability density for lattice V_i,

$$P(\boldsymbol{x}_{V_i}) = \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) \dots P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}) d\boldsymbol{x}_{V_0} d\boldsymbol{x}_{V_1} \dots d\boldsymbol{x}_{V_{i-1}}.$$

If necessary, we define a differentiable extension $\tilde{P}(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})$, which in turn defines uniquely a differentiable extension of the marginal,

$$\tilde{P}(\boldsymbol{x}_{V_i}) = \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) \dots \tilde{P}(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}) d\boldsymbol{x}_{V_0} d\boldsymbol{x}_{V_1} \dots d\boldsymbol{x}_{V_{i-1}},$$

analogously to the technique used in Chapter 4.

Assume that $P(\boldsymbol{x}_{V_i}) > 0$ is strictly positive and can thus be written as

$$P(\boldsymbol{x}_{V_i}) = \exp\left(W(\boldsymbol{x}_{V_i})\right) \Big/ Z_{V_i}.$$

For $u \in V_i$ we wish to obtain an approximation to

$$\frac{\partial W(\boldsymbol{x}_{V_i})}{\partial x_u} = \frac{\partial}{\partial x_u} \ln P(\boldsymbol{x}_{V_i})$$

within the space spanned by a basis ϕ. Plugging in the definition of the marginal density $P(\boldsymbol{x}_{V_i})$ we obtain the generalized fast marginalization equation

$$\begin{aligned} \frac{\partial W}{\partial x_u} &= \frac{\partial}{\partial x_u} \ln P(\boldsymbol{x}_{V_i}) \\ &= \mathbb{E}\left[\frac{\partial P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})}{\partial x_u} \Big/ P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}) \Big| \boldsymbol{x}_{V_i} \right]. \end{aligned}$$

The remaining work closely resembles previous developments. We project the target function

$$\mathcal{F}(\boldsymbol{x}_{V_i}) = \mathbb{E}\left[\frac{\partial P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})}{\partial x_u} \Big/ P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}) \Big| \boldsymbol{x}_{V_i} \right]$$

in the least squares sense onto the basis ϕ by constructing a matrix $A = (A_{kl})$, $A_{kl} = \langle \phi_k, \phi_l \rangle$, and right hand side vector $b_k = \langle \mathcal{F}, \phi_k \rangle$. To simplify notation, define

$$P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) \dots P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}).$$

Using the inner product defined in Section 4.1.1 we obtain

$$\begin{aligned} A_{kl} &= \langle \phi_k, \phi_l \rangle \\ &= \int \phi_k(\boldsymbol{x}_{V_i}) \phi_l(\boldsymbol{x}_{V_i}) \left(P(\boldsymbol{x}_{V_i}) \Big/ Q(\boldsymbol{x}_{V_i}) \right) d\boldsymbol{x}_{V_i} \end{aligned}$$

$$
\begin{aligned}
&= \int \left(\frac{\phi_k(\boldsymbol{x}_{V_i})\phi_l(\boldsymbol{x}_{V_i})}{Q(\boldsymbol{x}_{V_i})} \right) \\
&\qquad \times \int P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_0, V_1, \dots, V_i} \\
&= \mathbb{E}\left[\frac{\phi_k(\boldsymbol{x}_{V_i})\phi_l(\boldsymbol{x}_{V_i})}{Q(\boldsymbol{x}_{V_i})} \right],
\end{aligned}
$$

where the marginal distribution $P(\boldsymbol{x}_{V_i})$ is replaced with its definition. The right hand side vector becomes

$$
\begin{aligned}
b_k &= \langle \mathcal{F}, \phi_k \rangle \\
&= \int \phi_k(\boldsymbol{x}_{V_i}) \mathcal{F}(\boldsymbol{x}_{V_i}) \left(P(\boldsymbol{x}_{V_i}) \Big/ Q(\boldsymbol{x}_{V_i}) \right) d\boldsymbol{x}_{V_i} \\
&= \int \phi_k(\boldsymbol{x}_{V_i}) \left(\frac{P(\boldsymbol{x}_{V_i})}{Q(\boldsymbol{x}_{V_i})} \right) \\
&\qquad \times \frac{\int P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}) \left(\frac{P'(\boldsymbol{x}_{V_i} | \boldsymbol{x}_{V_{i-1}})}{P(\boldsymbol{x}_{V_i} | \boldsymbol{x}_{V_{i-1}})} \right) d\boldsymbol{x}_{V_0, V_1, \dots, V_{i-1}}}{\int P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_0, V_1, \dots, V_{i-1}}} d\boldsymbol{x}_{V_i}.
\end{aligned}
$$

Notice that the denumerator, coming from $\mathcal{F}(\boldsymbol{x}_{V_i})$, is equal to the marginal $P(\boldsymbol{x}_{V_i})$. Canceling the two out yields

$$
\begin{aligned}
b_k &= \int \frac{\phi_k(\boldsymbol{x}_{V_i})}{Q(\boldsymbol{x}_{V_i})} \int P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}) \\
&\qquad \times \left(\frac{P'(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})}{P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})} \right) d\boldsymbol{x}_{V_0, V_1, \dots, V_i} \\
&= \mathbb{E}\left[\frac{\phi_k}{Q} \frac{P'(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})}{P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})} \right],
\end{aligned}
$$

showing again that the weight $P(\boldsymbol{x}_{V_i})$ in the inner product definition is necessary to transform A_{kl} and b_k into expectations over the complete set of variables $\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots, \boldsymbol{x}_{V_i}$.

6.3.1 *The case of discrete variables*

We meet the challenge of handling probabilities defined over discrete variables in a manner very similar to the original fast marginalization. We extend x_u for $u \in V_i$ to take real values, renaming it χ_u to avoid confusion,

and define a differentiable extension $\tilde{P}(\chi_u, \boldsymbol{x}_{V_i \setminus u} \mid \boldsymbol{x}_{V_{i-1}})$ of the coarsening rule $P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}})$ that agrees with the original rule whenever χ_u takes any of the original discrete values.

We use the mixed continuous-discrete representation described in Section 4.2.3. Therefore, the matrix $A_{kl}(\chi_u)$ and $b_k(\chi_u)$ are functions of the continuous variable χ_u, which is to take values equal to Gaussian integration nodes. Using an inner product

$$\langle f, g \rangle_{\chi_u} = \int f(\boldsymbol{x}_{V_i \setminus u}, \chi_u) g(\boldsymbol{x}_{V_i \setminus u}, \chi_u) \frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} d\boldsymbol{x}_{V_i \setminus u}$$

we have

$$\begin{aligned} A_{kl}(\chi_u) &= \langle \phi_k, \phi_l \rangle_{\chi_u} \\ &= \mathbb{E}\left[\frac{\phi_k(\boldsymbol{x}_{V_i \setminus u}) \phi_l(\boldsymbol{x}_{V_i \setminus u})}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} \tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u \mid \boldsymbol{x}_{V_{i-1}}) \right]. \end{aligned}$$

Similarly, the formula for $b_k(\chi_u)$ becomes

$$\begin{aligned} b_k(\chi_u) &= \langle \mathcal{F}, \phi_k \rangle_{\chi_u} \\ &= \mathbb{E}\left[\frac{\phi_k(\boldsymbol{x}_{V_i \setminus u}, \chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} \tilde{P}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u \mid \boldsymbol{x}_{V_{i-1}}) \right], \end{aligned}$$

which is equivalent to the continuous formula except for the fact that χ_u is fixed. Solving the linear system $A(\chi_u)\boldsymbol{c}(\chi_u) = \boldsymbol{b}(\chi_u)$ yields renormalized coupling coefficients $\boldsymbol{c}(\chi_u)$, whose integral allows one to recover the discrete energy difference $\Delta_u W(\boldsymbol{x}_{V_i \setminus u}; a, b)$ and hence a the marginal probability distribution $P(\boldsymbol{x}_{V_i})$, as in Section 4.1.4.

6.3.2 *Computing the expected values*

The fact that the coarse variables $\boldsymbol{x}_{V_i}$, $i > 0$, are not a subset of the original variables $\boldsymbol{x}_{V_0}$ causes a mild difficulty when computing the expected values. The expected values defined above are defined not over the original probability density, but over the extended probability density that includes the coarse variables $\boldsymbol{x}_{V_i}$ for $i > 0$,

$$P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \ldots, \boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) \ldots P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}).$$

If the coarse densities are not yet available, the coarse variables may be sampled given the original variables, following the method used since

Ma (1976). This sampling direction is the easier one, assuming that we can sample from $P(\boldsymbol{x}_{V_0})$.

We begin by sampling $\boldsymbol{x}_{V_0} \sim P(\boldsymbol{x}_{V_0})$ using a method such as the Markov Chain Monte Carlo (MCMC). The coarser levels are then sampled consecutively using the coarsening rules,

$$x_{V_i} \sim P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i-1}}),$$

a task accomplished easily because all variables within $\boldsymbol{x}_{V_i}$ are conditionally independent of each other given the values of $\boldsymbol{x}_{V_{i-1}}$,

$$x_u \perp\!\!\!\perp x_v \mid \boldsymbol{x}_{V_{i-1}} \text{ for any } u, v \in V_i.$$

This makes both the sampling from the extended probability distribution and computation of the expected values required by generalized fast marginalization a straightforward enterprise.

6.3.3 *Symmetrization*

The handling of discrete PDFs through differentiable extension leads to the appearance of basis functions that do not satisfy the symmetry constraints expected of $W(\boldsymbol{x}_{V_i})$. We eliminate this difficulty through symmetrization, similarly to what was done in the original fast marginalization algorithm in Section 4.2.4.

We provide here the fully-symmetrized formulae. Assume that the Hamiltonian $W(\boldsymbol{x}_{V_i})$ is even; we project the even function

$$\mathcal{F}(\boldsymbol{x}_{V_i \setminus u}, \chi_u) = \frac{1}{2}\left(\frac{\tilde{P}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u)}{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} + \frac{\tilde{P}'(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)}{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)}\right)$$

under the χ_u-dependent inner product

$$\langle f, g\rangle_{\chi_u} = \frac{1}{2}\int f(\boldsymbol{x}_{V_i \setminus u}) g(\boldsymbol{x}_{V_i \setminus u}) \left(\frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} + \frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)}\right).$$

Because of the different in the inner product, both the $A_{kl}(\chi_u)$ and $b_k(\chi_u)$ formulae change. The matrix entry $A_{kl}(\chi_u)$ becomes

$$\begin{aligned} A_{kl}(\chi_u) &= \langle \phi_k, \phi_l\rangle_{\chi_u} \\ &= \frac{1}{2}\int \phi_k(\boldsymbol{x}_{V_i \setminus u}, \chi_u)\phi_l(\boldsymbol{x}_{V_i \setminus u}, \chi_u) \end{aligned}$$

$$\times \left(\frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} + \frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)}{Q(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)} \right) d\boldsymbol{x}_{V_i \setminus u}$$

$$= \frac{1}{2} \mathbb{E} \Bigg[\phi_k(\boldsymbol{x}_{V_i \setminus u}, \chi_u) \phi_l(\boldsymbol{x}_{V_i \setminus u}, \chi_u)$$

$$\times \left(\frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, \chi_u \mid \boldsymbol{x}_{V_{i-1}})}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} + \frac{\tilde{P}(\boldsymbol{x}_{V_i \setminus u}, -\chi_u \mid \boldsymbol{x}_{V_{i-1}})}{Q(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)} \right) \Bigg] .$$

The derivation of the right hand side vector entry $b_k(\chi_u)$ is more involved, but we obtain

$$b_k(\chi_u) = \frac{1}{4} \mathbb{E} \Bigg[\phi_k(\boldsymbol{x}_{V_i \setminus u}, \chi_u)$$

$$\times \left(\tilde{P}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u \mid \boldsymbol{x}_{V_{i-1}}) \left(\frac{1}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} + \frac{\mathcal{R}(-\chi_u, \boldsymbol{x}_{N_{V_i}(u)})}{Q(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)} \right) \right.$$

$$\left. + \tilde{P}'(\boldsymbol{x}_{V_i \setminus u}, -\chi_u \mid \boldsymbol{x}_{V_{i-1}}) \left(\frac{1}{Q(\boldsymbol{x}_{V_i \setminus u}, -\chi_u)} + \frac{\mathcal{R}(\chi_u, \boldsymbol{x}_{N_{V_i}(u)})}{Q(\boldsymbol{x}_{V_i \setminus u}, \chi_u)} \right) \right) \Bigg] .$$

While the derivation of the formulae appears discouragingly complex, in practice its use amounts to a minor correction to the non-symmetrized method. Similarly as in Section 4.2.4.1, the terms

$$\mathcal{R}(\chi_u, \boldsymbol{x}_{N_{V_i}(u)}) = \exp \left(\int_{-\chi_u}^{\chi_u} \tilde{W}'(\boldsymbol{x}_{V_i \setminus u}, s) ds \right)$$

must be computed approximately using the approximation of $\tilde{W}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u)$ obtained using generalized fast marginalization. Thus, the equations of the symmetrized generalized fast marginalization are implicit, requiring an iterative solution. As in fast marginalization, we solve them using a fixed-point iteration.

The function $\tilde{W}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u)$ may have a difficult to integrate shape: the majority of the mass of the function may be concentrated within a small region of the χ_u interval, or it may develop integrable singularities. Therefore, the shape of $\tilde{W}'(\boldsymbol{x}_{V_i \setminus u}, \chi_u)$ must be inspected for such difficulties and may require a specialized quadrature rule.

6.4 SAMPLING

We begin with an in-depth analysis of the transition between two levels of the lattice formed by the generalized acyclic Monte Carlo. Assume that we have already sampled the variables $\boldsymbol{x}_{V_{i+1}}$, know the value of the trial probability density $P_{\approx}(\boldsymbol{x}_{V_{i+1}})$ and the joint distribution of $P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}})$. Given these assumptions, we show that — while it is possible to sample $\boldsymbol{x}_{V_i}$ given the values of $\boldsymbol{x}_{V_{i+1}}$ — it is computationally infeasible to compute the trial probability $P_{\approx}(\boldsymbol{x}_{V_i})$.

In order to obtain $\boldsymbol{x}_{V_i}$ we need to sample $\boldsymbol{x}_{V_i}$ from $P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i+1}})$, satisfying

$$P(\boldsymbol{x}_{V_{i+1}})P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i+1}}) = P(\boldsymbol{x}_{V_i})P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i}).$$

The infeasibility lies in the fact that the marginal density $P(\boldsymbol{x}_{V_{i+1}})$ is known only approximately as $P_{\approx}(\boldsymbol{x}_{V_{i+1}})$. The exact value may be computed by summing the known joint probability distribution $P(\boldsymbol{x}_{V_i})P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i})$ over all possible states $\boldsymbol{x}_{V_i}$, which is infeasible.

The result is that it is impossible to compute the probability $P_{\approx}(\boldsymbol{x}_{V_0})$ of generating a state $\boldsymbol{x}_{V_0}$ by sampling the ladder of lattices. Therefore, the trial probability $P_{\approx}(\boldsymbol{x}_{V_0})$ can be neither corrected through importance sampling nor improved through particle filtering. However, we may still sample from the trial distribution, as described below (see also Brandt and Ron, 2001b).

6.4.1 *Ladder sampling*

Although the conditional probability $P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i+1}})$ cannot be computed, it may be evaluated up to a factor dependent only on $\boldsymbol{x}_{V_{i+1}}$,

$$P(\boldsymbol{x}_{V_i} \mid \boldsymbol{x}_{V_{i+1}}) = \frac{P(\boldsymbol{x}_{V_i}P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i})}{P(\boldsymbol{x}_{V_{i+1}})}.$$

Since the denumerator is constant in $\boldsymbol{x}_{V_i}$, a sampling method may ignore it if $\boldsymbol{x}_{V_{i+1}}$ is known and held constant. Therefore, we sample $\boldsymbol{x}_{V_i}$ from the joint probability distribution

$$P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}}) = P(\boldsymbol{x}_{V_i}P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i})$$

using MCMC, keeping $\boldsymbol{x}_{V_{i+1}}$ constant. The most straightforward method proceeds in two phases. An initial state $\boldsymbol{x}^0_{V_i}$ is selected at random with the constraint that

$$P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}}) \neq 0.$$

In the terminology of Brandt and Ron (2001b), the states $\boldsymbol{x}_{V_i}$ and $\boldsymbol{x}_{V_{i+1}}$ are said to be *compatible*. During the second phase the initial state $\boldsymbol{x}^0_{V_i}$ is iteratively updated using MCMC. The number of steps n necessary to reach a state $\boldsymbol{x}^n_{V_i}$, distributed according to a probability distribution closely approximating $P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}})$, is typically small, because of the strong influence of the conditional term $P(\boldsymbol{x}_{V_{i+1}} \mid \boldsymbol{x}_{V_i})$ (Brandt and Ron, 2001b). As in the multigrid method of linear algebra, the long-range correlations between variables are eliminated by the coarser variables $\boldsymbol{x}_{V_{i+1}}$, leaving only short-range correlations that can be relaxed quickly using local MCMC updates.

6.4.2 *Post-relaxation*

Ladder sampling produces samples from a trial probability distribution

$$P_{\approx}(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \ldots, \boldsymbol{x}_{V_m}) = P_{\approx}(\boldsymbol{x}_{V_m}) P_{\approx}(\boldsymbol{x}_{V_{m-1}} \mid \boldsymbol{x}_{V_m}) \ldots P_{\approx}(\boldsymbol{x}_{V_0} \mid \boldsymbol{x}_{V_1}),$$

which cannot be computed even up to a multiplicative constant. Therefore, the trial probability distribution cannot be corrected using the importance sampling algorithm. Instead, Brandt and Ron (2001b) suggest a technique called *post-relaxation* for correcting the trial distribution without knowing it.

The source of the errors in the trial probability is that the variables of the coarser level $\boldsymbol{x}_{V_{i+1}}$ were sampled from a probability distribution $P(\boldsymbol{x}_{V_{i+1}})$ that was an approximate marginal of $P(\boldsymbol{x}_{V_i}, \boldsymbol{x}_{V_{i+1}})$. Therefore, holding the variables $\boldsymbol{x}_{V_{i+1}}$ introduces a bias that must be removed. Post-relaxation does this by performing MCMC sweeps from $P(\boldsymbol{x}_{V_i})$, ignoring the constraints imposed by the values of $\boldsymbol{x}_{V_{i+1}}$.

The difficulty in using post-relaxation is the fact that the difference between the trial distribution $P_{\approx}(\boldsymbol{x}_{V_i})$ and the approximate marginal distribution $P(\boldsymbol{x}_{V_{i+1}})$ is not known. Therefore, the number of required post-relaxation sweeps must be estimated experimentally.

6.5 REDUCTION TO ACYCLIC MONTE CARLO

In what follows we will show that the original acyclic Monte Carlo is a special case of the generalized acyclic Monte Carlo, obtained when the coarsening rule is

$$P(x_v \mid \boldsymbol{x}_{U_v}) = \delta(x_v - x_u) \quad \text{for a fixed} \quad u \in U_v,$$

also known as decimation. The most straightforward way to prove the equivalence is by showing that the expected value

$$\begin{aligned}
&\mathbb{E}\left[f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots \boldsymbol{x}_{V_i})\right] \\
&\quad = \int f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots \boldsymbol{x}_{V_i}) P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}, \dots \boldsymbol{x}_{V_i}) d\boldsymbol{x}_{V_0, V_1, \dots, V_i}
\end{aligned}$$

reduces to an expected value over the original probability distribution $P(\boldsymbol{x}_{V_0})$. For ease of notation, consider the case of only two levels, V_0 and V_1, and let $U_0 \subset V_0$ be the subset of V_0 that can be mapped bijectively to the nodes of V_1. The expected value with respect to $P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1})$ simplifies

$$\begin{aligned}
\mathbb{E}\left[f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1})\right] &= \int f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} d\boldsymbol{x}_{V_1} \\
&= \int f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) P(\boldsymbol{x}_{V_0}) \delta(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0}) d\boldsymbol{x}_{V_1} d\boldsymbol{x}_{V_0} \\
&= \int \left(\int f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) \delta(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0}) d\boldsymbol{x}_{V_1} \right) P(\boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} \\
&= \int f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{U_0}) P(\boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} \\
&= \mathbb{E}_{P(\boldsymbol{x}_{V_0})}\left[f(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{U_0})\right],
\end{aligned}$$

reducing itself to an expected value with respect to the original distribution. An example with more lattice levels may be constructed analogously, with the δ distribution used to remove the coarse lattice variables by replacing with the appropriate subsets of the fine variables $\boldsymbol{x}_{V_0}$.

In particular, the function projected in the generalized fast marginalization framework is

$$\begin{aligned}
f(\boldsymbol{x}_{V_1}) &= \mathbb{E}\left[\frac{P'(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0})}{P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0})} \middle| \boldsymbol{x}_{V_1} \right] \\
&= \mathbb{E}\left[\frac{\delta'(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0})}{\delta(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0})} \middle| \boldsymbol{x}_{V_1} \right],
\end{aligned} \tag{6.1}$$

where δ' is the derivative of the δ distribution defined through

$$\int \delta'(x-y)f(x)dx = f'(y).$$

Equation 6.1 is only a formal expectation and an abuse of notation. However, expanding the definition of the conditional expected value we obtain

$$\begin{aligned} f(\boldsymbol{x}_{V_1}) &= \frac{\int \delta'(x_{V_1} - \boldsymbol{x}_{U_0}) P(\boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0}}{\int P(\boldsymbol{x}_{V_0}) \delta(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0}) d\boldsymbol{x}_{V_0}} \\ &= \frac{\int \delta'(x_{V_1} - \boldsymbol{x}_{U_0}) P(\boldsymbol{x}_{V_0 \setminus U_0}, \boldsymbol{x}_{U_0}) d\boldsymbol{x}_{U_0} d\boldsymbol{x}_{V_0 \setminus U_0}}{\int P(\boldsymbol{x}_{V_0 \setminus U_0}) \delta(\boldsymbol{x}_{V_1} - \boldsymbol{x}_{U_0}) d\boldsymbol{x}_{U_0} d\boldsymbol{x}_{V_0 \setminus U_0}} \\ &= \frac{\int P'(\boldsymbol{x}_{V_0 \setminus U_0}, \boldsymbol{x}_{V_1}) d\boldsymbol{x}_{V_0 \setminus U_0}}{\int P(\boldsymbol{x}_{V_0 \setminus V_1}) d\boldsymbol{x}_{V_0 \setminus U_0}}, \end{aligned}$$

which matches the fast marginalization equation of Section 4.1.

6.6 MAJORITY RULE AND THE ISING MODEL

We close the discussion of the generalized acyclic Monte Carlo with a complete example of the generalized acyclic Monte Carlo applied to the Ising model in two dimensions, coarsened under the majority rule.

The majority rule is an interpolation rule designed for discrete variables and used widely in the physics community (see, e.g. Brandt and Ron, 2001b; Gupta and Cordery, 1984; Ron and Swendsen, 2001). Setting the coarse variable x_v to the average value of $\boldsymbol{x}_{U_v}$ would cause the coarse variable to take on a wider set of values than the original variables, leading to an increased complexity. Instead, the majority rule forces the coarse variable to take the value that occurs most frequently among the fine variables $\boldsymbol{x}_{U_v}$; in case of a tie, one of the values is chosen at random.

Table 6.4: Values of the majority coarsening rule $P(x_5 \mid \sum_{i=1}^4 x_i)$ for the two-dimensional Ising model. The differentiable extension $\tilde{P}(\chi_5 \mid \sum_{i=1}^4 x_i)$ and its partial derivative with respect to χ_5 is also included.

$\sum_{i=1}^4 x_i$	$P\left(x_5 \middle\vert \sum_{i=1}^4 x_i\right)$		$\tilde{P}\left(\chi_5 \middle\vert \sum_{i=1}^4 x_i\right)$	$\tilde{P}'\left(\chi_5 \middle\vert \sum_{i=1}^4 x_i\right)$
	$x_5 = -1$	$x_5 = 1$		
4	0	1	$\left(\frac{1+\chi_5}{2}\right)^p$	$\frac{p}{2}\left(\frac{1+\chi_5}{2}\right)^{p-1}$
2	0	1	$\left(\frac{1+\chi_5}{2}\right)^p$	$\frac{p}{2}\left(\frac{1+\chi_5}{2}\right)^{p-1}$
0	$1/2$	$1/2$	$1/2$	0
-2	1	0	$\left(\frac{1-\chi_5}{2}\right)^p$	$-\frac{p}{2}\left(\frac{1-\chi_5}{2}\right)^{p-1}$
-4	1	0	$\left(\frac{1-\chi_5}{2}\right)^p$	$-\frac{p}{2}\left(\frac{1-\chi_5}{2}\right)^{p-1}$

6.6.1 *Coarsening rule*

Consider a two-dimensional lattice of size 16×16. We coarsen it by dividing the original variables V into subsets of size 2×2, leading to a set of 16×16 coarse nodes U. The coarsening block is build of $2 \times 2 + 1$ variables, the 4 fine variables x_1, x_2, x_3, x_4 and the assigned coarse variable x_5, which is subsequently made continuous and denoted χ_5.

Since the variables may only take the values of -1 or 1, there are $2^5 = 32$ possible states of the coarsening block. Therefore, we define $P(x_5 \mid x_1, x_2, x_3, x_4)$ for each of the 32 states; however, due to the symmetries present in the problem, there are only 10 distinguishable states that depend on the sum of the fine variables $\sum_{i=1}^4 x_i$ and the value of th coarse variable x_5. The values of the conditional probability $P(x_5 \mid \sum_{i=1}^4 x_i)$ are summarized in Table 6.4.

The differentiable extension $\tilde{P}\left(\tilde{x}_5 \middle\vert \sum_{i=1}^4 x_i\right)$ is then constructed as a simple polynomial passing through the given values for $\tilde{x}_5 = -1$ and 1. The additional parameter p is included for additional flexibility. We use it in Section 6.6.5 to show that the generalized fast marginalization is independent of the choice of differentiable extension.

6.6.2 *Generalized fast marginalization*

The generalized fast marginalization formulae may be evaluated by substituting the formulae from Table 6.4 into the equations developed in Section 6.3.

6.6.2.1 *Non-symmetrized projection*

Let the basis function ϕ_i^u, for a node $u \in U$, be the basis function ϕ_i centered around the node u. Given a sequence of MCMC samples $\boldsymbol{x}_{V\cup U}^k$, for $k = 1, 2, \dots, n$, we compute a matrix $A(\chi_u)$, on the lattice U and for each Gaussian integration node, using the formula

$$A_{ij}(\chi_u) = \frac{1}{|U|}\frac{1}{n}\sum_{u\in U}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_U^k)\phi_j^u(\boldsymbol{x}_U^k)\tilde{P}(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V).$$

We take advantage of the fact that the Ising model and our renormalized models are translation invariant; thus we may average the expected values over all variables on the given lattice.

Since the projection is performed on the lattice U with seven Gaussian integration nodes, we need to compute a total of seven projection matrices. Similarly, the right hand side vector $b_i(\chi_u)$ is estimated from the random samples through

$$b_i(\chi_u) = \frac{1}{|U|}\frac{1}{n}\sum_{u\in U}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_U^k)\tilde{P}'(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V).$$

As above, the set of variables $\boldsymbol{x}_{U_u}$, $U_u \subset V$, has the variable x_u, $u \in U$, assigned as the coarse variable. Note that the normalization constant is the same in both $A_{ij}(\chi_u)$ and $b_i(\chi_u)$, so that it cancels out.

6.6.2.2 *Partially-symmetrized projection*

In the partially-symmetrized case, the linear projection equations include the correction formula $\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})$. It must be evaluated numerically using the current approximation of the coefficients $c_i(\chi_u)$. Given a sequence of samples $\boldsymbol{x}_{V\cup U}^k$, for $k = 1, 2, \dots, n$, we compute the matrices $A(\chi_u)$ at each Gaussian integration node χ_u using the formula

$$A_{ij}(\chi_u) = \frac{1}{|U|}\frac{1}{n}\sum_{u\in U}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_U^k)\phi_j^u(\boldsymbol{x}_U^k)\tilde{P}(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V).$$

Similarly, the right hand side vector $b_i(\chi_u)$ is estimated from the random samples through

$$b_i(\chi_u) = \frac{1}{2}\frac{1}{|U|}\frac{1}{n}\sum_{u\in V_l}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_{V_l}^k) \times \tilde{P}'(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V)\left(1+\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})\right).$$

6.6.2.3 *Fully-symmetrized projection*

In the fully-symmetrized case, the linear projection equations become slightly more involved. Given a sequence of samples $\boldsymbol{x}_{V\cup U}^k$, for $k = 1, 2, \ldots, n$, we compute the matrices $A(\chi_u)$ at each Gaussian integration node χ_u using the formula

$$A_{ij}(\chi_u) = \frac{1}{2}\frac{1}{|U|}\frac{1}{n}\sum_{u\in U}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_U^k)\phi_j^u(\boldsymbol{x}_U^k) \times \left(\tilde{P}(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V) + \tilde{P}(\boldsymbol{x}_{U_u}^k, -\chi_u \mid \boldsymbol{x}_V)\right).$$

Similarly, the right hand side vector $b_i(\chi_u)$ is estimated from the random samples through

$$b_i(\chi_u) = \frac{1}{4}\frac{1}{|U|}\frac{1}{n}\sum_{u\in V_l}\sum_{k=1}^{n}\phi_i^u(\boldsymbol{x}_{V_l}^k) \times \Big(\tilde{P}'(\boldsymbol{x}_{U_u}^k, \chi_u \mid \boldsymbol{x}_V)\left(1+\mathcal{R}(\chi_u, \boldsymbol{x}_{N(u)})\right) + \tilde{P}'(\boldsymbol{x}_{U_u}^k, -\chi_u \mid \boldsymbol{x}_V)\left(1+\mathcal{R}(-\chi_u, \boldsymbol{x}_{N(u)})\right)\Big).$$

6.6.3 *Choice of basis*

We select the basis functions to ensure that the coefficients are comparable with the literature. As a result we will include linear and cubic functions of the lattice spins reaching up to a distance of $\sqrt{2}$, forming a neighborhood P_{12} in the notation of Brandt and Ron (2001b).

The probability distribution of the Ising model on a square Cartesian lattice is invariant with respect to rotations by $\pi/2$, π and $3\pi/2$, and flips along the 0, $\pi/4$, $\pi/2$ and $3\pi/4$ axes. As a result, the number of basis functions reduces significantly. For example, the four nearest neighbor basis functions

$$\phi_1^1 = x_{i+1,j}, \qquad \phi_1^2 = x_{i-1,j}$$

$$\phi_1^3 = x_{i,j+1}, \qquad \phi_1^4 = x_{i,j-1}$$

reduce to the single basis function $\phi_1^u(\boldsymbol{x}_{N(u)}) = x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1}$. For a node u at position (i, j) the complete set of basis functions used in this example calculation becomes:

$$\begin{aligned}
\phi_1^u(\boldsymbol{x}_{N(u)}) &= x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1} \\
\phi_2^u(\boldsymbol{x}_{N(u)}) &= x_{i+1,j+1} + x_{i+1,j-1} + x_{i-1,j+1} + x_{i-1,j-1}, \\
\phi_3^u(\boldsymbol{x}_{N(u)}) &= x_{i+1,j}x_{i+1,j+1}x_{i,j+1} + x_{i,j+1}x_{i-1,j+1}x_{i-1,j} \\
&\quad + x_{i-1,j}x_{i-1,j-1}x_{i,j-1} + x_{i,j-1}x_{i+1,j-1}x_{i+1,j}, \\
\phi_4^u(\boldsymbol{x}_{N(u)}) &= 1, \\
\phi_5^u(\boldsymbol{x}_{N(u)}) &= x_{i,j-1}x_{i-1,j} + x_{i,j+1}x_{i-1,j+1} + x_{i+1,j}x_{i+1,j-1} \\
&\quad + x_{i,j-1}x_{i+1,j} + x_{i,j+1}x_{i+1,j+1} + x_{i-1,j}x_{i-1,j-1} \\
&\quad + x_{i,j-1}x_{i-1,j-1} + x_{i-1,j}x_{i,j+1} + x_{i+1,j}x_{i+1,j+1} \\
&\quad + x_{i,j-1}x_{i+1,j-1} + x_{i+1,j}x_{i,j+1} + x_{i-1,j}x_{i-1,j+1}.
\end{aligned}$$

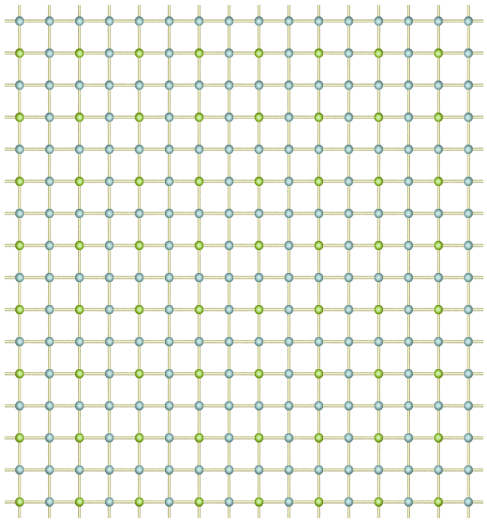
Because the last two interactions are even, they are omitted in case of the symmetrized projection schemes: the symmetrized projection obtains the odd part of the projected function, thus ensuring that the coefficients corresponding to these functions are zero. Our basis has been generated automatically by considering interactions whose radius is at most $\sqrt{2}$. We then reduce the resulting set of basis functions using lattice symmetries, following the Algorithms 4.1 and 4.2.

6.6.4 *Computational results*

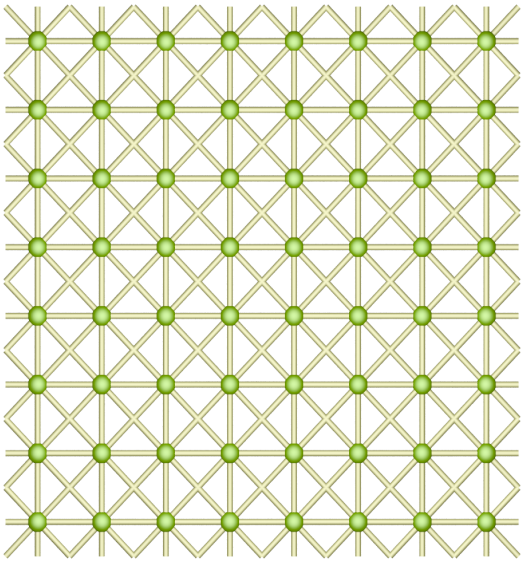
We use a fixed-point iteration with Robbins-Monro smoothing to compute the coefficients, beginning with all coefficients equal to zero (Robbins and Monro, 1951). Therefore, no symmetrization correction is applied initially. After three iterations the coefficients begin to stabilize, with further changes due only to the stochastic nature of the algorithm. The final values are collected in Tables 6.5, 6.6 and 6.7; they are analyzed in more detail and compared with the literature in Section 8.1.2.

6.6.4.1 *Decimation coefficients*

As a final step we show the ease with which the code may be adapted to different coarsening rules. We will implement the decimation rule, which recovers the coefficients that would be obtained using the original fast

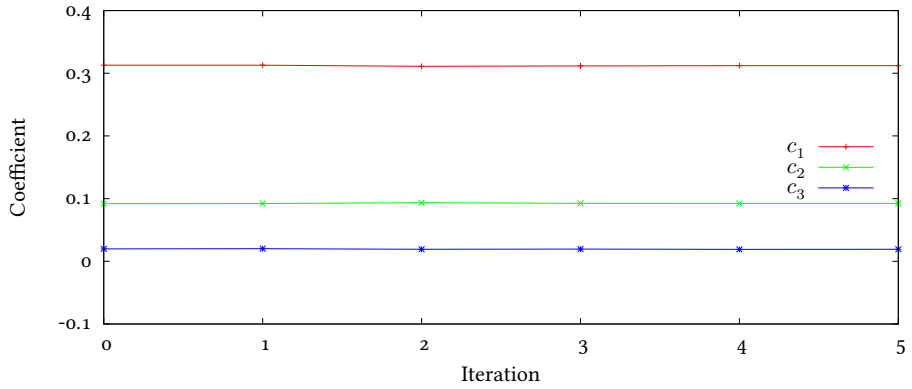


(a)

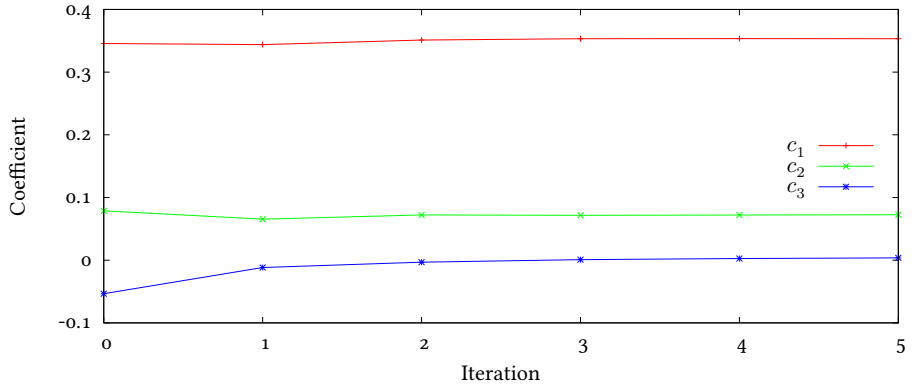


(b)

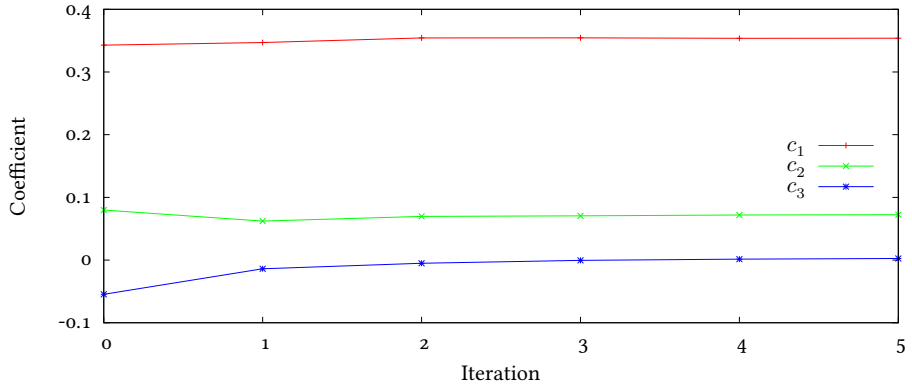
Figure 6.24: Visualization of the arrangement of nodes, showing (a) the original lattice V and (b) the sublattice U. The nodes $U \subset V$ are marked green on both images.



(a)



(b)



(c)

Figure 6.25: Convergence of the coefficients c_i of the relevant basis functions under no symmetrization, partial symmetrization and full symmetrization.

marginalization algorithm. The relevant coarsening rule and its differentiable extension are summarized in Table 6.8.

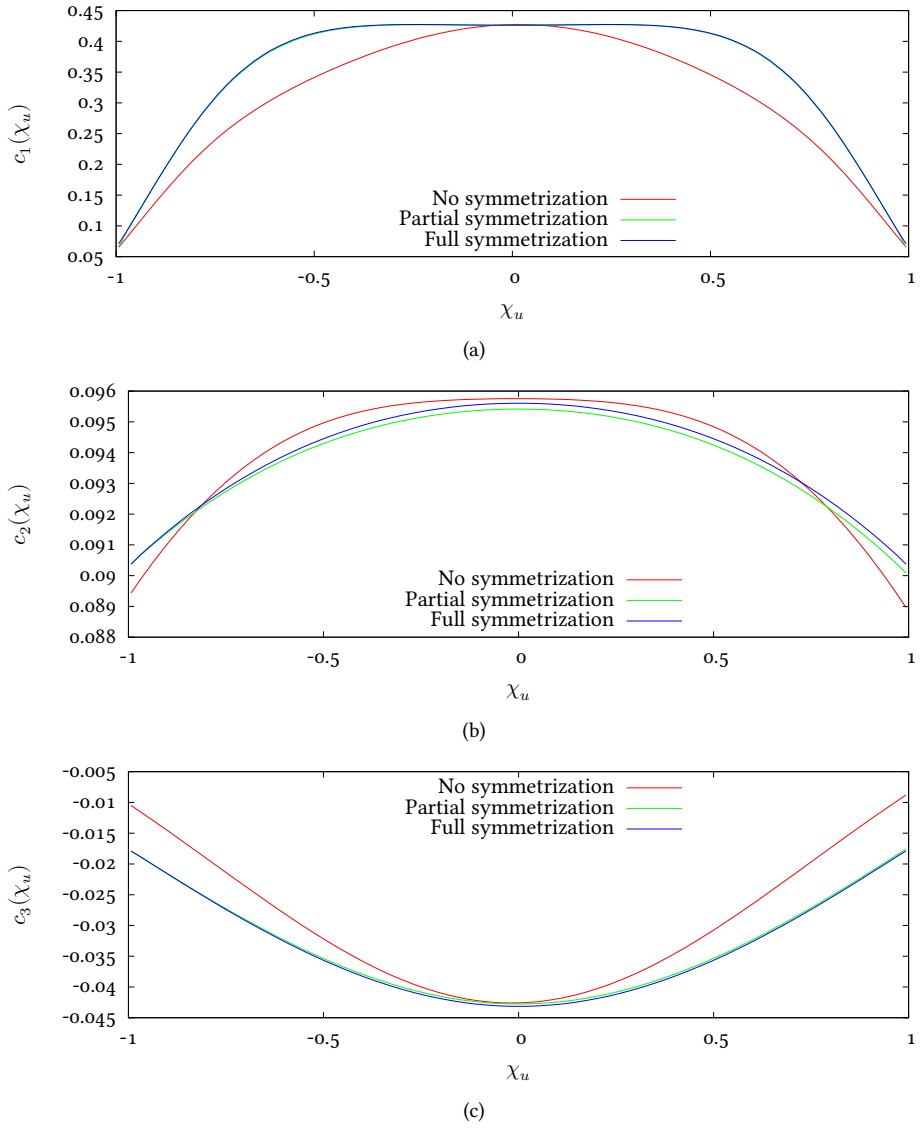


Figure 6.26: The χ_u dependence of the coefficients $c_i(\chi_u)$ of the relevant basis functions under (a) no symmetrization, (b) partial symmetrization and (c) full symmetrization.

The resulting coefficients are collected in Tables 6.9, 6.10 and 6.11, while a comparison with the results of Swendsen (1984b) is performed in Section 8.1.1.

Table 6.5: Values of the renormalized coefficients obtained using no symmetrization by renormalizing under majority rule a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$	$c_4(\chi_u)$	$c_5(\chi_u)$
-0.949	0.100056	0.037053	0.024955	-0.743498	0.032430
-0.742	0.244595	0.096873	0.086138	-0.618440	0.102021
-0.406	0.367577	0.107234	0.014835	-0.352476	0.079631
0.000	0.426543	0.096211	-0.061058	-0.003591	0.000394
0.406	0.371591	0.105524	0.012482	0.351198	-0.079562
0.742	0.245755	0.096303	0.087381	0.627242	-0.103484
0.949	0.100249	0.036878	0.025596	0.746029	-0.032889
c_i	0.311802	0.092527	0.019995	0.000401	-0.000139

Table 6.6: Values of the renormalized coefficients obtained using partial symmetrization by renormalizing under majority rule a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.117745	0.028566	0.023626
-0.742	0.313385	0.059362	0.066247
-0.406	0.423380	0.082639	-0.013794
0.000	0.426708	0.096146	-0.061208
0.406	0.423297	0.082874	-0.013998
0.742	0.313297	0.060108	0.065578
0.949	0.117662	0.028889	0.023343
c_i	0.353701	0.072120	0.003380

6.6.5 *Differentiable extension independence*

We close this chapter with a demonstration that the generalized fast marginalization algorithm is independent of the particular choice of the differentiable extension of the coarsening rule. As a corollary, the same holds for the original fast marginalization, which is a special case of the generalized algorithm.

We do so by considering the values of the basis coefficients at multiple values of the parameter p, varied between $p = 0.1$ and 10. The results

Table 6.7: Values of the renormalized coefficients obtained using full symmetrization by renormalizing under majority rule a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.118157	0.028355	0.023367
-0.742	0.314594	0.058797	0.065563
-0.406	0.424079	0.082218	-0.014110
0.000	0.427073	0.095902	-0.061364
0.406	0.424079	0.082218	-0.014110
0.742	0.314594	0.058797	0.065563
0.949	0.118157	0.028355	0.023367
c_i	0.354469	0.071552	0.003152

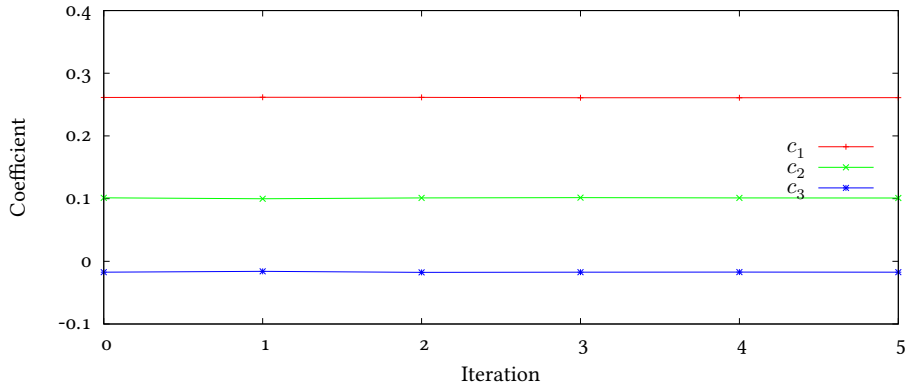
Table 6.8: Values of the decimation coarsening rule $P(x_5 \mid x_1)$ for the two-dimensional Ising model. The differentiable extension $\tilde{P}(\chi_5 \mid x_1)$ and its partial derivative with respect to χ_5 is also included.

x_1	$P(x_5 \mid x_1)$		$\tilde{P}(\chi_5 \mid x_1)$	$\tilde{P}'(\chi_5 \mid \lvert x_1)$
	$x_5 = -1$	$x_5 = 1$		
1	0	1	$\left(\frac{1+\chi_5}{2}\right)^p$	$\frac{p}{2}\left(\frac{1+\chi_5}{2}\right)^{p-1}$
-1	1	0	$\left(\frac{1-\chi_5}{2}\right)^p$	$-\frac{p}{2}\left(\frac{1-\chi_5}{2}\right)^{p-1}$

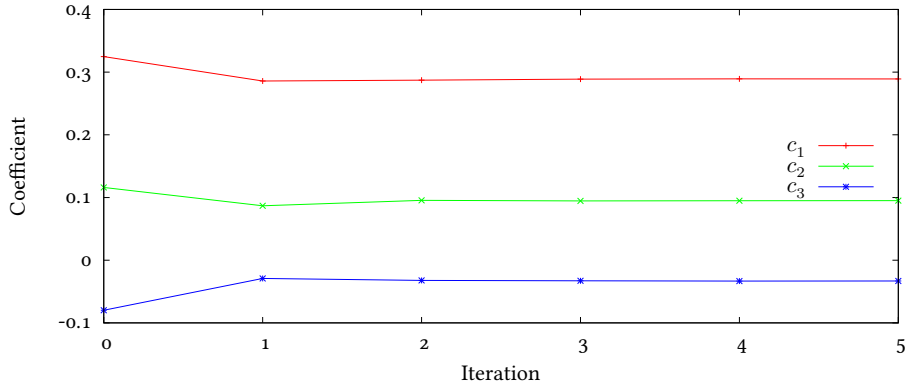
shown on Figure 6.28 show that the coefficients indeed plateau in $1 \leq p \leq 3$, but change rapidly beyond those values. Closer analysis shows this is due to the shape of the χ_u-dependence of those coefficients shown on Figure 6.29, as the Gaussian quadrature with five integration nodes is ill-equipped for handling them.

For $p < 1$, the coefficients develop integrable singularities at $\chi_u = \pm 1$. They are mild and could be handled using an appropriate Gauss-Jacobi quadrature. In the case of $p > 1$, the coefficients instead produce a hump around $\chi_u = 0$, which becomes increasingly steep as p grows. For $p > 3$, the peak is no longer captured well by the quadrature nodes, leading to significant integration errors.

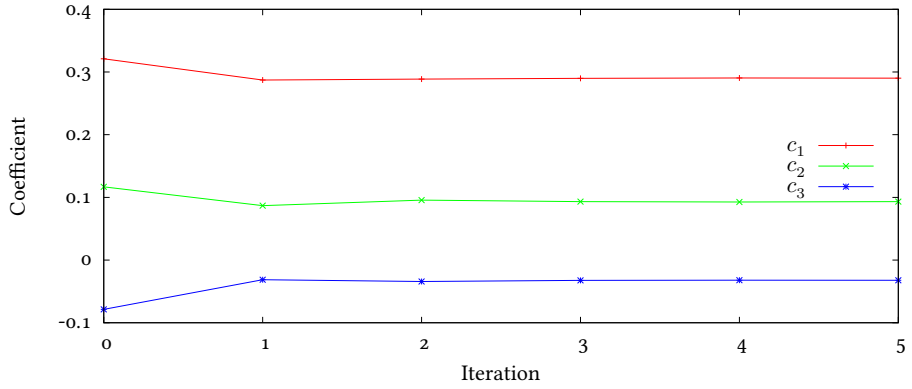
These results confirm that the generalized fast marginalization method is independent of the particular choice of the differentiable extension,



(a)



(b)



(c)

Figure 6.27: Convergence of the coefficients c_i of the relevant basis functions under no symmetrization, partial symmetrization and full symmetrization.

however, its numerical performance is not: different choices of extension lead to different shapes of $c_i(\chi_u)$, which may require sophisticated inte-

Table 6.9: Values of the renormalized coefficients obtained using no symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$	$c_4(\chi_u)$	$c_5(\chi_u)$
-0.949	0.026599	0.011800	0.008686	-0.974539	0.011677
-0.742	0.157752	0.068127	0.038562	-0.813903	0.061917
-0.406	0.332249	0.129183	-0.018009	-0.444314	0.070220
0.000	0.411021	0.147812	-0.102014	-0.002758	0.000273
0.406	0.334217	0.128856	-0.019396	0.440596	-0.070014
0.742	0.158838	0.068065	0.038513	0.813693	-0.062261
0.949	0.026788	0.011802	0.008734	0.974595	-0.011769
c_i	0.260866	0.100728	-0.016553	-0.001312	0.000042

Table 6.10: Values of the renormalized coefficients obtained using partial symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.037882	0.009924	0.000329
-0.742	0.201795	0.056809	0.022310
-0.406	0.374388	0.119285	-0.045461
0.000	0.411420	0.147539	-0.102164
0.406	0.374460	0.119166	-0.045411
0.742	0.202063	0.056327	0.022439
0.949	0.038240	0.009313	0.000427
c_i	0.290354	0.093424	-0.032392

gration methods. Therefore it is of utmost importance that the coefficients $c_i(\chi_u)$ are integrated correctly, as otherwise the iterative search may fail to converge.

Table 6.11: Values of the renormalized coefficients obtained using full symmetrization by renormalizing under decimation a 16×16 Ising lattice at $T = 2.269185$.

χ_u	$c_1(\chi_u)$	$c_2(\chi_u)$	$c_3(\chi_u)$
-0.949	0.038241	0.009295	0.000458
-0.742	0.201152	0.056953	0.022479
-0.406	0.373502	0.119936	-0.045382
0.000	0.410840	0.148085	-0.102168
0.406	0.373502	0.119936	-0.045382
0.742	0.201152	0.056953	0.022479
0.949	0.038241	0.009295	0.000458
c_i	0.289686	0.093875	-0.032332

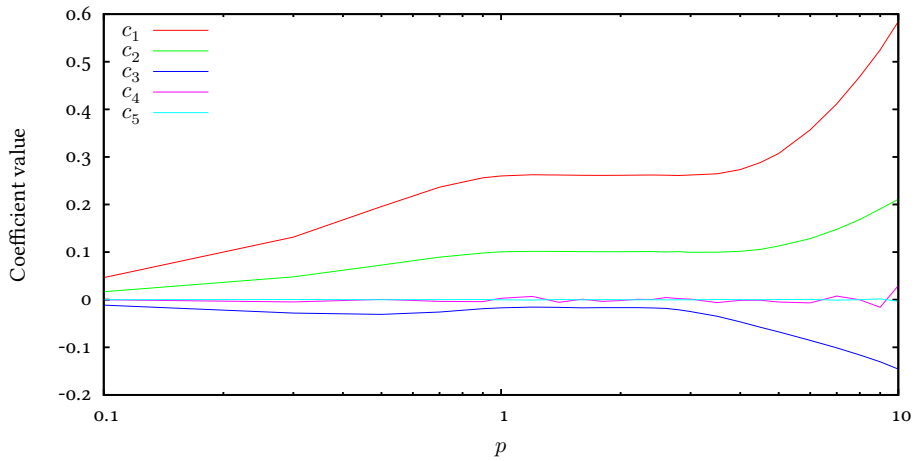


Figure 6.28: Dependence of the renormalized coupling coefficients c_i on the choice of the parameter p when integrated using the five-point Gaussian quadrature rule. The differences between values for different p are caused by the inadequate number of integration nodes.

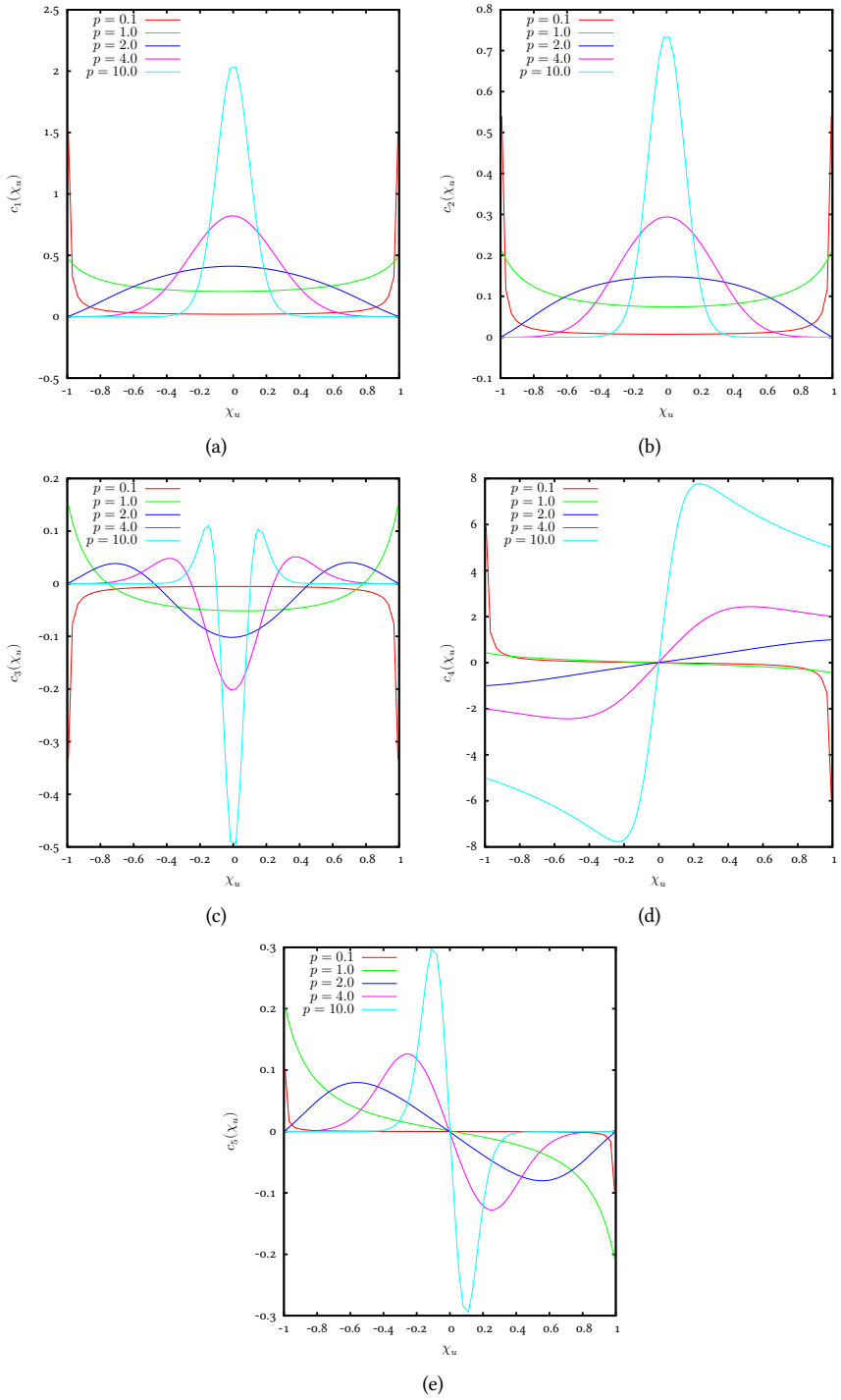


Figure 6.29: Shape of the coupling coefficients $c_i(\chi_u)$ for different values of the parameter p, showcasing the strong dependency on p. The optimal value of p leading to the least-complex shape is dependent on the coupling μ of the original model.

Part II

APPLICATIONS

7

RENORMALIZATION AND PARAMETER FLOW

In this chapter we will briefly discuss the parameter flow induced by renormalization. Parameter flows are often described in the literature, but often without much theoretical grounding. We begin by discussing one of the few rigorously definable cases of parameter flow, the one-dimensional Ising model undergoing coarsening under decimation. In later sections, we generalize this notion using linear projection and investigate the parameter flow of the two-dimensional Ising model under different coarsening rules.

7.1 EXACT PARAMETER FLOW

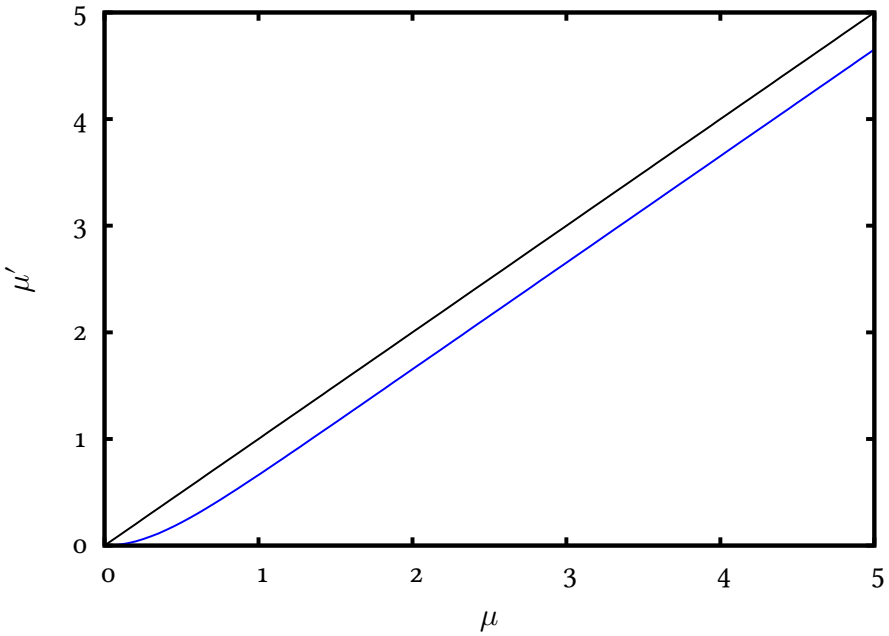
Appendix A shows in detail that a one-dimensional Ising model does not change its graphical structure when coarsened using decimation. The sole changes are the reduction in the number of variables and modification of the coupling constant, with the coarse constant μ' linked to the original coupling μ via

$$\mu' = R(\mu) = \frac{1}{2} \ln \cosh 2\mu.$$

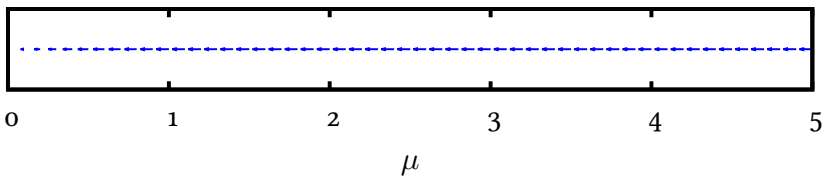
This function is a map $R : \mathbb{R} \longrightarrow \mathbb{R}$, transforming the coefficient μ in the original, fine probability distribution to the coefficient μ' of the coarse, renormalized probability distribution. Since we may think of $R(\mu)$ as moving the system in phase space, we are interested in the change in the coefficients caused by an application of $R(\mu)$. We say that the map $R(\mu)$ induces a vector field $F(\mu)$ defined as

$$F(\mu) = R(\mu) - \mu,$$

which describes the change in the coefficients due to the application of $R(\cdot)$ to a system described by μ. In the case of the one-dimensional Ising model the vector field reduces to a pseudo-scalar field. The vector field $F(\mu)$ specifies the direction and magnitude of the change in the coupling parameters. We will hereafter use $F(\mu)$ to describe the parameter flow under renormalization. Figures 7.30a and 7.30b show the map $R(\mu)$ and the resulting parameter flow field $F(\mu)$. From the above figures we see



(a) Coefficient mapping $R(\mu)$ and the identity mapping.



(b) Vector field $F(\mu)$.

Figure 7.30: Coefficient mapping $R(\mu)$ and the induced parameter flow vector field $F(\mu)$.

that the parameter flow gives useful information about the probabilistic model, in this case the one-dimensional Ising model. Since $R(\mu) \leq \mu$, the vector field $F(\mu)$ always points toward smaller couplings. As a result, the spins at subsequently coarser scales appear less and less coupled, eventually becoming entirely uncoupled as $\mu \longrightarrow 0$. The mapping $R(\mu)$ possesses only one fixed point $\mu_* = 0$, defined as the point μ_* where

$$R(\mu_*) = \mu_* \quad \text{or, equivalently,} \quad F(\mu_*) = 0.$$

As seen on Figure 7.30b, all the flow from all other coupling coefficients leads towards the zero coupling fixed point, decreasing the coupling.

The fixed points are the most interesting object that may be studied using the parameter flow, because they describe the behavior of the model at macroscopic scales. In statistical physics, unstable fixed points (Arnold, 1973; Coddington and Levinson, 1955), referred to as critical points, are of special interest. Their presence of indicates abrupt changes in the behavior of the system under study, e.g. phase transitions.

The one-dimensional Ising model has only one fixed point at $\mu_* = 0$. Since the fixed point is stable, that is, a perturbed system with $\mu = \epsilon$ returns to the fixed point when the mapping $R(\mu)$ is repeatedly applied to it. Therefore, there are no critical points and the one-dimensional Ising model is free of phase transitions.

7.2 PROJECTED PARAMETER FLOW

The $R(\boldsymbol{\mu})$ may be also defined in a general setting where the map can no longer be obtained in closed form. We proceed in the following order. First, we define the fine and coarse probability distributions using the approach used in Chapter 6. The couplings $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$ are then defined as the coefficients of expansions of the Hamiltonians associated with the fine and coarse lattices, respectively. In the following we consider a generalized Ising model $P(\boldsymbol{x}_{V_0})$ that allows further interactions in addition to the typical nearest neighbor coupling.

Let the fine probability distribution $P(\boldsymbol{x}_{V_0})$ describe a translation invariant probabilistic spin model on a square Cartesian lattice of size $n \times n$, n a power of two, with periodic boundary conditions. Define a set of coarse variables $\boldsymbol{x}_{V_1}$ by assigning a coarse variable to each subset of 2×2 fine variables $\boldsymbol{x}_{V_0}$ and let the conditional probability of the coarse variables $\boldsymbol{x}_{V_1}$, given the fine variables $\boldsymbol{x}_{V_0}$, be $P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0})$. The joint probability distribution of $\boldsymbol{x}_{V_0}$ and $\boldsymbol{x}_{V_1}$ is then

$$P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) = P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}),$$

allowing us to define the coarse probability $P(\boldsymbol{x}_{V_1})$ as the marginal probability distribution

$$\begin{aligned} P(\boldsymbol{x}_{V_1}) &= \int P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) d\boldsymbol{x}_{V_0} \\ &= \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0}. \end{aligned}$$

Consider the case where the conditional probability factors as

$$P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) = \prod_{u \in V_1} P(\boldsymbol{x}_u \mid \boldsymbol{x}_{U_u}),$$

with U_u being the 2×2 subset of V_0 that was assigned x_u, $u \in V_1$, as its coarse variable. Assuming that the probability $P(\boldsymbol{x}_u \mid \boldsymbol{x}_{U_u})$ has the same functional form for all $u \in V_1$, the translation invariance of the original probability $P(\boldsymbol{x}_{V_0})$ implies that $P(\boldsymbol{x}_{V_1})$ is also translation invariant. Consider the case where both $P(\boldsymbol{x}_{V_0}) > 0$ and $P(\boldsymbol{x}_{V_1}) > 0$. Thus, we define the Hamiltonians $W_0(\boldsymbol{x}_{V_0})$ and $W_1(\boldsymbol{x}_{V_1})$ as the logarithms of the respective probability densities,

$$W_0(\boldsymbol{x}_{V_0}) = \ln P(\boldsymbol{x}_{V_0}) \quad \text{and} \quad W_1(\boldsymbol{x}_{V_0}) = \ln P(\boldsymbol{x}_{V_1}).$$

Let X_0 and X_1 be the vector spaces of functions over the fine variables $\boldsymbol{x}_{V_0}$ and the coarse variables $\boldsymbol{x}_{V_1}$, respectively, and let them have bases $\boldsymbol{\xi}$ and $\boldsymbol{\chi}$. For any $u \in V_0$ and $v \in V_1$ define the functions

$$f(\boldsymbol{x}_{V_0}) = W_0(\boldsymbol{x}_{V_0 \setminus u}, x_u = 1) - W_0(\boldsymbol{x}_{V_0 \setminus u}, x_u = -1)$$

and

$$f'(\boldsymbol{x}_{V_1}) = W_1(\boldsymbol{x}_{V_1 \setminus v}, x_v = 1) - W_1(\boldsymbol{x}_{V_1 \setminus v}, x_v = -1).$$

Let $f \in X_0$ and $f' \in X_1$ be written as

$$f(\boldsymbol{x}_{V_0}) = \sum_{i=1}^{\dim X_0} \mu_i \xi_i \quad \text{and} \quad f'(\boldsymbol{x}_{V_1}) = \sum_{i=1}^{\dim X_1} \mu'_i \chi_i$$

in terms of the basis functions of the spaces X_0 and X_1, respectively. Finally, define the mapping R by

$$R(\boldsymbol{\mu}) = \boldsymbol{\mu}',$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2, \dots, \mu_{\dim X_0})$ and $\boldsymbol{\mu}' = (\mu'_1, \mu'_2, \dots, \mu'_{\dim X_1})$. Given a fixed choice of bases $\boldsymbol{\psi}$ and $\boldsymbol{\xi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$ are uniquely determined.

While the parameter flow of one-dimensional Ising model could be studied rigorously, this is not generally possible in the case of more complex models. The process of renormalization, i.e., marginalization of coarse variables leads to increased connectivity of the dependence graph and vastly increased number of functions necessary to describe $P(\boldsymbol{x}_{V_1})$. There-

fore, numerical approximations are necessary for the study of parameter flows.

The numerical studies available in literature implicitly restrict $R(\boldsymbol{\mu})$ to a subspace of $X_0 \cap X_1$ and we shall proceed in a similar fashion (Binney et al., 1992; Gupta and Cordery, 1984; Nauenberg and Nienhuis, 1974b; Nienhuis and Nauenberg, 1975). Let X_ϕ be vector space of dimension K spanned by a basis of functions $\phi = \{\phi_1, \phi_2, \dots, \phi_K\}$, such that X_ϕ is a subspace of both X_0 and X_1, written as $X_\phi \leq X_0 \cap X_1$. Finally, define a projection operator $\mathbb{P}_{X_\phi} : X \longrightarrow X_\phi$ projecting vectors of X onto X_ϕ. We can define a restricted map $\hat{R} : \mathbb{R}^K \longrightarrow \mathbb{R}^K$ as a projection of the original map R,

$$\hat{R}(\boldsymbol{\mu}) = \mathbb{P}_{X_\phi} R(\boldsymbol{\mu}).$$

The projection operator $\mathbb{P}_{X_\phi}$ will be chosen later, depending on the available tools. The resulting mapping $\hat{R}$ transforms the subspace X_ϕ spanned by the basis ϕ onto itself, allowing us to study its behavior using numerical methods.

Before we continue, we ask what may be the relation between the fixed points of the true mapping R and of the projected mapping $\hat{R}$. Unfortunately, a straightforward example visualized in Figure 7.31 shows that there may indeed be no relation between the two at all. Consider projecting the vector field onto the marked line, representing a one-dimensional subspace of the two-dimensional space. The projected vector field $\hat{F}(t) = \hat{R}(t) - t$ will only exhibit one fixed point where the vector field $\boldsymbol{F}(\boldsymbol{x}) = R(\boldsymbol{x}) - \boldsymbol{x}$ is orthogonal to the subspace, however that fixed point will not correspond to any true fixed point.

However, notice that although the bottom-left fixed point does not belong to the subspace, it lies close to it and therefore the projected vector field has low magnitude there. Thus, assuming that the distance between the marginal probability distribution and its projection onto the subspace X_ϕ is sufficiently small, one should be able to observe the main features of the true map R using the approximation $\hat{R}$. In particular, as $X_\phi \to X_1$ we will see that the approximate map approaches the true map, $\hat{R} \to R$.

7.2.1 *Direct projection*

We study the map and its induced parameter flow using two methods. The subspace X_ϕ is spanned by a basis ϕ composed of three basis functions,

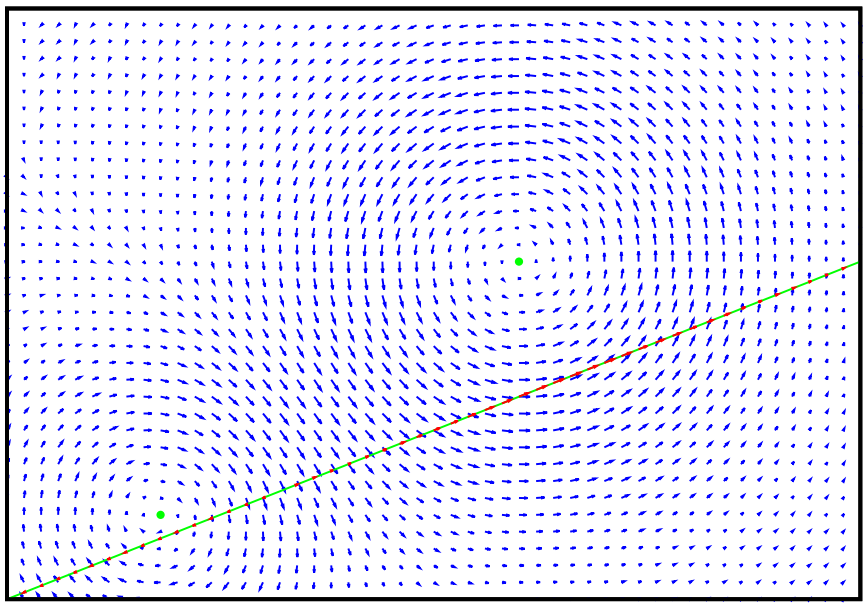


Figure 7.31: An example two-dimensional map visualized by its induced vector field. The dots mark the only finite fixed points of the original vector field, located at the centers of the swirling vortices. The projected vector field, visualized using red arrows, shows that it may be at times a poor approximation of the true vector field.

- nearest neighbor

$$\phi_1^u(\boldsymbol{x}_{V_l}) = x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1},$$

- second-nearest neighbor

$$\phi_2^u(\boldsymbol{x}_{V_l}) = x_{i+1,j+1} + x_{i+1,j-1} + x_{i-1,j+1} + x_{i-1,j-1},$$

- plaquette

$$\begin{aligned}\phi_3^u(\boldsymbol{x}_{V_l}) = {} & x_{i+1,j}x_{i+1,j+1}x_{i,j+1} + x_{i,j+1}x_{i-1,j+1}x_{i-1,j} \\ & + x_{i-1,j}x_{i-1,j-1}x_{i,j-1} + x_{i,j-1}x_{i+1,j-1}x_{i+1,j}.\end{aligned}$$

This is the biggest basis for which we may visualize the parameter flow in three dimensions; it was used by Binney et al. (1992) and Nauenberg and Nienhuis (1974b) to study the fixed points of the renormalization map R.

We will use the subspace X_ϕ spanned by this basis in the remainder of this chapter.

We begin with an exact projection of $P(\boldsymbol{x}_{V_1})$ onto an orthonormal basis. Following Binney et al. (1992, pp. 147–153), we project the coarse Hamiltonian

$$W_1(\boldsymbol{x}_{V_1}) = \ln P(\boldsymbol{x}_{V_1}) = \ln \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} \tag{7.1}$$

onto the subspace X_ϕ using a uniform inner product. We define the projection operator $\mathbb{P}_{X_\phi}$ as

$$\mathbb{P}_{X_\phi} f(\boldsymbol{x}_{V_1}) = \underset{g \in X_\phi}{\arg\min} \int \left(f(\boldsymbol{x}_{V_1}) - g(\boldsymbol{x}_{V_1}) \right)^2 d\boldsymbol{x}_{V_1},$$

leading to

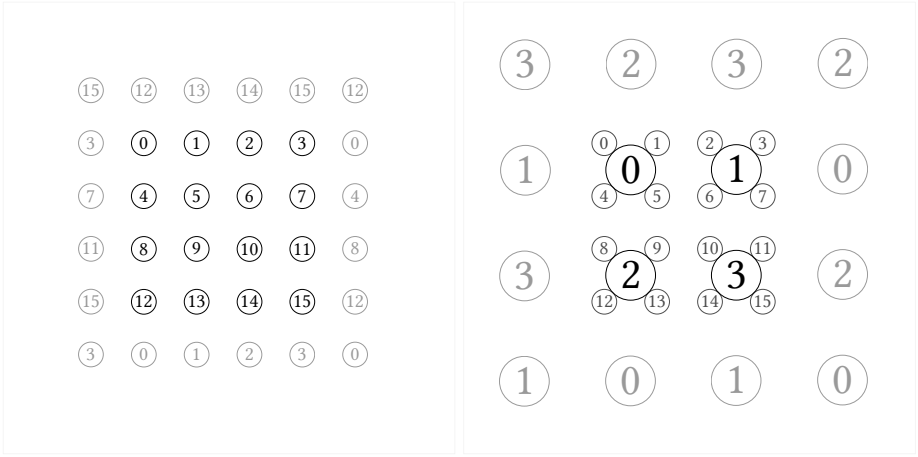
$$\begin{aligned} \mu_i' &= 2^{-n^2} \int W_1(\boldsymbol{x}_{V_1}) \phi_i(\boldsymbol{x}_{V_1}) d\boldsymbol{x}_{V_1} \\ &= 2^{-n^2} \int \phi_i(\boldsymbol{x}_{V_1}) \left(\ln \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} \right) d\boldsymbol{x}_{V_1}. \end{aligned} \tag{7.2}$$

The resulting formula may be evaluated on very small lattices V_0, e.g. 4×4 lattices, where it requires the summation over $2^{4\times4+2\times2} = 1,048,576$ possible states. Larger lattice sizes, e.g. 6×6 lattices, are beyond computational capabilities, showing clearly the limitations of the direct projection approach.

We perform the computation using a computer program that we describe here to illustrate some nuances due to the periodic boundary conditions and degeneracy that occurs on these small lattices. We define an 4×4 fine lattice V_0 and an 2×2 coarse lattice V_1. The variables are arranged row-first, with numbers growing left-to-right and top-to-bottom, as shown on Figure 7.32. As a result, the coarse spin 0 is the group variable for the fine spins $0, 1, 4$ and 5, as shown on Figure 7.32b.

The joint probability of the two lattices $P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1})$ is computed in two stages. The probability distribution $P(\boldsymbol{x}_{V_0})$ is computed as the exponent of the Hamiltonian $W(\boldsymbol{x}_{V_0})$ defined through

$$\begin{aligned} W(\boldsymbol{x}_{V_0}) = &\frac{\mu_1}{2} \sum_{ij} x_{ij} \left(x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1} \right) \\ &+ \frac{\mu_2}{2} \sum_{ij} x_{ij} \left(x_{i+1,j+1} + x_{i+1,j-1} + x_{i-1,j+1} + x_{i-1,j-1} \right) \end{aligned}$$



(a) Fine lattice V_0 (b) Coarse lattice V_1

Figure 7.32: Arrangement of spins on the fine lattice V_0 and the coarse lattice V_1. Periodic neighborhood is shown in lighter gray.

$$+\frac{\mu_3}{4}\sum_{ij} x_{ij}\left[x_{i+1,j}\left(x_{i+1,j+1}x_{i,j+1}+x_{i+1,j-1}x_{i,j-1}\right)\right.$$
$$\left.+x_{i-1,j}\left(x_{i-1,j+1}x_{i,j+1}+x_{i-1,j-1}x_{i,j-1}\right)\right].$$

The factors $1/2$ and $1/4$ account for double- and quad-counting, because each interaction is included oner per each variable it involves. Thus, the quadratic terms are double-counted, while the quartic term is quad-counted.

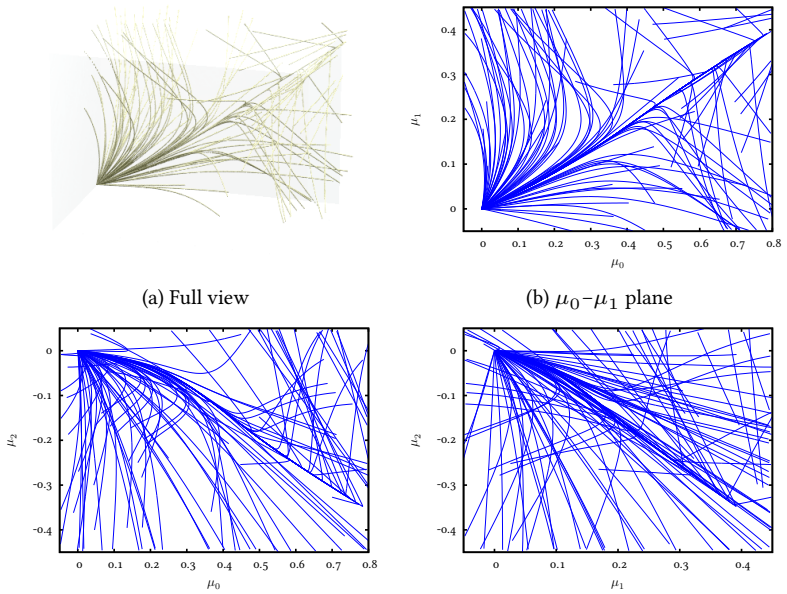
The unnormalized joint probability of the fine and coarse variables is computed as

$$P(\boldsymbol{x}_{V_0}, \boldsymbol{x}_{V_1}) = \exp\left(W_0(\boldsymbol{x}_{V_0})\right) P_\nu(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}),$$

where $P(\boldsymbol{x}_{V_1} \mod \boldsymbol{x}_{V_0})$ is a family of coarsening rules parametrized by $\nu \in [0,1]$. The value $\nu = 0$ corresponds to decimation rule, $\nu = 1$ to majority rule, while values in between to the linear combination of the two

$$P_\nu(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) = \nu P_{\text{majority}}(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) + (1-\nu) P_{\text{decimation}}(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}).$$

The conditional probabilities for decimation and majority rule are specified in Tables 6.8 and 6.4, respective. The use of the parameter $\nu \in [0,1]$ allows us to study both rules, especially the transition between the them.



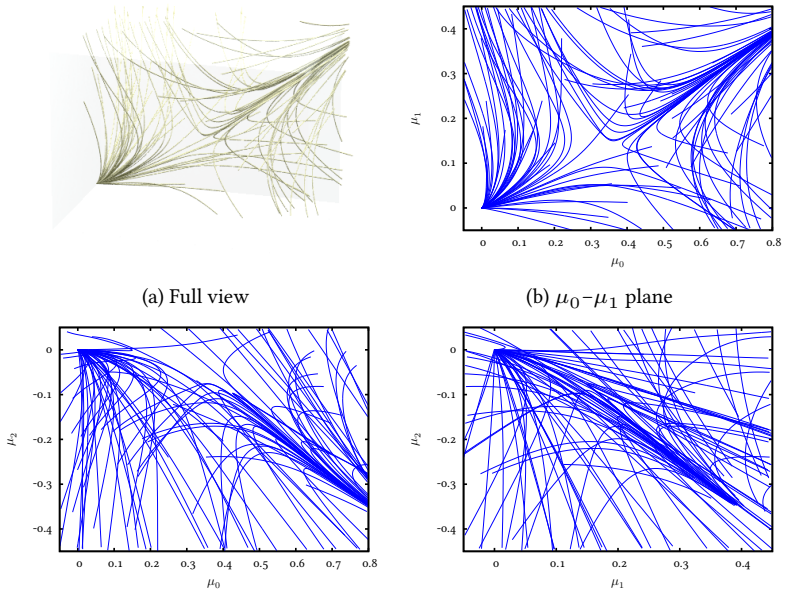
(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

Figure 7.33: Exact computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 0$ coarsening rule (decimation).



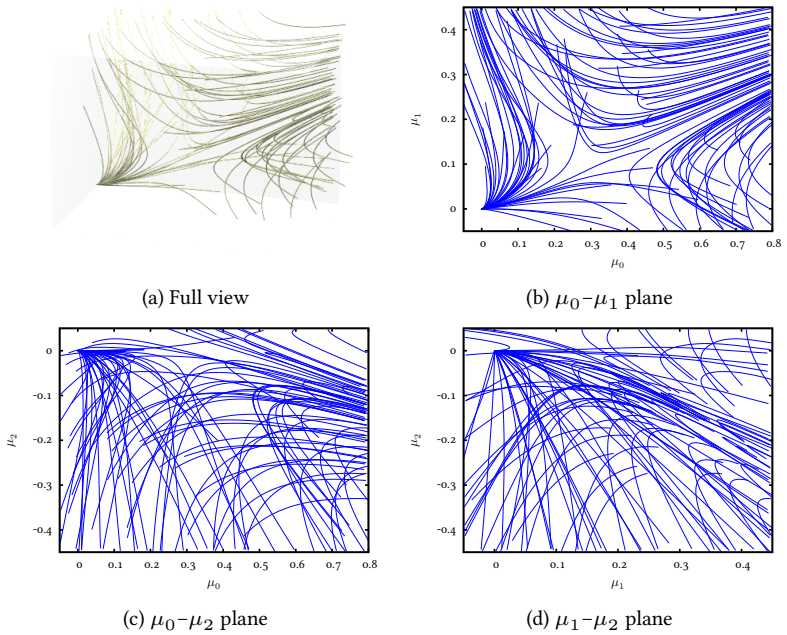
(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

Figure 7.34: Exact computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 1/2$ coarsening rule.



(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

Figure 7.35: Exact computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 1$ coarsening rule (majority rule).

The projected coefficients $\boldsymbol{\mu}' = (\mu_1', \mu_2', \mu_3')$ are computed using two loops. The inner loop computes the logarithm of the marginal of the joint distribution by recursively generating all possible states $\boldsymbol{x}_{V_0}$ for a given state $\boldsymbol{x}_{V_1}$, collecting the sum from Equation 7.1. The outer loop computes the actual projection of Equation 7.2 by recursively visiting all possible states $\boldsymbol{x}_{V_1}$ and performing the inner loop on each such state. This corresponds to computing the outer sum of

$$\int \phi_i(\boldsymbol{x}_{V_1}) \left(\ln \int P(\boldsymbol{x}_{V_0}) P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0} \right) d\boldsymbol{x}_{V_1}.$$

Once the complete integral is computed for a given i, we perform normalization, dividing each coefficient by 2^{n^2}, the constant that makes the basis functions orthonormal. Importantly, we also correct an over-counting error made by Binney et al. (1992). Figure 7.32b shows that the right-hand-side neighbor of node 0 is node 1, but the left-hand-side neighbor is also node 1. Therefore, the two coarse basis functions $x_{ij}x_{i+1,j}$ and $x_{ij}x_{i-1,j}$ cannot be distinguished by the projection algorithm and the resulting coefficient μ_1' is in fact twice as large as it should have been. We have in fact seen this situation in the exact computation of the Ising model, where a

lattice of four spins is reduced to only two spins. In that situation, the final coefficient becomes

$$\mu' = \ln\cosh(2\mu) \quad \text{rather than} \quad \mu' = \frac{1}{2}\ln\cosh(2\mu).$$

The situation is similar with the diagonal interaction $x_{ij}x_{i+1,j+1}$. In this case, this interaction connects nodes 0 and 3, but the equivalent interactions $x_{ij}x_{i+1,j-1}$, $x_{ij}x_{i-1,j-1}$ and $x_{ij}x_{i-1,j+1}$ all connect nodes 0 and 3. Similarly, the plaquette term connects the same set of four spins four times. Therefore, the coefficients μ_2' and μ_3' obtained by the projection algorithm are four times as large as they should be. We correct the overcounting errors by dividing μ_1' by two, while μ_2' and μ_3' are both divided by four.

Our code computes the projected coefficients for an entire list of original coefficient sets. The results presented here sample uniformly the rectangular cuboid defined by the bounding box

$$\begin{aligned}[\mu_0^{\min}, \mu_0^{\max}] \otimes [\mu_1^{\min}, \mu_1^{\max}] \otimes [\mu_2^{\min}, \mu_2^{\max}] \\ = [-0.05, 0.8] \otimes [-0.05, 0.45] \otimes [-0.45, 0.05],\end{aligned}$$

with each side subdivided into 30 intervals, with $N = 31$ points. Therefore, the code probes $N^3 = 29,791$ sets of initial coefficients $\boldsymbol{\mu}$ by computing the mapping $\boldsymbol{\mu}' = R(\boldsymbol{\mu})$ and vector $\boldsymbol{F}(\boldsymbol{\mu}) = R(\boldsymbol{\mu}) - \boldsymbol{\mu}$ for each one. The results are then saved as Visualization Toolkit (VTK) files for later visualization.

Figures 7.33, 7.34 and 7.35 show the results of the computation. The flow field is visualized by a set of streamlines, that is paths followed by massless marker particles flowing through the flow field. Notice that in all the images there is a clear stable fixed point at $\boldsymbol{\mu} = (0, 0, 0)$, corresponding to a zero-coupling state where variables are uncorrelated. When $\nu > 0$ an unstable fixed point, i.e. a critical point, suddenly appears and moves continuously toward the location of the fixed point of the majority rule as $\nu \to 1$. In all cases the critical point is a saddle (Arnold, 1973; Coddington and Levinson, 1955), where the streamlines enter the critical point along two directions corresponding to the negative eigenvalues of the Jacobian of the map $R(\boldsymbol{\mu})$ at $\boldsymbol{\mu}_*$. These critical streamlines flow out of the critical point along the single direction corresponding to the lone positive eigenvalue, pushing the system either toward the zero coupling state or an infinite coupling.

Classically, the flow with an unstable fixed point implies the existence of a phase transition. If the microscopic system lies in the basin of attraction

of the zero coupling state, the resulting macroscopic system, representing the microscopic system after multiple applications of the coarsening rule, is uncorrelated and unmagnetized. However, when the parameters $\boldsymbol{\mu}$ describing the system cross the boundary of the basin, the macroscopic system is instead pushed toward high couplings, resulting in a strongly correlated and magnetized state. Thus, the location of the boundary between the two attraction basins corresponds to the location of the phase transition. Since the original Ising model corresponds to a set of parameters $(\mu_0, 0, 0)$, we would expect set point $\boldsymbol{\mu}_c = (0.44068, 0, 0)$ to initiate a streamline passing through the critical point. Instead, however, we find that the continuous streamlines cannot be used to represent the evolution of the coupling coefficients undergoing subsequent steps renormalization, a discrete process. Therefore, the parameter flow under repeated renormalization can only be observed by the calculation of renormalized coefficients using a very large fine lattice and a sequence of successively smaller lattices; see, e.g., Table 8.19.

We turn to the study of the dependence of the position of the critical point $\boldsymbol{\mu}_*(\nu)$ on the coarsening rule, as the initial study has shown that decimation appears not to have a critical point at all. The critical point $\boldsymbol{\mu}_*(\nu)$ is at the closest point to the zero coupling in case of majority rule ($\nu = 0$) and steadily moves away when the coarsening rule becomes decimation, i.e. as $\nu \to 0$. Nauenberg and Nienhuis (1974a,b) reported the location of the $\nu = 1$ critical point to be $\boldsymbol{\mu}_*(1) = (0.307, 0.084, -0.004)$ (Nauenberg and Nienhuis, 1974b) or $\boldsymbol{\mu}_*(1) = (0.300, 0.0871, -0.00126)$ (Nauenberg and Nienhuis, 1974a). Using the direct projection method described above we have re-computed the location of the critical point using an iterative approach. Linearizing the parameter flow $\boldsymbol{F}(\boldsymbol{\mu})$ around the critical point $\boldsymbol{\mu}_*$, we obtain

$$\begin{aligned}\boldsymbol{F}(\boldsymbol{\mu}) &= \boldsymbol{F}(\boldsymbol{\mu}_*) + A(\boldsymbol{\mu} - \boldsymbol{\mu}_*) + \mathcal{O}(||\boldsymbol{\mu} - \boldsymbol{\mu}_*||^2) \\ &= A(\boldsymbol{\mu} - \boldsymbol{\mu}_*) + \mathcal{O}(||\boldsymbol{\mu} - \boldsymbol{\mu}_*||^2),\end{aligned}$$

where A is the Jacobian of the vector field at the critical point. Therefore, we iteratively solve for the critical point $\boldsymbol{\mu}_*$ using

$$\boldsymbol{\mu}_*^{n+1} = \boldsymbol{\mu}_*^n - A(\boldsymbol{\mu}_*^n)^{-1}\boldsymbol{F}(\boldsymbol{\mu}_*^n),$$

where $A(\boldsymbol{\mu}_*^n)$ is approximated numerically. For $\nu = 1$, starting from the initial location $\boldsymbol{\mu}_*^0 = (0.307, 0.084, -0.004)$ we converge to the value $\boldsymbol{\mu}_* = (0.29976120070883128, 0.087094327207973096, -0.0012586333545222166)$, at which point the change $||\mu_*^{n+1} - \mu_*^n||^2 \leq 10^{-12}$ becomes negligible. These

values seem to agree well with the value reported by Nauenberg and Nienhuis (1974a).

After determining the critical point μ_* for majority rule ($\nu = 1$), we vary ν and compute the function $\mu_*(\nu)$ for $\nu \in [0, 1]$. This may be done as long as the starting point μ_*^0 is close to the fixed point; therefore, we first find the value of $\mu_*(1)$ and slowly decrease ν, using the value $\mu_*(\nu + \epsilon)$ as the starting point for the $\mu_*(\nu)$ iteration. Figure 7.36 presents the dependence of the three components of $\boldsymbol{\mu}_*(\nu) = (\mu_1(\nu), \mu_2(\nu), \mu_3(\nu))$ on the coarsening rule used. Around the initial value of $\nu = 1$, the fixed point $\boldsymbol{\mu}_*(\nu)$ changes smoothly; however, as $\nu \to 0$ the position of the critical point appears to diverge.

We posit that the Ising model in two dimensions under decimation does not have a finite critical point, because further approximate studies of larger lattices, described below, show a similar behavior. Because of the wide-spread belief in the connection between the critical point of the renormalization transformation and the phase transitions of the underlying system, further studies of the topic appear much needed.

7.2.2 *Approximate projection*

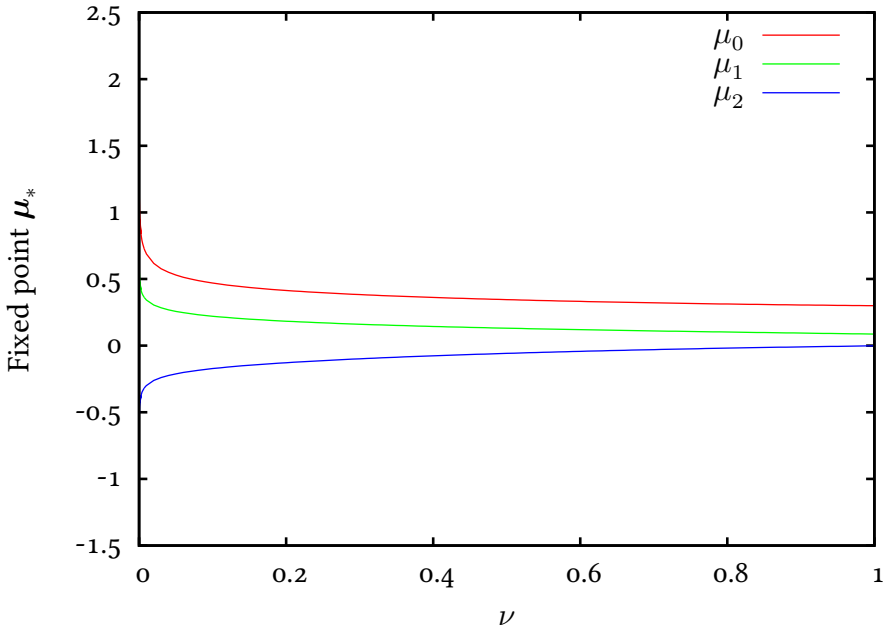
The direct projection method described in the previous section is unable to provide us with a truthful description of the parameter flow, due to the very small lattice size that can be handled. As we have seen above, this restriction is severe because results are dominated by finite size effects. Computing the renormalized probability distribution using a larger lattice necessitates an approximate approach.

We apply the generalized fast marginalization method and define the parameter map $\tilde{R}$ as the outcome of the projection performed by the resulting algorithm. Let $\mathbb{P}_{X_\phi}^{\chi_u}$ be the projection operator

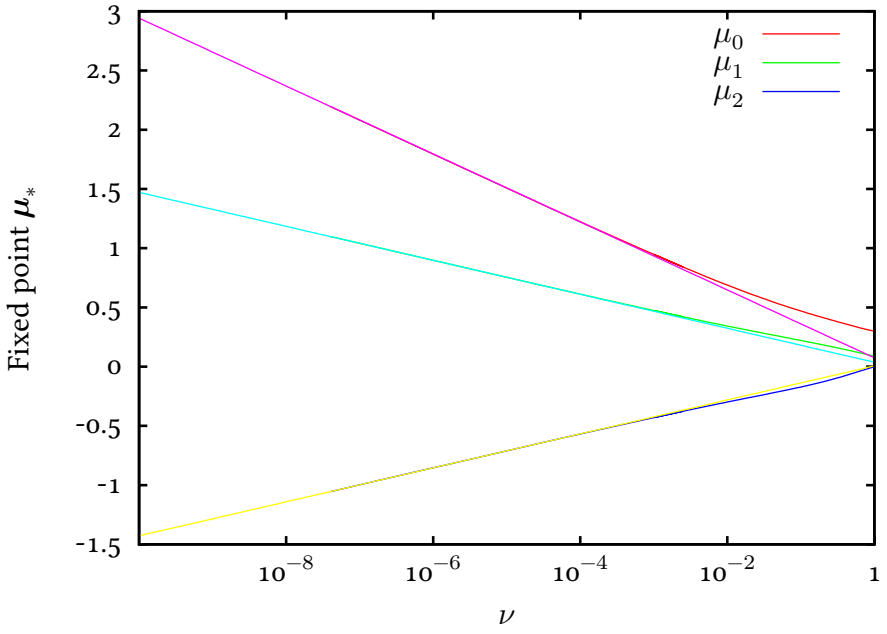
$$\mathbb{P}_{X_\phi}^{\chi_u} f(\boldsymbol{x}_{V_1}) = \arg\min_{g \in X_\phi} \int \Big(f(\boldsymbol{x}_{V_1 \setminus u}, \chi_u) - g(\boldsymbol{x}_{V_1 \setminus u}, \chi_u) \Big)^2 \times \Big(P(\boldsymbol{x}_{V_1 \setminus u}, \chi_u) + P(\boldsymbol{x}_{V_1 \setminus u}, -\chi_u) \Big)\, d\boldsymbol{x}_{V_1 \setminus u},$$

for $u \in V_1$, which we previously used in the fully-symmetrized generalized fast marginalization. We define the projection operator $\mathbb{P}_{X_\phi}$ as a sum

$$\mathbb{P}_{X_\phi} f(\boldsymbol{x}_{V_1}) = \int \mathbb{P}_{X_\phi}^{\chi_u} f(\boldsymbol{x}_{V_1}) d\chi_u,$$



(a) Linear scale



(b) Logarithmic scale

Figure 7.36: Dependence of the critical point location $\mu_*(\nu)$ on the coarsening rule. The critical couplings $\mu_*(\nu)$ diverge logarithmically as $\nu \to 0$, a fact made clear by the logarithmic fits on Figure (b).

where the integral may be approximated using a quadrature rule.

Lattices are coarsened by dividing the nodes into subsets of size 2×2, thus we reproduce the prior structure through renormalization of a $n \times n$ lattice to a ${}^{n}/_{2} \times {}^{n}/_{2}$ lattice. The variables $\boldsymbol{x}_{V_0}$ are sampled using a straightforward Gibbs sampler, while the coarser lattice is sampled using the conditional probability $P(\boldsymbol{x}_{V_1} \mid \boldsymbol{x}_{V_0})$ defining the particular coarsening rule being used.

We compute the matrices $A(t_j)$ and right hand side vectors $\boldsymbol{b}(t_j)$ at a set of seven Gaussian quadrature nodes t_j using multiple Markov chains running in parallel, averaging over the chains. We take advantage of the translation invariance by performing an averaging over all spins on a given lattice. LAPACK routines are used to solve the symmetric positive definite systems

$$A(t_j)\boldsymbol{\mu}'(t_j) = \boldsymbol{b}(t_j)$$

at each Gaussian quadrature node. The final coupling parameters $\boldsymbol{\mu}$ are then recovered through integration as

$$\boldsymbol{\mu}' = \int_{-1}^{1} \boldsymbol{\mu}'(\chi_u) d\chi_u = \sum_j \boldsymbol{\mu}'(t_j) w_j.$$

The coefficients $\boldsymbol{\mu}(\chi_u)$ are also integrated between the symmetrically placed quadrature nodes $-t_j$ and t_j, producing a set of coefficients

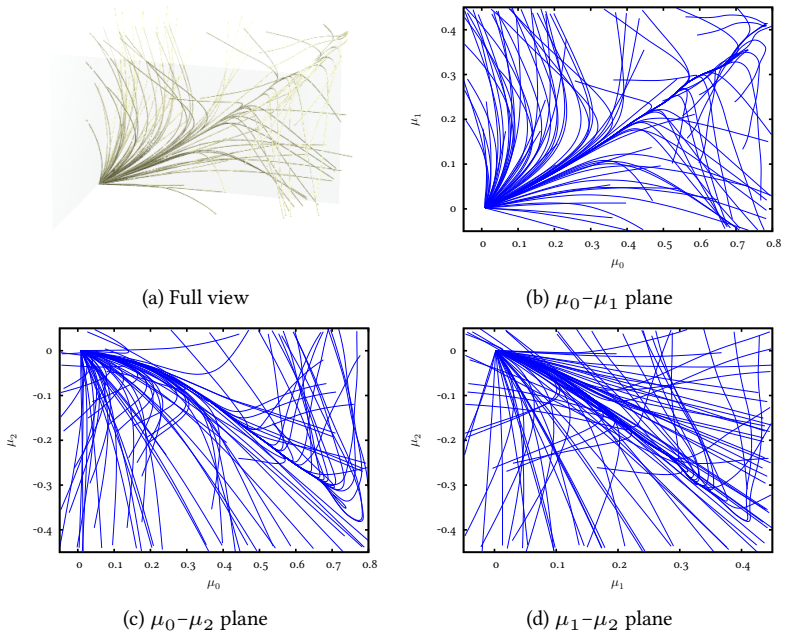
$$\boldsymbol{\mu}'_j = \int_{-t_j}^{t_j} \boldsymbol{\mu}'(\chi_u) d\chi_u,$$

which are used to compute the correction terms

$$\exp\left(\int_{-\chi_u}^{\chi_u} W'(\boldsymbol{x}_{V_i \setminus u}, s) ds\right) \quad \text{and} \quad \exp\left(-\int_{-\chi_u}^{\chi_u} W'(\boldsymbol{x}_{V_i \setminus u}, s) ds\right)$$

required by the full symmetrization scheme.

Because the fully-symmetrized generalized fast marginalization equation satisfied by the parameters $\boldsymbol{\mu}'$ is implicit, we solve it using fixed-point iteration with Robbins-Monro smoothing, repeating the algorithm for 50 iterations, with each iteration consisting of 10,000 random samples. Note that the fixed-point iteration is not related to the fixed points of the mapping $\widehat{R}$.



(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

Figure 7.37: Approximate computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 0$ coarsening rule (decimation).

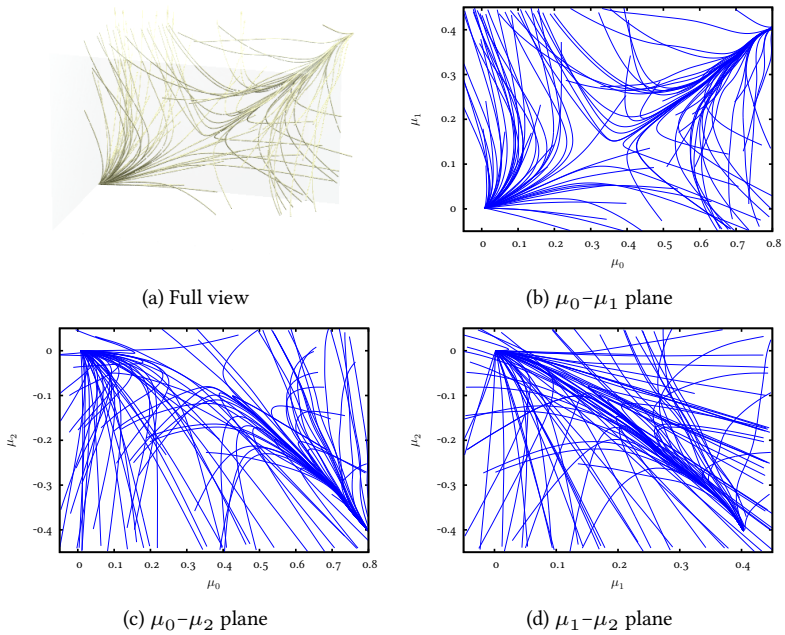
The compute code computes the renormalized coefficients $\boldsymbol{\mu}'$ for a set of original coupling coefficients $\boldsymbol{\mu}$, again sampling a rectangular cuboid defined by

$$[\mu_1^{\min}, \mu_1^{\max}] \otimes [\mu_2^{\min}, \mu_2^{\max}] \otimes [\mu_3^{\min}, \mu_3^{\max}] \\ = [-0.05, 0.8] \otimes [-0.05, 0.45] \otimes [-0.45, 0.05].$$

Due to the much greater computational resources needed to compute the approximate coefficients, we only subdivide each dimension into 6 intervals using $N = 7$ points. Thus, $N^3 = 343$ sets of initial coefficients are considered, producing a much lower resolution that the exact computation of previous section.

We first attempt to use a small lattice with $n = 4$, reproducing the previous computation. As we can see, Figures 7.37, 7.38 and 7.39 are very similar to Figures 7.33, 7.34 and 7.35, respectively, confirming that the approximate method works correctly: in the case of a 4×4 fine lattice, the basis ϕ is exact on the 2×2 coarse lattice, thus the two methods must agree up to the stochastic errors in the computation of expected values.

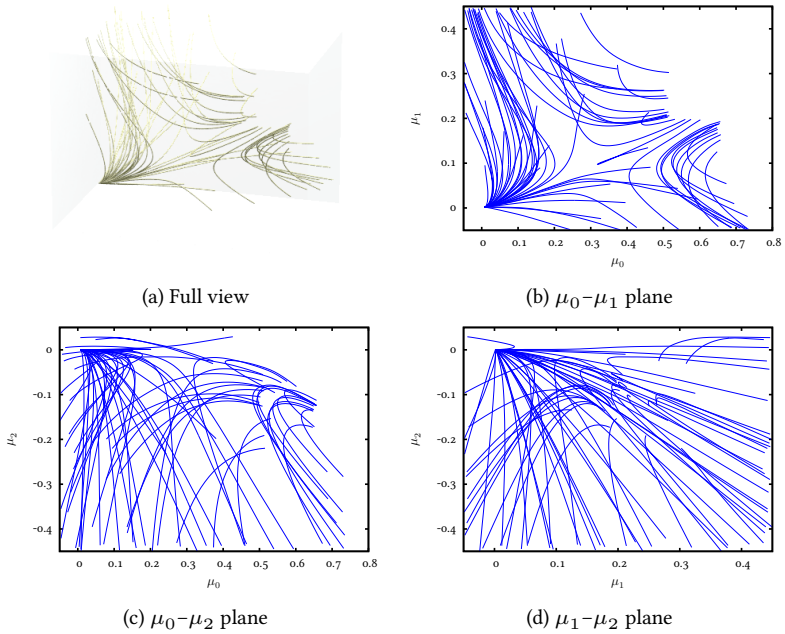
The only major difference we see occurs under majority rule on Figure 7.39, where a part of the visualization is missing. This is due to the fact that the projection method is limited to relatively weak couplings. Intuitively,



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

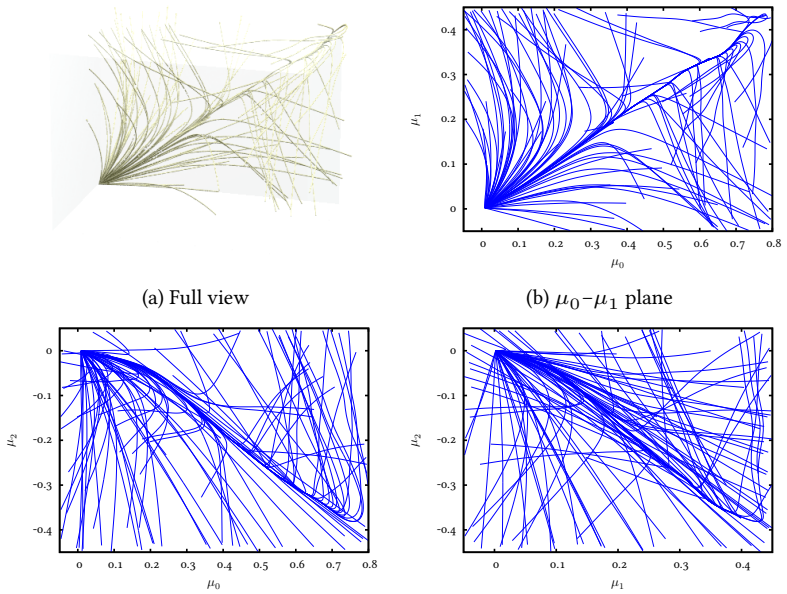
Figure 7.38: Approximate computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 1/2$ coarsening rule.



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

Figure 7.39: Approximate computation of the parameter flow for a 4×4 lattice V_0 and under $\nu = 1$ coarsening rule (majority rule).



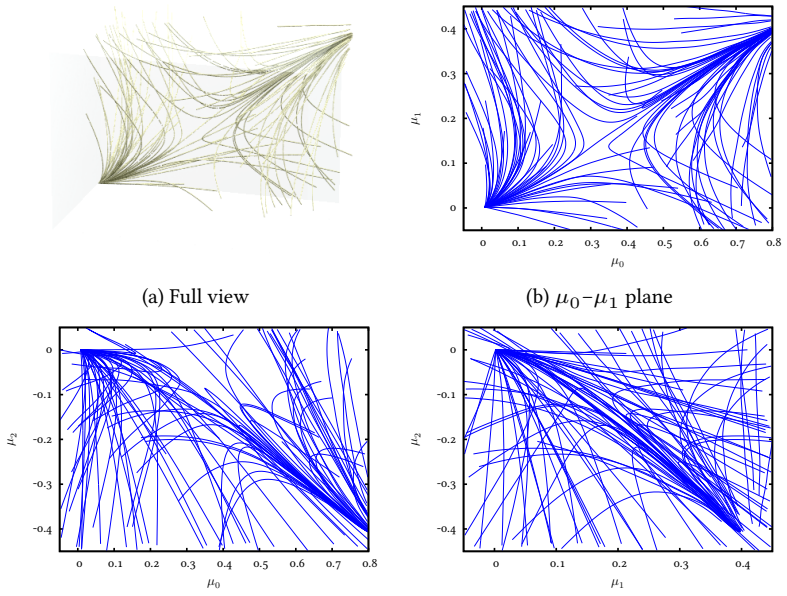
(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

Figure 7.40: Approximate computation of the parameter flow for a 8×8 lattice V_0 and under $\nu = 0$ coarsening rule (decimation).



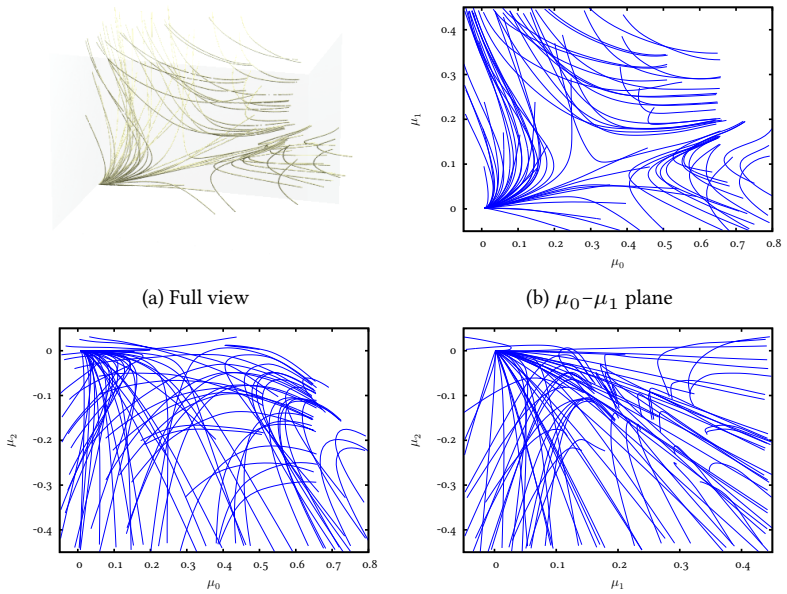
(a) Full view

(b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane

(d) μ_1–μ_2 plane

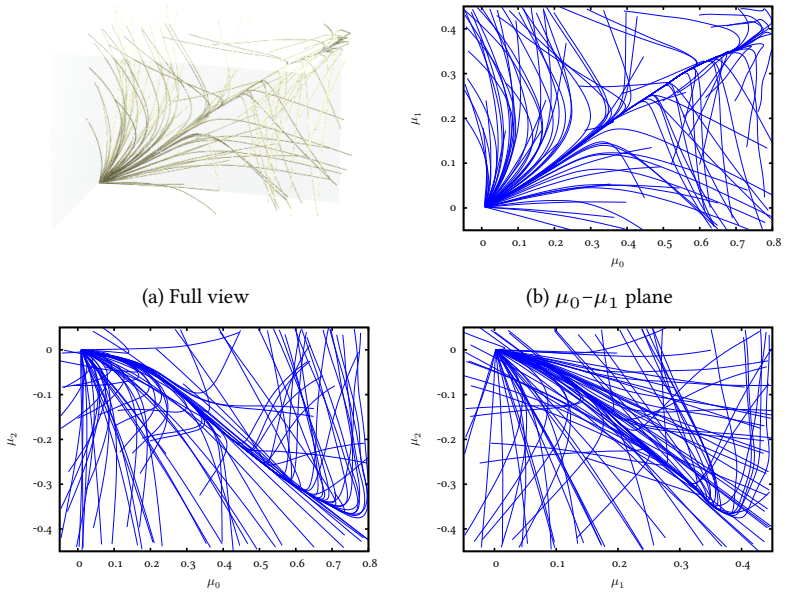
Figure 7.41: Approximate computation of the parameter flow for a 8×8 lattice V_0 and under $\nu = 1/2$ coarsening rule.



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

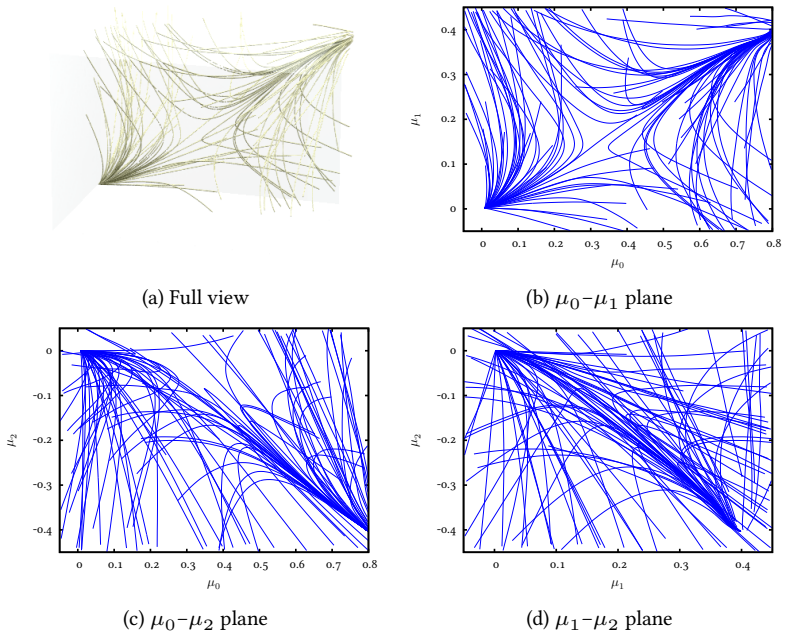
Figure 7.42: Approximate computation of the parameter flow for a 8×8 lattice V_0 and under $\nu = 1$ coarsening rule (majority rule).



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

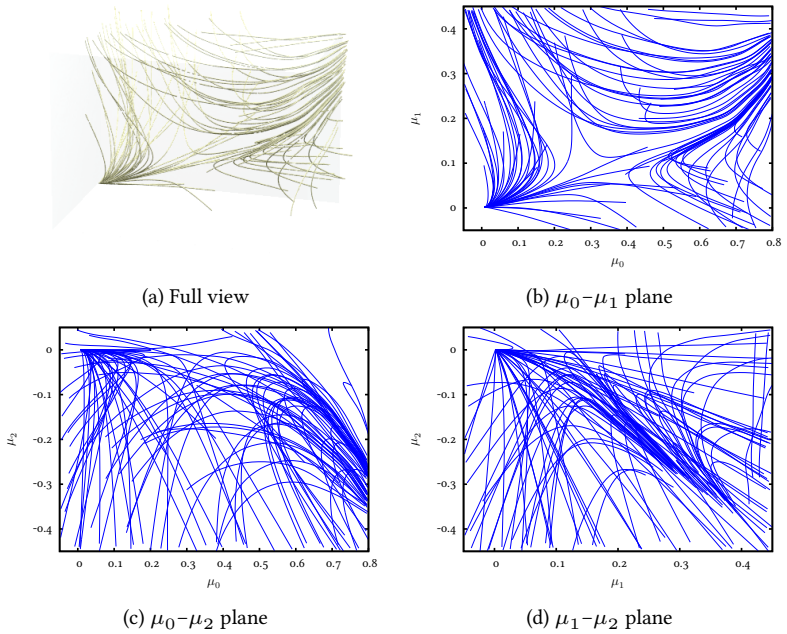
Figure 7.43: Approximate computation of the parameter flow for a 16×16 lattice V_0 and under $\nu = 0$ coarsening rule (decimation).



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

Figure 7.44: Approximate computation of the parameter flow for a 16×16 lattice V_0 and under $\nu = 1/2$ coarsening rule.



(a) Full view (b) μ_0–μ_1 plane

(c) μ_0–μ_2 plane (d) μ_1–μ_2 plane

Figure 7.45: Approximate computation of the parameter flow for a 16×16 lattice V_0 and under $\nu = 1$ coarsening rule (majority rule).

when the couplings are strong the spins are correlated and basis functions always take the same values. Therefore, the Gram matrices $A(\chi_u)$ become singular for large $|\boldsymbol{\mu}|$, limiting the usability of projection to the relatively small values of $|\boldsymbol{\mu}|$.

Moving to larger lattice sizes, we produce a set of visualizations corresponding to the results obtained using exact coarsening. Qualitatively, the behavior of the parameter flow on larger lattices does not change, as shown by the visualizations obtained using 8×8 and 16×16 lattices.

The parameter flow under decimation, shown on Figures 7.40 and 7.43, does not have a critical point. Instead, the coefficients $\boldsymbol{\mu}$ appear to collapse onto a single curve leading to the sole fixed point at $\boldsymbol{\mu} = 0$. On the other hand, the coarsening rules with $\nu = 1/4$ and 1 (majority rule) both produce a critical point, as was the case on the 4×4 lattice: see Figures 7.41, 7.42, 7.44 and 7.45.

8

ISING MODEL

The Ising model is a classic model of statistical physics due to Wilhelm Lenz (1920), but named after his student Ernst Ising (1925), who solved the one-dimensional model in his doctoral thesis. The model describes simplified interactions between grains of a ferromagnetic material. In the Ising model those grains are represented by the so-called Ising spins: discrete variables allowed to take values of either 1 or -1 — representing spin up or down — arranged in a regular Cartesian lattice with periodic boundary conditions. These variables interact with each other through nearest-neighbor interactions, where each pair of neighboring variables contributes either $-J$ or J to the potential energy of the system $H(\boldsymbol{x}_V)$, depending on whether the two variables are the same or different, respectively. Therefore, the potential energy $H(\boldsymbol{x}_V)$ may be written as

$$H(\boldsymbol{x}_V) = -\frac{J}{2} \sum_V x_u \sum_{N(u)} x_v,$$

where the first sum is over all nodes u on the lattice, while the latter over the neighbors v of the node u. The resulting probability distribution over $\boldsymbol{x}_V$ is a Gibbs measure

$$P(\boldsymbol{x}_V) = \frac{1}{Z} \exp\left(-\frac{H(\boldsymbol{x}_V)}{T}\right) = \frac{1}{Z} \exp\left(\frac{J}{2T} \sum_V x_u \sum_{N(u)} x_v\right),$$

where one typically specifies the system using the coupling parameter $\mu = J/T$, also called the inverse temperature when $J = 1$.

The main feature of the Ising model is the phase transition that occurs in the two- and three-dimensional Ising model at a finite coupling μ_c, whose value depends on the dimension of the model. The two-dimensional Ising model has been solved exactly by Onsager (1944), thus the exact location of the phase transition is known to be $\mu_c = \ln(1+\sqrt{2})/2 \approx 0.44068679$. For $\mu < \mu_c$ the coupling is weak and the spins are uncorrelated, yielding an unmagnetized state: one, where the average magnetization

$$\mathcal{M}(\mu) = \mathbb{E}_\mu \left[\frac{1}{n^2} \sum_u x_u\right]$$

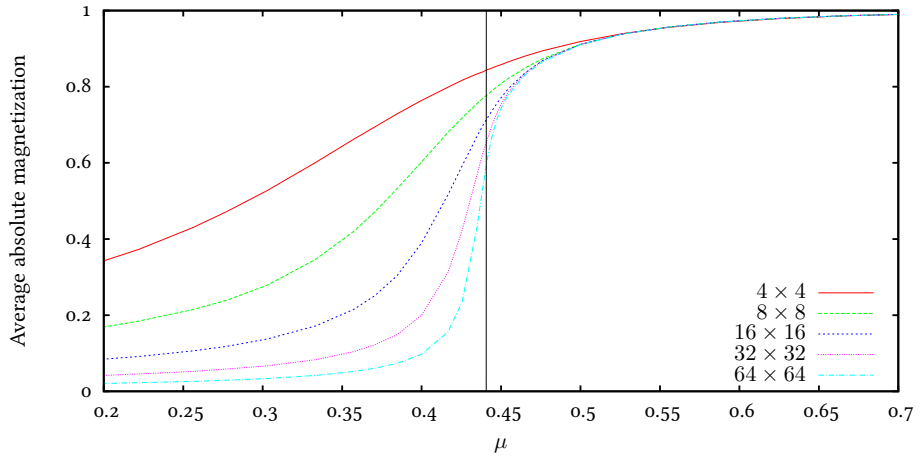
is zero. However, for $\mu > \mu_c$ the magnetization $\mathcal{M}(\mu)$ rises sharply, because the spins break the symmetry and undergo spontaneous magnetization. The actual phase transition occurs only in the case of an infinite lattice, but the behavior of the finite Ising model converges quickly to that of the infinite case. As shown on Figure 8.46a rapid change in magnetization $\mathcal{M}(\mu)$ can be observed around $\mu \approx \mu_c$, with the change becoming more abrupt as the lattice size increases. However, the exact position of the phase transition is usually determined using the Binder cumulant $U_4(\mu)$,

$$U_4(\mu) = 1 - \frac{\mathbb{E}_\mu\left[\frac{1}{n^2}\sum_u x_u^4\right]}{3\mathbb{E}_\mu\left[\frac{1}{n^2}\sum_u x_u^2\right]^2},$$

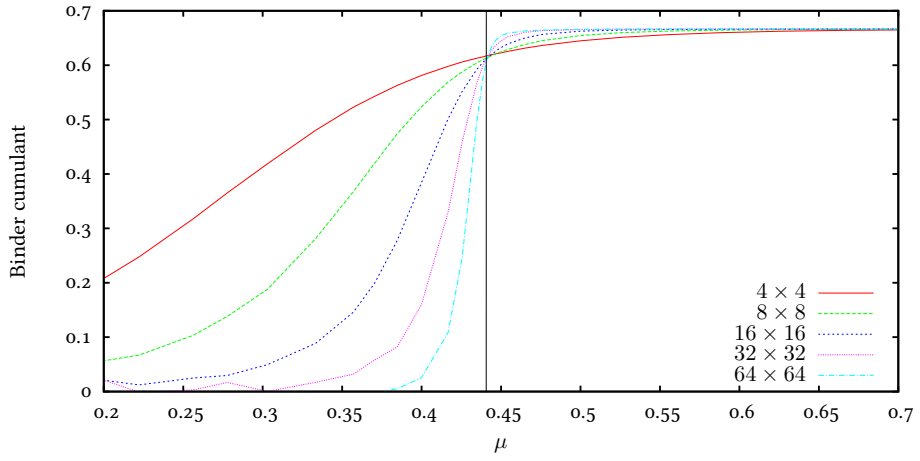
which has the property that at μ_c the value $U_4(\mu_c)$ is independent of the lattice size n. As a result, the precise location of the phase transition may be determined from the intersection of the Binder cumulant curves for lattices of different size, shown on Figure 8.46b.

The Ising model has long been studied theoretically and computationally, making it *de facto* the standard model for studying properties of numerical methods. The available literature on numerical renormalization and its use in sampling deals virtually exclusively with the Ising model at critical temperature; therefore, the examples shown within this chapter will always show the square-lattice Ising model at critical temperature. Although our methodology remains completely general, in the remainder of this chapter we restrict ourselves to the two-dimensional Ising model defined over a regular, square Cartesian lattice. We further assume that the lattice is of size $n \times n$, where n is a power of 2, and doubly-periodic boundary conditions are used.

The present chapter is divided into two parts. We begin by applying the fast marginalization method to the Ising model to obtain the renormalized coupling coefficients $\boldsymbol{\mu}_i$ on coarse lattices V_i, $i > 0$. These renormalized coupling coefficients will later be used to construct a proposal density for sampling the Ising model, however we begin the discussion of the numerical methods developed in this thesis by comparing our renormalized coupling coefficients to the values reported in the literature. Subsequently, we use the numerically computed coupling coefficients to construct random samples from the Ising model using the importance sampling frame-



(a) Absolute average magnetization.



(b) Binder cumulant

Figure 8.46: Dependency of the (a) absolute average magnetization $\mathcal{M}_{\text{abs}}(\mu) = \mathbb{E}_\mu\left[\frac{1}{n}\left|\sum_u x_u\right|\right]$, a slight modification of the average magnetization $\mathcal{M}(\mu)$ defined above, and (b) Binder cumulant $U_4(\mu)$ of the two-dimensional Ising model on the coupling parameter μ. As the coupling increases and reaches the critical coupling $\mu_c = \ln(1+\sqrt{2})/2 \approx 0.44068679$ (solid black line), the magnetization begins to grow rapidly and plateaus for μ above μ_c. The larger the lattice the more abrupt the change, eventually converging to a first order phase transition. The Binder cumulant also abruptly changes value in the vicinity of the phase transition, but the precise location of the transition is indicated by the intersection of the curves corresponding to different lattice sizes.

work and compare the results obtained using the Sequential Importance Sampling (SIS) and Partial Rejection Control (PRC) methods.

8.1 RENORMALIZED COEFFICIENTS

The original lattice $V = V_0$ of size $n \times n$ may be coarsened by dividing the lattice into subsets of 2×2 nodes and retaining only one node out of each. For the purposes of this section, we keep nodes at positions (i, j) such that both i and j are even. The resulting coarsened lattice V_1 of size ${}^{n}/_{2} \times {}^{n}/_{2}$ together with the marginal probability density

$$P(\boldsymbol{x}_{V_1}) = \int P(\boldsymbol{x}_{V_0}) d\boldsymbol{x}_{V_0 \setminus V_1}$$

have all the symmetry properties of the original Ising model. Repeating this coarsening procedure produces a sequence of lattices $V_1, V_2, \dots, V_m$ for which we will compute the renormalized coupling coefficients.

We will approximate the marginal probability density using a number of interactions ϕ_k, writing

$$\frac{\dfrac{\partial P(\boldsymbol{x}_{V_i})}{\partial \chi_u}}{P(\boldsymbol{x}_{V_m})} = \frac{\partial W(\boldsymbol{x}_{V_i})}{\partial \chi_u} = \sum_{k=1}^{K} c(\chi_u) \phi_k(\boldsymbol{x}_{N(u)}),$$

where $N(u)$ is the set of neighbors of the node $u \in V_m$. Following the literature, we choose ϕ_k to be polynomial functions in the neighbors of the node u. The basis functions ϕ_k are defined on Figure 8.47.

The initial five functions correspond to interactions between pairs of variables, with the interactions that are equivalent under lattice symmetries reduced due to form a single function. The functions are sorted by the distance between the variables, thus ϕ_1 is the interaction between variables at distance 1, ϕ_2 at distance $\sqrt{2}$, ϕ_3 at distance 2, ϕ_4 at distance $\sqrt{5}$ and ϕ_5 at distance $\sqrt{8}$. The remaining three terms are the four-variable interactions. ϕ_6 has never been included in the literature, though it is the interaction between closest neighboring variables arranged in an isosceles right-angle triangle with hypothenuse of length 2. The final two functions are called plaquettes, because the variables participating in the interaction are arranged into a square tile: ϕ_7 forms tiles of side length 1 while ϕ_8 of side length $\sqrt{2}$. This choice of ϕ_k matches closely that of Swendsen (1984b), allowing for a direct comparison of results; however, Swendsen (1984b) neglected the term ϕ_6, while our computations show that this terms is indeed significant.

We computed the coefficients using the fully-symmetrized generalized fast marginalization method with $Q(\boldsymbol{x}) = 1$, as described in Chapter 6. We ran a Markov Chain Monte Carlo (MCMC) computation with a Gibbs

Figure 8.47: The interactions ϕ_k used in the computation of the renormalized coupling coefficients. The first five functions are linear terms corresponding to interactions between pairs of variables, while the latter three functions are cubic and correspond to four-variable interactions.

$$
\begin{aligned}
\phi_1 &= x_{i,j-1} + x_{i,j+1} + x_{i-1,j} + x_{i+1,j} \\
\phi_2 &= x_{i-1,j-1} + x_{i+1,j+1} + x_{i+1,j-1} + x_{i-1,j+1} \\
\phi_3 &= x_{i,j-2} + x_{i,j+2} + x_{i-2,j} + x_{i+2,j} \\
\phi_4 &= x_{i-1,j-2} + x_{i+1,j+2} + x_{i+1,j-2} + x_{i-1,j+2} \\
&\quad + x_{i-2,j-1} + x_{i+2,j+1} + x_{i+2,j-1} + x_{i-2,j+1} \\
\phi_5 &= x_{i-2,j-2} + x_{i+2,j+2} + x_{i+2,j-2} + x_{i-2,j+2} \\
\phi_6 &= x_{i,j-1}x_{i-1,j}x_{i+1,j} + x_{i,j+1}x_{i-1,j+1}x_{i+1,j+1} \\
&\quad + x_{i+1,j}x_{i+1,j-1}x_{i+2,j} + x_{i-1,j}x_{i-1,j-1}x_{i-2,j} \\
&\quad + x_{i,j-1}x_{i-1,j}x_{i,j+1} + x_{i,j+1}x_{i-1,j+1}x_{i,j+2} \\
&\quad + x_{i+1,j}x_{i+1,j-1}x_{i+1,j+1} + x_{i,j-1}x_{i-1,j-1}x_{i,j-2} \\
&\quad + x_{i,j-1}x_{i+1,j}x_{i,j+1} + x_{i,j+1}x_{i+1,j+1}x_{i,j+2} \\
&\quad + x_{i-1,j}x_{i-1,j-1}x_{i-1,j+1} + x_{i,j-1}x_{i+1,j-1}x_{i,j-2} \\
&\quad + x_{i,j-1}x_{i-1,j-1}x_{i+1,j-1} + x_{i-1,j}x_{i+1,j}x_{i,j+1} \\
&\quad + x_{i+1,j}x_{i+1,j+1}x_{i+2,j} + x_{i-1,j}x_{i-1,j+1}x_{i-2,j} \\
\phi_7 &= x_{i,j-1}x_{i-1,j}x_{i-1,j-1} + x_{i-1,j}x_{i,j+1}x_{i-1,j+1} \\
&\quad + x_{i,j-1}x_{i+1,j}x_{i+1,j-1} + x_{i+1,j}x_{i,j+1}x_{i+1,j+1} \\
\phi_8 &= x_{i-1,j-1}x_{i+1,j-1}x_{i,j-2} + x_{i+1,j-1}x_{i+1,j+1}x_{i+2,j} \\
&\quad + x_{i-1,j-1}x_{i-1,j+1}x_{i-2,j} + x_{i-1,j+1}x_{i+1,j+1}x_{i,j+2}
\end{aligned}
$$

---

sampler that outputted a new sample every $10n^2$ individual variable flips. We collected the projection matrix and vector at seven Gaussian quadrature nodes using data from 1,000,000 samples. Additionally, we averaged the projection matrices over all variables on the given lattice. We computed the final coefficients iteratively using the fixed point algorithm, continuing for eight iterations. In order to smooth the convergence and make use of multiple iterations, we applied the Robbins-Monro algorithm with

Table 8.12: Values of the renormalized coefficients obtained by renormalizing under decimation i times a 16×16 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.346401	0.288902	0.279622	0.276477	0.275903
	2	———	0.094588	0.094831	0.086260	0.085956
	3	———	———	0.029574	0.021264	0.020447
	4	———	———	———	0.009208	0.008111
	5	———	———	———	———	0.003502
	6	———	———	-0.010100	-0.009640	-0.009701
	7	———	-0.033852	-0.016972	-0.016388	-0.015977
	8	———	———	-0.003974	-0.001904	-0.001690
2	1	0.291618	0.230339	0.215358	0.206900	0.205304
	2	———	0.119443	0.121750	0.107456	0.105085
	3	———	———	0.041268	0.034173	0.034290
	4	———	———	———	0.013505	0.012678
	5	———	———	———	———	0.003980
	6	———	———	-0.016778	-0.015379	-0.015410
	7	———	-0.057560	-0.020709	-0.020471	-0.019059
	8	———	———	-0.012023	-0.009063	-0.008864

$a_i = 1$ for $i < 3$ and $a_i = 1/(i-2)$ for the following iterations (Robbins and Monro, 1951). The resulting decimation coefficients are presented in Tables 8.12, 8.13 and 8.14, while the majority rule coefficients are collected in Tables 8.16, 8.17, 8.18 and 8.19.

8.1.1 *Decimation coefficients*

The renormalized coefficients obtained using decimation exhibit three main features. It appears that they depend not on the absolute size of the lattice, but on the size relative to that of the original lattice. For example, the coupling coefficients are nearly identical for a given i independently of the size of the initial lattice: the calculations performed with the 16×16, 32×32 and 64×64 lattices show very similar results. The coefficients decay slowly with distance; in fact, the higher the value of i the slower the decay, thus the approximation of $W(\boldsymbol{x}_{V_i})$ becomes increasingly difficult as i grows larger.

Table 8.13: Values of the renormalized coefficients obtained by renormalizing under decimation i times a 32×32 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.344981	0.288067	0.280326	0.276859	0.276752
	2	———	0.093201	0.093756	0.085901	0.085580
	3	———	———	0.029034	0.020627	0.020019
	4	———	———	———	0.009286	0.008123
	5	———	———	———	———	0.003456
	6	———	———	-0.010067	-0.009697	-0.009629
	7	———	-0.031002	-0.017165	-0.016364	-0.016419
	8	———	———	-0.002861	-0.001185	-0.001190
2	1	0.288819	0.228331	0.212335	0.203475	0.202416
	2	———	0.116334	0.113115	0.098318	0.096379
	3	———	———	0.048998	0.035229	0.032997
	4	———	———	———	0.017582	0.014884
	5	———	———	———	———	0.009857
	6	———	———	-0.015527	-0.014409	-0.014033
	7	———	-0.050593	-0.021712	-0.020305	-0.020196
	8	———	———	-0.008226	-0.004709	-0.004165
3	1	0.252836	0.194676	0.179348	0.165180	0.163611
	2	———	0.119176	0.119295	0.103769	0.102385
	3	———	———	0.047423	0.038987	0.038312
	4	———	———	———	0.017965	0.016255
	5	———	———	———	———	0.006571
	6	———	———	-0.018624	-0.016662	-0.016473
	7	———	-0.057689	-0.019758	-0.016735	-0.017108
	8	———	———	-0.015346	-0.013995	-0.013548

Finally, the coefficients show the the functions ϕ_k have a large overlap, that is, the addition of a basis function changes the coefficients of the existing functions. For example, the introduction of ϕ_2 and ϕ_7 to the basis reduces the coefficient of ϕ_1 from approximately 0.344 to 0.289 (Table 8.13). As a result, the coefficients depend very strongly on the included basis functions.

Table 8.14: Values of the renormalized coefficients obtained by renormalizing under decimation i times a 64×64 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.343977	0.287808	0.280288	0.277232	0.277182
	2	———	0.092388	0.093850	0.085924	0.085389
	3	———	———	0.028849	0.020365	0.019747
	4	———	———	———	0.009194	0.008152
	5	———	———	———	———	0.003306
	6	———	———	-0.010055	-0.009623	-0.009620
	7	———	-0.029689	-0.017062	-0.016510	-0.016395
	8	———	———	-0.002616	-0.001091	-0.000831
2	1	0.286261	0.226996	0.213783	0.205286	0.204718
	2	———	0.113933	0.112134	0.097888	0.096586
	3	———	———	0.047901	0.033834	0.032365
	4	———	———	———	0.017436	0.014895
	5	———	———	———	———	0.008640
	6	———	———	-0.015477	-0.014263	-0.014054
	7	———	-0.046073	-0.022136	-0.020860	-0.021064
	8	———	———	-0.006497	-0.003525	-0.003162
3	1	0.247946	0.191938	0.175508	0.160796	0.158770
	2	———	0.115576	0.110295	0.091722	0.088631
	3	———	———	0.057092	0.040212	0.037747
	4	———	———	———	0.024137	0.020366
	5	———	———	———	———	0.014774
	6	———	———	-0.017170	-0.015058	-0.014644
	7	———	-0.049786	-0.020119	-0.018067	-0.017405
	8	———	———	-0.010253	-0.006706	-0.005995
4	1	0.219973	0.166162	0.154218	0.138876	0.138025
	2	———	0.112112	0.111370	0.096414	0.093371
	3	———	———	0.045458	0.037541	0.038424
	4	———	———	———	0.019895	0.018156
	5	———	———	———	———	0.006643
	6	———	———	-0.018248	-0.015924	-0.016262
	7	———	-0.048858	-0.016361	-0.015139	-0.014694
	8	———	———	-0.013955	-0.013071	-0.012467

Table 8.15: Comparison of the values of the renormalized coefficients obtained by renormalizing under decimation i times a 32×32 Ising lattice at $T = 2.269185$ with those reported in the literature.

i	k	Results	Swendsen (1984b)
1	1	0.276752	0.254
	2	0.085580	0.086
	3	0.020019	0.015
	4	0.008123	0.008
	5	0.003456	0.004
	6	-0.009629	———
	7	-0.016419	-0.018
	8	-0.001190	-0.009
2	1	0.202416	0.186
	2	0.096379	0.089
	3	0.032997	0.028
	4	0.014884	0.016
	5	0.009857	0.012
	6	-0.014033	———
	7	-0.020196	-0.031
	8	-0.004165	-0.013
3	1	0.163611	0.146
	2	0.102385	0.097
	3	0.038312	0.033
	4	0.016255	0.019
	5	0.006571	0.010
	6	-0.016473	———
	7	-0.017108	-0.043
	8	-0.013548	-0.028

Our decimation coefficients are compared with available literature in Figure 8.15. The only other work that reported these coefficients, to the best of our knowledge, is Swendsen (1984b). The coefficients obtained in the present thesis agree about the order of magnitude with those of Swendsen (1984b), however there is no numerical agreement regarding the particular values. However, the related majority rule coefficients of

Table 8.16: Values of the renormalized coefficients obtained by renormalizing under majority rule i times a 16×16 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.426782	0.354804	0.356047	0.356727	0.356738
	2	———	0.072394	0.073559	0.077015	0.077122
	3	———	———	-0.017775	-0.014065	-0.013779
	4	———	———	———	-0.003554	-0.003064
	5	———	———	———	———	-0.001327
	6	———	———	0.010337	0.010236	0.010228
	7	———	0.005076	-0.011266	-0.011373	-0.011377
	8	———	———	-0.008944	-0.009725	-0.009787
2	1	0.424890	0.340252	0.345553	0.347499	0.347679
	2	———	0.081485	0.083042	0.088580	0.089148
	3	———	———	-0.019416	-0.016448	-0.016454
	4	———	———	———	-0.004458	-0.003831
	5	———	———	———	———	-0.001822
	6	———	———	0.009943	0.009826	0.009795
	7	———	0.007344	-0.013781	-0.014185	-0.014171
	8	———	———	-0.005204	-0.006162	-0.006292

Swendsen (1984b) do not agree well with those of other authors and thus we consider these results to be less accurate than ours.

8.1.2 *Majority rule coefficients*

The majority rule is not used elsewhere within this thesis, because it does not allow one to construct a sequential importance sampling algorithm. This is due to the fact that it is not possible to evaluate the conditional probability $P(\boldsymbol{x}_{V_{i-1} \setminus V_i} \mid \boldsymbol{x}_{V_i})$ and thus compute the proposal density. However, the generalized fast marginalization algorithm is a generalization of the fast marginalization algorithm and makes it possible to compute the majority rule coefficients on coarse lattices using the same machinery. We do so because the majority rule is the preferred coarsening rule used in the physics community and has been studied much more thoroughly; as a result, a larger set of published coefficients exists than it is the case with decimation.

Table 8.17: Values of the renormalized coefficients obtained by renormalizing under majority rule i times a 32×32 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.425698	0.355706	0.355919	0.356533	0.356544
	2	———	0.073928	0.073784	0.077057	0.077240
	3	———	———	-0.017491	-0.013736	-0.013570
	4	———	———	———	-0.003615	-0.003076
	5	———	———	———	———	-0.001424
	6	———	———	0.010470	0.010379	0.010390
	7	———	0.002901	-0.011286	-0.011384	-0.011427
	8	———	———	-0.009519	-0.010078	-0.010148
2	1	0.423128	0.341905	0.344757	0.345758	0.345681
	2	———	0.084371	0.086143	0.090010	0.090329
	3	———	———	-0.020645	-0.016450	-0.016188
	4	———	———	———	-0.004079	-0.003657
	5	———	———	———	———	-0.001225
	6	———	———	0.010349	0.010224	0.010206
	7	———	0.003576	-0.013364	-0.013654	-0.013484
	8	———	———	-0.007447	-0.008262	-0.008440
3	1	0.424534	0.338215	0.344130	0.345500	0.345963
	2	———	0.083864	0.084834	0.090862	0.091553
	3	———	———	-0.020585	-0.017721	-0.017352
	4	———	———	———	-0.004581	-0.003961
	5	———	———	———	———	-0.001468
	6	———	———	0.010252	0.010193	0.009931
	7	———	0.007091	-0.014292	-0.014254	-0.014654
	8	———	———	-0.004931	-0.006153	-0.006170

The parameter flow of majority rule exhibits a unique critical point, while that of decimation does not (see Chapter 7 and Swendsen, 1984b); thus the renormalized coefficients obtained using the Ising model at $\mu = \mu_c$ will tend toward the critical point. Because the critical point can be described fairly well using only a few coefficients, the renormalized majority rule coefficients corresponding to long-range interactions do not increase significantly with i as it was the case with decimation. In other

Table 8.18: Values of the renormalized coefficients obtained by renormalizing under majority rule i times a 64×64 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.425072	0.356183	0.355889	0.356539	0.356582
	2	———	0.074743	0.073844	0.077079	0.077240
	3	———	———	-0.017377	-0.013599	-0.013475
	4	———	———	———	-0.003689	-0.003085
	5	———	———	———	———	-0.001503
	6	———	———	0.010534	0.010443	0.010440
	7	———	0.001773	-0.011325	-0.011423	-0.011481
	8	———	———	-0.009686	-0.010179	-0.010265
2	1	0.421838	0.343139	0.344742	0.345637	0.345632
	2	———	0.085869	0.086379	0.090110	0.090398
	3	———	———	-0.020371	-0.015994	-0.015913
	4	———	———	———	-0.004195	-0.003776
	5	———	———	———	———	-0.001228
	6	———	———	0.010423	0.010305	0.010285
	7	———	0.001180	-0.013391	-0.013632	-0.013551
	8	———	———	-0.007968	-0.008638	-0.008712
3	1	0.422866	0.340279	0.343127	0.344367	0.344322
	2	———	0.086489	0.088301	0.092366	0.092674
	3	———	———	-0.021662	-0.017274	-0.017217
	4	———	———	———	-0.004362	-0.003802
	5	———	———	———	———	-0.001386
	6	———	———	0.010509	0.010344	0.010379
	7	———	0.003155	-0.013757	-0.014040	-0.014079
	8	———	———	-0.007237	-0.007883	-0.008145
4	1	0.424249	0.337158	0.344022	0.344948	0.345885
	2	———	0.083406	0.083958	0.091255	0.092082
	3	———	———	-0.021205	-0.017573	-0.017512
	4	———	———	———	-0.004910	-0.004577
	5	———	———	———	———	-0.001947
	6	———	———	0.010211	0.010229	0.010268
	7	———	0.008248	-0.013713	-0.014741	-0.015177
	8	———	———	-0.003730	-0.005589	-0.005825

Table 8.19: Values of the renormalized coefficients obtained by renormalizing under majority rule i times a 128×128 Ising lattice at $T = 2.269185$.

i	k	$K = 1$	$K = 3$	$K = 6$	$K = 7$	$K = 8$
1	1	0.424708	0.356423	0.355920	0.356561	0.356566
	2	———	0.075169	0.073876	0.077084	0.077229
	3	———	———	-0.017316	-0.013532	-0.013365
	4	———	———	———	-0.003700	-0.003093
	5	———	———	———	———	-0.001570
	6	———	———	0.010554	0.010474	0.010463
	7	———	0.001168	-0.011370	-0.011468	-0.011476
	8	———	———	-0.009790	-0.010276	-0.010303
2	1	0.421002	0.343751	0.344703	0.345572	0.345585
	2	———	0.086742	0.086516	0.090238	0.090391
	3	———	———	-0.020168	-0.015822	-0.015655
	4	———	———	———	-0.004279	-0.003702
	5	———	———	———	———	-0.001490
	6	———	———	0.010466	0.010359	0.010353
	7	———	-0.000187	-0.013447	-0.013561	-0.013605
	8	———	———	-0.008274	-0.008863	-0.008884
3	1	0.421189	0.341537	0.343116	0.344262	0.344234
	2	———	0.088135	0.088629	0.092622	0.092651
	3	———	———	-0.021442	-0.016884	-0.016677
	4	———	———	———	-0.004517	-0.003922
	5	———	———	———	———	-0.001544
	6	———	———	0.010614	0.010479	0.010487
	7	———	0.000442	-0.013777	-0.014012	-0.013921
	8	———	———	-0.007858	-0.008508	-0.008497
4	1	0.421893	0.339862	0.343443	0.344360	0.344318
	2	———	0.087017	0.087936	0.092540	0.092686
	3	———	———	-0.022491	-0.017645	-0.017764
	4	———	———	———	-0.004998	-0.004250
	5	———	———	———	———	-0.001854
	6	———	———	0.010636	0.010515	0.010663
	7	———	0.002643	-0.013968	-0.013939	-0.013948
	8	———	———	-0.006826	-0.007546	-0.007642

Table 8.20: Comparison of the values of the renormalized coefficients obtained by renormalizing under majority rule i times a 32×32 Ising lattice at $T = 2.269185$ with those reported in the literature.

i	k	Results	Swendsen (1984b)
1	1	0.356544	0.3643
	2	0.077240	0.0814
	3	-0.013570	-0.0068
	4	-0.003076	-0.0038
	5	-0.001424	-0.0023
	6	0.010390	———
	7	-0.011427	-0.0008
	8	-0.010148	-0.0026
2	1	0.345681	0.3527
	2	0.090329	0.0944
	3	-0.016188	-0.0094
	4	-0.003657	-0.0046
	5	-0.001225	-0.0019
	6	0.010206	———
	7	-0.013484	-0.0075
	8	-0.008440	0.0043
3	1	0.345963	0.3530
	2	0.091553	0.0950
	3	-0.017352	-0.0130
	4	-0.003961	-0.0020
	5	-0.001468	-0.0050
	6	0.009931	———
	7	-0.014654	-0.0040
	8	-0.006170	0.0050

words, the marginal Hamiltonian $W(\boldsymbol{x}_{V_i})$ obtained using the majority rule can be represented more compactly, requiring fewer basis functions.

Our coefficients compare favorably with those in the literature. The renormalized coefficients obtained using a fine lattice of size 32×32 agree qualitatively with Swendsen (1984b), though significant discrepancies exist, especially with regard to the long range and multi-variable interac-

Table 8.21: Comparison of the values of the renormalized coefficients obtained by renormalizing under majority rule i times a 64×64 Ising lattice at $T = 2.269185$ with those reported in the literature.

i	k	Results	Gupta and Cordery (1984)
1	1	0.356582	0.35357
	2	0.077240	0.07528
	3	-0.013475	-0.00756
	4	-0.003085	-0.00621
	5	-0.001503	-0.00269
	6	0.010440	0.00763
	7	-0.011481	-0.01515
	8	-0.010265	-0.00592

Table 8.22: Comparison of the values of the renormalized coefficients obtained by renormalizing under majority rule i times a 128×128 Ising lattice at $T = 2.269185$ with those reported in the literature.

i	k	Results	Ron et al. (2001)	Gupta et al. (1984)
1	1	0.356566	0.351436	0.35358
	2	0.077229	0.076717	0.07488
	3	-0.013365	-0.008779	-0.00758
	4	-0.003093	-0.006560	-0.00618
	5	-0.001570	———	-0.00293
	6	0.010463	0.006493	0.00760
	7	-0.011476	-0.014245	-0.01522
	8	-0.010303	-0.004410	-0.00582
2	1	0.345585	0.340138	0.34608
	2	0.090391	0.089627	0.08845
	3	-0.015655	-0.010948	-0.00929
	4	-0.003702	-0.007155	-0.00700
	5	-0.001490	———	-0.00273
	6	0.010353	0.006018	0.00733
	7	-0.013605	-0.017328	-0.01895
	8	-0.008884	-0.003119	-0.00524

tions. Values obtained by Gupta and Cordery (1984) using the 64×64 base lattice agree to a greater extent, however there are significant deviations in case of the long range and multi-variable interactions. The same difficulties appear in case of the 128×128 base lattice comparison with Gupta and Cordery (1984) and Ron and Swendsen (2001), however the reason is unclear.

8.2 SAMPLING

We move to the main results of this thesis, namely sampling a graphical model using a sequence of subsequently coarse renormalized graphical models. After the brief change to the coarsening scheme used in the previous section, where we coarsened the lattice by a fixed geometrical construction, in this section we construct the coarse lattice V_{i+1} by letting V_{i+1} be a Minimum Vertex Cover (MVC) of the set V_i as described in detail in Chapter 3.

The coarsening algorithm proceeds as follows. Given a graph $G_i = (V_i, E_i)$, the graph $G_{i+1} = (V_{i+1}, E_i)$ is constructed so that V_{i+1} is the MVC of V_i within the graph G_i. Subsequently, the edges E_{i+1} are formed to connect all pairs of nodes $u, v \in V_{i+1}$ such that $\rho(u, v) \leq C \min_{u \neq v} \rho(u, v)$, with $C = 1$. This step guarantees that the conditional probability $P(\boldsymbol{x}_{V_i \setminus V_{i+1}} \mid \boldsymbol{x}_{V_{i+1}})$ can be evaluated.

In order to improve the sparse dependency structure of E_i, a lateral densening step is employed. Because the structure of the lattice is known *a priori*, we selected the lateral graphs to be constructed using Algorithm 3.3 with C_j equal to 1, $\sqrt{2}$, 2, $\sqrt{5}$, $\sqrt{8}$ and 3; the maximum distance was varied between computations as it is the main force driving the accuracy of the method. The densest lateral graph G_i^j, i.e. the one with the highest C_j, was then used to construct a corrective probability density $P_*(\boldsymbol{x}_{V_i})$ that could be used to improve the quality of samples through particle filtering. Figure 8.48 presents a visualization of a representative dependency directed acyclic graph $D = (V, A)$ constructed using the algorithm.

The basis used was a polynomial basis constructed using Algorithm 4.2, using polynomials of order one and three. The third order polynomials were limited such that the radius of the associated clique does not exceed $\sqrt{2}$, therefore the only cubic basis function was typically the plaquette. Even-order polynomials were removed from the basis because the derivative of the Ising model Hamiltonian is odd-symmetric. Although the Ising model does exhibit a great number of symmetries, the lattice symmetries were not used in any way; that is, the SYMMETRYREDUCTION(ϕ, γ) routine was not used to construct the basis functions. As a result, on regular

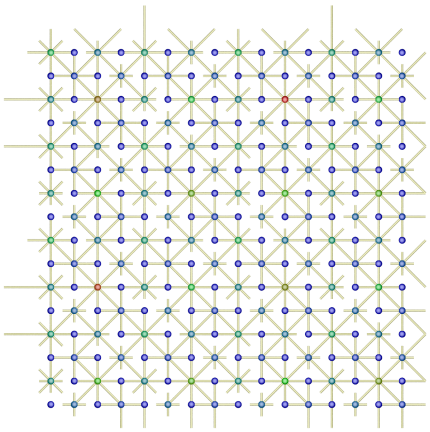
lattices each node had a basis of length up to 12, 16, 24, 28 or 32, depending on the choice of maximum basis width C_j. Additionally, the nodes were considered not to be equivalent and thus the basis coefficients of otherwise equivalent nodes differ from each other: the renormalized coefficients were not assumed to be translation invariant. However, we have employed an equalization step in order to average the coefficients corresponding to the same interaction on the lattice; consult Section 4.1.3.4 for details.

The renormalized coupling coefficients were obtained using the fully-symmetrized fast marginalization scheme with $Q(\boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{\bar{N}(u)})$. We used the mixed discrete-continuous representation and performed projection at five Gaussian quadrature nodes. The membership of the set $\bar{N}(u)$ was determined as u and its neighbors within the sparse graph $G_i = (V_i, E_i)$ without any lateral densening; however, the set of edges between these nodes was taken from the densest laterally coarsened graph $G_i^j = (V_i, E_i^j)$. The coefficients describing the individual conditional probabilities, the corrective marginal densities $P_*(\boldsymbol{x}_{V_i})$ and the weights $Q(\boldsymbol{x}_{V_i})$ were obtained simultaneously using a fixed-point iteration running for six iterations, with 20,000 random samples generated per iteration. The Robbins-Monro algorithm (Robbins and Monro, 1951) was also used with $a_i = 1$ for $i < 3$ and $a_i = 1/(i-2)$ during the following iterations.

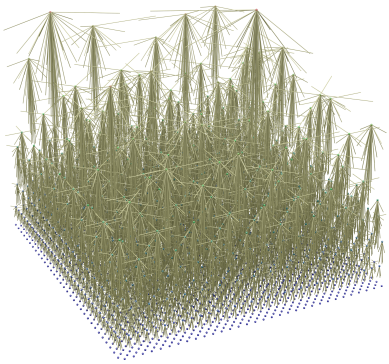
8.2.1 *Sequential importance sampler*

The basic sampler we are comparing against is the sequential importance sampler using directly the dependency graph $D = (V, A)$ and the associated conditional probabilities. Although the conditional probability used by this sampler is much improved, the sampler itself is the same as that used originally by Okunev (2005) and Chorin (2008). The SIS algorithm produces a sample $\boldsymbol{x}_V$ with proposal density $P_{\approx}(\boldsymbol{x}_V)$ calculated using the conditional probabilities of the individual variables x_u. To correct for the difference between the proposal density and the target density $P(\boldsymbol{x}_V)$, the sample $\boldsymbol{x}_V$ is given a weight $w(\boldsymbol{x}_V)$ equal to the ratio $w(\boldsymbol{x}_V) = P(\boldsymbol{x}_V)/P_{\approx}(\boldsymbol{x}_V)$. The weight ensures that

$$\lim_{N\to\infty} \left(\frac{\sum_{i=1}^{N} f(\boldsymbol{x}_V^i) w(\boldsymbol{x}_V^i)}{\sum_{i=1}^{N} w(\boldsymbol{x}_V^i)} \right) = \mathbb{E}\big[f(\boldsymbol{x}_V)\big].$$



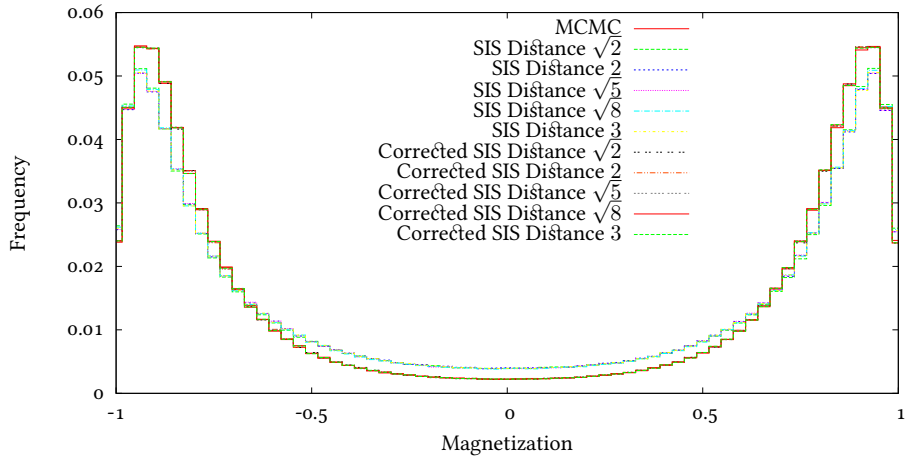
(a) A two-dimensional projection of the dependency graph.



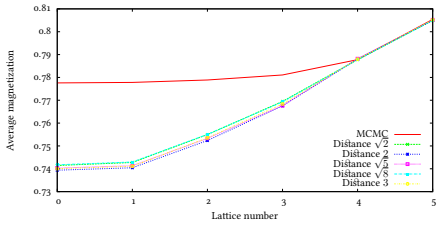
(b) A three-dimensional rendering of the dependency graph.

Figure 8.48: Visualization of the directed acyclic graph $D = (V, A)$ representing the dependencies between variables, constructed for a 64×64 Ising lattice using three stages of lateral densening. The nodes are color-coded according to the order in which they are sampled; red nodes are sampled first, followed by green and finally the blue nodes. Cylinders represent directed arcs $(u, v) \in A$, where an arc (u, v) from u to v implies that the node v depends on the value of the node u. The overwhelming complexity of the resulting structure shows how complicated are the algorithms and their results even for seemingly straightforward, regular graphical models.

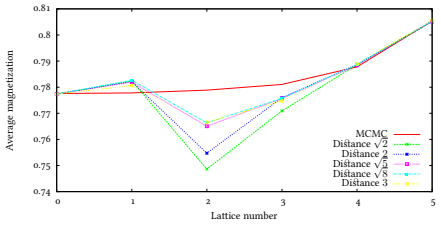
We apply the sequential importance sampler to a two-dimensional Ising lattice at critical coupling and vary the accuracy & cost of the method by



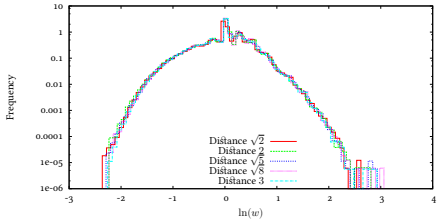
(a) Magnetization histogram.



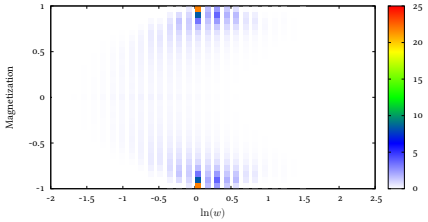
(b) Uncorrected average magnetization.



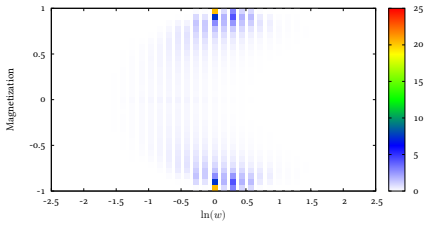
(c) Weight-corrected average magnetization.



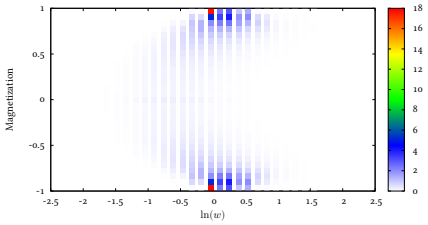
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



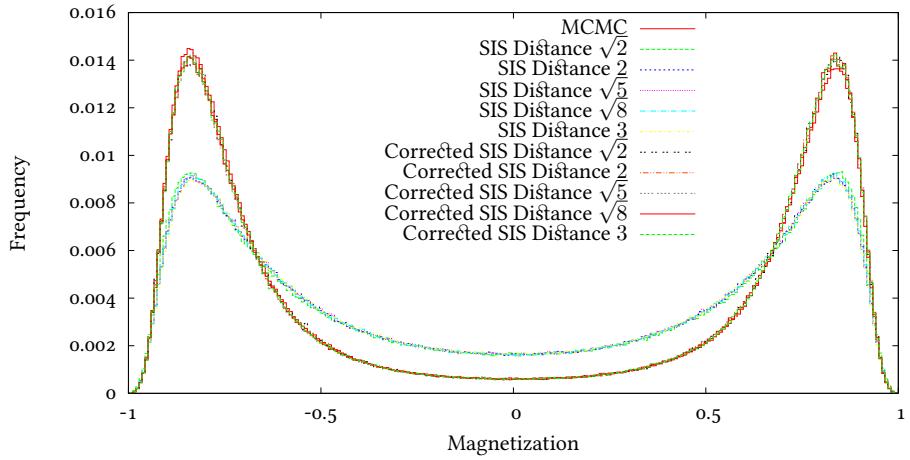
(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.



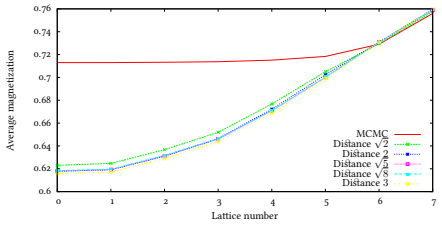
(g) Distribution of weights and magnetization for basis width 3.

Figure 8.49: Performance of the sequential importance sampler on a 8×8 Ising lattice at critical coupling $\mu = \mu_c$.

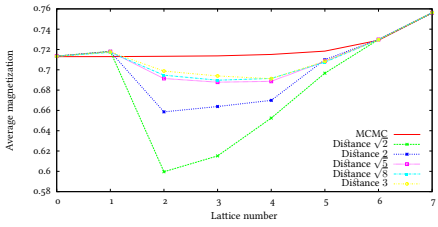
changing the maximum distance reached by the lateral densening graphs, varying from two lateral stages in case of distance $\sqrt{2}$ to six lateral stages



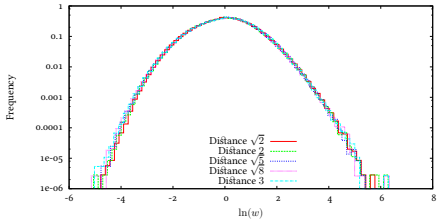
(a) Magnetization histogram.



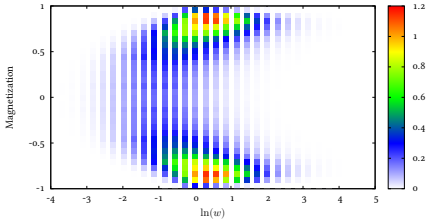
(b) Uncorrected average magnetization.



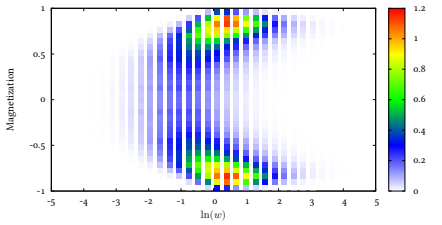
(c) Weight-corrected average magnetization.



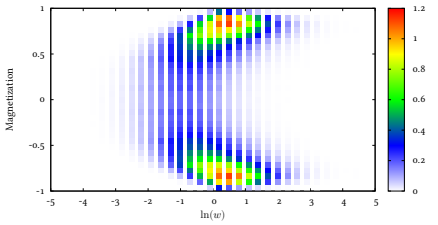
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



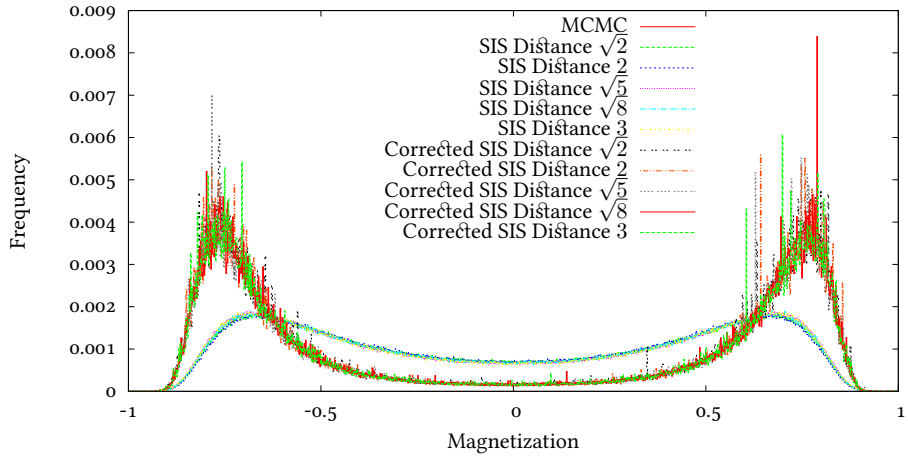
(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.



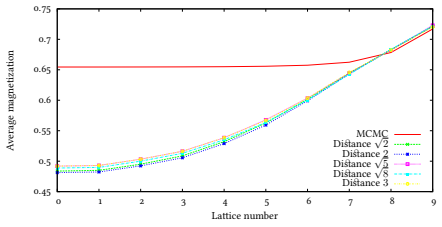
(g) Distribution of weights and magnetization for basis width 3.

Figure 8.50: Performance of the sequential importance sampler on a 16×16 Ising lattice at critical coupling $\mu = \mu_c$.

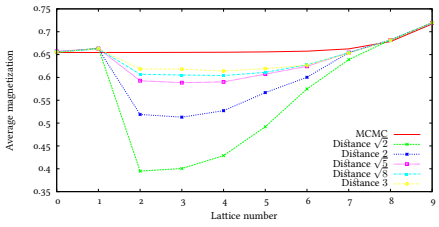
in case of distance 3. Beginning with a small lattice of size 8×8, we observe behavior described by the Figure 8.49. The three plots show, in



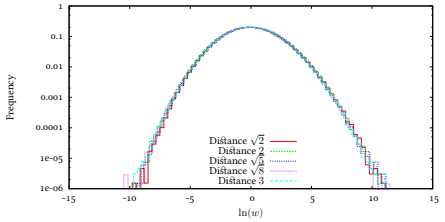
(a) Magnetization histogram.



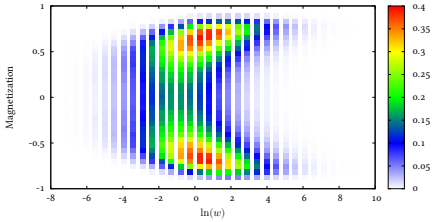
(b) Uncorrected average magnetization.



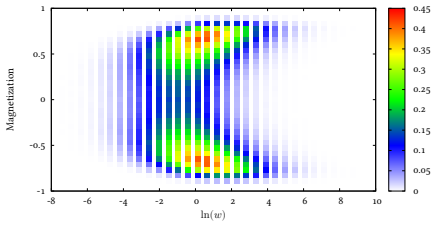
(c) Weight-corrected average magnetization.



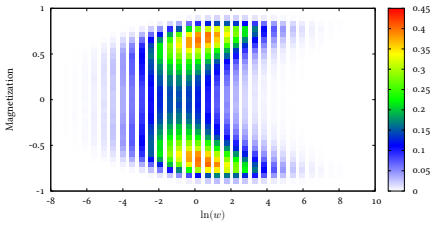
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.



(g) Distribution of weights and magnetization for basis width 3.

Figure 8.51: Performance of the sequential importance sampler on a 32×32 Ising lattice at critical coupling $\mu = \mu_c$.

order, (a) the histogram of magnetization and the average absolute magnetization $\mathcal{M}_{\text{abs}}$ generated by (b) the proposal density and (c) the weight-

corrected proposal density. Figures (c) through (g) show further weight-related benchmarks.

Figure 8.49a shows the distribution of magnetization among states produced from the target probability density sampled using MCMC and compares to it the distribution obtained from the states generated by the uncorrected proposal density $P_{\approx}(\boldsymbol{x}_V)$ (SIS). Also shown is the weight-corrected distribution (Corrected SIS), which should be exactly equal to the MCMC line. Multiple lines correspond to the different accuracy levels, with the distance referring to the maximum distance between variables included in the basis.

Figures 8.49b and 8.49c summarize the data for all lattices V_i by providing us with the average absolute magnetization computed either using the unweighted or weight-corrected proposal density. These curves are compared against the exact values obtained by sampling the original lattice V_0 through MCMC and computing the relevant quantities using the restricted lattices V_i, $i > 0$. The figure 8.49b shows the actual performance of the sampler, while 8.49c describes how well those results could be corrected at each stage of the computation using the corrective probability densities $P_*(\boldsymbol{x}_{V_i})$.

Lattice 0 corresponds to the original, fine lattice. The higher the lattice number the coarser the lattice, with the number of variables decreasing roughly by a factor of two with each step.

Finally, the Figure 8.49d shows the overall distribution of weights, while Figures 8.49e through 8.49g present the joint distribution of weights and magnetizations for different basis widths. The range spanned by weights is indicative of the performance of the method, with a wide range suggesting that the proposal density is not a good approximation of the target density. The joint distribution suggests whether the weights are dependent on magnetization, showing areas that are under- or over-sampled by the proposal density: large weights suggest under-sampling, while low weights suggest over-sampling.

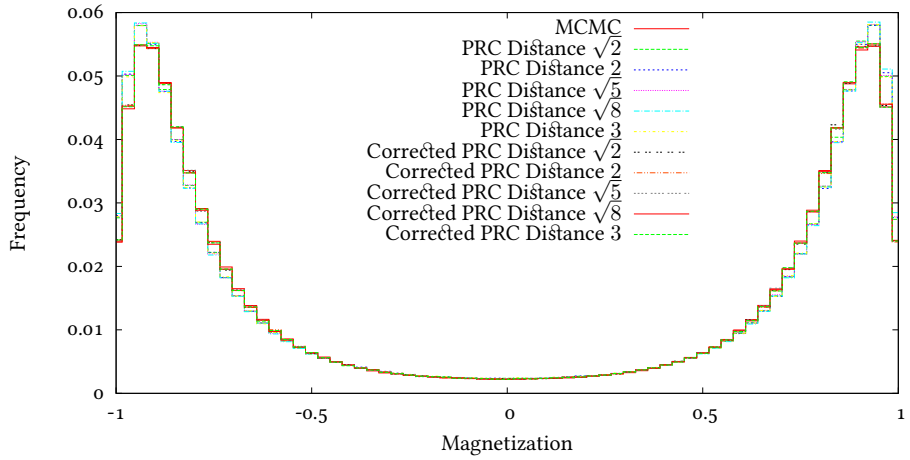
The close proximity between the magnetization curves on Figure 8.49b and the virtually indistinguishable distributions of weights on Figure 8.52d show that the increased accuracy of the proposal density obtained through lateral densening has little practical effect on the quality of the proposal density. This behavior is expected, as the increased accuracy coming from a wider basis is utilized by a small number of variables; see Section 3.5 for a detailed discussion. However, Figure 8.49c shows on the other hand that the corrective probability densities $P_*(\boldsymbol{x}_{V_i})$ computed on the densest lateral graph G_i^j indeed improve our ability to correct the results: as the basis width increases, so does the quality of the weight-corrected approximation on lattices V_i, $i > 0$, where the weights can be computed only approximately.

As the size of the original lattice increases to 16×16 (Figure 8.50) and 32×32 (Figure 8.51), the proposal density begins to produce a histogram that varies markedly from that obtained using the target distribution. It appears that the proposal density consistently underestimates the strength of the interactions between variables, producing samples of lower magnetization than expected: the histograms on Figures 8.50a and 8.51a are too flat compared to the correct shape. This can be seen clearly from the weights alone. Figures 8.50e–8.50g and 8.51e–8.51g, where the histograms take a characteristic *butterfly* shape: low magnetization states (middle of the graph) are over-sampled, while high magnetization states (extremes, or wings, of the graph) are under-sampled.

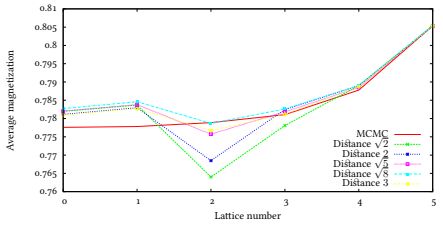
In the case of the 32×32 lattice, the sequential importance sampler no longer performs acceptably. In fact, the corrected histogram on Figure 8.51a is extremely noisy. The relatively poor sample quality can be observed directly through inspection of the weight distribution on Figure 8.51d: the ratio of the largest to the average weight is of the order of $\mathrm{e}^{10} \approx 22000$. Therefore, each of these very large weight samples equals approximately 22000 average samples, causing the noisy behavior observed on the magnetization histogram. Additional increases in the size of the original lattice decrease the performance even further, making it no longer possible to use weights to satisfactorily correct the mismatch between the proposal density and the target density.

We focus our attention on Figures 8.51b and 8.51c to point out two important observations. The magnetization curves of Figure 8.51b are of higher quality than some of those corrected using the dense corrective probability distributions $P_*(\boldsymbol{x}_{V_i})$. In particular, the corrective probability distribution using basis of width $\sqrt{2}$ performs worse than the uncorrected proposal density for lattices V_i with $2 \leq i \leq 7$. This suggests that the influence of the variables sampled on the coarsest lattices remains very strong, since it keeps the magnetization level above what would be obtained using the approximate marginals alone.

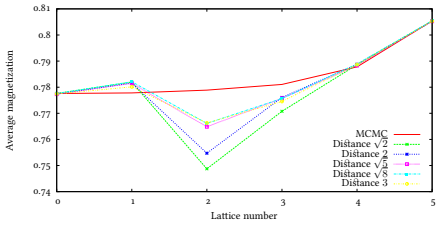
The use of a wider basis in the computation of $P_*(\boldsymbol{x}_{V_i})$ allows for an improvement over the states produced by the sequential importance sampler. Looking at the graph 8.51c, the bases of width $\sqrt{8}$ and 3 produce states whose magnetization never falls below 0.6, while the sequential sampler produces magnetization of the order of 0.5 and lower. Therefore, correcting the samples generated using the sequential sampler during the process of conditional sampling should bring a significant improvement.



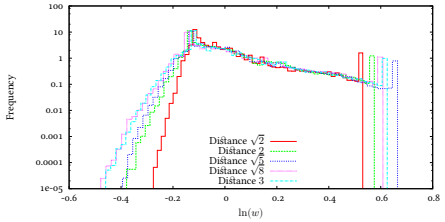
(a) Magnetization histogram.



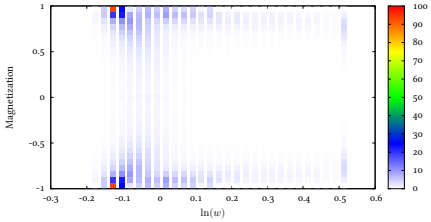
(b) Uncorrected average magnetization.



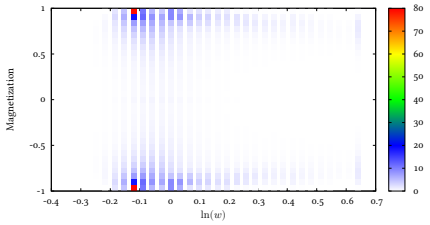
(c) Weight-corrected average magnetization.



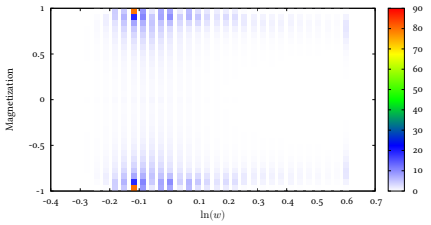
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.

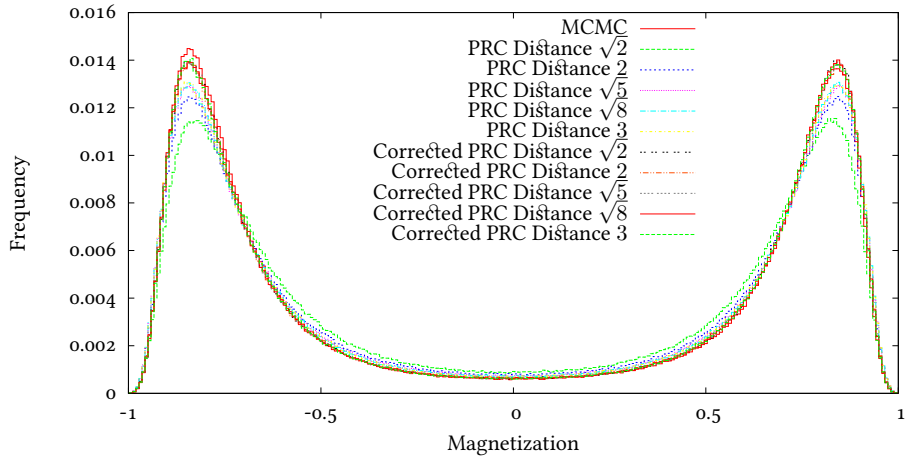


(g) Distribution of weights and magnetization for basis width 3.

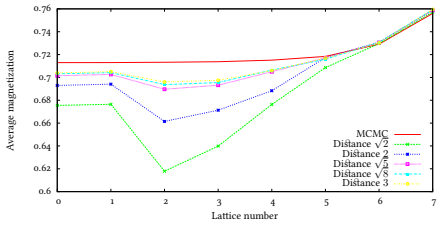
Figure 8.52: Performance of the partial rejection control sampler on a 8×8 Ising lattice at critical coupling $\mu = \mu_c$.

8.2.2 *Partial rejection control sampler*

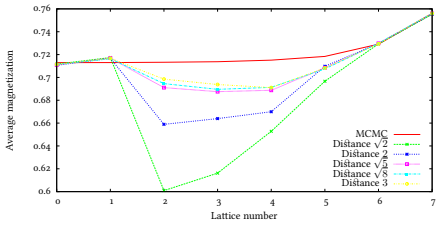
The partial rejection control sampler is an extension of the above sampler using particle filtering to improve the samples at intermediate lattices



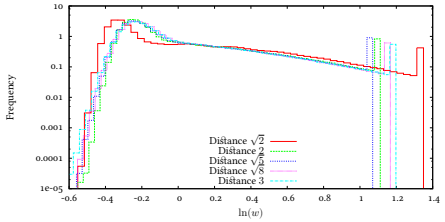
(a) Magnetization histogram.



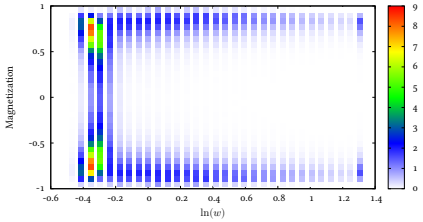
(b) Uncorrected average magnetization.



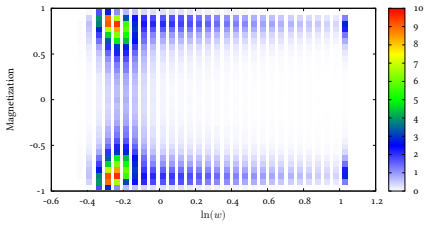
(c) Weight-corrected average magnetization.



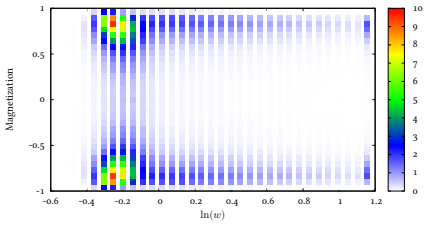
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



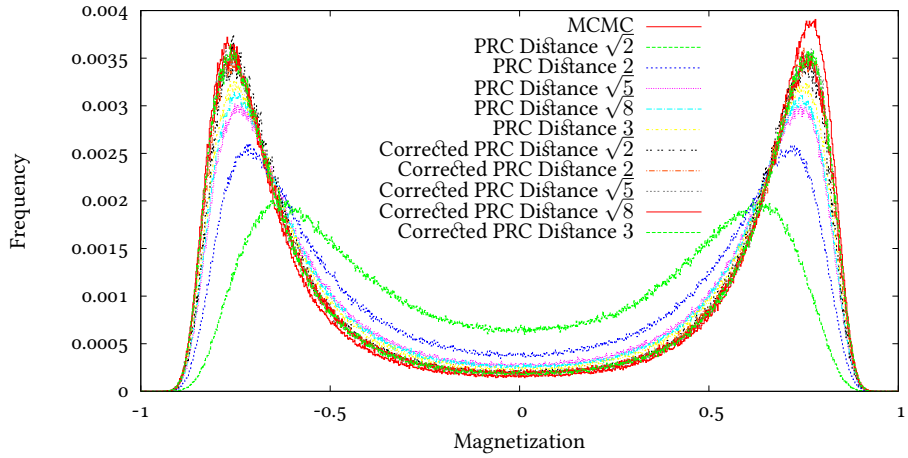
(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.



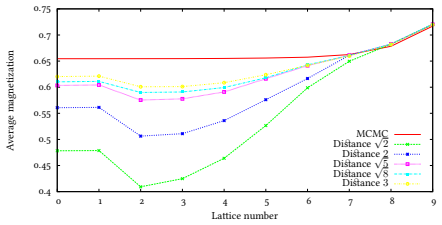
(g) Distribution of weights and magnetization for basis width 3.

Figure 8.53: Performance of the partial rejection control sampler on a 16×16 Ising lattice at critical coupling $\mu = \mu_c$.

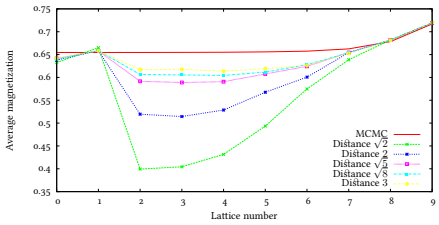
(Doucet, de Freitas, and Gordon, 2001, p. 233). In order to facilitate this improvement, at each intermediate lattice V_i we use the densest lateral



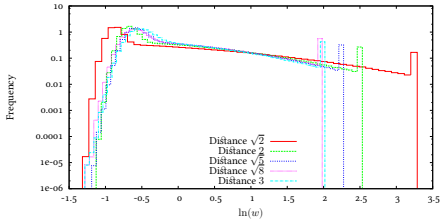
(a) Magnetization histogram.



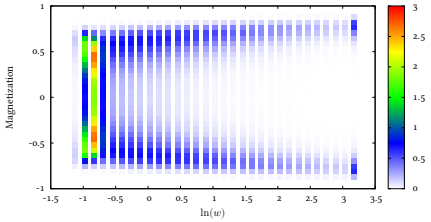
(b) Uncorrected average magnetization.



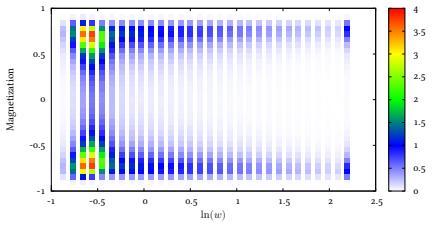
(c) Weight-corrected average magnetization.



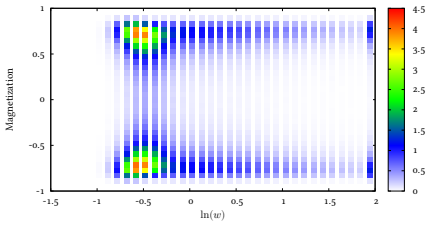
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{2}$.



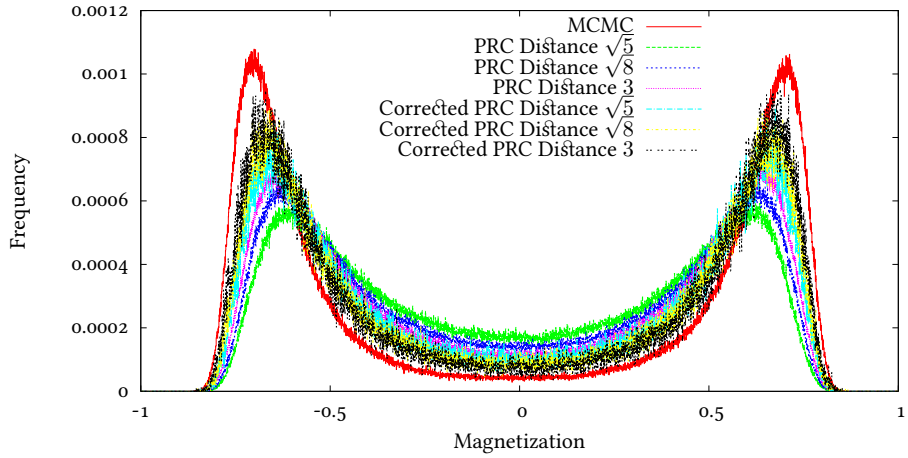
(f) Distribution of weights and magnetization for basis width $\sqrt{5}$.



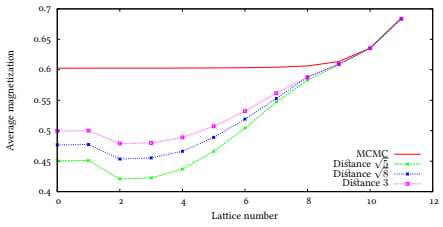
(g) Distribution of weights and magnetization for basis width 3.

Figure 8.54: Performance of the partial rejection control sampler on a 32×32 Ising lattice at critical coupling $\mu = \mu_c$.

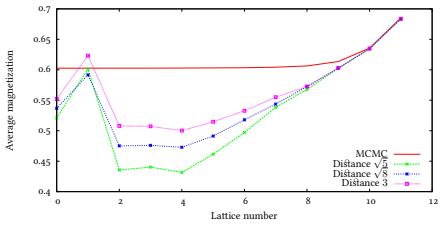
graph G_i^j to construct a corrective probability density $P_*(\boldsymbol{x}_{V_i})$. This cor-



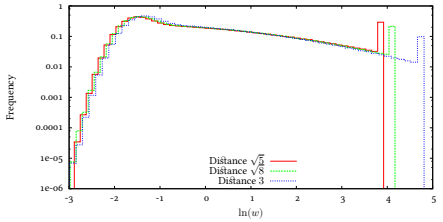
(a) Magnetization histogram.



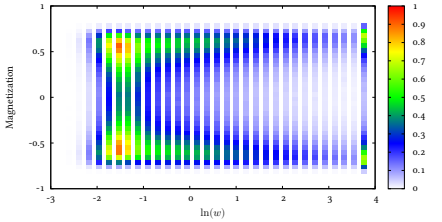
(b) Uncorrected average magnetization.



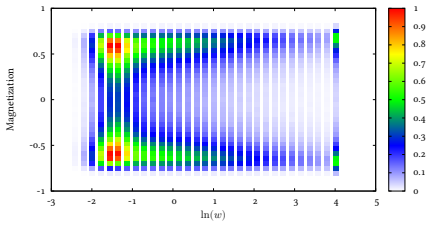
(c) Weight-corrected average magnetization.



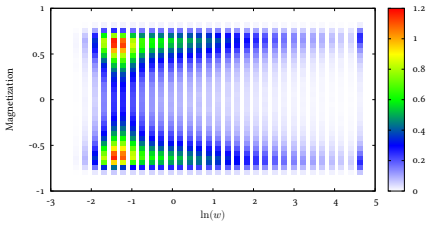
(d) Distribution of weights.



(e) Distribution of weights and magnetization for basis width $\sqrt{5}$.



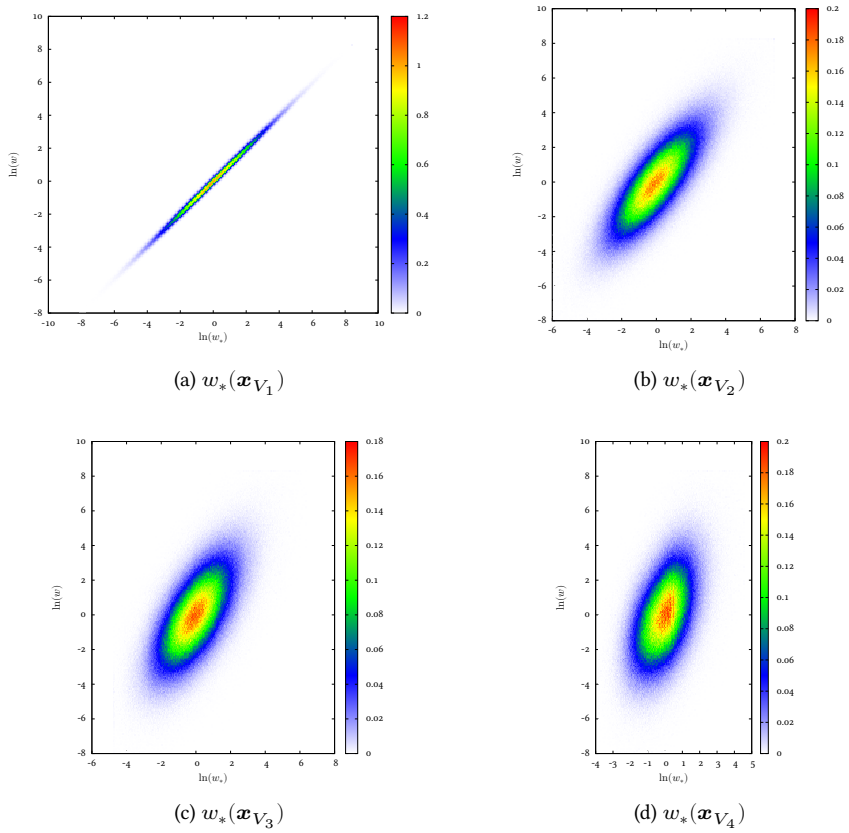
(f) Distribution of weights and magnetization for basis width $\sqrt{8}$.



(g) Distribution of weights and magnetization for basis width 3.

Figure 8.55: Performance of the partial rejection control sampler on a 64×64 Ising lattice at critical coupling $\mu = \mu_c$.

rective probability density is obtained together with the proposal density $P_{\approx}(\boldsymbol{x}_V)$ using the fast marginalization method.



(a) $w_*(\boldsymbol{x}_{V_1})$ (b) $w_*(\boldsymbol{x}_{V_2})$

(c) $w_*(\boldsymbol{x}_{V_3})$ (d) $w_*(\boldsymbol{x}_{V_4})$

Figure 8.56: Correlation between the approximate weights $w_*(\boldsymbol{x}_{V_i})$ and the final weights $w(\boldsymbol{x}_V)$ computed using a basis of width 3 on a 32×32 Ising lattice at $\mu = \mu_c$, showing the predictive value of the approximate weights. If the prediction were exact, the points would form a straight line; however, the strength of the correlation is limited due to the approximate nature of the weights $w_*(\boldsymbol{x}_{V_i})$ and changes in the proposal density.

Assume for now that for each lattice V_i we have an unnormalized corrective probability density $P_*(\boldsymbol{x}_{V_i})$ and a weight threshold c_i. We begin by sampling M samples $\boldsymbol{x}^p_{V_m}$ on the coarsest lattice V_m for $p = 1, 2, \dots, M$. These individual samples are referred to as particles and are assumed to follow the probability $P(\boldsymbol{x}_{V_m})$, thus we assign each particle the proposal density $P_\approx(\boldsymbol{x}^p_{V_m}) = P(\boldsymbol{x}^p_{V_m})$. With the assumption that the lattice V_{i+1} has been sampled, the transition to lattice V_i proceeds as follows. We sample the variables $\boldsymbol{x}^p_{V_i \setminus V_{i+1}}$ for each particle using the usual sequential im-

portance sampler, obtaining a completed state $\boldsymbol{x}^p_{V_i}$ with proposal density $P_{\approx}(\boldsymbol{x}^p_{V_i})$. For each particle we compute the corrective weight

$$w_*(\boldsymbol{x}^p_{V_i}) = \frac{P_*(\boldsymbol{x}^p_{V_i})}{P_{\approx}(\boldsymbol{x}^p_{V_i})}.$$

If the weight $w_*(\boldsymbol{x}^p_{V_i}) > c_i$, the particle is accepted unconditionally. Otherwise, the particle is accepted with probability $\min\{1, w_*(\boldsymbol{x}^p_{V_i})/c_i\}$, in which case the proposal density is updated to $P_*(\boldsymbol{x}^p_{V_i})/c_i$.

If the particle is rejected, we return to the lattice V_{i+1} and choose at random a particle $\boldsymbol{x}^q_{V_{i+1}}$, $1 \leq q \leq M$, with probability proportional to its weight $w_*(\boldsymbol{x}^q_{V_{i+1}})$. We then assign the resampled particle $\boldsymbol{x}'_{V_{i+1}}$ an initial proposal density

$$P_{\approx}(\boldsymbol{x}'_{V_{i+1}}) = \frac{M}{\sum_{k=1}^{M} w_*(\boldsymbol{x}^k_{V_{i+1}})} P_{\approx}(\boldsymbol{x}^q_{V_{i+1}})$$

and complete the state by sampling the variables $\boldsymbol{x}'_{V_i \setminus V_{i+1}}$ using the sequential importance sampler, obtaining a regenerated particle $\boldsymbol{x}'_{V_i}$ with proposal density $P_{\approx}(\boldsymbol{x}'_{V_i})$. Finally, we set $\boldsymbol{x}^p_{V_i} = \boldsymbol{x}'_{V_i}$ and $P_{\approx}(\boldsymbol{x}^p_{V_i}) = P_{\approx}(\boldsymbol{x}'_{V_i})$ and repeat the rejection step with weight threshold c_i. The regeneration and rejection process continues until the particle $\boldsymbol{x}^p_{V_i}$ is accepted.

The weight thresholds c_i are determined ahead of time by sampling a single, very large batch of approximately 1000 particles. When the particles reach a lattice V_i, we compute the weights $w_*(\boldsymbol{x}^p_{V_i})$ and choose the threshold c_i to be

$$c_i = \max\left\{\frac{C_{98}(w_*(\boldsymbol{x}^p_{V_i}))}{10}, \frac{Q_2(w_*(\boldsymbol{x}^p_{V_i})) + Q_3(w_*(\boldsymbol{x}^p_{V_i}))}{2}\right\},$$

where $C_k(\cdot)$ and $Q_k(\cdot)$ are the k^{th} percentile and k^{th} quantile of the weights, respectively. Once c_i is computed, the particles undergo a rejection step, with the accepted particles and proceeding further. Once the thresholds c_i are known for all the lattices V_i, including the the original lattice V_0, we sample particles in batches of 40 particles and perform resampling on every lattice V_i.

Figure 8.52 shows the performance of the partial rejection control sampler on the 8×8 Ising lattice. Comparing it with the analogous Figure 8.49 describing the performance of the sequential importance sampler, we see an immediate improvement. The weights span a much smaller range and

the uncorrected histogram shown on Figure 8.52a is almost exact. Performance remains similar in case of the 16×16 lattice.

The 32×32 lattice remains more difficult. We observe that weights span the range of about $\mathrm{e}^5 \approx 150$, which causes the noise visible in the histogram on Figure 8.54a. Increasing the width of the basis helps in reducing the ratio between the largest and smallest weights. Finally, Figure 8.55 shows that the sampler begins to fail in the case of the 64×64 lattice, where the weights span the range of about $\mathrm{e}^{10} \approx 22000$, marking the limits of applicability of the sampling method.

8.3 DISCUSSION

The performance of the acyclic Monte Carlo method applied to the Ising model is far from satisfactory, given that the 64×64 Ising lattice can be sampled quite successfully using the Markov Chain Monte Carlo (MCMC) method. The aim of the acyclic Monte Carlo method is not, however, to replace the cluster method of Wolff (1989) or other specialized methods; instead, we hoped to apply it to the Ising model in order to study its behavior in this relatively simple scenario.

The largest source of error is caused by the values of the numerically computed renormalized coupling coefficients. We found that the renormalized coefficients are continuous functions of the original couplings and thus do not undergo a rapid change across the phase transition, while the observable quantities vary significantly. Therefore, a relatively small error in the estimated coefficients leads to a large change in the behavior of the model, resulting in large sampling errors. This is visible especially strongly in the Ising model, where spins act in a coherent manner and thus the errors committed in the calculation of the renormalized coefficients tend to drive the system in the same direction: e.g. the magnetization is consistently underestimated because the coupling coefficients are consistently too low.

There are a number of approaches that proved successful in improving this situation. We find that with $Q(\boldsymbol{x}_{V_i}) = 1$ enlarging the basis does not bring a significant improvement to the quality of the approximate marginal probability density. Our suspicion is that this is due to the nature of the weighted inner product being used, which attempts to find an approximation to the derivative of the marginal Hamiltonian $\partial W(\boldsymbol{x}_{V_i})/\partial x_u$ that minimizes the approximation error in areas where the probability $P(\boldsymbol{x}_{V_i})$ — hence also the Hamiltonian $W(\boldsymbol{x}_{V_i})$ — is large. The unintended consequence of this fact is that such an approximation consistently underestimates the true values of the derivative, because $\partial W(\boldsymbol{x}_{V_i})/\partial x_u$ and

$W(\boldsymbol{x}_{V_i})$ are anti-correlated. We find that using $Q(\boldsymbol{x}_{V_i}) = P(\boldsymbol{x}_{\bar{N}(u)})$ flattens the weights and improves the renormalized coefficients, partially removing the bias introduced by the inner product. We saw in Table 8.14 that as the lattice becomes coarser, the relative strength of the long-range interactions increases as well. Therefore, it may become necessary to use a larger neighborhood of the node $u \in V_i$ in the flattening factor $Q(\boldsymbol{x}_{V_i})$ in order for it to be equally useful as on finer lattices.

The increased range of interactions is a property of the coarsening rule used. Within the present thesis we used decimation, as it is the only coarsening rule that allows for using sequential importance sampling and other advanced sampling techniques. When this premise is abandoned, the generalized fast marginalization could be used to compute renormalized coupling coefficients for arbitrary coarsening rules, in particular for a coarsening rule optimized for basis size. The frequently used majority rule in particular has a relatively small basis: Brandt and Ron (2001b) report that a basis constructed over a 20-node neighborhood (basis width of $\sqrt{5}$) already produces a very good approximation to the renormalized Hamiltonian, which in the case of decimation is not quite large enough. Therefore, the renormalized Hamiltonians computed using the generalized fast marginalization could be used together with a Markov Chain Monte Carlo (MCMC) sampling scheme described in Chapter 6 to sample the original probability distribution in the same way as the method of Brandt and Ron (2001b), however with the ability to use the method in a natural manner for both discrete and continuous systems. However, using general coarsening rules removes the ability to gauge the performance of the sampling method through analysis of the weight distribution, making it less appealing in more difficult applications, where no benchmark results might be known.

The acyclic Monte Carlo has two significant strengths. The samples generates by our method are entirely independent of each other, with the caveat that the particles within each batch are correlated due to the presence of the resampling stage. Therefore, the method does not suffer because of critical slowing down or long autocorrelation times. At the same time the method is very general and does not use any special properties of the Ising model, making us hopeful that other statistical models, where the errors committed by the method have a chance to cancel out, will see better performance than that observed in the case of the Ising model.

BIBLIOGRAPHY

Edoardo M. Airoldi (2007). "Getting Started in Probabilistic Graphical Models". In: *PLoS Computational Biology* 3 (12), pp. 2421–2425.

Vladimir I. Arnold (1973). *Ordinary Differential Equations*. Trans. by Richard A. Silverman. Cambridge, MA: MIT Press.

Carlo Berzuini, Nicola G. Best, Walter R. Gilks and Cristiana Larizza (1997). "Dynamic Conditional Independence Models and Markov Chain Monte Carlo Methods". In: *Journal of American Statistical Association* 92 (440), pp. 1403–1412.

James J. Binney, N. J. Dowrick, Andrew J. Fisher and Mark E. Newman (1992). *The Theory of Critical Phenomena*. Oxford, UK: The Clarendon Press.

Achi Brandt and Dorit Ron (2001a). "Renormalization multigrid (RMG): coarse-to-fine Monte Carlo acceleration and optimal derivation of macroscopic descriptions". In: *Multiscale Computational Methods in Chemistry and Physics*. Ed. by Achi Brandt, Jerzy Bernholc, and Kurt Binder. Vol. 177. NATO Science Series: Computer and System Sciences. Amsterdam: IOS Press, pp. 163–186.

——— (2001b). "Renormalization multigrid (RMG): Statistically optimal renormalization group flow and coarse-to-fine Monte Carlo acceleration". In: *Journal of Statistical Physics* 102 (1/2), pp. 231–257.

William L. Briggs, Van E. Henson and Steve F. McCormick (2000). *A Multigrid Tutorial*. 2nd ed. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Tony F. Chan and Hendrik A. van der Vorst (1997). "Approximate and Incomplete Factorizations". In: *Parallel Numerical Algorithms*. Ed. by David E. Keyes, Ahmed Sameh, and V. Venkatakrishnan. Vol. 4. ICASE/LaRC Interdisciplinary Series in Science and Engineering. Springer, pp. 167–202. ISBN: 978-94-010-6277-0. DOI: 10.1007/978-94-011-5412-3_6. URL: http://dx.doi.org/10.1007/978-94-011-5412-3_6.

Alexandre J. Chorin (2003). "Conditional expectations and renormalization". In: *Multiscale Modeling and Simulation* 1, pp. 105–118.

——— (2008). "Monte Carlo without chains". In: *Communications in Applied Mathematics and Computational Science* 3, pp. 77–93.

Alexandre J. Chorin, Ole H. Hald and Raz Kupferman (2000). "Optimal prediction and the Mori-Zwanzig representation of irreversible processes".

In: *Proceedings of the National Academy of Sciences USA* 97, pp. 2968–2973.

Alexandre J. Chorin, Ole H. Hald and Raz Kupferman (2002). "Optimal prediction with memory". In: *Physica D* 166, pp. 239–257.

Alexandre J. Chorin and Panagiotis Stinis (2005). "Problem reduction, renormalization, and memory". In: *Communications in Applied Mathematics and Computational Science* 1, pp. 1–27.

C. K. Chow and C. N. Liu (1968). "Approximating discrete probability distributions with dependence trees". In: *IEEE Transactions on Information Theory* IT-14 (3), pp. 462–467.

Earl A. Coddington and Norman Levinson (1955). *Theory of Ordinary Differential Equations*. Malabar, FL: Krieger Publishing Company.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein (1990). *Introduction to algorithms*. 3rd. Cambridge, MA: MIT Press.

Timothy A. Davis (2004). "Algorithm 832: UMFPACK V4.3—an unsymmetric-pattern multifrontal method". In: *ACM Transactions on Mathematical Software* 30 (2), pp. 196–199. DOI: 10.1145/992200.992206. URL: http://dx.doi.org/10.1145/992200.992206.

Timothy A. Davis, John R. Gilbert, Stefan I. Larimore and Esmond G. Ng (2004a). "A column approximate minimum degree ordering algorithm". In: *ACM Transactions on Mathematical Software* 30 (3), pp. 353–376. DOI: 10.1145/1024074.1024079. URL: http://doi.acm.org/10.1145/1024074.1024079.

——— (2004b). "Algorithm 836: COLAMD, a column approximate minimum degree ordering algorithm". In: *ACM Transactions on Mathematical Software* 30 (3), pp. 377–380. DOI: 10.1145/1024074.1024080. URL: http://doi.acm.org/10.1145/1024074.1024080.

James W. Demmel (1997). *Applied numerical linear algebra*. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Edsger W. Dijkstra (1959). "A note on two problems in connection with graphs". In: *Numerische Mathematik* 1 (1), pp. 269–271. DOI: 10.1007/BF01386390. URL: http://dx.doi.org/10.1007/BF01386390.

Arnaud Doucet, Nando de Freitas and Neil Gordon, eds. (2001). *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. New York, NY: Springer.

Lawrence C. Evans (1998). *Partial differential equations*. Vol. 19. Graduate Studies in Mathematics. Providence, RI: American Mathematical Society.

John G. Francis (1961). "The QR Transformation, I". In: *The Computer Journal* 4 (3), pp. 265–271.

John G. Francis (1962). "The QR Transformation, II". In: *The Computer Journal* 4 (4), pp. 332–345.

Jonathan Goodman and Alan D. Sokal (1989). "Multigrid Monte Carlo, conceptual foundations". In: *Physical Review D* 40, pp. 2035–2071.

Neil Gordon, David Salmond and Craig Ewing (1995). "Bayesian State Estimation for Tracking and Guidance Using the Bootstrap Filter". In: *Journal of Guidance, Control, and Dynamics* 18 (6), pp. 1434–1443. DOI: 10.2514/3.21565.

Rajan Gupta and Robert Cordery (1984). "Monte Carlo Renormalized Hamiltonian". In: *Physics Letters* 105A (8), pp. 415–417.

John M. Hammersley and Peter Clifford (1971). "Markov Fields on finite graphs and lattices". Unpublished manuscript. URL: http://www.statslab.ca.ac.uk/~grg/books/hammfest/hamm-cliff.pdf.

Pinar Heggernes, Stanley C. Eisenstat, Gary K. Kumfert and Alex Pothen (2001). *The Computational Complexity of the Minimum Degree Algorithm*. Tech. rep. Hampton, VA: Institute for Computer Applications in Science and Engineering. URL: http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA398632.

Ernst Ising (1925). "Beitrag zur Theorie des Ferromagnetismus". In: *Zeitschrift für Physik* 31 (1), pp. 253–258.

Michael I. Jordan (2004). "Graphical Models". In: *Statistical Science* 19 (1), pp. 140–155. DOI: 10.1214/088342304000000026.

Leo P. Kadanoff (1966). "Scaling Laws for Ising Models Near T_c". In: *Physics* 2, p. 263.

——— (1975). "Variational Principles and Approximate Renormalization Group Calculations". In: *Physical Review Letters* 34 (16), pp. 1005–1008.

——— (2002). *Statistical Physics, Statics, Dynamics, and Renormalization*. Singapore: World Scientific.

Leo P. Kadanoff and Anthony Houghton (1975). "Numerical evaluations of the critical properties of the two-dimensional Ising model". In: *Physical Review B* 11, pp. 377–386.

Richard M. Karp (1972). "Reducibility Among Combinatorial Problems". In: *Complexity of Computer Computations*. Ed. by R. E. Miller and J. W. Thatcher. New York, NY: Plenum, pp. 85–103.

David S. Kershaw (1978). "The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations". In: *Journal of Computational Physics* 26 (1), pp. 43–65. DOI: 10.1016/0021-9991(78)90098-0.

Daphne Koller and Nir Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. 1st. Cambridge, MA: MIT Press.

Vera N. Kublanovskaya (1962). “On some algorithms for the solution of the complete eigenvalue problem”. In: *USSR Computational Mathematics and Mathematical Physics* 1 (3), pp. 637–657. Trans. of Вера Н. Кублановская. “О некоторых алгорифмах для решения полной проблемы собственных значений”. В: *Журнал вычислительной математики и математической физики* 1 (4), с. 555—570.

Wilhelm Lenz (1920). “Beiträge zum Verständnis der magnetischen Eigenschaften in festen Körpern”. Deutsch. In: *Physikalische Zeitschrift* 21, S. 613–615.

Jun S. Liu (2001). *Monte Carlo Strategies in Scientific Computing*. New York, NY: Springer.

Jun S. Liu, Rong Chen and Tanya Logvinenko (2001). “A Theoretical Framework for Sequential Importance Sampling with Resampling”. In: *Sequential Monte Carlo Methods in Practice*. Ed. by Arnaud Doucet, Nando de Freitas, and Neil Gordon. Statistics for Engineering and Information Science. New York, NY: Springer, pp. 225–246.

Jun S. Liu, Rong Chen and Wing H. Wong (1998). “Rejection control and sequential importance sampling”. In: *Journal of the American Statistical Association* 93 (443), pp. 1022–1031.

Shang–keng Ma (1976). “Renormalization Group by Monte Carlo Methods”. In: *Physics Letters* 37, pp. 461–464.

J. A. Meijerink and Hendrik A. van der Vorst (1977). “An Iterative Solution Method for Linear Systems of Which the Coefficient Matrix is a Symmetric M-Matrix”. In: *Mathematics of Computation* 31 (137), pp. 148–162. DOI: 10.2307/2005786.

Nicolas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller and Edward Teller (1953). “Equation of State Calculations by Fast Computing Machines”. In: *Journal of Chemical Physics* 21 (6), pp. 1087–1092. DOI: 10.1063/1.1699114.

Alexander A. Migdal (1975). “Phase transitions in gauge and spin-lattice systems”. In: *Journal of Experimental and Theoretical Physics* 42 (4), pp. 743–746. Trans. of Александр А. Мигдал. “Фазовые переходы в калибровочных и спиновых решеточных моделях”. В: *Журнал Экспериментальной и Теоретической Физики* 69 (4), с. 1457—1465.

Niels Munksgaard (1980). “Solving Sparse Symmetric Sets of Linear Equations by Preconditioned Conjugate Gradients”. In: *ACM Transactions on Mathematical Software* 6 (2), pp. 206–219. DOI: 10.1145/355887.355893. URL: http://dx.doi.org/10.1145/355887.355893.

Michael Nauenberg and Bernard Nienhuis (1974a). “Critical Surface for Square Ising Spin Lattice”. In: *Physical Review Letters* 33 (16), pp. 944–946.

Michael Nauenberg and Bernard Nienhuis (1974b). "Renormalization-Group Approach to the Solution of General Ising Models". In: *Physical Review Letters* 33 (27), pp. 1598–1601.

Bernard Nienhuis and Michael Nauenberg (1975). "First-Order Phase Transitions in Renormalization-Group Theory". In: *Physical Review Letters* 35 (8), pp. 477–479.

Pavel Okunev (2005). "Renormalization methods with applications to spin physics and to finance". PhD thesis. University of California at Berkeley.

Lars Onsager (1944). "Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition". In: *Physical Review* 65 (3–4), pp. 117–149.

Joshua S. Paul and Yun S. Song (2010). "A Principled Approach to Deriving Approximate Conditional Sampling Distributions in Population Genetics Models with Recombination". In: *Genetics* 186 (1), pp. 321–338.

Hoifung Poon and Pedro Domingos (2011). "Sum–Product Networks: A New Deep Architecture". In: *Proceedings of the IEEE International Conference Computer Vision Workshops*. Barcelona, Spain, pp. 689–690. DOI: 10.1109/ICCVW.2011.6130310.

Herbert Robbins and Sutton Monro (1951). "A Stochastic Approximation Method". In: *Annals of Mathematical Statistics* 22 (3), pp. 400–407.

Christian P. Robert and George Casella (2004). *Monte Carlo statistical methods*. 2nd. New York, NY: Springer.

Dorit Ron and Robert H. Swendsen (2001). "Calculation of effective Hamiltonians for renormalized or non-Hamiltonian systems". In: *Physical Review E* 63, pp. 066128-1–066128-7.

——— (2002). "Importance of multispin couplings in renormalized Hamiltonians". In: *Physical Review E* 66, pp. 056106-1–056106-4.

Dorit Ron, Robert H. Swendsen and Achi Brandt (2002). "Inverse Monte Carlo renormalization group transformations for critical phenomena". In: *Physical Review Letters* 89 (27), pp. 275701-1–275701-4.

Yousef Saad (2003). *Iterative methods for sparse linear systems*. 2nd ed. Philadephia, PA: SIAM.

Sergei Shmulyian (1999). "Towards optimal multigrid Monte Carlo computations in two-dimensional O(n) non-linear σ-models". PhD thesis. Rehovot 76100, Israel: Weizmann Institute of Science.

Robert H. Swendsen (1979a). "Monte Carlo Renormalization Group". In: *Physical Review Letters* 42 (14), pp. 859–861.

——— (1979b). "Monte Carlo renormalization group studies of the $d = 2$ Ising model". In: *Physical Review B* 20, pp. 2080–2087.

——— (1981). "Monte Carlo Renormalization-Group Transformations in Momentum Space". In: *Physical Review Letters* 47 (16), pp. 1159–1162.

Robert H. Swendsen (1984a). "Monte Carlo Calculation of Renormalized Coupling Parameters". In: *Physical Review Letters* 52 (14), pp. 1165–1168.

——— (1984b). "Monte Carlo calculation of renormalized coupling parameters. I. $d = 2$ Ising model". In: *Physical Review B* 30 (7), pp. 3866–3874.

——— (1984c). "Monte Carlo calculation of renormalized coupling parameters. II. $d = 3$ Ising model". In: *Physical Review B* 30 (7), pp. 3875–3881.

Robert H. Swendsen and A. Nihat Berker (1983). "Critical behavior of the three-state Potts model: Monte Carlo renormalization group". In: *Physical Review B* 28 (7), pp. 3897–3903.

Robert H. Swendsen and Jian-Sheng Wang (1987). "Nonuniversal critical dynamics in Monte Carlo simulations". In: *Physical Review Letters* 58, pp. 86–88.

Martin J. Wainwright and Michael I. Jordan (2008). "Graphical Models, Exponential Families, and Variational Inference". In: *Foundations and Trends in Machine Learning* 1 (1–2), pp. 1–305. DOI: 10.1561/2200000001.

James W. Watts (1981). "A Conjugate Gradient-Truncated Direct Method for the Iterative Solution of the Reservoir Simulation Pressure Equation". In: *Journal of the Society of Petroleum Engineers* 21 (3), pp. 345–353. DOI: 10.2118/8252-PA.

Jonathan Weare (2007). "Efficient Monte Carlo sampling by parallel marginalization". In: *Proceedings of the National Academy of Sciences USA* 104, pp. 12657–12662.

Kenneth G. Wilson (1971a). "Renormalization Group and Critical Phenomena. I. Renormalization Group and the Kadanoff Scaling Picture". In: *Physical Review B* 4 (9), pp. 3174–3183.

——— (1971b). "Renormalization Group and Critical Phenomena. II. Phase-Space Cell Analysis of Critical Behavior". In: *Physical Review B* 4 (9), pp. 3184–3205.

——— (1980). "Monte–Carlo Calculations for the Lattice Gauge Theory". In: *Recent Developments in Gauge Theories*. Ed. by Gerard 't Hooft. New York, NY: Springer.

Ulli Wolff (1989). "Collective Monte Carlo Updating for Spin Systems". In: *Physical Review Letters* 62 (4), pp. 361–364. DOI: 10.1103/PhysRevLett.62.361.

Zahari Zlatev, Jerzy Wasniewski and Kjeld Schaumburg (1982). "Comparison of two algorithms for solving large linear systems". In: *SIAM Journal of Scientific and Statistical Computing* 3 (4), pp. 486–501.

Вера Н. Кублановская (1961). "О некоторых алгорифмах для решения полной проблемы собственных значений". Рус. В: *Журнал вычис-*

лительной математики и математической физики 1 (4), с. 555—570.

Александр А. Мигдал (1975). “Фазовые переходы в калибровочных и спиновых решеточных моделях”. Рус. В: *Журнал Экспериментальной и Теоретической Физики* 69 (4), с. 1457—1465.

Part III

APPENDICES

A

CALCULATION OF THE EXACT RENORMALIZATION OF THE ISING MODEL IN ONE DIMENSION

As in Chapter 2, we begin with the Ising model defined on a periodic chain of length $n = 2^m$, with $m \geq 2$. The spins $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$, with $x_i \in \{-1, 1\}$, have a probability distribution

$$P(\boldsymbol{x}) = \frac{1}{Z(\mu)} \exp\left[\frac{\mu}{2} \sum_{i=1}^{n} x_i \left(x_{i-1} + x_{i+1}\right)\right], \tag{A.1}$$

where $\mu = 1/T$ and $Z(\mu)$ are the inverse temperature and the partition function. Split the variables in $\boldsymbol{x}$ by putting the even-index variables into $\tilde{\boldsymbol{x}}$ and odd-index into $\hat{\boldsymbol{x}}$,

$$\tilde{\boldsymbol{x}} = \{x_2, x_4, x_6, \dots, x_n\} \quad \text{and} \quad \hat{\boldsymbol{x}} = \{x_1, x_3, x_5, \dots, x_{n-1}\}. \tag{A.2}$$

To obtain the behavior of the coarse lattice $\hat{\boldsymbol{x}}$ we define the marginal probability of $\hat{\boldsymbol{x}}$ as an integral of the joint probability $P(\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}})$ over $\tilde{\boldsymbol{x}}$,

$$P(\hat{\boldsymbol{x}}) = \int P(\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}}) d\tilde{\boldsymbol{x}}. \tag{A.3}$$

The variables are discrete, thus the integral becomes a sum. We rewrite the probability as

$$P(\boldsymbol{x}) = \frac{1}{Z(\mu)} \exp\left[\frac{\mu}{2} \sum_{i=1}^{n} x_i \left(x_{i-1} + x_{i+1}\right)\right] \tag{A.4}$$

$$= \frac{1}{Z(\mu)} \prod_{x_i \in \tilde{\boldsymbol{x}}} \exp\left[\mu x_i \left(x_{i-1} + x_{i+1}\right)\right]. \tag{A.5}$$

Performing the sum over $\tilde{\boldsymbol{x}} \in \Omega = \{-1, 1\}^n$, we obtain

$$P(\hat{\boldsymbol{x}}) = \sum_{\tilde{\boldsymbol{x}}} P(\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}}) \tag{A.6}$$

$$= \frac{1}{Z(\mu)} \sum_{\tilde{\boldsymbol{x}}} \prod_{x_i \in \tilde{\boldsymbol{x}}} \exp\left[\mu x_i \left(x_{i-1} + x_{i+1}\right)\right] \tag{A.7}$$

$$= \frac{1}{Z(\mu)} \prod_{x_i \in \tilde{\boldsymbol{x}}} \left[\exp\left[\mu\left(x_{i-1} + x_{i+1}\right)\right] + \exp\left[-\mu\left(x_{i-1} + x_{i+1}\right)\right]\right] \tag{A.8}$$

$$= \frac{1}{Z(\mu)} \prod_{x_i \in \tilde{\boldsymbol{x}}} 2\cosh\left[\mu\left(x_{i-1} + x_{i+1}\right)\right] \tag{A.9}$$

$$= \frac{2^{n/2}}{Z(\mu)} \prod_{x_i \in \tilde{\boldsymbol{x}}} \cosh\left[\mu\left(x_{i-1} + x_{i+1}\right)\right]. \tag{A.10}$$

The transition between Eq. A.7 to A.8 can be seen clearly in a simpler example:

$$\sum_{\boldsymbol{x}} \prod_{i=1}^{3} f_i(x_i) = \sum_{x_1}\sum_{x_2}\sum_{x_3} f_1(x_1) f_2(x_2) f_3(x_3) \tag{A.11}$$

$$= \sum_{x_1}\sum_{x_2} f_1(x_1) f_2(x_2) f_3(-1) + \sum_{x_1}\sum_{x_2} f_2(x_1) f_2(x_2) f_3(-1) \tag{A.12}$$

$$= \sum_{x_1}\sum_{x_2} f_1(x_1) f_2(x_2) \left[f_3(-1) + f_3(1)\right] \tag{A.13}$$

$$= \sum_{x_1} f_1(x_1) \left[f_2(-1) + f_2(1)\right] \left[f_3(-1) + f_3(1)\right] \tag{A.14}$$

$$= \left[f_1(-1) + f_1(1)\right] \left[f_2(-1) + f_2(1)\right] \left[f_3(-1) + f_3(1)\right] \tag{A.15}$$

$$= \prod_{i=1}^{3} \left[f_i(-1) + f_i(1)\right]. \tag{A.16}$$

Unfortunately, the final result in Eq. A.10 is not in the same form as Eq. A.1. However, because the formulas must only agree at discrete values $x_i = \pm 1$, we may attempt to write

$$C \exp\left(\hat{\mu} x_{i-1} x_{i+1}\right) = \cosh\left(\mu\left(x_{i-1} + x_{i+1}\right)\right) \tag{A.17}$$

and choose the values of $\hat{\mu}$ and C to ensure the two functions are equal at all possible combinations of x_{i-1} and x_{i+1}. There are four combinations,

(x_{i-1}, x_{i+1})	$\cosh(\mu(x_{i-1} + x_{i+1}))$	$C\exp(\hat{\mu} x_{i-1} x_{i+1})$
$(-1,-1)$	$\cosh(2\mu)$	$C\exp(\hat{\mu})$
$(-1,1)$	1	$C\exp(-\hat{\mu})$
$(1,-1)$	1	$C\exp(-\hat{\mu})$
$(1,1)$	$\cosh(2\mu)$	$C\exp(\hat{\mu})$

where the symmetry $\cosh(-x) = \cosh(x)$ and value $\cosh(0) = 1$ are used. From the middle equations we obtain $C = \exp(\hat{\mu})$, while the remaining equation gives

$$\exp(2\hat{\mu}) = \cosh(2\mu) \quad \Rightarrow \quad \hat{\mu} = {}^1\!/_2 \ln[\cosh(2\mu)], \tag{A.18}$$

the classical result quoted in Chapter 2.