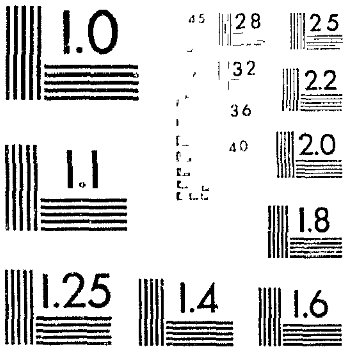
PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT



SYMBOLIC SOLUTION AND MICROCANONICAL SIMULATION OF THE POTTS MODEL

by

Peter Frempong-Mireku

submitted in Partial Fulfilment of the Requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
July, 1994

ISBN 0-612-01204-2

Name Peter Frempong-Mireku

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Mathematics | 0 | 4 | 0 | 5 | U·M·I

SUBJECT TERM — SUBJECT CODE

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture	0729
Art History	0377
Cinema	0900
Dance	0378
Fine Arts	0357
Information Science	0723
Journalism	0391
Library Science	0399
Mass Communications	0708
Music	0413
Speech Communication	0459
Theater	0465

EDUCATION

General	0515
Administration	0514
Adult and Continuing	0516
Agricultural	0517
Art	0273
Bilingual and Multicultural	0282
Business	0688
Community College	0275
Curriculum and Instruction	0727
Early Childhood	0518
Elementary	0524
Finance	0277
Guidance and Counseling	0519
Health	0680
Higher	0745
History of	0520
Home Economics	0278
Industrial	0521
Language and Literature	0279
Mathematics	0280
Music	0522
Philosophy of	0998
Physical	0523
Psychology	0525
Reading	0535
Religious	0527
Sciences	0714
Secondary	0533
Social Sciences	0534
Sociology of	0340
Special	0529
Teacher Training	0530
Technology	0710
Tests and Measurements	0288
Vocational	0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language	
General	0679
Ancient	0289
Linguistics	0290
Modern	0291
Literature	
General	0401
Classical	0294
Comparative	0295
Medieval	0297
Modern	0298
African	0316
American	0591
Asian	0305
Canadian (English)	0352
Canadian (French)	0355
English	0593
Germanic	0311
Latin American	0312
Middle Eastern	0315
Romance	0313
Slavic and East European	0314

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy	0422
Religion	
General	0318
Biblical Studies	0321
Clergy	0319
History of	0320
Philosophy of	0322
Theology	0469

SOCIAL SCIENCES

American Studies	0323
Anthropology	
Archaeology	0324
Cultural	0326
Physical	0327
Business Administration	
General	0310
Accounting	0272
Banking	0770
Management	0454
Marketing	0338
Canadian Studies	0385
Economics	
General	0501
Agricultural	0503
Commerce-Business	0505
Finance	0508
History	0509
Labor	0510
Theory	0511
Folklore	0358
Geography	0366
Gerontology	0351
History	
General	0578
Ancient	0579
Medieval	0581
Modern	0582
Black	0328
African	0331
Asia, Australia and Oceania	0332
Canadian	0334
European	0335
Latin American	0336
Middle Eastern	0333
United States	0337
History of Science	0585
Law	0398
Political Science	
General	0615
International Law and Relations	0616
Public Administration	0617
Recreation	0814
Social Work	0452
Sociology	
General	0626
Criminology and Penology	0627
Demography	0938
Ethnic and Racial Studies	0631
Individual and Family Studies	0628
Industrial and Labor Relations	0629
Public and Social Welfare	0630
Social Structure and Development	0700
Theory and Methods	0344
Transportation	0709
Urban and Regional Planning	0999
Women's Studies	0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture	
General	0473
Agronomy	0285
Animal Culture and Nutrition	0475
Animal Pathology	0476
Food Science and Technology	0359
Forestry and Wildlife	0478
Plant Culture	0479
Plant Pathology	0480
Plant Physiology	0817
Range Management	0777
Wood Technology	0746
Biology	
General	0306
Anatomy	0287
Biostatistics	0308
Botany	0309
Cell	0379
Ecology	0329
Entomology	0353
Genetics	0369
Limnology	0793
Microbiology	0410
Molecular	0307
Neuroscience	0317
Oceanography	0416
Physiology	0433
Radiation	0821
Veterinary Science	0778
Zoology	0472
Biophysics	
General	0786
Medical	0760

EARTH SCIENCES

Biogeochemistry	0425
Geochemistry	0996
Geodesy	0370
Geology	0372
Geophysics	0373
Hydrology	0388
Mineralogy	0411
Paleobotany	0345
Paleoecology	0426
Paleontology	0418
Paleozoology	0985
Palynology	0427
Physical Geography	0368
Physical Oceanography	0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences	0768
Health Sciences	
General	0566
Audiology	0300
Chemotherapy	0992
Dentistry	0567
Education	0350
Hospital Management	0769
Human Development	0758
Immunology	0982
Medicine and Surgery	0564
Mental Health	0347
Nursing	0569
Nutrition	0570
Obstetrics and Gynecology	0380
Occupational Health and Therapy	0354
Ophthalmology	0381
Pathology	0571
Pharmacology	0419
Pharmacy	0572
Physical Therapy	0382
Public Health	0573
Radiology	0574
Recreation	0575
Speech Pathology	0460
Toxicology	0383
Home Economics	0386

PHYSICAL SCIENCES

Pure Sciences

Chemistry	
General	0485
Agricultural	0749
Analytical	0486
Biochemistry	0487
Inorganic	0488
Nuclear	0738
Organic	0490
Pharmaceutical	0491
Physical	0494
Polymer	0495
Radiation	0754
✓ Mathematics	0405
Physics	
General	0605
Acoustics	0986
Astronomy and Astrophysics	0606
Atmospheric Science	0608
Atomic	0748
Electronics and Electricity	0607
Elementary Particles and High Energy	0798
Fluid and Plasma	0759
Molecular	0609
Nuclear	0610
Optics	0752
Radiation	0756
Solid State	0611
Statistics	0463

Applied Sciences

Applied Mechanics	0346
Computer Science	0984
Engineering	
General	0537
Aerospace	0538
Agricultural	0539
Automotive	0540
Biomedical	0541
Chemical	0542
Civil	0543
Electronics and Electrical	0544
Heat and Thermodynamics	0348
Hydraulic	0545
Industrial	0546
Marine	0547
Materials Science	0794
Mechanical	0548
Metallurgy	0743
Mining	0551
Nuclear	0552
Packaging	0549
Petroleum	0765
Sanitary and Municipal	0554
System Science	0790
Geotechnology	0428
Operations Research	0796
Plastics Technology	0795
Textile Technology	0994

PSYCHOLOGY

General	0621
Behavioral	0384
Clinical	0622
Developmental	0620
Experimental	0623
Industrial	0624
Personality	0625
Physiological	0989
Psychobiology	0349
Psychometrics	0632
Social	0451

DALHOUSIE UNIVERSITY

FACULTY OF GRADUATE STUDIES

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "Symbolic Solution and Microcanonical Simulation of the Potts Model"

by Peter Frempong-Mireku

in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated September 8, 1994

External Examiner [redacted]

Research Supervisor

Examining Committee

Dedicated to my Lord

Jesus Christ

and to my wife
Augustina Frempong-Mireku

TABLE OF CONTENTS

ILLUSTRATIONS AND TABLES

ABSTRACT

An analysis of the Ising-Potts model using symbolic computation is presented. The work considers the Kramers and Wannier V-matrix in two-dimensions and its extension to three-dimensions. Some of the properties of the V-matrices are also considered. The computation of the partition function of the Potts-Ising model is carried out using the perturbation theory. The computation of the partition function has been completed on a two-dimensional square net and on a three-dimensional cubic lattice as well. The eigenvectors needed to analyze the propagation of order in the crystal, and to compute long-range order in crystals have been given. An Onsager complete solution for the two-dimensional model has been incorporated as well as the two-dimensional n-state Potts model. A construction of symbolic proof that a order-disorder transition actually takes place in crystals has also been considered. The computation of the ground state entropy which provides a formal connection to the coloring of graphs has been examined.

The second part of the thesis examines the three-state Potts model on a three-dimensional cubic lattice. Using the microcanonical simulation method, the dynamic critical exponent z and the critical exponent v were measured to be $z = 2.11 \pm 0.05$ and $v = 0.613 \pm 0.005$ respectively. Also a general theorem for computing the average demon energy and an important consequence of the theorem has been presented.

ACKNOWLEDGEMENTS

Great thanks to the Lord Jesus Christ, my strength [Ps 91], who gave me wisdom to do this work and helped me through hard times, moreover to his powerful words which enabled me to stand through the storms of life to the completion of this work.

I would also like to express my sincere thanks to Dr. Peter Filmore, who was a great source of encouragement to me, and to all members of the Mathematics Department who helped me to complete this work. Especially to my Supervisors, Dr. K. J. M. Moriarty, and to Dr. T. Trappenberg and Dr. S. Sanielevici whose contribution and supervision I benefited greatly from, to them I extend my heartfelt appreciation.

Also my deepest thanks go to the Canadian Commonwealth, which granted me the scholarship to enable this work to be done.

Lastly, my sincere thanks to my wife and to all my Christian brothers who encouraged me through out all the work of the thesis.

Chapter 1

GENERAL INTRODUCTION

1.1 Introduction

For the Potts model, each site of a lattice can be in one of q distinct states. The Potts model is known to fall into the same universality class as many other statistical mechanics models and real physical systems. Many physicochemical systems can be approximated by a lattice arrangement of the molecules with nearest neighbor interactions. The Potts model is a generalization of the Ising model to more than two components. It has been the subject of increasingly intense research in recent years. It has some important applications, such as the calculation of hadronic properties in lattice gauge theory and phase transitions in condensed matter. It has also been related to models of percolation, absorption of rare gases on graphite, cubic ferromagnets, crystallographic transition, spin glasses and many other systems [1]. Because of the nature of the Potts models problem, it is almost impossible to find an analytic solution. In the absence of an analytic solution, perturbation theory is the method usually employed to deal with the problem. Important as the model is, there is no symbolic computation engine adequate to address the subject. The basic aim of this thesis is to address this problem, as well as providing the tools that will be necessary to back up the simulation methods that are already available.

The main parts of this thesis, symbolic solutions and microcanonical simulation of the Potts model, are covered in chapter three and chapter four, respectively. In chapter two, an overview of the microcanonical ensemble is given, preceded by a short historical review of the Potts model. This chapter also contains a description of a type of Monte Carlo sampling method which was first proposed by Metropolis et al. [2]. It allows for the evaluation of multidimensional integrals that arise in statistical mechanics. It is simple but powerful, and can be adapted to solve various simulation problems.

There are also a number of different techniques for the theoretical description of the two-dimensional ferromagnetic models. Commonly applied techniques are Monte Carlo simulation, series expansion, Monte Carlo renormalization, finite-size scaling and exact analytical treatments. It is clear from these methods that the phase transition between the ordered phase and the disordered phase occurs at a value of the coupling, K_c, which is given exactly by $\ln(\sqrt{q}+1)$ [3], for the simple quadratic q-state Potts model. Due to its popularity, the Monte Carlo simulation will be examined in more detail in chapter 2.

Chapter three is the core of the thesis. Its main aim is to present a symbolic solution of the Potts model. With more powerful symbolic computation systems emerging, great progress will be achieved in every area of science. Some complex integrals that required numerical computation because they were too tedious and difficult to integrate can now be evaluated symbolically, and the numerical value can easily be evaluated by substituting the numerical data. The basic symbolic algebra tool used in this thesis is Maple. Almost the entire program is written thanks to the powerful Maple computational engine. Mathematica also has been helpful in testing the temperature expression derived for use in the simulation algorithm. The task that is accomplished is the development of the symbolic series solution of the Potts model.

First a one-dimensional (1-d) case was tackled. The 1-d case is the simplest model and it exhibits no phase transition. This conclusion agrees with the known original work of Ising. The 1-d solution is based on a matrix approach. Despite the fact that the mathematics involved in the 1-d case is easy, it is a good start because it gives insight into the more complex mathematics involved in the 2-d and 3-d cases. The three different methods for the 2-d solution are based on the solutions of Onsager [4] and Kihara, Midzuno and Shizume [5], and on the perturbation method of Kramers and Wannier [6]. Kramers and Wannier developed an interesting matrix called the V-matrix, which I have implemented in my program. The 3-d construction of the series expansion uses an extended construction of the Kramers and Wannier V-matrix, which was developed in a paper by Oguchi [7]. The basic idea is the same for the 2-d and the 3-d construction except that the elements of the 3-d model are block matrices while those of the 2-d are not. The names of the programs that we have developed to compute the 2-d and 3-d Kramers and Wannier V-matrices and some important transformations are called *d2cvmat, d2lnvmat, d2upvmat, d2stmat, d3cvmat, d3lnvmat,* and *d3upvmat* [8]. This work also has an additional objective, the programs are designed to be a teaching tool, hence the basic matrices used in the construction can be accessed. The detailed construction of the V-matrix will be described in chapter three. The four functions we have built to compute the partition function of 2-d series expansion are *d2lnpps, d2lppps, d2rppps,* and *d2rnpps.* The four functions which we have developed to compute the partition function of 3-d series expansion are *d3lnpps, d3lppps, d3rppps,* and *d3rnpps.* Also the computation of 2-d lattices with many components, instead of the two component lattice considered by Kramers and Wannier, has been developed. The names of the programs are *d2mcltpf* and *d2mchtpf.* They compute the 2-d many-components low (high) temperature partition function in series. This function, created after the the work of Kihara, Midzuno and Shizume, is computed to only 16 terms.

1.2 Computers and Simulation

The technological advances in both hardware and software have enabled computers to be versatile and effective in dealing with many problems that were impossible to handle some decades ago. With the improved capabilities of current computers, the simulation of statistical systems defined by their Hamiltonian equations has become an essential component in the work of theoretical physicists and mathematicians. More than that, computer simulation is now an integral part of contemporary basic and applied science. Indeed, modern computers have become indispensable for progress in all scientific areas. By using fast computers, many integrals of statistical mechanics may be evaluated more accurately. Computers are becoming as important as the experiments to theoretical science. Hence the ability to compute is part of the essential repertoire of a research scientist. On the analytical front, the computer is used to calculate deterministic statistical mechanics equations to obtain numerical results, and these results are compared with known results, thereby evaluating the validity of the solution and the simplifying ideas underlying the initial equation. In this case, we aim at trying to expand the initial intractable integral in terms of some suitable parameter and some simpler integrals that can be evaluated on the computer. An example is the high-temperature series expansion. In chapter four, the microcanonical simulation of the Potts model will be considered. The microcanonical simulation is a fast and simple way to simulate statistical systems. It can easily be implemented on most general purpose computers at a reasonable speed. This enables us to obtain most of the important relational parameters. These parameters give us access to investigate some of the properties of the material under consideration. This will be fully expanded on in chapters three and four.

Another phenomenon that is of great interest to scientists are phase transitions. They are associated with abrupt changes, discontinuities and strong fluctuations. Such behavior is assumed to be due to interactions between microscopic constituents of matter. The number of constituents of a macroscopic system is generally of the order of 10^{23}. Each constituent will carry several degrees of freedom. So the theoretical description of phase transition, which takes into account the microscopic nature, will be very difficult. Statistical mechanics should give precise mathematical relations between the microscopic and macroscopic descriptions of physical systems. Thus it actually gives a theoretical foundation that describes the existence of this phenomenon.

The present thesis will simulate the three-state Potts model in order to obtain some basic properties of ferromagnets. In chapter 4, only the fundamental interactions between the microscopic components are considered. Only this fundamental interactions are taken into account within the microcanonical simulation. For this simulation, the question of using a computer to simulate directly the behavior of a physical system, taking as its starting point the fundamental equations of statistical mechanics, will be addressed. The simulation is carried out on a well-defined lattice system, and there is control over all the parameters. It allows for a new discovery that otherwise could not be inferred from the basic physical laws of interaction. Simulation is very important, because its results can be compared with experimental data as well as predictions of analytical theories. It illuminates and illustrates subtle basic conceptual relations in scientific reasoning that cannot easily be recognized or deduced from the basic physical laws. This program to simulate the model uses an algorithm similar to the work done by Drouffe and Moriarty [9]. The three-state Potts model is simulated, and it is used to identify the phase transition, and to measure critical exponents. Hereby the critical temperature T_c, the dynamic critical exponents z, and the critical exponent ν have been obtained. The relations between magnetization, temperature and energy are illustrated. Part of the work is the derivation of a general temperature expression for the simulation algorithm which uses the demon approach. Also I have derived an approximate or simpler temperature expression. This new expression gives almost the same results as those obtained using the full temperature expression. The result is given in Theorem 1 of chapter four.

The Maple work presented here will form part of the long-term strategy in developing a symbolic computation software for the Potts and Ising models.

Chapter 2

MICROCANONICAL ENSEMBLE

2.1 Introduction

Considerable efforts have been invested over recent years to develop numerical simulation methods and apply them to statistical systems. For some time the stochastic methods used in applications concerning spin or gauge systems have been Monte Carlo methods. However, in searching for a method that is relatively fast even for use on general purpose computers, we have settled on microcanonical simulations. This chapter presents a general overview of the microcanonical ensemble. In section 2.2, a brief historical review of the Potts model will be given. Then in section 2.3, some properties of statistical systems are examined, and in section 2.4, a general review of the canonical ensemble will be given. In section 2.5, only a basic idea of the canonical ensemble is presented. Lastly, in section

2.6, the chapter will be concluded with a type of Monte Carlo simulation generally called Metropolis algorithm. Monte Carlo simulation is essential because it gives a practical method of obtaining a representative sample of the total number of microstates. It is widely used in most statistical mechanics problems because of its simplicity and adaptability.

2.2 Brief Historical Review of the Potts Model

The history of the Potts model is directly linked with that of the Lenz-Ising model. Therefore a short introduction to the Lenz-Ising model will be appropriate here. Owing to the simplicity of the Ising model as an approximation of intermolecular forces, it was doubtful at first whether it could be made applicable to any real system. But now it is known that the essential features of cooperative phenomena depend on the mechanism of propagation of long-range order, especially at the critical point, rather than on the details of the intermolecular forces. Thus the Ising model offers, despite its simplicity, a satisfying representation of the mechanism. The importance of the model is that it can represent a wide class of physical systems: for example, glass-liquid critical phenomena, magnetic Curie points, order-disorder and transitions in alloys. All these can be described fairly well by the same model. Moreover, it has been used in a number of sciences, including physics, chemistry, metallurgy, mathematics and biology [1].

The model commonly referred to as Ising's was originally proposed by Ising's supervisor, Wilhelm Lenz, in 1920. The details of the model will be given in chapter four when the simulation of the three-state Potts model is examined. Lenz suggested that dipole atoms in crystals may be free to rotate about a free rest position in a lattice. In a paper published in 1925, Ising carried out an exact calculation of the partition function of a one-dimensional lattice. His work showed that there was no phase transition to a ferromagnetic ordered state at any temperature. Ising failed to predict phase transition in two and three dimensions.

This apparent failure caused Heisenberg to develop his theory to explain ferromagnetism. His theory deals with nearest neighbor interactions with more degrees of freedom between spins. Since then, there have been many other models. The details of all the various models cannot be given here, so a summary of the common ones is given. The appropriate Hamiltonian, which will be ideal for the work at hand, is given below:

$$\mathcal{H} = -\frac{1}{\sigma^2}\sum_{<ij>} J_{ij}\vec{\sigma_i}\vec{\sigma_j} - \frac{m}{\sigma}\sum_{i=1}^{N}\vec{\sigma_i}\vec{H}_i, \tag{2.1}$$

where J is the coupling constant, m the magnetic moments of the spin, H is the magnetic field, and the spins $\vec{\sigma_i}$ are D-dimensional vectors given by

$$\vec{\sigma_i} = \sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{iD}.$$

The summation is over all lattice points and all nearest neighbors $< ij >$. The interaction between the spins takes the form of a scalar product $\vec{\sigma_i}\vec{\sigma_j}$ and Eq. (2.1) defines the so called D-vector model. The spins $\vec{\sigma_i}$ can be treated as quantum mechanical operators or as classical vectors. In the case of a quantum mechanical operator, one obtains for example the quantum Heisenberg model ($D = 3$) and the quantum X-Y model ($D = 2$, spin-spin coupling along the z-axis being zero). When σ tends to infinity, so that the $\vec{\sigma}_i/\sigma$ becomes the classical D–dimensional unit vectors, one realizes the classical D-vector model. For special values of D, the classical vector model reduces to the following models:

1. $D = 1$, $\vec{\sigma}_i\vec{\sigma}_j = \sigma_i^z\sigma_j^z$	(Ising model)
2. $D = 2$, $\vec{\sigma}_i\vec{\sigma}_j = \sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y$	(planar or X-Y model)
3. $D = 3$, $\vec{\sigma}_i\vec{\sigma}_j = \sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y + \sigma_i^z\sigma_j^z$	(classical Heisenberg model)
4. $D = 0$	(extended volume problem)
5. $D = \infty$	(spherical model)

The variations that can be incorporated into the Hamiltonian include

(i) lattice anisotropy,

(ii) spin anisotropy, and

(iii) nearest neighbor interactions.

The problems (i) and (ii) can be tackled by developing series expansion around the isotropic case to compute the partition function. However, in this work, we will confine ourselves to (iii) above.

In an attempt to extend the exact results obtained for the standard Ising model in one and two dimensions, Potts (1952) [3] was led, at the suggestions of Domb, to introduce a model in which each site of a lattice can be in one of q distinct states. If $\sigma_k = 1, 2, \ldots, q$ characterizes the state of site k, then the Potts model Hamiltonian is given by

$$\mathcal{H} = -J\sum_{<ij>}\delta(\sigma_i, \sigma_j) - \sum_{\sigma=1}^{q}\sum_{j=1}^{N} h_\sigma\delta(\sigma, \sigma_j),$$

where δ is the Kronecker function and $h_\sigma, \sigma = 1, 2, \ldots, q$ is the field for the state σ. Two neighboring sites interact only if they are in the same state. The Ising model corresponds to the case $q = 2$. In this thesis the three-state Potts model without an external field, with the Hamiltonian

$$\mathcal{H} = -K \sum_{<ij>} \delta(\sigma_i, \sigma_j),$$

will be investigated. This will be further elaborated on in chapter 4.

2.3 Some Properties of Statistical Systems

The properties of a statistical system are governed by its Hamiltonian defined by its mechanical variables, here represented by Ω. The probability associated with each microstate can be expressed in terms of its canonical density function by

$$\rho(\Omega) = \frac{e^{-\mathcal{H}(\Omega)/k_B T}}{Z}, \tag{2.2}$$

where Z is the normalization factor or the partition function. The form of the partition function is given by

$$Z = \int_\Omega e^{-\mathcal{H}(\Omega)/k_B T} d\Omega, \tag{2.3}$$

where T is the absolute temperature and k_B is the Boltzmann constant. For a given probability distribution of the microstates, the thermodynamic value of a measurable physical quantity, $f(\Omega)$, is obtained in the canonical ensemble as

$$< f > = \int_\Omega f(\Omega)\rho(\Omega) d\Omega. \tag{2.4}$$

This equation constitutes the formal connection between the microscopic and macroscopic worlds. When the partition function exhibits singularities, the singularities can be associated with phase transitions. The phase transitions can be classified as continuous or first-order transitions using the free energy, which is defined as

$$F = -k_B T \ln Z. \tag{2.5}$$

If the first derivatives of F with respect to T are continuous at the singular point, then the phase transition is continuous; otherwise, it is a first-order transition. A phase transition is characterized by a spontaneously broken symmetry. Symmetry-breaking is described in terms of the order parameter, Φ. We define the order parameter by

$$\Phi = \left(-\frac{\partial F}{\partial h_\Phi}\right)_T,$$

where h_Φ is a thermodynamic conjugate field associated with the order parameter. The corresponding term in the free energy is $-h_\Phi \Phi$. By Fisher's classification of phase transitions [10], Φ is discontinuous at a first-order transition and goes to zero continuously at a continuous transition (critical point). The behavior of the order parameter in computer simulation is therefore often used in order to determine the nature of the phase transition [11].

Another important quantity that is influenced by critical fluctuations is defined below:

Definition 1 *The isothermal ordering susceptibility* χ_Φ *is defined by*

$$\chi_\Phi = (\partial^2 F/\partial {h_\Phi}^2)_T = (\partial \Phi/\partial h_\Phi)_T.$$

If we introduce the order parameter $< \Phi >$, then we have

$$\chi_\Phi = (k_B T)^{-1}(< \Phi^2 > - < \Phi >^2). \tag{2.6}$$

This is significant for computer simulations because it implies that the response function may be evaluated directly from the thermal fluctuations. The fluctuations theorem also relates the fluctuations in internal energy to the specific heat as

$$C_h = (\delta E/\delta T)_h = (k_B T^2)^{-1}(< H^2 > - < H >^2). \tag{2.7}$$

Wisdom (1965) [12] was the first to analyze the scaling hypothesis for static critical phenomena. By this hypothesis the singular part, $\tilde{F}$, of the free energy is a generalized homogeneous function

$$\tilde{F}(\lambda^a t, \lambda^b h_\Phi) = \lambda \tilde{F}(t, h_\Phi), \tag{2.8}$$

where $t = (T - T_c)/T_c$ is a measure of the relative distance from the critical temperature T_c. Taking the derivative of $\tilde{F}$, with respect to h_Φ, the corresponding singular behavior of the order parameter can be obtained. We find

$$\Phi(t, h = 0) \sim (-t)^\beta, \tag{2.9}$$

with the critical exponents $\beta = (1 - b)/a$. Power-law singularities can be derived for other quantities as follows:

$$\begin{aligned} C_h(t) &\sim (-t)^{-\alpha}, \; T > T_c \\ &\sim (-t)^{-\alpha'}, \; T < T_c \end{aligned} \tag{2.10}$$

$$\begin{aligned} \chi_\phi(t) &\sim (-t)^{-\gamma}, \; T > T_c \\ &\sim (-t)^{-\gamma'}, \; T < T_c. \end{aligned} \tag{2.11}$$

According to Stanley [13], scaling implies that not all critical exponents are independent. For example a and b in Eq. (2.8) are related to the critical exponent β. This two-factor scale invariance could be expressed in terms of the scaling relation, which can be expressed as

$$\begin{aligned} \alpha' + 2\beta + \gamma' &= 2 \\ \alpha &= \alpha' \\ \gamma &= \gamma'. \end{aligned}$$

Corrections to scaling are important. The order parameter has the form

$$\Phi(t) \simeq B(-t)^{\beta}(1 + \sum_{i=1} a_i |t|^{\theta_i}), \qquad (2.12)$$

where θ_i are correction-to-scaling exponents.

The universality principle states that continuous phase transitions can be classified in a few universality classes, each class giving rise to a certain set of exponents. These classes are determined by a few fundamental properties of the systems, such as spatial dimension (d), range of interaction, symmetry, and degree of freedom of the fields. The physical idea behind the universality principle is that at the critical point, all details of the microscopic interactions are overshadowed by the long wavelength fluctuations. Renormalization group theory provides the mathematical explanation for the concept of universality.

2.4 Microcanonical Ensemble

The notion of what a microstate or configuration is, is contained in the following definition:

Definition 2 *A microstate, or configuration, of a system is defined to be a set of variables Ω, which contains the values of all possible degrees of freedom for each* *particle of the system. The phase space is defined to be the space spanned by all possible microstates of a system.*

Examples of microstates are spatial position, velocity, and magnetic moment. Consider a closed system in which the volume V, the number of particles N, and the energy E are fixed. Also assume that the system is isolated; that is, the influences of external parameters such as gravitational force and external magnetic fields are ignored. Generally, closed macroscopic systems tend to a time-independent state of equilibrium of maximum entropy. The macrostate of the system is specified by the values V, N, and E. At the macroscopic level, there are many different ways or configurations in which the macrostate comprising of V, N, and E can be realized. A particular configuration or microstate is accessible if its properties are consistent with the known macroscopic quantities of the system. At any given time, the system is equally likely to be in any of its accessible microstates. Let an isolated system have an accessible macrostate Ω, then the probability P_s of finding the system in microstates s is

$$P_s = \begin{cases} \frac{1}{\Omega}, & \text{if } s \text{ is accessible} \\ 0, & \text{otherwise,} \end{cases} \tag{2.13}$$

where $\sum_{s \epsilon \Omega} P_s = 1$.

In the laboratory, the physical quantities are measured over a span of time sufficiently long to allow the system to sample a large number of its accessible microstates. The meaning of the probabilities in Eq. (2.13) that is consistent with such a time average is that during a sequence of observations, P_s yields the fraction of times that a single system is found in a given microstate. Normally, measurements are not done on a single system. Consider a collection of systems that are identical replicas characterized by the same macrostate. Then Eq (2.13) describes an ensemble of identical systems. With these basic concepts outlined, the definition of microcanonical ensemble follows:

Definition 3 *An ensemble of systems specified by E, N, V that is described by the probability distribution of the form*

$$P_s = \begin{cases} \frac{1}{\Omega}, & \textit{if } s \textit{ is accessible} \\ 0, & \textit{otherwise}, \end{cases}$$

is called a microcanonical ensemble.

As an illustration of the above definition, consider a physical quantity A, which has the value A_s when the system is in microstate s. Then the ensemble average of A is given by

$$< A > = \sum_s A_s P_s,$$

where P_s is given by Eq. (2.13). We clarify the above definitions with an example.

Example 1 *Let a model consist of N distinguishable particles that are noninteracting, and have velocities v and $-v$ only. The ensemble of configurations consistent with $N = 5$ and $E = \frac{5}{2}v^2$ with a unit mass is described by the binomial coefficients $\binom{N}{i}$.*

Then it is not difficult to calculate the ensemble averages for the physical system. For example, the mean number of particles moving in the right is

$$< n >= \frac{1}{2^N} \sum_{i=1}^{N} i \binom{N}{i}.$$

2.5 Canonical Ensemble

Most physical systems are not isolated but exchange energy with their environment. Such systems are usually small in comparison with their environment. We can therefore assume that a change in energy in the smaller system does not have a significant effect on the temperature of the larger system. Thus the larger system acts as a heat bath at a fixed absolute temperature T. If a small but microscopic system is placed in thermal contact with a heat bath, the system reaches thermal equilibrium by exchanging energy with the heat bath, until the system attains the temperature of the bath. Let us imagine an infinitely large number of copies of the system and the heat bath. The probability, P_s, that the system is in microstate s with energy E_s is given by

$$P_s = \frac{1}{Z} e^{-E_s/k_B T}, \tag{2.14}$$

where Z is the normalization constant. The ensemble defined by Eq. (2.14) is known as the canonical ensemble. Taking note of the fact that $\sum P_s = 1$, Z can be obtained as

$$Z = \sum_{s=1}^{M} e^{-E_s/k_B T}. \tag{2.15}$$

The sum in Eq. (2.15) is over all M microstates of the system. The quantity Z is known as the partition function of the system. We can use Eq. (2.14) to obtain the ensemble average of physical quantities; for example, the mean energy is given by

$$< E >= \sum_s E_s P_s = \sum_s E_s e^{-\beta E_s}. \tag{2.16}$$

In contrast to the microcanonical ensemble which is specified by E, N, and V, with the probability distribution given in Eq. (2.13), the canonical ensemble is specified by T, N, and V, with the probability distribution given by the Eq. (2.14). In the canonical ensemble the energy can fluctuate, while in the microcanonical ensemble the energy is fixed. In the microcanonical ensemble the temperature can fluctuate, while in the canonical ensemble the temperature is fixed.

Monte Carlo simulations that shall evaluate properties of physical systems, the canonical distribution function is used quite often. In section 2.6, we give a detailed description of the method.

2.6 The Monte Carlo Simulation Method

Our prime objective here is to give the solid mathematical theory on which the Monte Carlo method is based. The approach will not attack any particular problem. But most problems can easily be adapted to the method. Some important phenomena and properties relating to the method will also be examined. Numerical simulation methods are concerned with procedures based on random numbers. Basically one has to generate and process a large number of random numbers. Hence the appearance of modern computers has stimulated the use of the method. The Monte Carlo method is extremely useful as a mathematical tool for solving numerically problems that are too complex to be solved analytically. The problems attacked by the method normally fall into two classes: probabilistic and deterministic. The probabilistic problems are solved by simulating directly the random process inherent in the problem. Some examples are the simulation of neutron diffusion in reactors and the simulation of random fluctuations on a telephone network. The deterministic problem is transformed to one of stochastic nature, with the requirement that the original problem and the transformed one have solutions that differ by a controlled amount. An example of such a problem is that of multidimensional integrals in many-body theory. The Monte Carlo method can be tailored to meet one's needs depending on the nature of the problem. Different Monte Carlo methods can be distinguished by the sampling techniques they employ. The sampling technique also depends on the bias imposed on the sampling scheme. In importance sampling, statistical information is collected according to its importance for a particular problem. A kind of Monte Carlo importance sampling method first proposed by Metropolis et al. (1953) [2] has proved to be very successful in statistical mechar ics. Though the method and its realization are extremely simple, it has proved to be very powerful. The method can be introduced to achieve an efficient numerical evaluation of the multidimensional integral Eq. (2.4) in many-body systems.

For a physical system with a known Hamiltonian, the energy of a given microstate $E_i = \mathcal{H}(\Omega_i)$ can be easily computed. Then Eq. (2.4) determines the properties of the system in thermodynamic equilibrium. Assuming a discrete phase space $\{\Omega_i\}_{i=1}^{M}$, the fundamental mathematical object Eq. (2.4) can be written as

$$< f > = \sum_i f(\Omega_i)\rho(\Omega_i). \qquad (2.17)$$

Th3 idea is to introduce stochastic elements into the computation. The direct attack of introducing unbiased choice will be quite unfavorable since the Boltzmann weights vary by many orders of magnitude in the neighborhood of the phase transitions. So the best method is to impose a bias and choose a finite set of points, $\{\Omega_i\}_{i=1}^{M}$, with each state Ω_i having a frequency proportional to the Boltzmann probability, Eq. (2.2). Thus the microstates are sampled according to their importance, and the estimate

$$< f > = M^{-1} \sum_{i=1}^{M} f(\Omega_i) \tag{2.18}$$

is obtained. This partition set, $\{\Omega_i\}_{i=1}^{M}$, is generated by setting a random walk in the phase space. A Markov chain is associated with the discrete set that can be parameterized by a discrete time parameter t, the Markov time. The sequence $\Omega_1, \Omega_2, ..., \Omega_M$ of states is a realization of a Markov chain determined by the initial configuration Ω_1 and a stochastic matrix P with elements p_{ij} where $1 \leq i, j \leq M$, which are subject to the conditions

$$\forall i, j : p_{ij} \geq 0, \ \ \forall i : \sum_{i=1}^{M} p_{ij} = 1. \tag{2.19}$$

Here the p_{ij} is the conditional probability for the one-step transition $\Omega_i \rightarrow \Omega_j$ in the chain.

Confining oneself to homogeneous Markov chains, defined by time-independent transition probabilities, the configurational average in Eq. (2.17) is replaced by the time average in Eq. (2.18). The matrix P must be constructed to make the limit distribution of the chain π_j equal to $\rho(\Omega_j)$. The n-step transition probabilities $p_{ij}^{(n)}$ are recursively given as

$$\forall i, j : p_{ij}^{(n)} = \sum_{k=1}^{M} p_{ik}^{(n-1)} p_{kj}; \quad p_{ij}^{(1)} = p_{ij}. \tag{2.20}$$

The above condition can be formulated as

$$\forall j : \lim_{n \rightarrow \infty} p_{ij}^{(n)} = \pi_j > 0. \tag{2.21}$$

By the homogeneous Markov chains theory the limit exists if the chain is irreducible, i.e, when all elements are in the same ergodic class. The assumption that two averages yield equivalent results is called the ergodic hypothesis or, more accurately the quasi-ergodic hypothesis. The limit distribution is then independent of j, and the initial state $t = 1$. It is determined uniquely by the normalization and steady state conditions

$$\sum_{j=1}^{M} \pi_j = 1, \tag{2.22}$$

$$\forall j : \ \pi_j = \sum_{i=1}^{M} \pi_i p_{ij}. \tag{2.23}$$

This condition is usually fulfilled by imposing a stronger requirement of microscopic irreversibility:

$$\forall j, k : \pi_j p_{jk} = \pi_k p_{kj}. \tag{2.24}$$

The above conditions on the stochastic matrix P imply $2M$ linear equations. This gives considerable freedom to choose the $M \times M$ matrix elements. A much simpler method has been suggested by Metropolis et al. (1953) [2] for choosing P^* with elements p^*_{ij} with the associated irreducible ergodic Markov chain. In terms of P^*, the matrix P is defined as

$$p_{jk} = p^*_{ij}, \quad \pi_j/\pi_i \geq 1$$

$$p_{jk} = p^*_{ij}\pi_j/\pi_i, \quad \pi_j/\pi_i < 1 \tag{2.25}$$

$$p_{ii} = 1 - \sum_{k(\neq i)}^{M} p_{ik}. \tag{2.26}$$

The presentation of the Monte Carlo method in terms of a Markov chain with a time-parameter makes it easy to visualize that the process of generating a chain of microstates may be given a dynamical interpretation.

The Markov process thus described above is governed by the equation

$$\frac{d\pi_i(t)}{dt} = -\sum_j p_{ij}\pi_i(t) + \sum_j p_{ji}\pi_j(t).$$

It may be noted that in thermal equilibrium, $d\pi_i(t)/dt = 0$ and $\lim_{t\to\infty} \pi_i(t) = \rho(\Omega_i)$. This dynamics is not the true physical dynamics of the system because the actual equation of motion has not been used. However, in many cases the time evrlution given by the Markov chain closely resembles the kinematic of the approach to thermodynamic equilibrium [14-20]. It must also be noted that time average and ensemble average coincide in the limit as $\lim_{t\to\infty}$ and $\lim_{n\to\infty}$ respectively.

At this point we are ready to describe a realization of the Monte Carlo importance sampling method. Let us consider a system of N particles with a microstate Ω where $\Omega = (m_1, m_2, .., m_N) = \{m_i\}_{i=1}^N$.

To present the general case normally encountered in the applications, to each single-particle state m there is an internal degeneracy D_m associated. The canonical density function Eq. (2.2) is then written as

$$\rho(\{m_i\}_{i=1}^N) = Z^{-1}(\prod_{j=1}^N D_{m_j})\exp[-H(\{m_i\}_{i=1}^N)]/k_BT], \tag{2.27}$$

where Z is the generalized partition function

$$Z = \sum_{\{m\}}(\prod_{j=1}^N D_{m_j})\exp[-H(\{m_i\}_{i=1}^N)]/k_BT].$$

For simplicity choose a P^* that corresponds to single site-transitions, $m_k \to m_k^{'}$. The description can be generalized to account for any combination of single-site excitations. For convenience, we introduce the internal energy change

$$\Delta U = H(\Omega') - H(\Omega), \tag{2.28}$$

and the internal entropy associated with the transition is as follows:

$$\Delta S = k_B \ln(D_{m_k'}/D_{m_k}). \tag{2.29}$$

A possible realization of the Markov chain may then be described by the following algorithm:

1. Choose an arbitrary (e.g., random) initial configuration, Ω_1.

2. Pick a trial state, Ω_2', according to the probability p_{12}^*.

3. If $\Delta U - T\Delta S \leq 0$, which is equivalent to $D_{m_k'}/D_{m_k} \exp(-\Delta U/k_BT) \geq 1$, the trial state is accepted as the next element in the chain $\Omega_2 = \Omega_2'$.

4. If $\Delta U - T\Delta S > 0$, i.e. $\lambda = D_{m_k'}/D_{m_k} \exp(-\Delta U/k_BT) < 1$, a random number $\eta \in [0,1]$ is generated. If $\lambda > \eta$, the trial state is accepted as the next element in the chain, $\Omega_2 = \Omega_2'$. If $\lambda < \eta$, the original state is duplicated, $\Omega_2 = \Omega_1$.

5. A new trial state, Ω_3', is considered.

The process is repeated until sufficient data have been generated. It is also assumed that the thermodynamic temperature is positive. If the ergodic requirement is satisfied, the procedure above will, in the limit of a large number of Monte Carlo steps, lead to a distribution of states given by the canonical distribution function Eq. (2.27). This limit distribution of states constitutes the equilibrium ensemble at temperature T for the model under consideration. In chapter 4, where our simulation procedure is described, a different method called microcanonical simulation is used. It will be shown that it is more efficient than the one just described. In the next chapter, the symbolic solution of the Potts-Ising model is tackled.

Chapter 3

SYMBOLIC SOLUTION OF THE POTTS MODEL

3.1 Introduction

The heart of this thesis is the symbolic solution of the Potts (Ising) model. The approach adopted demonstrates how the Potts model not only is widely used practically, but also has a rich mathematical representation. It will also provide a general solution to the long-standing problem of computing the coefficients of low-temperature series expansion of the Ising model. The high-temperature series is easily deduced from the same method; hence, it will not be provided in this work.

After the eigenvalue method in statistics is examined in section 3.2, it will be applied to the one-dimensional closed chain and the one-dimensional open chain in sections 3.3 and 3.4 respectively. In section 3.5 the famous Onsager solution [4] for the rectangular square net will be presented, followed by an application to large crystals. Then the construction of an algorithm that computes the partition function of the two-dimensional Ising net with examples based on the Onsager work will be given. Lastly, in sections 3.7 and 3.8, the computation of the partition function for the two-dimensional and the three-dimensional Ising model using the Kramers and Wannier approach will be presented. The chapter will close with the development of the algorithms and some examples.

The various aspects of mathematics that are relevant to the Potts (Ising) model are given below:

(a) group analysis

(b) series solution

(c) matrix or algebraic solution

(d) graph theory analysis

(e) combinatorial method

(f) C^*-algebra approach

This work will focus on the series and the matrix or algebraic solution of the Ising model. The next section begins by analyzing the eigenvalue method in statistics.

3.2 The Eigenvalue Method

In the treatment of statistical cooperation in crystals, the partition function evaluation of properties of statistical systems is simplified if the interacting units have a fixed location. Also, the periodicity of the systems allows a simplifying transformation. This was discovered independently by Kramers and Wannier [6], Lassettre and Howe [21] and Montroll and Mayer [22]. The transformation can be derived under the following general assumptions:

(a) The identical units in the structure are lined up as beads on a string

(b) Number them as $0, 1, 2, 3, \ldots n$

(c) Describe the state of each unit by the discrete or continuous variable $\sigma_1, \sigma_2, \ldots, \sigma_n$

(d) Represent the potential energy of the system by the sum of the interactions of each molecule with its nearest neighbors

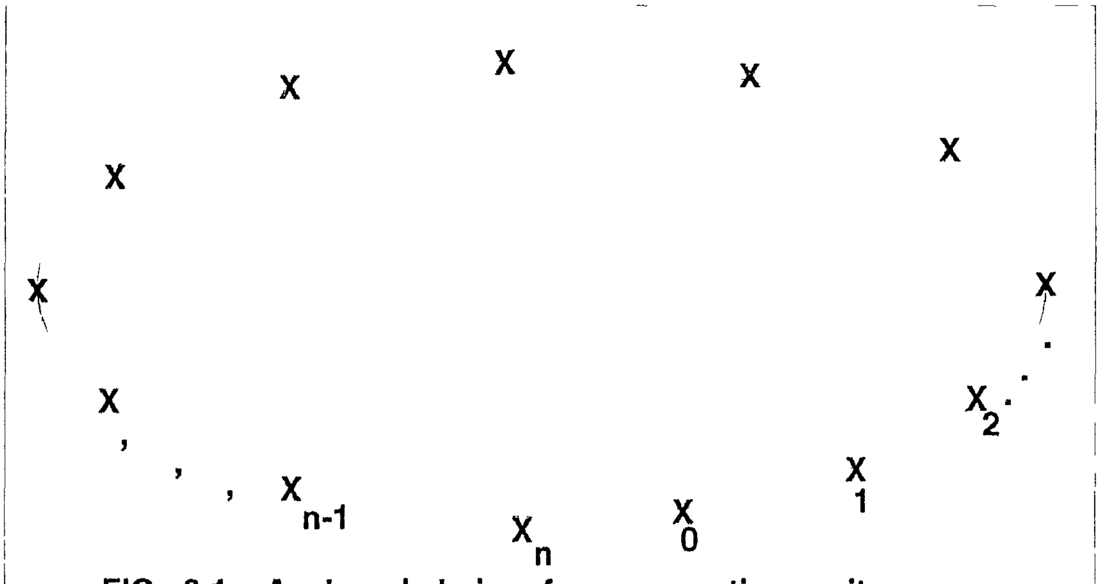


FIG. 3.1. A closed chain of *n* cooperating units.

The reason for (d) is that the number of molecu'es is large, so the intermolecular forces are sufficiently short-ranged to permit the total potential energy to be represented by the sum of the interactions of each molecule with its nearest neighbors. The interaction is symbolized by connecting lines as in Fig. 3.1. Let the interacting potential be denoted by $V(\sigma_i, \sigma_j)$. Then the probability for a given state of the assembly is proportional to the Boltzmann exponential

$$\exp[-\frac{1}{k_B T}(V(\sigma_1, \sigma_2) + V(\sigma_3, \sigma_4) + \ldots + V(\sigma_n, \sigma_1))].$$

The partition function can be formed by either summation or integration as

$$Z = \sum_{\alpha_1 = \pm 1} \cdots \sum_{\alpha_n = \pm 1} \exp\left[-\frac{1}{k_B T}(V(\sigma_1, \sigma_2) + \ldots + V(\sigma_n, \sigma_1))\right]. \tag{3.1}$$

If Z is known, one can obtain some of the physical properties of the crystals; for example, the energy U and the total magnetization M can be given in the form

$$U = k_B T^2 \frac{\partial \ln Z}{\partial T}$$

and

$$M = k_B T \frac{\partial \ln Z}{\partial H}, \tag{3.2}$$

where H is the applied magnetic field. From probability calculus [5] one can associate the following eigenvalue problem with Eq. (3.1),

$$\sum_{\sigma} \exp\left[-\frac{1}{k_B T} V(\sigma, \sigma')\right] A(\sigma) = \lambda A(\sigma'). \tag{3.3}$$

Here λ may have a series of different eigenvalues λ_ν. To each λ_r, there corresponds one eigenvector A_r if multiple values are counted as often as they arise. One can obtain an orthogonal relation between the A's which is given by

$$\sum_{\sigma} A_\mu(\sigma) A_\nu(\sigma) = \delta_{\mu\nu}. \tag{3.4}$$

For integral equations, the kernel can be developed in terms of the eigenvectors. This is also valid for the matrices below because the two sides have identical eigenvectors and eigenvalues. Thus the diagonalized expansion of $\exp[-V(\alpha,\alpha')/k_BT]$ is

$$\exp\left[-\frac{1}{k_BT}V(\sigma,\sigma')\right] = \sum_{\nu} \lambda_\nu A_\nu(\sigma) A_\nu(\sigma'). \tag{3.5}$$

Substitituting Eq. (3.5) in Eq. (3.1) and carrying out the summations over $\sigma_1, \sigma_2, ..., \sigma_n$ explicitly with the help of Eq. (3.4) one obtains

$$Z = \sum_{\nu} \lambda_\nu^n. \tag{3.6}$$

When the number of the interacting units n is large, all but the largest eigenvalue can be neglected. Hence Eq. (3.7) takes the form

$$Z = \lambda_{\max}^n. \tag{3.7}$$

In the same way in three dimensions $|A_{\max}(\sigma_i)|^2$ can be interpreted as the relative probability that any layer in the system is in the ith configuration, or the probability that the internal coordinates have a particular value, say σ_i. However, for a chain with free ends, $A_{\max}$ measures the probability for a state A_i in the end member. At a temperature of absolute zero, the components of $A_{\max}$ correspond to a completely ordered state with a value $1/\sqrt{n_0}$. The n_0 is the number of configurations possible with complete order, the other components being zero. At a temperature sufficiently high for thermal agitation to cause complete disorder, all configurations are equally probable and $A_{\max} \rightarrow (1, 1, ...1)/\sqrt{n}$. The above considerations are applied to the one-dimensional Ising model in the next section.

3.3 Transfer Matrix Solution of the Closed Chain

The one-dimensional Ising model is the simplest of the models. The approach here will be very similar to that described by Kramers and Wannier.

Definition 4 *The total energy E, for a closed Ising chain of N spins in an external magnetic field H, takes the form*

$$E = -\frac{1}{2}J\sum_{i}^{N}\sigma_i\sigma_{i+1} - mH\sum_{i}^{N}\sigma_i, \tag{3.8}$$

where m is the magnetic moment, J is the coupling constant between nearest neighbor spins or the energy of interaction between nearest neighbor spins, and $\sigma_i = \pm 1$.

As mentioned above, most statistical questions concerning Eq. (3.8) can be answered if the partition function

$$Z = \sum_{\{\sigma_i=\pm 1\}} \exp[-\frac{E}{kT}],$$

where σ_{N+1} is identified with σ_1, in order to get a closed chain. With

$$K = \frac{J}{2k_BT}$$

and

$$C = \frac{mH}{k_BT},$$

the partition function above can be expressed in the form

$$Z = \sum_{\{\sigma\}}\prod_{i=1}^{N} L(\sigma_i,\sigma_{i+1}), \tag{3.9}$$

where

$$L(\sigma_i,\sigma_{i+1}) = \exp[K\sigma_i\sigma_{i+1} + \frac{C}{2}(\sigma_i + \sigma_{i+1})]. \tag{3.10}$$

The sum over configurations in Eq. (3.9) has the form of a matrix product; i.e., after summing over $\sigma_2 = \pm 1, \sigma_3 = \pm 1, ...\sigma_N = \pm 1$, we have

$$Z = \sum_{\sigma_1 = \pm 1} L^N(\sigma_1, \sigma_1), \tag{3.11}$$

where $L^N(\sigma, \sigma')$ denotes the element of the matrix **L**, with the components (3.10) raised to the Nth power. In matrix form we write

$$\mathbf{L} = \begin{pmatrix} L(+1,+1) & L(+1,-1) \\ L(-1,+1) & L(-1,-1) \end{pmatrix} = \begin{pmatrix} e^{K+C} & e^{-K} \\ e^{-K} & e^{K-C} \end{pmatrix}.$$

The eigenvalue method will be applied to evaluate the partition function Z. Assume we have a linear chain of finite length $n+1$. Let $A_1 = \{\sigma_i\}$, $i = 0 \ldots n-1$ be any arrangement with n spins. Then by Boltzmann's theorem

$$P(A_1) \propto \exp(-E/k_BT) = \exp[K(\sigma_0\sigma_1 + ... + \sigma_{n-2}\sigma_{n-1}) + C(\sigma_0 + ... + \sigma_{n-1})], \tag{3.12}$$

because each arrangement has a weight of 1, and E is the total energy. Also let $A_2 = \{\sigma_i\}$, $i = 0, \ldots, n$ be the arrangement of adding the $n+1$-th spin, then

$$P(A_2) - P(A_1) = \exp[K\sigma_{n-1}\sigma_n + C\sigma_n].$$

From Eq. (3.12) the probability $P(\sigma_{n-1})$ that the σ_{n-1} has either values irrespective of the value of $\sigma_0, \sigma_1, ..., \sigma_{n-2}$ is

$$P(\sigma_{n-1}) \propto \sum_{\sigma_0, ..., \sigma_{n-2} = \pm 1} \exp[K(\sigma_0\sigma_1 + ... + \sigma_{n-2}\sigma_{n-1}) + C(\sigma_0 + ... + \sigma_{n-1})]$$

also

$$P(\sigma_{n-1}, \sigma_n) \propto \sum_{\sigma_0, ..., \sigma_{n-2} = \pm 1} \exp[K(\sigma_0\sigma_1 + ... + \sigma_{n-2}\sigma_{n-1} + \sigma_{n-1}\sigma_n) + C(\sigma_0 + ... + \sigma_n)]$$

Thus

$$\lambda P(\sigma_{n-1}, \sigma_n) = P(\sigma_{n-1}) \exp[K\sigma_{n-1}\sigma_n + C\sigma_n]. \tag{3.13}$$

The factor λ is present because Boltzmann exponentials are only proportional to the probabilities. Assume the chain is very long, then

$$\sum_{\sigma_{n-1}=\pm1} \lambda P(\sigma_{n-1},\sigma_n) = \lambda P(\sigma_n) = \sum_{\sigma_{n-1}=\pm1} P(\sigma_{n-1}) \exp[K\sigma_{n-1}\sigma_n + C'\sigma_n].$$

The matrix equations have the form of matrix eigenvalue problems. The matrix can be symmetrized by the substitution

$$A(\sigma) = P(\sigma)\exp(\frac{1}{2}C'\sigma),$$

which reduces the problem to the form

$$\sum_{\sigma'=\pm1} L(\sigma,\sigma')A(\sigma') = \lambda A(\sigma), \qquad (3.14)$$

where

$$L(\sigma,\sigma') = \exp[K\sigma\sigma' + \frac{1}{2}C'\sigma + \frac{1}{2}C'\sigma'].$$

Using the theorem that develops any matrix into eigenvectors, the matrix Eq. (3.14) can be written as

$$L(\sigma_1,\sigma_2) = \lambda_1 A_1(\sigma_1)A_1(\sigma_2) + \lambda_2 A_2(\sigma_1)A_2(\sigma_2),$$

where $A_1(\sigma)$ and $A_2(\sigma)$ are the eigenvectors belonging to λ_1 and λ_2, respectively. Since they are orthogonal and may be assumed to be normalized,

$$\sum_{\sigma=\pm1} A_i(\sigma)A_k(\sigma) = \delta_{ik}.$$

Thus we can unite N of the L's to

$$\sum_{\sigma_2}\cdots\sum_{\sigma_N} L(\sigma_1\sigma_2)L(\sigma_2\sigma_3)\ldots L(\sigma_N\sigma_{N+1}) = \lambda_1^N A_1(\sigma_1)A_1(\sigma_{N+1}) + \lambda_2^N A_2(\sigma_1)A_2(\sigma_{N+1}).$$

Now, since we are considering a closed ring as in Fig. 3.1, by setting $\sigma_{N+1} = \sigma_1$ and summing over the last spin, we get

$$\sum_{\sigma_1}\cdots\sum_{\sigma_N} L(\sigma_1,\sigma_2)L(\sigma_2,\sigma_3)\ldots L(\sigma_N,\sigma_1) = \lambda_1^N + \lambda_2^N.$$

The left-hand side is exactly the partition function Z, which from Eq. (3.11) can be identified as the trace of L^N, and hence

$$Z = \lambda_1^N + \lambda_2^N. \tag{3.15}$$

If the length N of the chain tends to infinity, the smaller root λ_2 may be neglected. The values λ_1 and λ_2 can easily be computed. By taking note of the matrix L, the eigenvalue problem can be written as

$$\begin{pmatrix} e^{K+C} & e^{-K} \\ e^{-K} & e^{K-C} \end{pmatrix} \begin{pmatrix} A(+) \\ A(-) \end{pmatrix} = \lambda \begin{pmatrix} A(+) \\ A(-) \end{pmatrix},$$

which can also be given by the eigenvalue equation

$$\det(\mathbf{L} - \lambda \mathbf{I}) = \lambda^2 - 2\lambda e^K \cosh(C) + 2\sinh(2K) = 0.$$

Solving, we get

$$\lambda_{1,2} = e^K \cosh(C) \pm (e^{2K} \sinh^2(C) + e^{-2K})^{\frac{1}{2}},$$

and hence the maximum characteristic value equation is

$$\lambda_1 = e^K \cosh(C) + (e^{2K} \sinh^2(C) + e^{-2K})^{\frac{1}{2}}.$$

Indeed, since $\lambda_2/\lambda_1 < 1, \forall\, C > 0$, we have

$$\begin{aligned} \lim_{N\to\infty} N^{-1} \ln Z &= \lim_{N\to\infty} N^{-1} \ln \lambda_1^N (1 + (\lambda_2/\lambda_1)^N) \\ &= \ln \lambda_1 + \lim_{N\to\infty} N^{-1} \ln(1 + (\lambda_2/\lambda_1)^N) \\ &= \ln \lambda_1 \\ &= \ln(e^K \cosh(C) + (e^{2K} \sinh^2(C) + e^{-2K})^{\frac{1}{2}}). \end{aligned}$$

The total magnetization for the linear chain of length N according to Eq. (3.2) and Eq. (3.15) is

$$M = m\frac{\partial \ln Z}{\partial C} mN \sinh(C)(\sinh^2(C) + e^{-4K})^{-\frac{1}{2}}.$$

This expression vanishes with H; hence the linear chain is not ferromagnetic. A plot of magnetization (M/mN) against the magnetic field $C = mH/k_BT$ for seve al values of $K = \frac{1}{2}J/k_BT$ is shown in Fig. 3.2 below. Here, it has been assumed that $m/k_B = J/k_B = 1$.

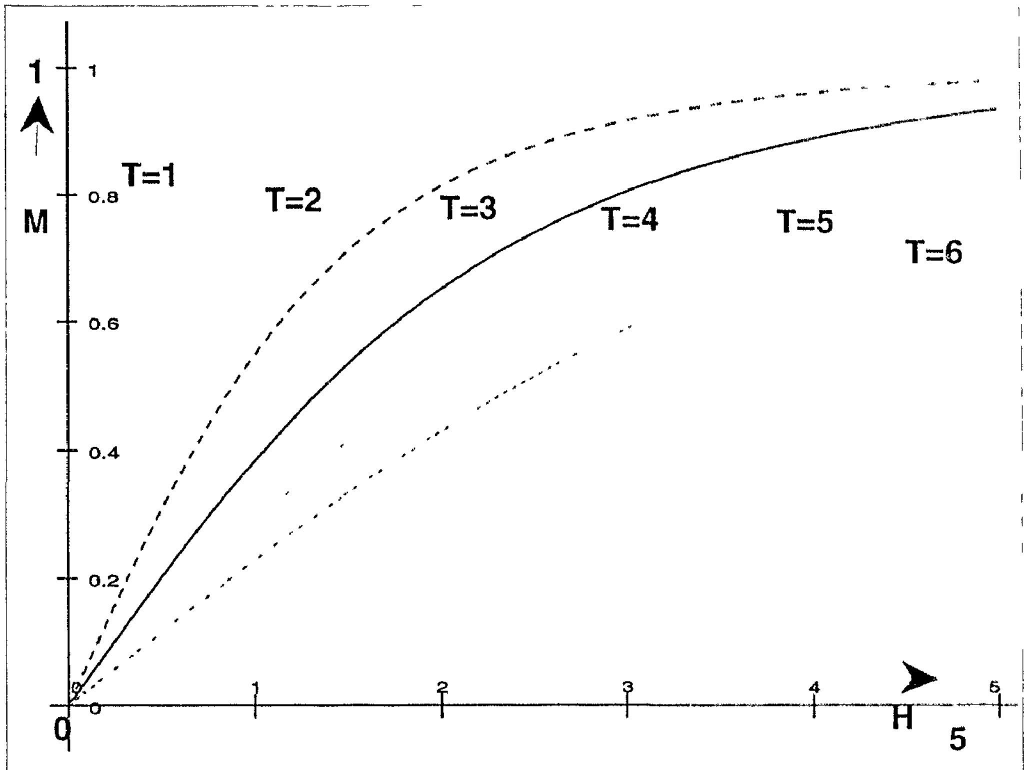


FIG. 3. 2. Magnetization *M(H)* of 1-d Ising model at various temperatures (*T*) for *m/k*$_B$*=J/k*$_B$*=1*

The molecular paramagnetic susceptibility is defined as

$$\chi = (m^2/k_BT)\exp(J/k_BT).$$

The partition function in the absence of a magnetic field is given by

$$Z = (2\cosh(K))^N,$$

which gives the energy U from Eq. (3.2) as a function of temperature as

$$U = -\frac{1}{2}J\tanh(K) = -\frac{1}{2}J\tanh(J/2k_BT).$$

3.4 The Open Chain Without Magnetic Field

The partition function of the open chain without the magnetic field is given by

$$Z = \sum_{\sigma_1=\pm 1} \cdots \sum_{\sigma_N=\pm 1} \prod_{j=1}^{N-1} \exp(K\sigma_j\sigma_{j+1}).$$

Only one of the above factors involves σ_N; in other words, in the one-dimensional structure we can separate off and sum over the Nth spin to get

$$\begin{aligned} \sum_{\sigma_N=\pm 1} \exp(K\sigma_{N-1}\sigma_N) &= \exp(K\sigma_{N-1}) + \exp(-K\sigma_{N-1}) = 2\cosh(K\sigma_{N-1}) \\ &= 2\cosh(K), \end{aligned}$$

and therefore

$$Z = 2\cosh(K\sigma_{N-1}) \sum_{\sigma_1=\pm 1} \cdots \sum_{\sigma_{N-1}=\pm 1} \prod_{j=1}^{N-2} \exp(K\sigma_j\sigma_{j+1}).$$

Since $\cosh(K) = \cosh(-K)$ and $\sigma_{N-1} = \pm 1$, we set

$$\cosh(K\sigma_{N-1}) = \cosh(K),$$

and thus

$$\begin{aligned} Z &= 2\cosh(K) \sum_{\sigma_1=\pm 1} \cdots \sum_{\sigma_{N-1}=\pm 1} \prod_{j=1}^{N-2} \exp(K\sigma_j\sigma_{j+1}) \\ &= 2\cosh(K) Z_{N-1}. \end{aligned}$$

The recursive relation $Z = 2\cosh(K)Z_{N-1}$ can be solved by iteration and by noting the additional fact that $Z_1 = 2$. Hence

$$Z = 2^N \cosh^{N-1}(K).$$

By evaluating the limit $\lim_{N\to\infty} N^{-1} \ln Z = \ln(2\cosh(K))$, it is seen to be a completely analytic function of K, and hence of all temperatures, $\forall\, T > 0$. Thus, as expected, there is no phase transition.

3.5 The Complete Solution of the 2-D Ising Model on a Square Lattice

At this point, we will consider the basic concepts underlying the Onsager solution of the Ising model on a square lattice. The symmetry method does not reveal anything about the thermal behavior of the Ising net in the neighborhood of the singular temperature. Fortunately, this information is provided by Onsager [4], who derived the complete solution for the Ising model on the rectangular lattice.

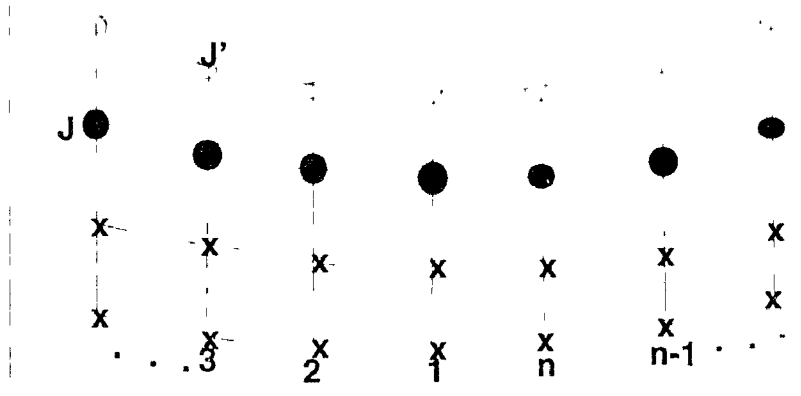


FIG. 3.3. Adaptation of Ising net to eigenvalue method

The approach Onsager adopted uses the operator algebra. In his work he divided the lattice into n parallel chains. The chains were built simultaneously tier by tier, adding one atom to each chain in each step as shown in Fig. 3.3. The jth chain gets the variable σ_j capable of the values ± 1.

The operator that describes the addition of a new tier with the interaction energy

$$u_1((\sigma),(\sigma')) = -J\sum_{j=1}^{n}\sigma_j\sigma_j' = -k_BTH\sum_{j=1}^{n}\sigma_j\sigma_j'$$

is

$$V_1 = \prod_{j=1}^{n}(e^{H} + e^{H}C_j) = (2\sinh(2H))^{n/2}e^{H*B},$$

where $B \equiv \sum_{j=1}^{n} C_j$. The individual operators $C_1, ..., C_n$ have the following effect:

$$((C_j, \psi(\sigma_1, ..., \sigma_j, ..., \sigma_n)) = \psi(\sigma_1, ..., \sigma_{j-1}, -\sigma_j, \sigma_{j+1} ..., \sigma_n). \tag{3.16}$$

Assume a similar interaction between the adjacent atoms in a tier, only with the independent value J' for the pairwise energy of interaction. For symmetry, let the nth atom be neighbor to the first. Then the periodic condition $\sigma_{j+n} = \sigma_j$ holds. Hence the total tierwise interaction energy is

$$u_2(\sigma_1, ..., \sigma_n) = -J'\sum_{j=1}^{n}\sigma_j\sigma_{j+1} = -k_BTH'\sum_{j=1}^{n}\sigma_j\sigma_{j+1}.$$

The effect of this interaction is to multiply the general term of the partition function represented by one of the 2^n vector components, $\psi(\sigma_1, ..., \sigma_n)$, by the appropriate factor $\exp(u_2(\sigma_1, ..., \sigma_n))/k_BT$. The corresponding operator V_2 has a diagonal matrix in the representation. It can be constructed from the simple operators $s_1, ...s_n$ which multiply ψ by its $first, ..., n$th argument, as follows:

$$(s_j, \psi(\sigma_1, \ldots, \sigma_n)) = \sigma_j\psi(\sigma_1, \ldots, \sigma_n), \tag{3.17}$$

$$A = \sum_{k=1}^{n} s_js_{j-1}, \quad \text{and} \quad V_2 = \exp(H'A).$$

The crystal can be built in alternate steps, adding a new tier of atoms as illustrated in Fig. 3.4, then introducing interaction among atoms in the same tier, and then adding another tier, etc. The alternate modifications of the partition function are given by the product $...V_2V_1V_2V_1V_2V_1$,

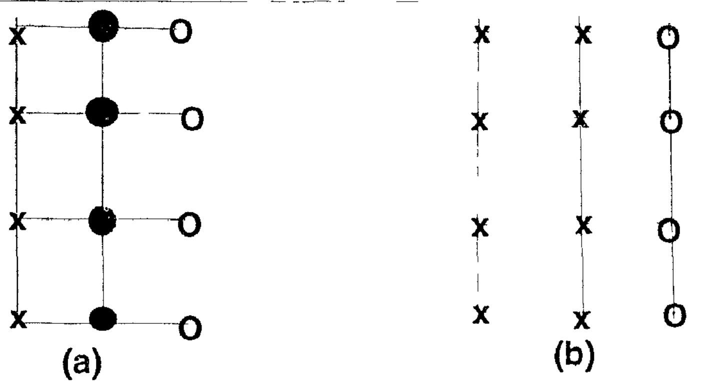


FIG. 3.4. Two-step extension of a 2-d crystal. (a) A new tier of atoms O is added (V_1); their configuration depends on that of the atoms ● in the previous marginal position. (b) Interaction energy between marginal atoms O is introduced (V_2), which modifies the distribution of confuguration in this tier of atoms.

with alternating factors. The addition of one tier of atoms with interaction both ways is represented by $V = (V_2V_1)$. The eigenvalue problem which yields the partition function of the crystals is therefore given as

$$\lambda\psi = (V, \psi) = (V_2V_1, \psi) = (2\sinh(2H))^{n/2}(\exp(H'A)\exp(H^*B), \psi).$$

Thus the basic matrix equation that needs to be solved is

$$\lambda\psi = \exp(H'\sum_{j=1}^{n}\sigma_j\sigma_{j+1}) \sum_{\sigma_1'...\sigma_n'=\pm 1} \exp(H\sum_{j=1}^{n}\sigma_j\sigma_j')\psi(\sigma_1'...\sigma_n'). \tag{3.18}$$

The operators (3.16) and (3.17) form a complete generating basis for the matrix algebra and satisfy the following conditions:

$$s_j^2 = C_j^2 = 1, \quad s_jC_j = -C_js_j,$$

$$s_j s_k = s_k s_j, \;\; C_j C_k = C_k C_j, \tag{3.19}$$

$$s_j C_k = -C_k s_j, \;\; (j \neq k).$$

The operator

$$\mathcal{H} = \exp(H' \sum_{j=1}^{n} \sigma_j \sigma_{j+1}) \sum_{\sigma_1' \dots \sigma_n' = \pm 1} \exp(H \sum_{j=1}^{n} \sigma_j \sigma_j') \tag{3.20}$$

can be expressed in terms of the s's and the C's. Onsager constructed a subalgebra containing $\mathcal{H}$ which is invariant with respect to the cylinder in Fig. 3.3. The subalgebra can be generated as a direct product of mutually commutating quantenion algebra [4]. This quantenion basis has the property that $\mathcal{H}$ can be written as a direct product of operators belonging to each basis,

$$\mathcal{H} = \mathcal{H}_1 \times \mathcal{H}_2 \times \dots \times \mathcal{H}_n.$$

Each quantenion basis is two-dimensional, so the problem only demands the solution of a series of quadratic equations. The eigenvalue obtained is a product of the form

$$\lambda = \prod_{r=0}^{n} \lambda_r$$

$$\lambda_r = e^{\frac{1}{2}\gamma_r}, \;\; e^{-\frac{1}{2}\gamma_r} \text{ or } 1 \text{ for } r = 0, n,$$

$$\lambda_r = e^{\gamma_r}, \;\; e^{-\gamma_r} \text{ or } 1 \text{ for } r \neq 0, n,$$

with $\gamma_0 = K^* - K'$ and $\gamma_n = K' + K^*$.

3.6 Thermodynamic Properties of Large Crystals

To compute the partition function per atom

$$\lambda = \lambda_\infty = \lim_{n \to \infty} (\lambda_{\max})^{1/n},$$

for an infinite crystal, replace the sum with an integral. Thus we have

$$\ln \lambda_\infty = \frac{1}{2}\ln(2\sinh(2H)) + \frac{1}{2\pi}\int_0^\pi \gamma(\omega)d\omega, \tag{3.21}$$

where

$$\gamma(\omega) = \cosh^{-1}(\cosh(2H')\cosh(2H^*) - \sinh(2H')\sinh(2H^*)\cos(\omega)).$$

Using the identity

$$\int_0^{2\pi} \ln(2\cosh(x) - 2\cos(\omega))d\omega = 2\pi x,$$

the above integral can be converted into a double integral to describe a symmetrical function of H and H', namely

$$\ln(\lambda/2) = \frac{1}{2}\pi^{-2}\int_0^\pi\int_0^\pi \ln(\cosh(2H)\cosh(2H') - \sinh(2H)\cos(\omega) - \sinh(2H')\cos(\omega'))d\omega d\omega'. \tag{3.22}$$

Using the notations $2\kappa = \tanh(2H)/\cosh(2H')$ and $2\kappa' = \tanh(2H')/\cosh(2H)$, a generalization of the Kramers and Wannier series can be obtained. By expanding the logarithm in powers of κ and κ' and integrating term by term, the following can be obtained:

$$\begin{aligned}\ln\lambda - \frac{1}{2}\ln(4\cosh(2H)\cosh(2H')) &= \frac{1}{2}\pi^{-2}\int_0^\pi\int_0^\pi \ln(1 - 2\kappa\cos(\omega) - 2\kappa'\cos(\omega'))d\omega d\omega' \\ &= \frac{1}{2}\sum_{r+s>0}(2r+2s-1)!(r!)^{-2}(s!)^{-2}\kappa^{2r}\kappa'^{2s}. \end{aligned}\tag{3.23}$$

Considering the case when $H = H'$ and $\kappa = \kappa'$, the quadratic symmetry yields

$$\begin{aligned}\ln\lambda - \ln(2\cosh(2H)) &= \frac{1}{2}\pi^{-2}\int_0^\pi\int_0^\pi \ln(1 - 4\kappa\cos(\omega_1)\cos(\omega_2))d\omega_1 d\omega_2 \\ &= -\sum_{n=1}^{\infty}\left(\binom{2n}{n}^2(4n)^{-1}\right)\kappa^{2n}. \end{aligned}\tag{3.24}$$

Even though the convergent of the series solution cannot easily be inferred from the matrix approach, it can easily be deduced here. The expansions converge for all values of H and H' because

$$|2\kappa\cos(\omega) + 2\kappa'\cos(\omega')| \leq 2\kappa + 2\kappa' = \sin(g+g') \leq 1,$$

where $2\kappa = \sin(g)\cos(g')$ and $2\kappa' = \cos(g)\sin(g')$. The angles g' and g are defined through the gudermannian angles as

$$g' = gd(2H'), \quad g = gd(2H) = 1/2\pi - gd(2H^*).$$

The limit of convergence of the series is the transition point. For the special case of quadratic symmetry $H = H'$, the computation of the thermodynamic functions can be simplified. Assuming the number of spins fitting one tier is large and $k_1 = 4\kappa$, the partition function has this form:

$$\ln(\lambda/2\cosh(2H)) = \frac{1}{2\pi}\int_0^\pi \ln(\frac{1}{2}(1 + (1 - k_1^2\sin^2(\phi))^{\frac{1}{2}})d\phi. \tag{3.25}$$

The following conclusions can easily be deduced. The partition function (3.21) yields the free energy, which is given as

$$F = U - TS = -Nk_BT\ln\lambda,$$

$$U = F - T(dF/dT) = Nk_BT^2d(\ln(\lambda)/dT),$$

$$C = dU/dT,$$

for a crystal of N atoms. For the simpler case, $H = H'$, differentiating Eq. (3.25) under the integral sign yields, for the energy U,

$$U = -NJ\frac{d\ln\lambda}{dH} = -NJcoth2H(1 + \frac{2}{\pi}s_1K_1(k_1)),$$

where $s_1 = \pm(1 - k_1^2)^{\frac{1}{2}}$. The specific heat C is given as

$$C = Nk_B \frac{d^2 \ln \lambda}{dH^2} = Nk_B(H coth 2H)^2(2/\pi)(2K_1(k_1) - E_1(k_1) - (1 - s_1)(\frac{1}{2}\pi + s_1 K_1(k_1))).$$

The K_1 and the E_1 are the complete elliptic integrals of the first kind, given below:

$$K_1 = K(k_1) = \int_0^{\pi} \frac{1}{2}(1 - k_1^2 \sin^2(\phi))^{-\frac{1}{2}} d\phi,$$

and

$$E_1 = E(k_1) = \int_0^{\pi} \frac{1}{2}(1 - k_1^2 \sin^2(\phi))^{\frac{1}{2}} d\phi.$$

The plus (+) sign in (s_1) holds below the transition (Curie) point, and the minus (−) sign above the Curie point. The integral (3.25) cannot be expressed in a closed form, but a convergent series can be given with the following notation:

$$K_1' = K(k_1'),\ \ \ln(q_1) = \pi \tau_1 i = -\pi K_1'/K_1 = 1/2\tau_0 = 1/2\tau,$$

$$G = \sum_{i=0}^{\infty} (-1)^i (2i + 1)^{-2}.$$

The number G is the Catalan's constant, and τ is the ratio of periods. The convergent series to Eq. (3.25) is given as

$$\ln(\lambda) = \frac{1}{2}\ln(2\sinh(2H)) - \frac{1}{4}\ln(q_1) + \sum_{n=1}^{\infty}(-1)^n(2n - 1)\ln(1 - q_1^{2n-1}),$$

which converges except in the immediate neighborhood of the critical point. For the region near the critical point we have

$$\ln(\lambda) = \frac{1}{2}\ln(2\sinh(2H)) + \frac{2}{\pi}G + \frac{1}{\pi}\sum_{n=0}^{\infty}(-1)^n \frac{(1 + (2n+1)(\pi i/\tau_1) - \exp[-(2n+1)\pi i/\tau_1])}{(2n+1)^2 \sinh^2((n + \frac{1}{2})\pi i/\tau_1)}.$$

The following fact has been applied:

$$\int_0^{\pi/2} \ln(cot(x)dx) = G.$$

A singularity occurs for $H = H_c = \frac{1}{2}\ln cot\pi/8$, in which case $k_1 = 1, K_1 = \infty, K_1' = \frac{1}{2}\pi, E_1 = 1$. The specific heat becomes infinity at the critical point, and the energy is continuous because $s_1 = 0$. The analytic nature of the singularity is evident from the approximate formulas

$$K_1 \sim \ln(4/k_1') \sim log(2^{\frac{1}{2}}/|H - H_c|),$$

$$C/Nk_B \sim (2/\pi)(\ln cot\pi/8)^2(K_1 - 1 - \frac{1}{4}\pi).$$

The following are the critical values at the critical point:

$$\begin{aligned} H_c &= J/k_BT_c = \ln cot\pi/8 = 0.4406867935, && (3.26)\\ -F_c/Nk_BT_c &= \ln\lambda_c = \frac{1}{2}\ln 2 + (2/\pi)G = 0.9296953983, && (3.27)\\ -U_c/NJ &= 2^{\frac{1}{2}} = 1.4142135624, && (3.28)\\ S_c/Nk_B &= \ln\lambda_c - 2^{\frac{1}{2}}H_c = 0.3064701582 = \ln 1.3586209232. && (3.29) \end{aligned}$$

The exact value of λ_c is given by

$$\lambda_c = 2^{\frac{1}{2}}e^{\frac{2}{\pi}G}.$$

The Kramers and Wannier estimation of $\lambda_c = 2.5335$ was very close to the exact value given above.

The above work is of great interest because it enables us to construct the exact partition function for both low and high temperature to any degree of accuracy needed. In fact, for the quadratic symmetry, with $H = H'$ and $\kappa = \kappa'$, one can obtain a closed form solution of the Ising model on a square lattice. This is a generalization of the partition function computed by Kramers and Wannier. The symbolic solution of the above work is given the name *ex2dseries* (exact two-dimensional series) for the Ising model on a square lattice. The program takes the order as an argument and computes the series to this order. It uses Eq. (3.24) to directly compute the series. The following illustrative examples show how the program works. This can be used as a test result for other approximate results. The program is given in the Appendix A. Some examples are given below:

Example 1.

ex2dseries(10);

$$pf = 2\cosh(H)(1 - x^2 - 4x^4 - 29x^6 - 265x^8 - 2745x^{10})$$

Example 2.

ex2dseries(20);

$$pf = 2\cosh(H)(1 - x^2 - 4x^4 - 29x^6 - 265x^8 - 2745x^{10} - 30773x^{12}$$
$$-364315x^{14} - 4488749x^{16} - 57020414x^{18} - 741999700x^{20})$$

3.7 2-D Perturbation Method of the Ising Model

In considering the theory of the 2-d Ising model on a simple square lattice, Kramers and Wannier introduced the screw construction, which is a simplification of the matrix constr.ction. It can be simply described as having the lattice sites regularly distributed along a continuous line twisting its way in a screwise fashion over the surface of a torus. Suppose that it consists of m pitches of n spins, and the configuration on each pitch is denoted by $(\sigma_1, \sigma_2, ..., \sigma_n)$. The coordinates $\sigma_1, \sigma_2, ..., \sigma_{mn}$ can take values $+1$ or -1. The total energy can be written as

$$E = -\frac{1}{2}J \sum_{<i,j>} \sigma_i \sigma_j,$$

where the sum is over all pairs $< i, j >$ which are nearest neighbors. Let n be the number of spins that make up a pitch of the screw, by adding the nth spin, we notice that it interacts with the 0th and the $(n-1)$th spins only, because only nearest neighbors are considered. Thus the two interaction energies are

$$-K\sigma_n\sigma_{n-1}$$

and

$$-K\sigma_n\sigma_0,$$

where $K = J/2kT$. Let the probability of the arrangement $\sigma_{n-1}, \sigma_{n-2}, ..., \sigma_0$ be $A(\sigma_{n-1}, \sigma_{n-2}, ..., \sigma_0)$, and the one including σ_n at th next position be given by $P(\sigma_n, \sigma_{n-1}, ..., \sigma_0)$. By Boltzmann's theorem, the probability for any particular arrangement is proportional to the $\exp(-E/kT)$. Applying this theorem one obtains

$$\lambda P(\sigma_n, ..., \sigma_0) \quad = \quad A(\sigma_{n-1}, ..., \sigma_0)exp[K\sigma_n(\sigma_{n-1} + \tag{3.30}$$

Summing $P(\sigma_n, ..., \sigma_0)$ over σ_0 gives $\sum_{\sigma_0} P(\sigma_n, ..., \sigma_0) = \lambda A(\sigma_n, ..., \sigma_1)$. This case is identical to the one described by $A(\sigma_{n-1}, \sigma_{n-2}, ..., \sigma_0)$, if the screw is very long. The difference being that σ_1 takes the place of σ_0 and σ_2 of σ_1, etc. Thus one gets

$$\sum_{\sigma_0} exp[K\sigma_n(\sigma_{n-1} + \sigma_0)]A(\sigma_{n-1}, ..., \sigma_0) = \lambda A(\sigma_n, ..., \sigma_1), \tag{3.31}$$

Since the matrix $\mathcal{M} = \sum_{\sigma_0} \exp[K\sigma_n(\sigma_{n-1} + \sigma_0)]$ is not symmetric, it is necessary to introduce right- and left-handed characteristic vectors $\mathbf{A}_q$ and $\mathbf{B}_p$, satisfying

$$\mathcal{M}\mathbf{A}_q = \lambda_q\mathbf{A}_q \text{ and } \mathbf{B}_p^t\mathcal{M} = \lambda_q\mathbf{B}_p^t.$$

Normalizing the characteristic vectors $\mathbf{A_q}$ and $\mathbf{B_p}$, it can be made to satisfy orthogonality conditions

$$\mathbf{B}_p^t \mathbf{A}_q = \sum_\alpha \mathbf{B}_p(\alpha)\mathbf{A}_q(\alpha) = \delta_{pq}. \tag{3.32}$$

The matrix element $(\alpha_1|\mathcal{M}|\alpha_2)$ can be expressed as a bilinear combination of the components $\mathbf{A_p}(\alpha_2)$ and $\mathbf{B_p}(\alpha_1)$, and we find

$$\sum_{q=1}^{2^n} \lambda_q^n \mathbf{A}_p(\alpha_2)\mathbf{B}_p(\alpha_1) = \exp\left[-\frac{\mathrm{v}(\alpha_2) + \mathrm{v}(\alpha_1, \alpha_2)}{k_B T}\right], \tag{3.33}$$

where $\mathrm{v}(\alpha_2)$ is the total energy of interaction between nearest neighboring atoms of pitch 2, and $\mathrm{v}(\alpha_1, \alpha_2)$ is the energy of interaction between the pitches 1 and 2. By repeated use of Eq. (3.32) and Eq. (3.33), one finds the partition function to be

$$Z = \sum_{p=1}^{2^n} \lambda_p^{mn}.$$

If one compares the 3-d case in section 3.10 with the 2-d case, it is seen that λ^n of the 2-d case is the analogue of λ in 3-d. The advantage of this result is that λ has a meaning which is independent of the size of the crystal. Also the logarithm of $\lambda_{\max}$ is the free energy per particle. Thus $\lambda_{\max}$ enables one to study the variation of the properties of the crystal as it becomes infinite in two directions [23]. The matrix $\mathcal{M}(K)$ is brought to its standard form by arranging it in some definite order. This can be done in the following way, as suggested by Kramers and Wannier. For any configuration $(- + - + - + -)$, replace every $+$ by 0 and every $-$ by 1. Then read the number in the base two system. For example, in the given configuration, one gets 1010101, which gives the order to be 83. The configuration will be separated into two parts. Those with

$$\sigma_n = 1 \text{ are } 0, 1, ..., 2^{n-1} - 1,$$

and the rest that belong to the class

$$\sigma_n = -1 \text{ are } 2^n - 1, 2^n - 2, ..., 2^{n-1}.$$

The reason for the arrangement is that if α and $\bar{\alpha}$ belong to corresponding places in the two classes, then their order numbers add up to 2^{n-1} and are conjugate to each other, in the sense that by reversing all the signs of one, the other can be obtained. Let

$$K = \frac{J}{2k_BT},$$

also let

$$\alpha = e^{2K}, \text{ and } \beta = e^{-2K}.$$

Then the standard form of the matrix $\mathcal{M}$ is given by,

$$\mathcal{M}(K) = \left|\begin{array}{cccc|cccc}
\alpha\ 1 & & & & & & & \\
 & \alpha\ 1 & & & & & & \\
 & & \cdots & & & & & \\
 & & & \alpha\ 1 & & & & \\
 & & & & & & & \beta\ 1 \\
 & & & & & & \cdots & \\
 & & & & & \beta\ 1 & & \\
 & & & & \beta\ 1 & & & \\
\hline
 & & & & \alpha\ 1 & & & \\
 & & & & & \alpha\ 1 & & \\
 & & & & & & \cdots & \\
 & & & & & & & \alpha\ 1 \\
 & & & \beta\ 1 & & & & \\
 & & \cdots & & & & & \\
 & \beta\ 1 & & & & & & \\
\beta\ 1 & & & & & & &
\end{array}\right|,$$

where the blank spaces indicate zeros.

The matrix $\mathcal{M}(K)$ is simplified using the transformation matrix **H** defined as

$$\mathbf{H} = \frac{1}{\sqrt{2}}\left|\begin{array}{c|c} \mathbf{I} & \mathbf{I} \\ \hline \mathbf{I} & -\mathbf{I} \end{array}\right|, \tag{3.34}$$

where **I** is the unit matrix of order 2^{n-1}. Performing the matrix product operations $\mathbf{H}\mathcal{M}(\mathbf{K})\mathbf{H}^{-1}$, one obtains two submatrices $\mathbf{V}_{+}(K)$ and $\mathbf{V}_{-}(K)$ which are defined to be the upper positive and lower negative V-matrices of Kramers and Wannier. Their forms are given as

$$\mathbf{V}_+(K) = \begin{vmatrix} \alpha & 1 & & & & & & \\ & & \alpha & 1 & & & & \\ & & & & \cdots & & & \\ & & & & & & \alpha & 1 \\ & & & & & & \beta & 1 \\ & & & & \cdots & & & \\ & & \beta & 1 & & & & \\ \beta & 1 & & & & & & \end{vmatrix},$$

and

$$\mathbf{V}_-(K) = \begin{vmatrix} \alpha & 1 & & & & & & \\ & & \alpha & 1 & & & & \\ & & & & \cdots & & & \\ & & & & & & \alpha & 1 \\ & & & & & & -\beta & -1 \\ & & & & \cdots & & & \\ & & -\beta & -1 & & & & \\ -\beta & -1 & & & & & & \end{vmatrix}.$$

These two submatrices are known as the V-matrix. The characteristic values and characteristic vectors of $\mathcal{M}(K)$ fall into two classes. Let $\psi_+(K)$ and $\psi_-(K)$ be the right-handed characteristic vectors of $\mathbf{V}_+(K)$ and $\mathbf{V}_-(K)$, respectively, corresponding to the characteristic values $\lambda_+(K)$ and $\lambda_-(K)$, then the associated vectors of $\mathcal{M}(K)$ are

$$\mathbf{A}_+(K) = \mathbf{H}^{-1} = \begin{vmatrix} \psi_+(K) \\ 0 \end{vmatrix} = \begin{vmatrix} \psi_+(K) \\ \psi_+(K) \end{vmatrix}$$

and

$$\mathbf{A}_-(K) = \mathbf{H}^{-1} = \begin{vmatrix} 0 \\ \psi_-(K) \end{vmatrix} = \begin{vmatrix} \psi_-(K) \\ -\psi_-(K) \end{vmatrix}.$$

The components are arranged in the order of the two groups $\sigma_n = 1$ and $\sigma_n = -1$. In a similar way, for the left-handed characteristic vectors $\phi_+^t(K)$, $\phi_-^t(K)$ of $\mathbf{V}_+(K)$ and $\mathbf{V}_-(K)$, the corresponding vectors are

$$\mathbf{B}_+^t(K) = | \, \phi_+^t(K), \phi_+^t(K) \, |$$

and

$$\mathbf{B}_-^t(K) = | \, \phi_-^t(K), -\phi_-^t(K) \, | \, .$$

It is worthwhile mentioning that the antiferromagnetic case can be easily treated once the ferromagnetic case is discussed. The necessary properties are these: in the antiferromagnetic case the interaction energy is $J < 0$ and hence the parameter $K < 0$, but in the ferromagnetic case the $K > 0$. Let $\mathbf{R}$ be a permutation that changes every alternate atom on a pitch from $+$ to $-$ and vice versa, then it can be verified that

$$\mathbf{R}\mathbf{H}\mathcal{M}(-\mathbf{K})\mathbf{H}^{-1}\mathbf{R}^{-1} = \left|\begin{array}{c|c} \mathbf{V}_+(K) & 0 \\ \hline 0 & -\mathbf{V}_-(K) \end{array}\right|.$$

The characteristic values of $\mathcal{M}(-K)$ are $\lambda_+(K)$ and $-\lambda_-(K)$, and the characteristic vectors $\mathbf{A}_+(-K)$, etc., are obtained from $A_+^t(K), B_+^t(K)$... by permuting the components according to R. Next we tackle the problem of finding the characteristic values ar d the characteristic vectors of $V_+(K)$. In the limiting case of an infinitely large crystal, the maximum characteristic value of $\mathbf{V}_-$ is equal to the maximum characteristic value of $\mathbf{V}_+$

3.8 Low-Temperature Solution of the Matrix Problem

For the low temperature limit, we have

$$\lim_{T\to 0} K = \lim_{T\to 0} \frac{J}{2k_B T} \to +\infty$$

Thus K is large and positive, hence it is ferromagnetic. The parameter β satisfies

$$\lim_{T\to 0} \beta = \lim_{T\to 0} e^{-2K} = 0.$$

Hence the expansion can be carried out on powers of β. Define the matrix $\mathrm{U}_+(\beta)$ as

$$\mathbf{V}_+(K) = \alpha \left| \begin{array}{cccccc} 1 & \beta & & & & \\ & 1 & \beta & & & \\ & & \cdots & & & \\ & & & & 1 & \beta \\ & & & & \beta^2 & \beta \\ & & \cdots & & & \\ & \beta^2 & \beta & & & \\ \beta^2 & \beta & & & & \end{array} \right| = \frac{1}{\beta}\mathrm{U}_+(\beta),$$

where

$$
\mathrm{U}_{+}(\beta) = \begin{vmatrix}
1 & 0 & & & & & & \\
 & & 1 & 0 & & & & \\
 & & & & \cdots & & & \\
 & & & & & & 1 & 0 \\
 & & & & & & & \\
 & & & & & & & \\
 & & & & & & & \\
 & & & & & & &
\end{vmatrix}
+ \beta \begin{vmatrix}
0 & 1 & & & & & & \\
 & & 0 & 1 & & & & \\
 & & & & \cdots & & & \\
 & & & & & & 0 & 1 \\
 & & & & & & 0 & 1 \\
 & & & & \cdots & & & \\
 & & 0 & 1 & & & & \\
0 & 1 & & & & & &
\end{vmatrix}
$$

$$
+\beta^2 \begin{vmatrix}
 & & & & & & & \\
 & & & & & & & \\
 & & & & & & & \\
 & & & & & & 0 & 1 \\
 & & & & & & & \\
 & & & & \cdots & & & \\
 & & 0 & 1 & & & & \\
0 & 1 & & & & & &
\end{vmatrix}.
$$

Setting the constant matrices in the equation $\mathrm{U}_{+}(\beta)$ to $\mathbf{U_0}$, $\mathbf{U_1}$, and $\mathbf{U_2}$, one can write

$$
\mathrm{U}_{+}(\beta) = \mathbf{U_0} + \beta \mathbf{U_1} + \beta^2 \mathbf{U_2}.
$$

Assume that there are n atoms in a pitch, then these matrices will have a dimension $2^{n-1}, 2^{n-1}$. This corresponds to the configurations $0, 1, 2, \ldots, 2^{n-1} - 1$

with $\sigma_n = 1$. What is required is to obtain a power series expansion of $\lambda_{\max}$, the maximum characteristic value, and its corresponding characteristic vector $\psi_{\max}$ in the parameter β. The question is whether there is justification in seeking a power series expansion for $\lambda_{\max}$ and $\psi_{\max}$. An examination of $\mathbf{U}_0$ reveals that it has a single nondegenerate characteristic value of unity, and a $2^{n-1}-1$-fold degenerate value of zero. This means that $\lambda_{\max}$ will be well separated from the other characteristic values at low temperatures. Hence it is an analytic function of β in the neighborhood of $\beta = 0$, thus justifying the approximation by power series. Now let

$$\lambda = \lambda_{\max} = \sum_p \lambda_p \beta^p \tag{3.35}$$

and

$$\psi = \psi_{\max} = \sum_p \psi_p \beta^p. \tag{3.36}$$

Inserting Eq. (3.35) and Eq. (3.36) into the Eq. (3.37)

$$\mathbf{U}_+(\beta)\psi = \lambda\psi, \tag{3.37}$$

and equating the coefficients of like powers of β, a set of equations for λ_p and ψ_p can be derived as

$$\mathbf{U}_0\psi_p + \mathbf{U}_1\psi_{p-1} + \mathbf{U}_2\psi_{p-2} = \sum_{q=0}^{p} \lambda_q \psi_{p-q} \quad p \geq 0. \tag{3.38}$$

From Eq. (3.38), for $p = 0, 1, 2$, one obtains

$$U_0\psi_0 = \lambda_0\psi_0, \tag{3.39}$$

$$U_0\psi_1 + U_1\psi_0 = \lambda_0\psi_1 + \lambda_1\psi_0, \tag{3.40}$$

$$U_0\psi_2 + U_1\psi_1 + U_2\psi_0 = \lambda_0\psi_2 + \lambda_1\psi_1 + \lambda_2\psi_0. \tag{3.41}$$

By raising $\mathbf{U}_0$ to a higher power, it can be seen that $\lambda_0 = 1$, and $\psi_0(0) = 1$, and $\psi_0(\alpha) = 0$ for $\alpha = 1, 2, ...2^{n-1} - 1$. Since $\mathbf{U}_1\psi_0 = 0$, it is easy to deduce that $\lambda_1 = 0$. The equation for ψ_1 becomes $\mathbf{U}_0\psi_1 = \psi_1$. For consistency, and to remove arbitrariness, set $\psi_1(\alpha) = 0$ for $\alpha = 0, 1, 2, ...2^{n-1} - 1$. To enhance the speed of the computation and simplification of the vector ψ, I have modified Eq. (3.38) to the set of equations given in the form

$$(\mathbf{I} - \mathbf{U}_0)\psi_i = \mathbf{U}_1\psi_{\mathbf{i-1}} + \mathbf{U}_2\psi_{\mathbf{i-2}} - \sum_{j=0}^{i-3} \psi_{i-j-3}\lambda_{j+3}, \; i \geq 3. \qquad (3.42)$$

Here I is the identity matrix of the same dimension as the matrix $\mathbf{U}_0$. The whole idea of the perturbation method depends on solving the set of equations of the type given in Eq. (3.42). Lastly, from orthogonal considerations, ψ_0 is expected to be orthogonal to each of the ψ_j, giving $\lambda_r = \psi_0'\mathbf{U}_1\psi_{r-1}$. However, since $\psi_0'\mathbf{U}_1 = |010...0|$,

$$\lambda_r = \psi_{r-1}(1).$$

3.9 2-D Perturbation Algorithm and Examples

The functions I have developed to compute the partition function by series expansion are called *d2lnpps, d2lppps, d2rppps,* and *d2rnpps.* The names are taken from the two-dimensional left (right) positive (negative) Potts perturbation series. The functions take two arguments. The first is the size of the matrix, and the second is the order of the series one wants to compute. One has to be careful of the size of the matrix because it increases rapidly as (2^{n-1}) with n. For example, if one sets $n = 5$, it produces a V-matrix of dimension $(16, 16)$. Matrices of any size can be calculated.

However, because of the rapid growth of the matrix, it takes considerably longer to compute the bigger arrays than the smaller ones. Also any desirable degree of accuracy can be set. Nevertheless, it takes longer to compute higher orders of the series as computation becomes more complex. Briefly, the main idea behind the algorithm is that, when any of the functions *d2lnpps, d2lppps, d2rppps,* or *d2rnpps* receives its arguments, it creates the matrices U_0, U_1 and U_2, of Eq. (3.38). Then using Eq. (3.42), an identity is set up to evaluate ϕ and ψ, which are the coefficients of the series. From the evaluated coefficients the series is composed. The eigenvalues series is given in terms of $b = e^{-2K}$ where $K = \frac{J}{2k_BT}$. The algorithm is summarized by the following Pseudocode:

1. Input size of array
2. Input order of the series
3. Create the constant matrices U_0, U_1, U_2
4. Set up matrix identities using Eq. (3.42)
5. Compute series coefficients
6. Compute matrix coefficients
7. Create the partition function series in terms of $b = e^{-2K}$
8. Create the eigenvector series in terms of $b = e^{-2K}$

Our program took only 7 minutes on the Sun workstation to compute the eigenvalues series of degree 50 for $n = 5$, an array of $(32, 32)$. This

computation is given below as an example. The programs are given in the Appendix A.

pf=d2rppps(5,50);

$$
\begin{aligned}
pf = 1 \quad &+ \quad b^4 + 3b^6 + 15b^8 + 48b^{10} + 70b^{12} - 253b^{14} - 3118b^{16} - 18680b^{18} \\
&- \quad 75512b^{20} - 172286b^{22} + 336855b^{24} + 6593837b^{26} + 44592836b^{28} \\
&+ \quad 199989549b^{30} + 533833645b^{32} - 567620125b^{34} - 17855094024b^{36} \\
&- \quad 132068426624b^{38} - 635288749768b^{40} - 1881418778662b^{42} \\
&+ \quad 7146239389l7b^{44} + 54551718936085b^{46} + 436165539114081b^{48} \\
&+ \quad 2219269737289813b^{58}.
\end{aligned}
$$

The eigenvector series is computed along with the computation of the eigenvalues series. For the sake of space, only 20 terms are represented for a smaller matrix $n = 4$. This eigenvector series is given below as

$$
\begin{bmatrix} 0 \\ 54143b^{20} \\ 18887b^{20} \\ 51200b^{20} \\ 11884b^{20} \\ -2038b^{20} \\ 25058b^{20} \\ 28309b^{20} \end{bmatrix}
+
\begin{bmatrix} 0 \\ 15410b^{19} \\ 5048b^{19} \\ 13678b^{19} \\ 2091b^{19} \\ -3486b^{19} \\ 3122b^{19} \\ 5566b^{19} \end{bmatrix}
+
\begin{bmatrix} 0 \\ -1128b^{18} \\ -1685b^{18} \\ -90b^{18} \\ -1905b^{18} \\ -2557b^{18} \\ -2622b^{18} \\ 457b^{18} \end{bmatrix}
+
\begin{bmatrix} 0 \\ -3309b^{17} \\ -2153b^{17} \\ -3243b^{17} \\ -2080b^{17} \\ -1652b^{17} \\ -3088b^{17} \\ 1392b^{17} \end{bmatrix}
+
$$

$$\begin{bmatrix} 0 \\ -2763b^{16} \\ -1314b^{16} \\ -2547b^{16} \\ -1140b^{16} \\ -776b^{16} \\ -2163b^{16} \\ -1483b^{16} \end{bmatrix} + \begin{bmatrix} 0 \\ -2001b^{15} \\ -1005b^{15} \\ -1745b^{15} \\ -792b^{15} \\ -354b^{15} \\ -1189b^{15} \\ -713b^{15} \end{bmatrix} + \begin{bmatrix} 0 \\ -891b^{14} \\ -399b^{14} \\ -899b^{14} \\ -330b^{14} \\ -144b^{14} \\ -592b^{14} \\ -487b^{14} \end{bmatrix} + \begin{bmatrix} 0 \\ -458b^{13} \\ -180b^{13} \\ -398b^{13} \\ -126b^{13} \\ -29b^{13} \\ -248b^{13} \\ -210b^{13} \end{bmatrix} +$$

$$\begin{bmatrix} 0 \\ -188b^{12} \\ -88b^{12} \\ -180b^{12} \\ -64b^{12} \\ -4b^{12} \\ 83b^{12} \\ -65b^{12} \end{bmatrix} + \begin{bmatrix} 0 \\ -35b^{11} \\ -3b^{11} \\ -44b^{11} \\ -b^{11} \\ 9b^{11} \\ -19b^{11} \\ -40b^{11} \end{bmatrix} + \begin{bmatrix} 0 \\ -15b^{10} \\ -4b^{10} \\ -6b^{10} \\ b^{10} \\ 10b^{10} \\ 6b^{10} \\ b^{10} \end{bmatrix} + \begin{bmatrix} 0 \\ 10b^{9} \\ 5b^{9} \\ 5b^{9} \\ 4b^{9} \\ 5b^{9} \\ 9b^{9} \\ 3b^{9} \end{bmatrix} + \begin{bmatrix} 0 \\ 11b^{8} \\ 8b^{8} \\ 10b^{8} \\ 7b^{8} \\ 4b^{8} \\ 7b^{8} \\ 2b^{8} \end{bmatrix} +$$

$$\begin{bmatrix} 0 \\ 5b^{7} \\ 2b^{7} \\ 5b^{7} \\ 2b^{7} \\ 2b^{7} \\ 5b^{7} \\ 5b^{7} \end{bmatrix} + \begin{bmatrix} 0 \\ 6b^{6} \\ 3b^{6} \\ 4b^{6} \\ 2b^{6} \\ b^{6} \\ 3b^{6} \\ b^{6} \end{bmatrix} + \begin{bmatrix} 0 \\ 2b^{5} \\ b^{5} \\ 3b^{5} \\ b^{5} \\ b^{5} \\ 2b^{5} \\ b^{5} \end{bmatrix} + \begin{bmatrix} 0 \\ b^{4} \\ 0 \\ b^{4} \\ 0 \\ b^{4} \\ b^{4} \\ b^{4} \end{bmatrix} + \begin{bmatrix} 0 \\ b^{3} \\ b^{3} \\ b^{3} \\ b^{3} \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ b^{2} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

It is accessed by the Maple command *map(evalm, cv)*. The *cv* means characteristic vectors. To demonstrate the power of the algorithm, some examples are shown for each of the functions we have developed for a matrix of dimension (64, 64).

ex5=d2rppps(5,20);

$$ex5 = 1 + b^4 + 3b^6 + 15b^8 + 48b^{10} + 70b^{12} - 253b^{14} - 3118b^{16} - 18680b^{18} - 75512b^{20}$$

ex6=d2rnpps(5,20);

$$ex6 = 1 - b^4 - b^6 - 5b^8 - 14b^{10} - 70b^{12} - 221b^{14} - 1122b^{16} - 4154b^{18} - 19714b^{20}$$

ex7=d2lnpps(5,20);

$$ex7 = 1 + b^4 + b^6 - 5b^8 - 20b^10 + 8b^{12} + 245b^{14} + 488b^{16} - 2066b^{18} - 11902b^{20}$$

ex8=d2lppps(5,20);

$$ex8 = 1 + b^4 + 2b^6 + 3b^8 + 15b^{10} + 58b^{12} + 150b^{14} + 128b^{16} - 963b^{18} - 7982b^{20}$$

The constant matrices that are used in the computation are U_0, U_1, and U_2. They are evaluated using the Maple command *evalm*. The matrix U_0,

$$U_0 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix},$$

is one of the constant matrices obtained from the partition function series computation in the example ex8 above. The computation is done for $n = 5$, whose dimension is the same as that of the V-matrix from which it is derived. The next matrix U_1 of dimension $(16, 16)$ is the second constant matrix derived from the V-matrix for $n = 5$. It is obtained by the command $evalm(U_1)$, and it is available after the computation of the partition function series in the example ex8. It is represented as

$$U_1 = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}.$$

The last constant matrix to be derived from the V-matrix for $n = 5$ is U_2 of dimension $(16, 16)$. It is obtained by the command $evalm(U_2)$, and it is available after the computation of the partition function series. This particular matrix is obtained after the computation of example ex8 above. Notice that all the constant matrices have the same dimension as that of the V-matrix from which they are derived. The matrix is displayed below:

$$U_2 = \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}.$$

As explained above, by following a binary number system one can obtain the complete matrix. Kramers and Wannier neglected the lower matrix in their work. However, the complete matrix has been built into our program, and it is available in the form shown below. As an example, the complete V-matrix of Kramers and Wannier for $n = 4$ is obtained by the command *d2cvmat*. It must also be mentioned that from the complete matrix the lower and upper parts of Kramers and Wannier's V-matrix can easily be obtained. The transformation H through which this is accomplished is given the name two-dimensional special transformation matrix (*d2stmat*). A demonstration of its effect on the complete V-matrix will be be given below. The matrix below is an example of a (16, 16) 2-d complete V-matrix with default values e^{2K} and e^{-2K}.

d2cvmat(4);

$$\begin{bmatrix}
e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-2K} & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-2K} & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & e^{-2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & e^{-2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & e^{-2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
e^{-2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

The upper positive and lower negative components of the complete V-matrix given above are also available by the commands *d2upvmat* and *d2lnvmat*, respectively. The next example is the upper positive V-matrix obtained from the complete V-matrix given above. This form is obtained through the transformation H, defined by Eq. (3.34). In this example, the dimensions are half those of the complete V-matrix given above, which in this case is (8,8). Also the default values used in the above example have been offset by using the optional arguments s and t.

d2upvmat(4,s,t);

$$\begin{bmatrix} s & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & s & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & s & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & s & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & t & 1 \\ 0 & 0 & 0 & 0 & t & 1 & 0 & 0 \\ 0 & 0 & t & 1 & 0 & 0 & 0 & 0 \\ t & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

As mentioned above, these programs are built to be used as a teaching tool so all the various components are built into the program. This technique shows the flexibility of our program. Also it can be used when solving any problem that needs these special matrix constructions. The example given below is in the form of a lower negative V-matrix. It is also derived from the complete V-matrix given in the example d2cvmat(4). It uses the default elements e^{2K} and e^{-2K} since only one argument is given.

The dimensions are again (8, 8).

d2lnvmat(4);

$$\begin{bmatrix} e^{2K} & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{2K} & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{2K} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{2K} & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -e^{-2K} & -1 \\ 0 & 0 & 0 & 0 & -e^{-2K} & -1 & 0 & 0 \\ 0 & 0 & -e^{-2K} & -1 & 0 & 0 & 0 & 0 \\ -e^{-2K} & -1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

These three functions, *d2cvmat*, *d2upvmat*, and *d2lnvmat*, take three arguments, of which the second and third are optional. One can set his own elements to be used for the array as in the example given above for the *d2upvmat*. However, if no optional arguments are set the default arguments $\exp(2K)$ and $\exp(-2K)$ are used as noted in the other two examples. The result obtained through the special transformation matrix *d2stmat* for $n = 4$ is shown below. It must be noted that the result is obtained through the operation $H\mathcal{M}H^{-1}$, where $\mathcal{M}$ is a given complete V-matrix, and H is the transformation matrix defined through the identity matrix as shown earlier in this section in Eq. (3.34). In Maple's terminology, it is given as

```
em=evalm(d2stmat(4)&*d2cvmat(4)&*inverse(d2stmat(4)));
```

$$\left[\begin{array}{cccccccccccccccc}
e^t & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & e^t & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & e^t & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & e^t & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & e^{-t} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & e^{-t} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & e^{-t} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
e^{-t} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^t & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^t & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^t & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^t & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -e^{-t} & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -e^{-t} & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -e^{-t} & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -e^{-t} & -1 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right],$$

where $t = 2K$.

In what follows, the symbolic proof that an order-disorder actually takes place between states of finite long-range order and those with no long-range is presented. The proof, using matrix theory, consists of proving the fact that for sufficiently high temperature, the maximum characteristic value of the matrix $\mathcal{M}(K)$ no longer degenerates. For high temperature,

$K = J/2k_BT \to 0$, so $\alpha = \beta = 1$. Now using our function *d2upvmat* and for $n = 5$, a two-dimensional upper positive V-matrix is generated using 1 as the default element as explained above. The result is

$V_+(0) := d2upvmat(5, 1, 1);$

$$V_+(0) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

In the same way, using our function *d2lnvmat*, the two-dimensional lower negative V-matrix for $n = 5$ is generated. The default elements needed for the construction are 1, as is demanded by the explanation given at the beginning of the problem. From the previous and the next matrix, it is obvious why the name V-matrix is chosen.

$V_-(0) := \mathit{d2lnvmat}(5,1,1);$

$$V_-(0) = \left[\begin{array}{cccccccccccccccc}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right].$$

Using the Maple function *evalm*, if the upper positive matrix $V_+(0)$ is raised to the the fourth power, then one obtains the matrix $(V_+(0))^4$ whose elements are all ones, as is clear from the representation given below. It must also be noted that by choosing an arbitrary upper positive matrix, the power that produces this result changes. Thus for an arbitrary upper positive matrix there exists a positive number, say (r), such that raising the matrix $V_{+}(0)$ to that power gives a matrix $(V_{+}(0))^{r}$ whose elements are all ones. The resulting matrix is as

```
evalm (V_{+}(0))^4;
```

$$(V_{+}(0))^{4} = \begin{bmatrix} 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \end{bmatrix}.$$

When the same operation specified above is performed on the lower negative matrix $V_{-}(0)$, one obtains a matrix that has alternating plus ones in the odd rows and minus ones in the even rows. Also the same generalization noted for the upper positive V-matrix holds for the lower negative V-matrix. Thus there exists an arbitrary positive number r (say) such that $(V_-(0))^r$ produces a matrix that has alternate rows of plus ones in the odd rows and minus ones in the even rows. The resulting matrix is $(V_-(0))^4)$, for $r = 4$, and is evaluated as evalm $(V_-(0))^4)$.

$$(V_-(0))^4 =$$

$$\left[\begin{array}{cccccccccccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1
\end{array}\right].$$

Computing the characteristic values of the fourth power of the upper positive V-matrix, $(V_+(0))^4$, using Maple's command *eigenvals*, one obtains *eigenvals*$(V_+(0))^4$;

$$0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,16.$$

It is observed that since the trace of $(V_+(0))^4)$ is 2^4, the characteristic values of $(V_+(0))^4)$ are $(2^4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$, giving the maximum characteristic value of $V_+(0)$ as 2. Also the characteristic values of the fourth power of the negative matrix $(V_-(0))^4$ are computed using the same Maple command *eigenvals*, to give

```
 eigenvals (V_{-}(0))^4);
```

$$0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0.$$

The trace of $(V_-(0))^4$ vanishes; hence the characteristic value vanishes. For any upper V-matrix $W_+(0)$ and lower V-matrix $W_-(0)$ of an arbitrary size, there exists a positive integer r such that, since the rank of $(W_+(0))^r$ and $(W_-(0))^r$ are one, the characteristic equations reduce to

$$\lambda^{2^r} - (trace\,\mathbf{W} \pm (0)^r)\lambda^{2^r-1} = 0.$$

The trace of $(W_+(0))^r$ is 2^r; hence the characteristic values of $(W_+(0))^r$ are $2^r,\ldots,0$, giving the maximum characteristic value of $W_+(0)$ as 2, and all the rest are 0. However, the trace of $(W_-(0))^r$ is 0. Hence all the characteristic values vanish and so are the characteristic values of $W_-(0)$. This proves that for sufficiently high temperature, the maximum characteristic value of $\mathcal{M}$ is nondegenerate. Hence, it has no long-range order.

Note: The symbolic method shown here gives a good idea of how a full proof can be constructed by induction.

3.10 3-D Perturbation Method of the Ising Model

It is possible to regard the three-dimensional treatment of the Ising model of ferromagnetism as several pile of layers of the two-dimensional case. Since the matrix method of the two-dimensional Ising ferromagnet has been treated, there will be an obvious extension to the three-dimensional treatment. This has been pointed out by Oguchi [7]. We begin by analyzing the general three-dimensional matrix method. Following the notation of Montroll [24], consider a binary alloy AB whose crystal forms a simple cubic. Let the dimension of the crystal be L, M, N in units of the lattice distances. The crystal is thus made up of L layers of $M \times N$ atoms each. Each layer can be regarded as having 2^{MN} distinct configurations. The sites in each of the layers can take each of the atoms A or B. Let the $v(\alpha_j)$ be the total potential energy with respect to the coordinate α of interaction between all neighboring cells of a configuration in the jth layer, while $v(\alpha_j, \alpha_{j+1})$ is the total energy of interaction between the jth layer in configuration α_j with their nearest neighboring atoms in the $j+1$th layer in α_{j+1}. If the periodic boundary is considered by allowing the first layer to interact with layer L, the partition function takes the form

$$Z = \sum_{[\alpha_1]} \cdots \sum_{[\alpha_L]} \exp\left[\frac{-(v(\alpha_1) + \dots + v(\alpha_L) + v(\alpha_1, \alpha_2) + \dots + v(\alpha_{L-1}, \alpha_L) + v(\alpha_L, \alpha_1))}{k_B T}\right].$$

Substituting $V(\alpha_i, \alpha_j) = \frac{1}{2}v(\alpha_i) + v(\alpha_i, \alpha_j) + \frac{1}{2}v(\alpha_j)$ in the equation immediatly above we get

$$Z = \sum_{[\alpha_1]} \cdots \sum_{[\alpha_L]} \prod_{i=1}^{L} \exp\left[\frac{-V(\alpha_i, \alpha_{i+1})}{k_B T}\right], \tag{3.43}$$

with the periodic condition $\alpha_{L+1} \equiv \alpha_1$. Now set

$$\mathcal{M}(\alpha, \alpha') = \exp\left[\frac{-V(\alpha, \alpha')}{k_B T}\right].$$

Define a symmetrical matrix $\mathcal{M}$ with $2^{MN} \times 2^{MN}$ rows and columns labeled by $\alpha', \alpha'', \ldots$, in accordance with the 2^{MN} possible configurations α. The principal elements relevant to the theory are the characteristic values λ_r and the characteristic vectors ψ_r $(r = 1, 3, \ldots, 2^{MN})$. Now our interest is in solving the characteristic values problem

$$\mathcal{M}(\psi) = \lambda\psi. \tag{3.44}$$

This enables one to answer all questions of thermodynamical and statistical interest for the crystals, including the order-disorder phenomenon. A characteristic vector ψ_r has 2^{MN} components $\psi_r(\alpha)$, one for each possible configuration α. Assume that the λ_r and $\psi_r(\alpha)$ are known, then the matrix element can be written as

$$\mathcal{M}(\alpha, \alpha') = \exp[\frac{-V(\alpha, \alpha')}{k_B T}] = \sum_{r=1}^{2^{MN}} \lambda_r \psi_r(\alpha)\psi_r(\alpha'), \tag{3.45}$$

provided the ψ_r is normalized to unity,

$$\sum_{\alpha} \psi_r(\alpha)\psi_s(\alpha) = \delta_{rs}. \tag{3.46}$$

Substituting Eq. (3.45) into Eq. (3.43) and applying Eq. (3.46), it can be proved that

$$Z = \sum_{r=1}^{2^{MN}} \lambda_r^L.$$

The evaluation of Z and consequently the investigation of the thermodynamic quantities like entropy, energy, and specific heat are reduced to the evaluation of the characteristic value problem (3.44). Because the number of layers is very large for an actual crystal, we need only solve for $\lambda_{\max}$, the highest characteristic value of the matrix equation, and the corresponding characteristic vector. If $\lambda_{\max}$ is d-fold degenerate, we get

$$Z = d\lambda_{\max}^{L}.$$

This equation also shows that we can regard $\lambda_{\max}$ as the partition function per individual spin.

3.11 Extension of Kramers and Wannier V-matrix

This section deals basically with the extension of Kramers and Wannier's matrix construction. It will be appropriate here to give some details of the construction. To investigate the three-dimensional ferromagnet, one may be tempted to regard one layer as a constituent element and to pile up layers one by one. However, the matrices will be so large that, mathematically, it will be impractical to handle them. A manageable approach is to add spins one by one. For the simple cubic lattice, it can be divided into many layers. Starting from a first arbitrary position, one adds a spin beyond the one just placed previously. This construction is continued until a full line is arranged. The next line is then arranged in the same sequence until the whole arrangement is completed. Then moving to the first position in the next layer, the same process is repeated. Consider the mth layer, if the kth spin is to be added next, then since the interactions are restricted to only nearest neighbors, only the $(k-1)$th, the jth placed immediately beside it in a preceding line, and the 0th spin placed just under the kth spin in the $(m-1)$th layer interact with it as shown in Fig. 3.5 below

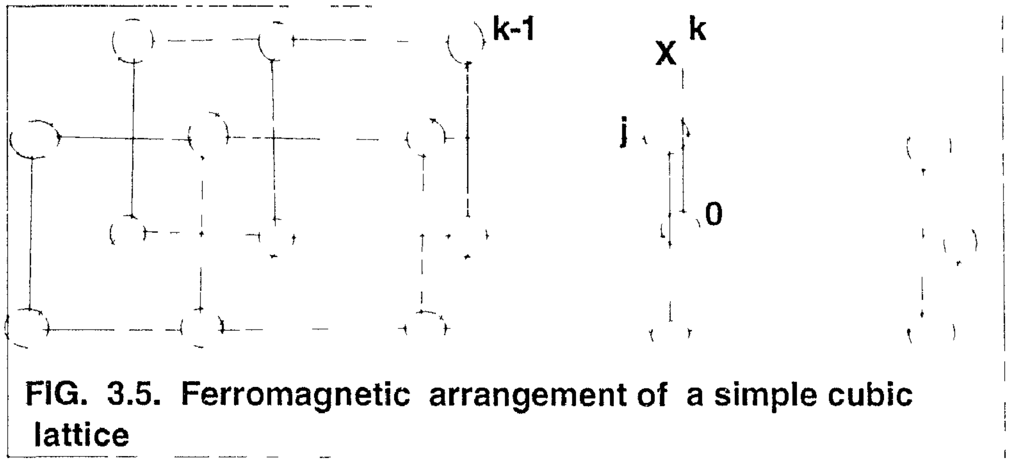


FIG. 3.5. Ferromagnetic arrangement of a simple cubic lattice

Each spin has two orientations, $\sigma_i = \pm 1$. Thus the interaction energies are $\mp\frac{1}{2}J$ for parallel and antiparallel spins, respectively. The three interactions stated above are

$$-K\sigma_k\sigma_{k-1}, \quad -K\sigma_k\sigma_j, \quad -K\sigma_k\sigma_0,$$

where $K = J/2k_BT$. Let the arrangement $\sigma_{k-1}, ..., \sigma_{j-1}, ..., \sigma_0$, have the probability $A(\sigma_{k-1}, ..., \sigma_{j-1}, ..., \sigma_0)$, and the probability $A(\sigma_k, ..., \sigma_j, ..., \sigma_1)$ be the one in which σ_1 occupies the place of σ_0, σ_2 takes the place of σ_1, etc. By the Boltzmann theorem, the probability of any particular arrangement of spins is proportional to $\exp(-E/k_BT)$, because every arrangement has weight 1. Hence the following relation is obtained:

$$\lambda A(\sigma_k, ..., \sigma_j, ..., \sigma_1) = \sum_{\sigma_0=\pm 1} \exp[K\sigma_k(\sigma_{k-1} + \sigma_j + \sigma_0)]A(\sigma_{k-1}, ..., \sigma_{j-1}, ..., \sigma_0). \quad (3.47)$$

The λ which enters into the above equation is the result of the Boltzmann exponential being proportional to probabilities. It becomes the eigenvalue of that equation. In a zero field the total energy is given as

$$E = -J/2 \sum_{<i,j>} \sigma_i \sigma_k.$$

The sum is over all pairs of (i, j) which are nearest neighbors. The partition function has the following form:

$$Z = \sum_{\sigma_1,\sigma_2 \ldots \sigma_N = \pm 1} \exp[K \sum_{i=k}^{(L-1)k-1} (\sigma_i \sigma_i + 1) + K \sum_{i=1}^{Lk} (\sigma_{j+i} \sigma_{k+i}) + K \sum_{i=1}^{Lk} (\sigma_i \sigma_{k+i})],$$

where N is the total number of spins. As in the two-dimensional case, left- and right-handed eigenvectors have to be considered be^ause the matrix is not symmetrical. Call the left-handed eigenvectors $B(\sigma_k, \ldots, \sigma_j, \ldots, \sigma_1)$. Obviously this satisfies the equation

$$\sum_{\sigma_k} \exp[K\sigma_k(\sigma_{k-1} + \sigma_j + \sigma_0)] B(\sigma_{k-1}, \ldots, \sigma_{j-1}, \ldots, \sigma_0) = \lambda B(\sigma_k, \ldots, \sigma_j, \ldots, \sigma_1). \quad (3.48)$$

After normalization, A_q and B_p satisfy the orthogonal conditions

$$\sum_{\sigma_i} B_p(\sigma_k, \ldots, \sigma_j, \ldots, \sigma_1) A_q(\sigma_k, \ldots, \sigma_j, \ldots, \sigma_1) = \delta_{pq}. \quad (3.49)$$

If the matrix operation is repeated k times, one obtains from Eqs.(3.47) and (3.48),

$$\sum_{\sigma_1 \ldots \sigma_k} \exp[K \sum_{i=1}^{2k-1} \sigma_i \sigma_{i+1} + K \sum_{i=1}^{k} \sigma_{j+i} \sigma_{k+i} + K \sum_{i=1}^{k} \sigma_i \sigma_{k+i}] = A_q(\sigma_k, \ldots, \sigma_1) = \lambda^k A_q(\sigma_{2k}, \ldots, \sigma_{k+1}) \quad (3.50)$$

and

$$\sum_{\sigma_{k+1} \ldots \sigma_{2k}} \exp[K \sum_{i=k}^{2k-1} \sigma_i \sigma_{i+1} + K \sum_{i=1}^{k} \sigma_{j+i} \sigma_{k+i} + K \sum_{i=1}^{k} \sigma_i \sigma_{k+i}] = B_p(\sigma_k, \ldots, \sigma_{k+1}) = \lambda^k B_p(\sigma_k, \ldots, \sigma_1). \quad (3.51)$$

From Eqs. (3.49), (3.50) and (3.51) one obtains

$$\exp[K \sum_{i=k}^{2k-1} \sigma_i \sigma_{i+1} + K \sum_{i=1}^{k} \sigma_{j+i} \sigma_{k+i} + K \sum_{i=1}^{k} \sigma_i \sigma_{k+i}] = \sum_{p=1}^{2^k} \lambda_p^k A_p(\sigma_{2k}, \ldots, \sigma_{k+1}) B_p(\sigma_k, \ldots, \sigma_1). \quad (3.52)$$

For the next layer, the same formula can be written as

$$\exp[K\sum_{i=2k}^{3k-1}\sigma_i\sigma_{i+1}+K\sum_{i=k+1}^{2k}\sigma_{j+i}\sigma_{k+i}+K\sum_{i=k+1}^{2k}\sigma_i\sigma_{k+i}]=$$
$$\sum_{p=1}^{2^k}\lambda_p^k A_p(\sigma_{3k},...,\sigma_{2k+1})B_p(\sigma_{2k},...,\sigma_{k+1}). \tag{3.53}$$

Multiplying Eqs. (3.52) and (3.53) and summing over the spins of the middle layer $\sigma_{2k},...,\sigma_{k+1}$, one gets

$$\sum_{\sigma_{k+1}..\sigma_{2k}}\exp[K\sum_{i=k}^{3k-2}\sigma_i\sigma_{i+1}+K\sum_{i=1}^{2k}\sigma_{j+i}\sigma_{k+i}+K\sum_{i=1}^{2k}\sigma_i\sigma_{k+i}]=$$
$$\sum_{p=1}^{2^k}\lambda_p^k A_p(\sigma_{3k},...,\sigma_{2k+1})B_p(\sigma_k,...,\sigma_1). \tag{3.54}$$

The boundary effects can be eliminated by setting $\sigma_{Lk+i}\equiv\sigma_i$, which finally yields

$$Z=\sum_{p=1}^{2k}\lambda_p^{Lk}=\sum_{p=1}^{2k}\lambda_p^N\approx\lambda_{\max}^N.$$

The perturbation solution will be considered as in the two-dimensional case to the eigenvalue problem (3.47). If the configurations are separated into two classes according to the sign of $\sigma_k=\pm1$, and the $\sigma_k=+1$ configurations are grouped with order numbers $1,2,...,2^{k-1}$, and the $\sigma_k=-1$ configurations with order numbers $2^k-1,...,2^{k-1}+1$, then the matrix can be brought to its standard form by using the binary numbering system as suggested by Kramers and Wannier. For more details on the binary system see section 3.7. This form of the matrix is called the Oguchi extended complete V-matrix of Kramers and Wannier. Or simply, the three-dimensional complete V-matrix. It was discovered that the number of submatrices needed for the correct computation of the partition function must be 1 if n is odd and 2 if n is even. The three-dimensional matrix $\mathcal{M}(K)$ is presented in the form shown below as

$$\mathcal{M}(K) = \left|\begin{array}{cccc|cccc}
P & & & & & & & \\
 & Q & & & & & & \\
 & & \cdots & & & & & \\
 & & & Q & & & & \\
 & & & & & & & S \\
 & & & & & & \cdots & \\
 & & & & & S & & \\
 & & & & R & & & \\
\hline
 & & & & P & & & \\
 & & & & & Q & & \\
 & & & & & & \cdots & \\
 & & & & & & & Q \\
 & & & S & & & & \\
 & & \cdots & & & & & \\
 & S & & & & & & \\
R & & & & & & &
\end{array}\right|,$$

where the submatrices P, Q, R, and S have the form

$$\mathbf{P} = \left|\begin{array}{cccccccc}
\zeta^3 & \zeta & & & & & & \\
 & & \zeta^3 & \zeta & & & & \\
 & & & & \cdots & & & \\
 & & & & & & \zeta^3 & \zeta \\
 & & & & & & &
\end{array}\right|, \quad \mathbf{Q} = \left|\begin{array}{cccccccc}
\zeta & \eta & & & & & & \\
 & & \zeta & \eta & & & & \\
 & & & & \cdots & & & \\
 & & & & & & \zeta & \eta
\end{array}\right|,$$

$$\mathbf{R} = \left|\begin{array}{cccccc} & & & & \eta^3 & \eta \\ & & & \cdots & & \\ & \eta^3 & \eta & & & \\ \eta^3 & \eta & & & & \end{array}\right|, \quad \mathbf{S} = \left|\begin{array}{cccccc} & & & & \eta & \zeta \\ & & & \cdots & & \\ & \eta & \zeta & & & \\ \eta & \zeta & & & & \end{array}\right|,$$

where $\zeta = \exp(K)$ and $\eta = \exp(-K)$. The order of the matrix $\mathcal{M}(K)$ is 2^k. The P, Q, R, and S are submatrices of $\mathcal{M}(K)$. They all have the same dimensions of $(2^j - 1, 2^j)$. Despite the fact that $\mathcal{M}(K)$ is very complicated, it is reducible with an orthogonal transformation similar to the two-dimensional case. Thus the two irreducible matrices obtained from $\mathcal{M}(K)$ are $\mathbf{V}_+$ and $\mathbf{V}_-$. Each has an order of 2^{k-1}. They have the form

$$\mathbf{V}_+(K) = \left|\begin{array}{ccccc} P & & & & \\ & Q & & & \\ & & \cdots & & \\ & & & P & \\ & & & & Q \\ & & & & Q \\ & & & P & \\ & & \cdots & & \\ & Q & & & \\ P & & & & \end{array}\right|, \quad \mathbf{V}_-(K) = \left|\begin{array}{ccccc} P & & & & \\ & Q & & & \\ & & \cdots & & \\ & & & P & \\ & & & & Q \\ & & & & -Q \\ & & & -P & \\ & & \cdots & & \\ & -Q & & & \\ -P & & & & \end{array}\right|.$$

The eigenvalue valid for low temperatures is obtained, first of all, by transforming $V_+(K)$ to $U_+(\eta^2)$ as

$$V_+(K) = \frac{1}{\eta^3}U_+(\eta^2) = \frac{1}{\beta^{3/2}}U_+(\beta),$$

where $\beta = \eta^2$. Next expand $U_+(\beta)$ in the form

$$U_+(\beta) = U_0 + \beta U_1 + \beta^2 U_2 + \beta^3 U_3.$$

The matrices U_0, U_1, U_2 and U_3 are constant matrices derived from $V_+(K)$. They are used in the computation of the partition function series, and they have the form given below as

$$U_0 = \left|\begin{array}{cccc} P1 & & & \\ & 0 & & \\ & & \cdots & \\ & & & 0 \\ & & & \\ & & & \\ & & & \\ & & & \end{array}\right| , \; U_1 = \left|\begin{array}{ccccc} P2 & & & & \\ & P1 & & & \\ & & \cdots & & \\ & & & P2 & \\ & & & & P1 \\ & & & & Q1 \\ & & & 0 & \\ & & \cdots & & \\ & Q1 & & & \\ 0 & & & & \end{array}\right| ,$$

$$U_2 = \begin{vmatrix} 0 & & & & \\ & P2 & & & \\ & & \cdots & & \\ & & & 0 & \\ & & & & P2 \\ & & & & Q1 \\ & & & Q2 & \\ & & \cdots & & \\ & Q1 & & & \\ Q2 & & & & \end{vmatrix}, \quad U_3 = \begin{vmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & 0 \\ & & & Q1 & \\ & & \cdots & & \\ & 0 & & & \\ Q1 & & & & \end{vmatrix},$$

where

$$P1 = \begin{vmatrix} 1 & 0 & & & \\ & & \cdots & & \\ & & & 1 & 0 \end{vmatrix}, \quad P2 = \begin{vmatrix} 0 & 1 & & & \\ & & \cdots & & \\ & & & 0 & 1 \end{vmatrix}, \quad Q1 = \begin{vmatrix} & & & 1 & 0 \\ & & \cdots & & \\ 1 & 0 & & & \end{vmatrix}, \quad Q2 = \begin{vmatrix} & & & 0 & 1 \\ & & \cdots & & \\ 0 & 1 & & & \end{vmatrix}.$$

The U_0 has a single nondegenerate eigenvalue of unity and a $(2^{k-1} - 1)$–fold degenerate value of zero. Hence the maximum eigenvalue and the maximum eigenvectors of U_+ can be expanded into the power series in β. So

$$\lambda = \lambda_{\max} = \lambda_0 + \beta\lambda_1 + \beta^2\lambda_2 + \beta^3\lambda_3 + ...,$$

and

$$\psi = \psi_{\max} = \psi_0 + \beta\psi_1 + \beta^2\psi_2 + \beta^3\psi_3 + ...$$

The eigenvalue equation at β^p is

$$U_0\psi_p + U_1\psi_{p-1} + U_2\psi_{p-2} + U_3\psi_{p-3} = \sum_{q=0}^{p} \lambda_q\psi_{p-q},$$

where ψ_1, ψ_2, and ψ_3 are zero. The perturbation method was given in detail when considering the two-dimensional case. Thus one only need emphasize that, for an infinite crystal, the left- and right-handed eigenvalues of V_+ and V_- coincide.

The equivalent program for the partition function of the three-dimensional cubic lattice has been developed. It is almost equivalent to the two-dimensional case, except in the construction of the constant matrices used in computing the series. There are four of these programs, whose names are *d3lnpps, d3lppps, d3rppps,* and *d3rnpps.* The *d3* tells us that the computation is in three dimensions. The number of arguments is two. The first is the size of the matrix used in the computation, and the second is the number of terms of the series to be computed. The programs are given in Appendix A. Four examples for each of the functions we have created are given below. The matrix used has a dimension of 32, 32, which corresponds to $n = 5$. The order of the series computed is 20.

ex2:=d3rnpps(5,20);

$$ex2 := 1+b^6-b^9+3b^{10}-3b^{11}-3b^{12}-7b^{13}+15b^{14}-8b^{15}-23b^{16}-6b^{17}+85b^{18}+33b^{19}-196b^{20}$$

ex3:=d3lnpps(5,20);

$$ex3 := 1+b^6-b^9+3b^{10}-3b^{11}-3b^{12}-7b^{13}+15b^{14}-8b^{15}-23b^{16}-6b^{17}+85b^{18}+33b^{19}-196b^{20}$$

ex4:=d3lppps(5,20);

$$ex4 := 1+b^6+b^9+3b^{10}+3b^{11}-3b^{12}+7b^{13}+15b^{14}+8b^{15}-23b^{16}+6b^{17}+85b^{18}-33b^{19}-196b^{20}$$

ex1:=d3rppps(7,38);

$$ex1 := 1 + b^6 + 1\,b^{10} - b^{12} + 24\,b^{14} - 7\,b^{16} + 130\,b^{18} - 181\,b^{20} + 643\,b^{22} - 2079\,b^{24} +$$

$$3786\, b^{26} - 18366\, b^{28} + 29180\, b^{30} - 147381\, b^{32} + 275306\, b^{34} - 1115822\, b^{36} + 2758242\, b^{38}$$

It can be observed that the coefficients of the series in the above examples are the same except for sign. One can easily obtain the eigenvectors series by using the Maple command $map(evalm, ev)$.

3.12 2-D Lattices with Many Components

The model presented here will closely follow that of Kihara et al. [5]. The model is very resourceful. It gives the connection between gr \ph coloring and the series solution of the Potts model. The formal connection will be shown when the overview of the Potts model is presented in the next section. Furthermore, this section links the computation of the series coefficients to the number of ways of coloring a given graph with a certain number of colors. The model also generalizes some of the work of Kramers and Wannier. However, they do not provide any specified method for computing the coefficients of the series. This is a great handicap of the method. The results of this model shall be compared with those of the perturbation method.

Consider a lattice of N equal lattice sites, each with an internal coordinate σ which takes s values $e_1, e_2, \ldots e_s$, corresponding to the s different states which satisfy the relation

$$e_l e_m = \begin{cases} 1 & \text{for } l = m, \\ 0 & \text{for } l \neq m. \end{cases} \tag{3.55}$$

The interaction energy takes the form

$$E = -\frac{1}{2}J\sum_{<i,k>}(2\sigma_i\sigma_k - 1).$$

Here J is the energy gained when a pair of unlike neighbors changes to a pair of like neighbors; $\sum_{<i,k>}$ means the summation is carried over all pairs (i,j) of nearest neighbors. The partition function of the system is

$$Z = \sum_{\sigma_i=e_1..e_s} \exp[K\sum_{<i,k>}(2\sigma_i\sigma_k - 1)], \tag{3.56}$$

where $K = J/2k_BT$, and $\sum_{\sigma_i=e_1..e_s}$ is the sum over all possible states of the system, which is explicitly written as

$$\sum_{\sigma_i=e_1..e_s} \equiv \sum_{\sigma_1=e_1..e_s}\cdots\sum_{\sigma_N=e_1..e_s}.$$

The Eq. (3.56) is a sum of terms, each of which is a product of factors of the type $\exp[K(2\sigma\sigma' - 1)]$ where σ, σ' are the internal coordinates of direct neighbors. The factor $\exp[K(2\sigma\sigma' - 1)]$ can be replaced by other appropriate expressions to yield the two values e^K, e^{-K}. Hence we introduce the following dual transformation:

$$e^{(2\sigma\sigma'-1)} = F_s(K)[e^{K^*} + (s\sigma\sigma' - 1)e^{K^*}]. \tag{3.57}$$

The $e^{K^*}, F_s(K)$ are determined by

$$e^{-K} = F_s(K)[e^{K^*} - e^{-K^*}],\ e^K = F_s(K)[e^{K^*} + (s-1)e^{-K^*}].$$

From which we obtain

$$(s-1)e^{-2K}e^{-2K^*} + e^{-2K} + e^{-2K^*} - 1 = 0, \tag{3.58}$$

and

$$[F_s(K)]^2 = \frac{1}{s}\frac{e^{2K} + (s-1)e^{-2K}}{e^{2K^*} + (s-1)e^{-2K^*}}. \tag{3.59}$$

Now the partition function takes the form

$$Z = [F_s(K)]^M \sum_{\sigma_i = e_1 \ldots e_s} \prod_{r=1}^{M} [e^{2K^*} + (s\sigma\sigma' - 1)e^{-K^*}]. \tag{3.60}$$

The product over r extends over all connecting rods in the lattice, and M is the total number of such connecting rods. For example, $M = 2N$ and $M = 3N$ for a square and simple cubic lattice respectively. In what follows the connection between the partition function and graph coloring will be derived. It should be noted that in developing the product of Eq. (3.60), we get the sum of e^{2K^*} and $(s\sigma\sigma' - 1)e^{-K^*}$. Each product can be characterized in the lattice by a polygon, not necessarily closed. Connecting rods not included in the polygon contribute a factor of e^{K^*} to the product, and those included in the polygon contribute a factor of e^{-K^*}. Thus we get

$$\begin{aligned} Z &= [F_s(K)]^M \sum_{\sigma_i = e_1 \ldots e_s} \sum_{polygon} \prod_{r=1}^{l} (r)(s\sigma_r\sigma_r' - 1)e^{(M-2l)K^*} \\ &= [F_s(K)]^M \sum_{polygon} \gamma(\Gamma) e^{(M-2l)} K^*, \end{aligned} \tag{3.61}$$

where l is the number of connecting rods included in the polygon Γ; $\prod_{r=1}^{l}(r)$ means the product is taken over all connecting rods of the polygon Γ; $\sum_{polygon}$ indicates the sum over all possible polygons denoted by

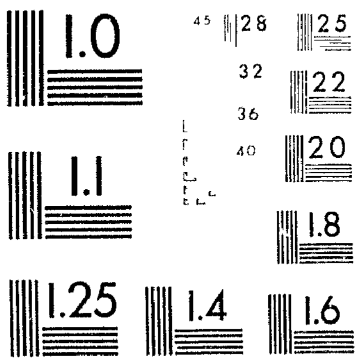
$$\gamma(\Gamma) = \sum_{\sigma_i = e_1, \ldots, e_s} \prod_{r=1}^{l} (r)(s\sigma_r\sigma_r' - 1). \tag{3.62}$$

In the product development (3.61), $\gamma(\Gamma)$ vanishes when the polygon is not closed. Let Γ consist of two polygons Γ' and Γ'' with only one common lattice site, then

$$(1/s^N)\gamma(\Gamma) = (1/s^N)\gamma(\Gamma') \times (1/s^N)\gamma(\Gamma''),$$

2

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT



and for isolated rods we have

$$\sum_{\sigma_1=e_1,\ldots,e_q}\ \sum_{\sigma_2=e_1,\ldots,e_q}(s\sigma_1\sigma_2-1)=s\times s-s^2=0.$$

Thus only closed polygons with bridges give nonzero contribution. Hence

$$Z=[F_s(K)]^M\sum_{closed\ polygon}\gamma(\Gamma)e^{(M-2l)}K^{*}. \qquad (3.63)$$

For a lattice on a simply connected surface, every closed polygon divides the surface into several regions. Thus each term in the product can be characterized by an arrangement of s states, $v_i = e_1, e_2, \ldots e_s$ in the dual net. All the lattice sites in a region are in the same state and two sites in neighboring regions are in different states, as shown in Fig. 3.6.

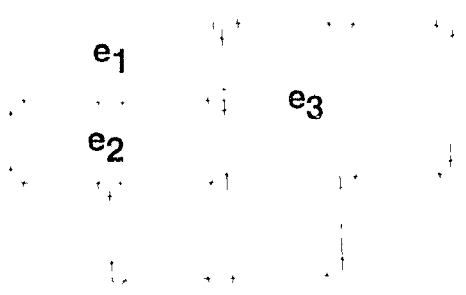


FIG. 3.6. Characterization of the product terms by arrangement of *s* states

Let $(v_1^0, \ldots, v_{N_*}^0; \Gamma)$ be one such arrangement corresponding to the polygon Γ. Writing

$$e^{(M-2l)}K^{*}=\exp\left[\sum_{j=1}^{M}K^{*}(2\nu_r^0\nu_r^{0'}-1)\right],$$

it is easily proved by induction that

$$\gamma(\Gamma)=s^{N-1}p(\Gamma).$$

The $p(\Gamma)$ is the number of different ways to paint, by s different colors (states), a map that has boundary lines corresponding to the polygon Γ. A general form of the derivation of the temperature symmetry given by Kramers and Wannier is given below. Starting from Eq. (3.63), one gets

$$\begin{aligned} Z &= [F_s(K)]^M \sum_{closed\ polygon} s^{N-1} p(\Gamma) \exp[\sum_r K^*(2\nu_r^0 \nu_r^{0'} - 1)] \\ &= [F_s(K)]^M s^{N-1} \sum_{\nu_i = \epsilon_1, \ldots, \epsilon_s} \exp[\sum_r K^*(2\nu_r^0 \nu_r^{0'} - 1)] \\ &= [F_s(K)]^M s^{N-1} Z^*, \end{aligned} \tag{3.64}$$

where $Z^*(T)$ is the partition function for the dual net, and T^* the dual temperature, given by $K^* = J/2k_B T^*$. The equation

$$\frac{Z}{s^{N/2}[e^{2K} + (s-1)e^{-2K}]^{M/2}} = \frac{Z^*}{s^{N^*/2}[e^{2K^*} + (s-1)e^{-2K^*}]^{M/2}}$$

is obtained by substituting Eq. (3.59) in Eq. (3.64), where N^* is the total number of lattice sites of the dual net, and $N + N^* = M + 2$. Euler's theorem has been applied.

For a square lattice, one can deduce the following relation:

$$Z^* = Z, \quad N^* = N \quad M = 2N,$$

from which we get the relation

$$\frac{Z}{[e^{2K} + (s-1)e^{-2K}]^N} = \frac{Z^*}{[e^{2K^*} + (s-1)e^{-2K^*}]^N}. \tag{3.65}$$

This indeed is the generalized form of Kramers-Wannier's relation of the temperature symmetry for the square net. Setting

$$\lambda(K) = \frac{Z^{1/N}}{e^{2K} + (s-1)e^{-2K}},$$

we get

$$\chi(K) = \chi(K^*). \tag{3.66}$$

The energy of the system is given as

$$E(K) = -NJ\left[\frac{e^{2K} - (s-1)e^{-2K}}{e^{2K} + (s-1)e^{-2K}} + \frac{1}{2}\frac{d}{dK}\ln\chi(K)\right], \tag{3.67}$$

and the specific heat is given as

$$C(K) = NkK^2\left[\frac{16(s-1)}{[e^{2K} + (s-1)e^{-2K}]^2} + \frac{d^2}{dK^2}\ln\chi(K)\right]. \tag{3.68}$$

It is clear from Eq. (3.66) that if $\chi(K)$ has a singular point at a temperature T_0, then it must also have a singular point at T_0^* dual to T_0. Hence if there exists only one singular point of $\chi(K)$, it must occur at T_c for which $K^* = K$. It follows from Eq. (3.58) that the transition temperature (Curie temperature) is given by

$$(s-1)e^{-4K_c} + 2e^{-2K_c} - 1 = 0, \tag{3.69}$$

from which we get

$$e^{2K_c} = 1 + \sqrt{s}, \tag{3.70}$$

where $K_c = J/2k_BT_c$. From Eq. (3.58) the following relation holds at the Curie temperature

$$\left(\frac{dK^*}{dK}\right)_{K=K_c} = -1, \quad \left(\frac{d^2K^*}{dK^2}\right)_{K=K_c} = \frac{1}{\sqrt{s}}.$$

Hence from Eq. (3.66)

$$\left(\frac{d\chi}{dK}\right)_{K_c+0} + \left(\frac{d\chi}{dK}\right)_{K_c-0} = 0, \tag{3.71}$$

and

$$\left(\frac{d^2\chi}{dK^2}\right)_{K_c+0} - \left(\frac{d^2\chi}{dK^2}\right)_{K_c-0} = \frac{2}{\sqrt{s}}\left[\left(\frac{d\chi}{dK}\right)_{K_c+0} - \left(\frac{d\chi}{dK}\right)_{K_c-0}\right]. \tag{3.72}$$

We can use Eq. (3.67) and Eq. (3.71) to compute the arithmetic mean, $E(K+0)$ and $E(K-0)$, as

$$E(K) = -N\frac{J}{\sqrt{s}}.$$

When the energy is continuous at the Curie temperature, $E(K)$ becomes the total energy of the system. It follows from Eq. (3.67), Eq. (3.68) and Eq. (3.72) that if the energy is continuous at the Curie temperature, the specific heat is either continuous or infinite.

Now consider the partition function for the square lattice. The partition function (3.56) can be expressed in the form

$$Z = \sum_i g(i)\exp(-E_i/k_B T).$$

Here g_i is the number of states of the system at the energy level E_i. The $\sum_i$ is taken over all energy levels. At low temperatures, corresponding low-energy terms predominate. At the lowest energy level, all the lattice sites are in a state such that $E_0 = NJ$ and $g_0 = s$. At the first excited level, a single site is in another state so $E_1 = NJ + 4J$ and $g_1 = N(s-1)s$. We may proceed in this way to find a power series expansion valid for low temperatures. However, we must note that the computation of the g_i at energy level E_i is difficult. We have not been able to derive any mathematical formula to compute this. The low-temperature series expansion has the form

$$Z = x^N[s + N(s-1)sx^4 + 2N(N-1)sx^6 + ...],$$

with $s \equiv e^{J/k_B T}$. The Nth root can be taken to obtain

$$[Z]^1/N = x^{-1}P(x),$$

where

$$P(x) = 1 + (s-1)x^4 + 2(s-1)x^6 + \ldots = 1 + (s-1)\sum_{i=4}^{N} a_i x_i. \qquad (3.73)$$

The coefficients a_i for $4 \leq i \leq 16$ are given in the symbolic computation. By virtue of the symmetry relation (3.65), the high temperature expansion can be deduced from the low-temperature expansion:

$$Z = \frac{s}{(1-u)(1+(s-1)u)} P(u),$$

where $u = \frac{1-s}{1+(s-1)x}$ and $P(u)$ is the same as in Eq. (3.73). The energy per site can be computed from these two expansions as

$$E/N = k_B T^2 \frac{\partial}{\partial T}[\ln Z^{1/N}],$$

and the specific heat per site as

$$C_v/N = 1/N \frac{\partial E}{\partial T}.$$

Here again we developed the function that computes the series expansion of the work presented in this section. Two functions have been developed. They compute the series expansion for the partition function, energy, and the specific heat for the many-component states. The names of the programs are *d2mcltpf* and *d2mchtpf*. They are taken from two-dimensional many components low (high)-temperature partition function. The function takes one argument. However, there are two additional optional arguments. The first argument is the order of the series to be computed. The second argument is the number of components or the number of states. The last argument is the variable which one wants to use in the series computation. An example of high temperature series expansion is given below in Maple's format:

```
pf:=d2mchtpf(8,2,t);
```

$$pf := 1 + \frac{(1-t)^4}{(1+t)^4} + \frac{2\,(1-t)^6}{(1+t)^6} + \frac{5\,(1-t)^8}{(1+t)^8}\left(1 - \frac{1-t}{1+t}\right)^{-1}.$$

Maple can be used to simplify the above partition function to get

```
>pf= simplify(pf);
```

$$pf = \frac{9}{-}\ \frac{40\,t - 216\,t^3 + 406\,t^4 - 216\,t^5 + 172\,t^6 - 40\,t^7 + 172\,t^2 + 9\,t^8}{2\,(1+t)^7\,t}.$$

The following examples are given for the low-temperature series expansion. The successive optional arguments are used for a four-component state. When no component state is specified, it is assumed to be a two-component state.

```
pf1:=d2mcltpf(8);
```

$$pf1 := 1 + e^{-\frac{J}{k_B T}}{}^{4} + 2\,e^{-\frac{J}{k_B T}}{}^{6} + 5\,e^{-\frac{J}{k_B T}}{}^{8}\,e^{-\frac{J}{k_B T}}{}^{-1}$$

```
>spf1= d2mcltpf(8,4);
```

$$spf1 = 1 + 3\,e^{-\frac{J}{k_B T}}{}^{4} + 6\,e^{-\frac{J}{k_B T}}{}^{6} + 12\ e^{-\frac{J}{k_B T}}{}^{7} + 3\,e^{-\frac{J}{k_B T}}{}^{8}\,e^{-\frac{J}{k_B T}}{}^{-1}$$

```
> tpf1 = d2mcltpf(8,4,x);
```

$$tpf1 = \frac{1 + 3\,x^4 + 6\,x^6 + 12\,x^7 + 3\,x^8}{x}.$$

3.13 Comparison of the Results

In this section the results are validated by comparing them with those of others who have done the same computation or similar computations in this area. This enables us to see how our symbolic computation results agree with theirs.

Table 3.1 shows some of my results for the two-dimensional Ising model computed for the 5-screw and the 9-screw case for a series of order 20, compared with Domb computation for $n = 5$. The first row of the 2-d table in Table 3.1, gives the order of the series whose coefficients are compared. The next two rows give the coefficients of our symbolic computation for $n = 5$ (S Co(5)) and $n = 9$ (S Co(5)). The fourth row shows the coefficients of Domb's computation for $n = 5$. Domb's method is very similar to the method used in my symbolic computation. Hence it is seen that the coefficients are the same, attesting to the correctness of the symbolic method.

Var	x^0	x^4	x^6	x^8	x^{10}	x^{12}	x^{14}	x^{16}	x^{18}	x^{20}
S Co(5)	1	1	3	15	48	70	-253	-3118	-18680	-75512
S Co(9)	1	1	2	5	15	80	598	3436	13141	28995
D Co(5)	1	1	3	15	48	70	-253	-	-	-

Table 3.1. *Comparison of the coefficients for 2-d Ising model*

Also the symbolic computation can compute much more than what Domb computed. In fact the symbolic computation is not limited to the order of the series which one needs to compute. The eigenvector series are also computed along-side of the partition function series, but Domb did not show the computation of the eigenvector series. It is important to note that as n increases the symbolic computation coefficients tend to the exact value or a limiting value.

Var	x^0	x^4	x^6	x^8	x^{10}	x^{12}	x^{14}	x^{16}
K&M& Co	1	1	2	5	14	44	152	518

Table 3.2. *Coefficients of 2-d Ising model by Kihara, Midzuno and Shizume*

Table 3.2 shows the coefficients of Kihara, Midzuno and Shizume. Their method considers the state of the lattice sites at the lowest energy level, then the first excited energy level in which a single site is in another state, etc. Continuing in this manner the coefficients of the partition function are computed. This computation tends to be closer to the limiting value than the matrix approach used in the symbolic computation. However, one cannot control the lattice size when using the method of Kihara, Midzuno and Shizume. Also the magnetization cannot be analyzed by their method because it does not contain an external magnetic field. Also their method is limited to the order of the series due to the complexity of the computation. If one has large computer resources then the symbolic computation is the best method. The reason is that, apart from the partition function, one can also use the eigenvector series to analyze the magnetization, propagation of order, and long-range order in crystals. However, if one is interested only in the limiting value of the partition function without an external magnetic field, then the Kihara, Midzuno and Shizume method seems best. For $n = 9$ my result agrees with the Kihara, Midzuno and

Shizume coefficients to the 8th order of the series.

Table 3.3 shows the symbolic computation coefficients of the partition function of the three-dimensional Ising model for the 7-screw and the 9-screw case for a series of order 22, compared with the coefficients of the computations of Wakefield and Oguchi.

Var	S Co(7)	S Co(9)	W Co	O Co
x^0	1	1	1	1
x^6	1	1	1	1
x^{10}	3	3	3	3
x^{12}	-2	-3	-3	-3
x^{14}	21	15	15	15
x^{16}	-9	-27	-30	-30
x^{18}	139	94	101	101
x^{20}	-148	-216	-261	-
x^{22}	21	685	807	-

Table 3.3.*Comparison of the coefficients for 3-d Ising model*

The first column of the table above gives the order of the series whose coefficients are compared. The next two columns give the coefficients of our symbolic computation for $n = 7$ (S Co(7)) and $n = 9$ (S Co(9)). The fourth column shows the coefficients of Wakefield [25] (*WCo*), whose method is similar to that used by Kihara, Midzuno and Shizume. The last column shows the coefficients of Oguchi (O Co). His method may be similar to Wakefield's. Any missing terms in column 1 means the coefficients are zero. As n increases the symbolic computation coefficients tend to the limiting value. The symbolic computation can compute as many terms as needed, but all the other methods are limited due to the complexity of the computation. As the number of terms increases the computation becomes so tedious that symbolic computation becomes the only adequate method. The missing coefficients in the last column means that they were computed

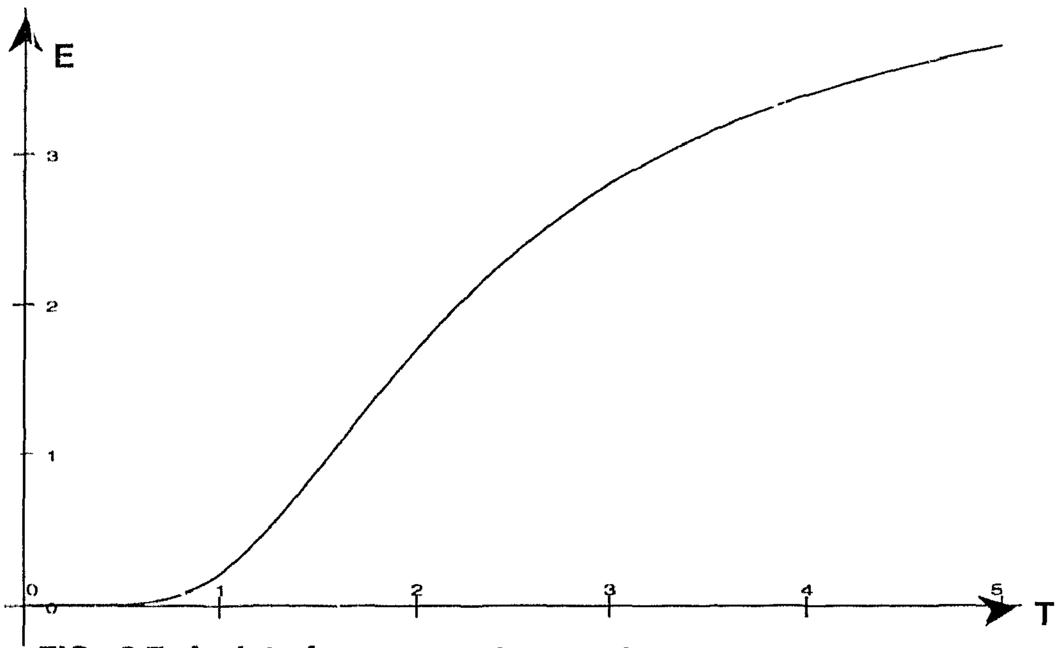


FIG. 3.7. A plot of energy vs. temperature

only to the indicated order. When one wants to consider finite lattices, then the symbolic computation is the best method, since any lattice size can be considered and analyzed. Also if one wants to analyze magnetization and propagation of order in a crystal as well as long-range order in crystals, the symbolic computation is the best method. From Table 3.3 it is clear that for $n = 7$ the symbolic computation agrees with the work of Wakefield and Oguchi to the 10th order of the series. For $n = 9$ the symbolic computation agrees with the work of Wakefield and Oguchi to the 14th order of the series. Thus as n increases the symbolic computation coefficients become closer to the coefficients of Wakefield and Oguchi. Hence, if one needs only the limiting value of the partition function, the method of Wakefield and Oguchi seems closer to the limiting value.

This section ends with two plots for the 3-screw case. The plot of energy against temperature for the 3-screw case is shown in Fig. 3.7, the plot shown in Fig. 3.8. is the energy against inverse temperature for the 3-screw. The shape is comparable to what is given by Kramers and Wannier [6] in Fig. 7.

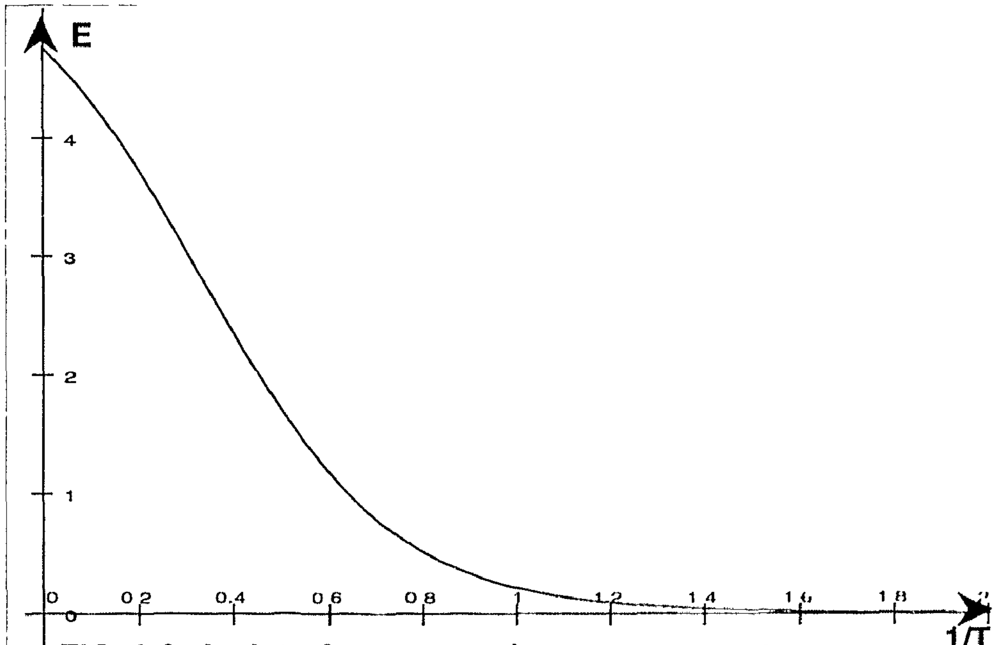


FIG. 3.8. A plot of energy vs. inverse temperature

3.14 The Potts Model

The original problem that Domb proposed was to regard the Ising model as a system of interacting spins that can be either parallel or antiparallel.

Then an appropriate generalization would be to consider a system of spins confined in a plane, with each spin pointing to one of the q equally spaced directions specified by the angles

$$\Theta_n = 2\Pi n/q, \qquad n = 0, 1, \ldots, q-1. \tag{3.74}$$

Generally, the form of the nearest-neighbor interaction depends only on the relative angle between the two vectors. This is commonly known as a system of $Z(q)$ symmetry whose Hamiltonian is given below by

$$\mathcal{H} = -\sum_{<i,j>} J(\Theta_{i,j}), \tag{3.75}$$

where the function $J(\Theta)$ is 2π periodic and $\Theta_{ij} = \Theta_{n_i} - \Theta_{n_j}$, the angle between the two spins at neighboring sites i and j. The $Z(q)$ model plays an important role in the lattice gauge theories and has attracted growing interest. The model suggested by Domb [3] (Potts [26]) is to choose

$$J(\Theta_{ij}) = -\epsilon_1 \cos(\Theta_{ij}). \tag{3.76}$$

Using analysis of the Kramers and Wannier [6] type, Potts was able to determine the critical point of this model on the square lattice for $q = 2, 3, 4$. Although unable to extend this finding to $q > 4$, Potts [27, 28] reported, as a remark at the end of his paper, the critical point for all q of the following model:

$$J(\Theta_{ij}) = -\epsilon_2 \delta_{K_r}(n_i, n_j). \tag{3.77}$$

It is a q-component model that has attracted the most attention.

Following the suggestion of Domb [29-31], the model given by Eq. (3.76) will be named the planar Potts model, and the model given by the

Eq. (3.77) is called the standard Potts model, or simply the Potts model. Other names for these models have also appeared in the literature. The planar Potts model has been referred to as the vector Potts model and also as the clock model in recent literature; the standard Potts model has often been called the Ashkin-Teller-Potts model, for historical reasons. It appears that Domb's suggestion is simplest, and should be adopted in conjunction with using the name of the Ashkin-Teller [32] model for the four-component model with (and without) symmetry breakings.

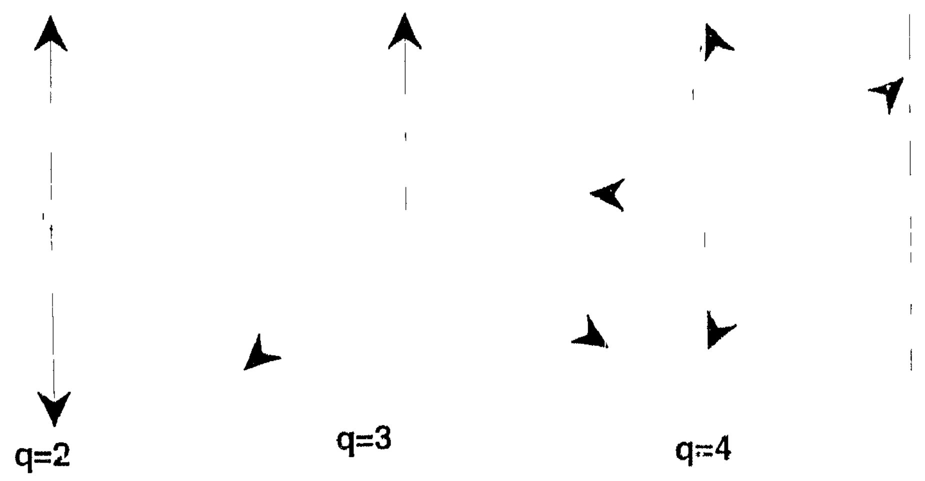


FIG. 3.9. The vectors pointing in the q-symmetric direction of a hypertetrahedron in $q-1$ dimensions

The (standard) Potts model is ferromagnetic if $\epsilon_2 > 0$ and antiferromagnetic if $\epsilon_2 < 0$. The interaction shown in Eq (3.77) can be alternately formulated to reflect its full symmetry in a $q-1$-dimensional space. This is achieved by writing, in Eq. (3.77),

$$\delta_{K_r}(\alpha, \beta) = \frac{1}{q}[1 + (q-1)e^{\alpha}e^{\beta}], \tag{3.78}$$

where $e^{\alpha}, \alpha = 0, 1, ..., q-1$ are unit vectors pointing in the q-symmetric directions of a hypertetrahedron in $q-1$ dimensions, and δ_{K_r} is the Kronecker delta function. The three figures shown in Fig. 3.9 are examples of these vectors for $q = 2$, $q = 3$, and $q = 4$, respectively. The Hamiltonian in the form of (3.77) and (3.78) has proved convenient to use in the continuous-spin formulation of the Potts model. The planar and standard models are identical for $q = 2$ (Ising) and $q = 3$ with $\epsilon_2 = 2\epsilon_1$ and $\epsilon_2 = 3\epsilon_1/2$, respectively. Also, the four-state planar model is reducible to the $q = 2$ models (Betts, [33, 34]) and this equivalence is valid for arbitrary lattices (Kasteleyn, [35, 36]). No apparent relations exist between the planar and standard models for $q > 4$.

In addition to the two-site interactions, there can be multisite interactions as well as external fields. For a Potts model on a lattice G of N sites, the Hamiltonian $\mathcal{H}$ generally takes the form

$$\beta\mathcal{H} = L\sum_{i}\delta_{K_r}(\delta_i, 0) + K\sum_{i,j}\delta_{K_r}(\delta_i, \delta_j) + K_3\sum_{i,j,k}\delta_{K_r}(\delta_i, \delta_j, \delta_k) + ...,$$

where $\beta = 1/k_BT$, and $\sigma_i = 0, 1, ...q-1$ specifies the spin states at the ith site and

$$\begin{aligned}\delta_{K_r}(\delta_i, ..., \delta_k) &= 1 \qquad if\ \sigma_i = ... = \sigma_k \\ &= 0, \quad otherwise.\end{aligned}$$

Here $K = \beta\epsilon_2$, K_n $n \geq 3$ is the strength of the n-site interactions, and L is an external field applied to the spin state 0. The partition function is

$$Z_G(q, L, K, K_n) = \sum_{\sigma_i=0}^{q-1} e^{-\beta H}. \tag{3.79}$$

The physical properties of the system are derived in the usual way by taking the thermodynamic limit. Relevant thermodynamic quantities include the per site "free energy"

$$f(q, L, K, K_n) = \lim_{N\to\infty} \frac{1}{N} \ln Z_G(q, L, K, K_n), \tag{3.80}$$

the per site energy

$$E(q, L, K, K_n) = -\frac{\partial}{\partial\beta} f(q, L, K, K_n), \tag{3.81}$$

and the per site "magnetization,"

$$M(q, L, K, K_n) = -\frac{\partial}{\partial L} f(q, L, K, K_n). \tag{3.82}$$

The order parameter m, which takes the values 0 and 1 for completely disordered and ordered systems, respectively, is defined to be

$$m(q, L, K, K_n) = (qM - 1)/(q - 1). \tag{3.83}$$

A ferromagnetic transition is then accompanied with the onset of a spontaneous ordering

$$m_0 = m(q, 0+, K, K_n). \tag{3.84}$$

As shown in chapter 2, the critical exponents $\alpha, \alpha', \beta, \gamma, \gamma'$... can be defined in the usual fashion from a singular behavior of these thermodynamic quantities near the critical temperature T_c. The two-point correlation function $\Gamma_{\alpha,\alpha}(r_1, r_2)$ of the zero-field Potts model is

$$\Gamma_{\alpha,\alpha}(r_1, r_2) = P_{\alpha,\alpha}(r_1, r_2) - q^{-2}, \tag{3.85}$$

where $P_{\alpha,\alpha}$ is the probability that the sites at r_1 and r_2 are both in the same spin state α. Clearly, $\Gamma_{\alpha,\alpha}$ takes the respective values 0 and $(q-1)/q^2$ for completely disordered and completely ordered systems. This then suggests the following relation between the large distance correlation and the spontaneous ordering:

$$\lim_{|r_1-r_2|\to\infty} \Gamma_{\alpha,\alpha}(r_1, r_2) = (q-1)(m_0/q)^2.$$

Indeed, the relation above, which first appeared as a footnote in Potts and Ward [37], for $q = 2$, can be established by a decomposition of the correlation function into those of the extremum states (Kunz and Wu , [38]). It has also been established rigorously that $\Gamma_{\alpha,\alpha}$ decays exponentially above the critical temperature T_c. The decay of $\Gamma_{\alpha,\alpha}$ for $T \leq T_c$ is not known except for $q = 2$. Furthermore, surface tension for the generalized Potts model has been discussed by Fontaine and Gruber [39]. It can be shown that, in two dimensions, the surface tension is related to the two-point correlation function of the dual model. The wide application of the Potts model is its close relation to the problem of graph colorings, so it is useful to introduce here the needed definitions. Let $P_G(q)$ be the number of ways that the vertices of a graph G can be colored in q which is known as the chromatic function for the graph G. Consider next an antiferromagnetic Potts model on G with pure two-site interactions $K < 0$. Consider further the zero-temperature limit of $K \to -\infty$. It is clear that in this limit the partition function (3.79) reduces to

$$Z_G(q, K \to -\infty) = P_G(q). \tag{3.86}$$

This simple connection between the Potts partition function and the chromatic function is valid for G in any dimension. In addition, a graphical interpretation of $q = -1$ has been given by Stanley [40, 41]. For a lattice G of N sites, the free energy Eq. (3.80) in the zero-temperature limit of $K \to -\infty$ becomes the ground state entropy

$$W_G \equiv \lim_{N \to \infty} \frac{1}{N} \ln P_G(q). \tag{3.87}$$

The existence of this limit has been discussed by Biggs [42] using the technique of graph coloring. In particular he showed that the chromatic limit exists. However, there are three exact results on $W_G(q)$ for $q \geq 3$. These are the values for the $q = 3$ square lattice, $q = 4$ triangular lattice:

$$\begin{aligned} W_{sq}(3) &= (4/3)^{\frac{3}{2}} \\ W_{tri}(4) &= \prod_{n=1}^{\infty} \left[\tfrac{(3n-1)^2}{3n(3n-2)}\right] \\ W_{Kagome}(3) &= [W_{tri}(4)]^{\frac{1}{3}} \end{aligned}$$

We have designed the Maple program that computes these physical quantities. It is known from the above that these quantities are basically the ground state entropy, so the name of the program is *grsten*. It has two arguments and a third optional argument. The limit of the square lattice with $q = 3$ (i.e. coloring with three colors or equivalently each spin pointing to one of the three equally spaced directions) is constant. Only q and square (*sq*) need to be specified. However, for the triangular (*tri*) and the Kagome lattice, one needs to specify n, the number of terms of the series. The advantage of the program is that any desired precision can be set. The program is given in the Appendix A. Examples are given below. In Example 1, we compute the Potts ground state entropy for a square lattice for $q = 3$.

Example 1

```
grsten(3,sq);
```

$$W\_sq(3) = \frac{4}{9}\sqrt{4}\sqrt{3}$$

In Example 2, we compute the Potts ground state entopy for a triangular lattice, for $q = 3$. To get a higher precision the digit is set to 200. The value is obtained by using the *evalf* function in Maple to obtain the floating point value.

Example 2

```
>evalf(grsten(4,200,tri));
```

$$W\_tri(4) = 1.4601877225626119272$$

In Example 3, we compute the Potts ground state entropy for $q = 3$ on a Kagome lattice. The digit is set to 100 and the value is evaluated automatically by using Maple's evalf function.

Example 3

```
> evalf(grsten(3,100,kagome));
```

$$W\_kagome(3) = 1.1342862673811495332$$

The programs for these constants are given in the Appendix A.

Chapter 4

DESCRIPTION OF THE SIMULATION CODE

4.1 Introduction

The Ising model is considered to be made up of part of a system which is large enough for statistical concepts to be useful. Thus we expect the forces which make the connections to be sufficiently weak. To do this construction within the context of discrete energy levels, consider a large number N of identical Ising models of $\mathcal{M}$ rows and $\mathcal{N}$ columns. Connect them together by infinitely weak forces that enable the model to exchange energy but do not contribute to the total energy of the system. Our interest is in one of the Ising models, while the others serve to define the temperature. We know that a collection of such systems is called an ensemble. Let the total energy of the system be E_T, but the energies of each of the Ising model that make it up are unknown. Then all such distributions of the total energy are equally probable. An ensemble with this probability function is called a microcanonical ensemble. Suppose σ_n is the set of variables that specify the nth Ising model and let $\mathcal{E}_n\sigma_n$ be the energy corresponding to σ_n. If the total energy E_T is known, then the probability that the N Ising models have the configurations specified by $\sigma_1, ..., \sigma_n$ is

$$P(\sigma_1, ...\sigma_n, E_T) = \frac{\delta_{E_T,\mathcal{E}_T}}{\Omega(E_T)}. \tag{4.1}$$

Here

$$\mathcal{E}_T = \sum_{n=1}^{N} \mathcal{E}_n(\sigma_n),$$

where $\delta_{j,j'}$ is the Kronecker's delta, which is 1 if $j = j'$ and 0 otherwise, and

$$\Omega(E_T) = \sum_{\sigma_1} ... \sum_{\sigma_n} \delta_{E_T,\mathcal{E}_T},$$

where $\sum_{\sigma_n}$ is the summation over all configurations σ_n, of the nth Ising model. Since we are interested in a particular Ising model, we need to find the probability $P(\sigma_1; E_T)$ that one Ising model is in some particular state σ_1 while the rest may be in any state subject to the fact that the energy E_T is constant. This probability is easily computed from Eq. (4.1) as [43]

$$P(\sigma; E_T) = \frac{\sum_{\sigma_1} \cdots \sum_{\sigma_n} \delta_{E_T,\mathcal{E}_T}}{\Omega(E_T)}. \tag{4.2}$$

If we take the limit as $N \to \infty$, it will correspond to a physical situation of having one small Ising model attached to an external system or heat bath. A collection of Ising models with a probability determined by Eq. (4.2) in the limit as $N \to \infty$ is called a canonical ensemble.

4.2 Temperature Measurement and Microcanonical Simulation

Our numerical simulation procedure, described in section 4.3, should generate configurations of the spin system (denoted by C) and of the demon system (denoted by c) with equal probability and with constant total energy E_T. The average over passes of any observable $\Theta(C,c)$ should approach the following ensemble average:

$$\Theta(E_T) = \frac{\sum_{C,c} \delta_{E_s(C)+4E_d(c),E_T} \Theta(C,c)}{\sum_{C,c} \delta_{E_s(C)+4E_d(c),E_T}}. \tag{4.3}$$

Assume $g(E_d)$ and $G(E_s)$ to be the number of states of the demons and the spin system at energy E_d and E_s, respectively. Then

$$\sum_{C,c} \delta_{E_s(C)+4E_d(c),E_T} = \sum_{E_s,E_d} G(E_s) g(E_d) \delta_{E_s+4E_d,E_T}. \tag{4.4}$$

In the absence of demons, the microcanonical average may be defined by

$$\Theta(E) = \frac{\sum_C \delta_{E_s(C),E} \Theta(C)}{\sum_E G(E)}. \tag{4.5}$$

The exact known solution for the Ising model gives an analytic expression for canonical averages [44-46]:

$$\langle \Theta \rangle_\beta = \frac{\sum_E G(E) e^{-\beta E} \Theta(E)}{\sum_E G(E) e^{-\beta E}}. \tag{4.6}$$

Therefore, one has to relate the measured averages of Eq. (4.3) to the predicted averaged of Eq. (4.6). The average of Eq. (4.5) is an intermediate step. With these observations, I derive a general expression that relates the expectation value of the demon energy in an n-dimensional model. This enables one, in microcanonical simulation that uses demons, to measure the temperature or the inverse temperature.

In the microcanonical approach the total energy of the system (spin plus demon) is held fixed, while the temperature of the system is a derived quantity. We measured the temperature by measuring the average energy of the demons. From the point of view of each demon the rest of the system is a large heat bath, and the possible states of the demon will therefore be distributed canonically according to the temperature $T = 1/\beta$ of the bath.

One of my contributions is the following theorem which gives a general expression for the determination of the temperature for simulations that uses the demons approach.

Theorem 1 *Let $Z = e^{-4\beta}$, then the average demon energy is given by the expression*

$$< E_d >= n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}, \tag{4.7}$$

for any n nonzero possible changes in the energy states of the system or the possible values of the demons. This equation can also be used in microcanonical simulation to obtain the inverse temperature β by numerical inversion.

Proof of Theorem. For an n-dimensional model each spin has $2n$ nearest neighbors. If an Ising spin is flipped in a zero magnetic field, the minimun nonzero decrease in energy is $4J$. Thus the possible changes in the energy of the system upon flipping a spin are $0, \pm 4 \ \pm 8, ..., \pm 4n$. Each demon is allowed $n + 1$ energy states with values $0, \ 4 \ 8, ..., 4n$. Removing the factor 4, the allowed energy states of the demons becomes $E_d = 0, \ 1 \ 2, ..., n$. The demons become thermalized after a number of passes through the lattice. On a large system the values of the kinetic variable should become exponentially distributed with the Boltzmann weights corresponding to the temperature $T = 1/\beta$ of the system. Hence the probability for the demon to have energy E_d is given by

$$P(E_d) = \frac{1}{Z}\exp(-E_d\beta), \tag{4.8}$$

where Z is the normalization constant such that the sum over all states of the demon is 1.

Thus the expectation value of E_d or the average demon energy which gives a means of measuring the temperature is given as

$$4 < E_d >= \frac{\sum_{b=0}^{n} 4be^{-4b\beta}}{\sum_{b=0}^{n} e^{-4b\beta}}, \tag{4.9}$$

where E_d represents the demons or the momentum variable conjugate to the spins. Using the transformation $Z = e^{-4\beta}$, write Eq. (4.9) as

$$< E_d >= \frac{Z + 2Z^2 + ... + nZ^n}{1 + Z + Z^2 + ... + Z^n}. \tag{4.10}$$

Multiply the numerator and denominator of Eq. (4.10) by $1 - Z$ and write the numerator as $(1 - Z)[n + nZ + ... + nZ^n - n - (n-1)Z - ... - Z^{n-1}]$. Thus one can simplify the denominator to obtain

$$< E_d >= \frac{[n(1 - Z)(1 + Z + Z^2 + ... + Z^n)] - [(1 - Z)(n + (n-1)Z + ... + Z^{n-1})]}{1 - Z^{n+1}}. \tag{4.11}$$

By writing $(1-Z)(n+(n-1)Z+...+Z^{n-1})$ as $1-Z^n+1-Z^{n-1}+...+1-Z^2+1-Z$, Eq. (4.11) takes the form

$$< E_d >= \frac{n(1 - Z^{n+1}) - (n+1) + (1 - Z^{n+1})/(1 - Z)}{1 - Z^{n+1}}. \tag{4.12}$$

Simplifying Eq. (4.12) one obtains the required equation (4.7).

$$< E_d >= n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$$

We derived the following corollary that gives a simpler formula for computing the inverse temperature, it the number of possible states of the demons is large.

Corollary 1 *Let n be the number of the possible states of the demons. Let $|Z| < 1$, then the expression for the average demon energy, which is given by*

$$< E_d >\approx \frac{1}{1-Z} - 1, \tag{4.13}$$

can be used in the microcanonical simulation that uses the demon approach to compute the inverse temperature $\beta = 1/T$ numerically.

Proof of Corollary.

Assume $\beta > 0$, then considering the last expression of Eq. (4.7), $|Z|$ is less than 1. Thus $Z < 1$ so $|Z^{n+1}|$ tends to zero if n is large. Hence

$$\frac{-(n+1)}{1-Z^{n+1}} \approx -(n+1).$$

Substituting this approximation in Eq. (4.7) we get

$$< E_d >\approx \frac{1}{1-Z} - 1,$$

which is the required result.

The results of computing the temperature (T) at various iteration points, for 8 possible states of the demon energy (E_d) (i.e., $n = 7$) are shown in Table 4.1. The first column gives the two demon energy expressions (E_d), the second column is the temperature (T) computation using

(E_d) expressions. The third column is the absolute value of the difference in temperature obtained from using the two demon energy expressions to compute the temperature. The fourth column is the number of iterations, (i) used in determining the temperature. The various iteration points are chosen arbitrarily.

E_d	T	$\|\Delta T\|$	i	T	$\|\Delta T\|$	i
$\frac{1}{1-Z} - 1$	1.81697	0.00001	20	1.82885	0.00001	40
$n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$	1.81698		20	1.82884		40
$\frac{1}{1-Z} - 1$	1.82529	0.00001	60	1.81833	000002	80
$n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$	1.82528		60	1.81835		80
$\frac{1}{1-Z} - 1$	1.82459	0 .00002	100	1.81425	0 .00001	120
$n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$	1.82457		100	1.81426		120
$\frac{1}{1-Z} - 1$	1.82109	0.C0001	140	1.81971	0.00002	200
$n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$	1.82108		140	1.81969		200
$\frac{1}{1-Z} - 1$	1.80497	0	400	1.81425	0.00001	1000
$n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$	1.80497		400	1.81426		1000

Table 4.1. *Simulation results comparing the two energy expressions,* $\frac{1}{1-Z} - 1$ *and* $n - \frac{n+1}{1-Z^{n+1}} + \frac{1}{1-Z}$.

The important observation about expression (4.13) is that it reveals that the inverse temperature is independent of the number of possible states n. This is true in the three-state Potts model we are considering. By using each of the two Eqs. (4.13) and (4.7) separately in our temperature measurement routine, almost equivalent results were obtained in each case as shown in Table 4.1.

For our model under consideration, since there are eight possible states for our demons, the average demon energy takes the form

$$< E_d >= \frac{\sum_{b=0}^{7} 4be^{-4b\beta}}{\sum_{b=0}^{7} be^{-4b\beta}}.$$

Using our theorem for $n = 8$ we obtain

$$< E_d >= 7 - \frac{8}{1 - Z^8} + \frac{1}{1 - Z}.$$

The above expression gives β as a function of the average demon energy. By measuring the average demon energy, Z can be estimated numerically, and hence the inverse temperature β can be evaluated. Strictly speaking the average of the demon energy is taken both over all demons and over time. In practice, however, we have computed $\beta(t)$ at time t by averaging over all of the demons, and then computed the equilibrium temperature by averaging $\beta(t)$ over time. Since we work on a large lattice the discrepancy between the two procedures is sufficiently small, even near the critical point. In view of the importance of the temperature measurement we have developed a program that computes the inversion to obtain β. All that is needed is to input the number of the states of the demons. The program is constructed with the help of the computer algebra system Mathematica.

```
(*This is a temperature measurement for the Potts Model*)

PottsTemp[n_Integer?Positive]:=

    Module[{t1,t2},

           t1=(n-1)-n/(1-Z^n);
```

```
        t2=1/(1-Z);
        t1+t2]
```

We illustrate with two examples:

Example 1.

```
In[1]:= PottsTemp[4]
```

$$Out[1] = 3 + \frac{1}{(1-Z)} - \frac{4}{(1-Z)^4}$$

We can easily check our results by expanding Out[1] in the form of a rational polynomial:

```
In[2]:= Together[ExpandAll[Together[PottsTemp[4]]]]
```

$$Out[2] = \frac{Z + 2Z^2 + 3Z^3}{1 + Z + Z^2 + Z^3}$$

For our model, which has 8 possible states of the demons, we get

```
In[3]:= PottsTemp[8]
```

$$Out[3] = 7 + \frac{1}{1-Z} - \frac{8}{1-Z^8}$$

This polynomial can be expanded out to obtain the following rational number:

```
In[4]:= Together[ExpandAll[Together[PottsTemp[8]]]]
```

$$Out[4] = \frac{Z + 2\,Z^2 + 3\,Z^3 + 4\,Z^4 + 5\,Z^5 + 6\,Z^6 + 7\,Z^7}{1 + Z + Z^2 + Z^3 + Z^4 + Z^5 + Z^6 + Z^7}.$$

4.3 Fortran Code for the Microcanonical Simulation of the Three-State Potts Model

The microcanonical simulation of statistical systems uses a concept that is different from that of Monte Carlo algorithms. It uses a deterministic updating procedure. The main advantage is that the algorithm is very efficient, does not use random number generators and conditional branching, and is adaptable for parallelization and vectorization techniques. The problem with this method is that one must check whether or not the procedure is stochastic and correctly covers the phase space.

The implementation of the microcanonical simulation has beem outlined elsewhere [47, 48, 49, 50] and is particularly simple for the Ising model. One introduces a demon that visits in succession all spins with an energy bag of finite size. It flips the spin systematically if it can, that is, if the required amount of energy can be provided from its bag. This process can be described as a succession of logical operations on single bits. Parallelization is naturally introduced when one groups the bits into computer words. In this section we will discuss the implementation for the three-state Potts model. Let us recall that the action of this model is

$$S = \beta \sum_{<i,j>} \delta_{\sigma_i,\sigma_j}, \tag{4.14}$$

where β is the inverse temperature. The spins $\{\sigma_i\}$ take three possible states. The difficulty to be overcome is that the demon has to choose betwee › two possible states when it wants to flip a given spin. We may choose to solve this problem in a deterministic way (example $1 \rightarrow 2 \rightarrow$

$3 \to 1$); however, correlations will be induced, and the stochasticity of the process will be lost. The best way to solve this difficulty is by reintroducing a random choice. The demon will visit in succession all spins. For each one, one of the two possible flips is chosen at random (with equal probabilities $\frac{1}{2}$). Then the flip is done if the demon has the correct amount of energy to do it; if this is not the case, the spin remains unchanged. As will become obvious, the generation of an equal probability toss will not affect the efficiency of the algorithm.

In section 4.5, we show how the process can be expressed as a succession of logical operations performed on single bits. Section 4.6 discusses other problems such as the various possible implementations for a given lattice, the measurement of the temperature and the implementation of the random generator of bits. The code for the three-dimensional cubic lattice is given in the Appendix B.

4.4 Implementation at the Spin Level

The three states of a given spin can be represented as a set of two bits 00, 01 or 10 (11 is excluded). Let us denote these two bits as a and b and let a_j and b_j be the corresponding states of a neighboring spin. The energy of the link is related to

$$l_j = NOT(OR(XOR(a, a_j), XOR(b, b_j))). \tag{4.15}$$

Indeed, the contribution to the action is β if $l = 1$, and 0 otherwise. While visiting the spin, the demon has to choose at random its new trial state

$$a'b' : ab \longrightarrow a'b'.$$

This is performed by using a random bit τ according to the rule

$$\begin{cases} 00 \rightarrow 10 \rightarrow 01 \rightarrow 00, \text{ if } \tau = 0, \\ 00 \rightarrow 01 \rightarrow 10 \rightarrow 00, \text{ if } \tau = 1, \end{cases}$$

which is implemented by the operations

$$b' = AND(COMPL(b), XOR(a,\tau)),$$

$$a' = AND(COMPL(a), XOR(COMPL(b),\tau)). \tag{4.16}$$

The bag of the visiting demon contains the energy $E = \beta k$, with the constraint $k = 0, ..., k_{\max}$. The new state causes this amount to change to $E' = \beta k'$ with

$$k' = k + \sum_j l_j - \sum_j l'_j,$$

and the trial is accepted if

$$0 \leq k' \leq k_{\max} = 2^n - 1. \tag{4.17}$$

The bag energy of the demon is represented by a string of n bits, and thus $k_{\max} = 2^n + 1$. According to Eq. (4.17), the new state is acceptable if the binary representation of k' only involves n bits. If $|k - k'|$ is bounded by 2^n, the arithmetic can be performed with only $n+1$ binary digits, with the one of highest significance checking the condition (4.16). For instance, let us consider a cubic lattice with nearest neighbor interaction. The quantity $k - k'$ ranges from -6 to 6. Choosing $k_{\max}$, one sees that $k' + 8$ ranges from 2 to 21. The admissible range $(8, 15)$ is characterized by the fourth bit of the binary representation set to one. Thus, the updating algorithm is as follows:

Choose at random τ and compute the trial state $a'b'$ using (4.16). From the bag, energy k, represented by the three bits xyz, computes the binary representation $ABCD$ of

$$k + 8(A \leftarrow 1, B \leftarrow x, C \leftarrow y, D \leftarrow z).$$

For each neighbor j, compute l_j and l'_j using Eq. (4.15). Add and subtract respectively these quantities to $ABCD$. This is performed using

$$\begin{aligned}
A &\leftarrow XOR(AND(l_j, B, C, D), A),\\
B &\leftarrow XOR(AND(l_j, C, D), B),\\
C &\leftarrow XOR(AND(l_j, D), C),\\
D &\leftarrow XOR(l_j, D),\\
A &\leftarrow XOR(AND(l'_j, NOT(D), NOT(C), NOT(B)), A),\\
B &\leftarrow XOR(AND(l'_j, NOT(D), NOT(C)), B),\\
C &\leftarrow XOR(AND(l'_j, NOT(D)), C),\\
D &\leftarrow XOR(l'_j, D).
\end{aligned} \tag{4.18}$$

Now accept the changes $ab \rightarrow a'b'$, $xyz \rightarrow BCD$ if $A = 1$, and reject otherwise. This is done using

$$\begin{aligned}
a &\leftarrow XOR(AND(A, a'), AND(NOT(A), a)),\\
b &\leftarrow XOR(AND(A, b'), AND(NOT(A), b)),\\
x &\leftarrow XOR(AND(A, B), AND(NOT(A), x)),\\
y &\leftarrow XOR(AND(A, C), AND(NOT(A), y)),\\
z &\leftarrow XOR(AND(A, D), AND(NOT(A), z)).
\end{aligned} \tag{4.19}$$

For the general implementation, one should add to the previous description some details to allow full performance of the algorithm. The parallelization is ensured by grouping the bits into words. Independent demons work for each bit position. One should be careful to treat simultaneously non-interacting spins. As usual, a first possibility is to treat simultaneously independent replicas of the system, with one demon per replica. A second possibility is to treat a row of only non-interacting spins of the same system by a "battalion" of demons. We choose this option and give in Appendix B the code for a three-dimensional cubic lattice. To avoid interactions, the system is divided into odd-numbered and even-numbered subsystems which are treated in succession.

Simulation is done at a given energy rather than at a given temperature. The temperature is measured in the bags of the demons which are in thermal equilibrium with the system. Indeed the relation between the mean energy and the temperature is known for such a simple system. Two ways of measuring the temperature were shown in the last section of chapter three. The generation of the random bits τ in parallel is based on the theory of "primitive polynomials modulo 2". For instance, the polynomial

$$x^{18} + x^5 + x^2 + x + 1$$

was used to generate a random sequence of length $2^{18} - 1$ by the recurrence relation

$$\tau^{(n)} = XOR(\tau^{(n-18)} + \tau^{(n-5)} + \tau^{(n-2)} + \tau^{(n-1)}).$$

The following subroutine allows simulation of the Potts model on a three-dimensional cubic lattice. They are written in standard FORTRAN-77.

However, some compiler dependencies might occur. One uses extensively the logical (generally built-in) functions OR (logical or), XOR (logical exclusive or), AND (logical and), NOT (bitwise complement), ISHFTC (circular shift) and MASK (mask generation). Variables representing strings of bits should be conveniently declared. They are declared here as INTEGER, but some computers might require different declarations (e.g. INTEGER*8 if one wants to use words with 64 bits).

The function ISHFT (word, n) is used to construct a logical circular shift of "word" (assumed to be 64 bits long) of "n" bits toward the right (e.g ISHFTC(4,1) → 2). The implementation of this function is unfortunately not universal and should be checked on a given computer.

The lattice is supposed to have a size (LX*LY* 2w), where w is the number of bits per word (depending on the computer). The configuration is described by two arrays, SPIN and SPIN2, containing (2*LX*LY) words each. Skew periodic boundary conditions are set in directions x and y, such that the neighbors of a given spin are located in words at relative positions ± 2 and $\pm 2 * LX(modulo[2 * LX * LY])$. Let us first give the subroutine that performs one updating of the lattice. The arguments of the routine are the configuration (two arrays) and the demons (array of three words).

```
*This function ensures periodic boundary conditions in the lattice
*If LX and LY are powers of 2, it can be advantageously replaced by
        SUBROUTINE MONTE(SPIN1,SPIN2,DMN)
*       PARAMETER(LX=32,LY=32,LW=16)
        PARAMETER(LX=64,LY=64,LW=32)
* LX and LY are the dimensions of the lattice
```

```
      PARAMETER(NSIZE=2*LX*LY-1)
      PARAMETER(IHOP=13)
* This number IHOP should be prime with LX and LY
      IMPLICIT INTEGER (A-Z)
      DIMENSION SPIN1(0:NSIZE),SPIN2(0:NSIZE),DMN(3)
      DIMENSION NEIGH1(6),NEIGH2(6)
*     INLAT(I)=MOD(I,NSIZE+1)
* This function ensures periodic boundary conditions in the lattice
* If LX and LY are powers of 2, it can be advantageously replaced by
      INLAT(I)=AND(I,NSIZE)
      J=0
      K=1
      DO 1 I=0,NSIZE
* setting the neighbors
      NEIGH1(1)=SPIN1(J+K)
      NEIGH2(1)=SPIN2(J+K)
      NEIGH1(2)=ISHFTC(SPIN1(J+K),K,LW)
      NEIGH2(2)=ISHFTC(SPIN2(J+K),K,LW)
      NEIGH1(3)=SPIN1(INLAT(J+2))
      NEIGH2(3)=SPIN2(INLAT(J+2))
      NEIGH1(4)=SPIN1(INLAT(J-2))
      NEIGH2(4)=SPIN2(INLAT(J-2))
      NEIGH1(5)=SPIN1(INLAT(J+2*LX))
      NEIGH2(5)=SPIN2(INLAT(J+2*LX))
      NEIGH1(6)=SPIN1(INLAT(J-2*LX))
```

```
        NEIGH2(6)=SPIN2(INLAT(J-2*LX))
        CHOICE=IRDBIT(0)
        NEW1=AND(NOT(SPIN1(J)),XOR(NOT(SPIN2(J)),CHOICE))
        NEW2=AND(NOT(SPIN2(J)),XOR(SPIN1(J),CHOICE))
* now compute 8 + energy_of_demon + number_of_old_links_with_equal_spins
*   - number_of_new_links_with_equal_spins
        DP1=DMN(1)
        DP2=DMN(2)
        DP3=DMN(3)
        ACCEPT=-1
        DO 2 M=1,6
* adds old energy
        LINK=NOT(OR(XOR(SPIN1(J),NEIGH1(M)),XOR(SPIN2(J),NEIGH2(M))))
        CARRY1=AND(LINK,DP1)
        DP1=XOR(DP1,LINK)
        CARRY2=AND(CARRY1,DP2)
        DP2=XOR(DP2,CARRY1)
        CARRY1=AND(CARRY2,DP3)
        DP3=XOR(DP3,CARRY2)
        ACCEPT=XOR(ACCEPT,CARRY1)
* subtracts new energy
        LINK=NOT(OR(XOR(NEW1,NEIGH1(M)),XOR(NEW2,NEIGH2(M))))
        CARRY1=AND(LINK,NOT(DP1))
        DP1=XOR(DP1,LINK)
        CARRY2=AND(CARRY1,NOT(DP2))
```

```
         DP2=XOR(DP2,CARRY1)
         CARRY1=AND(CARRY2,NOT(DP3))
         DP3=XOR(DP3,CARRY2)
         ACCEPT=XOR(ACCEPT,CARRY1)
2        CONTINUE
* accepts or rejects the change
         SPIN1(J)=XOR(AND(ACCEPT,NEW1),AND(NOT(ACCEPT),SPIN1(J)))
         SPIN2(J)=XOR(AND(ACCEPT,NEW2),AND(NOT(ACCEPT),SPIN2(J)))
         DMN(1)=XOR(AND(ACCEPT,DP1),AND(NOT(ACCEPT),DMN(1)))
         DMN(2)=XOR(AND(ACCEPT,DP2),AND(NOT(ACCEPT),DMN(2)))
         DMN(3)=XOR(AND(ACCEPT,DP3),AND(NOT(ACCEPT),DMN(3)))
* End of the loop
         J=INLAT(J+IHOP)
         K=-K
1        CONTINUE
         RETURN
         END
```

The function also returning a string of random bits is given. If its argument is 0, the next string is returned. If the argument is not 0, it is used as a seed for the generation. Note that 18 seeds are necessary to initialize completely the generator. The data are random, but the following condition is required: the OR of all these data should not contain any 0 bits.

```
         FUNCTION IRDBIT(INIT)
         IMPLICIT INTEGER (A-Z)
```

```
      DIMENSION ITAB(0:31)
      SAVE I18,I5,I2,I1,I0,IFIRST
      DATA IFIRST/0/,I18/0/,I5/13/,I2/16/,I1/17/,I0/18/
```

* Standard initialization (random bits, here given in hexadecimal * for words with 64 bits; truncate if necessary)

```
*     DATA ITAB/
*    x '7EB722A0C9743C06'Z,'534AB95D97ECF94A'Z,
*    x 'FD3CD86EFCCC61DE'Z,'3B341E5A9A1160B4'Z,
*    x '5CDA1DE25BB8E8F5'Z,'76EDDA93192BC357'Z,
*    x '1CD3CF66101C4CBD'Z,'0C7216C2C95676A8'Z,
*    x 'ACF117D1EF24D606'Z,'AF452B2A2FB48E98'Z,
*    x '5E7C758368B24840'Z,'FF29D95A6F897866'Z,
*    x 'D1A46D4C9F62639A'Z,'CA05FFE020E049BD'Z,
*    x '7102A31B08C39D1E'Z,'E8DE18695A18CA02'Z,
*    x '98A33097B9C2250E'Z,'4556037DC5A2CC1A'Z,14*0/
      DATA ITAB/
     x Z'C9743C06',Z'97ECF94A',
     x Z'FCCC61DE',Z'9A1160B4',
     x Z'5BB8E8F5',Z'192BC357',
     x Z'101C4CBD',Z'C95676A8',
     x Z'EF24D606',Z'2FB48E98',
     x Z'68B24840',Z'6F897866',
     x Z'9F62639A',Z'20E049BD',
     x Z'08C39D1E',Z'5A18CA02',
     x Z'B9C2250E',Z'C5A2CC1A',14*0/
```

```
      IF(INIT.NE.0) THEN
```

* Initialization (with a check avoiding the generation of a sequence of zeros)

```
              ITAB(I1)=INIT
              J=INIT
              I=AND(31,I18+1)
              DO 1 K=1,16
                     J=OR(J,ITAB(I))
                     I=AND(31,I+1)
1             CONTINUE
              ITAB(I18)=OR(ITAB(I18),NOT(J))
      ENDIF
      ITAB(I0)=XOR(XOR(ITAB(I18),ITAB(I5)),XOR(ITAB(I2),ITAB(I1)))
      IRDBIT=ITAB(I0)
      I0=AND(31,I0+1)
      I1=AND(31,I1+1)
      I2=AND(31,I2+1)
      I5=AND(31,I5+1)
      I18=AND(31,I18+1)
      RETURN
      END
```

Now let us give some additional information for using the previous routines in a simulation. The main program has to build a starting configuration, for instance, ordered or disordered. In this construction, one should not forget that the state "11" is forbidden; that is, AND(SPIN(i),SPIN(i)) is always zero. In the simulation, it wise to scramble the energy bags of the demons at each updating step, for example, using

```
        DO 11 I = 1,3
11      DMN(I) = AMIX(DMN(I))
*with a scrambling function AMIX (written for words of 64 bits)
        INTEGER FUNCTION AMIX(WORD)
        PARAMETER(LW=32,LH=LW/2)
        IMPLICIT INTEGER (A-Z)
REAL RANF
        L=INT(LH*RANF())
        P1=AND(MASK(L),WORD)
        P2=AND(MASK(L),ISHFTC(WORD,-L,LW))
        P3=AND(NOT(MASK(2*L)),WORD)
        AMIX=ISHFTC(OR(OR(P2,ISHFTC(P1,L,LW)),P3),INT(LW*RANF()),LW)
        RETURN
        END
```

In the function AMIX, one exchanges the "L" lower order bits with the "L" next lower order bits. Thus, L should be less than 32 for 64-bit words. The energy of the demons is measured using demon energy = IBCOUNT(DMN(1) +2*IBCOUNT(DMN(2))+4* IBCOUNT(DMN(3)) where the function IBCOUNT counts the number of 1bits in the binary representation of its arguments. In the function IBCOUNT, we use

ISHIFT(word,n), which shifts "word" "n" times toward the right (e.g. ISHIFT(12,1) $\rightarrow$ 60). The mean energy per demon allows one to compute the temperature through the routine

```
       FUNCTION BETA(DENERG)
        XL=0
        XH=1
        DO 1 N=1,36
        XN=0.5*(XL+XH)
        A=1-XN**8
        FN=(8*(A+XN-1)-7*A*XN)/A/(1-XN)
        IF(FN.GT.DENERG) THEN
                XH=XN
        ELSE
                XL=XN
        ENDIF
1       CONTINUE
        BETA=-0.25*ALOG(XN)
        RETURN
        END
```

The full code is given in the Appendix B.

4.5 Numerical Simulation Results

This section shows the results of the simulation examined in section 4.4. The first result shows how soon the system establishes equilibrium. The faster the system equilibrates the better, since measurements are taken after equilibrium is established. The utility that determines how fast a system equilibrates is the dynamic critical exponent. Later on in this chapter, I will compute this value. A rough estimate of how fast the system equilibrates is obtained by keeping the energy constant and steadily increasing the iterations. The magnetization and the temperature were measured and the following two graphs were constructed.

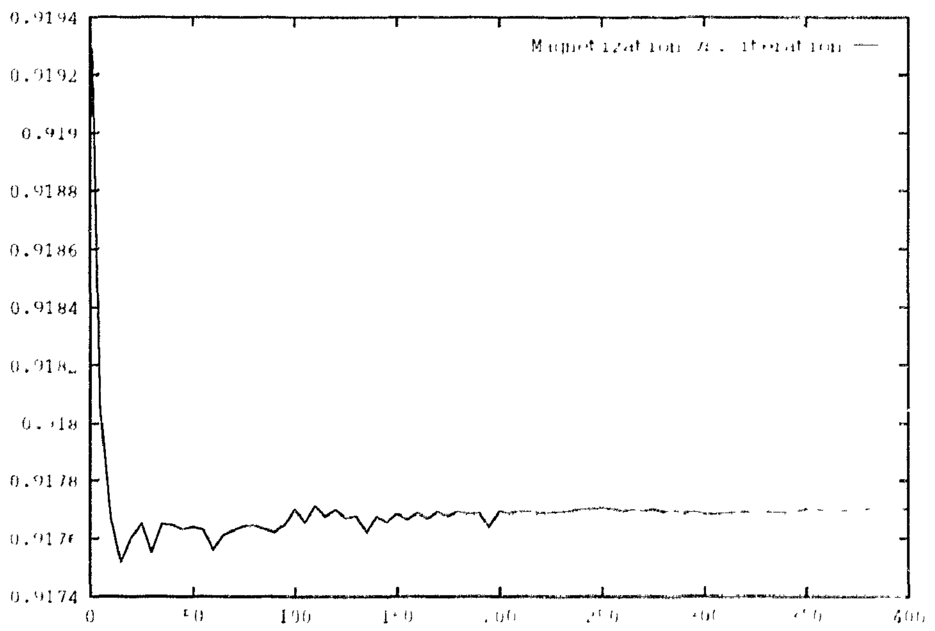


FIG. 4.1. *A graph of magnetization m vs. iteration i*

The graph of temperature against iteration follows below.

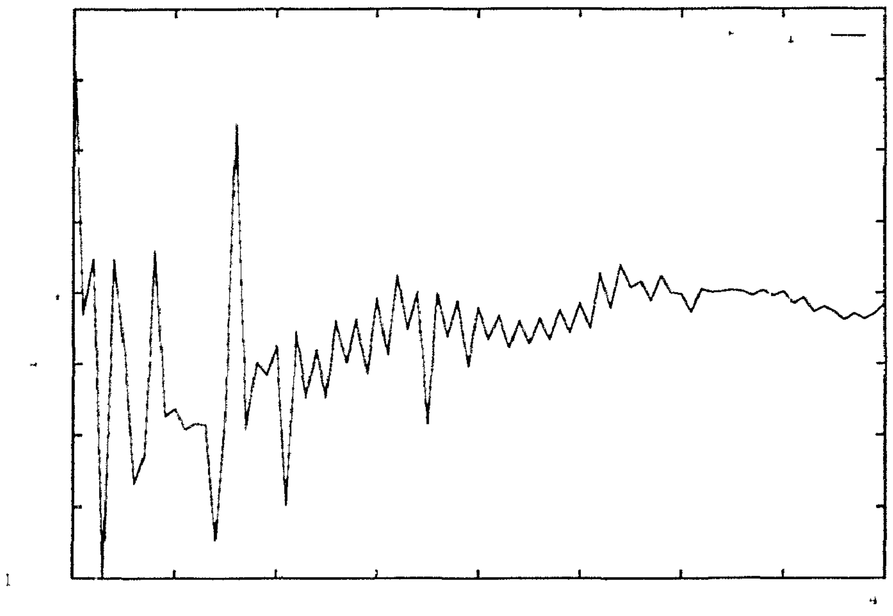


FIG. 4.2. *A graph of temperature T vs. iteration i*

The two plots show that magnetization and temperature stabilizes quickly. This shows that the system equilibrates quickly too, because the temperature is measured by demon energy which is in equilibrium with the system. Hence reasonably good measurements can be taken after 350 iterations.

Working on a lattice $64 \times 64 \times 32$, we steadily increase the energy E between $0 < E < 1$ by an increment of 0.1. The iterations were fixed at 20,000 and ten measurements were taken for magnetization and temperature. The first measurements were discarded to allow equilibrium to be established. For each energy, the average magnetization and the average temperature were computed. The results of the averages computed are shown in the following table. The way the simulation is terminated depends on the number of iterations that is set. By setting $NSWEEP = 100$, $NMEAS = 20$, and $NBATCH = 10$, the main loop is run 100 times, then measurements are taken. Then it is repeated 20 times and average measurements are taken. The whole process is repeated 10 times and the process is terminated.

E	0.1	0.2	0.3	0.4	0.5
$< M >$	0.917763	0.824314	0.716367	0.583866	0.394458
$< T >$	1.80598	1.809554	1.815777	1.815163	1.815996
E	0.6	0.615	0.7	0.8	0.9
$< M >$	0.000794	-0.000456	-0.000157	0.000174	0.000133
$< T >$	1.817117	1.820132	1.827005	1.852209	1.888042

TABLE 4.2. *Simulation results of total energy, average magnetization and temperature*

The basic goal here is to determine the phase transition. Thus more measurements were taken around the critical point, and they are given in the table below.

E	0.55	0.5771	0.578	0.58	0.59	0.625
$< M >$	0.231782	0.083975	0.008972	-0.001522	0.003952	0.000202
$< T >$	1.815179	1.815979	1.816213	1.816115	1.80379	1.817651

TABLE 4.3. *Simulation results of total energy, average magnetization and temperature near the phase transition*

Figure 4.3 is the average magnetization vs. temperature (T).

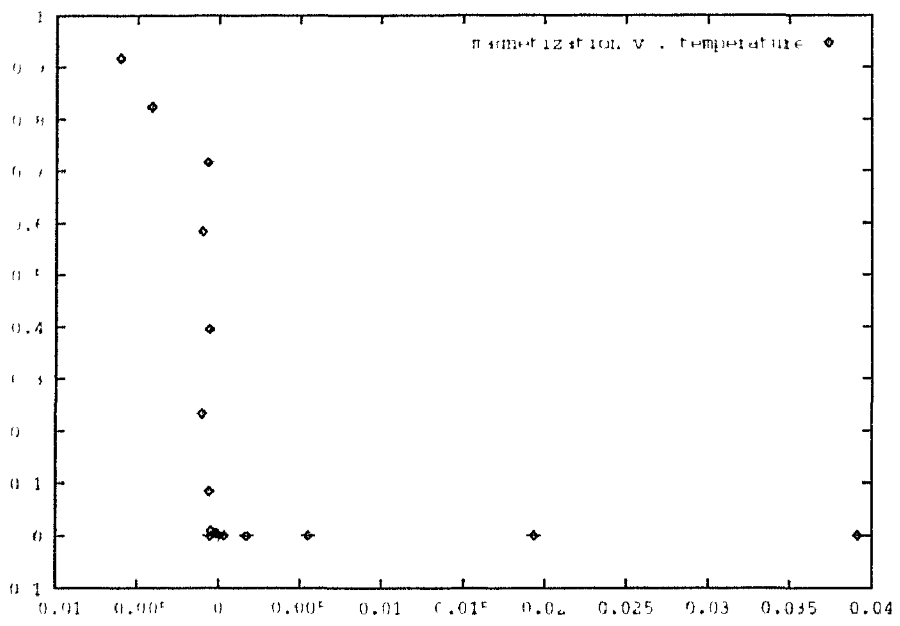


FIG. 4.3. *Average magnetization* $< m >$ *vs. the temperature* (T).

The graph in Fig. 4.4 is a plot of energy against average magnetization.

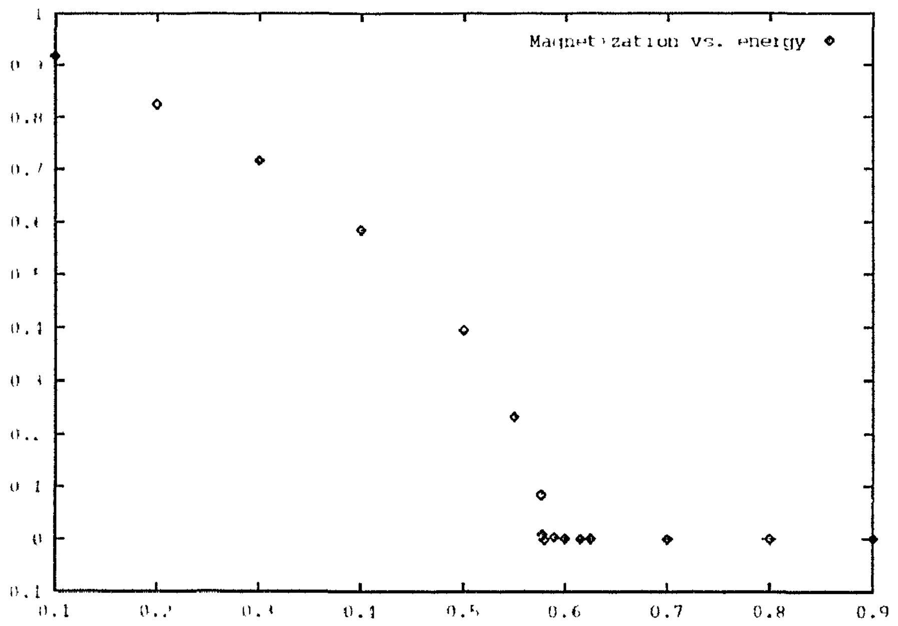


FIG. 4.4. *A graph of the energy* E *vs. average magnetization* $< m >$.

The graph of Fig. 4.5 is a plot of energy against average temperature. On this graph the energy at which phase transition occurs is very distinct.

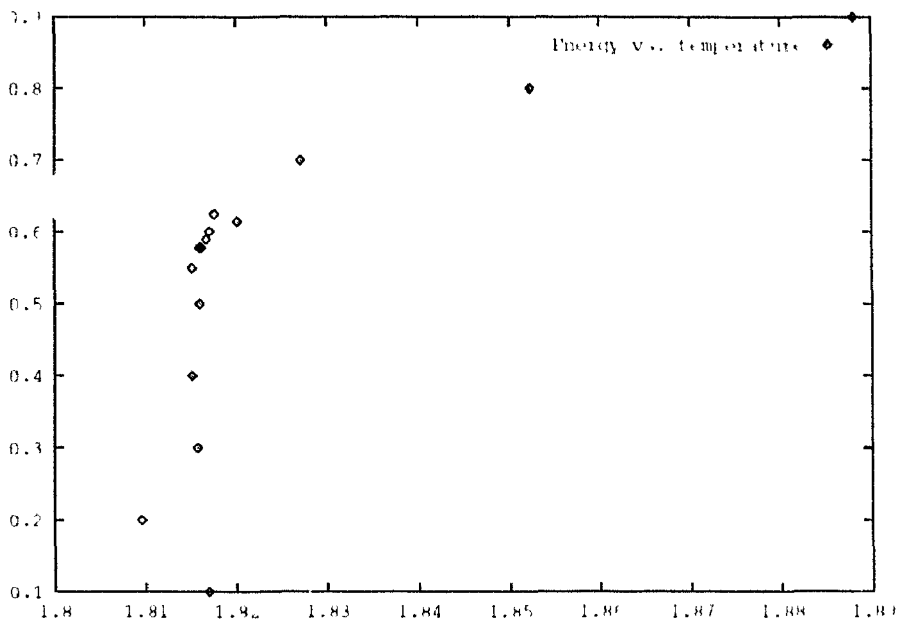


FIG. 4.5. *Variation of energy E vs. the average temperature $< T >$.*

From the Fig. 4.3 the critical temperature T_c was estimated.

4.6 Finite-Size Effects

The most serious drawback of a computer simulation approach to the study of phase transitions is that one must deal with finite systems. No finite system with a nonsingular Hamiltonian can exhibit a true phase transition. This becomes obvious when noting that the partition function in Eq. (2.3) cannot develop a singularity when the integral of the finite bounded Boltzmann function is performed over a finite space. Nevertheless, finite systems are reminiscent of phase transitions, and systematic studies of these pseudo-transitions as functions of system size may reveal information about the phase transitions in the infinite system [51]. A finite system gives an accurate description of the infinite system as long as the correlation length $\xi \leq L$, the linear extension of the system. When $\xi \geq L$ (e.g., near a critical point), the properties of the finite system will reflect the nature of the boundary conditions. For systems with periodic boundary conditions, the fluctuations will be "overcorrelated" and the various properties will be "rounded". As an example, the long-range order will persist above the phase transition, and singularities in the specific heat and the ordering susceptibility will be rounded and shifted in temperature. Ferdinard and Fisher [52] also Fisher [53] developed a finite-size scaling theory for critical phenomena which is extremely useful to guide the extrapolation of Monte Carlo finite-system properties to the thermodynamic limit. According to this theory, the free energy of a finite system is given by the homogeneous function

$$F(N,T) = L^{(2-\alpha)/\nu}\tilde{F}(tL^{1/\nu}), \tag{4.20}$$

where α and ν are the critical exponents pertaining to the specific heat and to the correlation length, respectively. $\tilde{F}$ is a scaling function involving the scaled variable $tL^{1/\nu}$ only, and $t = (T - T_c)/T_c$ with $T_c = T_c(L = \infty)$. Fisher [53] suggested the position of the maximum of the specific heat (or alternatively the ordering susceptibility) as an appropriate definition of the "transition temperature" of the finite system, $T_c(L)$. According to the finite-size scaling theory, the shift in critical temperature then scales as

$$\delta T_c = T_c(L = \infty) - T_c(L) \sim L^{-1/\nu}.$$

From the free energy in Eq. (4.20), the scaling properties of other thermodynamics functions may be derived, e.g.,

$$\Phi = L^{-\beta/\nu}\tilde{M}(tL^{1/\nu}). \tag{4.21}$$

In the limit $t \to 0$ and $L \to \infty$, Eq. (4.21) has to reduce to the infinite-system singular behavior, Eq. (2.9), and therefore in this limit, the order parameter scaling function is given by

$$\tilde{M}(tL^{1/\nu}) \sim (-tL^{1/\nu})^{\beta}.$$

The various scaling functions and amplitudes are nonuniversal properties which depend on the details of the system under consideration. In particular, these nonuniversal properties are functions of the boundary conditions chosen. The validity of the finite-size scaling theory has been demonstrated through extensive Monte Carlo simulations on two- and three-dimensional Ising models with periodic boundary conditions as well as free surfaces [54, 55]. It appears from these calculations that the systems with $L \gtrsim 10$ are well inside the asymptotic region described by Eq. (4.21). Therefore corrections to finite-size scaling, e.g.,

$$\delta T_c \sim L^{-1/\nu}(1 + aL^{-\Delta} + \ldots),$$

where Δ is the correction-to-scaling exponents ($\Delta \sim \frac{1}{2}$) (cf. Eq. (2.12)), need not be invoked. This is a very important result because it anticipates that Monte Carlo simulations of critical phenomena are feasible using system sizes and statistics compatible with current computer capacities.

4.7 Determining the Nature of a Phase Transition

Analyzing the nature of phase transitions by computer simulation techniques is hampered by finite-size effects. A phase transition is infinitely sharp only in an infinite system. The characteristic discontinuities and singularities accompanying phase transitions will appear rounded and smeared in the finite systems employed in a computer simulation. In real systems, phase transitions will also appear smeared to a degree that depends on the size of the system. Also it depends especially on the concentration of imperfections and impurities.

In principle, it is impossible, by any laboratory experiment or computer simulation, to prove that a phase transition is continuous. It can always be postulated that possible first-order discontinuities are below the resolution of the experiment. Similarly, it may be argued that experimentally observed metastabilities do not signal first-order phase transitions but are nonequilibrium effects associated with continuous transitions. The best one can do is to analyze as many properties as possible in the neighborhood of the phase transition and compare them with phase transitions of well-established nature. Thus, laboratory and computer experiments share a deficiency in their inability to determine unambiguously the nature of a phase transition. In a computer simulation study of the cooperative behavior of an interacting many-body system, the primary question to be answered is related to the existence of a stable ordered phase at finite temperatures. Theoretically, this is known to be a very difficult problem (see e.g. Griffiths 1972) [56]. In computer simulation studies, this problem is approached by a finite-size analysis of the possible long-range order parameters. The possible types of ordering in the finite system are suggested by the simulation itself. If the order parameter for finite temperatures approaches a nonzero value as $L \to \infty$, it is concluded that a phase with long-range order remains stable at finite temperatures. Therefore a finite temperature phase transition must exist. As the existence and structure of the ordered phase have been established, the question of the nature of the phase transition will now be examined. The transition is triggered when a thermodynamic parameter, such as the temperature, is varied. For the sake of simplicity, let us restrict ourselves to situations with a single phase transition. Only first-order and continuous phase transitions shall be distinguished.

A first-order phase transition is characterized by a discontinuity in the order parameter. The specific heat has a δ-function singularity superimposed on its discontinuity at T_c. The energy content of the δ-function represents the enthalpy of the transition. An important indication of a first-order phase transition is the presence of metastable states in the transition region. In this region, the free energy has two minima, and the system may become trapped in the upper metastable one for a finite time $t < \tau(T)$, where $\tau(T)$ is the relaxation time of the metastable state. In Figure 4.6 shows schematically the variation with temperature of the order parameter, $\Phi(T)$, and the inverse relaxation time in the region around first-order transition. The metastable branches of $\Phi(T)$ are indicated. The endpoints of these branches are pseudospinodal points at which the metastable system becomes thermodynamically unstable. The inverse relaxation time, accordingly, has two branches. The relaxation out

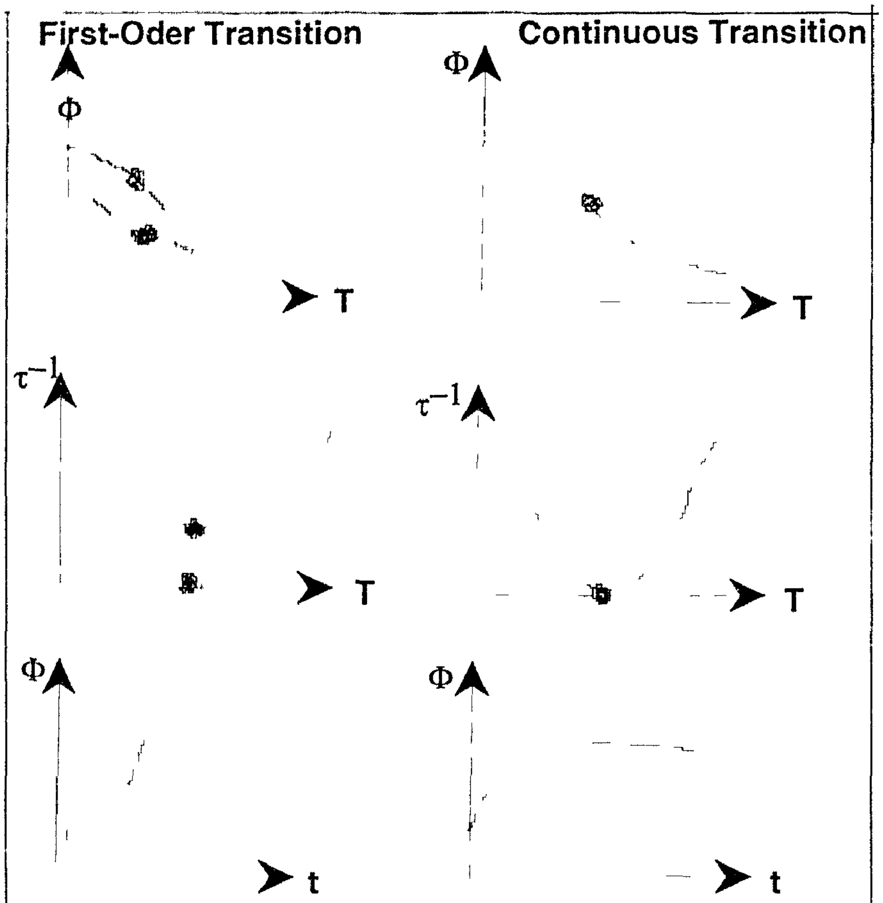


FIG. 4.6 Variation of order parameters and inverse relaxation time with temperature, and variation of parameters with Markov time.

of the metastable states becomes infinitely slow as the equilibrium transition point is approached from either side. When the system is perturbed in a metastable state, e.g. by an appropriate change in temperature, it may exhibit a two-step relaxation behavior which is an exceptional feature of systems undergoing first-order transitions: Firstly, the system relaxes into a new metastable state characterized by the new temperature and then eventually it relaxes into the true equilibrium state. The important point to stress is that the second step of the relaxation may not set in for a long time and that therefore in some cases the metastable state may mistakenly be confused with the equilibrium state. The microscopic consequence of trapping in the metastable state is the observability of hysteresis; i.e., the behavior of the system in the transition region depends on its thermal history. There are a few important exceptions to the above description. Firstly, some systems may have several local minima of the free energy in the transition region and may therefore give rise to a more complicated pattern of metastable states. This may in some cases lead to a cascading relaxation behavior associated with a whole staircase of steps. Secondly, the presence of hysteresis may not necessarily signal a first-order transition. If the system under consideration has more order parameter components than physical dimensions, an extremely slow domain-growth kinetics may result when the system is taken below the transition point. In that case, the annealing of domain characterized by different order parameters will be the rate-determining process, and in many cases it is impossible within a reasonable observation time to bring the system into a uniformly ordered phase. A global hysteresis will then result irrespective of the specific nature of the phase transition.

At a continuous transition, the order parameter vanishes in a continuous manner as in Eq. (2.9), and the fluctuation quantities may diverge, cf. Eqs. (2.10) and (2.11). The free energy only has a single minimum and no metastabilities are expected. However, the relaxation to equilibrium becomes slower as T_c is approached and the relaxation time diverges at the critical point ("critical slowing down"). Thus, very close to T_c and for sufficiently short observation times, a system with a continuous transition may behave as being effectively in a metastable state. The temperature variation of $\Phi(T)$ and $t^1(T)$ is drawn schematically in Fig. 4.6. Note this figure applies for a finite system which supports long-range ordering even of the temperatures above T_c.

In computer simulations on finite systems, first-order phase transitions will appear as partly smeared. The discontinuities are reduced and the δ-function of the specific heat is broadened. For first-order transitions associated with strong fluctuations, these observations are the usual consequences of finite-size rounding. However, there is always a contribution to the smearing from the effective averaging of information from metastable and stable states. Since a finite system can give rise to only a finite free energy barrier between the two minima of the free energy function, there is a finite probability of crossing the barrier within the observation time. This crossing is not only allowed from the local (metastable) minimum to the global (stable) minimum, but a finite system may perform a shifting between the two. If the barrier is very low (which it will be close to T_c), this will lead to a complete smearing, and it may be difficult to resolve the two states. In that case, the first-order transition effectively appears as a continuous transition strongly dominated by fluctuations. However, by increasing the lattice size, the two minima may be resolved and the complete free energy surface may be accurately probed by the simulation. This is a remarkable advantage of computer simulation over conventional theoretical calculations which are usually only able to probe the equilibrium properties. Detection of the metastable states is facilitated by studying the time-evolution of coarse-grained averages and the structure of distribution functions of internal energy and the order-parameter. The distribution functions are particularly useful if the system has several order-parameter components. In that case, the shifting between ordered and disordered states near T_c is coupled with a shifting between the various components of the order parameter. If the shifting between the stable and metastable states is sufficiently rapid to sample accurately the complete distribution function for, say, the order parameter $P(\Phi)$, a unique way is offered for determining the equilibrium first-order phase transition temperature. It shows $P(\Phi)$ in the transition region of a finite system with a single order parameter component. $P(\Phi)$ is a double-peaked function throughout the transition region where metastable states persist. At $T \simeq T_c$, the two peaks have the same intensity. This indicates that both phases are equally likely and that we are therefore at the only point where the two phases can co-exist, i.e., at the equilibrium transition point. To be consistent with the first-order transition, it is an important requirement that its two peaks move apart as the system size increases. Glosli and Plischke [49] have pointed out that $P(\Phi)$ is related to the free energy functional

$$F(\Phi) = -k_B T \ln P(\Phi), \tag{4.22}$$

which is part of the total free energy. The equivalent use of $F(\Phi)$ to analyze the nature of the phase transition is perhaps somewhat more heuristic. If the complete distribution function, $P(\Phi)$, cannot be obtained for systems with first-order transition, a few other methods may be called upon to determine T_c. The first one uses the classic Maxwell equal-area rule. This method is not very accurate because it presupposes, incorrectly, that the relaxation time is symmetric about T_c. A second method, which has found wide use among high-energy physicists for locating phase transition in lattice gauge theories (see e.g. Creutz et al.) [50], is a mixed-phase calculation. The idea is to initiate the simulation by a configuration which is a one-to-one mixture of the two phases. The equilibrium phase is then determined as the phase of the mixture which grows as the ensemble is built up. This method requires less computer time but is not as accurate as the method which uses the complete $P(\Phi)$ function. The third method requires an evaluation of the free energy function itself. Since $F(T)$ is not a thermal average, its evaluation presupposes knowledge of the partition function, Z. However, Z is not available from a Monte Carlo calculation which is built on Eq. (2.25). Therefore, $F(T)$ has to be determined indirectly, e.g. by a numerical integration of the internal energy

$$F(T) = \frac{T}{T_i} F(T_i) + T \int_{T_i^{-1}}^{T^{-1}} E(x) dx.$$

Such a procedure requires detailed information on the internal energy over a wide range of temperature around the transition as well as precise knowledge of the appropriate high and low temperature boundaries, $F(T_i)$, where the integration is started. If the internal energy is known along the metastable branches of the hysteresis loop, the point at which the two resulting free energy branches intersect may be determined fairly accurately.

This point is the equilibrium transition point. From the difference in the slope of the two $F(T)$ branches in this point, the enthalpy and entropy of transition may be derived:

$$\frac{\Delta S}{T_c} = \Delta E = \Delta(\partial F(T)/\partial T)_{T_c}.$$

The distribution function, $P(\Phi)$, for a finite system with one order-parameter component undergoing a continuous transition has a single peak for all temperatures, and the position of the peak moves continually as the temperature is varied through the phase transition. The corresponding free energy functional, $F(\Phi)$ Eq. (4.22), has a single minimum. These characteristics are distinctly different from those found for first-order transitions. The finite-size behavior of the various thermodynamics functions is conveniently analyzed in terms of the finite-size scaling theory to yield critical temperature and exponents.

4.8 General Correlation Function Series

In this section our aim is to measure the critical exponents ν and the critical temperature at $L = \infty$. We first note that the basic quantity of a spin system is the spin-spin correlation function $\Gamma_{\alpha\beta}(r_{ij})T$ which is a measure of the degree of correlation between spin fluctuations at lattice sites i and j. For $T > T_c$ and in zero external field, $\Gamma_{\alpha\beta}$ takes the following form of systems with no off-diagonal interactions

$$\begin{aligned}\Gamma_{\alpha\beta}(\bar{r_{ij}}, T) &\equiv < (\sigma_{i\alpha} - < \sigma_{i\alpha} >)(\sigma_{j\beta} - < \sigma_{j\beta} >) > /Tr(\sigma_{\alpha}{}^2) \\ &= \delta_{\alpha\beta} < \sigma_{i\alpha}\sigma_{j\beta} > /Tr(\sigma_{\alpha}{}^2).\end{aligned}$$

$\sigma_{i\alpha}$ is one of the components of the spin vector $\bar{\sigma}_i = (\sigma_{ix}, \sigma_{iy}, \sigma_{iz})$. A variety of physical quantities can be constructed from the correlation function. First, the internal energy of systems with pair interactions is simply a linear combination of pair-correlations within the interaction sphere. The specific heat can be derived from the internal energy. Second, by the fluctuation theorem the wave vector-dependent susceptibility tensor $\chi_\alpha(\bar{q}, T)$ is the Fourier transform of $\Gamma_{\alpha\beta}$. Third, the wave vector-dependent spherical moments

$$\sigma_{\alpha,n}(q,T) \equiv \sum_{j(\neq i)} (|\bar{r_{ij}}|/r_0)^n e^{i\bar{q}.\bar{r_{ij}}} \Gamma_{\alpha\alpha}(\bar{r_{ij}}, T)$$

enable us to determine the correlation length

$$\xi_\alpha(q,T) = r_0[\sigma_{\alpha,2}(q,T)/2d\sigma_{\alpha,0}(\bar{q},T)]^{\frac{1}{2}},$$

where r_0 is the lattice constant and d is the spatial dimension. If $T \to T_c$, the correlation length defined from the second spherical moment is expected to be proportional to the true correlation length (Fisher and Burford 1967) [42]:

$$\xi_\alpha(q,T) \equiv \lim_{|r_{ij}| \to \infty} r_0 |r_{ij}|^{-1} \ln |e^{i\bar{q}.\bar{r_{ij}}} \Gamma_{\alpha\alpha}(\bar{r_{ij}}, T)|.$$

This general approach constitutes a convenient way to calculate correlation function series. The availability of autocorrelation and paircorrelation functions seen as a function r_{ij} allows studies of several thermodynamic quantities which are expected to display critical fluctuations. These important relations of critical exponents and their dependency on correlation lengths are summarized below:

$$\begin{aligned}
\sigma_{\alpha,n}(\bar{q} = \bar{q}_0, T) &\sim (T - T_c)^{\gamma - n\nu},\ T \to T_c \\
\sigma_{\alpha,0}(q = \bar{q}_0, T) &= k_B T N^{-1} \chi_\alpha(q_0, T) - \Gamma_{\alpha\alpha}(0, T) \\
&\simeq k_B T N^{-1} \chi_\alpha(q_0, T) - \Gamma_{\alpha\alpha}(0, T),\ T \to T_c \\
&\sim (T - T_c)^{-\gamma},\ T \to T_c
\end{aligned}$$

$$\xi_\alpha(\bar{q} = \bar{q}_0, T) \sim (T - T_c)^{\nu},\ T \to T_c,$$

where q_0 is the wave vector characterizing the ordered phase. As has been pointed out, the relatively small size of our system is the most serious limitation in computer simulation studies. Since one can simulate only finite lattices, it is difficult to apply the above definitions to compute the critical exponents directly. However, with the help of finite-size scaling analysis, the finite length can be extrapolated to infinity. According to finite-size scaling one gets the following relation:

$$\xi(T) \sim L \sim |T - T_c|^{\nu}, \tag{4.23}$$

$$T_c(L) - T_c(L = \infty) \sim aL^{1/\nu}. \tag{4.24}$$

In our work, the lengths $L = 4$, 8, 16, 32 and $L = 64$ were used for the following different energies: $E = 0.1$, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7 and $E = 0.8$ at $22K$. Eleven measurements were taken, but the first measurements were neglected to allow equilibrium to be established. A graph of temperature against magnetization was constructed, and the critical temperature was estimated from the graph. The results of our measurements are summarized in the Table 4.4 below including our estimated critical temperature for the various lengths.

L	T	T_c
4	0.481296	0.481022
8	0.788997	0.788469
16	1.130654	1.128900
32	1.480854	1.474803
64	1.827005	1.817117

Table 4.4. *Length of the lattices, near and at the critical temperat* .

By plotting $\ln(L)$ against $\ln(T - T_c)$ the critical exponent ν was obtained by taking the gradient. The graph of our measurement is shown in Fig. 4.7.

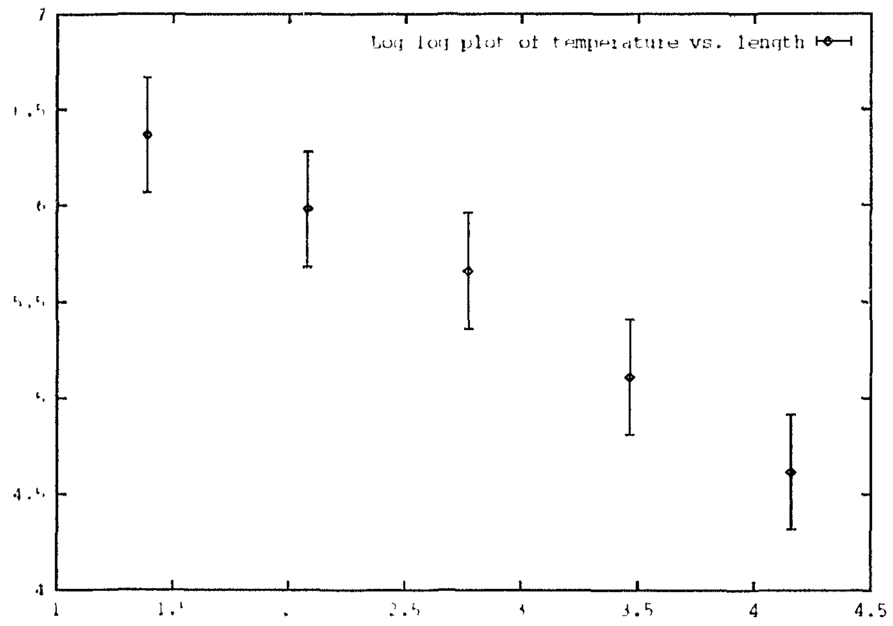


FIG. 4.7. *Determining the critical exponent ν by plotting* $\ln(L)$ *vs.* $\ln(T - T_c)$.

The critical exponent ν estimation from our measurements was found to be equal to 0.631 ± 0.005. The error measurement used is the standard deviation which is obtained directly from Maple.

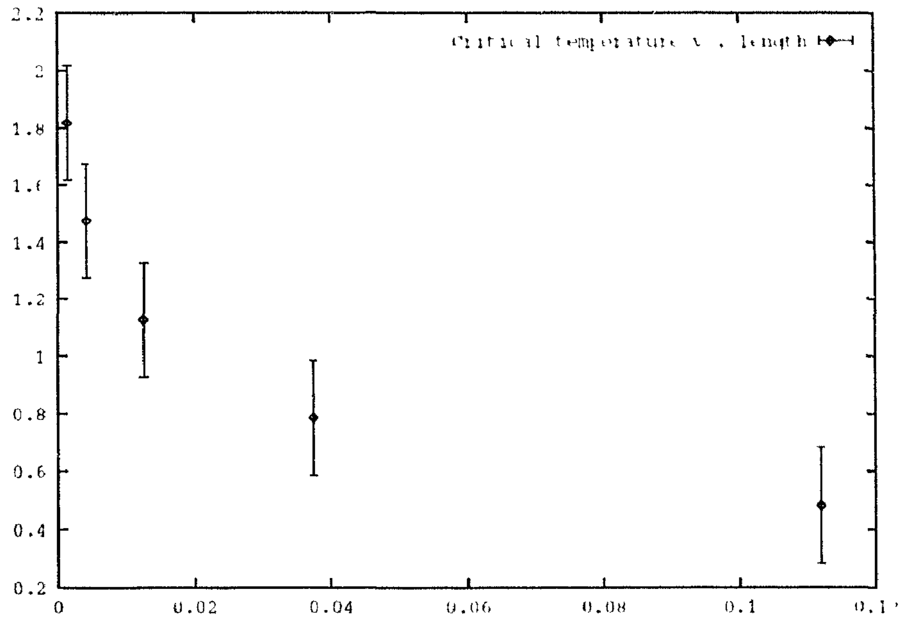


FIG. 4.8. *Critical temperature $T_c(L = \infty)$ for an infinity system, obtained by plotting $T_c(L)$ for finite length L vs. $L^{-1/\nu}$.*

Once the critical exponent ν is obtained, a graph of $L^{-1/\nu}$ against $T_c(L)$ can be plotted and the value of $T_c(L = \infty)$ is obtained by taking the intersection on the vertical axis. The smallest length $L = 4$ is know to deviate from the linearity of the curve so it was neglected. The graph of our measurements is shown in Fig. 4.8. For smaller lattice size, there is a deviation from the linearity of the curve. However, one cannot use very large lattice sizes because it will take a long time before any reasonable measurements can be taken. Hence for Fig. 4.8, the last two measurements were neglected. The best straight line was fitted using the rest of the points. The errors used are the mean deviation.

The critical temperature for an infinite-lattice was estimated to be $T_c(L = \infty) = 2.15 \pm 0.05$

4.9 The Dynamical Critical Exponent Measurement

The principle of dynamic scaling, as confirmed by renormalization-group theory of the critical dynamics, asserts that the decay rate Γ of the order-parameter fluctuations sufficiently close to the critical point assumes the form

$$\Gamma = k^z \Omega(k\xi).$$

It also predicts that the shear viscosity η will diverge as

$$\eta_s = \xi^{\phi}.$$

The dynamic scaling function $\Omega(y)$ satisfies the boundary conditions

$$\lim_{y \to 0} \Omega(y) = C_0 y^{-(1+\phi)}$$

$$\lim_{y \to \infty} \Omega(y) = C_\infty,$$

where C_0 and C_∞ are constants. The dynamic critical exponents z and ϕ satisfy the relation

$$z = 3 + \phi.$$

At a given temperature, i.e. at a given value of ξ, Γ will vary as k^2 for $k\xi \ll 1$ in agreement with the laws of hydrodynamics. On the other hand, Γ will vary as k^z in the critical region $k\xi \gg 1$. The dynamic critical phenomenon can be characterized if systems can be grouped in dynamic universality classes. Systems within the same dynamic universality class have identical dynamic critical exponents as well as the dynamical scaling function $\Gamma(y)$ when properly normalized.

Creutz's deterministic microcanonical dynamics is an interesting alternative to standard Monte Carlo simulations of classic statistical systems. The main advantage of microcanonical dynamics is increased speed in numerical simulations. Microcanonical algorithms are easily vectorized, can often be implemented without the use of floating point arithmetic, and may not require the generation of random numbers. What will eventually decide the utility of the microcanonical approach, however, is how quickly the system equilibrates in the critical region. A fundamental measure of how fast a system reaches equilibrium is the dynamical critical exponent z. We show how to measure this exponent for the microcanonical three-dimensional Potts model. We then compare our results with what others have obtained. For example, Brower et al. [47] obtained $z = 2.26 \pm 0.05$, which is comparable to the dynamical critical exponent found for Monte Carlo algorithms. Their measurements demonstrate that microcanonical simulation equilibrates at roughly the same rate as the nondeterministic algorithms, while using much less computer time.

In general, any statistical dynamics will exhibit an autocorrelation singularity at a critical point. If $A(t)$ is an observable at time t, the autocorrelation time τ is defined by

$$< A(T)A(t+T) > \sim e^{-T/\tau} + const, \tag{4.25}$$

where the average is taken over the sampling time t. In the critical region the autocorrelation time increases like

$$\tau \sim C_A \xi^z$$

as the correlation length ξ diverges. This is known as "critical slowing down," because in numerical simulations one must run much longer times to make uncorrelated measurements. C_A is a constant that can depend both on the particular observable A and on the particular dynamics used. In contrast, the dynamical critical exponent z is believed to be universal. The basic universality classes that have been proposed are as follows:

(a) dynamics with no local conservation laws (model A),

(b) dynamics with local order-parameter conservation (model B),

(c) dynamics with a conserved density other than the order parameter, such as an energy density (model C).

Standard Metropolis or heat bath algorithms are in the "model A" universality class. For the three-dimensional Ising model the dynamical critical exponent has been measured for these Monte Carlo algorithms to be $z \simeq 2.08$ by Yalabik and Gunton [58], $z = 2.17 \pm 0.06$ by Chakrabarti et al. [59], $z = 2.11 \pm 0.03$ by Jan et al. [60], and $z = 1.965 \pm 0.010$ by Kalle [61]. The semilocal implementation of the microcanonical dynamics is expected to be "model A," and indeed the system measurement of $z = 2.26 \pm 0.05$ agrees with the Monte Carlo measurements as much as they agree among themselves [47]. The important difference between microcanonical simulation and these previous results is that the algorithm uses the faster deterministic dynamics.

To determine z the magnetization autocorrelation function $Q(T)$ was measured by

$$Q(T) = \frac{1}{c} < m(t)m(t+T) >,$$

where the average is over the sampling time t and $m(t)$ is the instantaneous magnetization given as

$$m(t) = \frac{1}{N}\sum_{i=1}^{n} s_i(t).$$

By working at temperatures below the critical value we found $< m(t) > 0$, so that the constant in Eq. (4.25) could be neglected. Guided by the results obtained in section 4.6 we took average measurements of the magnetization and temperature for different values of Monte Carlo steps and at different energy values below the critical point. The values of $Q(T)$ were computed. The autocorrelation time τ was determined as the gradient from a linear fit of $InQ(T)$ to T. We could always obtain good linear fits, which demonstrates the correctness of using Eq. (4.25) to determine τ. The equation

$$\xi(T) \sim (T - T_c)^{-\nu}$$

is used to compute ν, and once the correlation time τ for various values of the temperature was determined, we were in a position to obtain the dynamical critical exponent z by noting that near the critical point

$$\tau \sim (T - T_c)^{-\nu z}.$$

Our results for $\ln(\tau)$ and $\ln(T - T_c)$ are shown in Table 4.5 below.

$\ln(T - T_c)$	7.3540	5.8158	6.2607	6.8880
$\ln(\tau)$	1.2809	4.1026	4.4257	3.6434
$\ln(T - T_c)$	6.4890	7.3385	5.8328	6.6377
$\ln(\tau)$	3.9343	1.2929	4.8709	3.7645

Table 4.5. *A table of* $\ln T - T_c$ *and* $\ln \tau$

We fit $In(\tau)$ versus $In(T - T_c)$ shown in the Fig. 4.9.

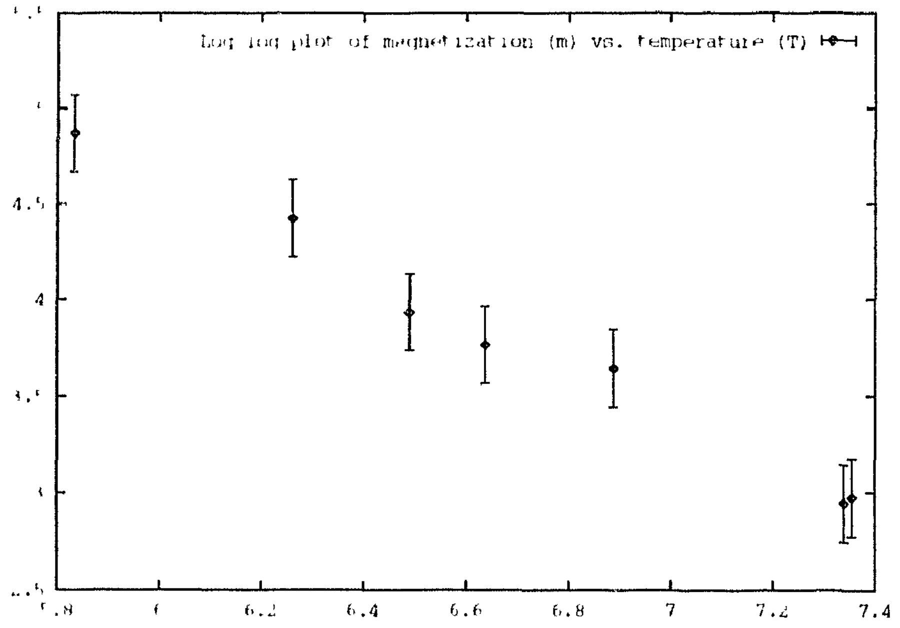


FIG. 4.9. *A graph of logarithm of the autocorrelation coefficient* τ *against temperature* $\ln(T - T_c)$ *to measure the dynamic critical exponent* z.

We measured the gradient, and since ν has been determined, the exponent z is easily computed to be the gradient divided by ν. Our result for the dynamic critical exponent is $z = 2.11 \pm 0.05$. The error measurement used is the standard deviation which is obtained directly from Maple.

Chapter 5

CONCLUSION

Much has been written about the Potts-Ising model, but very little on the symbolic computation of the Potts-Ising model. This thesis therefore serves to begin bridging this long-standing gap. In this thesis, I have considered the symbolic solution of the two-dimensional Potts-Ising model by the perturbation method. The one-dimensional case is not interesting because the partition function is analytic and does not reveal any phase transition. However, from the one-dimensional computation, a method for attacking the two-dimensional case is made clear. Using the screw method of Kramers and Wannier, I developed an algorithm using Maple for computing the low-temperature partition function and the eigenvector series without an external magnetic field, which has not be done before. The algorithm was carefully designed so that one can use it to teach the construction of the partition function by Kramers and Wannier. Moreover the algorithms are easily adaptable to be used with any work that requires the construction of V-matrices for both numerical and symbolic computation. The transformation matrix which is used to simplify the complete V-matrix has also been built into the program.

I have also incorporated the two-dimensional exact series solution of Onsager's computation on a square lattice. The series solution of the two-dimensional Ising model with many components has also been built. The parameter s is the number of components one needs to input. For $s = 2$, the many components lattice reduces to the Kramers and Wannier model. I have also built an algorithm that computes the three-dimensional Ising model. It is an extension of the work of Kramers and Wannier which was started by Oguchi. My symbolic computation reveals the additional information that if n is odd, the number of the submatrices of the V-matrix must be 1 each. Otherwise the computation of the partition function computation will be false. If n is even, then twice the number of the submatrices P, Q, R, and S must be taken. In fact the computation of the eigenvectors of the partition function would be almost impossible without the symbolic computation. Until now only 3-screw and 5-screw computation has been attempted. My work, however, has generalized the screw approach to the n-screw. Another achievement is the algorithm that computes the ground state entropy, $W_G(q)$, for a lattice, G, where $q \geq 3$ is an integer. The computation is carried out on square, triangular, and Kagome lattices, for the number of colors $q = 3$, $q = 4$, and $q = 3$, respectively.

The only concern for the algorithm based on Kramers and Wannier V-matrices and its extension is that the size of the matrices used in the computation grows rapidly. Therefore for large matrices the program is slow (for specific examples see chapter 3).

Nevertheless this work demonstrates the first use of Maple symbolic computation to attack the partition function of the Ising model based on the V-matrices of Kramers and Wannier and other statistical mechanics problems.

For the microcanonical simulation, an important achievement is the general formula which has been derived to compute the temperature or the inverse temperature of a system that uses the demon approach. The deduction from the general formula is also significant. With only one condition to check, one can use a simpler formula with a relatively very small error. Moreover, it shows that in general the demon's energy is independent of the dimension of the lattices when considering nearest neighbor interaction. The simulation results are also comparable to the theoretical results. The graphs obtained from my simulation results truly confirm theoretical prediction.

As a possible extension to the work so far achieved, several alternative approaches to the solution of the Potts-Ising model exist. A natural extension to this work is the high-temperature computation of partition function of the model. Another immediate need to be addressed is the inclusion of the an external magnetic field into the Hamiltonian equation. There are also other methods based on group theory that should be implemented. For example, the set $\mathcal{U}$ can be thought of as points of a countable set,

$$\mathcal{U} := \{A : A \in \mathcal{Z}^{\nu}\},$$

which is the set of ν-tuples of integers of lattice sites per each cell or a sum of several lattice sites per unit cell.

If one considers a set $A \subset V(G)$, then the group of graphs is given as

$$\Lambda'' : \mathcal{G}_{\mathcal{B}_\Lambda} = \prod_{b \in \mathcal{B}_\Lambda} \mathcal{G}_b = \{1 = (l_b)_{b \in \mathcal{B}_\Lambda}\},$$

where $\mathcal{G}_b$ denotes the dual group of $\mathcal{G}'_b$ $x \notin M$. The Hamiltonian definition for this Ising model $(G, \mathcal{G}, \mathcal{K})$ could be given as

$$\mathcal{H}(\sigma) = - \sum_{e \in E(G)} K_e \sigma_e, \ \sigma \in \mathcal{G}. \tag{5.1}$$

For graph theory, one can link the coefficients of the partition function of the series to the number of ways of coloring a graph G with a given number of colors, which also can be linked to the computation of the chromatic polynomial.

From combinatorial theory, at zero magnetic field, one can obtain the power series expansion by counting the number of ways of forming closed paths of a given length along the bonds of the lattice. In this case, the partition function can be in the form

$$Z = \sum_{\sigma_1 = \pm 1} \cdots \sum_{\sigma_N = \pm 1} \prod_{n.n} \exp(K \sigma_i \sigma_j).$$

The product over $n.n$ denotes the product over values of i and j corresponding to nearest neighbor points of the lattice [44].

The C^*-algebra consideration of the Ising model can consider the Hamiltonian

$$\mathcal{H}(n) - \frac{1}{2} \beta m H (\sigma_i + \sigma_{i+1}) - \beta J \sigma_i \sigma_{i+1},$$

where $\sigma : \mathcal{Z} \to \{1, -1\}$, and $\mathcal{Z}$ is the set of integers. One can denote $\mathcal{GK}_{[a,b]}$, the algebra of all observables relative to the interval $[a, b]$ which can be equipped with the sup-nom:

$$||A|| = \sup_{\sigma \in \mathcal{K}_{[a,b]}} |A(\sigma_a, \ldots \sigma_b)|,$$

where $\mathcal{K}_{[a,b]}$ is the configuration space for the interval $[a, b]$ [44].

Also the complicated derivation of Onsager's complete solution of the Ising model on a square net can be tackled in a less complicated form by considering dimer statistics. A dimer is simply a figure that may be drawn on a lattice that covers two nearest-neighbor sites and the bond that joins them. In this case the generating function for closest-packed dimer configurations is related to the Pfaffians, which is an antisymmetric matrix whose associated determinants can be evaluated. This Pfaffians plays a crucial role in the analysis of the two-dimensional Ising model [44]. All these are issues which need to be addressed with symbolic computation in the future.

The strategy which I would hope to follow in the future with the work of the symbolic computation and the simulation method, which is beyond this scope of thesis is to send information from the simulation to the symbolic and vice versa. In this way the results of the symbolic computation could be piped to the various components of the simulation which need them. Ideally, the work so far achieved is a stepping stone which, if the various approaches are followed satisfactorily, should be integrated into a comprehensive package of tools for dealing with the Potts-Ising model.

Finally it is anticipated that it will be fully developed into a marketable software. Because the model has a wide application, I hope that it will be helpful as a tool for research as well as in the classroom.

APPENDIX A

```
# Lars Onsager generalization of the Kramers and Wannier series solution of the
# Ising square net. x=tanh(2H)/cosh(2H)  H=J/kT, J is the interaction strength.
# x is normally given to be kappa (k) k=tanh(H)/(2*cosh(H)) or
# 2k=sinh(H)/cosh^2(H)
# This progrm computes the exact 2-d partition function of Ising square net
# in series.
ex2dseries:=proc(t) local t1,t2;
t1:=-sum(binomial(2*n,n)^2*(4*n)^(-1)*x^(2*n),n=1..t+2);
K_:=x=tanh(H)/(2*cosh(H));
pf=2*cosh(H)*(value(subs({ln(e)=1,0=0},series(e^t1,x=0,t+2))));
end;


#This computes the associated two-dimensional partition function on a square
#lattice for the lower right negative Kramers and Wannier matrix defined
#on the LEFT by perturbation method given in series.


d2lnpps:= proc(n,p) local c1,c2,Id,i,ii,j,k,ms,n1,q,u0,u1,u2,ls;
#'n' is the size of the matrices and 'p' is the length of the series
   #options trace;
    n1:=2^n/2;


#compute the needed matrices.
```

```
   u0:=matrix(n1,n1,[]): #the constant matrix U0
     for i to n1 do
        for j to n1 do
         if j=2*i-1 then u0[i,j]:=1
         else           u0[i,j]:=0
        fi;
       od;
    od;
  u1:=matrix(n1,n1,[]): #the constant matrix U1
   for i to n1 do
     for j to n1 do
        if j=2*i then   u1[i,j]:=1
         elif  j=2*(n1-i)+2 then u1[i,j]:=-1
        else           u1[i,j]:=0
       fi;
     od;
   od;
u2:=matrix(n1,n1,[]): #the constant matrix U2
 for i to n1 do
   for j to n1 do
    if j=2*(n1-i)+1 then u2[i,j]:=-1
       else           u2[i,j]:=0
    fi;
   od;
   od;


#perturbation method begins here.

#set up array to be used in the perturbation computation


    for i from 0 to p do

     f.i:=matrix(1,n1,[seq(ff.j,j=0..n1-1)]);

  od;


#begin perturbation computation.

#compute initial perturbation results.


  Id:= array(identity, 1..n1,1..n1);

  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);

  f0:=matrix(1,n1,[1,0$n1-1]);

  L1:=matrix(1,1,[0]);

  ls:=map(evalm,f1&*(Id-u0)=f0&*u1);

    for i from 1 to n1 do

      m.i:=lhs(ls)[1,i]=rhs(ls)[1,i];

    od;

  assign(solve({seq(m.i,i=1..n1)} { seq(ff.k,k=1..n1-1)}));

  f1:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);

  readlib(unassign):

  unassign(evaln(ff.(1..n1-1)));

  L2:=evalm(-matrix(1,1,[f0[1,n1]]));
```

```
#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

  for i from 2 to p do
     U.i:= f.i&*(Id-u0)=f.(i-1)&*u1+f.(i-2)&*u2-
     convert([seq(L.(q+2)&*f.(i-q-2),  q=0..i-2)],'+');q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[1,ii]= rhs(U.i)[1,ii];
       od;ii:='ii':
    assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
    f.i:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
    readlib(unassign):
    unassign(evaln(ff.(1..n1-1)));
    L.(i+1):=evalm(-matrix(1,1,[f.(i-1)[1,n1]]));
     od;

#matrices used on the perturbation computation

U_0:=u0:
U_1:=u1:
U_2:=u2:

#eigenvalues and eigenvectors series
```

```
cv:=sort(convert([seq(b^i*f.i, i=0..p)],'+')):

#cv:=map(evalm,cv);

convert([seq(b^i*L.i[1,1], i=0..p)],'+');

end;



#This computes the associated two-dimensional partition function on a square

#lattice for the upper left positive Kramers and Wannier matrix defined on the #


d2lppps:= proc(n,p) local c1,c2,Id,i,ii,j,k,ms,n1,q,u0,u1,u2;

#'n' is the size of the matrices and 'p' is the length of the series

   #options trace;

    n1:=2^n/2;


#compute the needed matrices.


   u0:=matrix(n1,n1,[]):

       for i to n1 do

          for j to n1 do

           if j=2*i-1 then u0[i,j]:=1

           else           u0[i,j]:=0

          fi;

         od;

      od;
```

```
  u1:=matrix(n1,n1,[]):
     for i to n1 do
       for j to n1 do
          if j=2*i or j=2*(n1-i)+2 then u1[i,j]:=1
          else          u1[i,j]:=0
         fi;
       od;
     od;
u2:=matrix(n1,n1,[]):
   for i to n1 do
     for j to n1 do
      if j=2*(n1-i)+1 then u2[i,j]:=1
         else          u2[i,j]:=0
      fi;
     od;
   od;


#perturbation method begins here.
#set up array to be used in the perturbation computation


  for i from 0 to p do #frist define all the fi's
     f.i:=matrix(1,n1,[seq(ff.j,j=0..n1-1)]);
  od; i:='i': j:='j':


#begin perturbation computation.
```

```
#compute initial perturbation results.

  Id:= array(identity, 1..n1,1..n1);
  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);
  f0:=matrix(1,n1,[1,0$n1-1]);
  L1:=matrix(1,1,[0]);
  f1:=matrix(1,n1, [0$(n1-2),1,0]);
  L2:=matrix(1,1,[0]);
  f2:=matrix(1,n1,[0$(n1-1),1]);
  L3:=matrix(1,1,[0]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

for i from 3 to p do
     U.i:= f.i&*(Id-u0)=f.(i-1)&*u1+f.(i-2)&*u2-
     convert([seq(L.(q+3)&*f.(i-q-3),  q=0..i-3)],`+`);q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[1,ii]= rhs(U.i)[1,ii];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
          readlib(unassign):
           unassign(evaln(ff.(1..n1-1)));
          L.(i+1):=matrix(1,1,[f.(i-1)[1,n1]]);
     od;

#matrices used on the perturbation computation

U_0:=u0:
U_1:=u1:
U_2:=u2:

#eigenvalues and eigenvectors series

c_v:=sort(convert([seq(b^i*f.i, i=0..p)],'+')):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],'+');
end;

#This computes the associated two-dimensional partition function on a square
#lattice for the upper left positive Kramers and Wannier matrix defined on the
#RIGHT by perturbation method given in series.

d2rppps:= proc(n,p) local c1,c2,Id,i,ii,j,k,ms,n1,q,u0,u1,u2;
#'n' is the size of the matrices and 'p' is the length of the series
   #options trace;
    n1:=2^n/2;
```

```
#compute the needed matrices.

  u0:=matrix(n1,n1,[]):
       for i to n1 do
          for j to n1 do
           if j=2*i-1 then u0[i,j]:=1
           else           u0[i,j]:=0
          fi;
         od;
      od;
  u1:=matrix(n1,n1,[]):
     for i to n1 do
       for j to n1 do
          if j=2*i or j=2*(n1-i)+2 then u1[i,j]:=1
          else          u1[i,j]:=0
         fi;
       od;
     od;
u2:=matrix(n1,n1,[]):
   for i to n1 do
     for j to n1 do
      if j=2*(n1-i)+1 then u2[i,j]:=1
         else          u2[i,j]:=0
      fi;
     od;
   od;


#perturbation method begins here.

#set up array to be used in the perturbation computation


  for i from 0 to p do

     f.i:=matrix(n1,1,[seq(ff.j,j=0..n1-1)]);

  od; i:='i': j:='j':


#begin perturbation computation.

#compute initial perturbation results.


  Id:= array(identity, 1..n1,1..n1);

  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);

  f0:=matrix(n1,1,[1,0$n1-1]);

  #f0t:=matrix(1,n1,[1,0$n1-1]);

  L1:=matrix(1,1,[0]);

  f1:=matrix(n1,1, [0$n1]);

  L2:=matrix(1,1,[0]);

  f2:=matrix(n1,1,[0$n1-1,1]);

  L3:=matrix(1,1,[0]);


#set up recurring formula for perturbation computation and

#compute the eigenvalues and the eigenvectors.
```

```
for i from 3 to p do
     U.i:= (Id-u0)&*f.i=u1&*f.(i-1)+u2&*f.(i-2)-
     convert([seq(f.(i-q-3)&*L.(q+3),  q=0..i-3)],`+`);q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[ii,1]= rhs(U.i)[ii,1];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(n1,1,[0,seq(ff.k,k=1..n1-1)]);
          readlib(unassign):
           unassign(evaln(ff.(1..n1-1)));
          L.(i+1):=matrix(1,1,[f.i[2,1]]);
     od;

#matrices used on the perturbation computation


U_0:=u0:
U_1:=u1:
U_2:=u2:


#eigenvalues and eigenvectors series


cv:=sort(convert([seq(b^i*f.i, i=0..p)],`+`)):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],`+`);
```

```
end;



#This computes the associated two-dimensional partition function on a square
#lattice for the lower right negative Kramers and Wannier matrix defined on the
#RIGHT by perturbation method given in series.

d2rnpps:= proc(n,p) local c1,c2,Id,i,ii,j,k,ms,n1,q,u0,u1,u2;
#'n' is the size of the matrices and 'p' is the length of the series
   #options trace;
    n1:=2^n/2;

#compute the needed matrices.

     u0:=matrix(n1,n1,[]):
       for i to n1 do
          for j to n1 do
           if j=2*i-1 then u0[i,j]:=1
           else           u0[i,j]:=0
          fi;
         od;
      od;
    u1:=matrix(n1,n1,[]):
     for i to n1 do
       for j to n1 do
```

```
          if j=2*i then   u1[i,j]:=1
           elif  j=2*(n1-i)+2 then u1[i,j]:=-1
          else          u1[i,j]:=0
         fi;
       od;
     od;
  u2:=matrix(n1,n1,[]):
   for i to n1 do
     for j to n1 do
      if j=2*(n1-i)+1 then u2[i,j]:=-1
         else          u2[i,j]:=0
      fi;
     od;
   od;


#perturbation method begins here.
#set up array to be used in the perturbation computation


  for i from 0 to p do
   f.i:=matrix(n1,1,[seq(ff.j,j=0..n1-1)]);
  od; i:='i': j:='j':


#begin perturbation computation.
#compute initial perturbation results.
```

```
  Id:= array(identity, 1..n1,1..n1);
  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);
  f0:=matrix(n1,1,[1,0$n1-1]);
  #f0t:=matrix(1,n1,[1,0$n1-1]);
  L1:=matrix(1,1,[0]);
  f1:=matrix(n1,1, [0$n1]);
  L2:=matrix(1,1,[0]);
  f2:=matrix(n1,1,[0$n1-1,1]);
  L3:=matrix(1,1,[0]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

for i from 3 to p do
     U.i:= (Id-u0)&*f.i=u1&*f.(i-1)+u2&*f.(i-2)-
     convert([seq(f.(i-q-3)&*L.(q+3),  q=0..i-3)],`+`);q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[ii,1]= rhs(U.i)[ii,1];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(n1,1,[0,seq(ff.k,k=1..n1-1)]);
          readlib(unassign):
           unassign(evaln(ff.(1..n1-1)));
          L.(i+1):=matrix(1,1,[f.i[2,1]]);
```

```
    od;

#matrices used on the perturbation computation

U_0:=u0:
U_1:=u1:
U_2:=u2:

#eigenvalues and eigenvectors series

cv:=sort(convert([seq(b^i*f.i, i=0..p)],'+')):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],'+');
end;



#This program calculates the complete V matrix  of Kramers and Wannier

d2cvmat:= proc(n) local c1,c2,a,b,u0,u1,u2,f0,f2,n1;
#options trace;
if nargs=1 then
   a:=exp(2*K): b:=exp(-2*K):
  else
    a:=args[2]; b:=args[3];
fi;
```

```
      n1:=2^n/2;
  u0:=matrix(n1,n1,[]):
       for i to n1 do
          for j to n1 do
           if j=2*i-1 then u0[i,j]:=a
           elif j=2*i then u0[i,j]:=1
           else           u0[i,j]:=0
          fi;
         od;
      od;
u2:=matrix(n1,n1,[]):
   for i to n1 do
     for j to n1 do
      if j=2*(n1-i)+1 then u2[i,j]:=b
      elif j=2*(n1-i)+2 then u2[i,j]:=1
         else          u2[i,j]:=0
      fi;
     od;
   od;
   cvm:=augment(stack(u0,u2),stack(u2,u0));


  end;


#This program calculates the lower negative V matrix
```

```
d2lnvmat:= proc(n) local c1,c2,a,b,u0,u1,u2,f0,f2,n1;
#options trace;
if nargs=1 then
   a:=exp(2*K): b:=exp(-2*K):
  else
    a:=args[2]; b:=args[3];
fi;
      n1:=2^n/2;
  u0:=matrix(n1,n1,[]):
       for i to n1 do
          for j to n1 do
           if j=2*i-1 then u0[i,j]:=a
           elif j=2*i then u0[i,j]:=1
           else           u0[i,j]:=0
          fi;
         od;
      od;
 u2:=matrix(n1,n1,[]):
   for i to n1 do
     for j to n1 do
      if j=2*(n1-i)+1 then u2[i,j]:=-b
      elif j=2*(n1-i)+2 then u2[i,j]:=-1
         else          u2[i,j]:=0
      fi;
     od;
   od;

:

     f0:=delrows(u0,n1/2+1..n1);

    f2:=delrows(u2,1..n1/2);

    stack(f0,f2);


  end;


#This program calculates the upper V matrix of Kramers and Wannier.


d2upvmat:= proc(n) local c1,c2,a,b,f0,f2,u0,u1,u2,n1;

#options trace;

if nargs=1 then

   a:=exp(2*K): b:=exp(-2*K):

  else

    a:=args[2]; b:=args[3];

fi;

      n1:=2^n/2;

   u0:=matrix(n1,n1,[]):

       for i to n1 do

          for j to n1 do

           if j=2*i-1 then u0[i,j]:=a

           elif j=2*i then u0[i,j]:=1

           else           u0[i,j]:=0

          fi;
```

```
         od;

      od;

 u2:=matrix(n1,n1,[]):

   for i to n1 do

     for j to n1 do

      if j=2*(n1-i)+1 then u2[i,j]:=b

      elif j=2*(n1-i)+2 then u2[i,j]:=1

         else          u2[i,j]:=0

      fi;

     od;

   od;

    f0:=delrows(u0,n1/2+1..n1);

    f2:=delrows(u2,1..n1/2);

   stack(f0,f2);


  end;


$This is the transformation matrix which transforms the a square matrix into

#left upper and right lower parts. In particular the complete Kramer and

#Wannier matrix is transformed into two V matrices.


stmatrix:=proc(n) local H1,H2,H,n1 ;

#options trace;

n1:=2^n;

H1:=array(identity,1..n1/2,1..n1/2);
```

```
H2:=evalm(-array(identity,1..n1/2,1..n1/2));
H:=copyinto(H2,stack(augment(H1,H1),augment(H1,H1)),n1/2+1,n1/2+1);
end;
```

The three-dimensional perturbation method follows

```
#This computes the associated three-dimensional partition function on a cubic
#lattice for the lower right negative Kramers and Wannier matrix defined on the
#LEFT by perturbation method given in series.

d3lnpps:=proc(n,p) local n1,j1,P1,01,P2,k;
#'n' is the size of the matrices and 'p' is the length of the series
  #options trace;
  with(linalg):
    n1:=2^n/2;
if type(n,odd) then
  k:=1;
elif type(n, even) then
  k:=2;
else ERROR(`n must be a positive integer`);
fi;
     if n<3 then
      ERROR(`n must be 3 or greater`);
     fi;
    if n >=3  then
       j1:=n-1-k;
```

```
        nb:=k;
    fi;

#create the needed matrices.

            P1:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2*i-1 then P1[i,j]:=1:
           else           P1[i,j]:=0:
          fi:
         od:
      od:
print(P1);

P2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2*i then P2[i,j]:=1:
           else           P2[i,j]:=0:
          fi:
         od:
      od:
print(P2);
Q1:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2^j1-1-2*(i-1) then Q1[i,j]:=1:
           else           Q1[i,j]:=0:
          fi:
         od:
      od:
print(Q1);
Q2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2^j1-2*(i-1) then Q2[i,j]:=1:
           else           Q2[i,j]:=0:
          fi:
         od:
      od:
print(Q2);


u0:=matrix(n1,n1,0);
for i from 1 to nb do
     u0:=copyinto(P1,u0,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
od;


u1:=matrix(n1,n1,0);
for i from 1 to nb do
```

```
     u1:=copyinto(P2,u1,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     u1:=copyinto(P1,u1,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u1:=copyinto(evalm(-Q2),u1,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
od;

u2:=matrix(n1,n1,0);
for i from 1 to nb do
     u2:=copyinto(P2,u2,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u2:=copyinto(evalm(-Q1),u2,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
     u2:=copyinto(evalm(-Q2),u2,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

u3:=matrix(n1,n1,0);
for i from 1 to nb do
     u3:=copyinto(evalm(-Q1),u3,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

u3:=matrix(n1,n1,0);
for i from 1 to nb do
     u3:=copyinto(Q1,u3,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

#perturbation method begins here.
#define array to be used in the computation
```

```
for i from 0 to p do #define array to be used in the computation
     f.i:=matrix(1,n1,[seq(ff.j,j=0..n1-1)]);
  od; i:='i': j:='j':


#begin perturbation computation.
#compute initial perturbation results.


  Id:= array(identity, 1..n1,1..n1);
  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);
  f0:=matrix(1,n1,[1,0$n1-1]);
 #f0t:=matrix(1,n1,[1,0$n1-1]); if each eqn is multiplied through by this
 #matrix the L's are easily computed
   L1:=matrix(1,1,[0]);
  ls:=map(evalm,f1&*(Id-u0)=f0&*u1);
    for i from 1 to n1 do
      m.i:=lhs(ls)[1,i]=rhs(ls)[1,i];
    od;
  assign(solve({seq(m.i,i=1..n1)},{seq(ff.k,k=1..n1-1)}));
  f1:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
  readlib(unassign):
  unassign(evaln(ff.(1..n1-1)));
  L2:=matrix(1,1,[0]);


 ls:=map(evalm,f2&*(Id-u0)=f1&*u1+f0&*u2);
    for i from 1 to n1 do
```

```
      m.i:=lhs(ls)[1,i]=rhs(ls)[1,i];
    od;
  assign(solve({seq(m.i,i=1..n1)},{seq(ff.k,k=1..n1-1)}));
  f2:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
  readlib(unassign):
  unassign(evaln(ff.(1..n1-1)));
  L3:=matrix(1,1,[0]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

  for i from 3 to p do
     U.i:= f.i&*(Id-u0)=f.(i-1)&*u1+f.(i-2)&*u2+f.(i-3)&*u3-
     convert([seq(L.(q+3)&*f.(i-q-3),  q=0..i-3)],'+');q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[1,ii]= rhs(U.i)[1,ii];
       od;ii:='ii':
    assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
    f.i:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
    readlib(unassign):
    unassign(evaln(ff.(1..n1-1)));
    L.(i+1):=matrix(1,1,[f.(i-2)[1,n1]]);
   od;
```

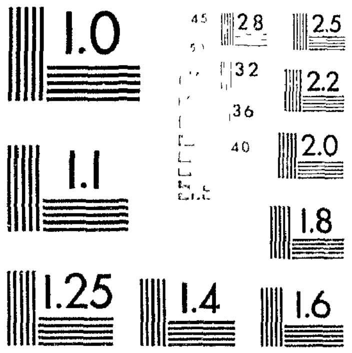
```
#matrices used on the perturbation computation
U_0:=u0:
U_1:=u1:
U_2:=u2:
U_3:=u3:

#eigenvalues and eigenvectors series

cv:=sort(convert([seq(b^i*f.i, i=0..p)],'+')):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],'+');
end;

#This computes the associated three-dimensional partition function on a cubic
#lattice for the lower right negative Kramers and Wannier matrix defined on
#RIGHT by perturbation method given in series.

d3rnpps:=proc(n,p) local Id,n1,j1,P1,P2,Q1,Q2,k;
#'n' is the size of the matrices and 'p' is the length of the series
  #options trace;
    n1:=2^n/2;
if type(n,odd) then
 k:=1;
elif type(n, even) then
 k:=2;
```

```
else ERROR('n must be a positive integer');
fi;
     if n<3 then
      ERROR('n must be 3 or greater');
     fi;
    if n >=3  then
        j1:=n-1-k;
        nb:=k;
    fi;
#compute the needed matrices.

             P1:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2*i-1 then P1[i,j]:=1:
           else           P1[i,j]:=0:
          fi:
         od:
      od:
print(P1);

P2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2*i then P2[i,j]:=1:
```

```
          else            P2[i,j]:=0:
         fi:
        od:
     od:
print(P2);
Q1:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
         for j to 2^j1 do
          if j=2^j1-1-2*(i-1) then Q1[i,j]:=1:
          else            Q1[i,j]:=0:
         fi:
        od:
     od:
print(Q1);
Q2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
         for j to 2^j1 do
          if j=2^j1-2*(i-1) then Q2[i,j]:=1:
          else            Q2[i,j]:=0:
         fi:
        od:
     od:
print(Q2);

u0:=matrix(n1,n1,0);
```

3 of/de 3

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT



```
for i from 1 to nb do

     u0:=copyinto(P1,u0,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);

od;


u1:=matrix(n1,n1,0);

for i from 1 to nb do

     u1:=copyinto(P2,u1,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);

     u1:=copyinto(P1,u1,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);

     u1:=copyinto(evalm(-Q2),u1,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);

od;


u2:=matrix(n1,n1,0);

for i from 1 to nb do

     u2:=copyinto(P2,u2,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);

     u2:=copyinto(evalm(-Q1),u2,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);

     u2:=copyinto(evalm(-Q2),u2,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);

od;


u3:=matrix(n1,n1,0);

for i from 1 to nb do

     u3:=copyinto(evalm(-Q1),u3,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);

od;


#perturbation method begins here.

#set up array to be used in the perturbation computation
```

```
 for i from 0 to p do
     f.i:=matrix(n1,1,[seq(ff.j,j=0..n1-1)]);
  od;

#begin perturbation computation.
#compute initial perturbation results.


  Id:= array(identity, 1..n1,1..n1);
  L0:=matrix(1,1,[1]);
  f0:=matrix(n1,1,[1,0$n1-1]);
 #f0t:=matrix(1,n1,[1,0$n1-1]); if each eqn is multiplied through by this
 #matrix the L's are easily computed
  L1:=matrix(1,1,[0]);
  f1:=matrix(n1,1,[0$n1]);
  L2:=matrix(1,1,[0]);
  f2:=matrix(n1,1,[0$n1]);
  L3:=matrix(1,1,[0]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

for i from 3 to p do
     U.i:= (Id-u0)&*f.i=u1&*f.(i-1)+u2&*f.(i-2)+u3&*f.(i-3)-
     convert([seq(f.(i-q-3)&*L.(q+3),  q=0..i-3)],'+');q:='q':
```

```
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[ii,1]= rhs(U.i)[ii,1];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(n1,1,[0,seq(ff.k,k=1..n1-1)]);
          readlib(unassign):
           unassign(evaln(ff.(1..n1-1)));
          L.(i+1):=matrix(1,1,[f.i[2,1]]);
     od;


#matrices used on the perturbation computation
U_0:=u0:
U_1:=u1:
U_2:=u2:
U_3:=u3:


#eigenvalues and eigenvectors series


cv:=sort(convert([seq(b^i*f.i, i=0..p)],`+`)):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],`+`);
end;


#This computes the associated three-dimensional partition function on a cubic
```

```
#lattice for the left upper left positive Kramers and Wannier matrix defined

# on the LEFT by perturbation method given in series.


d3lppps:=proc(n,p) local n1,j1,P1,O1,P2,k;

#'n' is the size of the matrices and 'p' is the length of the series

# options trace;

    n1:=2^n/2;

if type(n,odd) then

 k:=1;

elif type(n, even) then

 k:=2;

else ERROR(`n must be a positive integer`);

fi;

     if n<3 then

      ERROR(`n must be 3 or greater`);

     fi;

    if n >=3  then

        j1:=n-1-k;

        nb:=k;

    fi;

#compute the needed matrices.


             P1:=matrix(2^(j1-1),2^j1,[]):

       for i to 2^(j1-1) do

          for j to 2^j1 do
```

```
            if j=2*i-1 then P1[i,j]:=1:
            else            P1[i,j]:=0:
           fi:
          od:
      od:
print(P1);


P2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
           for j to 2^j1 do
            if j=2*i then P2[i,j]:=1:
            else            P2[i,j]:=0:
           fi:
          od:
      od:
print(P2);
Q1:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
           for j to 2^j1 do
            if j=2^j1-1-2*(i-1) then Q1[i,j]:=1:
            else            Q1[i,j]:=0:
           fi:
          od:
      od:
print(Q1);
```

```
Q2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2^j1-2*(i-1) then Q2[i,j]:=1:
           else           Q2[i,j]:=0:
          fi:
         od:
      od:
print(Q2);

u0:=matrix(n1,n1,0);
for i from 1 to nb do
     u0:=copyinto(P1,u0,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
od;

u1:=matrix(n1,n1,0);
for i from 1 to nb do
     u1:=copyinto(P2,u1,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     u1:=copyinto(P1,u1,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u1:=copyinto(Q2,u1,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
od;

u2:=matrix(n1,n1,0);
for i from 1 to nb do
     u2:=copyinto(P2,u2,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u2:=copyinto(Q1,u2,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);

     u2:=copyinto(Q2,u2,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);

od;


u3:=matrix(n1,n1,0);

for i from 1 to nb do

     u3:=copyinto(Q1,u3,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);

od;


#perturbation method begins here.

#define array to be used in the computation


for i from 0 to p do

     f.i:=matrix(1,n1,[seq(ff.j,j=0..n1-1)]);

  od; i:='i': j:='j':


#begin perturbation computation.

#compute initial perturbation results.


  Id:= array(identity, 1..n1,1..n1);

  L0:=matrix(1,1,[1]);   #linsolve(multiply(u0,f0),f0);

  f0:=matrix(1,n1,[1,0$n1-1]);

#f0t:=matrix(1,n1,[1,0$n1-1]); if each eqn is multiplied through by this

#matrix the L's are easily computed

  L1:=matrix(1,1,[0]);
  ls:=map(evalm,f1&*(Id-u0)=f0&*u1);
    for i from 1 to n1 do
      m.i:=lhs(ls)[1,i]=rhs(ls)[1,i];
    od;
  assign(solve({seq(m.i,i=1..n1)},{seq(ff.k,k=1..n1-1)}));
  f1:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
  readlib(unassign):
  unassign(evaln(ff.(1..n1-1)));
  L2:=matrix(1,1,[0]);

 ls:=map(evalm,f2&*(Id-u0)=f1&*u1+f0&*u2);
    for i from 1 to n1 do
      m.i:=lhs(ls)[1,i]=rhs(ls)[1,i];
    od;
  assign(solve({seq(m.i,i=1..n1)},{seq(ff.k,k=1..n1-1)}));
  f2:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
  readlib(unassign):
  unassign(evaln(ff.(1..n1-1)));
  L3:=matrix(1,1,[0]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

  for i from 3 to p do
     U.i:= f.i&*(Id-u0)=f.(i-1)&*u1+f.(i-2)&*u2+f.(i-3)&*u3-
```

```
      convert([seq(L.(q+3)&*f.(i-q-3),  q=0..i-3)],`+`);q:='q':
      U.i:=map(evalm,U.i);
        for ii from 1 to n1 do
           ms[ii]:=lhs(U.i)[1,ii]= rhs(U.i)[1,ii];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(1,n1,[0,seq(ff.k,k=1..n1-1)]);
     readlib(unassign):
     unassign(evaln(ff.(1..n1-1)));
     L.(i+1):=matrix(1,1,[f.(i-2)[1,n1]]);
    od;

#matrices used on the perturbation computation
U_0:=u0:
U_1:=u1:
U_2:=u2:
U_3:=u3:

#eigenvalues and eigenvectors series


cv:=sort(convert([seq(b^i*f.i, i=0..p)],`+`)):
#cv:=map(evalm,cv);
convert([seq(b^i*L.i[1,1], i=0..p)],`+`);
end;
```

```
#This computes the associated three-dimensional partition function on a cubic
#lattice for the upper left positive Kramers and Wannier matrix defined on the
#RIGHT by perturbation method given in series.


d3rppps:=proc(n,p) local Id,n1,j1,P1,P2,Q1,Q2,k;
#'n' is the size of the matrices and 'p' is the length of the series
 # options trace;
    n1:=2^n/2;
if type(n,odd) then
 k:=1;
elif type(n, even) then
 k:=2;
else ERROR(`n must be a positive integer`);
fi;
     if n<3 then
      ERROR(`n must be 3 or greater`);
     fi;
    if n >=3  then
        j1:=n-1-k;
        nb:=k;
    fi;
#compute the needed matrices.


      P1:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
```

```
            for j to 2^j1 do
             if j=2*i-1 then P1[i,j]:=1:
             else            P1[i,j]:=0:
            fi:
           od:
       od:
print(P1);


P2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
            for j to 2^j1 do
             if j=2*i then P2[i,j]:=1:
             else            P2[i,j]:=0:
            fi:
           od:
       od:
print(P2);
Q1:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
            for j to 2^j1 do
             if j=2^j1-1-2*(i-1) then Q1[i,j]:=1:
             else            Q1[i,j]:=0:
            fi:
           od:
       od:
```

```
print(Q1);
Q2:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if j=2^j1-2*(i-1) then Q2[i,j]:=1:
           else          Q2[i,j]:=0:
          fi:
         od:
      od:
print(Q2);


u0:=matrix(n1,n1,0);
for i from 1 to nb do
     u0:=copyinto(P1,u0,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
od;


u1:=matrix(n1,n1,0);
for i from 1 to nb do
     u1:=copyinto(P2,u1,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     u1:=copyinto(P1,u1,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u1:=copyinto(Q2,u1,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
od;


u2:=matrix(n1,n1,0);
for i from 1 to nb do
```

```
     u2:=copyinto(P2,u2,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     u2:=copyinto(Q1,u2,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
     u2:=copyinto(Q2,u2,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

u3:=matrix(n1,n1,0);
for i from 1 to nb do
     u3:=copyinto(Q1,u3,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

#perturbation method begins here.
#set up array to be used in the perturbation computation

 for i from 0 to p do
     f.i:=matrix(n1,1,[seq(ff.j,j=0..n1-1)]);
  od;

#begin perturbation computation.
#compute initial perturbation results.

  Id:= array(identity, 1..n1,1..n1);
  L0:=matrix(1,1,[1]);
  f0:=matrix(n1,1,[1,0$n1-1]);
#f0t:=matrix(1,n1,[1,0$n1-1]); if each eqn is multiplied through by this
#matrix the L's are easily computed
```

```
  L1:=matrix(1,1,[0]);
  f1:=matrix(n1,1,[0$n1]);
  L2:=matrix(1,1,[0]);
  f2:=matrix(n1,1,[0$n1]);

#set up recurring formula for perturbation computation and
#compute the eigenvalues and the eigenvectors.

for i from 3 to p do
     U.i:= (Id-u0)&*f.i=u1&*f.(i-1)+u2&*f.(i-2)+u3&*f.(i-3)-
     convert([seq(f.(i-q-3)&*L.(q+3),  q=0..i-3)],`+`);q:='q':
     U.i:=map(evalm,U.i);
       for ii from 1 to n1 do
          ms[ii]:=lhs(U.i)[ii,1]= rhs(U.i)[ii,1];
        od;ii:='ii':
     assign(solve({seq(ms[k],k=1..n1)},{seq(ff.k,k=1..n1-1)}));
     f.i:=matrix(n1,1,[0,seq(ff.k,k=1..n1-1)]);
          readlib(unassign):
           unassign(evaln(ff.(1..n1-1)));
          L.(i+1):=matrix(1,1,[f.i[2,1]]);
     od;

#matrices used on the perturbation computation
U_0:=u0:
U_1:=u1:
```

```
U_2:=u2:

U_3:=u3:


#eigenvalues and eigenvectors series


cv:=sort(convert([seq(b^i*f.i, i=0..p)],'+')):

#cv:=map(evalm,cv);

convert([seq(b^i*L.i[1,1], i=0..p)],'+');

end;


$This is the transformation matrix which transforms the a square matrix into

#left upper and right lower parts. In particular the complete Kramers and #Wann:


d2stmat:=proc(n) local H1,H2,H,n1 ;

#options trace;

n1:=2^n;

H1:=array(identity,1..n1/2,1..n1/2);

H2:=evalm(-array(identity,1..n1/2,1..n1/2));

H:=copyinto(H2,stack(augment(H1,H1),augment(H1,H1)),n1/2+1,n1/2+1);

end;


#This program calculates the complete V matrix  of Kramers and Wannier


d3cvmat:= proc(n) local P,Q,a,b,R,S,j1,nb,k,V0,V2;

#options trace;
```

```
with(linalg):
if nargs=1 then
   a:=exp(K): b:=exp(-K):
  else
    a:=args[2]; b:=args[3];
fi;
 n1:=2^n/2;
if type(n,odd) then
 k:=1;
elif type(n, even) then
 k:=2;
else ERROR(`n must be a positive integer`);
fi;
     if n<3 then
      ERROR(`n must be 3 or greater`);
     fi;
    if n >=3  then
      j1:=n-1-k;
      nb:=k;
   fi;
#compute the needed matrices.


P:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
          for j to 2^j1 do
          if   j=2*i-1 then P[i,j]:=a^3:
          elif j=2*i then   P[i,j]:=a:
          else              P[i,j]:=0:
         fi:
        od:
     od:

Q:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
    for j to 2^j1 do
      if         j=2*i-1 then Q[i,j]:=a:
      elif       j=2*i then   Q[i,j]:=b:
      else                    Q[i,j]:=0:
          fi:
         od:
      od:

R:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  R[i,j]:=b^3:
           elif  j=2^j1-2*(i-1) then    R[i,j]:=b:
           else                         R[i,j]:=0:
          fi:
         od:
```

```
      od:


S:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  S[i,j]:=b:
           elif  j=2^j1-2*(i-1) then    S[i,j]:=a:
           else                         S[i,j]:=0:
          fi:
         od:
      od:
#the block of matrices needed


V0:=matrix(n1,n1,0);
for i from 1 to nb do
     V0:=copyinto(P,V0,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     V0:=copyinto(Q,V0,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
od;


V2:=matrix(n1,n1,0);
for i from 1 to nb do
     V2:=copyinto(S,V2,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
     V2:=copyinto(R,V2,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;
#gather matrices together to form the d3 complete V matrix
```

```
 augment(stack(V0,V2),stack(V2,V0));

end;

#This program calculates the complete V matrix  of Kramers and Wannier

d3lnvmat:= proc(n) local P,Q,a,b,R,S,j1,nb,k,V;
#options trace;
with(linalg):
if nargs=1 then
   a:=exp(K): b:=exp(-K):
  else
    a:=args[2]; b:=args[3];
fi;
 n1:=2^n/2;
if type(n,odd) then
   k:=1;
elif type(n, even) then
   k:=2;
else ERROR('n must be a positive integer');
  fi;
#k:=round((n-3)/2);
  if n<3 then
      ERROR('n must be 3 or greater');
  fi;
  if n >=3  then
        j1:=n-1-k;
        nb:=k;
 fi;
#compute the needed matrices.

P:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
          for j to 2^j1 do
           if   j=2*i-1 then P[i,j]:=a^3:
           elif j=2*i then   P[i,j]:=a:
           else              P[i,j]:=0:
          fi:
         od:
      od:

Q:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
    for j to 2^j1 do
      if        j=2*i-1 then Q[i,j]:=a:
      elif      j=2*i then   Q[i,j]:=b:
      else                   Q[i,j]:=0:
          fi:
         od:
      od:
```

```
R:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  R[i,j]:=b^3:
           elif  j=2^j1-2*(i-1) then    R[i,j]:=b:
           else                         R[i,j]:=0:
          fi:
         od:
      od:

S:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  S[i,j]:=b:
           elif  j=2^j1-2*(i-1) then    S[i,j]:=a:
           else                         S[i,j]:=0:
          fi:
         od:
      od:

V:=matrix(n1,n1,0);
for i from 1 to nb do
     V:=copyinto(P,V,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     V:=copyinto(Q,V,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     V:=copyinto(evalm(-S),V,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
     V:=copyinto(evalm(-R),V,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

end;

#This program calculates the complete V matrix  of Kramers and Wannier
d3upvmat:= proc(n) local P,Q,a,b,R,S,j1,nb,k,V;
#options trace;
with(linalg):
if nargs=1 then
   a:=exp(K): b:=exp(-K):
  else
    a:=args[2]; b:=args[3];
fi;
 n1:=2^n/2;
if type(n,odd) then
   k:=1;
elif type(n, even) then
   k:=2;
else ERROR('n must be a positive integer');
  fi;
#k:=round((n-3)/2);
  if n<3 then
      ERROR('n must be 3 or greater');
  fi;
  if n >=3  then
        j1:=n-1-k;
        nb:=k;
 fi;

#compute the needed matrices.
P:=matrix(2^(j1-1),2^j1,[]): #the constant matrix U0
       for i to 2^(j1-1) do
          for j to 2^j1 do
           if   j=2*i-1 then P[i,j]:=a^3:
           elif j=2*i then   P[i,j]:=a:
           else              P[i,j]:=0:
          fi:
         od:
      od:

Q:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
    for j to 2^j1 do
      if         j=2*i-1 then Q[i,j]:=a:
      elif       j=2*i then   Q[i,j]:=b:
      else                    Q[i,j]:=0:
          fi:
         od:
```

```
      od:


R:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  R[i,j]:=b^3:
           elif  j=2^j1-2*(i-1) then    R[i,j]:=b:
           else                         R[i,j]:=0:
          fi:
         od:
      od:


S:=matrix(2^(j1-1),2^j1,[]):
   for i to 2^(j1-1) do
          for j to 2^j1 do
           if    j=2^j1-1-2*(i-1) then  S[i,j]:=b:
           elif  j=2^j1-2*(i-1) then    S[i,j]:=a:
           else                         S[i,j]:=0:
          fi:
         od:
      od:


V:=matrix(n1,n1,0);
for i from 1 to nb do
     V:=copyinto(P,V,(2^(j1-1))*2*(i-1)+1,2^j1*2*(i-1)+1);
     V:=copyinto(Q,V,2^(j1-1)*(2*i-1)+1,2^j1*(2*i-1)+1);
     V:=copyinto(S,V,n1/2+2^(j1-1)*2*(i-1)+1,n1-2^j1*(2*i-1)+1);
     V:=copyinto(R,V,n1/2+2^(j1-1)*(2*i-1)+1,n1-2^j1*2*i+1);
od;

end;

CRITEMP
#This function computes the ctitical temperature.
# s is the different states which each lattice site can take.
# if s=2 we get the Ising lattice which is a typical model of phase
#transition of the second order.
critemp:=proc(s) local x,CT:
if type(s,algebraic) then
  CT=1/2*ln(1+s^(1/2));
elif type(s,positive) and type(s,integer) then
  CT=1/2*ln(1+s^(1/2));
else ERROR ('s must be a positive integer') fi;
end;

D2MCLTPF
# s is the different states which each lattice site can take.
# if s=2 we get the Ising lattice which is a typical model of phase
#transition of the second order.
#potts low temp series partition function energy and specific heat per site
```

```
#many components high temp. partition function

d2mcltpf:=proc(n) local x,i,a:

if nargs =1 and  type(n,integer) then

      s:=2:

      x:=exp(-J/(k*T)):

 elif nargs =2  and type(args[2],integer)  then

      s:=args[2]:

      x:=exp(-J/(k*T)):

elif nargs =3 and type(args[2],integer) and not type(args[3],integer)  then

     s:=args[2]:  x:=args[3]:

 fi;

if type(n,integer) and n>16 then ERROR(`series is not computed beyond 16`);

elif nargs =1 and not type(n,integer) then

  x:=exp(-J/(k*T));

 x^(-1)*(1+(s-1)*sum(a(i)*x^i, i=4..n)):

 D2MCLTE:= U/N=k*T^2*diff(ln("),T):

 D2MCLTSH:=C/N=1/N*diff(rhs(D2MCLTE),T):

"""";

elif type(n,positive) and type(n,integer) then

  a(4):= 1:

  a(5):= 0:

  a(6):= 2:

  a(7):= 2*(s-2):

  a(8):= -2*s+9:

  a(9):= 12*(s-2):
```

```
  a(10):= 2*(3*s^2-16*s+27):
  a(11):= 2*(s-2)*(-5*s+32):
  a(12):= s^3+55*s^2-243*s+302:
  a(13):= 2*(s-2)*(9*s^2-71*s+180):
  a(14):= -37*s^3+561*s^2-1802*s+1808:
  a(15):= 2*(s-2)*(4*s^3+130*s^2-704*s+1189):
  a(16):= 55*s^4-998*s^3+6269*s^2-15162*s+12870:
  x^(-1)*(1+(s-1)*sum(a(i)*x^i, i=4..n)):
 D2MCLTE:= E/N=k*T^2*diff(ln("),T):
 D2MCLTSH:=C/N=1/N*diff(rhs(D2MCLTE),T):
 """;
 fi;
end;


D2MCHTPF
# s is the different states which each lattice site can take.
# if s=2 we get the Ising lattice which is a typical model of phase
#transition of the second order.
#Pots high temp specific heat per site.
#many components high temp. partition function
d2mchtpf:=proc(n) local a,i,x,u;
if nargs =1 and  type(n,integer) then
      s:=2:
      x:=exp(-J/(k*T)):
 elif nargs =2  and type(args[2],integer)  then
```

```
        s:=args[2]:
        x:=exp(-J/(k*T)):
elif nargs =3 and type(args[2],integer) and not type(args[3],integer)  then
      s:=args[2]:  x:=args[3]:
 fi;
if type(n,integer) and n>16 then ERROR(`series is not computed beyond 16`);
elif  nargs =1 and not type(n,integer) then
  x:=exp(-J/(k*T)):
  u:=(1-x)/(1+(s-1)*x):
 s/((1-u)*(1+(s-1)^u))*(1+(s-1)*Sum(a(i)*u^i, i=4..n)):
 D2MCHTE:= U/N=k*T^2*diff(ln("),T):
 D2MCHTSH:=C/N=1/N*diff(rhs(D2MCHTE),T):
 """;
elif type(n,positive) and type(n,integer) then
a(4):= 1:
a(5):= 0:
a(6):= 2:
a(7):= 2*(s-2):
a(8):= -2*s+9:
a(9):= 12*(s-2):
a(10):= 2*(3*s^2-16*s+27):
a(11):= 2*(s-2)(-5*s+32):
a(12):= s^3+55*s^2-243*s+302:
a(13):= 2*(s-2)*(9*s^2-71*s+180):
a(14):= -37*s^3+561*s^2-1802*s+1808:
```

```
a(15):= 2*(s-2)*(4*s^3+130*s^2-704*s+1189):

a(16):= 55*s^4-998*s^3+6269*s^2-15162*s+12870:

u:=(1-x)/(1+(s-1)*x):

s/((1-u)*(1+(s-1)^u))*(1+(s-1)*sum(a(i)*u^i, i=4..n)):

D2MCHTE:= U=N*k*T^2*diff(ln("),T):

D2MCHTSH:=C/N=1/N*diff(rhs(D2MCHTE),T):

"""";

 fi;

end;


GRSTEN:

#This calculates the ground state entropy

grsten:=proc(q,n) local i,W_tri;

    options trace;

    if q=3 and n='sq' then

      RETURN( W_sq(3)=(4/3)^(3/2));

    elif q=4 and type(n, posint) and args[3]='tri'then

     RETURN( W_tri(4)=product((3*i-1)^2/(3*i*(3*i-2)),i=1..n));

    elif q=3 and type(n, posint) and args[3]='kagome' then

         W_tri(4):=product((3*i-1)^2/(3*i*(3*i-2`),i=1..n);

      RETURN( W_kagome(3)=(W_tri(4))^(1/3));

fi; end;
```

APPENDIX B

```
      PROGRAM POTTS
      PARAMETER(LX=64,LY=64,LW=32)
* LX, LY: lattice sizes in x and y directions
* LW=length of the computer word=0.5*lattice size in z direction
      PARAMETER(NSIZE=2*LX*LY-1)
*     DOUBLE PRECISION E,BETA,BET
      IMPLICIT INTEGER (A-Z)
      REAL E,BETA,BET
      COMPLEX MAG,CMAG
      DIMENSION SPIN1(0:NSIZE),SPIN2(0:NSIZE),DMN(3)

*         Initializing spins and demons
*  Parameter E determines the average energy per bond (ordered=0<E<1)
      E=0.75

      IE=LX*LY*LW*E
      K=1
      DO 1 I=LX*LY,1,-1
      SPIN1(2*I-1)=0
      SPIN1(2*I-2)=0
      SPIN2(2*I-K)=0
      K=3-K
      J=IE/I
```

```
        IE=IE-J
        SPIN2(2*I-K)=AMIX(MASK(J))
1       CONTINUE
ILNK=ILINK(SPIN1,SPIN2)
* WRITE(6,*) 'ILNK=',ILNK


        DMN(1)=0
        DMN(2)=0
        DMN(3)=0


*                SIMULATION
*       Parameters are
*               1- NSWEEP
*               2- NMEAS
*               3- NBATCH
*       One sweeps NSWEEP times, then a measurement is performed
*       This is repeated NMEAS times; then average measurements are printed
*       The entire process resumes NBATCH times
        NSWEEP=13
        NMEAS=5
        NBATCH=10


        DO 20 IBATCH=1,NBATCH
* Prepares accumulation of data
        IED=0
        CMAG=0
        DO 21 IMEAS=1,NMEAS
        DO 22 ISWEEP=1,NSWEEP
* scramble demons
        DO 23 I=1,3
23      DMN(I)=AMIX(DMN(I))
* Performs one simulation step
        CALL MONTE(SPIN1,SPIN2,DMN)
ILNK=ILINK(SPIN1,SPIN2)
IDMN=IBCOUNT(DMN(1))+2*IBCOUNT(DMN(2))+4*IBCOUNT(DMN(3))
* WRITE(6,*)'ILNK=',ILNK,',IDMN=',IDMN,ILNK-IDMN
22      CONTINUE
* accumulates demon energy
        IED=IED+IBCOUNT(DMN(1))+2*IBCOUNT(DMN(2))+4*IBCOUNT(DMN(3))
        CMAG=CMAG+MAG(SPIN1,SPIN2)
21      CONTINUE
* computes the inverse temperature
        BET=BETA(FLOAT(IED)/FLOAT(LW*NMEAS*(NSIZE+1)))
        PRINT *,' Beta=',BET
        CMAG=CMAG/NMEAS
        PRINT *,' Magnetization=',CMAG
20      CONTINUE
        STOP
        END
```

```
      SUBROUTINE MONTE(SPIN1,SPIN2,DMN)
*     PARAMETER(LX=32,LY=32,LW=16)
      PARAMETER(LX=64,LY=64,LW=32)
* LX and LY are the dimensions of the lattice
      PARAMETER(NSIZE=2*LX*LY-1)
      PARAMETER(IHOP=13)
* This number IHOP should be prime with LX and LY
      IMPLICIT INTEGER (A-Z)
      DIMENSION SPIN1(0:NSIZE),SPIN2(0:NSIZE),DMN(3)
      DIMENSION NEIGH1(6),NEIGH2(6)
*     INLAT(I)=MOD(I,NSIZE+1)
* This function ensures periodic boundary conditions in the lattice
* If LX and LY are powers of 2, it can be advantageously replaced by
      INLAT(I)=AND(I,NSIZE)
      J=0
      K=1
      DO 1 I=0,NSIZE
* setting the neighbors
      NEIGH1(1)=SPIN1(J+K)
      NEIGH2(1)=SPIN2(J+K)
      NEIGH1(2)=ISHFTC(SPIN1(J+K),K,LW)
      NEIGH2(2)=ISHFTC(SPIN2(J+K),K,LW)
      NEIGH1(3)=SPIN1(INLAT(J+2))
      NEIGH2(3)=SPIN2(INLAT(J+2))
      NEIGH1(4)=SPIN1(INLAT(J-2))
      NEIGH2(4)=SPIN2(INLAT(J-2))
      NEIGH1(5)=SPIN1(INLAT(J+2*LX))
      NEIGH2(5)=SPIN2(INLAT(J+2*LX))
      NEIGH1(6)=SPIN1(INLAT(J-2*LX))
      NEIGH2(6)=SPIN2(INLAT(J-2*LX))
* Randomly changing the spin
*  SPIN1 SPIN2   CHOICE NEW1 NEW2   CHOICE NEW1 NEW2
*    0     0        0     1    0       1     0    1
*    0     1        0     0    0       1     1    0
*    1     0        0     0    1       1     0    0
      CHOICE=IRDBIT(0)
      NEW1=AND(NOT(SPIN1(J)),XOR(NOT(SPIN2(J)),CHOICE))
      NEW2=AND(NOT(SPIN2(J)),XOR(SPIN1(J),CHOICE))
* now compute 8 + energy_of_daemon + number_of_old_links_with_equal_spins
*   - number_of_new_links_with_equal_spins
      DP1=DMN(1)
      DP2=DMN(2)
      DP3=DMN(3)
      ACCEPT=-1
      DO 2 M=1,6
* adds old energy
      LINK=NOT(OR(XOR(SPIN1(J),NEIGH1(M)),XOR(SPIN2(J),NEIGH2(M))))
      CARRY1=AND(LINK,DP1)
      DP1=XOR(DP1,LINK)
      CARRY2=AND(CARRY1,DP2)
      DP2=XOR(DP2,CARRY1)
      CARRY1=AND(CARRY2,DP3)
      DP3=XOR(DP3,CARRY2)
      ACCEPT=XOR(ACCEPT,CARRY1)
* subtracts new energy
      LINK=NOT(OR(XOR(NEW1,NEIGH1(M)),XOR(NEW2,NEIGH2(M))))
      CARRY1=AND(LINK,NOT(DP1))
      DP1=XOR(DP1,LINK)
      CARRY2=AND(CARRY1,NOT(DP2))
      DP2=XOR(DP2,CARRY1)
      CARRY1=AND(CARRY2,NOT(DP3))
      DP3=XOR(DP3,CARRY2)
      ACCEPT=XOR(ACCEPT,CARRY1)
2     CONTINUE
* accepts or rejects the change
      SPIN1(J)=XOR(AND(ACCEPT,NEW1),AND(NOT(ACCEPT),SPIN1(J)))
      SPIN2(J)=XOR(AND(ACCEPT,NEW2),AND(NOT(ACCEPT),SPIN2(J)))
      DMN(1)=XOR(AND(ACCEPT,DP1),AND(NOT(ACCEPT),DMN(1)))
      DMN(2)=XOR(AND(ACCEPT,DP2),AND(NOT(ACCEPT),DMN(2)))
      DMN(3)=XOR(AND(ACCEPT,DP3),AND(NOT(ACCEPT),DMN(3)))
* End of the loop
      J=INLAT(J+IHOP)
      K=-K
1     CONTINUE
      RETURN
```

```
        END


        FUNCTION ILINK(SPIN1,SPIN2)
* Returns the number of unsatisfied links (hence the energy)
        PARAMETER(LX=64,LY=64,LW=32)
* LX and LY are the dimensions of the lattice
        PARAMETER(NSIZE=2*LX*LY 1)
        IMPLICIT INTEGER (A-Z)
        DIMENSION SPIN1(0:NSIZE),SPIN2(0:NSIZE)
        DIMENSION NEIGH1(6),NEIGH2(6)
*       INLAT(I)=MOD(I,NSIZE+1)
* This function ensures periodic boundary conditions in the lattice
* If LX and LY are powers of 2, it can be advantageously replaced by
        INLAT(I)=AND(I,NSIZE)
        K=1
ILINK=6*(NSIZE+1)*LW
        DO 1 J=0,NSIZE
* setting the neighbors
        NEIGH1(1)=SPIN1(J+K)
        NEIGH2(1)=SPIN2(J+K)
        NEIGH1(2)=ISHFTC(SPIN1(J+K),K,LW)
        NEIGH2(2)=ISHFTC(SPIN2(J+K),K,LW)
        NEIGH1(3)=SPIN1(INLAT(J+2))
        NEIGH2(3)=SPIN2(INLAT(J+2))
        NEIGH1(4)=SPIN1(INLAT(J-2))
        NEIGH2(4)=SPIN2(INLAT(J-2))
        NEIGH1(5)=SPIN1(INLAT(J+2*LX))
        NEIGH2(5)=SPIN2(INLAT(J+2*LX))
        NEIGH1(6)=SPIN1(INLAT(J-2*LX))
        NEIGH2(6)=SPIN2(INLAT(J-2*LX))
* now compute  number_of_links_with_equal_spins
        DP1=0
        DP2=0
        DP3=0
        DO 2 M=1,6
        LINK=NOT(OR(XOR(SPIN1(J),NEIGH1(M)),XOR(SPIN2(J),NEIGH2(M))))
        CARRY1=AND(LINK,DP1)
        DP1=XOR(DP1,LINK)
        CARRY2=AND(CARRY1,DP2)
        DP2=XOR(DP2,CARRY1)
        DP3=XOR(DP3,CARRY2)
2       CONTINUE
ILINK=ILINK-(IBCOUNT(DP1)+2*IBCOUNT(DP2)+4*IBCOUNT(DP3))
* End of the loop
        K=-K
1       CONTINUE
ILINK=ILINK/2
        RETURN
        END
```

```
      FUNCTION IRDBIT(INIT)
      IMPLICIT INTEGER (A-Z)
      DIMENSION ITAB(0:31)
      SAVE I18,I5,I2,I1,I0,IFIRST
      DATA IFIRST/0/,I18/0/,I5/13/,I2/16/,I1/17/,I0/18/
* Standard initialization (random bits, here given in hexadecimal
* for words with 64 bits; truncate if necessary)
*     DATA ITAB/
*    x '7EB722A0C9743C06'Z,'534AB95D97ECF94A'Z,
*    x 'FD3CD86EFCCC61DE'Z,'3B341E5A9A1160B4'Z,
*    x '5CDA1DE25BB8E8F5'Z,'76EDDA93192BC357'Z,
*    x '1CD3CF66101C4CBD'Z,'0C7216C2C95676A8'Z,
*    x 'ACF117D1EF24D606'Z,'AF452B2A2FB48E98'Z,
*    x '5E7C758368B24840'Z,'FF29D95A6F897866'Z,
*    x 'D1A46D4C9F62639A'Z,'CA05FFE020E049BD'Z,
*    x '7102A31B08C39D1E'Z,'E8DE18695A18CA02'Z,
*    x '98A33097B9C2250E'Z,'4556037DC5A2CC1A'Z,14*0/
      DATA ITAB/
     x Z'C9743C06',Z'97ECF94A',
     x Z'FCCC61DE',Z'9A1160B4',
     x Z'5BB8E8F5',Z'192BC357',
     x Z'101C4CBD',Z'C95676A8',
     x Z'EF24D606',Z'2FB48E98',
     x Z'68B24840',Z'6F897866',
     x Z'9F62639A',Z'20E049BD',
     x  Z'08C39D1E',Z'5A18CA02',
     x  Z'B9C2250E',Z'C5A2CC1A',14*0/
        IF(INIT.NE.0) THEN
* Initialization (with a check avoiding the generation of a sequence of zeros)
                ITAB(I1)=INIT
                J=INIT
                I=AND(31,I18+1)
                DO 1 K=1,16
                        J=OR(J,ITAB(I))
                        I=AND(31,I+1)
1               CONTINUE
                ITAB(I18)=OR(ITAB(I18),NOT(J))
        ENDIF
        ITAB(I0)=XOR(XOR(ITAB(I18),ITAB(I5)),XOR(ITAB(I2),ITAB(I1)))
        IRDBIT=ITAB(I0)
        I0=AND(31,I0+1)
        I1=AND(31,I1+1)
        I2=AND(31,I2+1)
        I5=AND(31,I5+1)
        I18=AND(31,I18+1)
        RETURN
        END

        INTEGER FUNCTION AMIX(WORD)
        PARAMETER(LW=32,LH=LW/2)
        IMPLICIT INTEGER (A-Z)
REAL RANF
        L=INT(LH*RANF())
        P1=AND(MASK(L),WORD)
        P2=AND(MASK(L),ISHFTC(WORD,-L,LW))
        P3=AND(NOT(MASK(2*L)),WORD)
        AMIX=ISHFTC(OR(OR(P2,ISHFTC(P1,L,LW)),P3),INT(LW*RANF()),LW)
        RETURN
        END

        FUNCTION BETA(DENERG)
        XL=0
        XH=1
        DO 1 N=1,36
        XN=0.5*(XL+XH)
        A=1-XN**8
        FN=(8*(A+XN-1)-7*A*XN)/A/(1-XN)
        IF(FN.GT.DENERG) THEN
                XH=XN
        ELSE
                XL=XN
        ENDIF
1       CONTINUE
        BETA=-0.25*ALOG(XN)
        RETURN
        END



      FUNCTION IBCOUNT(X)
      INTEGER Y,X,IBCOUNT
      IBCOUNT=0
      Y=X
      DO 1 N=1,16
*     IBCOUNT=IBCOUNT+AND('0001000100010001'X,Y)
      IBCOUNT=IBCOUNT+AND(X'00010001',Y)
    1 Y=ISHFT(Y,-1)
*     IBCOUNT=AND((IBCOUNT+SHIFT(IBCOUNT,16)
*    X+SHIFT(IBCOUNT,32)+SHIFT(IBCOUNT,48)),127)
      IBCOUNT=AND((IBCOUNT+ISHFT(IBCOUNT,-16)
     X ),127)
      RETURN
      END


        COMPLEX FUNCTION MAG(SPIN1,SPIN2)
*       PARAMETER(LX=32,LY=32,LW=16)
        PARAMETER(LX=64,LY=64,LW=32)
        PARAMETER(NSIZE=2*LX*LY-1)
        INTEGER SPIN1,SPIN2
        DIMENSION SPIN1(0:NSIZE),SPIN2(0:NSIZE)
        MAG=0
        DO 1 I=0,NSIZE
        MAG=MAG+CMPLX(
     $ LW-1.5*IBCOUNT(OR(SPIN1(I),SPIN2(I))),
     $ 0.732*(IBCOUNT(SPIN1(I))-IBCOUNT(SPIN2(I)))
     $ )
1       CONTINUE
        MAG=MAG/((NSIZE+1)*LW)
        RETURN
        END

        INTEGER FUNCTION MASK(L)
        MASK=ISHFT(1,L)-1
        RETURN
        END

FUNCTION RANF()
RANF=RAND(0)
RETURN
END
```

Bibliography

[1] Brush S. G., (1967), Rev. Mod. Phys. Vol. 39 No. 4, 883.

[2] Metropolis N., Rosenbluth M., Rosenbluth, M., Teller A., and Teller E., (1953), J. Chem. Phys. 21, 1087.

[3] Domb, C., (1974b), in Phase Trans. and Critical Phenomena, edited by C. Domb and M.S. Green (Academic Press, London), Vol. 3, p. 1.

[4] Onsager L., (1944), Phys. Rev. 65, 117.

[5] Kihara T., Midzuno Y., Shizume T., (1954), J. Phys. Soc. Japan 9, No 5, 682.

[6] Kramers H. A., and Wannier G. H., (1941), Phys. Rev. 60, 252 and 263.

[7] Oguchi T., (1950), J. Phys. Soc. Japan 5, 75.

[8] Frempong-Mireku P., and Moriarty K.J.M., (1994), Proceed. Maple Summer Workshop and Symposium, p. 97.

[9] Drouffe J. M., and Moriarty K. J. M., (1991), Comp. Phys. Comm. 64, 207.

[10] Fisher, M. E., (1960), Rep. Prog. Phys. 30, 615.

[11] Yang, C. N., and Lee, T. D., (1952), Phys. Rev 87, 404.

[12] Wisdom, B., (1965), J. Chem. Phys. 43, 3898.

[13] Stanley H. E., (1971), Introduction to Phase Trans. and Critical Phenomena (Oxford, Claredon Press).

[14] Mouritsen O.G., (1984), Computer Studies of Phase Trans. and Critical Phenomena (Springer, Verlag NY, Tokyo).

[15] Binder K. and Kalos M. H., (1979), in Monte Carlo Meth. in Stats. Phys. I, edited by K. Binder, (Springer-Verlag, Heidelberg), p.225.

[16] Rushbrooke G. A., Baker Jr. and Wood P. J., (1974), in Phase Trans. and Phenomena, edited by C. Domb and M. S. Green, (Academic Press, London), Vol.3, ch.5

[17] Barber M. N., (1984), in Phase trans. and critical Phenomena, edited by C. Domb and J. L. Lebowitz, (Academic Press, London), Vol. 8.

[18] Barber M.N., (1980), Phys. Rep. 59, 376.

[19] Gould H.and Tobachnik J., (1983), An Introduction to Computer Simulation Methods, (Addison-Wesley Publishing Company), Vol 2.

[20] Goldman A. J., Tucker A. W., (1956), Polyhedral convex cones. in Linear inequalities and related systems, edited by H. W. Kuhn and A. W. Tucker, (Princeton University Press, Princeton, N.J), p. 19ff.

[21] Lassettre E. N., and Howe J. P., (1941), J. Chem. Phys. Rev. 9, 747 and 801.

[22] Montroll E. W., and Mayer J. E., (1941), J. Chem. Phys. Rev. 9, 626.

[23] Askin J., and Lamb Jr., (1943), Phys. Rev. 64, 159.

[24] Montroll E. W., (1941), J. Chem. Phys. Rev. 9, 706.

[25] Wakefield A. J., (1950), J. Chem. Phys. Rev. 9,419.

[26] Potts, R. B., (1952), J. Phys. C 11, L337.

[27] Baxter, R. J., (1973), J. Phys. C 6, L445.

[28] Domb C., (1960), Adv. Phys. 9, 149.

[29] Domb C., (1974), J. Phys. A. 7, 1335.

[30] Domb C., and Pearse C. J., (1976), J. Phys. A 9, L137.

[31] Domb C., (1960), Adv. Phys. 9, 149.

[32] Ashkin, J., and Teller E., (1943), Phys. Rev. 64, 178.

[33] Betts D. D., (1974), in Phase Trans. and Critical Phenomena, Vol. 3, Domb Eds.C. and Green M.S., London, Academic Press ch.8.

[34] Betts, D. D., (1964), Can. J. Phys. 42, 1564.

[35] Kasteleyn P. W., (1964), Report in Aachen Conference on Statistical Mechanics.

[36] Kasteleyn P. W. and Boel R. J., (1978), Commum. Math. Phys. 61, 191.

[37] Potts R. B. and Ward J. C., (1955), Prog. Theor. Phys. 13, 38.

[38] Kunz, H. and Wu F. Y., (1978), J. Phys. C 11, L1.

[39] Fontaine J. R. and Gruber Ch., (1990), Commum. Math. Phys. 70, 243-269.

[40] Stanley H. E., (1974), in Phase Trans. and Critical Phenomena, edited by C. Domb and M. S. Green, (Academic Press, London) Vol. 3, ch.7

[41] Stanley R. P., (1973), Disc. Maths. 5, 171.

[42] Biggs N. L., (1975), J. Phys. A 8, L110.

[43] McCoy B. M. and Wu T.T. (1973), 2-d Ising Model (Harvard University Press, Cambridge, Mass).

[44] Bhanot G., Creutz M., and Neuberger H., (1984), Nucl. Phys. B235[FS11] 417.

[45] Creutz M., and Moriarty K.J.M., (1984), Comput. Phys. Commun. 33, 361.

[46] Drouffe M., and Moriarty K.J.M., (1990), Int. J. High Speed Comput. 2(2), 133.

[47] Brower R. C., Moriarty K.J.M., Myers E., Orland P., and Tamayo P., (1988), Phys. Rev. B. Vol 38, No. 16.

[48] Creutz M., (1983), Phys. Rev. Lett. 50, 1441.

[49] Glosli, J., Plischke M., (1983), Can. J. Phys. 61, 1515.

[50] Creutz M., Jacobs L., and Rebbi C., (1979), Phys. Rev. D 20, 1915.

[51] Baxter R. J., M.F. Sykes, and M. G. Watts, (1975), J. Phys. A 8, 245.

[52] Ferdinard K. A., and Fisher M. E., (1969), Phys. Rev. 185, 832.

[53] Fisher M. E., (1971), In Proc. Int. School of Phys. (Enrico Fermi), edited by M. S. Green, (Academic Press, New York), Course LI p.1.

[54] Landau D. P., (1976a), Phys. Rev. B 13, 2997.

[55] Landau D. P., (1976b), Phys. Rev. B 14, 255.

[56] Griffiths, R. B., (1972), Rigorous Results and Theorems. In: Phase trans. and critical phenomena, edited by C. Domb and M. S. Green, (Academic Press, London), Vol. 1. p. 72ff.

[57] Yalabik M. C., and Gunton J. D., (1982), Phys. Rev. B 25, 534.

[58] Fisher M. E., and Burford R. J., (1967), Phys. Rev. 156, 583.

[59] Chakrabarti C. K., Baumgaertel H. G., and Stauffer D. Z, (1981), Phys. Rev. B 44, 333.

[60] Jan L. L., Moseley, and Stauffer D., (1983), J, Stat. Phys. 33, 1.

[61] Kalle J., (1984), J. Phys. A. 17, L801.