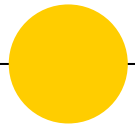
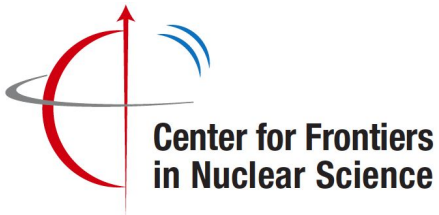
Introduction to lattice quantum chromodynamics

CFNS Summer School 2022 on the physics of the electron ion collider



Chris Monahan

William & Mary



Lecture outline

Lecture 1

- Path integrals and lattice field theory
- Quantum chromodynamics

Lecture 2

- Spin models
- Numerical integration
- Monte Carlo methods

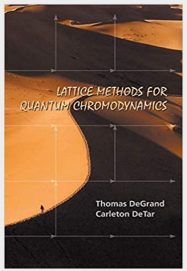
Lecture 3

- Gauge field Monte Carlo
- Correlation functions

Summary: lecture 1

- Lattice field theory provides rigorous definition of nonperturbative QFT
- QCD describes fundamental interactions of the strong nuclear force in terms of quarks and gluons
- Lattice QCD formulated with quarks on the lattice sites and *SU(3)* link variables representing gluons
- Gluon action built from closed loops of link variables, such as plaquettes
- Fermions on the lattice suffer from fermion doubling; intimately connected to existence of chiral fermions in a regulated theory
- Different discretisations possible, but any choice should lead to QCD in the continuum limit of vanishing lattice spacing

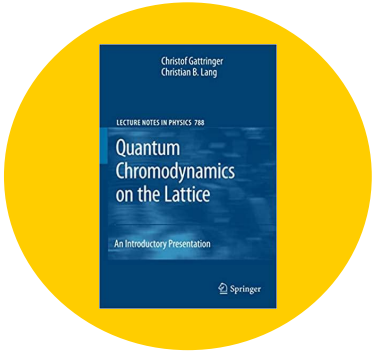
Some references



DeGrand and DeTar

Lattice methods for quantum chromodynamics

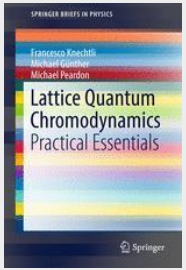
World Scientific 2006



Gattringer and Lang

Quantum chromodynamics on the lattice

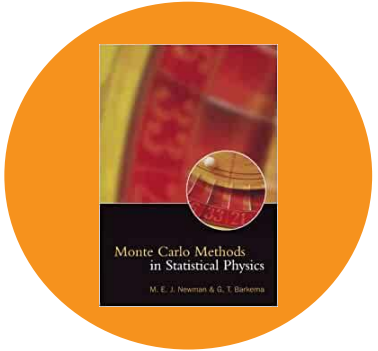
Springer 2010



Knechtli, Günther & Peardon

Lattice quantum chromodynamics: practical essentials

Springer 2017



Newman and Barkema

Monte Carlo methods for statistical physics

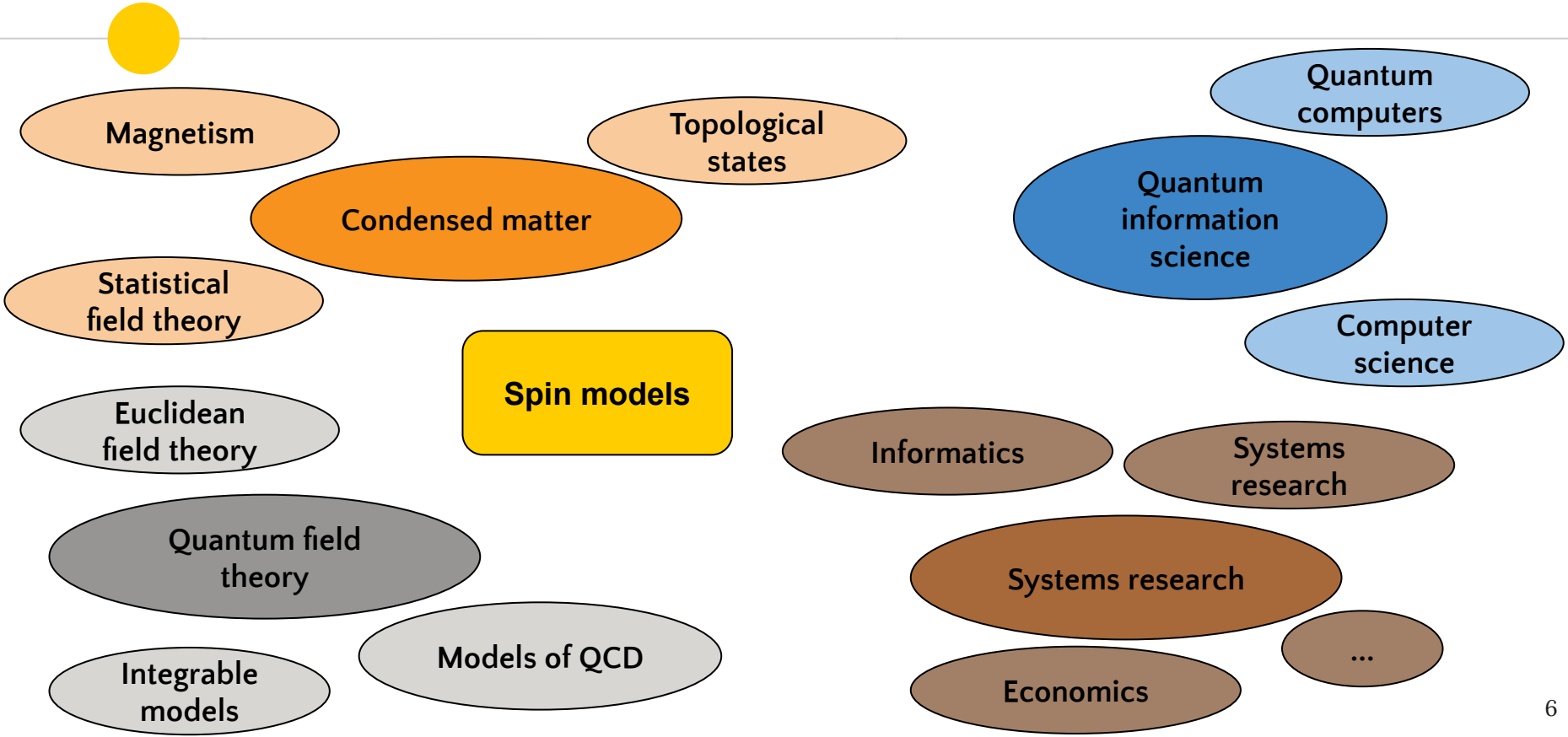
Clarendon 1999

2.1

Spin models

From magnets to Monte Carlo

Spin models everywhere



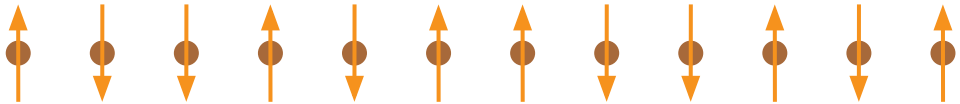
ga

Spin models

1D: solved by Ising in his Ph.D. thesis (1924)
2D: solved by Onsager in 1944
3D: remains analytically unsolved

Example: Ising model

Lattice of coupled spins that can point up or down



System described by a Hamiltonian

$$H(\sigma) = -J \sum_{\langle i\, j \rangle} \sigma_i \sigma_j$$

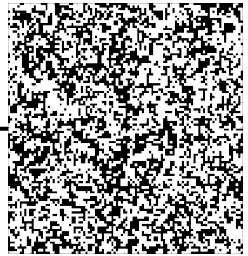
Probability of finding a given configuration of spins

$$P_\beta(\sigma) = \frac{e^{-\beta H(\sigma)}}{Z_\beta}$$

the ***Boltzmann weight factor***

Simplest model to have a *phase transition* in 2D

$M \approx 0$



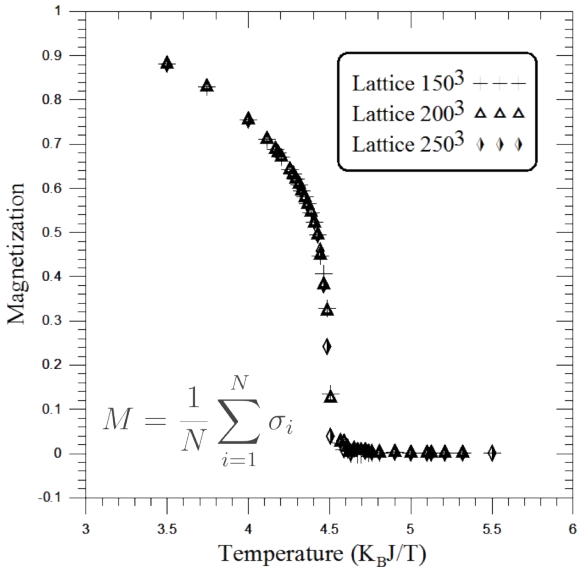
$M \approx -1$

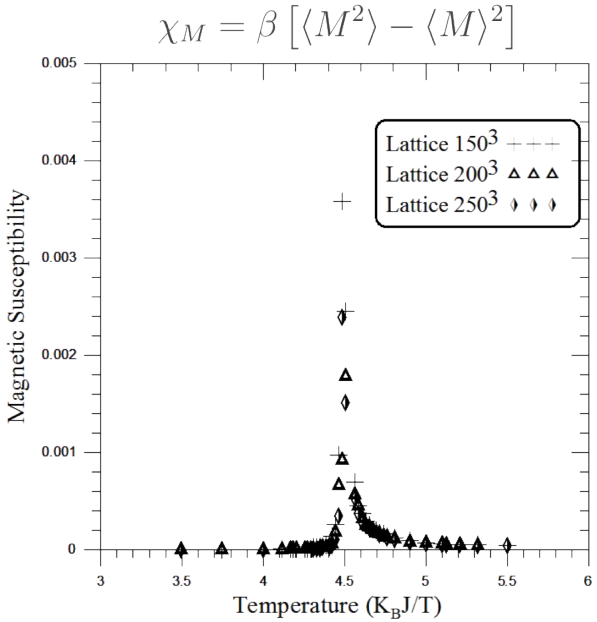
Monte Carlo investigation of the Ising model, Fricke, 2006

Spin models

Simplest model to have a *phase transition* in 2D

Example: Ising model





Sonsin et al., J. of Phys. (CS), 630 (2015) 012507

Spin models

Example: Ising model

"Solving" this system means $Z_\beta = \sum_\sigma e^{-\beta H(\sigma)}$

- partition function
- "observables" $\langle f_\beta \rangle = \sum_\sigma f(\sigma) P_\beta(\sigma)$
- correspond to "n-point" functions (or correlation functions or Green's functions or …)

$$\langle \sigma_{i_1} \sigma_{i_2} \cdots \sigma_{i_n} \rangle$$

For example, in one dimension

$$\langle \sigma_i \sigma_j \rangle \sim \exp\left[-k_\beta |i-j|\right]$$

the "propagator" of the Ising model

Spin models and path integrals

Calculating correlation functions in spin models

$$\langle \sigma_i \sigma_j \rangle = \frac{1}{Z_\beta} \sum_{\{\sigma\}} \sigma_i \sigma_j e^{-\beta H(\sigma)}$$

looks a lot like calculating path integrals

$$\langle 0 | \hat{\phi}(\mathbf{x}_2, t_2) \hat{\phi}(\mathbf{x}_1, t_1) | 0 \rangle = \int \mathcal{D}\phi(x)\, \phi(\mathbf{x}_2, \tau_2) \phi(\mathbf{x}_1, \tau_1)\, e^{-\mathcal{S}_\mathrm{E}[\phi(x)]}$$

Example: Ising model

Calculating the magnetisation

$$M = \frac{1}{N} \sum_{i=1}^{N} \sigma_i$$

$$\langle M \rangle = \frac{1}{N Z_\beta} \sum_{\{\sigma\}} \Big[\sum_{i=1}^{N} \sigma_i \Big] e^{-\beta H(\sigma)}$$

$$\chi_M = \beta \left[\langle M^2 \rangle - \langle M \rangle^2 \right]$$

$M \approx 0$

Statistical (spin) system in d spatial dimensions corresponds to a quantum system in d spacetime dimensions

Outcome of this: we've reduced understanding QFT/spin models to numerical integration

Spin models and QCD?

Example: Nonlinear sigma model

See e.g. https://arxiv.org/abs/2111.11942

Generalises Ising model to spins that can point in arbitrary directions

- simplest example of spin model with internal symmetry

Two-dimensional *O(3)* model is a toy model for QCD: exhibits **mass gap** and **asymptotic freedom**

$$S_E = \frac{\beta}{2} \int d^2x \sum_{i=1}^{2} (\partial_i \vec{e})^2$$

Admits a topological **theta term** and provides a simple model of **topological theories**

Spin models and QCD?

Proof of existence of a **mass gap** in pure Yang-Mills theory (i.e. just a bunch of gluons) is worth $\$10^6$ (1st Clay Millenium problem).

Example: Nonlinear sigma model

See e.g. https://arxiv.org/abs/2111.11942

Generalises Ising model to spins that can point in arbitrary directions

- simplest example of spin model with internal symmetry

Two-dimensional *O(3)* model is a toy model for QCD: exhibits **mass gap** and **asymptotic freedom**

$$S_E = \frac{\beta}{2} \int d^2x \sum_{i=1}^{2} (\partial_i \vec{e})^2$$

Admits a topological **theta term** and provides a simple model of **topological theories**

The **strong CP problem** is one of the most significant open questions in QCD - why is the coefficient of the **theta term** so tiny (current experimental bounds $< 10^{-10}$)?

Closely related to another deep problem - where is all the CP violation? Put another way, why is there anything rather than nothing?

Brief interlude: neutron electric dipole moment

Required for the existence of matter following the big bang, AKA why is there something and not nothing (in the physicist's sense)?

Three Sakharov conditions for **baryon asymmetry** require **CP violation**

Standard model sources insufficient

- Phases of Cabibbo-Kobayashi-Maskawa matrix
- QCD theta term

Permanent electric dipole moment of the neutron

- evidence of CP violation

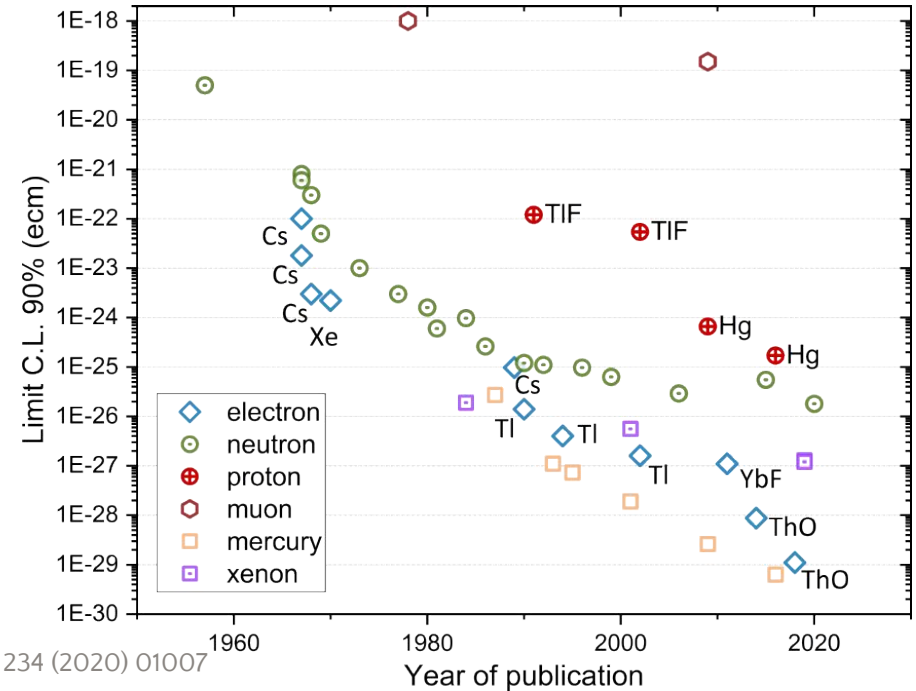
$$d_n^{\mathrm{exp}} = (0.0 \pm 1.1_{\mathrm{stat}} \pm 0.2_{\mathrm{sys}}) \times 10^{-26}\,\mathrm{e\,cm}$$

Abel *et al.*, PRL 124 (2020) 081803

Interpreted as a bound on QCD theta term

- strong CP problem

$$d_n^{\mathrm{QCD}} \sim 10^{-16}\theta\,\mathrm{e\,cm}$$



Kirch & Schmidt-Wellenburg, EPJ 234 (2020) 01007

Summary: lecture 2.1

- Spin models are ubiquitous
  - Condensed matter systems
  - Toy models for quantum field theories, coupled systems, social networks ...
- Ising model
  - coupled chain of spins that point up or down
  - simplest model with phase transition in 2D
- Nonlinear sigma model
  - coupled array of spins that can point in any direction
  - toy model of QCD, with a mass gap, asymptotic freedom and topological effects

2.2 Numerical methods

From pi to the proton

Path integrals for lattice QCD

Correlation functions calculated via the path integral representation on a Euclidean spacetime lattice

Unfortunately, the resulting integrals are not easily solved

In scalar field theory, a 16x16x16x16 lattice gives a 65536-dim integral

In QCD, things are much worse!

A typical lattice is 128x64x64x64, each gluon field is a linear combination of 8 3x3 matrices, and each flavor of quark has 3x4 degrees of freedom - 10^9-dim integral

Integrals must be computed numerically!

The challenge

Numerically determine π

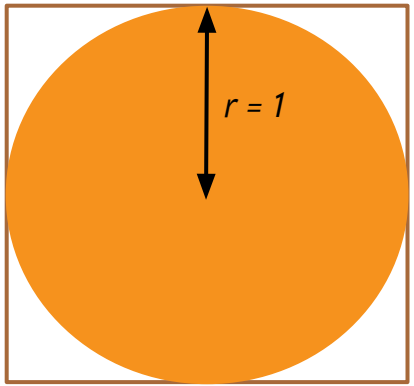
But without using a computer algebra system (like Mathematica)

Determining π

Numerically determine π

Numerical approach - use the ratio of two areas:

1. circle of radius equal to one
2. square of side length equal to two



This ratio is equal to π/4!

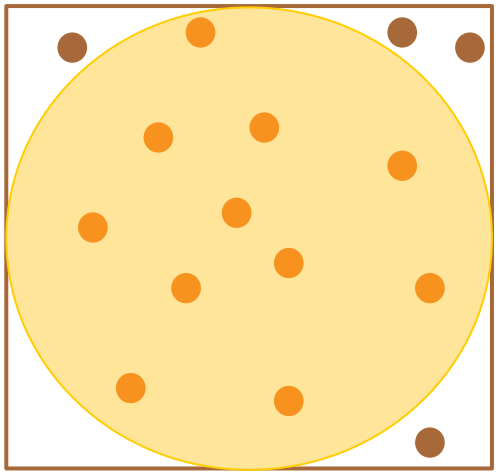
OK, so...

How do we numerically determine the area of a circle of unit radius?

How do we numerically determine the area of a circle of unit radius?

Estimating areas stochastically

Imagine throwing darts at a dart board, surrounded by a square of unit side



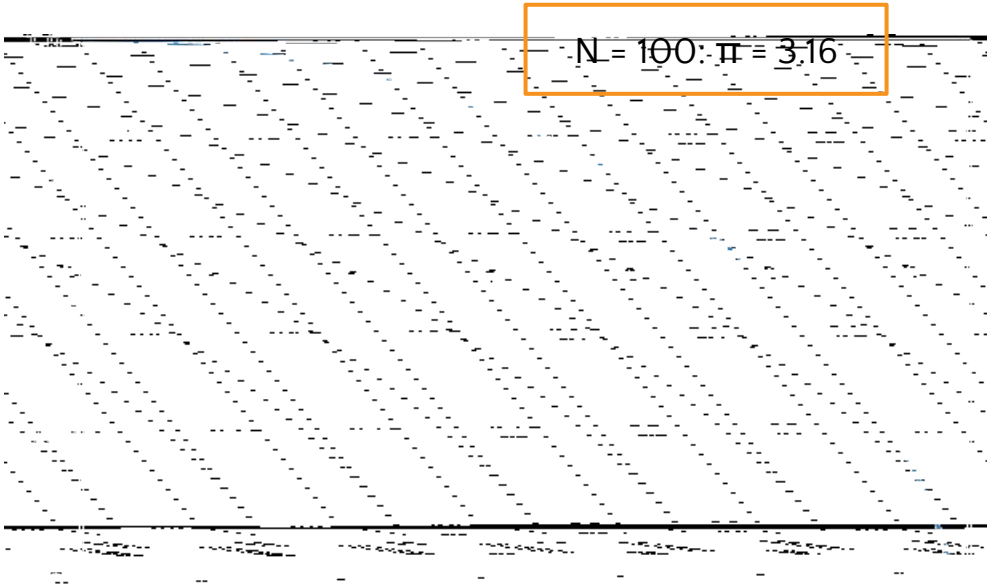
Ratio of “hits” to total “throws” is an estimate of the ratio of areas

How do we numerically determine the area of a circle of unit radius?

Estimating areas stochastically

Computationally, this corresponds to:

1. choosing two random real numbers (a,b) between 0 and 1
2. calculating the “distance” from the origin (0,0) and (a,b)
3. points with distance is less than one fall within our circle, others do not
4. calculate the ratio of the number of points with distance less than one to the total number of points



Sampling nonuniform distributions

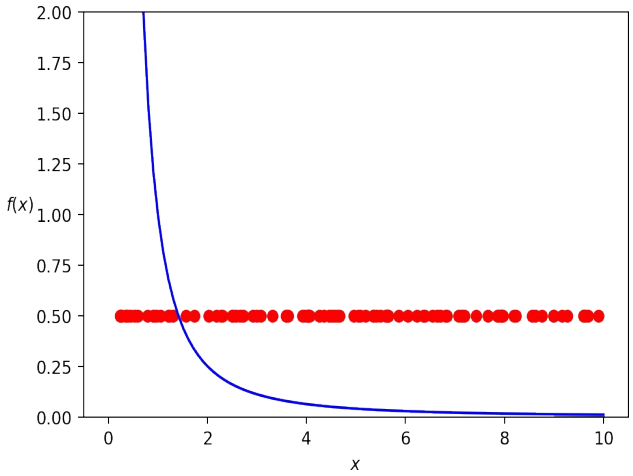
Numerically determining π is relatively straightforward

Sampling a uniform distribution

In essence, we are calculating the integral of a flat integrand

Path integrals are far from flat!

Example: peaked integrand



$$f(x) = \frac{1}{x^2 + \epsilon^2}$$

$$\int_0^\infty f(x) = \frac{\pi}{2\epsilon}$$

Sampling nonuniform distributions

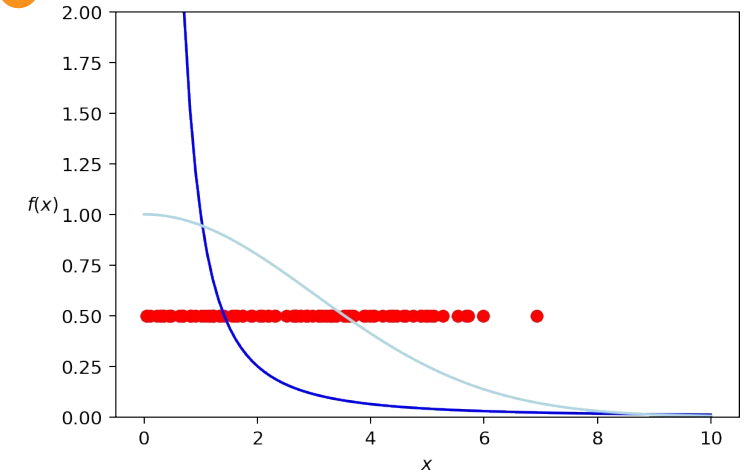
Numerically determining π is relatively straightforward

In essence, we are calculating the integral of a flat integrand

Path integrals are far from flat!

Could use a Gaussian (normal) distribution

Example: peaked integrand



$$f(x) = \frac{1}{x^2 + \epsilon^2}$$

$$\int_0^\infty f(x) = \frac{\pi}{2\epsilon}$$

Sampling nonuniform distributions

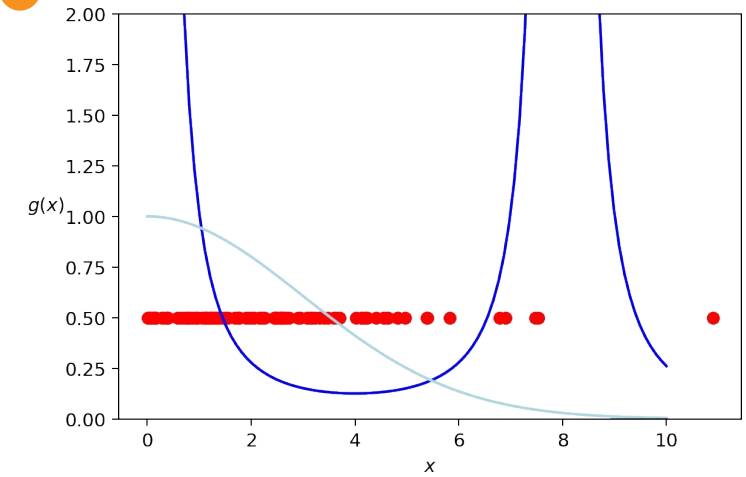
Numerically determining π is relatively straightforward

In essence, we are calculating the integral of a flat integrand

Path integrals are far from flat!

Could use a Gaussian (I guess)?

Example: doubly-peaked integrand



$$g(x) = \frac{1}{x^2 + \epsilon^2} + \frac{1}{(x-8)^2 + \epsilon^2}$$

$$\int_0^\infty g(x) = \frac{1}{\epsilon}\left[\pi + \arctan\left(\frac{8}{\epsilon}\right)\right]$$

Sampling nonuniform distributions

Much better to use **adaptive Monte Carlo**, which can adapt to the shape of the integrand, without prior assumptions.

`VEGAS` is one well-known algorithm (by G.P. Lepage) for adaptive numerical integration. Works well for integrals with less than about ten dimensions.

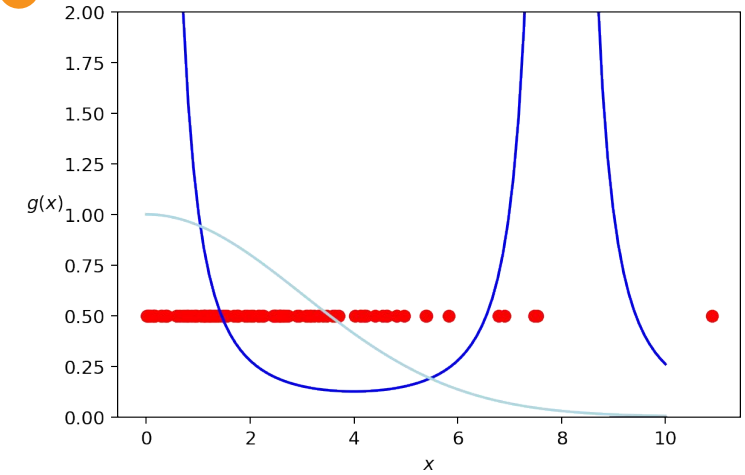
Numerically determining π is relatively straightforward

In essence, we are calculating the integral of a flat integrand

Path integrals are far from flat!

Could use a Gaussian (I guess)?

Example: doubly-peaked integrand



$$g(x) = \frac{1}{x^2+\epsilon^2} + \frac{1}{(x-8)^2+\epsilon^2}$$

$$\int_0^\infty g(x) = \frac{1}{\epsilon}\left[\pi + \arctan\left(\frac{8}{\epsilon}\right)\right]$$

Summary: lecture 2.2

- Random sampling works well for uniform integrands
- Peaked integrands require adaptive Monte Carlo (e.g. `VEGAS`)
- Adaptive Monte Carlo works well for integrals with up to approximately ten dimensions
- Correlation functions in quantum field theories are *many* orders of magnitude more complicated adaptive Monte Carlo can handle

2.3 Monte Carlo methods

From magnets to Markov chains

Monte Carlo integration

Recall: we want to calculate the Euclidean path integral

$$\langle 0|\hat{\phi}(\mathbf{x}_2, t_2)\hat{\phi}(\mathbf{x}_1, t_1)|0\rangle = \int \mathcal{D}\phi(x)\, \phi(\mathbf{x}_2, \tau_2)\phi(\mathbf{x}_1, \tau_1)\, e^{-\mathcal{S}_\mathrm{E}[\phi(x)]}$$

Steps:

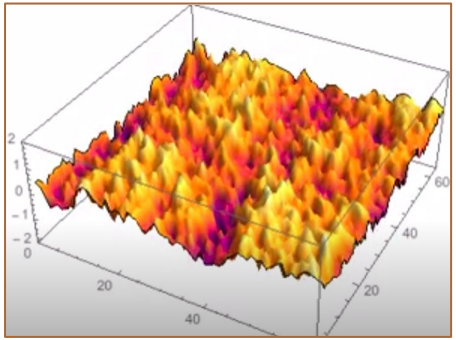
1. Discretize spacetime
2. Generate **configurations** according to probability

   $$\mathcal{P}[\phi(x)] \propto e^{-\mathcal{S}_\mathrm{E}[\phi(x)]}$$

3. **Unweighted average** over these configurations

   $$\langle 0|\hat{\phi}_{x_2}\hat{\phi}_{x_1}|0\rangle \approx \frac{1}{N_\mathrm{conf}} \sum_{m=1}^{N_\mathrm{conf}} \phi_{x_2}\phi_{x_1}\Big|_{\{\phi\}_m}$$

   is our Monte Carlo estimate of this path integral



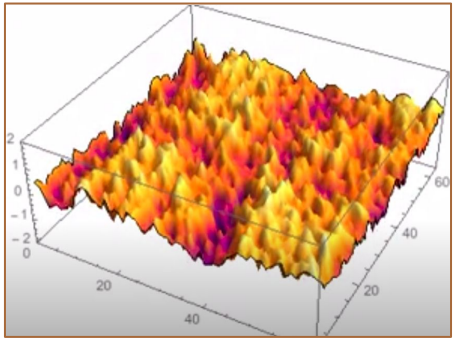
Generating probability distributions

Started by reducing calculations of correlation functions to numerical integration

Now reduced that to determining probability distributions

For that, we use **Markov chain Monte Carlo** to **update** configurations by

1. Start from a specific configuration, say,
   $$\{\sigma_x\}_0 = \{\sigma_{11}, \sigma_{21}, \cdots, \sigma_{NN}\} = \{1, 1, \cdots, 1\}$$
2. Visit each site on the lattice
3. Randomly modify spin at that site according to a specific procedure



Reasonable algorithms must satisfy:

1. **Ergodicity**
2. **Detailed balance**
   $$P_{ba} = P_{ab} \exp\left[-\beta \left(E_b - E_a\right)\right]$$

Clearly the trick lies in how we achieve this *update step*

Markov chains

Definition: Stochastic process

Set of random variables

$$\{\eta_i^{(n)}\} \qquad i, n \in \{0, 1, \cdots\}$$

with associated probability of a specific **state**

$$P(s^{(n)}) = P(\{\eta_i^{(n)}\} = s^{(n)})$$

Example: the 1D two-spin Ising model

$$\eta_i^{(n)} = \uparrow, \downarrow \qquad s^{(n)} = \begin{cases} \{\uparrow, \uparrow\} \\ \{\uparrow, \downarrow\} \\ \{\downarrow, \uparrow\} \\ \{\downarrow, \downarrow\} \end{cases}$$

Definition: Markov chain

Stochastic process that satisfies the **Markov property**

$$P\left(s^{(n)} \middle| s^{(n-1)}, \cdots, s^{(0)}\right) = P\left(s^{(n)} \middle| s^{(n-1)}\right)$$

Markov chains are "forgetful"

Example: the 1D two-spin Ising model

$$P\left(\{\uparrow\downarrow\} \middle| \{\{\downarrow\downarrow\}, \{\downarrow, \uparrow\}, \{\uparrow, \uparrow\}\}\right) = P\left(\{\uparrow\downarrow\} \middle| \{\downarrow\downarrow\}\right)$$

Markov chain properties

Introduce **transition probability**

$$P_{ba} = P\left(s^{(n+1)} = b \middle| s^{(n)} = a\right)$$

then a Markov chain is **time homogeneous** if

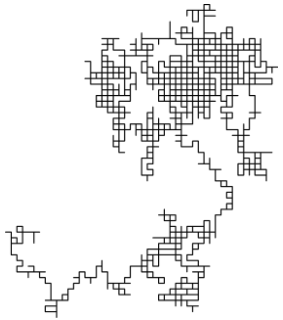
$$P\left(s^{(n+1)} = b \middle| s^{(n)} = a\right) = P\left(s^{(n)} = b \middle| s^{(n-1)} = a\right)$$

Time homogeneous Markov chains are specified entirely by

- initial state
- transition probability

Example: random walk

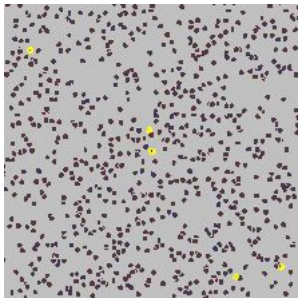
Markov chains are discrete time processes



Wikimedia

Example: Brownian motion

Markov processes are continuous time processes



Wikimedia

Metropolis algorithm

1. Generate a uniformly distributed random number
   $\xi \in (-\epsilon, \epsilon)$
2. Modify the value at a specific lattice site
   $\phi_{x_i} \to \phi'_{x_i} = \phi_{x_i} + \xi$
3. Calculate the change in the action generated by this modification
4. If
   $\Delta S < 0$
   accept the change and move on to the next site [return to 1.]
5. Else if
   $\Delta S > 0$
   generate a new random number
   $\eta \in (0, 1)$
6. If
   $\exp\left[-\Delta S\right] > \eta$
   accept the change, otherwise reject the change
7. Move on to the next site [return to 1.]

Metropolis algorithm

intermediate updates discarded to avoid *autocorrelations*

Comments:

1. paths should be statistically independent
2. size of ε is critical
   a. ε should be tuned so about 50% of proposed changes are accepted
3. results should not depend on **initial configurations**

too small and the configurations are too similar
too large and the changes are always rejected

Example: "cold start"

$$\{\phi_x\}_0 = \{\phi_{11}, \phi_{21}, \cdots, \phi_{NN}\} = \{0, 0, \cdots, 0\}$$

Example: "hot start"

$$\{\phi_x\}_0 = \{0.1347, 0.5678, \cdots, 0.9973\}$$

configurations require *thermalization*: discard the first *X* configurations generated

Metropolis algorithm

Example: 2D Ising model

1. Flip the spin at a site
2. Determine the resulting change in the energy
3. If
   $\Delta E < 0$
   then accept the spin flip
4. If
   $\Delta E > 0$
   then generate a random number
   $\eta \in (0, 1)$
   and accept the spin flip if
   $\exp\left[-\Delta E\right] > \eta$
5. Go to the next site and return to 1.

Metropolis algorithm

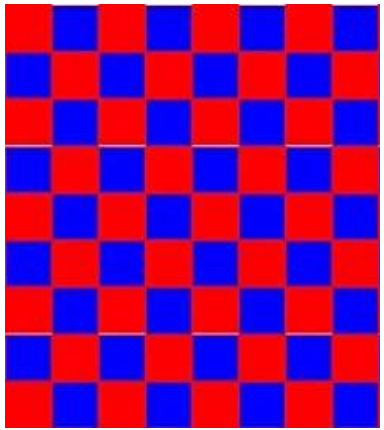
Metropolis algorithm is strictly **local**

Ergodicity means that all possible states can be reached

- **ergodic** only with an infinite number of update steps
- spin configurations can become stuck in local minima (of the action)
- configurations have **autocorrelations**

Shake things up with **global update algorithms**

Global update algorithms



Example: efficient checkerboard

Color each spin **red** or **blue**, alternating colors

Then the change in action for each **blue** spin is dictated by four **red** neighbors

Steps:

1. Fix all **red** spins
2. Apply the **Metropolis** algorithm to each **blue** spin in parallel
3. Fix all **blue** spins
4. Apply the **Metropolis** algorithm to each **red** spin in parallel
5. Return to 2. and repeat

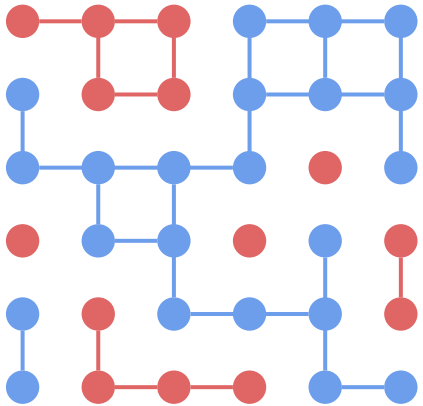
Cluster algorithms

Cluster algorithms are a family of global update algorithms based on the **Fortuin-Kasteleyn decomposition**

$$Z_\beta = \sum_\sigma e^{-\beta E} = \sum_{\text{bonds}} (1 - e^{-\beta J}) e^{-\beta J n} 2^{N_c}$$

maps spins to random cluster models

Construct **clusters** of spins by joining neighbouring identical spins with **bonds**



Update entire clusters at once

Global updates reduce move system away from local minima

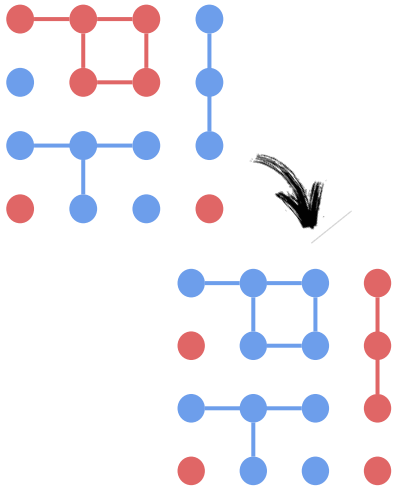
- improve ergodicity
- reduce autocorrelations

Cluster algorithms

Cluster algorithms are a family of global update algorithms based on the **Fortuin–Kasteleyn decomposition**

Example: Swendsen–Wang algorithm

1. Start from a given configuration
2. For each pair of neighboring identical spins:
   - pick a random number $\eta \in (0, 1)$
   - and form a bond if $\exp\left[-2\beta J\right] > \eta$
3. For each cluster, flip spins of entire cluster with probability ½
4. Erase all the bonds and return to 1.



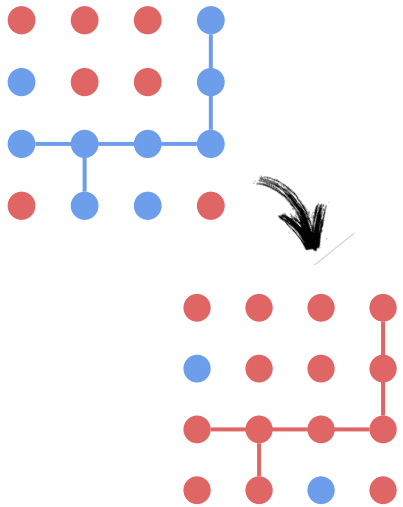
Swendsen & Wang, PRL 58 (1987) 86

Cluster algorithms

Cluster algorithms are a family of global update algorithms based on the **Fortuin–Kasteleyn decomposition**

Example: Wolff algorithm

1. Pick a random site *i*
2. For each neighboring site *j*:
   - pick a random number $\eta \in (0, 1)$
   - and form a bond if $\exp[-2\beta J] > \eta$
3. For each neighboring site with a bond, return to step 2.
4. Repeat 2. and 3. until no further bonds are formed
5. Flip the spins in the cluster
6. Erase all the bonds and return to 1.



Wolff, PRL 62 (1989) 361

Cluster algorithms

Cluster algorithms are rejection free!

Algorithms can be extended to other spin models
- for example, for the O(3) model, project spins along one axis and flip those projections

Work very well near **phase transitions**
- correlation lengths diverge, so large clusters can form

exactly where local update algorithms perform poorly

Do not work well (or at all)
- far from phase transitions
- in frustrated spin systems
- if you can't construct a symmetric reflection operation
- in the presence of strong external fields

Unfortunately more sophisticated techniques are needed for gauge theories, like QCD

Summary: lecture 2.3

- Numerical evaluation of high-dimensional integrals requires **Markov chain Monte Carlo**
- Markov chain Monte Carlo consists of two basic steps
  - propose local update
  - accept/reject update
- Local-update algorithms suffer large **autocorrelations** - introduce **global-update** algorithms
- Global-update algorithms work well near phase transitions

Looking ahead: lecture 3

- Monte Carlo for gauge theories – generalising our algorithms from spin models
- Examples of algorithms for gauge field generation
  - Metropolis, heat bath, hybrid Monte Carlo
- Constructing bound states on the lattice
- Extracting physics from lattice calculations

Thank you

***Questions*?**

You can find me at

- cjmonahan@wm.edu
- www.cjmonahan.net

Adaptive Monte Carlo

This is the classic **importance sampling** method of the original `VEGAS`. The modern incarnation also implements adaptive stratified sampling.

Illustrative example: the `VEGAS` algorithm

1. Transform variables to a new variable that can be sampled uniformly

$$I = \int_a^b \mathrm{d}x\, f(x) \longrightarrow I = \int_0^1 \mathrm{d}y\, J(y) f(x(y))$$

where the Jacobian is

$$J(y(x)) = \frac{1}{|f(x)|} \int_a^b \mathrm{d}x\, |f(x)|$$

2. Determine the new integral through Monte Carlo integration

$$I \approx I_{\mathrm{MC}} = \frac{1}{N} \sum_{y=1}^{N} J(y) f(x(y))$$

I_{MC} a random variable drawn from a distribution with mean = I.
The Jacobian is chosen to minimize the standard deviation of this distribution, which is Gaussian in the limit of large N.

Adaptive Monte Carlo

Much more detail at
https://vegas.readthedocs.io/en/latest/

VEGAS carries out this variable transformation through an iterative grid

1. Divide *x* into a grid with *n* points

$$\begin{aligned} x_0 &= a \\ x_1 &= x_0 + \delta x_0 \\ &\vdots \\ x_n &= x_{n-1} + \delta x_{n-1} = b \end{aligned}$$

2. Specify the transformation function

$$x(y) = x_i \qquad y = \frac{i}{n} \qquad i = \{0, \cdots, n\}$$

and use linear interpolation between those points

3. Corresponding Jacobian is

$$J(y) = J_i = n\delta x_i \qquad \frac{i}{n} < y < \frac{i+1}{n}$$

4. Standard deviation minimized for

$$n^2 \delta x_i \int_{x_i}^{x_{i+1}} \mathrm{d}x \, f^2(x) = \text{constant}$$

Number of iterations of the VEGAS algorithm

Result of each iteration

itn	integral	wgt average	chi2/dof	Q
1	946(118)	946(118)	0.00	1.00
2	1572.7(2.7)	1572.4(2.7)	28.36	0.00
3	1570.7(1.2)	1571.0(1.1)	14.34	0.00
4	1571.03(59)	1571.03(52)	9.56	0.00
5	1570.89(25)	1570.92(23)	7.18	0.00
6	1570.65(12)	1570.71(11)	5.96	0.00
7	1570.574(65)	1570.611(55)	5.17	0.00
8	1570.496(44)	1570.540(34)	4.81	0.00
9	1570.632(35)	1570.585(25)	4.65	0.00
10	1570.622(32)	1570.599(19)	4.23	0.00
11	1570.601(30)	1570.599(16)	3.80	0.00
12	1570.610(29)	1570.602(14)	3.47	0.00
13	1570.586(28)	1570.598(13)	3.20	0.00
14	1570.611(30)	1570.600(12)	2.97	0.00
15	1570.546(29)	1570.593(11)	2.97	0.00
16	1570.581(32)	1570.592(10)	2.78	0.00
17	1570.597(32)	1570.5920(98)	2.61	0.00
18	1570.554(31)	1570.5886(93)	2.53	0.00
19	1570.563(34)	1570.5868(90)	2.42	0.00
20	1570.530(34)	1570.5831(87)	2.43	0.00

1570.7963267948965

Each iteration uses an initial grid based on the previous iteration.

Each iteration is statistically independent: set of Gaussian numbers (neval large enough) whose mean is the exact value of the integral.

Small Q-values indicate a larger number of evaluations needed.

Weighted average of all iterations so far

$$\chi^2 = \sum_{i=1}^{N_{\text{it}}} \frac{(I_i - \overline{I})^2}{\sigma_i^2}$$

Q-value = probability that a larger chi^2/dof could have occurred by chance