Introducción a las simulaciones Monte Carlo a través de modelos de espín clásico

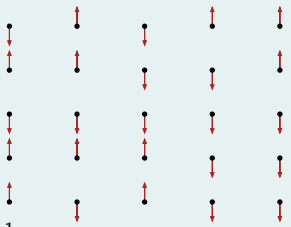
Jaime Fabián Nieto Castellanos
Instituto de Ciencias Nucleares

8 de diciembre de 2022

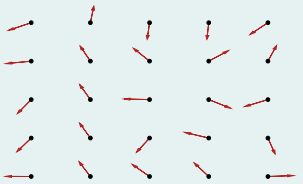
Modelos de espín clásico

- Modelos en donde el espín se modela como un vector.
- Sirven como modelos de juguete para estudiar transiciones de fase, rompimientos de simetría, métodos numéricos, etc.
- El **modelo de Ising** sin campo externo está definido por

$$H = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j, \quad \sigma_i = \pm 1$$



- El **modelo XY** está definido por



$$H = -J \sum_{\langle ij \rangle} \vec{\sigma}_i \cdot \vec{\sigma}_j = -J \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j), \quad \vec{\sigma}_i \in \mathbb{S}^1$$

Funciones de partición

- Para el modelo de Ising

$$Z_I = \sum_{[\sigma_i]} e^{-\beta H}, \quad \beta \equiv \frac{1}{T}.$$

- Para el modelo XY

$$\begin{aligned} Z_{XY} &= \int_0^{2\pi} \prod_i \frac{d\theta_i}{2\pi} \exp\left\{ \beta \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j) \right\} \\ &= \int_0^{2\pi} \prod_i \frac{d\theta_i}{2\pi} \prod_{\langle ij \rangle} \exp\{\beta \cos(\theta_i - \theta_j)\}. \end{aligned}$$

- Los valores de expectación se obtienen a través de

$$\langle O \rangle = \frac{1}{Z} \sum_{[\sigma]} e^{-\beta H[\sigma]} O[\sigma].$$

- Generar numéricamente configuraciones distribuidas de acuerdo a

$$p[\sigma] = \frac{1}{Z} e^{-\beta H[\sigma]},$$

para integrar *estocásticamente*.

- Evaluar valores de expectación para $N \gg 1$ configuraciones,

$$\langle O \rangle \approx \frac{1}{N} \sum_{[\sigma]} O[\sigma].$$

- Los métodos Monte Carlo permiten generar configuraciones aleatorias distribuidas de acuerdo a $p[\sigma]$.

Simulación Monte Carlo: conceptos importantes

- Cadena de Markov: cadena de configuraciones generadas estocásticamente

$$[\sigma_1] \to [\sigma_2] \to \cdots \to [\sigma_n] \to [\sigma_{n+1}] \to \cdots$$

Solo se necesita de $[\sigma_n]$ para generar $[\sigma_{n+1}]$. Para lograr esto se requiere de una probabilidad de transición independiente de n

$$T(\sigma' = \sigma_{n+1}|\sigma_n) = T(\sigma'|\sigma), \quad (\text{probabilidad de ir de } [\sigma] \text{ a } [\sigma'])$$

$$\sum_{[\sigma']} T(\sigma'|\sigma) = 1.$$

- Para que las configuraciones alcancen la distribución de *equilibrio* $p[\sigma]$, se debe cumplir

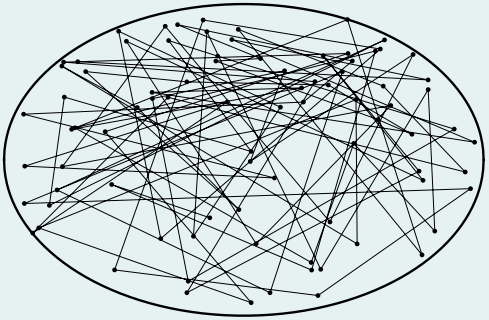
$$p[\sigma] = \sum_{[\sigma']} p[\sigma']\, T(\sigma|\sigma').$$

Esto se puede lograr a través de la condición de *balance detallado*

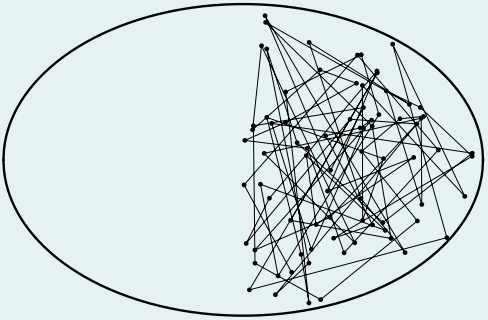
$$T(\sigma'|\sigma)p[\sigma] = T(\sigma|\sigma')p[\sigma'].$$

- Los algoritmos Monte Carlo son capaces de generar este tipo de cadenas. Se trabaja con cadenas *ergódicas*, es decir, para cualesquiera $[\sigma]$ y $[\sigma']$ se debe satisfacer $T(\sigma'|\sigma) \neq 0$.

Cadena ergódica

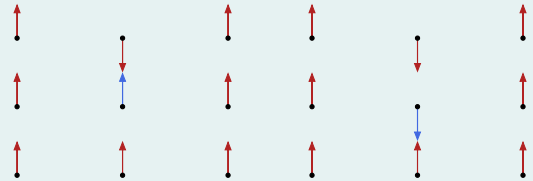


Cadena no ergódica



Algoritmo de Metropolis para el modelo de Ising

1. Generar una configuración inicial aleatoria $[\sigma]$.
2. Elegir un sitio x con valor $\sigma_x \in [\sigma]$ y hacer el cambio $\sigma'_x = -\sigma_x$.
3. Se implementa balance detallado



$$\frac{T(\sigma'|\sigma)}{T(\sigma|\sigma')} = \frac{p[\sigma']}{p[\sigma]} = e^{-\beta\Delta H}, \quad \Delta H[\sigma,\sigma'] = H[\sigma'] - H[\sigma]$$

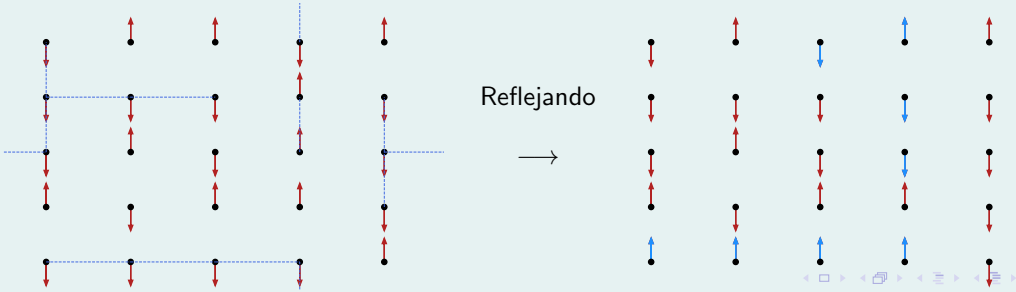
de la siguiente forma: si $\Delta H[\sigma,\sigma'] \leq 0$, entonces la nueva configuración de la cadena es $[\sigma']$, de lo contrario, el cambio $[\sigma] \to [\sigma']$ se acepta con probabilidad $\exp(-\beta\Delta H)$.

4. Se repiten los pasos anteriores $\forall x$.

Estos cuatro pasos deben realizarse múltiples veces hasta tener configuraciones distribuidas de acuerdo a $p[\sigma]$ (termalización). El algoritmo de Metropolis es *local*.

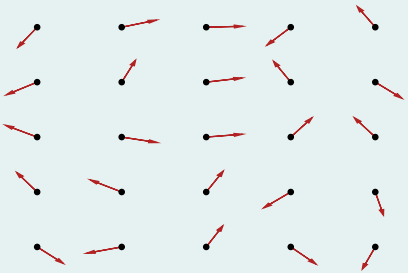
Algoritmo de cluster para modelo de Ising

1. Generar una configuración inicial $[\sigma]$ como se desee.
2. Se crean enlaces entre los espines de la red de la siguiente forma: Si dos sitios vecinos tienen el mismo espín, entonces se genera un enlace entre ellos con probabilidad $p = 1 - \exp(-2\beta)$. Si el espín es opuesto, no se crea enlace. Se realiza este proceso en toda la red.
3. Se identifican los *clusters* de espines. Se puede usar el algoritmo de Hoshen-Kopelman.
4. Se cambia el signo de los espines de cada cluster con probabilidad $1/2$.

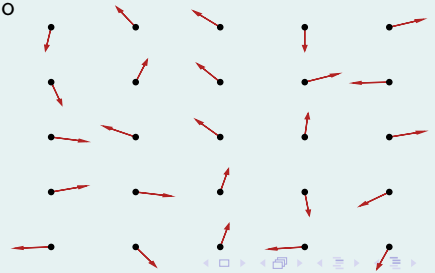


Algoritmo de cluster para modelo XY

1. Generar una configuración inicial $[\sigma]$ con $\sigma_i \in \mathbb{S}^1$.
2. Se define una dirección con un vector aleatorio $\vec{r} \in \mathbb{S}^1$. Para dos sitios vecinos con espines $\vec{\sigma}_i$, $\vec{\sigma}_j$, se crea un enlace con probabilidad $p = 1 - \exp\left[-2\beta(\vec{r}\cdot\vec{\sigma}_i)(\vec{r}\cdot\vec{\sigma}_j)\right]$. Se usa el mismo vector aleatorio en toda la red para generar todos los enlaces en la red.
3. Se identifican los clusters.
4. Se reflejan los espines de los clusters respecto a la dirección ortogonal a $\vec{r}$, $\vec{\sigma}_i \to \vec{\sigma}_i - 2\vec{r}(\vec{r}\cdot\vec{\sigma}_i)$, con probabilidad $1/2$.



Reflejando todo respecto al ortogonal a $\vec{r}$

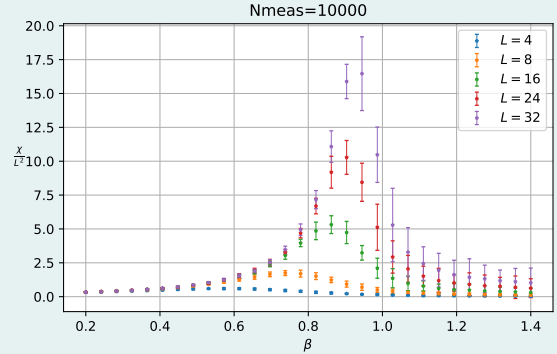


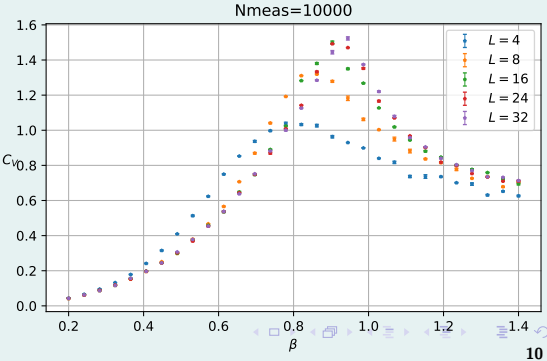
Observables modelo XY

Se pueden medir diversas observables

$$C_V = \frac{\beta^2}{V}(\langle E^2\rangle - \langle E\rangle^2), \quad M = \sum_i \sqrt{\sigma_i^x\sigma_i^x + \sigma_i^y\sigma_i^y}$$

$$\chi_M = \frac{1}{V}(\langle M^2\rangle - \langle M\rangle^2), \quad V = L^2 \,(\text{volumen}).$$





Algoritmo de gusano para modelo XY

- Algoritmo que considera un espacio de configuraciones equivalente al de espines.

$$Z = \int_0^{2\pi} \prod_i \frac{d\theta_i}{2\pi} \prod_{\langle ij \rangle} \exp\left[\beta \cos(\theta_i - \theta_j)\right].$$

Usando

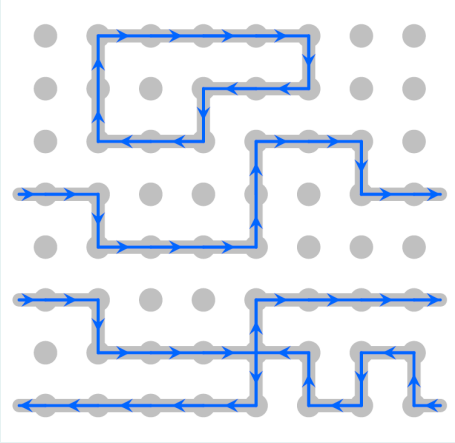
$$\sum_{\nu=-\infty}^{\infty} I_\nu(\beta) \exp(i\nu\theta) = \exp(\beta \cos\theta),$$

entonces

$$Z = \int_0^{2\pi} \prod_i \frac{d\theta_i}{2\pi} \prod_{\langle ij \rangle} \left(\sum_{J_{ij}=-\infty}^{\infty} I_{J_{ij}}(\beta) \exp[iJ_{ij}(\theta_i - \theta_j)] \right)$$

$$Z = \sum_{J_{CP}} \prod_{\langle ij \rangle} I_{J_{ij}}(\beta),$$

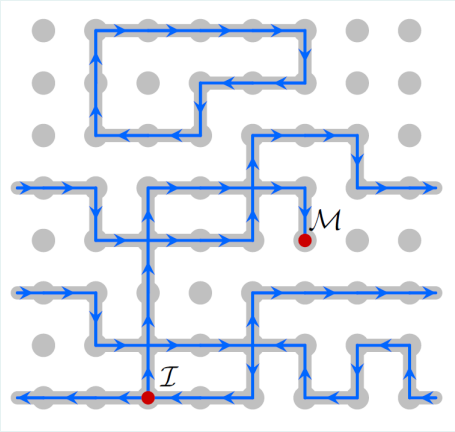
donde CP quiere decir "closed path", es decir, aquellas configuraciones tales que $\boldsymbol{\nabla} \cdot J_i = \sum_j J_{ij} = 0$.



De forma similar se puede probar que

$$G \equiv \langle \vec{\sigma}_I \cdot \vec{\sigma}_M \rangle = \sum_{J_{OP}} \prod_{\langle ij \rangle} I_{J_{ij}}(\beta),$$

donde OP quiere decir "open path", que se refiere a aquellas configuraciones donde en sitios de la red se tiene $\boldsymbol{\nabla} \cdot J = \pm 1$.



Implementación

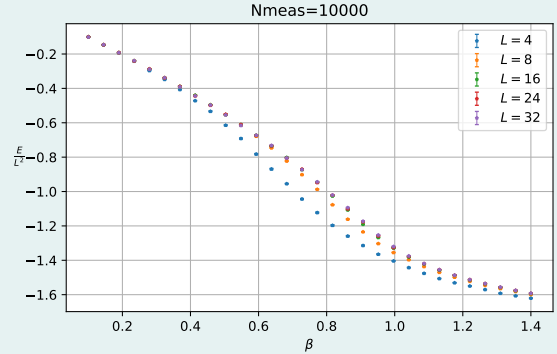
La idea es generar configuraciones en el espacio G, de manera que cuando $I = M$ se recuperen configuraciones en Z.

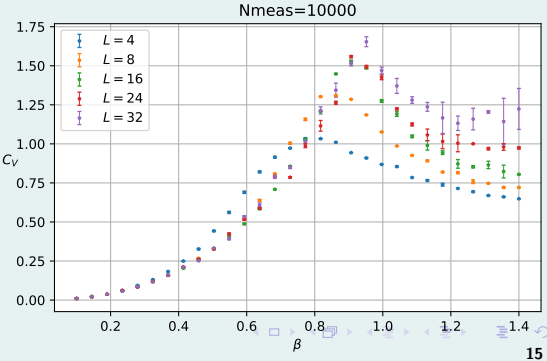
1. Se inicia con $I = M$ en algún sitio aleatorio de la red.
2. Se elige alguno de los vecinos de M, supongamos N, y se propone el cambio de flujo J_{MN} a $J'_{MN} = J_{MN} + 1$.
3. Se acepta la actualización con probabilidad $p = \min\left(1, \frac{I_{J'_{MN}}(\beta)}{I_{J_{MN}}(\beta)}\right)$. Si la actualización se acepta, se elige ahora $M = N$, de los contrario se repite el paso anterior.
4. Se repiten los pasos 2 y 3 hasta que $I = M$. Cuando esto ocurre se elige otro sitio aleatorio en la red y se fija ahí $I = M$. Posteriormente se repiten los pasos 2 y 3.

Observables

$$\langle E \rangle = -\frac{1}{Z}\frac{\partial Z}{\partial \beta} = -\left\langle \sum_{\langle ij \rangle} \frac{I'_{J_{ij}}(\beta)}{I_{J_{ij}}(\beta)} \right\rangle,$$

$$\langle E^2 \rangle = \left\langle \sum_{\langle ij \rangle} \frac{I''_{J_{ij}}(\beta)}{I_{J_{ij}}(\beta)} \right\rangle - \left\langle \sum_{\langle ij \rangle} \left(\frac{I'_{J_{ij}}(\beta)}{I_{J_{ij}}(\beta)} \right)^2 \right\rangle + \left\langle \left(\sum_{\langle ij \rangle} \frac{I'_{J_{ij}}(\beta)}{I_{J_{ij}}(\beta)} \right)^2 \right\rangle.$$





Ventajas y desventajas

Algoritmo de gusano

- Sencillo de programar.
- Es computacionalmente eficiente.
- Conceptualmente más complicado.
- Medir observables es, en general, más difícil.

Algoritmo de cluster

- Es computacionalmente eficiente.
- Medir observables no tiene complicación.
- Programar un algoritmo que identifique clusters no es trivial.

Programas, explicaciones más detalladas y lista completa de referencias en:
https://github.com/Fabian2598/O-2-Model
jafanica@ciencias.unam.mx