Trabajo final de física computacional

El mono

Abstract

Descripción del trabajo.

1 Introducción y motivación

Nuestra intención es usar integración monte carlo para calcular observables de un sistema termodinámico. Esto requiere muestrear en espacios de estados (los ensambles) enormes. En general, demasiado grandes para hacer muestreo directo o *importance sampling*. Necesitamos una forma de tomar muestras de una distribución en espacios grandes, en este contexto entra Markov Chain Monte Carlo.

Estudiamos MCMC de [LS07; McE16; Kle07]. Tiene muchísimas variantes, como Metrópolis, Glauber, Heat bath, Swendsen, Wolff (ver [Jan12; Met+53]) pensadas para esquivar ciertos problemas cuando la distribución que queremos samplear está cerca de un punto crítico; o Gibbs, Hamilton (ver [Kle07; McE16]) pensadas para generar muestras de forma más eficiente (más adelante explicamos en qué sentido).

El algoritmo consiste en generar una cadena de markov cuya distribución invariante sea la deseada. Es una idea brillante, pero justamente por su naturaleza tiene un par de problemas importantes: Primero, pasa un tiempo hasta que la cadena alcanza la distribución invariante, por lo que las primeras muestras deben ser descartadas, esto se conoce como el *burn-in*. Segundo, las muestras no son independientes entre si, estan *autocorrelacionados*, esto resulta en que estamos subestimando la varianza de la integral y debemos corregirla.

Hay tests para estudiar estos problemas (ver [Roy20; McE16; LS07]), en particular, McElreath menciona que R (un software reconocido en estadistica) reporta el *tiempo integrado de autocorrelación* τ_{int} y el $\hat{R}$ de Gelman-Rubin (ver [GR92]). Es por esto que nos decantamos por estos tests.

La idea del trabajo es estudiar estos fenómenos, con esa excusa, vamos a programar metrópolis y gibbs y usarlos en el modelo de Ising. Vamos a medirles el τ_{int}, el $\hat{R}$ y el τ_{exp} también.

2 Preliminares

Pequeña introducción diciendo que vamos a describir brevemente los algoritmos y los tests. Mi idea es que en esta sección se entiendda por qué existe el Gibbs sampler y cómo funcionan los tests.

2.1 Algoritmos de MCMC

Aca describimos un poco metrópolis, aunque ya lo hicimos en el TP 4. Tambien describimos Gibbs sampler, que la idea es usar la información de las distribuciones marginales para hacer mas efectivo el sampleo.

2.2 Tests

En este lugar destacamos que los tests se tienen que hacer sobre una función escalar de la muestra, no se pueden hacer sobre la muestra directamente porque el espacio de estados podría no tener suficiente estructura ($\mathbb{R}$, por ejemplo, tiene producto, distancia, etc. Suficiente para hablar de correlación). Nosotros vamos a usar, como observable, la magnetización, porque distingue bien si la muestra es bimodal o no. Estoy considerando agregar la energía al analisis, pero creo que no se gana mucho con eso.

Aca describimos, para empezar, el $\hat{R}$, que es un test para el *burn-in*. Luego hablamos de la autocorrelación (que se mide igual siempre, no es que se necesite un test específico, mas que nada se aproxima la autocor posta con el estimador estandar) y quiero agregar una pequeña discusión sobre el *gap espectral* de un proceso de markov, que nos garantiza la existencia de τ_{exp}, que también puede usarse para estimar el burn-in.

2.3 Implementación

Implementamos nosotros mismos las simulaciones, estan en este github (es publico así que deberían poder verlo sin problema). Generamos los números aleatorios con el Mersenne Twister (ver [MN98]), detalle que podría ser de interes. Tratamos de dejar todo prolijito y documentado en el código.

3 Resultados

Generamos TAL y TAL con TAL parametros y bla bla. Quiero tomar una temperatura cercana a la crítica pero inferior, así la distribución objetivo es bimodal y la magnetización resulta un buen observable para comprobar que las cadenas de markov esten funcionando correctamente. Mas adelante puedo repetir el experimento con otras temperaturas.

Una vez fijados los parámetros (tamaño del modelo, temperatura de la distribución), todo lo que tenemos que hacer es generar muestras y hacerles los tests correspondientes. Imagino que en esta sección habrá, entonces, tablas comparando los valores de $\hat{R}$, τ_{int} y τ_{exp} para ambos algoritmos (Metrópolis y Gibbs sampler), quizás para varias temperaturas. Tal vez agregue también calculos de complejidad algoritmica y tiempos de computo (no aporta mucho, pero para efecto dramático...)

4 Discusión

Esperamos que Gibbs saque jugo de conocer, justamente, las distribuciones marginales de lo que esta sampleando (esa es la joda del Gibbs sampler) y trabaje mucho mas rápido. Honestamente no espero que tenga tiempos de autocorrelación diferentes ni un $\hat{R}$ distinto.

5 Conclusiones

Ni idea no programé nada todavía.

References

[GR92] Andrew Gelman and Donald B. Rubin. "Inference from Iterative Simulation Using Multiple Sequences". In: *Statistical Science* 7.4 (1992), pp. 457–472.

[Jan12] Wolfhard Janke. "Monte Carlo Simulations in Statistical Physics". Lecture notes, University of Leipzig. 2012.

[Kle07] A. Klenke. *Probability Theory: A Comprehensive Course*. Universitext. Springer London, 2007.

[LS07] Wolfgang von der Linden and Ewald Schachinger. *Chapter 7: Markov Chain Monte Carlo*. `https://itp.tugraz.at/LV/wvl/Comp_Simulationen/`. Part of the lecture notes *Computer Simulationen*, Graz University of Technology. 2007.

[McE16] R. McElreath. *Statistical Rethinking: A Bayesian Course with Examples in R and Stan*. A Chapman & Hall book. CRC Press/Taylor & Francis Group, 2016.

[Met+53] Nicholas Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

[MN98] Makoto Matsumoto and Takuji Nishimura. "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator". In: *ACM Trans. Model. Comput. Simul.* 8.1 (1998), pp. 3–30.

[Roy20] Vivekananda Roy. "Convergence Diagnostics for Markov Chain Monte Carlo". In: *Annual Review of Statistics and Its Application* 7.Volume 7, 2020 (2020), pp. 387–412.