Departamento de Física
UNIVERSIDADE DE AVEIRO

Modelação e Física Estatística

2020.07.15

Exame de Recurso
Duração: 3h

1. (4 valores) Pretende-se gerar números aleatórios com a distribuição gama truncada inferiormente $p_{k,\lambda,x_0}(x)$, com $k > 1$, $\lambda > 0$, $x_0 \geq 0$, $x \geq 0$ definida como $p_{k,\lambda,x_0}(x) = \frac{\lambda^k}{\Gamma(\lambda x_0,k)} x^{k-1} e^{-\lambda x}$ se $x \geq x_0$ e $p_{k,\lambda,x_0}(x) = 0$ se $x \leq x_0$ onde $\Gamma(\lambda x_0, k) = \int_{\lambda x_0}^{\infty} t^{k-1} e^{-t} dt$ é a função gama incompleta superior (obtida no Matlab como gammainc(lambda*x0,k,'upper') * gamma(k)).
   a. Faz uma função function x=ex1a(n, k,lambda,x0) que devolve um vetor x de n números aleatórios, com a distribuição gama truncada.
   b. Faz um script ex1b.m que calcula 10000 números aleatórios usando a função desenvolvida na alínea anterior. Faz um histograma normalizado e compara o resultado com $p_{k,\lambda,x_0}(x)$. Considera o caso x0=90; k=100; lambda=1;

2. (8 valores) Pretende-se fazer uma simulação de um gás de fotões tridimensional, confinado a um cubo de lado L, usando o algoritmo de Metropolis. Os estados de um gás de fotões, partículas indiscerníveis, são especificados pelo número de fotões, $n_{\vec{k}}$ com um vetor de onda, $\vec{k} = \frac{\pi}{L}(n_x, n_y, n_z)$ onde n_x , n_y , n_z tomam valores $1, 2, \cdots, \infty$. A energia de um fotão com vetor de onda $\vec{k}$ é dada por $E_{\vec{k}} = \hbar c k - \sqrt{3}\frac{hc}{2L}$ onde c é a velocidade da luz e $\hbar$ é a constante de Planck dividida por 2π. A subtração do termo constante a todas as energias dos fotões corresponde a usar um zero de energia em que os fotões menos energéticos têm energia zero. Podemos usar para unidade de energia, $u_E = \frac{hc}{2L}$ e uma unidade de temperatura $u_T = \frac{u_E}{k_B}$. Considera valores de $n_{x(\text{ou } y, ou\ z)} < n_{\max} = 10$.
   a. Faz uma função ex2a.m que simula o sistema usando o algoritmo de Metropolis a uma temperatura T, ou seja: **1)** inicializa o estado do sistema sem fotões e inicializa

a energia, E=0. **2)** para cada passo atualiza o estado do sistema n_{max}^3 vezes **3)** em cada atualização a) escolhe um k da rede ao acaso b) com probabilidade 0.5 tenta aumentar o número de fotões nesse k em uma unidade e com probabilidade 0.5 tenta diminuir esse número em uma unidade (o número de fotões não pode ser negativo) c) calcula a variação de energia do sistema, dE (dE é a energia necessária para criar ou remover um fotão com o k considerado) d) aceita o novo estado com probabilidade min([1, exp(-dE/T)]) **4)** simula o sistema durante npassos=nmedidas+nequi **5)** desprezando os nequi passos iniciais para equilibração, calcula nos restantes passos a energia média do sistema. A saída da função deve ser a energia média.

b. Faz um script ex2b.m que faz simulações do sistema com $n_{\max} = 10$ e 10 temperaturas entre 0.1 e $T_{max} = 1$. Considera nmedidas=10000 e nequi=1000. Compara com a expressão teórica para a energia média: $\langle E \rangle = \frac{L^3}{15} \frac{\pi^2 k_B^4}{\hbar^3 c^3} T^4$ ou seja com $\langle E \rangle / u_E = \frac{\pi^5}{15} (T/u_T)^4$.

3. (8 valores) Um sistema é formado por partículas do tipo A, B e C. As partículas A e B quando se encontram reagem de acordo com $A + B \to 2B$. Esta reação ocorre a uma taxa de probabilidade $\alpha \frac{N_B}{N}$ onde N_B é o número de partículas do tipo B e N é o número total de partículas. Uma partícula B dá origem a uma partícula C (de acordo com a reação $B \to C$) a uma taxa de probabilidade β, com $\alpha \leq \beta$. Um estado do sistema é especificado indicando o tipo de cada partícula $\vec{s} = (s_1, \dots, s_i, \dots, s_N)$ com $s_i = 1$ uma partícula do tipo A, $s_i = 2$, uma partícula do tipo B e $s_i = 3$, uma partícula do tipo C.

a. Faz uma função ex3a.m que simula uma cadeia de Markov de acordo com o seguinte algoritmo: **1)** Dado N, N_A, N_B, α, β e número de passos, npassos **2)** Constrói o estado inicial $\overrightarrow{s_0}$ 2) Em cada passo faz N atualizações 3) Cada atualização corresponde a: a) escolher uma partícula ao acaso b) se a partícula for do tipo B passar a partícula a tipo C c) se a partícula for do tipo A transformar a partícula em tipo B com probabilidade $\frac{\alpha}{\beta} \frac{N_B}{N}$ d) se a partícula for do tipo C não fazer nada **3)** repetir 2) e 3) por npassos **4)** Em cada passo regista o número de partículas observado nesse passo e definir o tempo de cada passo como $\mathrm{t(passo)} = \mathrm{t(passo} - 1) + \frac{1}{\beta}$. O output da função deve ser o número de partículas de cada tipo em cada passo e o vetor com os tempos correspondentes. Nota: cada atualização corresponde a um passo $dt = \frac{1}{\beta N}$ e por isso N atualizações correspondem a um incremento temporal $\frac{1}{\beta}$.

b. Faz um script ex3b.m que usa a função anterior para simular um sistema de $N = 10^4$ partículas para $\alpha = 0.2\ /dia$, $\beta = \frac{1}{15}/dia$ com inicialmente 99% de partículas A e 1% de partículas B. Simula durante 10 passos. Faz um gráfico da fração de partículas de cada espécie, $x_A = \frac{N_A}{N}$, $x_B = \frac{N_B}{N}$, $x_C = \frac{N_C}{N}$ em função do tempo. Compara o resultado com a solução do sistema de equações diferenciais:

$$\begin{aligned} \frac{dx_A}{dt} &= -\alpha x_A x_B \\ \frac{dx_B}{dt} &= \alpha x_A x_B - \beta x_B \\ \frac{dx_C}{dt} &= \beta x_B \end{aligned}$$

A solução no intervalo [0, tmax] pode ser obtida usando a função ode45 do Matlab:

```
tspan=[0, tmax]; x0=[NA,NB,NC]/N;
[t,x] = ode45(@(t,x) F(t,x,alfa,beta), tspan, x0);

function Fv=F(t,x,alfa, beta)
Fv(1,1)=-alfa*x(1)*x(2);
Fv(2,1)=alfa*x(1)*x(2)-beta*x(2);
Fv(3,1)=beta*x(2);
end
```