Computing the Universe '99:
Monte Carlo realization of distributions, Part 2

(3) Gaussian density fields and cosmological initial conditions

In most cosmological simulations, the initial conditions are a Gaussian random density field, with a specified power spectrum. To generate a 3-d Gaussian random field, one usually employs a Fast Fourier Transform (FFT) in order to perform the computation in a reasonable amount of computer time. However, if you want to gain intuition instead of learn calling procedures for FFTs, it is easier to work with 1-dimensional fields and "slow Fourier transform" (i.e., direct summation of Fourier modes). In this problem, you will generate 128-element, 1-d Gaussian fields.

You will need to use the random number generators `ranf` and `gennor` described in Part 1. In your programs for this part, you should also include lines to seed the random number generator. To do this, include the line `setall(iseed1,iseed2)` (C) or `call setall(iseed1,iseed2)` (fortran) in your programs before you call `ranf` or `gennor` for the first time. `iseed1` and `iseed2` are integer variables whose values you can either hardwire into the program or read as input parameters. Every time you run the program with the same seeds, you should get the same output, but if you change the seeds you will get a different sequence of random numbers, and hence a different realization of the density field. (Throughout this problem, I will refer to the "density field," but what we are really dealing with is the density *contrast* field $\delta = \rho/\overline{\rho} - 1$, which has a mean of zero by definition.)

(a) Random phase density field, simple case

Although a true Gaussian field has a specific distribution of the moduli of its Fourier components, if the number of components is very large it is sufficient that the complex phases of these components be random. So for a start, we'll generate random fields that are sums of cosine waves with random phases.

We want to compute density fields on a 1×128 array, which spans the spatial coordinate range $x = 0$ to $x = 1$. This field can represent Fourier modes from the fundamental $k = 1$ up to the Nyquist frequency $k_N = N/2 = 64$.

Write a fortran or C program that will

(i) Generate an array of random phases θ_k, which should be independent of each other and uniformly distributed in the range $0 - 2\pi$. Do this by using the uniform random number generator `ranf`.

(ii) Compute the density field

$$\delta(x) = \sum_{k=1}^{k=k_N} 2A_k \left[\cos(2\pi kx + \theta_k)\right]. \tag{1}$$

Note that although the index of the δ array runs from 1 to 128 (or 0 to 127 if you are a C user), this corresponds to x running from $x = 0$ to $x = 127/128$.

Then

(iii) Run the program with $A_k = 1$, $A_k = k^{1/4}$, and $A_k = k^{-1/4}$, corresponding to three different power spectra. In each case, use the same random number seed, so that the phases θ_k are the same for each power spectrum. Plot these three 1-d density fields (using whatever plotting package you wish), scaling the vertical axes so that each plot looks reasonable.

If you feel like being fancy, make a plot that actually shows all of the individual components so that one can see how they add up to produce the final field.

(iv) Compute and plot the histogram of $\delta(x)$ values in each case.

What do you conclude about the probability distribution $P(\delta)$ of random phase fields and about the influence of the power spectrum on the properties of these fields?

(b) Gaussian density field, more formal

Let's think more carefully about the basis for part (a). Each Fourier component $\widetilde{\delta}(k)$ is a complex number $A_k e^{i\theta_k}$, with modulus A_k and phase θ_k. The real space density contrast field $\delta(x)$ is related to its Fourier transform $\widetilde{\delta}(k)$ by the inverse Fourier transform relation:

$$
\begin{aligned}
\delta(x) &= \sum_{k=-k_N}^{k=k_N} \widetilde{\delta}(k) e^{-2\pi i k x} \\
&= \sum_{k=-k_N}^{k=k_N} A_k e^{i\theta_k} e^{-2\pi i k x} \\
&= \sum_{k=-k_N}^{k=k_N} A_k e^{i(-2\pi k x + \theta_k)} \\
&= \sum_{k=-k_N}^{k=k_N} A_k \left[\cos(-2\pi k x + \theta_k) + i \sin(-2\pi k x + \theta_k)\right].
\end{aligned}
\tag{2}
$$

For a continuous density field, these sums would be replaced by integrals, and the range of integration would be $k = -\infty$ to $k = +\infty$, but we are generating a discrete density field defined at a finite number of spatial positions x_i.

We want the real space density field $\delta(x)$ to be real-valued, not complex-valued, and in order for this to happen its Fourier transform $\widetilde{\delta}(k)$ must be *Hermitian*, which means that it satisfies the equation

$$\widetilde{\delta}(k) = \widetilde{\delta}^*(-k), \tag{3}$$

where the * denotes complex conjugate. With a small bit of algebra, you should be able to convince yourself that this constraint reduces the sum in equation (2) to the sum in equation (1). (Actually, there is a change of sign of θ_k, but because the phases are random anyway, we can ignore that.) The constraints imposed by equation (3) also reduce the amount of independent information contained in the $N = 2k_N$ complex numbers $\widetilde{\delta}(k)$ to equal the amount contained in the N real numbers $\delta(x)$.

The power spectrum $P(k)$ is the variance of the distribution of moduli A_k,

$$\langle A_k^2 \rangle = P(k). \tag{4}$$

The shortcoming of the method used in part (a) is that it sets $A_k = \sqrt{P(k)}$, which satisfies equation (4) but does not have the *distribution* of A_k values that is expected for a Gaussian random field.

The simple way to generate random values of the Fourier components $\widetilde{\delta}(k)$ that have the correct distribution is to choose the real and imaginary parts of $\widetilde{\delta}(k)$ from independent Gaussian distributions, each with variance $P(k)/2$. Since the real and imaginary parts are independent, the complex phase of $\delta(k)$ will be uniformly distributed between 0 and 2π. Since $A_k^2 = \mathrm{Re}^2\widetilde{\delta}(k) + \mathrm{Im}^2\widetilde{\delta}(k)$ and the real and imaginary parts are independent,

$$\langle A_k^2 \rangle = \langle \mathrm{Re}^2\widetilde{\delta}(k) \rangle + \langle \mathrm{Im}^2\widetilde{\delta}(k) \rangle = P(k), \tag{5}$$

in agreement with equation (4).

You could generate the Fourier components in this way, compute A_k and θ_k from the real and imaginary parts, and calculate the sum of equation (1) to generate your density field. However, this involves computing an arctangent to get θ_k, so it is a bit better to expand equation (1) to obtain

$$\begin{aligned}
\delta(x) &= \sum_{k=1}^{k=k_N} 2A_k \left[\cos(2\pi kx + \theta_k)\right] \\
&= 2\sum_{k=1}^{k=k_N} \cos(2\pi kx)A_k\cos(\theta_k) - \sin(2\pi kx)A_k\sin(\theta_k) \\
&= 2\sum_{k=1}^{k=k_N} \cos(2\pi kx)\mathrm{Re}\widetilde{\delta}(k) - \sin(2\pi kx)\mathrm{Im}\widetilde{\delta}(k).
\end{aligned} \tag{6}$$

Write a fortran or C program that will

(i) Generate an array of real and imaginary parts of the Fourier components $\widetilde{\delta}(k)$, drawing each from a Gaussian distribution of variance $P(k)/2$. You do this using the Gaussian deviate routine `gennor`, e.g.,

$$\mathtt{re(k)} = \mathtt{sqrt(P(k)/2.)} * \mathtt{gennor(0.0, 1.0)}.$$

(ii) Compute the density field $\delta(x)$ using equation (6).

Then

(iii) Run the program with $P(k) = 1$, $P(k) = k^{1/2}$, and $P(k) = k^{-1/2}$. Plot these three 1-d density fields (using whatever plotting package you wish), scaling the vertical axes so that each plot looks reasonable.

(iv) Compute and plot the histogram of $\delta(x)$ values in each case.

How do the results of this procedure compare to the results from part (a)?

(c) Particle positions

Now suppose that you want to use your density field to compute initial particle positions for a 1-d N-body simulation. Your unperturbed particle distribution is a 1-d "grid," i.e., a set of particles uniformly spaced at the positions $q_i = i/128$, with i running from 0 to 127. These particles have a uniform linear density $\overline{\lambda} = 1/(q_i - q_{i-1}) = 128$. You will perturb the particles, moving each one to a new position $x_i = q_i + \psi_i$. Some particles will move closer together, others further apart. The implied linear density at the position of particle i is $\lambda = 1/(x_i - x_{i-1})$.

(i) Argue that if particle orbits do not cross (i.e., the particles stay in their initial order), then

$$\frac{\lambda}{\overline{\lambda}} = \left(1 + \frac{d\psi}{dq}\right)^{-1}, \tag{7}$$

and that in the limit of small displacements ($|d\psi/dq| \ll 1$) this yields

$$\delta(q) \approx -\frac{d\psi}{dq} \approx -\frac{d\psi}{dx}. \tag{8}$$

(The last equality follows from the fact that for small displacements there is not much difference between the unperturbed coordinates q and the perturbed coordinates x.)

(ii) By combining equation (8) with the Fourier transform definition (equation 2 and a similar equation for ψ), one can obtain the relation between Fourier components of ψ and Fourier components of δ,

$$\widetilde{\psi}(k) = \frac{i}{2\pi k}\widetilde{\delta}(k). \tag{9}$$

Modify your program from part (b) so that it calculates the displacement field $\psi(x)$ in addition to the density field $\delta(x)$. Note that the multiplication by i transforms real numbers into imaginary numbers and vice versa.

(iii) The approximation of small displacements is roughly equivalent to the approximation that $\delta(x) \ll 1$ — in other words, you need to generate your initial conditions at a time when the density field still obeys linear perturbation theory. For each of the three power spectra that you used in part (b), multiply the power spectrum by an arbitrary factor so that the density field satisfies $\langle\delta^2\rangle \sim 0.1$. Then run your modified program, and make plots that show the 1-d particle distribution above the corresponding density field. Does your program seem to be producing reasonable results?

Comments and extensions

There are a few things you could do to improve your understanding of this subject if you have time after finishing the above. The first is to see what happens if you smooth your 1-d density fields over various scales. The easiest way to do this is "boxcar" smoothing, in which you replace the value of δ at each pixel by the value of δ averaged over several pixels. The number of pixels used for averaging defines the scale of the smoothing, and you could plot the histogram of δ values for different smoothing scales. If you want to be fancier, you could use a Gaussian smoothing window instead of a boxcar. How does the power spectrum influence the relative strength of fluctuations on different scales in real space?

If you're feeling more ambitious, you could revise your program from part (c) to work in two dimensions. Now each wavenumber $\mathbf{k}$ is a vector, and in the Fourier transform sums each component k_x and k_y runs from $-k_N$ to k_N. Equation (9) is modified to the vector equation

$$\widetilde{\psi}(\mathbf{k}) = \frac{i\mathbf{k}}{2\pi k^2}\widetilde{\delta}(\mathbf{k}).$$

If you plot your 2-d perturbed particle distribution, it will look much more like initial conditions for a real N-body simulation.

In order to compute the density field via equation (6), you had to perform operations on the $N/2$ Fourier components for each of the N values of the density field, so the total expense of the calculation scales as N^2. For $N = 128$ this is no problem, but if you imagine generating initial conditions on a 3-dimensional, $N = 128^3$ grid, you can see that this approach might become computationally impractical. The solution is to use an FFT, a clever numerical technique that reorders the calculation to eliminate redundancies and can therefore do this computation of the density field in a time that scales as $N \log N$ instead of N^2. However, the FFT is carrying out essentially the same operation as that in equation (6). If you want to look at an example of a program that generates Gaussian density fields in 3-dimensions using an FFT, you can find one in `/hosts/a/home/dhw/Examples/ranfield.f`.