Computational Methods IV

Martin Weinberg

June 25, 1999

1 The Monte Carlo Method

1.1 What is Monte Carlo?

Suppose we want to perform some integral:

$$I[f(\mathbf{x})] = \int_C d\mathbf{x}\, f(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ is some independent variable (which may be multidimensional) and C is the domain of $\mathbf{x}$. If we have a way of drawing random samples of $\mathbf{x}$ from C, $\{X_t : t = 1, \ldots, n\}$, then we may approximate the integral as

$$I_A[f(\mathbf{x})] = \frac{1}{n}\sum_{t=1}^{n} f(\mathbf{X_t}). \tag{2}$$

Clearly $I_A \to I$ as $n \to \infty$. The error term [3] is $O(n^{-1/2})$.

1.2 When to use Monte Carlo?

Is I_A a good or bad approximation to I? It depends on the number of dimensions. Suppose we do each dimension with an m-point Gauss-type rule. Since the integrand tends to be costly, let's compare to the

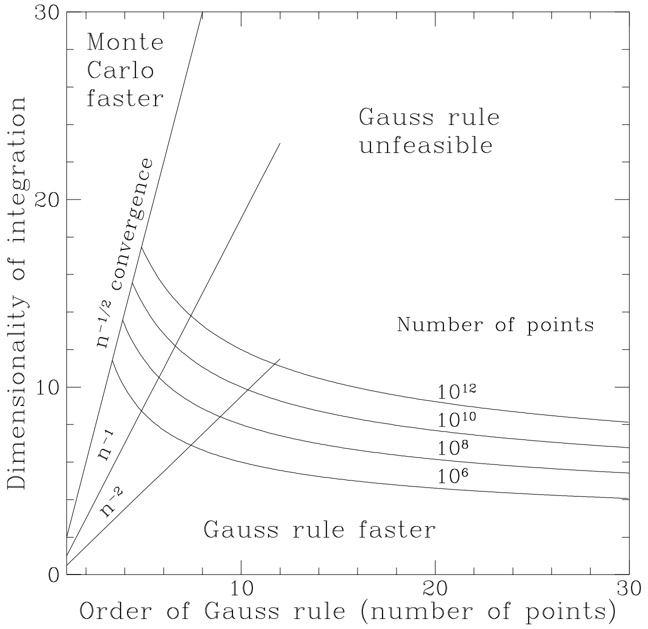

Figure 1: Computability of Monte Carlo vs Gaussian integration. Straight lines show the boundary between more rapidly converging power-law error terms and iterated Gaussian error terms. Four curves are labeled by number of evaluations needed as a function of order and dimension. Using the Gaussian rule is impossible above these lines. Monte Carlo is clear the method of choice for high-dimensional integrals.

Monte Carlo case keeping the number of evaluations n fixed. If we divide knots equally among all dimensions, there will be $k = n^{1/d}$ points per dimension. The truncation error is then $O(n^{-(2m-1)/d})$. Figure 1 compares this with Monte Carlo. For high dimensionality d, Monte Carlo will always win.

There are another reasons to prefer Monte Carlo. The most common is that we have some sort of irregular domain which makes standard quadrature intractable.

1.3 Generating probability distributions

Suppose that the integrand $f(\mathbf{x})$ is narrowly peaked about some $\mathbf{x}_o$. The convergence rate will still be proportional to $n^{-1/2}$ but penalized by some big numerical factor. This factor takes in to account all of the samples in domain C "wasted" in locating the peak. There are several ways of fixing this problem. The one I want to make sure you understand is change of integration variables.

Let us look at a one-dimensional example. Suppose we want to perform

$$I[f(x)] = \int_{-\infty}^{\infty} dx\, f(x) \tag{3}$$

and $f(x)$ is peaked about x_o. We can multiply and divide by some peaked function, $(1+(x-x_o)^2)$ say and change variables to $x = \tan(\theta)$. The integral becomes

$$I[f(x)] = \int_{-\pi/2}^{\pi/2} d\theta\, f(x_o + \arctan\theta)\sec^2\theta. \tag{4}$$

To perform this integral by Monte Carlo, we now sample θ uniformly in $[-\pi/2, \pi/2]$.

1.3.1 The CDF

More generally, this procedure replaces the integral

$$I[f(x)] = \int_C dx\, f(x) \tag{5}$$

by the integral

$$I[f(x)] = \int_C dx\, f_o(x)\frac{f(x)}{f_o(x)}. \tag{6}$$

Suppose that $C = [a,b]$. Then to do equation (6) by the Monte Carlo method, we choose x so that

$$\theta = \frac{\int_a^x dx' \, f_o(x')}{\int_a^b dx' \, f_o(x')} \tag{7}$$

is uniformly sampled in the interval $[0,1]$. This expression is called the *cumulative distribution function* or CDF for x. For efficiency's sake, one tries to choose f_o to make the CDF analytic or at least simple to perform numerically. The choice θ_i then implies x_i and the Monte Carlo approximation to I is

$$I_A[f(\mathbf{x})] = \frac{1}{n} \sum_{t=1}^{n} \frac{f(\mathbf{x_i}(\theta_\mathbf{i}))}{f_o(\mathbf{x_i}(\theta_\mathbf{i}))}. \tag{8}$$

In case the inverse implied by equation (7) is too difficult there are two commonly used implicit approaches to generating the values θ, each described below.

1.3.2 Acceptance-Rejection

We can interpret

$$P(x)\, dx \equiv \frac{f(x')\, dx}{\int_a^b dx' \, f(x')} \tag{9}$$

as the probability of obtaining x in the interval $[x, x+dx]$. Let P_{max} be the maximum of this quantity and $\mathcal{R}$ be a random variate in $[0,1]$. The algorithm is then as follows:

1. Generate a random value x;

2. Generate a random value $\mathcal{R}$;

3. Accept the value of x if $P(x)/P_{max} > \mathcal{R}$ otherwise go to Step (1).

The values of x will then be distributed as $P(x)$.

One doesn't get something for nothing! This procedure will be efficient if $P(x)$ is fairly uniform but very inefficient (that is, generating mostly rejections) if $P(x)$ is peaked. One can and *should* use the same trick used in developing equation (6) to alleviate peakiness.

1.3.3 Weighting

A related approach follows from the form of equation (6). Suppose we want to generate instances of x distributing according equation (9) and our application allows us to arbitrarily weight the importance of the instance[1]. Then, we can use acceptance-rejection or direction computation to generate x with the distribution proportional to $f_o(x)$ weight its influence by $f(x)/f_o(x)$. This procedure is sometimes called *biasing*.

1.4 Example: an N-body realization

In the dynamics lectures, we learned how to generate Boltzmann equilibria from distribution functions and/or density and gravitational potential profiles. Let's look at generating a phase-space distribution for the King model (or any spherical profile really).

The first step is to select a radial location for one's mass point. Note that the mass enclosed inside r,

$$M(< r) = 4\pi \int_0^r dr \rho(r) \tag{10}$$

is proportional to the CDF. Therefore inverting $\mathcal{R} = M(r)/M(r_t)$ gives the distribution of r.

[1]This is often true for applications in radiative transfer and n-body simulation

We now want to generate velocities. Just as for the mass, we want the CDF in velocity variables. For a fixed radius, we have:

$$\begin{aligned}\rho(r) &= \int_0^{v_{max}} dv_r \int_0^{\sqrt{v_{max}^2-v_r^2}} dv_T\, v_T\, f(\mathbf{r},\mathbf{v}) \qquad (11)\\ &= v_{max}^3 \int_0^1 dx\,(1-x^2) \int_0^{\sqrt{1-x^2}} dy\, y\, f(\mathbf{r},\mathbf{v}) \qquad (12)\end{aligned}$$

where $x = v_r/v_{max}$ and $y = v_{max}\sqrt{1-(v_r/v_{max})^2}$. In order to use CDFs to generate x and y we want to change variables to uniform variates. In other words we want $\mathcal{R} = \theta = 2/3(x - x^3/3)$. This cubic has the solution $x = 2\sin(\arcsin\theta/3)$. The y variable is much easier; we want $\mathcal{R} = \phi = y^2$ or $y = \sqrt{\mathcal{R}}$. Altogether we then have a two step processes to get the velocities:

1. $v_r = v_{max}x$
2. $v_t = v_{max}y\sqrt{1-x^2}$.

After generating x and y and thereby v_r and v_t, we use acceptance-rejection to generate the velocity part of the phase-space distribution.

2 Fourier Analysis for N-body Methods

Accurate computation of the gravitational force is key to meaningful simulation of astrophysical dynamical systems. We most often resort to methods *other than* direct summation for a variety of reasons, most importantly (1) numerical efficiency and (2) better approximation of the Collisionless Boltzmann Equation (CBE). A simulation with a modest number of particles N is not necessarily the same as the $N \to \infty$ required for the CBE.

Here we will discuss one of the more important approaches to the force problem: the Fourier representation.

2.1 The basic concept

Computing the force given a mass density distribution is equivalent to solving the Poisson equation:

$$\nabla^2 V = 4\pi G \rho. \tag{13}$$

An often used and productive method for solving PDEs like this one, is to find separable solutions. In the case of Poisson's equation, the separated solutions have the form of a Sturm-Liouville equation. The importance of solutions to the Sturm-Liouville equation is that the resulting set of solutions are orthogonal and complete after applying boundary conditions appropriate to the problem at hand (see [2] for a detailed discussion). This fact is underlies most of the special functions explored in electrostatics, e.g. Bessel functions and Legendre functions both Sturm-Liouville solutions. In Cartesian coordinates, these functions are sines and cosines.

2.2 Periodic box of length L

For a simple concrete and useful example, let us assume that our Universe is periodic. Then, all phase space quantities are periodic with modulus L. We can represent our sines and cosines in exponential form with no loss of generality and therefore

$$V, \rho \propto e^{i2\pi lx/L} e^{i2\pi my/L} e^{i2\pi nz/L} \tag{14}$$

where l, m, n are integers. Defining $k_j \equiv 2\pi j/L$, we may write this as

$$V, \rho \propto e^{i(k_l x + k_m y + k_n z)}. \tag{15}$$

Substituting this into the Poisson equation, we can solve for V given ρ and our problem is solved! Explicitly, our solution is the following Fourier series:

$$-\sum_{lmn}\left(k_l^2+k_m^2+k_n^2\right)V_{lmn}e^{i2\pi lx/L}e^{i2\pi my/L}e^{i2\pi nz/L} = \tag{16}$$

$$4\pi G\sum_{lmn}\rho_{lmn}e^{i2\pi lx/L}e^{i2\pi my/L}e^{i2\pi nz/L} \tag{17}$$

where V_{lmn} and ρ_{lmn} are coefficients (numbers!) and the sums are over all integers l,m,n. Because the sines and cosines are independent orthogonal functions, the coefficients must be equal term by term and the solution is then

$$V_{lmn} = -\frac{4\pi G}{\left(k_l^2+k_m^2+k_n^2\right)}\rho_{lmn}. \tag{18}$$

2.3 The DFT

To summarize, equation (18) tells us that the potential, and therefore the force, at any point in the Universe is known if we know the density. All we need to do, then is to find the coefficients ρ_{lmn}. Because the sines and cosines are orthogonal, we have

$$\rho_{lmn} = \frac{1}{L^3}\int_0^L dx\int_0^L dy\int_0^L dz e^{-i2\pi lx/L}e^{-i2\pi my/L}e^{-i2\pi nz/L}\rho(x,y,z). \tag{19}$$

By dividing the box length L into N subintervals we may approximate the continuous Fourier transform by a discrete sum:

$$\rho_{lmn} \approx \frac{1}{L^3}\sum_{q=0}^{N-1}\Delta\sum_{r=0}^{N-1}\Delta\sum_{s=0}^{N-1}\Delta e^{-i2\pi lx_q/L}e^{-i2\pi my_r/L}e^{-i2\pi nz_s/L}\rho(x_q,y_r,z_s) \tag{20}$$

where q, r, s are integers, $\Delta \equiv L/N$ and $x_q = q\Delta$ and etc. Simplifying, the above equation becomes

$$\rho_{lmn} \approx \frac{1}{N^3} \sum_{q=0}^{N-1} \sum_{r=0}^{N-1} \sum_{s=0}^{N-1} e^{-i2\pi lq/N} e^{-i2\pi mr/N} e^{-i2\pi ns/N} \rho(x_q, y_r, z_s). \tag{21}$$

This defines the three dimensional *Discrete Fourier Transform* of ρ. In other words, if we can sample ρ on the uniform grid of points defined by Δ, we can solve for the potential using the Fourier series in §2.2.

There are optimal procedures for distributing (or smoothing) arbitrarily spaced mass points onto a regular grid consistent with the Green function for the Poisson equation. For now, let us assume that we have an appropriately sampled density. Similarly, we are tacitly assuming a periodic Universe but this is not always what one wants. There are ways of manipulating the Fourier series to represent vacuum boundary conditions. Both of these issues will be discussed later on.

For now, let us continue on and ask, "What is the best way of evaluating the Fourier transform ρ_{lmn}?"

2.4 The FFT

2.4.1 The Concept of the Fast Fourier Transform

A fast Fourier transform is an algorithm for efficiently computing a the following Fourier sum:

$$F_k = \sum_{j=0}^{N-1} e^{2\pi i jk/N} f_j \tag{22}$$

where f_j is the complex data in time space (or in our case, position space) at the N equidistant mesh points. The values F is the Fourier transform F_k computed at points in frequency space (or k-space). Note that equation (22) is equivalent to one dimension of equation (21).

The machine evaluation of complex exponentials is costly. We greatly speed up the numerical computation of F_k by defining the following prefactor:

$$W_N \equiv e^{2\pi i/N} \tag{23}$$

The sum in equation (22) then becomes

$$F_k = \sum_{j=0}^{N-1} W_N^{jk} f_j \tag{24}$$

Even with this simplification, the sum has to be performed for every value of k—there are N values—and each sum has length N; the direct sum therefore requires of $O(N^2)$ operations.

With the *Fast Fourier Transform* (FFT) algorithm, the sum can be performed with $O(N\log_2 N)$ operations. The algorithm is based on the simple Danielson-Lanczos theorem (e.g. [7]). It states that a Fourier sum of length N can be computed as the sum of two Fourier transforms of length $N/2$ and is based on the following observation:

$$\begin{aligned} F_k &= \sum_{j=0}^{N-1} e^{2\pi i jk/N} f_j \\ &= \sum_{j=0}^{N/2-1} e^{2\pi i(2j)k/N} f_{2j} + \sum_{j=0}^{N/2-1} e^{2\pi i(2j+1)k/N} f_{2j+1} \\ &= \sum_{j=0}^{N/2-1} e^{2\pi i jk/(N/2)} f_{2j} + W_N^k \sum_{j=0}^{N/2-1} e^{2\pi i jk/(N/2)} f_{2j+1} \\ &= F_k^{even} + W_N^k F_k^{odd}. \end{aligned} \tag{25}$$

In words, equation (22) can be split into sums involving the even and odd grid points in f. Therefore, a Fourier transform of length

N can be performed as two Fourier transforms and of lengths $N/2$ on the even and odd elements of f. The two subtransforms can then be combined to give the desired Fourier transform F. Note that in equation (25), F_k^{even} and F_k^{odd} are periodic in k with length $N/2$ and the index $k \in [0, N-1]$.

Applying the Danielson-Lanczos theorem recursively leads to the Cooley-Tukey radix two FFT algorithm. Other schemes are possible, for example, in a radix four scheme, the Fourier transform would be partitioned into 4 subtransforms of length $N/2$.

2.4.2 Bit reversal

Efficient recursive application of the Danielson-Lanczos theorem suggests reordering the data to execute the Danielson-Lanczos algorithm with a minimum of address arithmetic. To motivate this, considering recursively applying the algorithm and subdividing the data until each transform sum only has one element. Clearly, this *one-point* transforms will be one of the input values f. To see which one, label even subdivisions of F by superscript 0 and odd subdivisions of F by superscript 1 at each level. For example, we have $F^{0110010\ldots1001} = f_l$ for some l. Note that in each subdivision, we are subdividing using lowest order binary bit in the index of f and then dividing by 2. Therefore, the value of l is represented by the bit reversal of the binary number in the superscript of F!

The FFT scheme, then has two steps:

1. Rearranging the input data into bit reversed order;
2. Iterative application of the Danielson-Lanczos algorithm to compute the transform.

Note that each transform value F has required $O(N)$ operations and there are $O(\log_2 N)$ possible combinations.

2.5 The general particle-field method

Given an n-body phase space, there is another way to use Fourier expansions of the Poisson equation to evaluate the potential (and force) from the density without binning the data or using the FFT. Note that the density profile of the n particles may be written as:

$$\rho(x,y,z) = \sum_{i=1}^{n} \delta(x-x_i)\delta(y-y_i)\delta(z-z_i) \tag{26}$$

where $\delta(u)$ is the Dirac delta function. Now use this expression for ρ in equation (19) without discretization. The coefficients are then

$$\rho_{lmn} = \frac{1}{L^3}\sum_{i=1}^{n} e^{-i2\pi l x_i/L} e^{-i2\pi m y_i/L} e^{-i2\pi n z_i/L} \rho(x_i,y_i,z_i) \tag{27}$$

and we are done! Because we use n particles to determine field coefficients and then use the field to generate particle motion, this approach is often referred to as a *particle-field* method. This scheme is $O(n)$ in the particle number. For very large n this is less efficient than an FFT scheme. However, the real power of the *particle-field method* is its applicability to non-Cartesian geometries for which no FFT exists.

For example, consider (again) a spherical stellar system. The Poisson equation separates, as you know, in spherical coordinates. The three resulting equations have the Sturm-Liouville form and result in trigonometric functions in the ϕ dimension, $e^{im\phi}$, associated Legendre polynomials in the θ direction, $P_l^m(\cos\theta)$, and Bessel functions in the radial direction, $q_{nl}J_{l+1/2}(\alpha_n r/R)$. (The coefficient q_{nl} normalizes the Bessel function.) The first two functions are combined to form the spherical harmonic, $Y_{lm}(\theta,\phi)$. The α_n follow from defining physical boundary conditions which assume that the distribution vanishes outside of some radius R. The desired coefficients which determine the gravitational potential,

$$V(r,\theta,\phi) = \sum_{l=0}^{l_{max}} \sum_{m=-l}^{l} \sum_{n=1}^{n_{max}} Y_{lm}(\theta_i,\phi_i) q_{nl} J_{l+1/2}(\alpha_n r/R), \tag{28}$$

are

$$c_{nlm} = -4\pi G \sum_{i=1}^{n} Y_{lm}(\theta_i,\phi_i) q_{nl} J_{l+1/2}(\alpha_n r_i/R). \tag{29}$$

See [5] for details on this particular set of functions for the sphere. NB: this set is not the best one to use in general but is easy to describe. In case you are interested in applying this algorithm, there are a variety of other potential-density function pairs which are all used the same way [1, 1, 6, 4, 8]. The final reference by yours truly describes a numerical procedure for generating potential-density pairs ideal for nearly any distribution in a separable coordinate system.

References

[1] M. Clutton-Brock. The gravitational field of three-dimensional galaxies. *Astrophys. Space. Sci.*, 23:55–69, July 1973.

[2] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Interscience, New York, 1953.

[3] G. Dahlquist and A. Bjork. *Numerical Methods*. Prentice-Hall, Englewood Cliffs, 1974.

[4] David J. D. Earn and J. A. Sellwood. The optimal n-body method for stability studies of galaxies. *ApJ*, 451:533, 1995.

[5] A. M. Fridman and V. L. Polyachenko. *Physics of Gravitating Systems*, volume 2, page 282. Springer-Verlag, New York, 1984.

[6] Lars Hernquist and Jeremiah P. Ostriker. A self-consistent field method for galactic dynamics. *ApJ*, 386:375, 1992.

[7] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1988.

[8] Martin D. Weinberg. An adaptive algorithm for n-body field expansions. *AJ*, 117:629, 1997. See also `astro-ph/9805357`.