Project due Wednesday Dec 10th

Consider the problem of computing integrals in high dimensions. Standard methods are based on one-dimensional integration methods which typically have a computational cost that grows exponentially with dimension. On the other hand, Monte Carlo integration methods use random sampling. The rate of convergence is relatively slow with the number of evaluations $\mathcal{O}(1/\sqrt{n})$ but does not grow exponentially with the dimension.

We will compare methods for computing integrals over the unit ball $B := \{\boldsymbol{x} \in \mathbb{R}^3 \mid \|\boldsymbol{x}\|_2 \leq 1\}$ in $\mathbb{R}^3$. A classical approach to integration over a unit ball is to use spherical polar coordinates:

$$\begin{aligned} x &= r\,\cos\theta\,\cos\phi, \\ y &= r\,\cos\theta\,\sin\phi, \\ z &= r\,\sin\theta, \end{aligned}$$

with ranges $r \geq 0$, $-\pi/2 \leq \theta \leq +\pi/2$, and $0 \leq \phi \leq 2\pi$. Here θ is the latitude and ϕ is the longitude. The multivariate change of variable formula for this case is

$$\iiint f(x,y,z)\,dx\,dy\,dz = \iiint f(r\cos\theta\cos\phi, r\cos\theta\sin\phi, r\sin\theta)\,r^2\,\cos\theta\,dr\,d\theta\,d\phi.$$

Nested one-dimensional integration rules can be applied to these three-dimensional integrals.

On the other hand, using a Monte-Carlo algorithm, we need methods to generate samples uniformly from the unit ball B. If $\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3, \ldots$ are independent samples generated this way, then

$$\iiint_B f(\boldsymbol{x})\,d\boldsymbol{x} \approx \frac{\mathrm{vol}(B)}{N} \sum_{i=1}^{N} f(\boldsymbol{X}_i).$$

The problem is that the error is typically $\mathcal{O}(1/\sqrt{N})$, meaning that N has to be very large to obtain an accurate estimate of the true integral. To understand the behavior of these estimates, we need to use the tools of statistics, especially expectations and variances of random variables.

Task 1: Show that the variance of $f(\boldsymbol{X})$ where $\boldsymbol{X} \sim \pi$ is a random variable with probability distribution (measure) π is given by $\mathbb{V}\mathrm{ar}\,[f(\boldsymbol{X})] = \int (f(\boldsymbol{x}) - \overline{f})^2 d\pi(\boldsymbol{x})$

with $\overline{f} = \mathbb{E}\left[f(\boldsymbol{X})\right] = \int f(\boldsymbol{x})\, d\pi(\boldsymbol{x})$. Combine this with the uniform probability distribution over B given by $d\pi(\boldsymbol{x}) = \chi_B(\boldsymbol{x})\, d\boldsymbol{x}/\text{vol}(B)$, where $\chi_B(\boldsymbol{z}) = 1$ if $\boldsymbol{z} \in B$ and zero otherwise, to give an integral formula for $\mathbb{V}\text{ar}\left[f(\boldsymbol{X})\right]$. Using results from statistics to write the variance of

$$S_N := \frac{\text{vol}(B)}{N} \sum_{i=1}^{N} f(\boldsymbol{X}_i)$$

in terms of N and the variance of $f(\boldsymbol{X})$. Also check that $\mathbb{E}\left[S_N\right] = \int_B f(\boldsymbol{x})\, d\boldsymbol{x}$.

Task 2: Implement an integration method for computing the integral $\int\int\int_B f(r\cos\theta\cos\phi, r\cos\theta\sin\phi,$ using spherical polar coordinates and standard integration methods. Use m points for each of r, θ, and ϕ, and therefore using a total of m^3 integration points.

Task 3: Implement the Monte-Carlo method for integrating a function over B, the unit ball in $\mathbb{R}^3$. To sample uniformly over B, first generate (pseudo-)random uniform points over $[-1, +1]^3$ by means of a standard uniform random number generator, and then reject the sample (that is, re-sample) whenever the sample point $\boldsymbol{X}$ is not in B. Estimate the rejection rate by computing $1 - \text{vol}(B)/\text{vol}([-1, +1]^3)$. [**Note:** The rejection rate gets much worse in high dimensions.]

Task 4: Plot the error in each method against N, the number of function evaluations, for the function $f(x, y, z) = (1 + x^2 + y^2)e^z - x/(1 + z^2)$. To estimate the true value, use a classical integration technique with large m. Since the error of the Monte-Carlo method is random, repeat the process, say, 10 times, and average the size of the error over these 10 trials.

Task 5: For each random sample $\boldsymbol{X} \in B$, instead of only evaluating $f(\boldsymbol{X})$, create a method that uses

$$\frac{1}{8} \sum_{j=1}^{8} f(Q_i \boldsymbol{X})$$

where the Q_i are the 3×3 diagonal matrices with ± 1 on the diagonal. Repeat Task 4 with this new method. Plot the error estimate against the number of function evaluations. Show that this method is exact whenever f is linear, and if f is bilinear as in $f(x, y, z) = xy$.

Task 6: Write up your methods, codes, and results in a report (code can be submitted separately) that includes your justifications. Provide two kinds of documentation for your code(s):

1. Documentation for a user to understand how to use the code (user documentation); and

2. Documentation explaining the internal operation of the code so that others can see how it works, and how it could be modified (developer documentation).

Otherwise, use whatever methods are appropriate (English prose/mathematics/code or pseudo-code/graphs or graphics) to explain your methods and results. You can use PDF, a word processor such as Microsoft Word, and/or Jupyter notebooks, to present your work and your results. (My favorite mathematical writing software is LyX available from `www.lyx.org`.)