MACM 316 – Computing Assignment 6

- Read the *Guidelines for Assignments* first.
- Submit a one-page PDF report to Canvas and upload you Matlab scripts (as m-files). Do not use any other file formats.
- Keep in mind that Canvas discussions are open forums.
- You must acknowledge any collaborations/assistance from colleagues, TAs, instructors etc.

Gambling your way to high dimensions: Monte Carlo integration

As we have/will see in class, numerical integration in one dimension is relatively straightforward. However, numerical integration becomes substantially more challenging in higher dimensions. Integrals of high-dimensional functions arise is all sorts of applications, including computational finance, computational physics and uncertainty quantification.

A typical problem in high-dimensional integration is to compute the volume of a shape. Suppose that Ω is a shape contained in the unit hypercube $[-1,1]^d$. Then

$$\mathrm{Vol}(\Omega) = \int_{-1}^{1} \int_{-1}^{1} \cdots \int_{-1}^{1} f(\underline{x}) \, \mathrm{d}\underline{x},$$

where f is a function defined by $f(\underline{x}) = 1$ if $\underline{x} \in \Omega$ and $f(\underline{x}) = 0$ otherwise.

One way to compute this integral is to construct a set of points in the hypercube $[-1,1]^d$, say $\underline{x}^{(1)}, \ldots, \underline{x}^{(N)}$, and count the number of 'hits', i.e. the number of these points which fall within Ω. Note that this is the same as approximating the integral by the *quadrature rule*

$$\int_{-1}^{1} \int_{-1}^{1} \cdots \int_{-1}^{1} f(\underline{x}) \, \mathrm{d}\underline{x} \approx \frac{2^d}{N} \sum_{n=1}^{N} f(\underline{x}^{(n)}). \tag{1}$$

The factor 2^d is a normalization factor. This raises the question: what is a good choice of points $X = \{\underline{x}^{(1)}, \ldots, \underline{x}^{(N)}\}$? Your goal in this assignment is to investigate this question. Specifically, you will consider the following choices:

(i) A set of N equally-spaced points.
(ii) A set of N points chosen randomly from $(-1,1)^d$.

The latter is known as *Monte Carlo* integration. The Matlab function *GeneratePoints.m* creates these points for you. Note that its inputs are the number of points N, the dimension d, and a variable `mode`, which should be set equal to 0 for case (i) and 1 for case (ii). The output is an $d \times N$ array, where the n^{th} column is the point $\underline{x}^{(n)}$.

Write a code that implements the quadrature rule (1) to approximate the volume of a hypersphere, i.e.

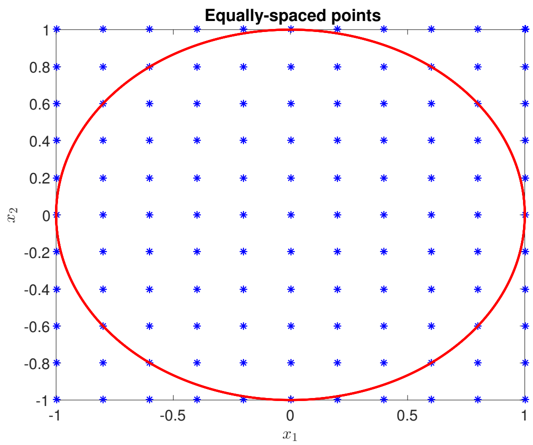
$$\Omega = \left\{ \underline{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d : x_1^2 + x_2^2 + \ldots + x_d^2 \leq 1 \right\}.$$

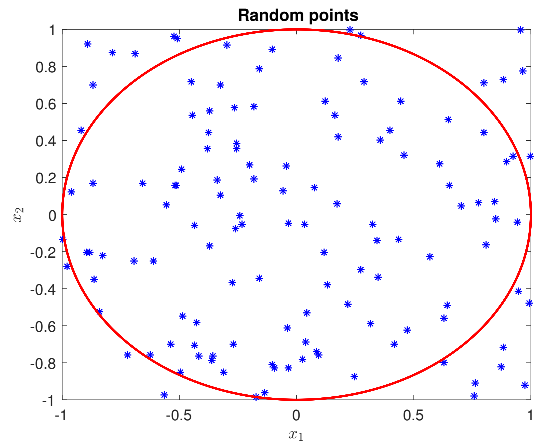
Compute the error of the approximations you get using the points (i) and (ii). Do this for a range of different N and plot the error versus N in a loglog scale. For the random points (ii) you should average over a reasonable number of trials.

Test several different values of the dimension d. You might find it best to produce a separate figure for each different value of d you use. How does the error depend on (a) the number of points

N, (b) the dimension d, and (c) the choice of points? Which points would you recommend using in practice? Present your conclusions in your report along with your figures.

Hint 1: The figures below are intended to help you understand how this procedure works. It shows the approximation of the area of a circle (i.e. the $d = 2$ case) using the two types of points. The value of $N = 128$ is used.





- In Case (i) (the left figure) there are 81 points inside the circle, meaning the approximation (1) to the area is $2^2 \times 81/128 = 2.5312$.
- In Case (i) (the right figure) there are 92 points inside the circle, meaning the approximation (1) area is $2^2 \times 92/128 = 2.8750$.

Hint 2: In order to compute the error you need to know the exact volume of a hypersphere. This is given by

$$\mathrm{Vol}(\Omega) = \frac{\pi^{d/2}}{\Gamma(d/2+1)}.$$

Γ is the so-called gamma function, which can be implemented in Matlab by the command `gamma(d/2+1)`.