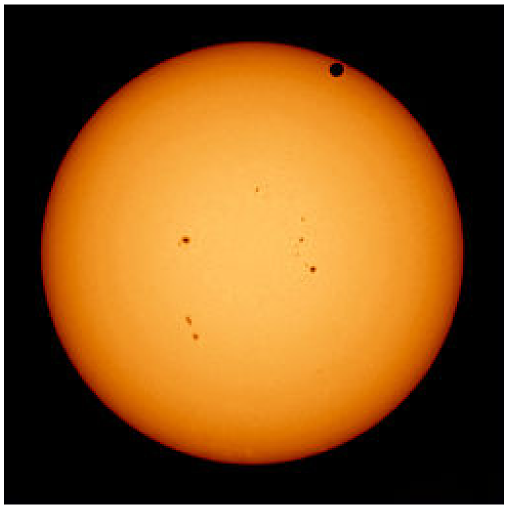

Figure 1: Limb darkening is observed for the Sun, where the edges have a lower luminosity than the centre. Credit: Wikipedia.

Computational Lab 10 : Random numbers, Monte Carlo integration

Due November 23rd, 2019 (@ 17:00)

- Read this document and do its suggested readings to help with the pre-labs and labs.
- This lab's topics revolve around computing solutions using random numbers and Monte Carlo techniques.
- Ask questions if you don't understand something in this background material: maybe we can explain things better, or maybe there are typos.
- Whether or not you are asked to hand in pseudocode, you **need** to strategize and pseudocode **before** you start coding. Writing code should be your last step, not your first step.
- Test your code as you go, **not** when it is finished. The easiest way to test code is with `print('')` statements. Print out values that you set or calculate to make sure they are what you think they are.
- Practice modularity. It is the concept of breaking up your code into pieces that as independent as possible form each other. That way, if anything goes wrong, you can test each piece independently.

Physics Background

Limb-darkening in plane-parallel, grey atmospheres (for Q2): Limb darkening describes the fact that a (normal) star seems to be darker at the edge than at its center. This can be clearly observed for the Sun (see Fig. 1). If the direction cosine (i.e., the cosine of angle between the radius vector and photon direction) is denoted by μ, one can approximate the angular dependency of the specific intensity by the expression

$$I(\mu) \approx I(\mu = 1)(0.4 + 0.6\mu) \tag{1}$$

where I_1 is the specific intensity for the radial direction, $\mu = 1$, i.e., the intensity observed at the center of a star. For $\mu = 0$, on the other hand, the angle between radial and photonâĂŹs direction is 90° (this condition is met at the limb of the star), and the corresponding region appears to be fainter, by a factor of roughly 0.4.
Equation 1 is an approximate solution of the equation of radiative transfer in 'plane-parallel symmetry', which assumes that the stellar photosphere is very thin compared to the stellar radius. For example, the solar photosphere is only a few hundred kilometers thick, while the Sun's radius is around 700 000 km). In addition, the absorption and emission processes assumed to be frequency independent, i.e. 'grey'. Note: Equation 1 has nothing to do with the presence of any temperature stratification (though it can be influenced by it), but is the consequence of a large number of absorption and emission processes in an atmosphere of finite optical depth, which describes how much absorption occurs when light travels through an absorbing medium.

Computational Background

Spherical polar to cartesian coordinates: Assuming spherical coordinates of radius r, inclination θ, azimuth ϕ, with suitable limits, the cartesian coordinates can be determined by

$$\begin{aligned} x &= r\sin\theta\cos\phi, & (2)\\ y &= r\sin\theta\sin\phi, & (3)\\ z &= r\cos\theta. & (4)\end{aligned}$$

Bit packed data: The Eath map for question 1 is stored in a file called 'Earth.npy'. This map is originally a 1/30 degree greyscale image showing the topographic height. This was converted into a binary file with of land/ocean points and compressed to save space using a the 'packbits' function — converting 8 logical values into 1 byte of data. To read the file you can used

```
import numpy as np
data = np.unpackbits(np.load("Earth.npy")).reshape(2160, 4320)
```

where the 2160 is the number of latitude points (180 times 30), and 4320 the number of longitude points. *After you load the data, plot it to make sure it's usuable data, and to figure out which value is land and which is ocean.*3 functions for simulating photon scattering (for Q2):

```
def get_tau_step():
  """calculate how far a photon travels before it gets scattered .
    Input : tau - optical depth of the atmosphere
    Output: optical depth traveled"""
    delta_tau = -np.log(np.random.random())
    return delta_tau

def emit_photon( tau_max ) :
  """Emit a photon from the stellar core.
    Input : tau max - max optical depth
    Output :
      tau: optical depth at which the photon is created
      mu: directional cosine of the photon emitted """
  tau = tau_max
  delta_tau = get_tau_step()
  mu = np.random.random()
  return tau-delta_tau*mu, mu

def scatter_photon(tau):
  """Scatter a photon .
    Input : tau âĹŠ optical depth of the atmosphere
    Output :
      tau: new optical depth
      mu: directional cosine of the photon scattered"""
  delta_tau = get_tau_step()
  # sample mu uniformly from -1 to 1
  mu = 2*np.random.random()-1
  tau = tau + delta_tau*mu
  return tau, mu
```

Questions

1. **[30%] A random point on the surface of the Earth (based on Newman 10.12** Suppose you wish to calculate the land area on the Earth, but don't have the resources to map every location on the Earth. You could instead choose a random point on the surface of the Earth and determine if that point was land or ocean. If you know the total area of the Earth then you can calculate approximately the land area.

   To calculate this value correctly, you want to choose a location on the planet at random such that all points are equally likely to be chosen.

   Recall that in spherical coordinates θ, ϕ the element of solid angle is $\sin\theta\, d\theta\, d\phi$, and the total solid angle in a whole sphere is 4π. Hence the probability of our point falling in a particular element is

$$p(\theta,\phi)\, d\theta\, d\phi = \frac{\sin\theta\, d\theta\, d\phi}{4\pi}.$$

   We can break this up into its θ part and its ϕ part thus:

$$p(\theta,\phi)\, d\theta\, d\phi = \frac{\sin\theta\, d\theta}{2} \times \frac{d\phi}{2\pi} = p(\theta)\, d\theta \times p(\phi)\, d\phi.$$

   (a) What are the ranges of the variables θ and ϕ? Verify that the two distributions $p(\theta)$ and $p(\phi)$ are correctly normalized—they integrate to 1 over the appropriate ranges.

   SUBMIT THE RANGES OF THE TWO VARIABLES.

   (b) Find formulas for generating angles θ and ϕ drawn from the distributions $p(\theta)$ and $p(\phi)$. Write a function that generates a random θ and ϕ using the formulas you worked out.

   NOTHING TO SUBMIT YET

   (c) Use your program to generate 500 such random points and plot those points either on a two dimensional (latitude vs longitude) plot or a 3D cartesian plot.

   SUBMIT THE PLOT

   (d) The calculation of land fraction can be numerically calculated without too much difficulty. You can integrate the area of a sphere using the solid angle equation above (whose integral should be 4π if you set radius to 1). Calculate, by numerical integation, the land fraction using the provided Earth map.

   The file 'Earth.npy' contains a binary bitpacked dataset of the Earth that (approximately) identifies water and land points. Alternate (higher and lower resolution) datasets can be found online.

   SUBMIT YOUR WRITTEN ANSWER

   (e) Using your random location generator. Generate N random locations on the planet and calculate the fraction of the planet that is covered by land (as opposed to ocean or sea–ice). One way of doing this is to randomly sample a location on your map and determine whether it was land or not, and calculate the ratio of land points you identified compared to the total number of samples.

   Calculate, using random sampling, the land fraction using the same Earth map as above. How many samples do you need to get 10% accuracy, 0.1% accuracy, 0.001% accuracy

   SUBMIT YOUR CODE AND WRITTEN ANSWER

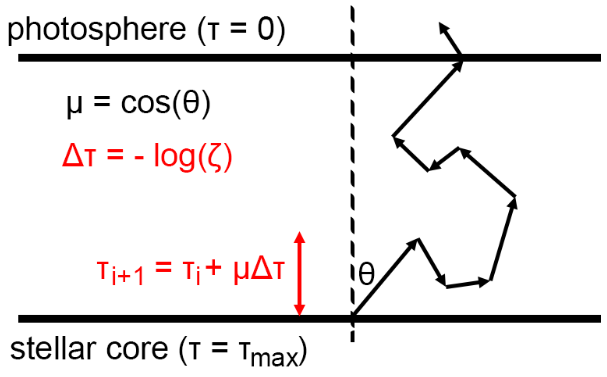

Figure 2: Random walk of a photon from the stellar core to the photosphere. The photon is emitted from the stellar core with some angle θ and scatters at some new optical depth τ_{i+1} for step i, and continues onward until one of the following happens: $\tau < 0$ where the photon has left the photosphere, or $\tau \geq \tau_{max}$ where the photon has scattered back into the stellar core. In the latter case, we emit a new photon to replace it. Since the probability that a photon travels an optical depth of τ without an interaction is $e^{-\tau}$, the optical depth is sampled from $\tau = -\log \zeta$, where ζ is a random number uniformaly sampled from 0 to 1.

2. **[40%] Limb darkening of star – Monte Carlo:** In order to avoid almost all subtleties of radiative transfer and corresponding approximate solutions, in this exercise, we will perform a Monte Carlo simulation to confirm the result of Eq. 1. The simulation will emit a photon from the stellar core and follow it until it leaves the atmosphere.

   (a) The concept of radiation transfer is simple: a photon is emitted, it travels a distance, and it scatters. To determine the final scattering angle of the photon (the angle at which it leaves the photosphere), we draw samples of optical depth and scattering angle from probability distributions. Thus, Monte Carlo methods are a good way to simulate this. Figure 2 shows a possible path of a photon, where a plane-parallel atmosphere is assumed. The scattering angle θ is the angle between the radial and photon direction, and for this problem, it is more convenient to express it as $\cos\theta$, which we will call μ.

   Using the three given functions as building blocks, write a function that, taking $\tau_{\max}$ as its input, emits a photon from the stellar core and follows its path until it leaves the atmosphere. At each step, update the optical depth of the photon and follow the photons until they have left the atmosphere (i.e. $\tau < 0$). Have your program print out the μ at last scattering and how many times the photon scattered. If a photon scatters back into the core, have your program release a new photon at the inner boundary.

   SUBMIT YOUR CODE.

   (b) Using your function, test it for 10^5 photons with an optical depth of $\tau_{\max} = 10$, and save the final μ values of each photon. Make a histogram plot of $N(\mu)/N(1)$ as a function of μ using `n_bins=20` as the number of bins.

   In our simulation, we have calculated the number of photons leaving the atmosphere with respect to a surface perpendicular to the radial direction. This number is proportional to the specific intensity weighted with the projection angle μ, since the specific intensity $I(\mu)$ is defined with respect to unit projected area:

$$N(\mu)d\mu \propto I(\mu)\mu d\mu. \tag{5}$$

   Therefore, we can take specific intensity $I(\mu) \propto \frac{N(\mu)}{\mu}$, which is obtained from the number of photons divided by the appropriate average of μ, i.e., centred at the middle of the corresponding bin.

   Write a function to calculate $I(\mu)/I(1)$ and compare it with the analytic expression from Equation 1. Use 10^5 photons and $\tau_{\max} = 10$ as before, and derive the limb-darkening coefficients from a linear regression to your results.

You may use `scipy.optimize.curve_fit` to determine the coefficients. State the coefficients (either on your plot or in your report) and plot both the analytic fit and the best fit from your results. Relate your results back to limb-darkening (e.g. where is specific intensity greatest and how does affect what an observer would see?).

SUBMIT YOUR CODE, PLOTS, AND EXPLANATORY NOTES.

(c) Limiting case of $\tau_{\max} \ll 1$. To convince yourself that limb-darkening is the consequence of a multitude of scattering effects, reduce $\tau_{\max}$ to a very small value ($\tau = 0.0001$). Which angular distribution do you expect for the specific intensity? Does your simulation verify your expectation? Make the same plots as in the previous part and use them to explain your results.

SUBMIT YOUR CODE, PLOTS, AND EXPLANATORY NOTES.

3. **[30%] Importance Sampling**

(a) The integral

$$\int_0^1 \frac{x^{-1/2}dx}{1+e^x}$$

is an integral whose integrand has a divergence. Section 10.2.3 discusses using importance sampling to evaluate this integral. To learn about the regular mean value method and the importance sampling methods, evaluate the integral using both methods with 10,000 sample points, then repeat this calculation 100 times for each method. For the importance sampling approach, use the weighting function $w(x) = x^{-1/2}$, which removes the singularity. You will need to determine the transformation needed to non-uniformly sample your region. The probability distribution you will be drawing from is

$$p(x) = \frac{1}{2\sqrt{x}} \tag{6}$$

(Pages 458-459 of the text explain how to find the transformation for a given probability distribution, to help you understand where this $p(x)$ comes from.) Make a histogram of the resulting integral values using 10 bins from 0.80 to 0.88. For an array of values (say it's called `I`), you do this with the command:

```
import pylab as plt
#...
plt.hist(I,10,range=[0.8, 0.88])
plt.show()
```

Comment on what your histograms from the mean value method and the importance sampling method tell you about each method.

SUBMIT YOUR CODE, PLOT, AND WRITTEN ANSWERS

(b) Importance sampling also helps evaluate rapidly varying integrands even if they aren't singular. For example, consider the integral

$$\int_0^{10} \cos(10(x-5))\, e^{(x-5)^4}\, dx \tag{7}$$

The integrand in this equation is sharply peaked near $x = 5$, and we can easily find its exact value. If we calculate it using points uniformly distributed on the interval $0 <= x <= 10$ we will be using a lot of points that don't contribute to the expectation value. To sample more points near $x = 5$ in order to increase the precision of the sampling method, choose an importance function that is Gaussian of the form

$$w(x) = \frac{1}{\sqrt{2\pi}} e^{-(x-5)^2/2}$$

that will lead you to draw from the normal distribution

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-(x-5)^2/2}$$

with mean of 5 and standard deviation of 1. You can use `numpy.random.normal` to do this.

Repeat part a using the mean value and importance sampling methods for this integral, and comment on the results. Try reducing the number of sample points from 10,000 to 100

SUBMIT YOUR CODE, PLOTS, WRITTEN ANSWERS.

Question:	1	2	3	Total
Points:	30	40	30	100