SESG6025 Coursework 2012: Numerical Integration

Due: Friday 7th December 2012 (12 midday). 20% weighting. There are a total of 100 marks

Aims: This coursework explores numerical integration using the trapezium rule and a Monte Carlo based method.

Objectives: Develop and use a numerical integration routine and write a Monte-Carlo integration routine.

Requirements and hand-in: You should email two files (together as attachments to a single email) to sesg6025@soton.ac.uk with the email subject "**Coursework**"

Files to email in: (1) The filename for the code (NOT an executable) must be "**integ.c**" and

(2) the figures in a single pdf "**integ.pdf**"

You will need to down load the file "random.txt" containing a list of high quality random numbers in the range 0 to 1 from:

http://www.soton.ac.uk/~sjc/teaching/sesg6025/random.txt

The file "integ.c", when compiled with an ANSI C compiler and executed, should give the results required for each part of the question in turn, clearly labelled. It should not require any input and should NOT, for example, pause at the end to wait for key.

For C Compiler information see e.g.: http://www.southampton.ac.uk/~sesg6025/toolsc.html

Note that for each question, what you need to produce is labelled thus:

"Figure X" means a figure labelled in the PDF,

"Output X" means a clearly labelled output to screen from the code, and

"Code X" means a clearly labelled piece of code which implements that part of the question.

Numerical Integration

a) **Plot** the function e^x using 10 equally spaced points between 0 and 4. Show circles for the points and join up the points with a straight line.

[Figure 1] [***5 marks***]

b) Write code to implement the trapezium rule and **estimate the integral** $\int_0^4 e^x$ i.e. for the function e^x using 10 points over the range 0 to 4.

[Code 1; Output 1] [***15 marks***]

c) Display the **exact** answer and add comments in your code as to how you have worked this out

[Code 2; Output 2] [***5 marks***]

d) **Evaluate** the integral as in (b) using N = 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, and 200 equally spaced points over the range and **plot** the absolute percentage value of the error = $\frac{\left|I_{exact} - I_{trapezium}\right|}{I_{exact}} \times 100\%$ as a function of the number of points used, N.

[Code 3; Figure 2] [***15 marks***]

CONTINUED OVER…

Monte Carlo integration

It is possible to estimate an integral using the following formula:

$$I = \int_{x=a}^{b} f(x)\,\mathrm{d}x \approx (b-a)\langle f(x_i)\rangle \pm \frac{(b-a)}{\sqrt{N}}\sqrt{\langle f(x_i)^2\rangle - \langle f(x_i)\rangle^2}\,,$$

where the x_i are chosen randomly between a and b, N is the number of sample points, and:

$\langle f(x_i)\rangle$ = `mean of f` and $\sqrt{\langle f(x_i)^2\rangle - \langle f(x_i)\rangle^2}$ = `standard deviation of f`.

e) Write a piece of code that will **load** in the list of random numbers from the file "random.txt" which can be found at: http://www.soton.ac.uk/~sjc/teaching/sesg6025/random.txt and display the 4444th number. The code should assume that the file is present in the directory that the executable runs from. You may find the link http://www.southampton.ac.uk/~sesg6025/faq.html useful when writing this code.

[Code 4 Output 3] [***5 marks***]

f) Write a piece of code that will calculate the **mean** of the list of numbers in the "random.txt" and display the mean.

[Code 5; Output 4] [***10 marks***]

g) Write a piece of code that will calculate the **standard deviation** of the list of numbers in the "random.txt" and display the standard deviation.

[Code 6; Output 5] [***10 marks***]

h) Using the Monte-Carlo method and, for x_i, random numbers in the range 0 to 4, **estimate the integral** $\int_0^4 e^x$ (as in (b) above) and give the one standard deviation error bound for it (this is just the ± part). Produce an appropriate figure showing how the error decreases as you use more random numbers

Hint: To scale random numbers in the range 0 to 1 to be in the range a to b, you should use `(b - a)* [numbers from random.txt] + a`, where $b > a$.

[Code 7; Output 6; Figure 3] [***35 marks***]

Other Notes

1) The file "random.txt" was created using this Matlab code (Matlab R2010a and R2010b)

```
%Get the current random number stream
s=RandStream('mt19937ar');
RandStream.setDefaultStream(s);
% Reset it
reset(s);
a=rand(50000,1);
% Native text file version in Matlab
save 'random.txt' -ascii -double a
% Analogous c-style version in Matlab
file_id=fopen('random2.txt','w');
fprintf(file_id,'%.16e\r\n',a);
fclose(file_id);
```

2) Why might you ever want to use this method? For multi-dimensional integrals in d dimensions, the error for the trapezium rule falls off as $N^{(-2/d)}$, where N is the number of points used/ function evaluations made. However, the $N^{(-1/2)}$ dependence of the error for the Monte Carlo integration is independent of the number of dimensions; hence for large d, the Monte Carlo method is actually more accurate for a given number of function evaluations. For Simpson's rule the error goes as $N^{(-4/d)}$ in d dimensions. By using quasi-random sequences it is possible to arrange for the error to drop of as $N^{(-1)}$ independent of d.

Please ask if you need help.Prof Simon Cox, sjc@soton.ac.uk. Building 25/2037 Phone Ext 23116.