PC5215 Numerical Recipes with Applications

Midterm test, close book, 6 Oct 2023

1. The Lagrange's polynomial $l_i(x)$ of degree N-1 constructed from a list of points $(x_1, x_2, \cdots, x_N)$ has the property that $l_i(x)$ is 0 if $x = x_j$ for $j \neq i$ and equals 1 when $x = x_i$. Define a Python function which can be used as

   lagrangle(x, i, x_n_list)

   that returns the value of $l_i(x)$, where x is the independent variable, the integer i = 1,2, …, is the i-th polynomial, and x_n_list is the list of $(x_1, x_2, \cdots, x_N)$.

   Ans:

```
def lagrangle(x,i,x_n_list):
    L=1.0
    for j in range(len(x_n_list)):
        if(j!=(i-1)):
            L*= (x-x_n_list[j])/(x_n_list[i-1]-x_n_list[j])
    return L
```

   The integer i need offset by 1 as i takes from 1, 2, …, N, but Python prefers starting list index from 0.

2. Consider the IEEE 754 representation of 32-bit float point numbers (single precision). It has 32 bits, with the highest bit for the sign, 8 bits for the biased exponent, and the rest as the fractional part.
   a. What value does this bit pattern in hexadecimal notation represent?
      3E80 0000
   b. What is the bit pattern for a decimal number $\pi \approx 3.14159265$?

   a) 0.25
   b) 4049 0FDB in hex notation.

3. Define the meaning of computational (time) complexity. And explain the computational complexity based on your understanding of the methods or algorithms, in terms of $O(N^k)$ here k is usually an integer.
   a. Compute the determinant of an $N \times N$ matrix, using the Laplace expansion.
   b. Solve $Ax = b$ where A is $N \times N$ and x and b are vectors of length N.
   c. Neville's algorithm for N-point interpolation.

d. Your resistor network problem of lab 1 of $N \times N$ squares.

a) $N!$
b) N^3
c) N^2
d) N^6

The definition of computational complexity is the asymptotic steps needed for a problem of size N for sufficiently large N.

4. The integration formulas are usually constructed by requiring that polynomials can be integrated exactly up to the highest order possible. Suppose only one point of function value is given (or can be computed) in an integration interval $[0, h]$, what is the most accurate method you can give? Give the method, and work out the error it has.

$\int_0^h f(x)dx = hf(\frac{h}{2})$. Error $O(f''h^3)$. Because this is just a Gaussian quadrature with one point, *N*=1. It integrates 1, x, exactly, and has errors for x^2 or higher polynomials. The orthogonal polynomials are $P_0 = 1, P_1 = x - h/2$. We might use $hf(0)$, or $hf(h)$, but these rectangle rules are less accurate as they works for 1 exactly, already have errors for x, the error is $O(h^2)$. The mid-point choice of x=h/2 is optimal.

1. In Monte Carlo method, we want to generate random numbers in the interval (0, ∞) with a probability density $p(x) = x\, e^{-x}$. This can be done with a transformation. Give a method for generating the random number x in terms of a uniformly distributed random number $\xi \in (0, 1)$.

We can try the inverse method. So, the cumulative distribution function $F(x) = \int_0^x p(x')dx' = 1 - e^{-x} - xe^{-x}$. We set $F(x)$ to be the random number ξ and solve for x. But we don't have an explicit analytic solution for the transcendental equation. A much better method is to use $x = -\ln(\xi_1 \xi_2)$, i.e., use the product of two independent, uniformly distributed random numbers and take the log. This can be shown to have the right distribution by computing $F(x) = \int_{e^{-x} < \xi_1 \xi_2} d\xi_1 d\xi_2$. I agree this is not elementary and tricky.

-- The end --