[QI;MP]
Basic numerical structures — Applications

Jérémie Gillet

Sunday, October 13th 2013

1 Matrix multiplication

Define three matrices A, B and C of respective dimensions $n_1 \times n_2$, $n_2 \times n_3$ and $n_1 \times n_3$ ($n_1 \neq n_2 \neq n_3$). Fill A and B with random numbers and calculate

$$C = A \cdot B,$$

with three different methods.

First method, calculate C with only `for` loops, using the formula

$$C_{ij} = \sum_{k=1}^{n_2} A_{ik} B_{kj}.$$

Second method, replace one loop with the `sum` function.

Last method, use the built-in MATLAB matrix multiplication.

Make the dimensions n_i large ($\sim 10^2$), make sure all three answers coincide and compare the computation times by using `tic...toc`.

2 Bubble sort

This is a very basic (and inefficient) sorting algorithm. We want to sort a list of elements from the smallest to the largest.

The algorithm is the following, we compare the first two elements, if the first one is bigger than the second one, we swap them. We them compare the second and third and do the same. After $N-1$ operations, the biggest number has 'bubbled' up to the right. Repeat the iteration all over again $N-1$ time and the list will be sorted.

You can make it more efficient by noting that after the first iteration, you do not need to test the last number (since it is guaranteed to be the largest), on the third iteration you do not need to check the last two, etc. You can also add a test variable, if during one whole iteration no swaps have been done, the list is sorted and you can stop.

3 Finding a root

First define the function

$$f(x) = x^4 - 4,$$

in a separate file and plot it for $x \in [-2, 2]$. As you will see, the function has two roots in the interval. Find one of them by using the following algorithm.

1. Input by hand the values an interval $[x_{\min}, x_{\max}]$ that include one (and only one) root. Make sure they do by checking if $f(x_{\min})f(x_{\max}) < 0$.

2. Calculate $x_0 = (x_{\min} + x_{\max})/2$.

3. if $f(x_0)$ is of the same sign as $f(x_{\min})$, the root is on the right of x_0 and you can set x_0 as your new $x_{\min}$. Otherwise the root is on the left of x_0 and you can set x_0 as your new $x_{\max}$. If $f(x_0) = 0$, you are done.

4. Repeat step 2 and 3 until the distance $x_{\max} - x_{\min}$ is smaller than ϵ ($\sim 10^{-8}$).

4 Basic statistic tools

Create a list of random values $x_1, \dots, x_N$ and calculate their mean and standard deviation with three methods.

First method, calculate the mean value μ and standard deviation σ with only `for` loops with the formulas

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad \sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}.$$

Second method, use the `sum` function.

Last method, use the built-in MATLAB functions `mean` and `std`. The `std` function is however using a slightly different definition of the standard deviation, research and fix this issue by using the command `help std`.

Make the list large ($\sim 10^8$), make sure all answers coincide and compare the computation times by using `tic...toc`.

5 Linear regression fit

The best linear curve $y = ax + b$ that goes through a set of points $\{(x_1, y_1), \dots, (x_N, y_N)\}$ (by the least square estimation) is given by

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \quad b = \bar{y} - a\bar{x},$$

with $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ et $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

Generate a set of 100 points $y_i = 2x_i + 20 + r_i$, where $i = 1, \dots, 100$, $r_i \in [-40, 40]$ is a random variable and $x_i = i + r'_i$ with $r_i \in [0, 1]$ another random variable. Plot those points along with the fitted curve $y = ax + b$.

6 Dice function

Define a function `dice(n1,n2,l)` which will send back a list of l random integers included between n_1 and n_2. Check the mean value and standard deviation of the list of integers with the code you wrote earlier, with a list big enough you should find something approaching

$$\mu = (n_1 + n_2)/2, \quad \sigma = \sqrt{(n_2 - n_1 + 1)/12}.$$