

Math 128a, Homework 3

due September 25, except for problem 5, which is due October 2.

1. We discussed the first of the two ideas in the FFT algorithm in class; the second one is the operation of *bit-reversal*. Fix n and $N = 2^n$, and let k be an integer, $0 \leq k \leq N - 1$. Expand k in binary: $k = \alpha_0 + \alpha_1 \cdot 2 + \alpha_2 \cdot 2^2 + \dots + \alpha_{n-1} \cdot 2^{n-1}$. The bit-reversal permutation ρ maps k to $\alpha_{n-1} + \alpha_{n-2} \cdot 2 + \dots + \alpha_1 \cdot 2^{n-2} + \alpha_0 \cdot 2^{n-1}$. It is easy to see that ρ is a bijection, in fact $\rho(\rho(k)) = k$ for all k in its domain. See the textbook for further details.

- (a) Let $N = 16$. Apply the operation of bit-reversal to $0 : 15$, and denote $g_i = f_{\rho(i)}$, $i = 0, 1, \dots, 15$. Verify that the FFT algorithm only combines the coefficients with consecutive indices: each of the 8 linear polynomials has coefficients determined by only one of the $\{\{g_0, g_1\}, \dots, \{g_{14}, g_{15}\}\}$, each of the 4 cubic polynomials has coefficients determined by only one of the consecutive 4-tuples, etc. That is, draw a diagram showing which coefficients of the polynomials of the m -th level are used in the calculation of which coefficients of the $(m+1)$ -st level, $m = 0, 1, 2, 3, 4$ (without calculating the coefficients explicitly).
- (b) Verify that the FFT algorithm gives the correct answer for $N = 8$. That is, start with $x_k = 2\pi k/8$ and f_0, f_1, \dots, f_7 , and obtain the coefficients of the interpolating trigonometric polynomial. Your write-up should include 4 8-tuples of quantities, the first one $\{f_{\rho(0)}, f_{\rho(1)}, \dots, f_{\rho(7)}\}$, the second one containing the coefficients of the linear polynomials, the third one containing the coefficients of the cubic polynomials, and the last one $\{\beta_0, \beta_1, \dots, \beta_7\}$ such that for $p(x) = \beta_0 + \beta_1 e^{ix} + \dots + \beta_7 e^{7ix}$, $p(x_k) = f_k$, $k = 0, 1, \dots, 7$.

2. Exercise 2.23.

3. In this problem you will compare different one-dimensional Matlab interpolation routines. Which routines are available depends on the version of Matlab, so make sure to note which one you are using. Use `interp1` with all the methods provided, and if necessary, also `interpft` and `pchip` from the previous assignment; also use your modification `pchip2`. For each of the functions below, draw the graph of the original function, and on the same plot (or on more than one if the plot becomes too cluttered) draw the interpolating functions. List the number of flops used by each program, and the estimate of the error, obtained by sampling both the given and the interpolating functions on a fine enough grid. For the last two functions, the error will always be infinite (why?), so instead list the estimated maximal value of the interpolating function. Discuss the quality and the speed of interpolation.

For all the functions below, the support abscissas are $x_k = k/10$, $k = 0, 1, \dots, 10$, and the support ordinates are $f_k = f(x_k)$ for the given function f .

(a) $f(x) = x^4$.

(b) $f(x) = \begin{cases} x^2 & \text{if } x \in [0, 0.5], \\ x & \text{if } x \in (0.5, 1]. \end{cases}$

(c) $f(x) = \begin{cases} x^2 & \text{if } x \in [0, 0.5], \\ x + 0.25 & \text{if } x \in (0.5, 1]. \end{cases}$

(d) $f(x) = \frac{1}{x-0.55}$.

(e) $f(x) = \frac{1}{(x-0.55)^2}$.

4. Exercise 2.29.

5. (Due October 2, a more detailed description to be given in the next assignment.) Implement Milne's rule for numerical quadrature. You can use the Matlab function implementing Simpson's rule as an example.