HOMEWORK 3

1 The Hadamard transform

We define the n-dimensional Hadamard transform on the set of functions $f : (\mathbb{Z}/2\mathbb{Z})^n \to \mathbb{C}$ as the operator[1]

$$T(f)(y) = \sum_{x \in (\mathbb{Z}/2\mathbb{Z})^n} f(x)(-1)^{\langle x,y \rangle},$$

where $\langle x, y \rangle = \sum_i x_i y_i$.

1. For $z \in (\mathbb{Z}/2\mathbb{Z})^n$, show that $T(T(f))(z) = 2^n f(z)$. What does this say about the inverse transform?

   To compute the Hadamard transform, we consider each function $f : (\mathbb{Z}/2\mathbb{Z})^n \to \mathbb{C}$ as a 2^n-dimensional vector, where the order is defined recursively as $(\mathbb{Z}/2\mathbb{Z})^n = (\mathbb{Z}/2\mathbb{Z})^{n-1} \times \{0\}, (\mathbb{Z}/2\mathbb{Z})^{n-1} \times \{1\}$. So, for example, if $n = 1$, then $f = [f(0)f(1)]^T$ and if $n = 2$, then $f = [f(00)f(10)f(01)f(11)]^T$.

2. Show that $T(f) = H_n f$, where H_n is called the *Hadamard matrix* of order n and is defined as $H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}$, for $n \geqslant 1$ and $H_0 = 1$.

3. Based on the previous matrix vector product formula, give a fast recursive algorithm for computing the Hadamard transform in $O(n2^n)$ operations over $\mathbb{C}$.

2 Logarithm and exponential

For polynomials $S, T \in \mathbb{K}[X]$ such that $S(0) = 0$ and $T(0) = 0$, we define

$$\exp_n(S(X)) = \sum_{k=0}^{n-1} \frac{S(X)^k}{k!} \bmod X^n$$

$$\log_n(1 + T(X)) = \sum_{k=1}^{n-1} (-1)^{k+1} \frac{T(X)^k}{k} \bmod X^n$$

1. Assume $A(0) = 0$, prove that

$$(A(X) + 1)^{-1} = \sum_{k=0}^{m-1} (-1)^k A(X)^k \mod X^m$$

[1] notice that $(-1)^x$ is well defined for $x \in \mathbb{Z}/2\mathbb{Z}$

2. Recall that $S(0) = 0$. Let $U_n(X) = S'(X)/(S(X)+1) = \sum_{k=0}^{n-2} u_k X^k \bmod X^{n-1}$ (note that $S(X)+1$ is invertible modulo X^n because $S(0)+1 \neq 0$). Prove that

$$\log_n(1+S(X)) = \sum_{k=1}^{n-1} u_{k-1} \frac{X^k}{k} \bmod X^n$$

(Hint: use question 1 and derive both sides of the equation).

3. Deduce a quasi-linear time algorithm to compute $\log_n(S(X)+1)$.

4. Prove that if $T(0) = 0$, then $\log_n(\exp_n(T(X)) = T(X)$ (remark that this is well defined because $\exp_n(T(0)) = 1$). (Hint: derive both sides of the expression above).

5. Let $Y = \exp_N(T(X)) - 1 \mod X^N$. Using the question above, we have that

$$f(Y) = \log_N(1+Y) - T(X) = 0 \mod X^N.$$

Using Hensel lifting, deduce an algorithm computing $Y = \exp_N(T(X)) - 1 \mod X^N$ using $O(M(N))$ operations in K. (Hint: remember that as M is super-linear, we have that $M(N) + M(N/2) + \cdots + M(N/n^k) + \cdots \leq 2M(N)$).

3 Determinant

Let $M \in \mathcal{M}_n(\mathbb{K}[X])$. Assume that all the entries of M have degree at most d. Give an evaluation interpolation algorithm for computing $\det(M)$. What is its complexity ?

4 Quasi-Cauchy matrices

Let $\mathbf{x} = (x_i)_{0 \leqslant i \leqslant n-1} \in K^n, \mathbf{y} = (y_i)_{0 \leqslant i \leqslant n-1} \in K^n$. We assume that $x_i \neq y_j$ for all i, j and that $x_i \neq x_j$ and $y_i \neq y_j$ for $i \neq j$. The *Cauchy matrix* associated to these n-uples is the matrix $C(x,y) = (1/(x_i - y_j))_{0 \leqslant i,j \leqslant n-1}$.

Let $\mathbf{w} := (w_0, \ldots, w_j)$. We define a $j \times j$ diagonal matrix $D(\mathbf{w})$ by

$$D(\mathbf{w}) = \begin{bmatrix} w_0 & 0 & \cdots & 0 \\ 0 & w_1 & \ddots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & w_{j-1} \end{bmatrix}.$$

Let now $\varphi_{\mathbf{x},\mathbf{y}} : \mathcal{M}_n(K) \to \mathcal{M}_n(K)$ defined by $\varphi_{\mathbf{x},\mathbf{y}}(A) = D(\mathbf{x}) \cdot A - A \cdot D(\mathbf{y})$. With these notations, define the $(\mathbf{x},\mathbf{y})$-*displacement rank* of A to be the rank of $\varphi_{\mathbf{x},\mathbf{y}}(A)$. We shall assume that $\varphi_{\mathbf{x},\mathbf{y}}(A)$ is an invertible mapping – an easy fact, the proof of which is of little interest.

1. What is the $(\mathbf{x},\mathbf{y})$-displacement rank of the Cauchy matrix $C(\mathbf{x},\mathbf{y})$?

2. Let $\mathbf{u}, \mathbf{v}$ be two (column) vectors in K^n. Prove that $\varphi^{-1}_{\mathbf{x},\mathbf{y}}(\mathbf{u} \cdot {}^t\mathbf{v}) = D(\mathbf{u})C(\mathbf{x},\mathbf{y})D(\mathbf{v})$.

3. Deduce from the previous question that if a matrix M has $(\mathbf{x},\mathbf{y})$-displacement rank α, there exist vectors $\mathbf{g}_1,\ldots,\mathbf{g}_\alpha,\mathbf{h}_1,\ldots,\mathbf{h}_\alpha$ such that

$$M=\sum_{j=1}^{\alpha} D(\mathbf{g}_j)C(\mathbf{x},\mathbf{y})D(\mathbf{h}_j). \tag{1}$$

   (Hint: recall that if N has rank α, then $N=\sum_{i=1}^{\alpha} N_i$ with N_i of rank 1.)

4. Prove conversely that if M is of the form (1), then M has $(\mathbf{x},\mathbf{y})$-displacement rank $\leqslant \alpha$.

   For the rest of the exercise, we shall say that a matrix with $(\mathbf{x},\mathbf{y})$-displacement rank α is represented by $(\mathbf{x},\mathbf{y})$-generators of size k if M is given as a pair of vector sequences $((\mathbf{g}_i)_{1\leqslant i\leqslant\alpha},(\mathbf{h}_i)_{1\leqslant i\leqslant\alpha})\in(K^\alpha)^2$ such that (1) holds. Overall, this means that, when α is small, we have a compact representation for M (of size $O(\alpha n)$), and we might wonder whether we can do basic matrix arithmetic – matrix/vector product, add, multiply, inverse, determinant – using this compact representation (the last two can be done but we'll not study them).

   Recall that if M is a Cauchy matrix, and v a vector, you can compute Mv using $O(M(n)\log n)$ operations in K.

5. If M is represented by $(\mathbf{x},\mathbf{y})$-generators of size α and v is a vector, prove that one can compute $M\cdot v$ in complexity $O(\alpha M(n)\log n)$.

6. If M,M' are represented by $(\mathbf{x},\mathbf{y})$-generators of size α and α', give $(\mathbf{x},\mathbf{y})$-generators of size $\alpha+\alpha'$ for $M+M'$, which can be computed in time $O((\alpha+\alpha')n)$.

7. If M,M' are represented by $(\mathbf{x},\mathbf{y})$-generators of size α (resp. by $(\mathbf{y},\mathbf{z})$-generators of size α'), give $(\mathbf{x},\mathbf{z})$-generators of size $\alpha+\alpha'$ for $M\cdot M'$, which can be computed in time $O(\alpha\alpha' M(n)\log n)$.