# COMP4121 Lecture Notes

The Discrete Fourier Transform, the Discrete Cosine Transform and JPEG

LiC: Aleks Ignjatovic

ignjat@cse.unsw.edu.au

THE UNIVERSITY OF
NEW SOUTH WALES


School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia

# 1 DFT as a change of basis

Recall that the *scalar product* (also called the *dot* product) of two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1})$ and $\boldsymbol{y} = (y_0, y_1, \ldots, y_{n-1})$, denoted by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, is defined as

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=0}^{n-1} x_i y_i.$$

If the coordinates of the two vectors are complex numbers, i.e., if $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^n$, then the scalar product is defined as

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=0}^{n-1} x_i \overline{y_i}, \tag{1.1}$$

where $\overline{z}$ denotes the complex conjugate of $z$, i.e., if $z = a + i\,b$ where $a, b \in \mathbb{R}$, then $\overline{z} = a - i\,b$. Note that $\langle \boldsymbol{y}, \boldsymbol{x} \rangle = \overline{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$. In fact, in a vector space $V$, any binary function $\langle \boldsymbol{x}, \boldsymbol{y} \rangle : V^2 \to \mathbb{C}$ which satisfies the following three properties is a scalar product because it has all of the important properties which the particular scalar product given by (1.1) has.

1. (Conjugate symmetry) $\langle \boldsymbol{y}, \boldsymbol{x} \rangle = \overline{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}$
2. (Linearity in the first argument) for every scalar $\alpha$, $\langle \alpha \boldsymbol{x}, \boldsymbol{y} \rangle = \alpha \langle \boldsymbol{x}, \boldsymbol{y} \rangle$
3. (Positive definitness) $\langle \boldsymbol{x}, \boldsymbol{x} \rangle \geq 0$ and $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0$ just in case $\boldsymbol{x} = 0$

**Exercise:** Define a scalar product on $\mathbb{C}^n$ which is different from the usual one, given by (1.1).
*Hint:* Try giving different "importances" to different coordinates...

Every scalar product also defines an associated *norm* of a vector by

$$\|\boldsymbol{x}\| = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle} \geq 0$$

Geometrically, the norm plays the role of the length of a vector (and, in case of $\mathbb{R}^2$ or $\mathbb{R}^3$, it is just the usual, Euclidean length of the vector).

The usual basis of both $\mathbb{R}^n$ and $\mathbb{C}^n$ is given by

$$\mathcal{B} = \{(1, 0, 0, 0, \ldots 0), (0, 1, 0, 0, \ldots, 0), \ldots, (0, 0, 0, \ldots, 1)\}$$

Note that for any vector $\boldsymbol{a} = (a_0, a_1, a_2, \ldots, a_{n-1})$, such that $\boldsymbol{a} \in \mathbb{R}^n$ or $\boldsymbol{a} \in \mathbb{C}^n$,

$$(a_0, a_1, a_2, \ldots, a_{n-1}) = a_0(1, 0, 0, 0, \ldots, 0) + a_1(0, 1, 0, 0, \ldots, 0) + \ldots + a_{n-1}(0, 0, 0, \ldots, 1)$$

Let us set

$$\boldsymbol{e}_0 = (1, 0, 0, \ldots, 0, 0); \qquad \boldsymbol{e}_1 = (0, 1, 0, \ldots, 0, 0); \qquad \ldots \qquad \boldsymbol{e}_{n-1} = (0, 0, 0, \ldots, 0, 1);$$

Thus,

$$(a_0, a_1, a_2, \ldots, a_{n-1}) = a_0\boldsymbol{e_0} + a_1\boldsymbol{e}_1 + \ldots + a_{n-1}\boldsymbol{e}_{n-1} = \sum_{i=0}^{n-1} a_i \boldsymbol{e}_i$$

Let us denote the complex number $e^{i\,\frac{2\pi}{n}} = \cos\frac{2\pi}{n} + i \sin\frac{2\pi}{n}$ by $\omega_n$; such a number is a *primitive root of unity* because $(\omega_n)^n = 1$, and the set of all complex numbers $z$ which satisfy $z^n = 1$ is precisely the set of the $n$ powers of $\omega_n$, i.e., $z^n = 1$ if an only if $z$ is of the form $z = (\omega_n)^k$ for some $k$ such that $0 \leq k \leq n-1$.

We now introduce another basis, this time only in $\mathbb{C}^n$, given by $\mathcal{F} = \{\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{n-1}\}$, where

$$\boldsymbol{f}_k = ((\omega_n^0)^k, (\omega_n^1)^k, \ldots, (\omega_n^{n-1})^k) = (1, \omega_n^k, \ldots, \omega_n^{k(n-1)}).$$

Thus, the coordinates of $\boldsymbol{f}_k$ are the $k^{th}$ powers of the sequence of the $n$ roots of unity.

To show that this is indeed a basis we have to show that these vectors are linearly independent. In fact, they form an orthogonal basis. Two vectors are mutually orthogonal if $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0$.

**Exercise:** Prove that if a set consists of vectors which are pairwise orthogonal, then such a set of vectors must be linearly independent.

To see that vectors $\boldsymbol{f}_k$ and $\boldsymbol{f}_m$ are orthogonal if $k \neq m$ we compute their scalar product:

$$\langle \boldsymbol{f}_k, \boldsymbol{f}_m \rangle = \sum_{i=0}^{n-1} (\omega_n)^{k\cdot i} \overline{(\omega_n)^{m\cdot i}} = \sum_{i=0}^{n-1} (\omega_n)^{k\cdot i} (\omega_n)^{-m\cdot i} = \sum_{i=0}^{n-1} (\omega_n)^{(k-m)\cdot i} = \sum_{i=0}^{n-1} (\omega_n^{k-m})^i$$

The last sum is a sum of a geometric progression with ratio $q = \omega_n^{k-m}$ and thus, using formula

$$\sum_{i=0}^{n-1} q^i = \frac{1-q^n}{1-q}$$

we get

$$\langle \boldsymbol{f}_k, \boldsymbol{f}_m \rangle = \frac{1-(\omega_n^{k-m})^n}{1-\omega_n^{k-m}} = 0$$

because $(\omega_n^{k-m})^n = (\omega_n^n)^{k-m} = 1$ (the denominator is different from 0 because we have assumed that $k \neq m$). Let us compute the norm of these vectors:

$$\|\boldsymbol{f}_k\|^2 = \langle \boldsymbol{f}_k, \boldsymbol{f}_k \rangle = \sum_{i=0}^{n-1} (\omega_n)^{k\cdot i} \overline{(\omega_n)^{k\cdot i}} = \sum_{i=0}^{n-1} (\omega_n)^{k\cdot i} (\omega_n)^{-k\cdot i} = \sum_{i=0}^{n-1} (\omega_n)^0 = \sum_{i=0}^{n-1} 1 = n.$$

Thus, $\|\boldsymbol{f}_k\| = \sqrt{n}$; consequently, if we define vectors $\boldsymbol{\varphi}_k = \boldsymbol{f}_k/\sqrt{n}$, then these $n$ vectors form an *orthonormal* basis, $\boldsymbol{\Phi} = \{\boldsymbol{\varphi}_0, \ldots, \boldsymbol{\varphi}_{n-1}\}$, i.e., a basis satisfying $\langle \boldsymbol{\varphi}_k, \boldsymbol{\varphi}_m \rangle = \delta(m-k)$ where $\delta(0) = 1$ and $\delta(m) = 0$ if $m \neq 0$.

We now show that every scalar product and its corresponding norm satisfy the Cauchy-Bunyakovsky-Schwarz inequality:

**Theorem 1.1**

$$|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leq \|\boldsymbol{x}\| \cdot \|\boldsymbol{y}\|.$$

**Proof:** Let $\boldsymbol{z} = \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}$ and $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{z}$; note that the scalar product of $\boldsymbol{z}$ and $\boldsymbol{u}$ satisfies

$$\begin{aligned}
\langle \boldsymbol{z}, \boldsymbol{u} \rangle &= \left\langle \boldsymbol{x} - \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \quad \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \right\rangle \\
&= \left\langle \boldsymbol{x}, \quad \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \right\rangle - \left\langle \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \quad \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \right\rangle \\
&= \overline{\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle} \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle - \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \overline{\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle} \langle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \\
&= 0,
\end{aligned}$$

i.e., $\boldsymbol{u}$ is orthogonal on $\boldsymbol{z}$. Thus, $\boldsymbol{z}$ is a vector corresponding to the orthogonal projection of vector $\boldsymbol{x}$ onto the unit vector $\boldsymbol{y}/\|\boldsymbol{y}\|$:

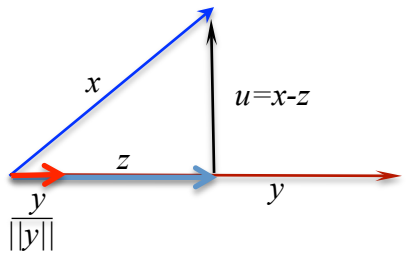



Figure 1.1:

We now have $\|\boldsymbol{u}\| \geq 0$ and thus

$$\begin{aligned}
0 &\leq \left\langle \boldsymbol{x} - \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \quad \boldsymbol{x} - \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \right\rangle \\
&= \|\boldsymbol{x}\|^2 - \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \langle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \boldsymbol{x} \rangle - \overline{\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle} \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle + \langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \overline{\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle} \langle \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \rangle \\
&= \|\boldsymbol{x}\|^2 - \frac{1}{\|\boldsymbol{y}\|^2} |\langle \boldsymbol{x}, \boldsymbol{y} \rangle|^2;
\end{aligned}$$

note that on the second line the second term (or the third, which is equal!) cancels out the fourth term. The last line implies the claim of the theorem. □

If the field of scalars is the field of real numbers $\mathbb{R}$, then the scalar product $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ of any two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is a real number and the inequality from Theorem 1.1 allows us to define angles between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ by

$$\angle(\boldsymbol{u}, \boldsymbol{v}) = \arccos \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\|\boldsymbol{u}\| \cdot \|\boldsymbol{v}\|}.$$

Thus, since we have a length function (the norm $\|x\|$ of a vector) and an angle function, a vector space with a scalar product has a well defined geometry. Also, if $\boldsymbol{y}$ is a unit vector, $\|\boldsymbol{y}\| = 1$, then we have $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \|x\| \cdot \cos(\angle \boldsymbol{x}, \boldsymbol{y})$, i.e., $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is just the length of the orthogonal projection of $\boldsymbol{x}$ onto the line to which $\boldsymbol{y}$ belongs; see Fig. 1.1.

**Theorem 1.2** *Let $\{\boldsymbol{\varphi}_m\}_{0 \le m < n}$ be any orthonormal basis of $\mathbb{C}^n$ and $\boldsymbol{c}$ any vector in $\mathbb{C}^n$; then*

$$\boldsymbol{c} = \sum_{m=0}^{n-1} \langle \boldsymbol{c}, \boldsymbol{\varphi}_m \rangle \boldsymbol{\varphi}_m; \tag{1.2}$$

*see also figure 1.2.*

**Proof:** Since $\boldsymbol{\Phi} = \{\boldsymbol{\varphi}_0, \ldots, \boldsymbol{\varphi}_{n-1}\}$ is a basis of $\mathbb{C}^n$, we have $\boldsymbol{c} = \lambda_0 \boldsymbol{\varphi}_0 + \ldots + \lambda_j \boldsymbol{\varphi}_j + \ldots + \lambda_{n-1} \boldsymbol{\varphi}_{n-1}$. Then, for every $0 \le j < n$ we have $\langle \boldsymbol{c}, \boldsymbol{\varphi}_j \rangle = \lambda_0 \langle \boldsymbol{\varphi}_0, \boldsymbol{\varphi}_j \rangle + \ldots + \lambda_j \langle \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j \rangle + \ldots + \lambda_{n-1} \langle \boldsymbol{\varphi}_{n-1}, \boldsymbol{\varphi}_j \rangle$ The only non-zero scalar product on the right is $\langle \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_j \rangle = 1$ and thus we obtain $\lambda_j = \langle c, \boldsymbol{\varphi}_j \rangle$, which implies (1.2); see Fig. 1.2. □

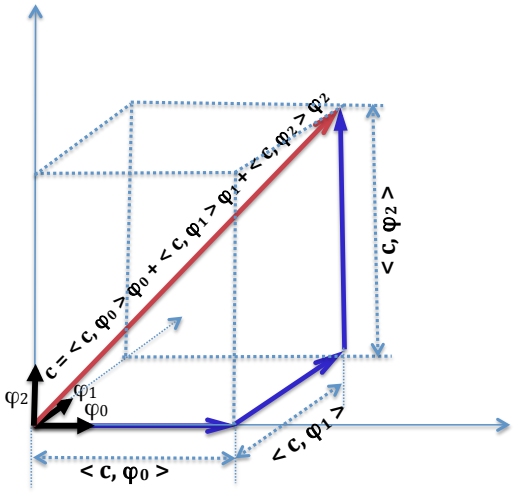



Figure 1.2:

Let us now calculate the vector $\widehat{\boldsymbol{c}} = (\widehat{c}(0), \widehat{c}(1), \ldots, \widehat{c}(n-1))$ of coordinates $\widehat{c}(m) = \langle \boldsymbol{c}, \boldsymbol{\varphi}_m \rangle$ of a vector $\boldsymbol{c}$ in basis $\boldsymbol{\Phi}$; using definitions of $\boldsymbol{\varphi}_m$ and of the scalar product in $\mathbb{C}^n$ we have

$$\widehat{c}(m) = \langle \boldsymbol{c}, \boldsymbol{\varphi}_m \rangle = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} c_k e^{-i \frac{2\pi}{n} m k} \tag{1.3}$$

Vector $\widehat{\boldsymbol{c}} = (\widehat{c}(0), \widehat{c}(1), \ldots, \widehat{c}(n-1))$ of coordinates $\widehat{c}(m) = \langle \boldsymbol{c}, \boldsymbol{\varphi}_m \rangle$ is called the **Discrete Fourier Transform (DFT)** of sequence (vector) $\boldsymbol{c}$.

*Thus, the DFT of a vector $\boldsymbol{c}$ is just the vector $\widehat{\boldsymbol{c}} = (\widehat{c}(0), \widehat{c}(1), \ldots, \widehat{c}(n-1))$ of the coordinates of $\boldsymbol{c}$ in the orthonormal basis $\boldsymbol{\Phi} = \{\boldsymbol{\varphi}_0, \ldots, \boldsymbol{\varphi}_{n-1}\}$ :*

$$\boldsymbol{c} = \sum_{m=0}^{n-1} \widehat{c}(m) \boldsymbol{\varphi}_m \tag{1.4}$$

Equating each coordinate of the lefthand side vector with the corresponding coordinate of the righthand side vector we obtain

$$c_k = \sum_{m=0}^{n-1} \widehat{c}(m) \boldsymbol{\varphi}_m(k) = \frac{1}{\sqrt{n}} \sum_{m=0}^{n-1} \widehat{c}(m) \mathrm{e}^{i \frac{2\pi k\, m}{n}} \tag{1.5}$$

Note that the "forward" transform formula for calculating $\{\widehat{c}(m)\}_{0\le m<n}$ from $\{c_k\}_{0\le k<n}$ and the inverse transform formula for calculating $\{c_k\}_{0\le k<n}$ from $\{\widehat{c}(m)\}_{0\le m<n}$ differ only by the sign of the exponents of the complex exponentials $\mathrm{e}^{\pm i\,\frac{2\pi k\,m}{n}}$. Also, by periodicity of the complex exponential $\mathrm{e}^{i\,x}$, we have that for every **integer** $k$ (but **NOT** for every real $k$!)

$$\mathrm{e}^{-i\,\frac{2\pi m}{n}\cdot k} = \mathrm{e}^{i\left(-\frac{2\pi m}{n}k+2\pi k\right)} = \mathrm{e}^{i\,\frac{2\pi(n-m)}{n}\cdot k}.$$

Thus, for every integer $k$,

$$\mathrm{e}^{i\,\frac{2\pi(n-m)}{n}\cdot k} = \mathrm{e}^{-i\,\frac{2\pi m}{n}\cdot k} = \overline{\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k}} \tag{1.6}$$

The last equality has two important consequences:

1. The imaginary parts in the righthand side sum in (1.5) will cancel out just in case $\widehat{c}(m) = \overline{\widehat{c}_{n-m}}$, because, then, for $t = k$, ($k$ an integer), using (1.6), we get that for integers $m, k, n$, such that $n \ge m, k > 0$

$$\begin{aligned}\widehat{c}(m)\,\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k} + \widehat{c}(n-m)\,\mathrm{e}^{i\,\frac{2\pi(n-m)}{n}\cdot k} &= \widehat{c}(m)\,\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k} + \overline{\widehat{c}(m)}\,\overline{\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k}} \\ &= \widehat{c}(m)\,\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k} + \overline{\widehat{c}(m)\,\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k}} \\ &= 2\mathcal{R}e(\widehat{c}(m)\,\mathrm{e}^{i\,\frac{2\pi m}{n}\cdot k}).\end{aligned}$$

   Thus, real discrete signals $\boldsymbol{c}$ of length $n$ are precisely the signals which satisfy $\widehat{c}(m) = \overline{\widehat{c}_{n-m}}$ for all $1 \le k \le n-1$.

   Note that, if signal $\boldsymbol{c}$ is real, then

$$\widehat{c}(0) = \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1} c_k e^{-\,i\,\frac{2\pi}{n}\,k\cdot 0} = \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1} c_k$$

   and consequently $\widehat{c}(0)$ is also real ($\widehat{c}(0)$ is called the *DC component* or *DC offset* of $\boldsymbol{c}$; DC stands for Direct Current). If, in addition, $n$ is even, then

$$\widehat{c}(n/2) = \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1} c_k \mathrm{e}^{-\,i\,\frac{2\pi}{n}\,k\cdot n/2} = \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1} c_k e^{-\,i\,\pi\,k} = \frac{1}{\sqrt{n}}\sum_{k=0}^{n-1}(-1)^k c_k$$

   and thus $\widehat{c}(n/2)$ is also real.

   So for real signals, if $n$ is even, in order to store $\widehat{\boldsymbol{c}}$ we have to store two floating point real numbers $\widehat{c}(0)$ and $\widehat{c}(n/2)$ plus $n/2-1$ complex numbers, $\widehat{c}(1), \ldots, \widehat{c}(n/2-1)$ because the remaining complex coefficients $\widehat{c}(n/2+k)$ can be obtained using (1.6). Thus, in total, to store $\widehat{\boldsymbol{c}}$ we have to store $2 + 2(n/2-1) = n$ floating point real numbers, exactly as many as to store the (real) vector $\boldsymbol{c}$.

   If $n$ is odd than we have to store one real number (i.e., $\widehat{c}(0)$) plus $(n-1)/2$ complex numbers, which again adds up to $n$ floating point numbers. Thus, both for real and complex vectors $\boldsymbol{c}$, storing $\boldsymbol{c}$ takes the same space as storing $\widehat{\boldsymbol{c}}$.

   Note that if $n$ is even, then the bin with index $n/2$ has a bin frequency $2\pi/n \cdot n/2 = \pi$, which is called the *Nyquist frequency*.

2. We can replace frequencies larger than $\pi$ with the corresponding negative frequencies, i.e., we can replace $2\pi(n-m)/n$ for $m < {}^n\!/_2$ with $-2\pi m/n$. This is useful if we are considering vectors obtained by sampling $\pi$-*bandlimited* signals, to be introduced in the next lecture.

# 2 But why consider such a complicated basis Φ??

To answer this question we look at the complex sinusoids of the form:

$$\mathrm{Si}_k^n(t) = \frac{1}{\sqrt{n}}\,\mathrm{e}^{i\,\frac{2\pi}{n}k\cdot t} = \frac{1}{\sqrt{n}}\left(\cos\left(\frac{2\pi}{n}k\cdot t\right) + i\sin\left(\frac{2\pi}{n}k\cdot t\right)\right)$$

of frequencies $2\pi k/n$, for all $0 \le k \le n-1$. Then each of the basis vectors $\boldsymbol{\varphi}_k = 1/\sqrt{n}(\omega_n^{k\cdot 0}, \omega_n^{k\cdot 1}, \ldots, \omega_n^{k\cdot(n-1)})$ is just a sequence of samples of the function $\mathrm{Si}_k^n(t)$, evaluated at integers $t = 0, 1, \ldots, n-1$:

$$\boldsymbol{\varphi}_k = (\mathrm{Si}_k^n(0), \ldots, (\mathrm{Si}_k^n(n-1))$$

Thus, if we represent a vector $\boldsymbol{c}$ in such a basis, i.e., as $\boldsymbol{c} = \sum_{k=0}^{n-1} \widehat{c}(k)\,\boldsymbol{\varphi}_k$, we have represented $\boldsymbol{c}$ as a linear combination of samples of such complex sinusoids. If we also see the sequence $\boldsymbol{c}$ as a sequence of samples of a continuous time (i.e., analog) signal $c(t)$, then over the interval $[0, n-1]$

$$c(t) \approx \sum_{k=0}^{n-1} \widehat{c}(k)\mathrm{Si}_k^n(t) = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} \widehat{c}(k)\mathrm{e}^{i\frac{2\pi k}{n}\cdot t} \tag{2.1}$$

where the equality is exact on integers $0, 1, \ldots, n-1$. Let us write each coordinate $\widehat{c}(k)$ in the polar form: $\widehat{c}(k) = |\widehat{c}(k)|\mathrm{e}^{i\arg(\widehat{c}(k))}$; then we get

$$\begin{aligned} c(t) &\approx \sum_{k=0}^{n-1} |\widehat{c}(k)|\mathrm{e}^{i\arg(\widehat{c}(k))}\frac{\mathrm{e}^{i\frac{2\pi k}{n}\cdot t}}{\sqrt{n}} = \sum_{k=0}^{n-1} \frac{|\widehat{c}(k)|}{\sqrt{n}}\mathrm{e}^{i\left(\frac{2\pi k}{n}\cdot t + \arg(\widehat{c}(k))\right)} \\ &= \sum_{k=0}^{n-1} \frac{|\widehat{c}(k)|}{\sqrt{n}}\left(\cos\left(\frac{2\pi k}{n}\cdot t + \arg(\widehat{c}(k))\right) + i\sin\left(\frac{2\pi k}{n}\cdot t + \arg(\widehat{c}(k))\right)\right). \end{aligned}$$

Note that the equality is exact only on integers $0, \ldots, n-1$, i.e., for $m$ such that $0 \le m \le n-1$, we have

$$\begin{aligned} c(m) &= \sum_{k=0}^{n-1} \frac{|\widehat{c}(k)|}{\sqrt{n}}\mathrm{e}^{i\left(\frac{2\pi k}{n}\cdot m + \arg(\widehat{c}(k))\right)} \\ &= \sum_{k=0}^{n-1} \frac{|\widehat{c}(k)|}{\sqrt{n}}\left(\cos\left(\frac{2\pi k}{n}\cdot m + \arg(\widehat{c}(k))\right) + i\sin\left(\frac{2\pi k}{n}\cdot m + \arg(\widehat{c}(k))\right)\right) \end{aligned}$$

Thus, we have obtained, what is called, a *spectral analysis* of $\boldsymbol{c}$, because the values $|\widehat{c}(k)|/\sqrt{n}$ represent the *amplitudes* of the *harmonics*, i.e., complex sinusoids of frequencies $\frac{2\pi k}{n}$, and the arguments $\arg(\widehat{c}(k))$ represent the *phase shifts* of these complex sinusoids. The values of $k$, $0 \le k < n$, are the indices of the corresponding *frequency bins*.

The above approximation given by (2.1) has two important shortcomings:

1. Evan if $\boldsymbol{c}$ is real, the sum (2.1) attains complex values for non integer values of $t$, because, as we have emphasized, (1.6) holds only for $t = k$, where $k$ is an integer.
2. It unnecessarily involves complex exponentials of frequencies larger than $\pi$.

This can easily be remedied by allowing complex exponentials of negative frequencies, thus letting

$$\mathrm{Si}_k^n(t) = \frac{1}{\sqrt{n}}\,\mathrm{e}^{i\frac{2\pi}{n}k\cdot t} = \frac{1}{\sqrt{n}}\left(\cos\left(\frac{2\pi}{n}k\cdot t\right) + i\sin\left(\frac{2\pi}{n}k\cdot t\right)\right)$$

for all $k$ such that $-\lfloor (n-1)/2\rfloor \le k \le \lfloor (n-1)/2\rfloor$, and, if $n$ is even, also

$$\mathrm{Si}_{n/2}^n(t) = \frac{1}{\sqrt{n}}\cos(\pi t).$$

These basis functions have frequencies at most $\pi$ and, for real valued $\boldsymbol{c}$, they result in a real valued interpolation function which is exact on the integers $0 \ldots n-1$ and with frequencies of its components at most equal to $\pm\pi$:

– for even $n$:

$$\begin{aligned} c(t) \approx \frac{1}{\sqrt{n}}\Big(&\widehat{c}(0) + \widehat{c}(1)\mathrm{e}^{i\frac{2\pi}{n}t} + \widehat{c}(2)\mathrm{e}^{i\frac{2\pi}{n}2t} + \ldots + \widehat{c}(n/2-1)\mathrm{e}^{i\frac{2\pi}{n}(n/2-1)t} + \widehat{c}(n/2)\cos\pi t + \\ &+ \widehat{c}(n/2+1)\mathrm{e}^{-i\frac{2\pi}{n}(n/2-1)t} + \ldots + \widehat{c}(n-1)\mathrm{e}^{-i\frac{2\pi}{n}t}\Big) \end{aligned}$$

– for odd $n$:

$$\begin{aligned} c(t) \approx \frac{1}{\sqrt{n}}\Big(&\widehat{c}(0) + \widehat{c}(1)\mathrm{e}^{i\frac{2\pi}{n}t} + \widehat{c}(2)\mathrm{e}^{i\frac{2\pi}{n}2t} + \ldots + \widehat{c}(\lfloor n/2\rfloor)\mathrm{e}^{i\frac{2\pi}{n}\lfloor n/2\rfloor t} + \\ &+ \widehat{c}(\lfloor n/2\rfloor + 1)\mathrm{e}^{-i\frac{2\pi}{n}\lfloor n/2\rfloor t} + \ldots + \widehat{c}(n-1)\mathrm{e}^{-i\frac{2\pi}{n}t}\Big) \end{aligned}$$

# 3 Application to signal compression

One of the main applications of spectral analysis of signals, namely signal compression, is based on the following heuristics:

*If a signal is not just noise, then only a small number of harmonics have a significant amplitude, i.e., only a small number of values $\widehat{c}(k)$ are significant; thus, all the remaining values $\widehat{c}(k)$ can be set to zero without causing an unacceptable distortion of the original signal.*

You might want to look at the Mathematica file used to produce the calculations and plots below, available at `http://www.cse.unsw.edu.au/~cs4121/DFT_and_DCT.nb`.

Let us start by considering the following signal:

$$s_1(t) = \cos\left(\frac{2\pi \cdot 3}{32}\, t + 2\right) + 2\cos\left(\frac{2\pi \cdot 7}{32}\, t - 1\right)$$

Note that this signal is real valued; thus, for each component we need two complex exponentials which are complex conjugates of each other, to cancel out the imaginary parts:

$$\begin{aligned} s_1(t) &= \frac{1}{2}\left(\mathrm{e}^{i\left(\frac{2\pi\cdot 3}{32}\, t+2\right)} + \mathrm{e}^{-i\left(\frac{2\pi\cdot 3}{32}\, t+2\right)}\right) + \mathrm{e}^{i\left(\frac{2\pi\cdot 7}{32}\, t-1\right)} + \mathrm{e}^{-i\left(\frac{2\pi\cdot 7}{32}\, t-1\right)} \\ &= \frac{1}{2}\left(\mathrm{e}^{i\left(\frac{2\pi\cdot 3}{32}\, t+2\right)} + \mathrm{e}^{i\left(\frac{2\pi\cdot 61}{32}\, t-2\right)}\right) + \mathrm{e}^{i\left(\frac{2\pi\cdot 7}{32}\, t-1\right)} + \mathrm{e}^{i\left(\frac{2\pi\cdot 57}{32}\, t+1\right)} \\ &= \frac{\mathrm{e}^{2\,i}}{2}\mathrm{e}^{i\,\frac{2\pi\cdot 3}{32}\, t} + \frac{\mathrm{e}^{-2\,i}}{2}\mathrm{e}^{i\,\frac{2\pi\cdot 61}{32}\, t} + \mathrm{e}^{-i}\mathrm{e}^{i\,\frac{2\pi\cdot 7}{32}\, t} + \mathrm{e}^{i}\mathrm{e}^{i\,\frac{2\pi\cdot 57}{32}\, t} \end{aligned}$$

Let us sample this signal at 32 integer points, thus obtaining the sequence $\boldsymbol{s}_1 = (s_1(0), s_1(1), \ldots, s_1(31))$. Let us also take the DFT of the sequence $\boldsymbol{s}_1$, denoted by $\widehat{\boldsymbol{s}}_1 = (\widehat{s}_1(0), \widehat{s}_1(1), \ldots, \widehat{s}_1(31))$ and then look at the sequence of the moduli of the components, $|\widehat{\boldsymbol{s}}_1| = (|\widehat{s}_1(0)|, |\widehat{s}_1(1)|, \ldots, |\widehat{s}_1(31)|)$. Using *Mathematica* we obtain Fig. 3.1 with a plot of $|\widehat{\boldsymbol{s}}_1|$ and which looks as expected, with 4 peaks at frequency bins $k = 3$, $k = 7$, $k = 32 - 7 = 25$ and $k = 32 - 3 = 29$.

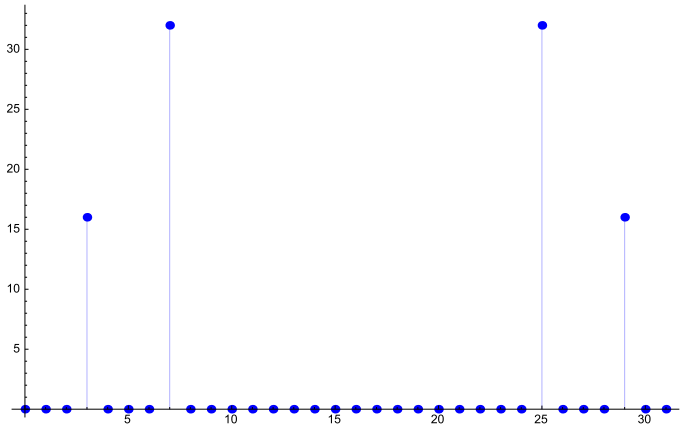



Figure 3.1:

Clearly, such a signal is extremely compressible in the *frequency domain* because all we need to store are 4 real numbers, namely the real and imaginary components of $\widehat{s}_1(3)$ and of $\widehat{s}_1(7)$ plus the corresponding bin indices 3 and 7; $\widehat{s}_1(25)$ can then be obtained as $\overline{\widehat{s}_1(7)}$ and $\widehat{s}_1(29)$ as $\overline{\widehat{s}_1(3)}$, and we can then recover all samples of the signal using formula (1.5).

But what happens if the samples $s_0, s_1, \ldots s_{n-1}$ come from a signal containing frequencies different from the bin frequencies, i.e., not of the form $2\pi k/n$? Let us now consider the following example. Define

$$s_2(t) = \cos\left(\frac{2\pi \cdot 1.35}{32}\, t + 2\right) + 2\cos\left(\frac{2\pi \cdot 4.05}{32}\, t - 2\right)$$

We again evaluate this sequence at integers $t = 0$ to $t = 31$ thus obtaining a sequence of values $\boldsymbol{s}_2$ and then compute the DFT $\widehat{\boldsymbol{s}}_2$ of such a sequence. Figure 3.2 shows plots of the real part (blue) of $\widehat{\boldsymbol{s}}_2$ and of its imaginary part (red).

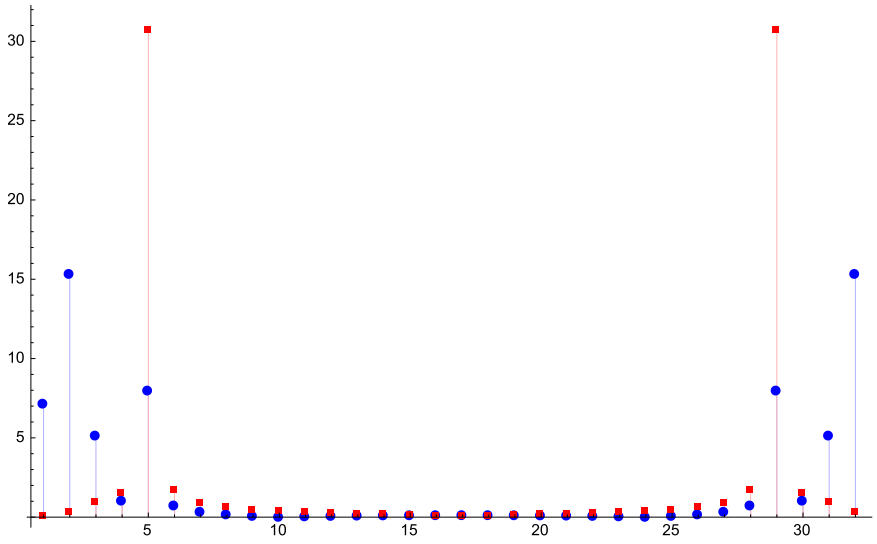



Figure 3.2: Plots of $|\mathcal{R}e(\widehat{s}_2)|$ (blue) and of $|\mathcal{I}m(\widehat{s}_2)|$ (red).

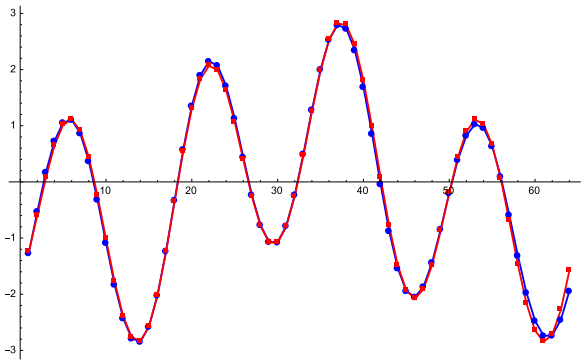
Since the components of the signal do not correspond to the frequencies of the bins of the DFT of length 32, for each of the four complex exponentials $\left\{e^{i\,\frac{2\pi\cdot 1.35}{32}\,t}, e^{-i\,\frac{2\pi\cdot 1.35}{32}\,t}, e^{i\,\frac{2\pi\cdot 4.05}{32}\,t}, e^{-i\,\frac{2\pi\cdot 4.05}{32}\,t}\right\}$ there are several complex exponentials with significant amplitudes (and with bin frequencies), because several adjacent complex exponentials have to superimpose in order to approximate a complex exponential of frequency between two bin frequencies. However, the signal is still compressible, because lots of components have small amplitudes.

As we have explained, out of 32 values of the DFT, we need to keep only the first 17 values $\widehat{s}_2(0), \ldots, \widehat{s}_2(16)$ (out of which $\widehat{s}_2(0)$ and $\widehat{s}_2(16)$ are real and the rest are complex), because the remaining values can be obtained from these by complex conjugation, using $\widehat{s}_2(32-k) = \overline{\widehat{s_2(k)}}$ for $k = 1$ to $k = 15$.

To compress our signal, let us keep only 8 largest components (real or imaginary) of the 17 numbers $\widehat{c}(0), \ldots \widehat{c}(16)$, setting all other components to zero, thus obtaining the sequence $\widehat{\sigma}_2(0), \ldots, \widehat{\sigma}_2(16)$, and then again obtain the remaining values as $\widehat{\sigma}_2(32-k) = \overline{\widehat{\sigma_2(k)}}$ for $k = 1$ to $k = 15$. We now use (1.5) in the form

$$s_2(m) \approx \sigma_2(m) = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} \widehat{\sigma}_2(k) e^{i\,\frac{2\pi k}{n}\cdot m} \tag{3.1}$$

to obtain the time domain representation of the corresponding approximation $\boldsymbol{\sigma}_2 = (\sigma_2(m) \ : \ 0 \leq m < 32)$. We plot the reconstructed sequence $\boldsymbol{\sigma}_2 = (\sigma_2(m) \ : \ 0 \leq m < 32)$ in blue and the original sequence $\boldsymbol{s}_2 = (s_2(m) \ : \ 0 \leq m < 32)$ in red:




So $\boldsymbol{\sigma}_2$ looks like a pretty good approximation of $\boldsymbol{s}_2$, given that we used only 8/32=1/4 of the original information. So one might be tempted to think that sampled signals which are not noise can be compressed by simply computing the DFT of the sampled signal and by setting to zero all real and imaginary components of

the elements of such a DFT which are smaller than a threshold (of course the floats can then be quantized to further improve compression). However, unfortunately, things are more complicated than that.

To see this, lets look at another signal:

$$s_3(t) = \cos\left(\frac{2\pi \cdot 1.34}{32}\, t\right) + 2\cos\left(\frac{2\pi \cdot 3.5}{32}\, t\right)$$

Sampling this signal at integers $0 - 31$ we obtain a vector of samples $\boldsymbol{s}_3$. Taking the DFT of this sequence we obtain a complex valued sequence $\widehat{\boldsymbol{s}}_3$; let us plot the real and imaginary parts of $\widehat{\boldsymbol{s}}_3$ (left on Fig. 3.3) and compare them with the real and imaginary parts of $\widehat{\boldsymbol{s}}_2$ (right).

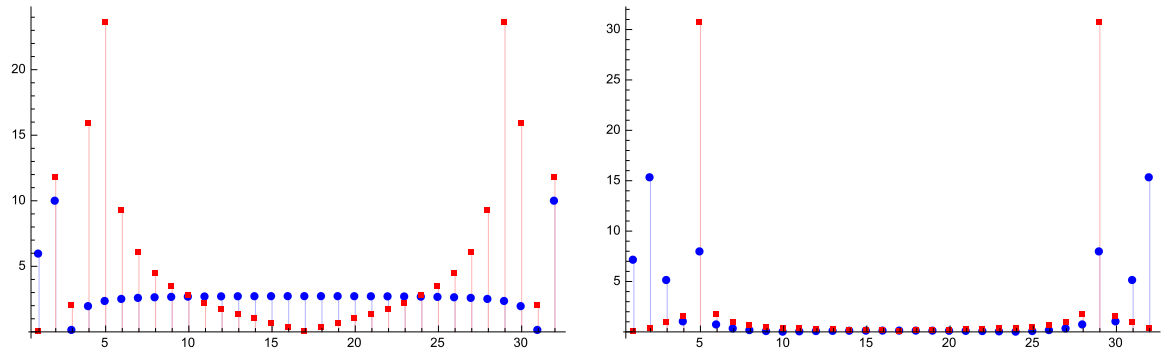



Figure 3.3: Real (blue) and imaginary (red) components of $\widehat{\boldsymbol{s}}_3$ (left) and of $\widehat{\boldsymbol{s}}_2$ (right).

Suddenly, a large number of components appear with significant amplitudes; thus, taking only 8 largest real or imaginary parts of the components of $\widehat{\boldsymbol{s}}_3$ will no longer result in good approximation. Fig. 3.4 on the left shows the plot of signal $\boldsymbol{c}_3$ (in blue) and the plot of the signal $\boldsymbol{\sigma}_3$ (in red), obtained by computing the DFT $\widehat{\boldsymbol{s}}_3$ of $\boldsymbol{s}_3$ and then replacing all the real and the imaginary components of the elements of $\widehat{\boldsymbol{s}}_3$ with zero, except for 8 largest such components, thus obtaining sequence $\widehat{\boldsymbol{\sigma}}_3$, and then using (1.5) to obtain the corresponding sequence $\boldsymbol{\sigma}_3$; on the right are shown sequences $\boldsymbol{s}_2$ and $\boldsymbol{\sigma}_2$ from the previous example.

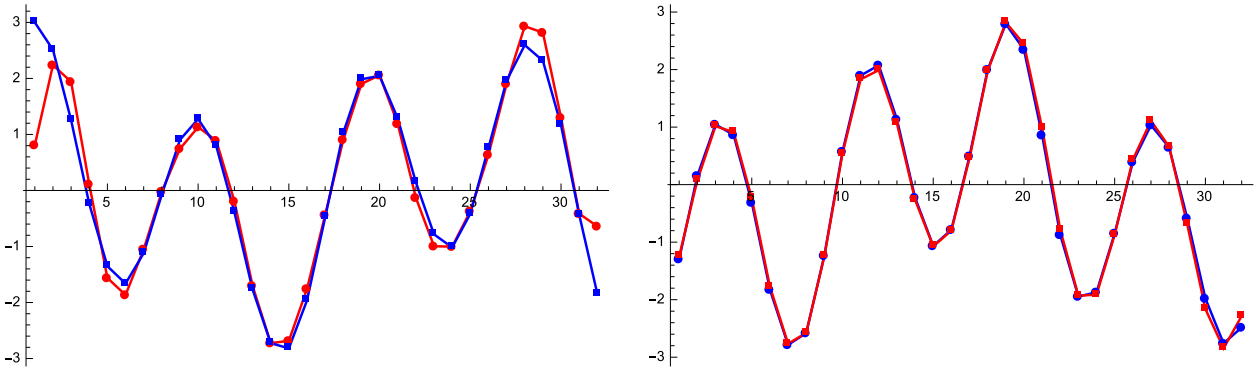



Figure 3.4: Left: signal $\boldsymbol{s}_3$ (blue) and its approximation $\boldsymbol{\sigma}_3$ (red); right: signal $\boldsymbol{s}_2$ and its approximation $\boldsymbol{\sigma}_2$.

Fig. 3.5 compares errors $\boldsymbol{s}_3 - \boldsymbol{\sigma}_3$ (blue) and $\boldsymbol{s}_2 - \boldsymbol{\sigma}_2$ (red) of approximation $\boldsymbol{\sigma}_3$ of $\boldsymbol{s}_3$ and approximation $\boldsymbol{\sigma}_2$ of $\boldsymbol{s}_2$. Clearly, $\boldsymbol{\sigma}_3$ does much worse job of approximating signal $\boldsymbol{s}_3$ then $\boldsymbol{\sigma}_2$ approximating $\boldsymbol{s}_2$, especially towards the edges of the interval $[0, 31]$. We now want to examine why this is so.

Note that formula (1.5), namely

$$c_k = \frac{1}{\sqrt{n}} \sum_{m=0}^{n-1} \widehat{c}(m) \mathrm{e}^{i\,\frac{2\pi k\, m}{n}} \tag{3.2}$$

can be evaluated not only for $k = 0 \ldots n-1$, but for arbitrary values of integer $k$; this results in a periodic repetition of the values $c_0, \ldots, c_{n-1}$. So, instead of plotting just one such period, let us plot on Fig. 3.6 two

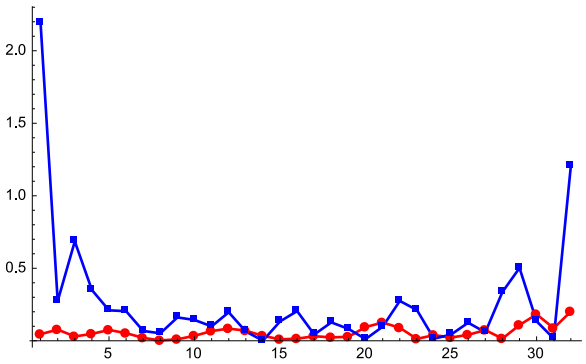



Figure 3.5: Approximation errors $|\boldsymbol{s}_3 - \boldsymbol{\sigma}_3|$ (blue) and $|\boldsymbol{s}_2 - \boldsymbol{\sigma}_2|$ (red).

consecutive complete periods of $\boldsymbol{s}_2$ (left) and of $\boldsymbol{s}_3$ (right); the first periods of both sequences are shown in red and the second periods in blue, joining the two periods with a dashed black line.

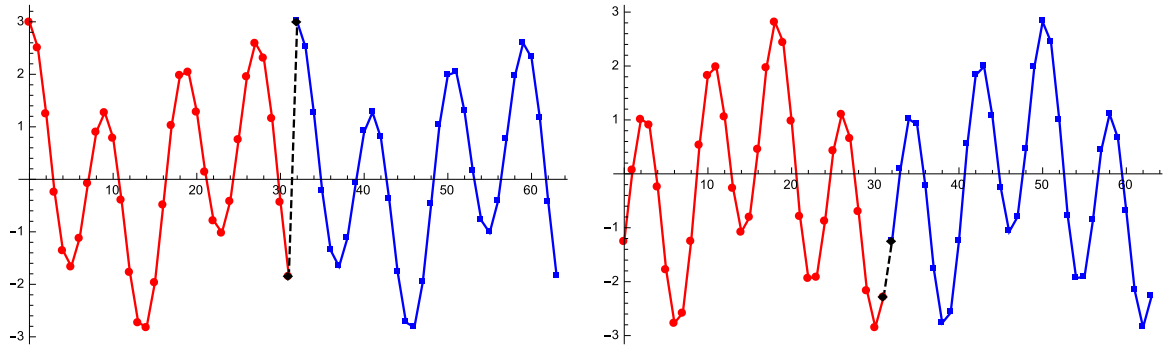



Figure 3.6: Two consecutive periods of $\boldsymbol{s}_3$ (left) and of $\boldsymbol{s}_2$ (right).

We now see that the initial samples of the second period of $\boldsymbol{s}_2$ (on the right) "continue the trend" of the final samples of the first period of $\boldsymbol{s}_2$; thus, the two periods taken together look like a sequence of samples of a continuous function. On the other hand, as it can be seen on the left plot, there is a large mismatch between the last sample of the first period of $\boldsymbol{s}_3$ and the first sample of its second period; the two periods taken together look more like samples of a signal which has a discontinuity somewhere between points $t = 31$ and $t = 32$. As a consequence, in order for $\boldsymbol{s}_3$ to "jump" between samples at $t = 31$ and $t = 32$, $\widehat{\boldsymbol{s}}_3$ needs to contain significant high frequency harmonics (which are more rapidly changing in value) and which enable $\boldsymbol{s}_3$ to make such a steep jump.

*Thus, these higher frequency harmonics are* ***NOT*** *"genuinely" present in the original signal but are* ***spurious artifacts*** *of our signal representation with complex exponentials!*

Since such spurious harmonics present in $\widehat{\boldsymbol{s}}_3$ are significant in size, they would also have to be encoded in the compressed encoding of the signal, to avoid large distortions after signal reconstruction (decompression) from its compressed spectral representation, as Fig. 3.3 has shown. However, this clearly reduces the level of compression we can achieve. Thus, we have to improve our signal model (i.e., our signal representation), in order to reduce the artifacts it introduces, so that they can be safely neglected during compression.

# 4 The Discrete Cosine Transform

In order to avoid artifacts caused by such jumps between the end point of a period and the first point of the next period, the JPEG algorithm uses another signal transform instead of the DFT, called the *Discrete Cosine Transform* or the DCT. In order to understand why DCT works, we will arrive to it via a somewhat roundabout way, but which is very much worth the effort.

In order to avoid artifacts caused by steep jumps between the end point of a period and the first point of the next period, JPEG relies on the following trick: given a signal $\boldsymbol{s}$ of length $n$, we first produce a new signal $\boldsymbol{S}$, of length $2n$, obtained by adding a "mirror image" of the original signal $\boldsymbol{s}$ at the end of $\boldsymbol{s}$; thus we let

$$\boldsymbol{S}(m) = \begin{cases} \boldsymbol{s}(m) & \text{if } \ 0 \leq m < n \\ \boldsymbol{s}(2n-1-m) & \text{if } \ n \leq m \leq 2n-1 \end{cases}$$

So

$$\boldsymbol{S} = (\boldsymbol{s}(0), \boldsymbol{s}(1), \ldots, \boldsymbol{s}(n-2), \boldsymbol{s}(n-1), \boldsymbol{s}(n-1), \boldsymbol{s}(n-2), \ldots, \boldsymbol{s}(1), \boldsymbol{s}(0)).$$

If we plot such a signal obtained from our signal $\boldsymbol{s}_3$ and showing its $\boldsymbol{s}_3$ part in red and the mirror image reflection of $\boldsymbol{s}_3$ in blue, we obtained the following plot:

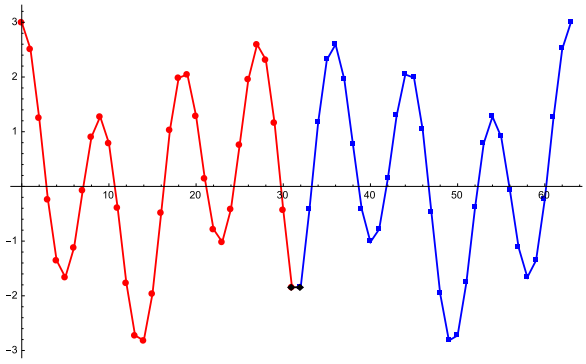



Figure 4.1: Signal $\boldsymbol{S}_3$ consisting of $\boldsymbol{s}_3$ (red) followed by the mirror image of $\boldsymbol{s}_3$ (blue)

Clearly, there is no jump at the junction of the signal and its mirror image; also the left end of $\boldsymbol{S}_3$ perfectly matches its right end, so, if we extend periodically $\boldsymbol{S}_3$, no large jumps will occur anywhare. Thus, now just the usual DFT of $\boldsymbol{S}_3$ will not contain spurious high frequency harmonics and will be readily compressible. To verify this, we compute $\widehat{\boldsymbol{S}}_3$ which is the DFT $\boldsymbol{S}_3$. On Fig. 4.2 we plot (on the left) the real part of $\widehat{\boldsymbol{S}}_3$ (blue) and imaginary part of $\widehat{\boldsymbol{S}}_3$ (red) and compare them with the real and imaginary parts of $\widehat{\boldsymbol{s}}_3$ which is the DFT of $\boldsymbol{s}_3$, plotted again on the right:

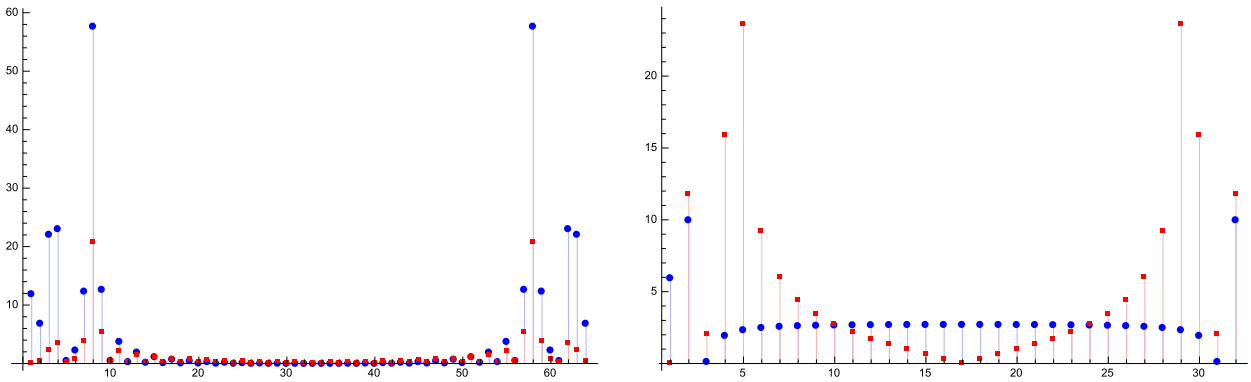



Figure 4.2: The real (blue) and the imaginary (red) parts of $\widehat{\boldsymbol{S}}_3$ (on the left), compared with the real and imaginary parts of $\widehat{\boldsymbol{s}}_3$ (on the right)

As it can be seen on Fig. 4.2, *almost all high frequency artifacts are gone.* Some of the artifacts remain because at the junction of a period of $\boldsymbol{s}_3$ with its adjacent mirror image, as well as between periods of $\boldsymbol{S}_3$, we might have something which looks like a cusp, while the complex exponentials are smooth, continuously differentiable functions. However, artifacts due to such cusps are much milder than the artifacts caused by a "discontinuity" (i.e., by a large jump) in the signal.

*This is an excellent example of what engineering is all about:* ***producing approximate models (i.e., solutions), physical or software, which meet the desired specifications with just a sufficient degree of accuracy for the purpose they are intended, while minimizing the incurred cost (financial, computational).*** *Making models which meet the specifications with an unnecessarily high degree of accuracy invariably makes such models unnecessarily costly for the intended purpose (this is called* ***over-engineering****).*

We now have to overcome another problem caused by our "trick". The length of $\widehat{\boldsymbol{S}}$ is $2n$. Since $\boldsymbol{S}$ is real $\widehat{\boldsymbol{S}}$ is "conjugate-symmetric", so it is enough to encode only $n$ components of $\widehat{\boldsymbol{S}}$, i.e., $\widehat{\boldsymbol{S}}(0), \ldots, \widehat{\boldsymbol{S}}(n-1)$. However, these $n$ components are complex, and this subsequence is NOT conjugate symmetric. So, it would appear that we would need $2n$ floating point numbers to represent $\widehat{\boldsymbol{S}}$. To avoid this, we have to slightly improve our transform.

Since the the sequence $\boldsymbol{S}$ is symmetric (i.e., a *palindrom*), equation (1.3) for $\widehat{\boldsymbol{S}}$ can be written in the following form, grouping equal components $S(k)$ and $S(2n-1-k)$ for $0 \le k < 2n$.

$$\widehat{S}(m) = \frac{1}{\sqrt{2n}} \sum_{k=0}^{2n-1} S_k e^{-i\,\frac{2\pi}{2n}\,m\,k} = \frac{1}{\sqrt{2n}} \sum_{k=0}^{n-1} S_k (e^{-i\,\frac{2\pi}{2n}\,m\,k} + e^{-i\,\frac{2\pi}{2n}\,m\,(2n-1-k)})$$

A conversion from an exponential to a trigonometric form shows that

$$e^{-i\,\frac{2\pi}{2n}\,m\,k} + e^{-i\,\frac{2\pi}{2n}\,m\,(2n-1-k)} = \cos\frac{\pi\,k\,m}{n} + \cos\frac{\pi\,m(2n-1-k)}{n} - i\left(\sin\frac{\pi\,k\,m}{n} + \sin\frac{\pi\,m(2n-1-k)}{n}\right)$$

Representing the imaginary part as a product we obtain

$$\sin\frac{\pi\,k\,m}{n} + \sin\frac{\pi\,m(2n-1-k)}{n} = 2\cos\frac{\pi m(2n-1-2k)}{2n}\sin\left(\pi m\left(1-\frac{1}{2n}\right)\right)$$

We now see that the sum $e^{-i\,\frac{2\pi}{2n}\,m\,k} + e^{-i\,\frac{2\pi}{2n}\,m\,(2n-1-k)}$ is "almost real": if instead of the factor $\sin\left(\pi m\left(1-\frac{1}{2n}\right)\right)$ on the righthand side of the above equation we had just $\sin(\pi m)$, since $m$ is an integer, $\sin(\pi m)$ would be equal to 0 and the imaginary part would vanish. So, to annihilate the imaginary part, we slightly rotate each component of $\widehat{S}(m)$, namely, we define a new transform $\widetilde{\boldsymbol{S}}$ via

$$\widetilde{S}(m) = \mathrm{e}^{i\,\frac{\pi\,m}{2n}}\,\widehat{S}(m) = \frac{1}{\sqrt{2n}} \sum_{k=0}^{2n-1} S_k e^{-i\,\frac{\pi}{n}\,m\,(k+\frac{1}{2})} \tag{4.1}$$

Since $S_k = S_{2n-1-k}$, by taking again the summands in pairs, we obtain that now imaginary parts cancel out and we get

$$e^{-i\,\frac{2\pi}{2n}\,m\,(k+\frac{1}{2})} + e^{-i\,\frac{2\pi}{2n}\,m\,(2n-1-k+\frac{1}{2})} = 2\cos\frac{\pi\,m(k+1/2)}{n} \tag{4.2}$$

Since $|\mathrm{e}^{i\,\frac{\pi\,m}{2n}}| = 1$ we have $|\widetilde{S}(m)| = |\widehat{S}(m)|$, so the amplitudes of the harmonics did not change and thus $\widetilde{\boldsymbol{S}}$ is also compressible. More over, we also have

$$\cos\frac{\pi\,(2n-m)(k+1/2)}{n} = \cos\left(\pi - \frac{m(k+1/2)}{n}\right) = -\cos\frac{m(k+1/2)}{n}$$

which implies that $\widetilde{\boldsymbol{S}}$ satisfies

$$\widetilde{S}(2n-m) = -\widetilde{S}(m), \tag{4.3}$$

and, in particular, $\widetilde{S}(n) = 0$. Thus, we only need to store $n$ real values $\widetilde{S}(0), \ldots, \widetilde{S}(n-1)$.

Vector $\widetilde{\boldsymbol{s}} = (\widetilde{S}(0), \ldots, \widetilde{S}(n-1))$ is called the **Discrete Cosine Transform (DCT) of type II** of vector $\boldsymbol{s}$ (not of $\boldsymbol{S}$ !), and (4.1) and (4.2) imply

$$\widetilde{S}(m) = \sqrt{\frac{2}{n}} \sum_{k=0}^{n-1} s_k \cos\frac{\pi\,m(k+1/2)}{n}$$

It is easy to verify by a direct calculation that the set of $2n$ vectors $\widetilde{\boldsymbol{\Phi}} = \{\widetilde{\boldsymbol{\phi}}_0, \ldots, \widetilde{\boldsymbol{\phi}}_{2n-1}\}$ where

$$\widetilde{\boldsymbol{\phi}}_k = \left(\frac{1}{\sqrt{2n}}\mathrm{e}^{i\,\frac{\pi k}{n}\cdot 1/2}, \frac{1}{\sqrt{2n}}\mathrm{e}^{i\,\frac{\pi k}{n}\cdot(1+1/2)}, \ldots, \frac{1}{\sqrt{2n}}\mathrm{e}^{i\,\frac{\pi k}{n}(2n-1+1/2)}\right)$$

forms an orthonormal basis of $\mathbb{C}^{2n}$. Thus, (4.1) simply states that $\widetilde{\boldsymbol{S}}$ is just the vector of the coordinates of vector $\boldsymbol{S} = (s_0, \ldots, s_{n-1}, s_{n-1}, \ldots, s_0)$ **in the basis** $\widetilde{\boldsymbol{\Phi}}$, which implies that $\boldsymbol{S} = \sum_{k=0}^{2n-1} \widetilde{S}(k)\widetilde{\boldsymbol{\phi}}_k$, which in turn implies that, coordinate-wise,

$$S(m) = \sum_{k=0}^{2n-1} \widetilde{S}(k)\widetilde{\boldsymbol{\phi}}_k(m) = \sum_{k=0}^{2n-1} \widetilde{S}(k)\frac{1}{\sqrt{2n}}\,\mathrm{e}^{i\,\frac{\pi k}{n}(m+1/2)}$$

Using (4.3) we now obtain that for all $m$ such that $0 \le m < 2n$ we have

$$S(m) = \frac{1}{\sqrt{2n}} \left( \widetilde{S}(0) + \sum_{k=1}^{n-1} \widetilde{S}(k) \left( \mathrm{e}^{i \frac{\pi k}{n}(m+1/2)} - \mathrm{e}^{i \frac{\pi(2n-k)}{n}(m+1/2)} \right) \right)$$

By replacing the complex exponentials with trigonometric functions we obtain

$$\mathrm{e}^{i \frac{\pi(m+1/2)k}{n}} - \mathrm{e}^{i \frac{\pi(m+1/2)(2n-k)}{n}} = 2 \cos \frac{\pi(m+1/2)k}{n}.$$

Letting $m$ range only from 0 to $n-1$ we obtain

$$s_m = S_m = \sqrt{\frac{2}{n}} \left( \frac{\widetilde{S}(0)}{2} + \sum_{k=1}^{n-1} \widetilde{S}(k) \cos \frac{\pi(m+1/2)k}{n} \right)$$

Sometimes we use the following, less intuitive but perhaps simpler versions:

$$\widetilde{S}(m) = 2 \sum_{k=0}^{n-1} s_k \cos \frac{\pi\, m(k+1/2)}{n}$$

and

$$s_m = \frac{1}{n} \left( \frac{\widetilde{S}(0)}{2} + \sum_{k=1}^{n-1} \widetilde{S}(k) \cos \frac{\pi(m+1/2)k}{n} \right)$$

Fig. 4.3 compares the absolute values of the DCT of signal $\boldsymbol{s}_3$ (left) with the real (blue) and the imaginary part of the DFT of the same signal (right).

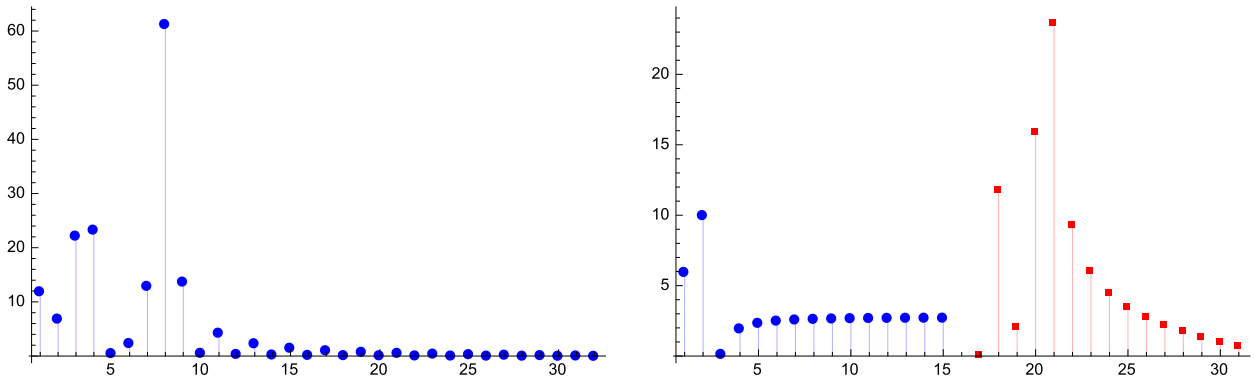



Figure 4.3: The DCT $\widetilde{\boldsymbol{s}}_3$ of signal $\boldsymbol{s}_3$ (left) and the real (blue) and the imaginary (red) parts of the DFT $\widehat{\boldsymbol{s}}_3$ of the same signal (right)

Clearly, the 32 values of $\widetilde{\boldsymbol{s}}_3$ are much more compressible with low artifacts than the 32 values of $\widehat{\boldsymbol{s}}_3$. In fact, choosing again only the 8 largest numbers of $\widetilde{\boldsymbol{s}}_3$ and setting the rest to 0 produces an approximation $\widetilde{\boldsymbol{\sigma}}_3$ of $\widetilde{\boldsymbol{s}}_3$. Inverting $\widetilde{\boldsymbol{\sigma}}_3$ we obtain an approximation $\boldsymbol{\sigma}_3^*$ of signal $\boldsymbol{s}_3$ which we compare on Fig. 4.4 with the previous approximation $\boldsymbol{\sigma}_3$ obtained by taking 8 largest components of the DFT $\widehat{\boldsymbol{\sigma}}_3$ of the same input signal $\boldsymbol{s}_3$ thus obtaining $\widehat{\boldsymbol{\sigma}}_3$ and then inverting $\widehat{\boldsymbol{\sigma}}_3$ to obtain the corresponding time domain sequence $\boldsymbol{\sigma}_3$. It is obvious from that plot that the DCT does a much better job than the DFT, especially towards the end points of the signal $\boldsymbol{s}_3$. This is further made even more obvious by considering the corresponding errors of approximations, shown on Fig.4.5.

# 5 The DFT and the DCT in two dimensions and the JPEG

Both the DFT and the DCT readily generalize to several dimensions. For 2 dimensions, i.e., for images, a basis for $n \times n$ matrices can be obtained as the outer product (which is also the tensor product) of 1D basis vectors, i.e., for all $0 \le k < n$ and $0 \le p < n$ we have a basis matrix

$$\boldsymbol{\varphi}(k,p) = \left\{ \frac{1}{n} \cdot \mathrm{e}^{i \frac{2\pi k\, m}{n}} \cdot \mathrm{e}^{i \frac{2\pi p\, q}{n}} \;:\; 0 \le m, q < n \right\} = \left\{ \frac{1}{n} \mathrm{e}^{2\pi\, i \frac{k\, m + p\, q}{n}} \;:\; 0 \le m, q < n \right\}$$

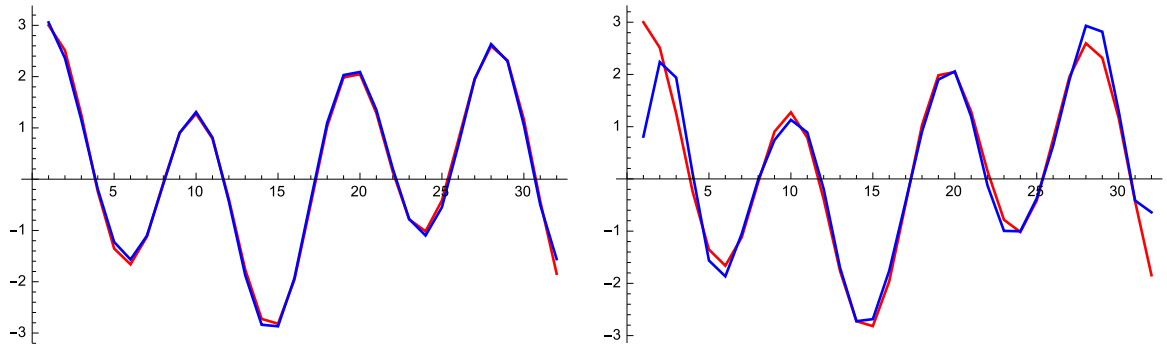



Figure 4.4: The DCT $\widetilde{\boldsymbol{s}}_3$ of signal $\boldsymbol{s}_3$ (left) and the real (blue) and the imaginary (red) parts of the DFT $\widehat{\boldsymbol{s}}_3$ of the same signal (right)

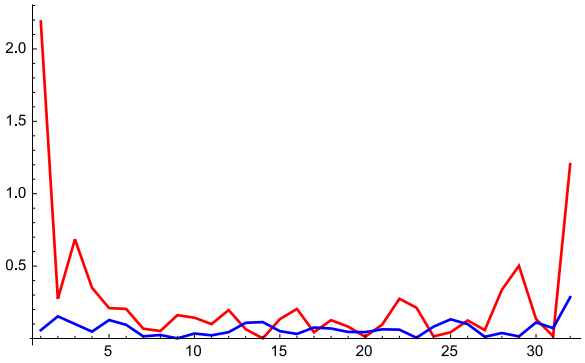
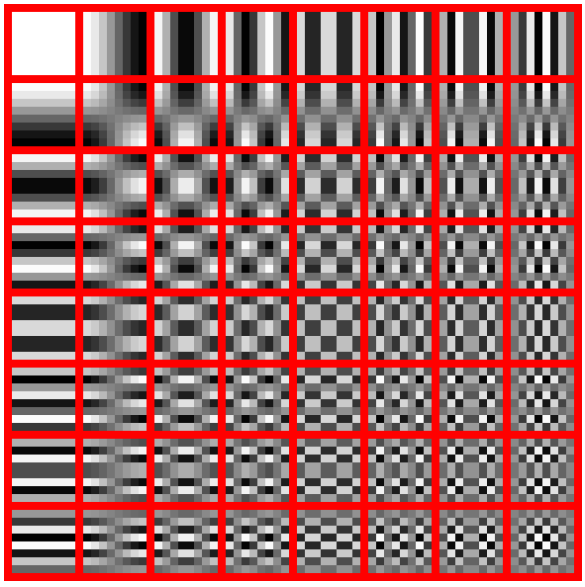



Figure 4.5: The DCT $\widetilde{\boldsymbol{s}}_3$ of signal $\boldsymbol{s}_3$ (left) and the real (blue) and the imaginary (red) parts of the DFT $\widehat{\boldsymbol{s}}_3$ of the same signal (right)

The DCT also generalizes in 2D in a straightforward way, using basis matrices

$$\widetilde{\boldsymbol{\phi}}(k,p) = \left( \cos\frac{\pi i(k+1/2)}{n} \cos\frac{\pi j(p+1/2)}{n} \;:\; 0 \le i,j < n \right).$$

These 64 matrices are shown below (image pinched from the Wikipedia article):



DCT generalized to 2 dimensions is given by the following two formulas:

$$\widetilde{X}(k,p) = \frac{1}{n} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} X(i,j) \cos\frac{\pi(j+1/2)k}{n} \cos\frac{\pi(i+1/2)p}{n},$$

and, letting $a(0) = 1/2$ and $a(k) = 1$ if $k \neq 0$,

$$X(k,p) = \frac{4}{n} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} a(i)a(j)\widetilde{X}(i,j) \cos \frac{\pi i(k+1/2)}{n} \cos \frac{\pi j(p+1/2)}{n}.$$

!!TO BE EXPANDED FOR NEXT YEAR's RUN!!

We very briefly sketch the JPEG image compression algorithm as applied to grey scale images; for more technical details please read, for example, the Wikipedia entry `https://en.wikipedia.org/wiki/JPEG#/media/File:Dctjpeg.png`. The uncompressed images are usually encoded by the values of each pixel, using 8 bits, with pixel values ranging from 0 to 255, where smaller values correspond to darker pixels. JPEG splits such an image into squares of size $8 \times 8$ pixels, which are encoded separately. As an example, let us consider one such block, taken from the famous photo of Lena of size $512 \times 512$ pixels, from her right eye, taking pixels belonging to rows 264 to 271 and columns 264 to 271, which is block (33,33):

Note that this is one of the "trickiest" blocks to compress, because it contains quite a bit of details. The values of the 64 pixels are given by the matrix

$$M = \begin{bmatrix} 50 & 40 & 46 & 48 & 76 & 62 & 69 & 94 \\ 45 & 39 & 36 & 40 & 88 & 87 & 65 & 86 \\ 48 & 38 & 36 & 39 & 56 & 90 & 65 & 50 \\ 76 & 48 & 41 & 43 & 69 & 112 & 77 & 56 \\ 90 & 72 & 66 & 66 & 90 & 108 & 74 & 53 \\ 98 & 86 & 86 & 91 & 83 & 72 & 57 & 66 \\ 79 & 87 & 90 & 80 & 77 & 55 & 65 & 113 \\ 57 & 54 & 61 & 58 & 65 & 77 & 107 & 160 \end{bmatrix}$$

Since our basis matrices attain both positive and negative values, the image is first centered around 0 (as opposed to 128) by subtracting from each pixel value 128, thus obtaining matrix

$$S = \begin{bmatrix} -78 & -88 & -82 & -80 & -52 & -66 & -59 & -34 \\ -83 & -89 & -92 & -88 & -40 & -41 & -63 & -42 \\ -80 & -90 & -92 & -89 & -72 & -38 & -63 & -78 \\ -52 & -80 & -87 & -85 & -59 & -16 & -51 & -72 \\ -38 & -56 & -62 & -62 & -38 & -20 & -54 & -75 \\ -30 & -42 & -42 & -37 & -45 & -56 & -71 & -62 \\ -49 & -41 & -38 & -48 & -51 & -73 & -63 & -15 \\ -71 & -74 & -67 & -70 & -63 & -51 & -21 & 32 \end{bmatrix}$$

We now compute the 2D DCT of this matrix. The values of each component of the DCT are now divided by corresponding value of the following "psycho-visual" matrix $Q$ and thus computed ratios are rounded to the nearest integer:

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

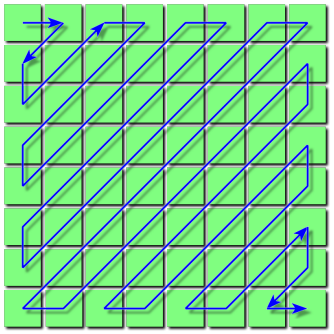
The idea is that human vision is not equally sensitive to all spacial frequencies; high spacial frequency components change more rapidly and can thus be represented with lower accuracy. Just think of a stain on a light plain shirt, versus the same stain on a shirt with a busy pattern; the same stain will be much less noticeable on a shirt with a busy pattern than on a plain shirt.

Let $C = \left( [\widehat{S}(i,j)/Q(i,j)] \ : \ 1 \leq i,j \leq 8 \right)$ where $[x]$ denotes the value of $x$ rounded to the nearest integer; in our example we obtain

$$C = \begin{bmatrix} -29 & -4 & 1 & 1 & 0 & 0 & 0 & 0 \\ -4 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -2 & 1 & -1 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The choice of $Q$ is determined by the desired level of compression; more aggressive compression will have larger entries, thus producing more zeros in matrix $C$. Note an interesting feature of $Q$: the matrix is not symmetric, indicating that human vision is not equally sensitive for horizontal and vertical patterns.

Such a resulting 2D matrix $C$ is turned into a 1D array by ordering the values using the "Cantor snake-like" path, where entry $C(i,j)$ is placed at the position $1/2(i+j)(i+j+1)+i$, as shown on the following diagram from the Wikipedia article: In this way zeros appear as long stretches. Such a sequence is obviously highly



compressible using standard lossless compression algorithms.

We now decompress matrix $C$ by multiplying it element by element with matrix $Q$, take the inverse DCT and then add value 128 which we originally subtracted. The result of such decompression is shown on figure below; on the left is the original, in the middle is the decompressed version of the same original and on the right is the error (inverted, so that larger values correspond to darker pixels.