Project Report

on

Fourier Transform And Its Application In Audio Compression

submitted towards the partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology
in
Computer Engineering

Submitted by

Ankit Kumar

2K20/B2/62

Anmol Singh

2K20/B2/70

Under the Supervision
of

Dr. Nilam Rathi

Department of Applied Mathematics

Delhi Technological University
Bawana Road. Delhi -110042
July -2021

DECLARATION

We hereby certify that the work, which is presented in the Project entitled **Fourier Transform And Its Application In Audio Compression** requirement for the award of the Degree of Bachelor of Technology in Computer Engineering and submitted to the Department of Applied Mathematics, Delhi Technological University, Delhi is an authentic record of our own, carried out under the supervision of **Dr. Nilam Rathi.**

The work presented in this report has not been submitted and not under consideration for the award for any other course/degree of this or any other Institute/University.

Ankit Kumar

2K20/B2/62

B.Tech, Computer Engineering

Anmol Singh

2K20/B2/70

B.Tech, Computer Engineering

SUPERVISOR CERTIFICATE

To the best of my knowledge, the report comprises original work and has not been submitted in part or full for any Course/Degree to this university or elsewhere as per the candidate's declaration.

Place: Delhi

Date

Supervisor name and signature

Department of Applied Mathematics

DELHI TECHNOLOGICAL UNIVERSITY
(FORMERLY Delhi College of Engineering)
Bawana Road, Delhi-110042

ABSTRACT

A major part of the information we receive and perceive every day is in the form of audio. Most of these sounds are transferred directly from the source to our ears, like when we have a face to face conversation with someone or listen to the sounds in a forest or a street. However, a considerable part of the sounds are generated by loudspeakers in various kinds of audio machines like cell phones, digital audio players, home cinemas, radios, television sets and so on. The sounds produced by these machines are either generated from information stored inside, or electromagnetic waves are picked up by an antenna, processed,

and then converted to sound. The sound that is stored inside the machines or picked up by the antennas is usually represented as *digital sound*. This has certain limitations, but at the same time makes it very easy to manipulate and process the sound on a computer. What we perceive as sound corresponds to the physical phenomenon of slight variations in air pressure near our ears. Larger variations mean louder sounds, while faster variations correspond to sounds with a higher pitch. The air pressure varies continuously with time, but at a given point in time it has a precise value. This means that sound can be considered to be a mathematical function.

A sound can be represented by a mathematical function, with time as the free variable. When a function represents a sound, it is often referred to as a *continuous signal*.

Department of Applied Mathematics

DELHI TECHNOLOGICAL UNIVERSITY

(FORMERLY Delhi College of Engineering)

Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

In performing this major project, We had to take the help and guideline of some respected people, who deserve our greatest gratitude. The completion of this assignment gives us much pleasure. We would like to show our gratitude **Dr. Nilam Rathi**, Mentor for this major project. Giving us a good guideline for report throughout numerous consultations. We would also like to extend our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

Many people, especially my classmates themselves, have made valuable comment suggestions on this proposal which gave me an inspiration to improve our assignment. We thank all the people for their help directly and indirectly to complete our assignment.

In addition, We would like to thank the **Department of Applied Mathematics**, Delhi Technological University for giving us the opportunity to work on this topic.

DIGITAL SOUND

On computers and various kinds of media players sound is digital which means that the sound is represented by a large number of function values, and not by a function defined for all times in some interval.

In most cases, a digital sound is sampled from an analog (continuous) audio signal. This is usually done with a technique called Pulse Code Modulation (PCM). The audio signal is sampled at regular intervals and the sampled values stored in a suitable number format. Both the sampling frequency, and the accuracy and number format used for storing the samples, may vary for different kinds of audio, and both influence the quality of the resulting sound. For simplicity the quality is often measured by the number of bits per second, i.e., the product of the sampling rate and the number of bits (binary digits) used to store each sample. This is also referred to as the bit rate. For the computer to be able to play a digital sound, samples must be stored in a file or in memory on a computer. To do this efficiently, digital sound formats are used. A couple of them are described in the examples below.

COMPRESSION OF SOUND AND THE MP3 STANDARD

Digital audio first became commonly available when the CD was introduced in the early 1980s. As the storage capacity and processing speeds of computers increased, it became possible to transfer audio files to computers and both play and manipulate the data, in ways such as in the previous section. However, audio was represented by a large amount of data and an obvious challenge was how to reduce the storage requirements. Lossless coding techniques like Huffman and Lempel-Ziv coding were known and with these kinds of techniques the file size could be reduced to about half of that required by the CD format. However, by allowing the data to be altered a little bit it turned out that it was possible to reduce the file size down to about ten percent of the CD format, without much loss in quality. The MP3 audio format takes advantage of this.

MP3, or more precisely MPEG-1 Audio Layer 3, is part of an audio-visual standard called MPEG. MPEG has evolved over the years, from MPEG-1 to MPEG-2, and then to MPEG-4. The data on a DVD disc can be stored with either MPEG-1 or MPEG-2, while the data on a bluray-disc can be stored with either MPEG-2 or MPEG-4. MP3 was developed by Philips, CCETT (Centre commun d'études de télévision et télécommunications), IRT (Institut für Rundfunktechnik) and Fraunhofer Society, and became an international standard in 1991. Virtually all audio software and music players support this format. MP3 is just a sound format and does not specify the details of how the encoding should be done. As a consequence, there are many different MP3 encoders available, of varying quality. In particular, an encoder which works well for higher bit rates (high quality sound) may not work so well for lower bit rates.

With MP3, the sound samples are transformed using methods we will go through in the next section. A frequency analysis of the sound is the basis for this transformation. Based on this frequency analysis, the sound is split into frequency bands, each band corresponding to a particular frequency range. With MP3, 32 frequency bands are used. Based on the frequency analysis, the encoder uses what is called a psycho-acoustic model to compute the significance of each band for the human perception of the sound.

When we hear a sound, there is a mechanical stimulation of the ear drum, and the amount of stimulus is directly related to the size of the sample values of the digital sound. The movement of the ear drum is then converted to electric impulses that travel to the brain where they are perceived as sound. The perception process uses a transformation of the sound so that a steady oscillation in air pressure is perceived as a sound with a fixed frequency. In this process certain kinds of perturbations of the sound are hardly noticed by the brain, and this is exploited in lossy audio compression.

More precisely, when the psycho-acoustic model is applied to the frequency content resulting from our frequency analysis, scale factors and masking thresholds are assigned for each band. The computed masking thresholds have to do with a phenomenon called masking effects. A simple example of this is that a loud sound will make a simultaneous low sound inaudible. For compression this means that if certain frequencies of a signal are very prominent, most of the other frequencies can be removed, even when they are quite large. If the sounds are below the masking threshold, it is simply omitted by the encoder, since the model says that the sound should be inaudible.

Masking effect is just one example of what is called a psycho-acoustic effect. Another obvious such effect regards computing the scale factors: the human auditory system can only perceive frequencies in the range 20 Hz – 20 000 Hz. An obvious way to do compression is therefore to remove frequencies outside this range, although there are indications that these frequencies may influence the listening experience inaudibly. The computed scaling factors tell the encoder about the precision to be used for each frequency band: If the model decides that one band is very important for our perception of the sound, it assigns a big scale factor to it, so that more effort is put into encoding it by the encoder (i.e. it uses more bits to encode this band).

Using appropriate scale factors and masking thresholds provide compression, since bits used to encode the sound are spent on parts important for our perception. Developing a useful psycho-acoustic model requires detailed knowledge of human perception of sound. Different MP3 encoders use different such models, so that may produce very different results, worse or better.

The information remaining after frequency analysis and using a psycho-acoustic model is coded efficiently with (a variant of) Huffman coding. MP3 supports bit rates from 32 to 320 kb/s and the sampling rates 32, 44.1, and 48 kHz. The format also supports variable bit rates (the bit rate varies in different parts of the file). An MP3 encoder also stores metadata about the sound, such as the title of the audio piece, album and artist name and other relevant data.

MP3 too has evolved in the same way as MPEG, from MP1 to MP2, and to MP3, each one more sophisticated than the other, providing better compression. MP3 is not the latest development of audio coding in the MPEG family: AAC (Advanced Audio Coding) is presented as the successor of MP3 by its principal developer, Fraunhofer Society, and can achieve better quality than MP3 at the same bit rate, particularly for bit rates below 192 kb/s. AAC became well known in April 2003 when Apple introduced this format (at 128 kb/s) as the standard format for their iTunes Music Store

and iPod music players. AAC is also supported by many other music players, including the most popular mobile phones.

The technologies behind AAC and MP3 are very similar. AAC supports more sample rates (from 8 kHz to 96 kHz) and up to 48 channels. AAC uses the same transformation as MP3, but AAC processes 1 024 samples at a time. AAC also uses much more sophisticated processing of frequencies above 16 kHz and has a number of other enhancements over MP3. AAC, as MP3, uses Huffman coding for efficient coding of the transformed values. Tests seem quite conclusive that AAC is better than MP3 for low bit rates (typically below 192 kb/s), but for higher rates it is not so easy to differentiate between the two formats. As for MP3 (and the other formats mentioned here), the quality of an AAC file depends crucially on the quality of the encoding program.

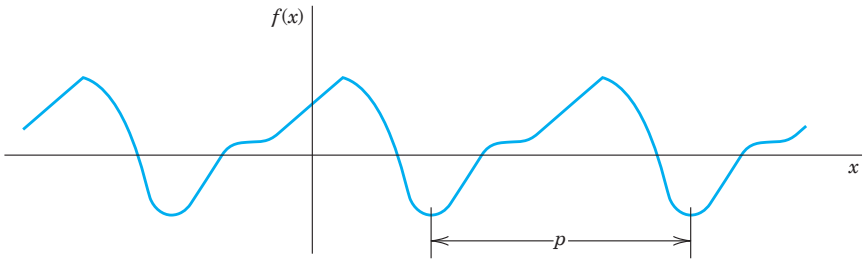
There are a number of variants of AAC, in particular AAC Low Delay (AAC-LD). This format was designed for use in two-way communication over a net-work, for example the internet. For this kind of application, the encoding (and decoding) must be fast to avoid delays (a delay of at most 20 ms can be tolerated).

FOURIER ANALYSIS FOR PERIODIC FUNCTIONS: FOURIER SERIES

We've identified audio signals with functions and discussed informally the idea of decomposing a sound into basis sounds to make its frequency content available. Now, we will make this kind of decomposition precise by discussing how a given function can be expressed in terms of the basic trigonometric functions. This is similar to Taylor series where functions are approximated by combinations of polynomials. But it is also different from Taylor series because polynomials are different from polynomials, and the approximations are computed in a very different way. The theory of approximation of functions with trigonometric functions is generally referred to as Fourier anal-ysis. This is a central tool in practical fields like image and signal processing, but it also an important field of research within pure mathematics. We will only discuss Fourier analysis for functions defined on a finite interval and for finite sequences (vectors), but Fourier analysis may also be applied to functions defined on the whole real line and to infinite sequences.

FOURIER SERIES

Fourier series are infinite series that represent periodic functions in terms of cosines and sines. As such, Fourier series are of greatest importance to the engineer and applied mathematician. To define Fourier series, we first need some background material. A function $f(x)$ is called a **periodic function** if $f(x)$ is defined for all real x, except possibly at some points, and if there is some positive number p, called a period of such that



Periodic function of period p

(1) $$f(x + p) = f(x) \qquad \text{for all } x.$$

(The function $f(x) = \tan x$ is a periodic function that is not defined for all real x but undefined for some points (more precisely, countably many points), that is $x = \pm\pi/2$, $\pm 3\pi/2, \cdots$.)

The graph of a periodic function has the characteristic that it can be obtained by periodic repetition of its graph in any interval of length p (Fig. 258).

The smallest positive period is often called the *fundamental period*. (See Probs. 2–4.)

Familiar periodic functions are the cosine, sine, tangent, and cotangent. Examples of functions that are not periodic are x, x^2, x^3, e^x, $\cosh x$, and $\ln x$, to mention just a few.

If $f(x)$ has period p, it also has the period $2p$ because (1) implies $f(x + 2p) = f(x + p + p) = f(x + p) = f(x)$, etc.; thus for any integer $n = 1, 2, 3, \cdots$,

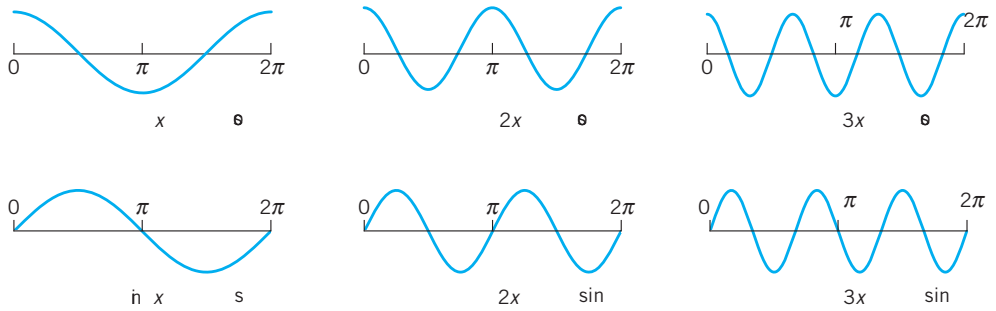
(2) $$f(x + np) = f(x) \qquad \text{for all } x.$$

Furthermore if $f(x)$ and $g(x)$ have period p, then $af(x) + bg(x)$ with any constants a and b also has the period p.

Our problem in the first few sections of this chapter will be the representation of various ***functions f(x) of period 2π*** in terms of the simple functions

(3) $$1, \quad \cos x, \quad \sin x, \quad \cos 2x, \quad \sin 2x, \cdots, \quad \cos nx, \quad \sin nx, \cdots.$$

All these functions have the period 2π. They form the so-called **trigonometric system**. Figure 259 shows the first few of them (except for the constant 1, which is periodic with any period).



Cosine and sine functions having the period 2π (the first few members of the trigonometric system (3), except for the constant 1)

The series to be obtained will be a **trigonometric series**, that is, a series of the form

$$a_0 + a_1 \cos x + b_1 \sin x + a_2 \cos 2x + b_2 \sin 2x + \cdots$$

$$(4) \qquad = a_0 + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx).$$

$a_0, a_1, b_1, a_2, b_2, \cdots$ are constants, called the **coefficients** of the series. We see that each term has the period 2π. Hence *if the coefficients are such that the series converges, its sum will be a function of period* 2π.

Expressions such as (4) will occur frequently in Fourier analysis. To compare the expression on the right with that on the left, simply write the terms in the summation. Convergence of one side implies convergence of the other and the sums will be the same.

Now suppose that $f(x)$ is a given function of period 2π and is such that it can be **represented** by a series (4), that is, (4) converges and, moreover, has the sum $f(x)$. Then, using the equality sign, we write

$$\textbf{(5)} \qquad f(x) = a_0 + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx)$$

and call (5) the **Fourier series** of $f(x)$. We shall prove that in this case the coefficients of (5) are the so-called **Fourier coefficients** of $f(x)$, given by the **Euler formulas**

$$\textbf{(0)} \qquad a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x)\, dx$$

$$\textbf{(6)} \qquad \textbf{(a)} \qquad a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos nx\, dx \qquad n = 1, 2, \cdots$$

$$\textbf{(b)} \qquad b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin nx\, dx \qquad n = 1, 2, \cdots.$$

Complex Fourier series

We saw how a function can be expanded in a series of sines and cosines. These functions are related to the complex exponential function via Eulers formula

$$e^{ix} = \cos x + i \sin x$$

where i is the imaginary unit with the property that $i^2 = -1$. Because the algebraic properties of the exponential function are much simpler than those of the cos and sin, it is often an advantage to work with complex numbers, even though the given setting is real numbers. This is definitely the case in Fourier analysis.

$$\cos(2\pi nt/T) = \frac{1}{2}\left(e^{2\pi int/T} + e^{-2\pi int/T}\right)$$
$$\sin(2\pi nt/T) = \frac{1}{2i}\left(e^{2\pi int/T} - e^{-2\pi int/T}\right)$$

From these identities it is clear that the set of complex ex-ponential functions $e^{2\pi int/T}$ also is a basis of periodic functions (with the same period) for $V_{N,T}$. We may therefore reformulate Definition 2.3 as follows:

Definition (Complex Fourier basis)**.** We define the set of functions

$$\mathcal{F}_{N,T} = \{e^{-2\pi iNt/T}, e^{-2\pi i(N-1)t/T}, \cdots, e^{-2\pi it/T},$$
$$1, e^{2\pi it/T}, \cdots, e^{2\pi i(N-1)t/T}, e^{2\pi iNt/T}\},$$

and call this the order N complex Fourier basis for $V_{N,T}$.

The function $e^{2\pi int/T}$ is also called a pure tone with frequency n/T, just as for sines and cosines. We would like to show that these functions also are orthogonal. To show this, we need to say more on the inner product we have defined by (2.1). A weakness with this definition is that we have assumed real functions f and g, so that this can not be used for the complex exponential functions $e^{2\pi int/T}$. For general complex functions we will extend the definition of the inner product as follows:

$$\langle f, g\rangle = \frac{1}{T}\int_0^T f\bar{g}\, dt.$$

The associated norm now becomes

$$\|f\| = \sqrt{\frac{1}{T}\int_0^T |f(t)|^2 dt}.$$

The motivation behind Equation 2.14, where we have conjugated the second function, lies in the definition of an *inner product for vector spaces over complex numbers.*

From before we are used to vector spaces over real numbers, but vector spaces over complex numbers are defined through the same set of axioms as for real vector spaces, only replacing real numbers with complex numbers. For complex vector spaces, the axioms defining an inner product are the same as for real vector spaces, except for that the axiom

$$\langle f, g\rangle = \langle g, f\rangle \tag{2.16}$$

is replaced with the axiom

$$\langle f, g\rangle = \overline{\langle g, f\rangle}, \tag{2.17}$$

i.e. a conjugation occurs when we switch the order of the functions. This new axiom can be used to prove the property $\langle f, cg\rangle = \bar{c}\langle f, g\rangle$, which is a somewhat different property from what we know for real inner product spaces. This property follows by writing

$$\langle f, cg\rangle = \overline{\langle cg, f\rangle} = \overline{c\langle g, f\rangle} = \bar{c}\overline{\langle g, f\rangle} = \bar{c}\langle f, g\rangle.$$

Clearly the inner product 2.14 satisfies Axiom 2.17. With this definition it is quite easy to see that the functions $e^{2\pi int/T}$ are orthonormal. Using the orthogonal decomposition theorem we can therefore write

$$\begin{aligned} f_N(t) &= \sum_{n=-N}^{N} \frac{\langle f, e^{2\pi int/T}\rangle}{\langle e^{2\pi int/T}, e^{2\pi int/T}\rangle} e^{2\pi int/T} = \sum_{n=-N}^{N} \langle f, e^{2\pi int/T}\rangle e^{2\pi int/T} \\ &= \sum_{n=-N}^{N} \left(\frac{1}{T}\int_0^T f(t)e^{-2\pi int/T}dt\right) e^{2\pi int/T}. \end{aligned}$$

Theorem We denote by $y_{-N}, \ldots, y_0, \ldots, y_N$ the coordinates of f_N in the basis $\mathcal{F}_{N,T}$, i.e.

$$f_N(t) = \sum_{n=-N}^{N} y_n e^{2\pi int/T}. \tag{2.18}$$

The y_n are called the complex Fourier coefficients of f, and they are given by.

$$y_n = \langle f, e^{2\pi int/T}\rangle = \frac{1}{T}\int_0^T f(t)e^{-2\pi int/T}dt. \tag{2.19}$$

If we reorder the real and complex Fourier bases so that the two functions $\{\cos(2\pi nt/T), \sin(2\pi nt/T)\}$ and $\{e^{2\pi int/T}, e^{-2\pi int/T}\}$ have the same index in the bases, equations (2.10)-(2.11) give us that the change of basis matrix[4] from

$\mathcal{D}_{N,T}$ to $\mathcal{F}_{N,T}$, denoted $P_{\mathcal{F}_{N,T}\leftarrow\mathcal{D}_{N,T}}$, is represented by repeating the matrix

$$\frac{1}{2}\begin{pmatrix}1 & 1/i\\ 1 & -1/i\end{pmatrix}$$

along the diagonal (with an additional 1 for the constant function 1). In other words, since a_n, b_n are coefficients relative to the real basis and y_n, y_{-n} the corresponding coefficients relative to the complex basis, we have for $n > 0$,

$$\begin{pmatrix}y_n\\ y_{-n}\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1 & 1/i\\ 1 & -1/i\end{pmatrix}\begin{pmatrix}a_n\\ b_n\end{pmatrix}.$$

This can be summarized by the following theorem:

Theorem (Change of coefficients between real and complex Fourier bases)**.** The complex Fourier coefficients y_n and the real Fourier coefficients a_n, b_n of a function f are related by

$$\begin{aligned} y_0 &= a_0,\\ y_n &= \frac{1}{2}(a_n - ib_n),\\ y_{-n} &= \frac{1}{2}(a_n + ib_n), \end{aligned}$$

for $n = 1, \ldots, N$.

Combining Theorems can help us state properties of complex Fourier coefficients for symmetric- and antisymmetric functions.

Due to the somewhat nicer formulas for the complex Fourier coefficients when compraed to the real Fourier coefficients, we will write most Fourier series in complex form in the following.

The Discrete Fourier Transform

Fourier analysis for finite vectors is focused around mapping a given vector from the standard basis to the Fourier basis, performing some operations on the Fourier representation, and then changing the result back to the standard basis. The Fourier matrix, which represents this change of basis, is therefore of crucial importance, and in this section we study some of its basic properties. We start by defining the Fourier matrix.

Definition (Discrete Fourier Transform)**.** The change of coordinates from the standard basis of $\mathbb{R}^N$ to the Fourier basis $\mathcal{F}_N$ is called the discrete Fourier transform (or DFT). The $N \times N$ matrix F_N that represents this change of basis is called the (N-point) Fourier matrix. If $\boldsymbol{x}$ is a vector in R^N, its coordinates $\boldsymbol{y} = (y_0, y_1, \ldots, y_{N-1})$ relative to the Fourier basis are called the Fourier coefficients of $\boldsymbol{x}$, in other words $\boldsymbol{y} = F_N \boldsymbol{x}$). The DFT of $\boldsymbol{x}$ is sometimes denoted by $\hat{\boldsymbol{x}}$.

We will normally write $\boldsymbol{x}$ for the given vector in $\mathbb{R}^N$, and $\boldsymbol{y}$ for the DFT of this vector. In applied fields, the Fourier basis vectors are also called *synthesis vectors*, since they can be used used to "synthesize" the vector $\boldsymbol{x}$, with weights provided by the DFT coefficients $\boldsymbol{y} = (y_n)_{n=0}^{N-1}$. To be more precise, we have that the change of coordinates performed by the DFT can be written as

$$\boldsymbol{x} = y_0\boldsymbol{\phi}_0 + y_1\boldsymbol{\phi}_1 + \cdots + y_{N-1}\boldsymbol{\phi}_{N-1} = \begin{pmatrix}\boldsymbol{\phi}_0 & \boldsymbol{\phi}_1 & \cdots & \boldsymbol{\phi}_{N-1}\end{pmatrix}\boldsymbol{y} = F_N^{-1}\boldsymbol{y}, \quad (3.3)$$

where we have used the inverse of the defining relation $\boldsymbol{y} = F_N\boldsymbol{x}$, and that the $\boldsymbol{\phi}_n$ are the columns in F_N^{-1} (this follows from the fact that F_N^{-1} is the change of coordinates matrix from the Fourier basis to the standard basis, and the Fourier basis vectors are clearly the columns in this matrix). Equation (3.3) is also called the synthesis equation.

Let us also find the matrix F_N itself. From Lemma 3.4 we know that the columns of F_N^{-1} are orthonormal. If the matrix was real, it would have been called orthogonal, and the inverse matrix could be obtained by transposing. F_N^{-1} is complex however, and it is easy to see that the conjugation present in the definition of the inner product (3.1) translates into that the inverse of a complex matrix with orthonormal columns is given by the matrix where the entries are both transposed and conjugated. Let us denote the conjugated transpose of T by T^H, and say that a complex matrix is unitary when $T^{-1} = T^H$. From our discussion it is clear that F_N^{-1} is a unitary matrix, i.e. its inverse, F_N, is its conjugate transpose. Moreover since F_N^{-1} is symmetric, its inverse is in fact just its conjugate,

$$F_N = \overline{F_N^{-1}}.$$

Theorem The Fourier matrix F_N is the unitary $N \times N$-matrix with entries given by

$$(F_N)_{nk} = \frac{1}{\sqrt{N}} e^{-2\pi i n k/N},$$

for $0 \leq n, k \leq N-1$.

Note that in the signal processing literature, it is not common to include the normalizing factor $1/\sqrt{N}$ in the definition of the DFT. From our more mathematical point of view this is useful since it makes the Fourier matrix unitary.

In practical applications of Fourier analysis one typically applies the DFT, performs some operations on the coefficients, and then maps the result back using the inverse Fourier matrix. This inverse transformation is so common that it deserves a name of its own.

Symmetric digital filters and the DCT

We have mentioned that most periodic functions can be approximated well by Fourier series on the form $\sum_n y_n e^{2\pi i n t/T}$. The convergence speed of this Fourier series may be slow however, and we mentioned that it depends on the regularity of the function. In particular, the convergence speed is higher when the function is continuous. For $f \in C[0,T]$ where $f(0) \neq f(T)$, we do not get a continuous function if we repeat the function in periods. However, we demonstrated that we could create a symmetric extension g, defined on $[0,\, 2T\,]$, so that $g(0) = g(2T)$. The Fourier series of g actually took the form of a cosine-series, and we saw that this series converged faster to g, than the Fourier series of f did to f.

In this section we will specialize this argument to vectors: by defining the symmetric extension of a vector, we will attempt to find a better approximation than we could with the Fourier basis vectors. This approach is useful for digital filters also, if the filters preserve these symmetric extensions. Let us summarize with the following idea:

Idea (Increasing the convergence speed of the DFT)**.** Assume that we have a function f, and that we take the samples $x_k = f(kT/N)$. From $\boldsymbol{x} \in \mathbb{R}^N$ we would like to create an extension $\breve{\boldsymbol{x}}$ so that the first and last values are equal. For such an extension, the Fourier basis vectors can give a very good approximation to f.

As our candidate for the extension of $\boldsymbol{x}$, we will consider the following:

Definition (Symmetric extension of a vector)**.** By the symmetric exten-sion of $\boldsymbol{x} \in \mathbb{R}^N$, we mean $\breve{\boldsymbol{x}} \in \mathbb{R}^{2N}$ defined by

$$\breve{\boldsymbol{x}}_k = \begin{cases} x_k & 0 \leq k < N \\ x_{2N-1-k} & N \leq k < 2N-1 \end{cases} \tag{3.27}$$

We say that a vector in $\mathbb{R}^{2N}$ is symmetric if it can be written as the symmetric extension of a vector in $\mathbb{R}^N$.

The symmetric extension $\breve{\boldsymbol{x}}$ thus has the original vector $\boldsymbol{x}$ as its first half, and a copy of $\boldsymbol{x}$ in reverse order as its second half. Clearly, the first and last values of $\breve{\boldsymbol{x}}$ are equal. In other words, a vector in $\mathbb{R}^{2N}$ is a symmetric extension of a vector in $\mathbb{R}^N$ if and only if it is symmetric about $N - \frac{1}{2}$. Clearly also the set of symmetric extensions is a vector space. Our idea in terms of filters is the following:

Idea (Increasing the precision of a digital filter)**.** If a filter maps a symmetric extension of one vector to a symmetric extension of another vector then it is a good approximation of an analog version in terms of Fourier series.

We will therefore be interested in finding filters which preserves symmetric extensions. We will show the following, which characterize such filters:

Theorem (Characterization of filters which preserve symmetric exten-sions)**.** A digital filter S of size $2N$ preverves symmetric extensions if and only if S is symmetric. Moreover, S is uniquely characterized by its restriction to $\mathbb{R}^N$, denoted by S_r, which is given by $S_1 + (S_2)^f$, where $S = \begin{pmatrix} S_1 & S_2 \\ S_3 & S_4 \end{pmatrix}$, and where $(S_2)^f$ is the matrix S_2 with the columns reversed. Moreover, if we define

$$d_{n,N} = \begin{cases} \sqrt{\frac{1}{N}} & , n = 0 \\ \sqrt{\frac{2}{N}} & , 1 \leq n < N \end{cases}$$

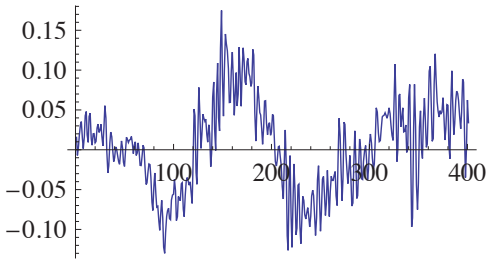
and $\boldsymbol{d}_n = d_{n,N} \cos\left(2\pi \frac{n}{2N}\left(k + \frac{1}{2}\right)\right)$ for $0 \leq n \leq N-1$, then $\{\boldsymbol{d}_0, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_{N-1}\}$ is an orthonormal basis of eigenvectors for S_r.

Use of DCT in lossy compression of sound

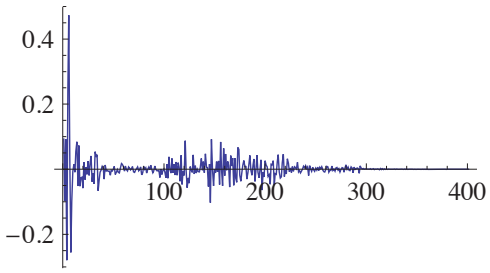
The DCT is particularly popular for processing the sound before compression. MP3 is based on applying a variant of the DCT (called the Modified Discrete Cosine Transform, MDCT) to groups of 576 (in special circumstances 192) samples. One does not actually apply the DCT directly. Rather one applies a much more complex transformation, which can be implemented in parts by using DCT in a clever way.

We mentioned previously that we could achieve compression by setting the Fourier coefficients which are small to zero. Translated to the DCT, we should set the DCT coefficients which are small to zero, and we apply the inverse DCT in order to reconstruct the signal in order to play it again. Let us test compression based on this idea. The plots in figure 3.10 illustrate the principle. A signal is shown in (a) and its DCT in (b). In (d) all values of the DCT with absolute value smaller than 0.02 have been set to zero. The signal can then be reconstructed with the inverse DCT of theorem **??**; the result of this is shown in (c). The two signals in (a) and (c) visually look almost the same even though the signal in (c) can be represented with less than 25 % of the information present in (a).

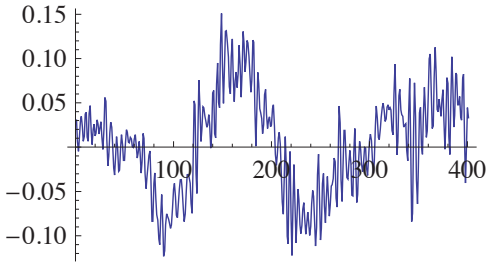
Opposed to lossless compression, lossy compression reduces perceptual redundancy; i.e. sounds which are considered perceptually irrelevant are coded with decreased accuracy or not coded at all.



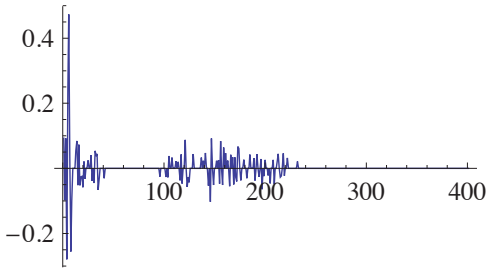
(a)



(b)



(c)



(d)

The signal in (a) are the sound samples from a small part of a song. The plot in (b) shows the DCT of the signal. In (d), all values of the DCT that are smaller than 0.02 in absolute value have been set to 0, a total of 309 values. In (c) the signal has been reconstructed from these perturbed values of the DCT. The values have been connected by straight lines to make it easier to interpret the plots.

We test this compression strategy on a data set that consists of 300 001 points. We compute the DCT and set all values smaller than a suitable tolerance to 0. With a tolerance of 0.04, a total of 142 541 values are set to zero. When we then reconstruct the sound with the inverse DCT, we obtain a signal that differs at most 0.019 from the original signal. To verify that the new file is not too different from the old file, we can take the read sound samples from `castanets.wav`, run the following function for different `eps`

```
function A=skipsmallvals(eps,A)
  B=dct(A);
  B=(B>=eps).*B;
  A=invdct(B);
```

and play the new samples. Finally we can store the signal by storing a `gzip`'ed version of the DCT-values (as 32-bit floating-point numbers) of the perturbed signal. This gives a file with 622 551 bytes, which is 88 % of the `gzip`'ed version of the original data.
The choice of the DCT in the MP3 standard has much to do with that the DCT, just as the DFT, has a very efficient implementation.

Summary

We defined the Fourier Series, Complex Fourier Series, Discrete Fourier transform, and Discrete Cosine transform. We exploited properties of the DFT, which corresponded nicely to the corresponding properties for Fourier series. We defined digital filters, which turned out to be linear transformations diagonalized by the DFT. Also we showed the techniques to speed up the convergence of the Fourier series, could also be used for the DFT. And thus we learned how many audio compression softwares make use of psychoacoustic property of the human auditory system by croping off the inaudible frequencies and reduce the bitrate of the less sensitive sound signals.