8th exam item

András Mátyás Biricz

June 23, 2019

Abstract

Jelfeldolgozás és idősor-analízis – Fourier-módszerek, FFT, a spektrum és a spektrogram, az átviteli és ablakfüggvények, Wiener-szűrő. Korrelációs függvények, a Wiener–Hincsin-tétel és a teljesítményspektrum. Konvolúció és dekonvolúció. Szűrők analóg és digitális megvalósítása, RLC-körök, FIR- és IIR-szűrők.

Signal processing and analysis of time series - Fourier methods, FFT, spectrum and spectrogram, transfer- and window functions, Wiener-filter. Correlation functions, Wiener-Khinchin theorem, power spectrum. Convolution, deconvolution. Realization of analog and digital filters, RLC-circuit, FIR and IIR filters.

1 Introduction

Signal processing is a subfield of electrical engineering that concerns the analysis, synthesis, and modification of signals[1] in order to extract information of phenomena through processing of audio, video and the outcome of various measurements. Signal processing techniques can be used to improve signal transmission, emphasize or detect interesting components of measured signal and to improve storage efficiency.

As a consequence of the technological advancement tremendous amount of data can be measured in various fields of science and it becomes even bigger challenge to deal with that vast amount of data. Fortunately, the application of the signal processing methods can help.

2 Fourier methods

Fourier methods play a key role in analysing signals, especially in case of time series data. These methods involve the Fourier-series and Fourier transform. As a result we can acquire valuable information as a (frequency) spectrum. In addition, these can be applicable more broadly, even to analyse spatial properties of 2 or 3 dimensional signals or to solve differential equations. Last but not least, it comes in handy in calculation of correlation and convolution.

2.1 Fourier series

A Fourier series is an expansion of a periodic function $f(x)$ in terms of an infinite sum of sines and cosines. Fourier series make use of the orthogonality relationships of the sine and cosine functions. The computation and study of Fourier series is known as harmonic analysis and is extremely useful as a way to break up an arbitrary periodic function into a set of simple terms that can be plugged in, solved individually, and then recombined to obtain the solution to the original problem or an approximation to it to whatever accuracy is desired or practical.

In particular, since the superposition principle holds for solutions of a linear homogeneous ordinary differential equation, if such an equation can be solved in the case of a single sinusoid, the solution for an arbitrary function is immediately available by expressing the original function as a Fourier series and then plugging in the solution for each sinusoidal component. In some special cases where the Fourier series can be summed in closed form, this technique can even yield analytic solutions.

[1] https://en.wikipedia.org/wiki/Signal_processing.

In addition, any set of functions that form a complete orthogonal system has a corresponding generalized Fourier series analogous to the Fourier series. For example, using orthogonality of the roots of a Bessel function of the first kind gives a so-called Fourier-Bessel series.
The computation of the Fourier series is based on integral identities (follow source link to view). For a 2π periodic function $f(x)$, the series expansion is the following:

$$f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n cos(nx) + \sum_{n=1}^{\infty} b_n sin(nx), \tag{1}$$

where

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\mathrm{d}x, \tag{2}$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)cos(nx)\mathrm{d}x, \tag{3}$$

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)sin(nx)\mathrm{d}x, \tag{4}$$

where a_0 is the coefficient of the constant term, a_n and b_n are the coefficients of the periodic terms and $n = 1, 2, 3, ...$.
The notion of a Fourier series can also be extended to complex coefficients. In case of a real-valued function $f(x)$:

$$f(x) = \sum_{n=-\infty}^{\infty} A_n e^{inx}. \tag{5}$$

It can be showed (follow source), that:

$$A_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{-inx}\mathrm{d}x, \tag{6}$$

which are coefficients hiding the a_0, a_n, b_n coefficients in a compact form.
Note, that not only 2π periodic functions can be expanded in a Fourier series, it can be generalized to the case of a function periodic in $[-L/2, L/2]$:

$$f(x) = \sum_{n=-\infty}^{\infty} A_n e^{i2\pi nx/L}, \tag{7}$$

$$A_n = \frac{1}{L}\int_{-L/2}^{L/2} f(x)e^{i2\pi nx/L}\mathrm{d}x. \tag{8}$$

The source and more information with examples can be found by following this link[2].

[2] http://mathworld.wolfram.com/FourierSeries.html

2.2 Fourier transform

The Fourier transform is a generalization of the complex Fourier series in the limit as $L \to \infty$. Replace the discrete A_n with the continuous $F(k)dk$ while letting $n/L \to k$. Then change the sum to an integral, and the equations become:

$$f(x) = \int_{-\infty}^{\infty} F(k)e^{i2\pi kx}\mathrm{d}k, \tag{9}$$

$$f(k) = \int_{-\infty}^{\infty} f(x)e^{-i2\pi kx}\mathrm{d}x. \tag{10}$$

Note that some authors (especially physicists) prefer to write the transform in terms of angular frequency $\omega \equiv 2\pi\nu$) instead of the oscillation frequency (ν). However, this destroys the symmetry, resulting in the transform pair:

$$H(\omega) = \mathcal{F}[h(t)] = \int_{-\infty}^{\infty} h(t)e^{-i\omega t}\mathrm{d}t, \tag{11}$$

$$h(t) = \mathcal{F}^{-1}[H(\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} H(\omega)e^{i\omega t}\mathrm{d}\omega. \tag{12}$$

A function $f(x)$ has a forward and inverse Fourier transform if:

- $\int_{-\infty}^{\infty} |f(x)|\mathrm{d}x$ exists,
- there are a finite number of discontinuities,
- the function has bounded variation. (A sufficient weaker condition is fulfillment of the Lipschitz condition.[3])

The Fourier transform is linear, which means the following for functions $f(x)$ and $g(x)$:

$$\mathcal{F}[af(x) + bg(x)] = a\mathcal{F}[f(x)] + b\mathcal{F}[g(x)] = aF(k) + bG(k). \tag{13}$$

The source and more information with examples can be found by following this link[4].

2.3 Fast Fourier Transform

A fast Fourier transform (FFT) is an algorithm that computes the discrete Fourier transform (DFT) of a sequence, or its inverse (IDFT). The DFT is obtained by decomposing a sequence of values into components of different frequencies. This operation is useful in many fields, but computing it directly from the definition is often too slow to be practical. An FFT rapidly computes such transformations by factorizing the DFT matrix into a product of sparse (mostly zero) factors. As a result, it manages to reduce the complexity of computing the DFT from $\mathcal{O}(N^2)$, which arises if one simply applies the definition of DFT, to $\mathcal{O}(NlogN)$, where N is the data size. The difference in speed can be enormous, especially for long data sets where N may be in the thousands or millions. In the presence of round-off error, many FFT algorithms are much more accurate than evaluating the DFT definition directly. There are many different

[3] http://mathworld.wolfram.com/LipschitzCondition.html.

[4] http://mathworld.wolfram.com/FourierTransform.html

FFT algorithms based on a wide range of published theories, from simple complex-number arithmetic to group theory and number theory.[5]

Let x_0, ..., x_{N-1} be complex numbers. The DFT is defined by the formula:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}, \quad k = 0, \dots, N-1, \tag{14}$$

where $e^{i2\pi/N}$ is a primitive N^{th} root of 1.

Evaluating this definition directly requires $\mathcal{O}(N^2)$ operations: there are N outputs X_k, and each output requires a sum of N terms. An FFT is any method to compute the same results in $\mathcal{O}(NlogN)$ operations. All known FFT algorithms require $\mathcal{O}(NlogN)$ operations, although there is no known proof that a lower complexity score is impossible.

To illustrate the savings of an FFT, consider the count of complex multiplications and additions for $N = 4096$ data points. Evaluating the DFT's sums directly involves N^2 complex multiplications and $N(N-1)$ complex additions, of which $\mathcal{O}(N)$ operations can be saved by eliminating trivial operations such as multiplications by 1. On the other hand, the Radix-2 (James W. Cooley and John W. Tukey (1965)) algorithm, for N a power of 2, can compute the same result with only $N/2log_2N$ complex multiplications (again, ignoring simplifications of multiplications by 1 and similar) and $Nlog_2N$ complex additions. In practice, actual performance on modern computers is usually dominated by factors other than the speed of arithmetic operations and the analysis is a complicated subject, but the overall improvement from $\mathcal{O}(N^2)$ to $\mathcal{O}(NlogN)$ remains.

A Radix-2 decimation-in-time (DIT) FFT is the simplest and most common form of the Cooley–Tukey algorithm, although highly optimized Cooley–Tukey implementations typically use other forms of the algorithm as described below. Radix-2 DIT divides a DFT of size N into two interleaved DFTs (hence the name "Radix-2") of size $N/2$ with each recursive stage.

The discrete Fourier transform (DFT) is defined by the formula:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk}, \tag{15}$$

where k is an integer ranging from 0 to $N-1$.

Radix-2 DIT first computes the DFTs of the even-indexed inputs ($x_{2m} = x_0, x_2, \dots, x_{N-2}$) and of the odd-indexed inputs ($x_{2m+1} = x_1, x_3, \dots, x_{N-1}$), and then combines those two results to produce the DFT of the whole sequence. This idea can then be performed recursively to reduce the overall runtime to $\mathcal{O}(NlogN)$. This simplified form assumes that N is a power of two, since the number of sample points N can usually be chosen freely by the application (e.g. by changing the sample rate or window, zero-padding, etcetera), this is often not an important restriction.

The Radix-2 DIT algorithm rearranges the DFT of the function x_n into two parts: a sum over the even-numbered indices ($n = 2m$) and a sum over the odd-numbered indices ($n = 2m+1$):

$$X_k \quad = \quad \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i}{N}(2m)k} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i}{N}(2m+1)k}. \tag{16}$$

One can factor a common multiplier $e^{-\frac{2\pi i}{N}k}$ out of the second sum, as shown in the equation below. It is then clear that the two sums are the DFT of the even-indexed part x_{2m} and the

[5] https://en.wikipedia.org/wiki/Fast_Fourier_transform.

DFT of odd-indexed part x_{2m+1} of the function x_n. Denote the DFT of the even-indexed inputs x_{2m} by E_k and the DFT of the odd-indexed inputs x_{2m+1} by O_k and we obtain:

$$X_k = \underbrace{\sum_{m=0}^{N/2-1} x_{2m}e^{-\frac{2\pi i}{N/2}mk}}_{\text{DFT of even}-\text{indexed part of } x_n} + e^{-\frac{2\pi i}{N}k} \underbrace{\sum_{m=0}^{N/2-1} x_{2m+1}e^{-\frac{2\pi i}{N/2}mk}}_{\text{DFT of odd}-\text{indexed part of } x_n} = E_k + e^{-\frac{2\pi i}{N}k}O_k. \tag{17}$$

Thanks to the periodicity of the complex exponentials, $X_{k+\frac{N}{2}}$ is also obtained from E_k and O_k:

$$\begin{aligned} X_{k+\frac{N}{2}} &= \sum_{m=0}^{N/2-1} x_{2m}e^{-\frac{2\pi i}{N/2}m(k+\frac{N}{2})} + e^{-\frac{2\pi i}{N}(k+\frac{N}{2})} \sum_{m=0}^{N/2-1} x_{2m+1}e^{-\frac{2\pi i}{N/2}m(k+\frac{N}{2})} \\ &= \sum_{m=0}^{N/2-1} x_{2m}e^{-\frac{2\pi i}{N/2}mk}e^{-2\pi mi} + e^{-\frac{2\pi i}{N}k}e^{-\pi i} \sum_{m=0}^{N/2-1} x_{2m+1}e^{-\frac{2\pi i}{N/2}mk}e^{-2\pi mi} \\ &= \sum_{m=0}^{N/2-1} x_{2m}e^{-\frac{2\pi i}{N/2}mk} - e^{-\frac{2\pi i}{N}k} \sum_{m=0}^{N/2-1} x_{2m+1}e^{-\frac{2\pi i}{N/2}mk} \\ &= E_k - e^{-\frac{2\pi i}{N}k}O_k. \end{aligned} \tag{18}$$

We can rewrite X_k as:

$$\begin{aligned} X_k &= E_k + e^{-\frac{2\pi i}{N}k}O_k, \\ X_{k+\frac{N}{2}} &= E_k - e^{-\frac{2\pi i}{N}k}O_k. \end{aligned} \tag{19}$$

This result, expressing the DFT of length N recursively in terms of two DFTs of size $N/2$, is the core of the Radix-2 DIT fast Fourier transform. The algorithm gains its speed by re-using the results of intermediate computations to compute multiple DFT outputs. Note that final outputs are obtained by a ± combination of E_k and $O_k \exp(-2\pi ik/N)$, which is simply a size-2 DFT (sometimes called a butterfly diagram in this context (see figure 1)); when this is generalized to larger radices, the size-2 DFT is replaced by a larger DFT (which itself can be evaluated with an FFT).
The source and more information with examples can be found by following these links[6][7].

2.4 Spectrum, spectrogram

The spectrum is the (frequency) domain representation of a signal. A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time (see figures 2 and 3). When applied to an audio signal, spectrograms are sometimes called sonographs, voiceprints, or voicegrams. When the data is represented in a 3D plot they may be called waterfalls. Spectrograms are used extensively in the fields of music, sonar, radar, and speech processing, seismology, and others. Spectrograms of audio can be used to identify spoken words phonetically, and to analyse the various calls of animals.

[6] https://en.wikipedia.org/wiki/Cooley%E2%80%93Tukey_FFT_algorithm,
[7] https://en.wikipedia.org/wiki/Fast_Fourier_transform.

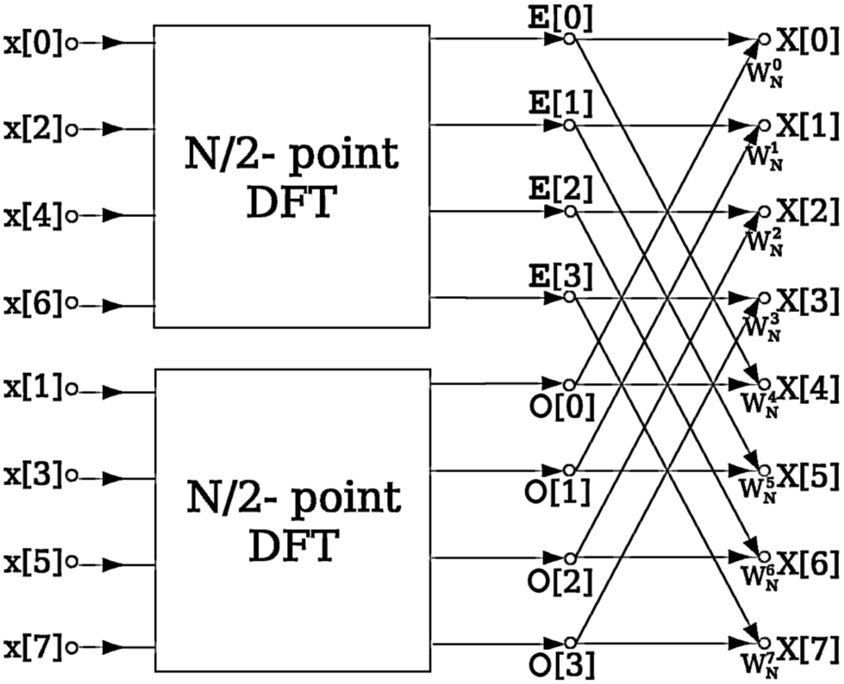

Figure 1: Data flow diagram for $N = 8$: a decimation-in-time radix-2 FFT breaks a length-N DFT into two length-$N/2$ DFTs followed by a combining stage consisting of many size-2 DFTs called "butterfly" operations (so-called because of the shape of the data-flow diagrams). The source was accessed in June 2019 (https://en.wikipedia.org/wiki/File:DIT-FFT-butterfly.png).

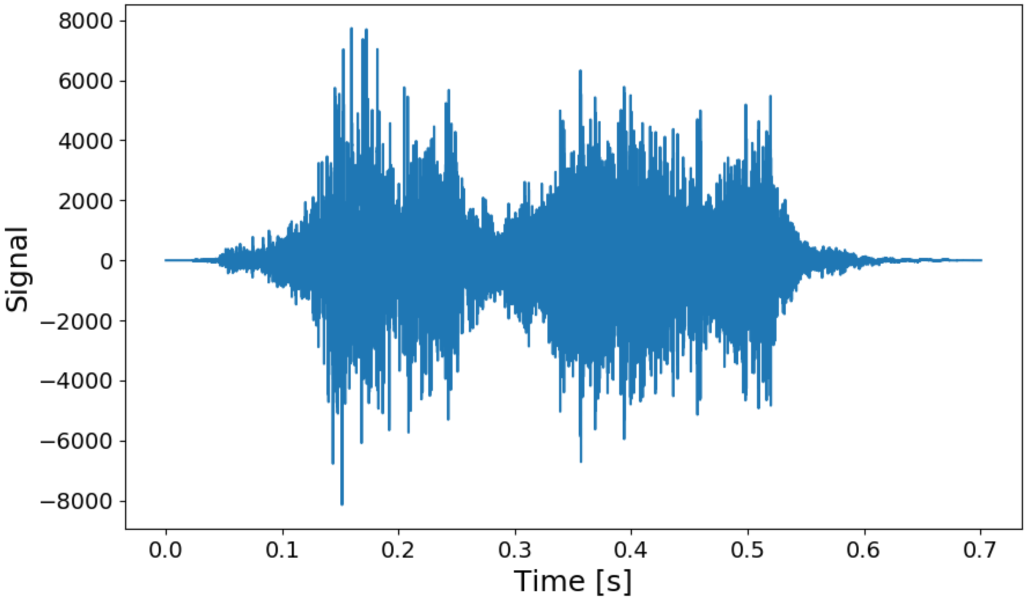

Figure 2: Yes sound. Source: https://www.pacdv.com/sounds/voices/yes-2.wav.

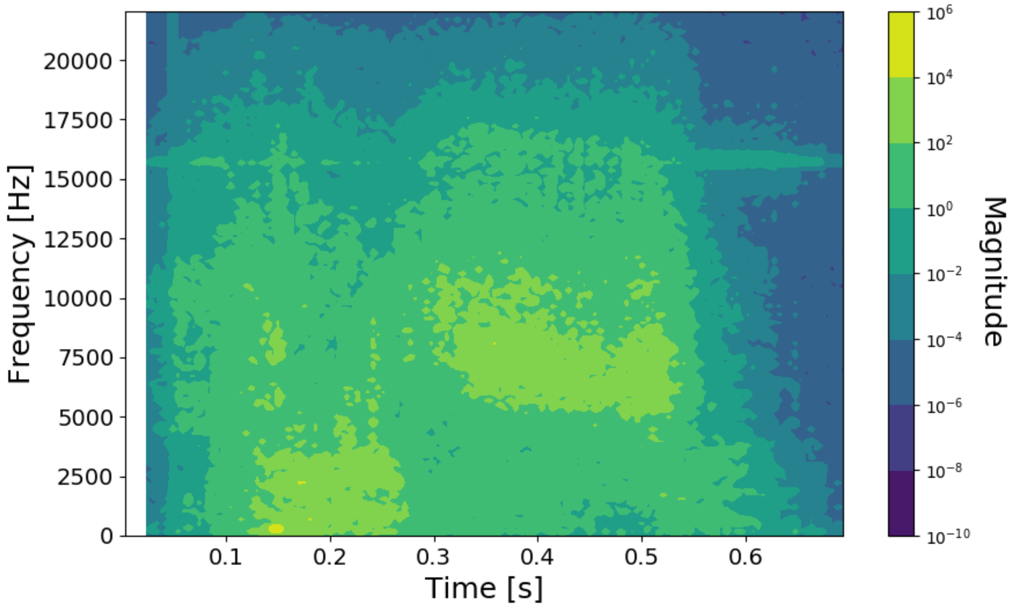

Figure 3: Spectrogram of the yes sound.

3 Correlation functions

4 Convolution

5 Transfer and window functions

6 Filters

6.1 Analog

6.2 Digital

6.3 RLC circuit

6.4 FIR

6.5 IIR

6.6 Wiener-filter

7 Conclusion

References

[1] James W. Cooley and John W. Tukey. An Algorithm for the Machine Calculation of Complex Fourier Series *Math. Comp.* **19** (1965), 297-301. https://www.ams.org/journals/mcom/1965-19-090/S0025-5718-1965-0178586-1/S0025-5718-1965-0178586-1.pdf