Non-stationary signals. Windowing and localisation. Short Time Fourier Transform. Spectrograms.

Mathematical Tools for ITS (11MAI)

Mathematical tools, 2020

Jan Přikryl

11MAI, lecture 3

Monday, October 12, 2020

version: 2020-10-06 15:05

Department of Applied Mathematics, CTU FTS

Lectue Contents

The derivatives and integrals (as primitive functions) of trigonometric functions are interconnected:

$$\frac{\mathrm{d}}{\mathrm{d}x}\sin \ell x = \ell \cos \ell x \Rightarrow \int \cos \ell x \,\mathrm{d}x = \frac{1}{\ell}\sin \ell x,$$
$$\frac{\mathrm{d}}{\mathrm{d}x}\cos \ell x = -\ell \sin \ell x \Rightarrow \int \sin \ell x \,\mathrm{d}x = -\frac{1}{\ell}\cos \ell x.$$

Product of trigonometric functions

Products of two trigonometric functions are expressible as

$$2\sin \ell x \sin mx = \cos(\ell - m)x - \cos(\ell + m)x,$$
$$2\cos \ell x \cos mx = \cos(\ell - m)x + \cos(\ell + m)x,$$
$$2\sin \ell x \cos mx = \sin(\ell - m)x + \sin(\ell + m)x.$$

Note

If $x \in [0, 2\pi)$ then for $x = \omega_0 t$ we have $t \in [0, T)$.

We have learnt that trigonometric functions $\cos \omega_k t$ and $\sin \omega_k t$ form Fourier basis for T-periodic functions.

Question

Is the basis set of $\cos mx$ and $\sin mx$ for $x \in [0, 2\pi)$ orthogonal?

Assume $\ell, m \in \mathbb{N}$.

We will study the scalar inner products of these functions for $\ell \neq m$ first:

$$\begin{aligned}
\langle \cos \ell x, \cos mx \rangle &= \int_0^{2\pi} \cos \ell x \cos mx \, \mathrm{d}x \\
&= \frac{1}{2} \int_0^{2\pi} \cos(\ell - m)x \, \mathrm{d}x + \frac{1}{2} \int_0^{2\pi} \cos(\ell + m)x \, \mathrm{d}x \\
&= \frac{1}{2(\ell - m)} \Big[\sin(\ell - m)x \Big]_0^{2\pi} + \frac{1}{2(\ell + m)} \Big[\sin(\ell + m)x \Big]_0^{2\pi} \\
&= \frac{0 - 0}{2(\ell - m)} + \frac{0 - 0}{2(\ell + m)} = 0
\end{aligned}$$

$$
\begin{aligned}
\langle \sin \ell x, \sin mx \rangle &= \int_0^{2\pi} \sin \ell x \sin mx \, \mathrm{d}x \\
&= \frac{1}{2} \int_0^{2\pi} \cos(\ell - m)x \, \mathrm{d}x - \frac{1}{2} \int_0^{2\pi} \cos(\ell + m)x \, \mathrm{d}x \\
&= \frac{1}{2(\ell - m)} \Big[\sin(\ell - m)x \Big]_0^{2\pi} - \frac{1}{2(\ell + m)} \Big[\sin(\ell + m)x \Big]_0^{2\pi} \\
&= \frac{0 - 0}{2(\ell - m)} - \frac{0 - 0}{2(\ell + m)} = 0
\end{aligned}
$$

Orthogonal basis

$$
\begin{aligned}
\langle \sin \ell x, \cos mx \rangle &= \int_0^{2\pi} \sin \ell x \cos mx \, \mathrm{d}x \\
&= \frac{1}{2} \int_0^{2\pi} \sin(\ell - m)x \, \mathrm{d}x + \frac{1}{2} \int_0^{2\pi} \sin(\ell + m)x \, \mathrm{d}x \\
&= -\frac{1}{2(\ell - m)} \Big[\cos(\ell - m)x \Big]_0^{2\pi} - \frac{1}{2(\ell + m)} \Big[\cos(\ell + m)x \Big]_0^{2\pi} \\
&= -\frac{1 - 1}{2(\ell - m)} - \frac{1 - 1}{2(\ell + m)} = 0
\end{aligned}
$$

We will study the case $\ell = m$ separately.

$$\langle \sin mx, \cos mx \rangle = \frac{1}{2} \int_0^{2\pi} \sin 2mx \, \mathrm{d}x = -\frac{1}{4m} \Big[\cos 2mx \Big]_0^{2\pi}$$
$$= -\frac{1-1}{4m} = 0$$

Finally the two cases of basis functions that should result in inner product being 1 if normalised.

$$\langle \cos mx, \cos mx \rangle = \int_0^{2\pi} \cos^2 mx \, \mathrm{d}x = \int_0^{2\pi} \frac{1 + \cos 2mx}{2} \, \mathrm{d}x$$
$$= \frac{1}{2}\Big[x\Big]_0^{2\pi} + \frac{1}{2m}\Big[\sin 2mx\Big]_0^{2\pi}$$

$$||\cos mx||^2 = \pi \qquad \|\cos m\omega_0 t\|^2 = \frac{T}{2}$$

$$\langle \sin mx, \sin mx \rangle = \int_0^{2\pi} \sin^2 mx \, \mathrm{d}x = \int_0^{2\pi} \frac{1 - \cos 2mx}{2} \, \mathrm{d}x$$
$$= \frac{1}{2}\Big[x\Big]_0^{2\pi} - \frac{1}{2m}\Big[\sin 2mx\Big]_0^{2\pi}$$

$$||\sin mx||^2 = \pi \qquad \|\sin m\omega_0 t\|^2 = \frac{T}{2}$$

Lectue Contents

1. T-periodic signal $x(t)$ representation:

$$x(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos(k\omega_0 t) + \sum_{k=1}^{\infty} b_k \sin(k\omega_0 t)$$

2. basis vectors $\cos(k\omega_0 t)$, $\sin(k\omega_0 t)$
3. $a_0 = \dfrac{1}{T}\displaystyle\int_0^T x(t)\,\mathrm{d}t$, $a_k = \dfrac{(x(t), \cos(k\omega_0 t))}{(\cos(k\omega_0 t), \cos(k\omega_0 t))} \equiv \dfrac{2}{T}\displaystyle\int_0^T x(t)\cos(k\omega_0 t)\,\mathrm{d}t$
4. $b_k = \dfrac{(x(t), \sin(k\omega_0 t))}{(\sin(k\omega_0 t), \sin(k\omega_0 t))} \equiv \dfrac{2}{T}\displaystyle\int_0^T x(t)\sin(k\omega_0 t)\,\mathrm{d}t$

Continuous signal and basis vectors

1. T-periodic signal representation $x(t) = \sum_{k=-\infty}^{\infty} c_k \exp(j\, k\omega_0 t)$
2. basis vector $\phi_k(t) = \exp(j\, k\omega_0 t)$
3. scalar product $c_k = \dfrac{(x(t), \phi_k(t))}{(\phi_k(t), \phi_k(t))} \equiv \dfrac{1}{T}\int_0^T x(t) \exp(-j\, k\omega_0 t))dt$
4. completness of basis vectors

   $(\phi_k(t), \phi_\ell(t)) = \dfrac{1}{T}\int_0^T \exp(j\, k\omega_0 t) \exp(-j\, \ell\omega_0 t)dt = \delta_{k,\ell}$

Continuous signal and basis vectors

1. Fourier series $x(t) = \sum_{k=-\infty}^{\infty} c_k e^{j\omega_k t}$

2. Partial sum of Fourier Series $x_N(t) = \sum_{k=-M}^{M} c_k e^{j\omega_k t}$ for $N = 2M + 1$

Definition (Dirichlet kernel)

Dirichlet kernels are the partial sums of exponential functions

$$D_M(\omega_0 t) = \sum_{k=-M}^{M} \exp(j\, k\omega_0 t) = 1 + 2\sum_{k=1}^{M} \cos(k\omega_0 t).$$

Show that $D_M(\omega_0 t) = \dfrac{\sin((M+1/2)\omega_0 t)}{\sin(\omega_0 t/2)}$.

Theorem (Convolution of Dirichlet kernel)

The convolution of $D_M(t)$ with an arbitrary T-periodic function $f(t) = f(t+T)$ is the M-th degree Fourier series approximation to $f(t)$.

$$D_M(t) * f(t) \equiv \frac{1}{T} \int_{-T/2}^{T/2} D_M(t-\tau) f(\tau) d\tau = \sum_{k=-M}^{M} c_k \exp(j\, k\omega_0 t),$$

where $c_k = \dfrac{1}{T} \displaystyle\int_{-T/2}^{T/2} f(t) \exp(-j\, k\omega_0 t)\, dt.$

Lectue Contents

Consider a continuous signal $x(t)$ defined as T-periodical signal, sampled N times during that period at timestamps $t = nT/N$ for $n = 0, 1, 2, \ldots, N-1$. This yields a discretised signal

$$\mathbf{x} = (x_0, x_1, x_2, \ldots, x_{N-1})$$

where $\mathbf{x}$ is a vector in $\mathbb{R}^N$ with N components $x_n = x(nT/N)$.

The sampled signal $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_{N-1})$ can be extended periodically with period N by modular definition

$$x_m = x_{(m \bmod N)}$$

for all $m \in \mathbb{Z}$.

Discrete signal and basis vectors

In order to form the discrete basis vectors we start with exponential basis

$$\phi_k(t) = \mathrm{e}^{\mathrm{j}\omega_k t} = \mathrm{e}^{\mathrm{j}2\pi kt/T}$$

and substitute $t \to nT/N$ yielding N components of the basis vector in $\mathbb{C}^N$

$$\phi_{k,n} \equiv \phi_k\left(\frac{nT}{N}\right) = \mathrm{e}^{\mathrm{j}2\pi kn/N}.$$

The k-th basis vector has the following complex components:

$$\phi_k = \begin{bmatrix} e^{j2\pi k \cdot 0/N} \\ e^{j2\pi k \cdot 1/N} \\ e^{j2\pi k \cdot 2/N} \\ \vdots \\ e^{j2\pi k \cdot (N-1)/N} \end{bmatrix}$$

DFT basis — Scaling factor $||\phi_k||^2$

On $\mathbb{C}^n$ the usual inner product is defined as

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^{\mathsf{T}} \overline{\mathbf{y}} = x_1 \overline{y}_1 + x_2 \overline{y}_2 + \ldots + x_n \overline{y}_n$$

The corresponding norm is

$$||\mathbf{x}||^2 = \langle \mathbf{x}, \mathbf{x} \rangle = x_1 \overline{x}_1 + x_2 \overline{x}_2 + \cdots + x_n \overline{x}_n = |x_1|^2 + |x_1|^2 + \cdots + |x_n|^2$$

which translates for our basis vector to

$$||\boldsymbol{\phi}_k||^2 = \phi_{k,0} \overline{\phi_{k,0}} + \phi_{k,1} \overline{\phi_{k,1}} + \cdots + \phi_{k,N-1} \overline{\phi_{k,N-1}} = 1 + 1 + \ldots + 1$$

as $\overline{\phi_{k,n}} = \mathrm{e}^{-\mathrm{j}2\pi kn/N}$ is a complex conjugate to $\phi_{k,n} = \mathrm{e}^{\mathrm{j}2\pi kn/N}$ and therefore $\phi_{k,n} \overline{\phi_{k,n}} = 1$, which results in the scaling factor being

$$||\boldsymbol{\phi}_k||^2 = N$$

.

We can prove that basis vectors $\boldsymbol{\phi}_k$ are orthogonal by verifying that $\langle \boldsymbol{\phi}_\ell, \boldsymbol{\phi}_m \rangle = 0$ for all $\ell \neq m$:

$$\langle \boldsymbol{\phi}_\ell, \boldsymbol{\phi}_m \rangle = \sum_{\nu=0}^{N-1} \phi_{\ell,\nu} \overline{\phi_{m,\nu}} = \sum_{\nu=0}^{N-1} \mathrm{e}^{\mathrm{j}2\pi(\ell-m)\nu/N} =$$
$$= \sum_{\nu=0}^{N-1} \left(\mathrm{e}^{\mathrm{j}2\pi(\ell-m)/N} \right)^{\nu}.$$

We have arrived at partial sum of the first N elements for geometric series.

For $\ell \neq m$ we have

$$\langle \boldsymbol{\phi}_\ell, \boldsymbol{\phi}_m \rangle = \frac{1 - \left(\mathrm{e}^{\mathrm{j}2\pi \frac{\ell-m}{N}} \right)^N}{1 - \mathrm{e}^{\mathrm{j}2\pi(\ell-m)/N}} = \frac{1 - \mathrm{e}^{\mathrm{j}2\pi(\ell-m)}}{1 - \mathrm{e}^{\mathrm{j}2\pi(\ell-m)/N}} = \frac{1-1}{1 - \mathrm{e}^{\mathrm{j}2\pi(\ell-m)/N}} = 0$$

Lectue Contents

Strang (2000):

> *The Fourier series is linear algebra in infinite dimensions. The "vectors" are functions $f(t)$; they are projected onto the sines and cosines; that produces the Fourier coefficients a_k and b_k. From this infinite sequence of sines and cosines, multiplied by a_k and b_k, we can reconstruct $f(t)$. That is the classical case, which Fourier dreamt about, but in actual calculations it is the discrete Fourier transform that we compute. Fourier still lives, but in finite dimensions.*

Definition of the DFT and IDFT

1. Let $\mathbf{x} \in \mathbb{C}^N$ be a vector $(x_0, x_1, x_2, \ldots, x_{N-1})$. The discrete Fourier transform (DFT) of $\mathbf{x}$ is the vector $\mathbf{X} \in \mathbb{C}^N$ with components

$$X_k = \langle \mathbf{x}, \Phi_k \rangle = \sum_{m=0}^{N-1} x_m \, \mathrm{e}^{-\mathrm{j}2\pi km/N}.$$

2. Let $\mathbf{X} \in \mathbb{C}^N$ be a vector $(X_0, X_1, X_2, \ldots, X_{N-1})$. The inverse discrete Fourier transform (IDFT) of $\mathbf{X}$ is the vector $\mathbf{x} \in \mathbb{C}^N$ with components

$$x_k = \frac{\langle \mathbf{X}, \mathbf{\Phi}_{-k} \rangle}{\langle \Phi_k, \Phi_k \rangle} = \frac{1}{N} \sum_{m=0}^{N-1} X_m \, \mathrm{e}^{\mathrm{j}2\pi km/N}.$$

DC coefficient X_0

The coefficient X_0/N measures the contribution of the basic waveform $(1, 1, 1, \ldots, 1)$ to $\mathbf{x}$. In fact

$$\frac{X_0}{N} = \frac{1}{N} \sum_{m=0}^{N-1} x_m$$

is the average value of $\mathbf{x}$.This coefficient is usually called as the DC coefficient, because it measures the strength of the direct current component of a signal.

The Fourier Transform can be defined for signals that are

- Discrete or continuous in time
- Finite or infinite duration
- Provided we denote the variable in time domain as $x(t)$, or $x(n)$, the transformed variables in frequency domain are correspondingly $X(j\omega)$ or $X(k)$.

This unification results in four cases.

An overview of Fourier transforms

	continuous in time	discrete in time periodic in frequency
continuous in frequency	$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(j\omega) e^{j\omega t} d\omega$	$x(n) = \frac{T}{2\pi} \int_{-\pi/T}^{+\pi/T} X(e^{j\omega T}) e^{jk\omega T} d\omega$
	$X(j\omega) = \int_{-\infty}^{\infty} e^{-j\omega t} x(t) dt$	$X(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} x(n) e^{-jk\omega T}$
	Fourier transform	Fourier transform $t = nT$ (DTFT)
discrete in frequency periodic in time	$x(t) = \sum_{k=-\infty}^{\infty} X(k) e^{jk\omega_0 t}$	$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{(j2\pi/N)kn}$
	$X(k) = \frac{\omega_0}{2\pi} \int_{-\pi/\omega_0}^{\pi/\omega_0} x(t) e^{-jn\omega_0 t} dt$	$X(k) = \sum_{n=0}^{N-1} x(n) e^{-(j2\pi/N)kn}$
	Fourier series	Discrete Fourier transform (DFT)

The DFT consists of inner products of the input sequence $x[n]$ with sampled complex sinusoidal sections

$$w_N^{kn} = \mathrm{e}^{\mathrm{j}2\pi nk/N}$$

yielding

$$X_k = \langle \mathbf{x}, \mathbf{w}_k \rangle = \mathbf{x}^{\mathsf{T}} \overline{\mathbf{w}_k} = \sum_{m=0}^{N-1} x_m \, \mathrm{e}^{-\mathrm{j}2\pi km/N}, \quad k = 0, 1, 2, \ldots, N-1.$$

By collecting the DFT output samples into a column vector, we have

$$\underbrace{\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{N-1} \end{bmatrix}}_{\mathbf{X}} = \underbrace{\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \overline{w_N^1} & \overline{w_N^2} & \cdots & \overline{w_N^{N-1}} \\ 1 & \overline{w_N^2} & \overline{w_N^4} & \cdots & \overline{w_N^{2(N-1)}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \overline{w_N^{N-1}} & \overline{w_N^{2(N-1)}} & \cdots & \overline{w_N^{(N-1)(N-1)}} \end{bmatrix}}_{\mathbf{W}_N^*} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix}}_{\mathbf{x}}$$

Finally we can write matrix representation as

$$\mathbf{X} = \mathbf{W}_N^* \mathbf{x}.$$

The matrix $\mathbf{W}_N^* = \overline{(\mathbf{W}_N)^\mathsf{T}}$ denotes the Hermitian transpose of the complex matrix $\mathbf{W}_N$. It can be shown that

$$\mathbf{W}_N^* \times \mathbf{W}_N = \begin{bmatrix} N & 0 & 0 & \cdots & 0 \\ 0 & N & 0 & \cdots & 0 \\ 0 & 0 & N & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & N \end{bmatrix} = N \cdot \mathbf{1}$$

and consequently the inversion of the Eq. (29) is

$$\mathbf{x} = \frac{1}{N}\mathbf{W}_N\mathbf{X}.$$

Some practical comments

If the number of digital samples in each time slice is a power of 2, one can use a faster version of the DFT known as the fast Fourier transform (FFT)

The FFT assumes that the samples being analyzed comprise one cycle of a periodic wave. In most cases it is not the case and analysis will contain many spurious frequencies not actually present in the signal.

Sample fast enough and long enough!

To recognize details in frequency domain use spectral interpolation.

What is aliasing?

It is easiest to describe in terms of a visual sampling:

We all know and love movies. If you have ever watched a western and seen the wheel of a rolling wagon appear to be going backwards, you have witnessed aliasing. The movie's frame rate is not adequate to describe the rotational frequency of the wheel, and our eyes are deceived by the misinformation.

The Nyquist Theorem tells us that we can successfully sample and play back frequency components up to one-half the sampling frequency.

Aliasing is the term used to describe what happens when we try to record and play back frequencies higher than one-half the sampling rate.

What is aliasing?

Consider a digital audio system with a sample rate of 48 KHz, recording a steadily rising sine wave tone. At lower frequency, the tone is sampled with many points per cycle. As the tone rises in frequency, the cycles get shorter and fewer and fewer points are available to describe it. At a frequency of 24 KHz, only two sample points are available per cycle, and we are at the limit of what Nyquist says we can do.

Still, those two frequency points are adequate, in a theoretical world, to recreate the tone after conversion back to analog and low-pass filtering.

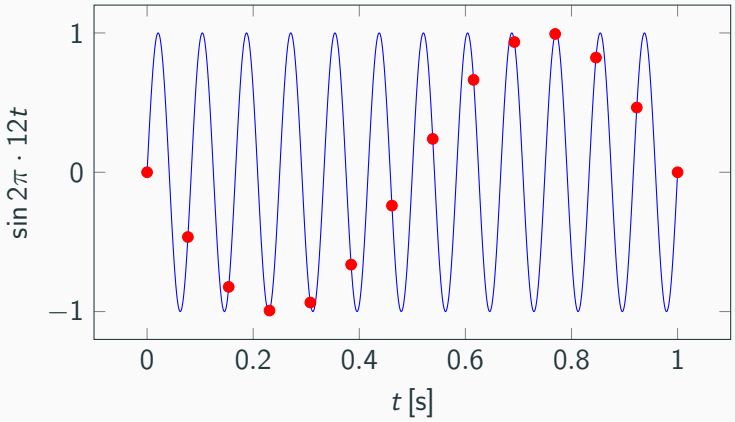
But, if the tone continues to rise, the number of samples per cycle is not adequate to describe the waveform, and the inadequate description is equivalent to one describing a lower frequency tone – this is aliasing.

In fact, the tone seems to reflect around the 24 KHz point:

- A 25 KHz tone becomes indistinguishable from a 23 KHz tone.
- A 30 KHz tone becomes an 18 KHz tone.

Aliasing due to insufficient sampling

The following figure illustrates what happens if a signal is sampled at regular time intervals that are slightly below the period of the original signal.



Zero padding

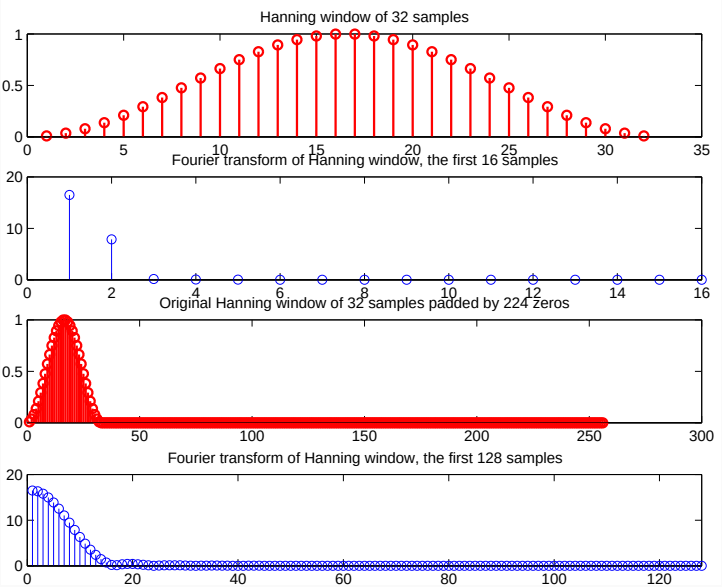
Zero padding consists of appending zeros to a signal. It maps a length N signal to a length $M > N$ signal. M does not need to be an integer multiple of N.

Zero padding in the time domain gives spectral interpolation in the frequency domain.

Similarly, zero padding in the frequency domain gives bandlimited interpolation in the time domain. This is how ideal sampling rate conversion is accomplished.

Usually we use FFT which requires signals of length $M = 2^m$ which means we chose the number of zeros equal to $2^m - N$.

Zero padding: How does it work?



Lectue Contents

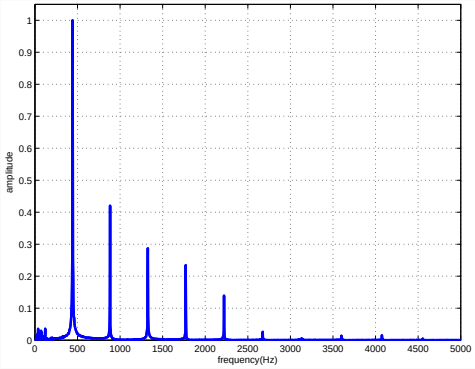
Definition (Nyquist-Shannon Sampling Theorem, 1927)

It is possible precisely to reconstruct a continuous-time signal from its samples, given that

1. the signal is bandlimited;
2. the sampling frequency f_s is greater than twice the signal bandwidth.

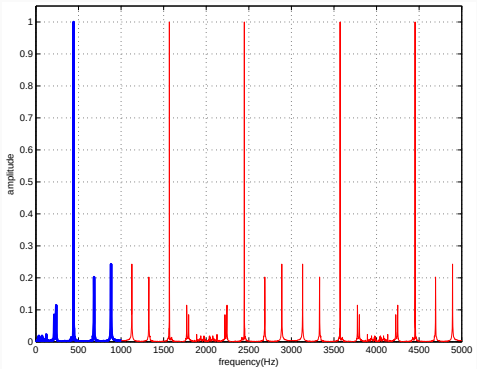
Aliasing in Audio

- The initial sound is a numerically synthesized piano-tone at 440 Hz. The sampling frequency is of 44.1 kHz (CD-quality).
- The harmonic frequencies at multiple of the fundamental tone (440 Hz) are clearly visible.



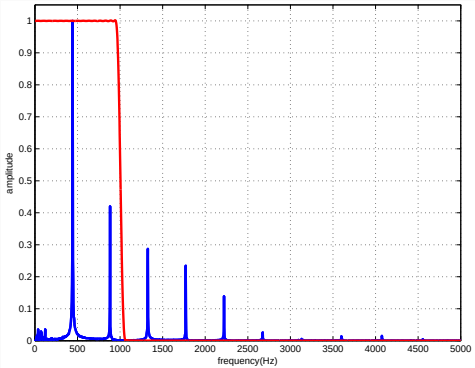
Aliasing in Audio

- The sound will be resampled at 2 kHz, without precautions against aliasing. The tone sounds rather strange.
- The aliasing is visible on the graphs as a "warping" of the frequencies against a "mirror" at the Nyquist frequency 1 kHz.



Aliasing in Audio

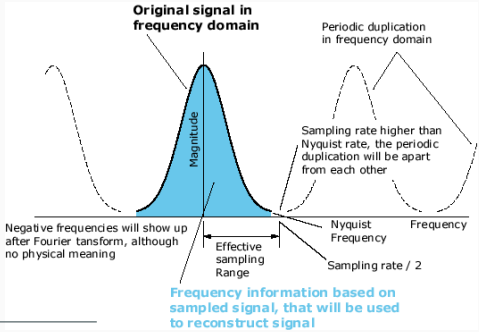
- In order to avoid aliasing, the spectrum of the signal should be zero at frequencies higher than the Nyquist frequency before resampling. A low-pass filter is used to achieve this



Aliasing and DFT

. . . for a digital signal processing with DFT there are limits:

- The signal must be band-limited. This means there is a frequency above which the signal is zero.
- Hence the maximum useable frequency in the DFT is $f_s/2$ - the Nyquist[1] frequency!



[1] Harry Nyquist 1889-1976

Lectue Contents

Nonlocality of DFT

DFT assumes signal is stationary. It cannot detect local frequency changes.

Example (Frequency hop)

Consider two different periodic signals $f(t)$ and $g(t)$ defined on $0 \leq t < 1$ with frequencies $f_1 = 96\,\text{Hz}$ and $f_2 = 235\,\text{Hz}$ as follows:

- $f(t) = 0.5\,\sin(2\pi f_1 t) + 0.5\,\sin(2\pi f_2 t)$
- $g(t) = \begin{cases} \sin(2\pi f_1 t) & \text{for } 0 \leq t < 0.5, \\ \sin(2\pi f_2 t) & \text{for } 0.5 \leq t < 1.0. \end{cases}$

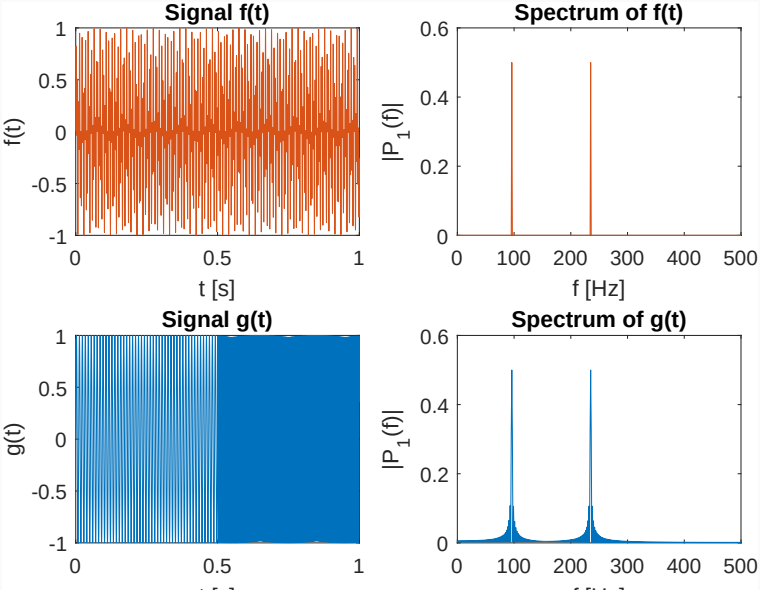
Use the sampling frequency $f_s = 1000\,\text{Hz}$ to produce sample vectors $\mathbf{f}$ and $\mathbf{g}$. Compute the DFT of each sampled signal.

Nonlocality of DFT

Two different signals $f(t)$ and $g(t)$ are constructed with Matlab commands

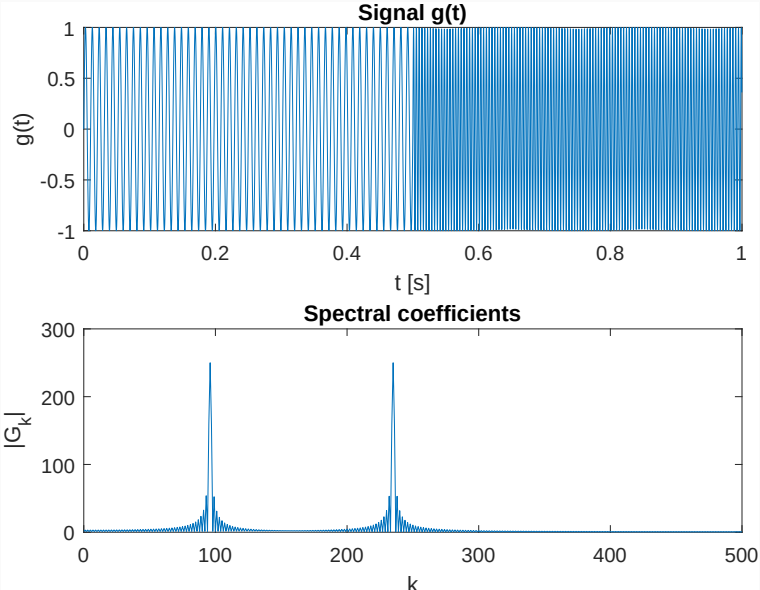
```
Fs = 1000; % sampling frequency
f1 = 96;
f2 = 235;
t1 = (0:499)/Fs; % time samples for 'g1'
t2 = (500:999)/Fs; % time samples for 'g2'
t = [t1 t2]; % time samples for 'f'
f = 0.5*sin(2*pi*f1*t)+0.5*sin(2*pi*f2*t);
g1 = [sin(2*pi*f1*t1) zeros(1,500)];
g2 = [zeros(1,500) sin(2*pi*f2*t2)];
g = g1+g2;
```

Magnitude of DFT for $f(t)$ and $g(t)$



- It is obvious that each signal contains dominant frequencies close to 96 Hz and 235 Hz and the magnitudes are fairly similar.
- But: The signals $f(t)$ and $g(t)$ are quite different in the time domain!
- The example illustrates one of the shortcomings of traditional Fourier transform: nonlocality or global nature of the basis vectors $\mathbf{w}_N$ or its constituting analog waveforms $e^{j2\pi kt/T}$.

Detail of signal $g(t)$



Summary:

- Discontinuities are particularly troublesome.
- The signal $g(t)$ consists of two sinusoids only, but the excitation of several G_ks in frequency domain around the dominant frequencies gives the impression that the entire signal is more oscillatory.
- We would like to have possibility to localize the frequency analysis to smaller portions for the signal.

These requirements led to development of windowed version of Fourier transform — the **short time Fourier transform, STFT**.

Consider a sampled signal $\mathbf{x} \in \mathbb{C}^N$, indexed from 0 to $N-1$. We wish to analyse the frequencies present in $\mathbf{x}$, but only within a certain time range. We choose integers $m \geq 0$ and M such that $m + M \leq N$ and define a vector $\mathbf{w} \in \mathbb{C}^N$ as

$$w_k = \begin{cases} 1 & \text{for } m \leq k \leq m + M - 1 \\ 0 & \text{otherwise} \end{cases}$$

We use $\mathbf{w}$ to define a new vector $\mathbf{y}$ with components

$$y_k = w_k x_k \quad \text{for } 0 \leq k \leq N - 1.$$

We use notation $\mathbf{y} = \mathbf{wx}$ and refer to the vector $\mathbf{w}$ as the (rectangular) window.

Proposition

Let $\mathbf{x}$ *and* $\mathbf{w}$ *be vectors in* $\mathbb{C}^N$ *with discrete Fourier transforms* $\mathbf{X}$ *and* $\mathbf{W}$*, respectively. Let* $\mathbf{y} = \mathbf{wx}$ *have DFT* $\mathbf{Y}$*. Then*

$$\mathbf{Y} = \frac{1}{N}\mathbf{X} * \mathbf{W},$$

where $*$ *is circular convolution in* $\mathbb{C}^N$*.*

Definition (Circular convolution)

The n-th element of an N-point circular convolution of N-periodic vectors $\mathbf{X}$ and $\mathbf{W}$ is

$$Y_n = \frac{1}{N}\sum_{m=0}^{N-1} X_m W_{(n-m) \bmod N}.$$

When processing a non-stationary signal we assume that the signal is short-time stationary and we perform a Fourier transform on these small blocks — we multiple the signal by a window function that is zero outside the defined "short-time" range.

Definition (Rectangular window)

The rectangular window is defined as:

$$w_n = \begin{cases} 1 & \text{for } 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}$$

The Matlab command `rectwin(N)` produces the N-point rectangular window.

Definition (Hamming window)

The most common windowing function in speech analysis is the Hamming window:

$$w_n = \begin{cases} 0.54 + 0.46 \cos\left(\frac{2\pi n}{N-1}\right) & \text{for } 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}$$

Matlab command `hamming(N)` produces the N-point Hamming window.

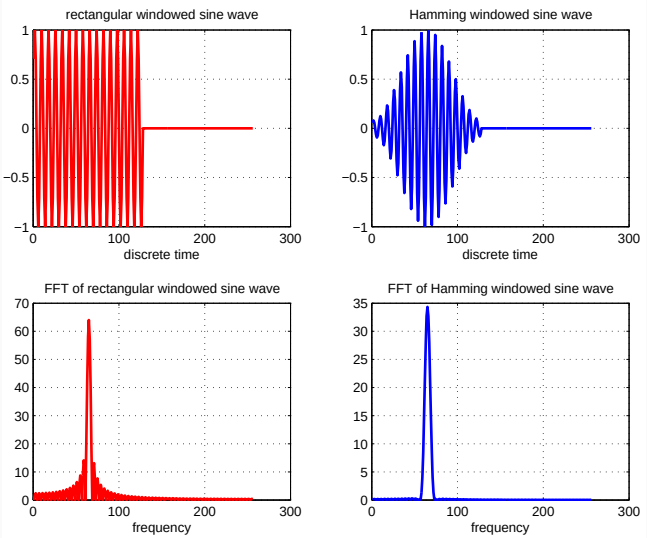
Definition (Blackman window)

Another common type of window is the Blackman window:

$$w_n = \begin{cases} 0.42 + 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.08\cos\left(\frac{4\pi n}{N-1}\right) & \text{for } 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}$$

Use `blackman(N)` to produce the N-point Blackman window.

Windowing result



Lectue Contents

Consider the analog signal

$$x(t) = 2.0\cos(2\pi\, 5t) + 0.8\sin(2\pi\, 12t) + 0.3\cos(2\pi\, 47t)$$

on the interval $t \in (0, 1)$. Sample this signal with period $T = 1/128\,\mathrm{s}$ and obtain sample vector $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_{127})$.

a) Make MATLAB m-file which plots signals $x(t)$ and $\mathbf{x}$

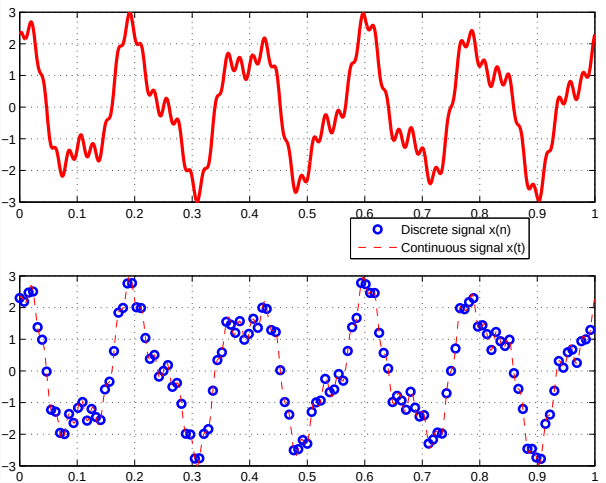
b) Using definition of the DFT find $\mathbf{X}$.

c) Use MATLAB command `fft(x)` to compute DFT of $\mathbf{X}$.

d) Make MATLAB m-file which computes DFT of $\mathbf{x}$ and plots signal and its spectrum.

e) Compute IDFT of the $\mathbf{X}$ and compare it with the original signal $x(t)$.

Matlab Session 3.1 — Input signal plots



Matlab Session 3.1 — Solution (1/2)

```
clear
% plots original and sampled signal
t = linspace(0,1,1001);
x = 2.0*cos(2*pi*5*t) + 0.8*sin(2*pi*12*t) + 0.3*cos(2*pi*47*t);
N = 128; % number of samples
tdelta = 1/N; % sampling period
ts(1) = 0;
xs(1) = x(1);
for k = 2:N
  ts(k) = (k-1)*tdelta;
  xs(k) = 2.0*cos(2*pi*5*(k-1)*tdelta) + ...
          0.8*sin(2*pi*12*(k-1)*tdelta) + ...
          0.3*cos(2*pi*47*(k-1)*tdelta);
end
```

Matlab Session 3.1 — Solution (2/2)


```
figure(1);
subplot(2,1,1);
plot(t,x,'LineWidth',2.5,'Color',[1 0 0]);
grid on;
subplot(2,1,2);
plot(ts, xs,'o','LineWidth',2.0,'Color',[0 0 1]);
hold on;
plot(t,x,'--','Color',[1 0 0]);
grid on;
legend('Discrete signal x(n)','Continuous signal x(t)');
hold off;
pause
```

Matlab Session 3.2 — Windowing (1/3)

Consider signal $f(t) = \sin(2\pi f_1 t) + 0.4\,\sin(2\pi f_2 t)$ defined on $0 \leq t \leq 1$ with frequencies $f_1 = 137\,\text{Hz}$ and $f_2 = 147\,\text{Hz}$:

a) Use Matlab to sample $f(t)$ at $N = 1000$ points $t_k = \{k/f_s\}_{k=0}^{N}$ with sampling frequency $f_s = 1000\,\text{Hz}$

```
N = 1000; % number of samples
Fs = 1000; % sampling frequency
f1 = 137; % 1. frequency
f2 = 147; % 2. frequency
tk = (0:(N-1))/Fs; % sampling times
f = sin(2*pi*f1*tk) + 0.4*sin(2*pi*f2*tk); % sampled signal
```

b) Compute the DFT of the signal with `F=fft(f)` resp. `F=fft(f,N)`.
Consult the Matlab documentation and explain the difference!

c) Display the magnitude of the Fourier transform with `plot(abs(F(0:501))`

d) Construct a rectangular windowed version of $f(n)$ for window length 200 with

```
fwa = f;
fwa(201:1000) = 0.0;
```

e) Compute the DFT of `fwa` and display the magnitude of the first 501 components.

f) Can you distinguish the two constituent frequencies?
Be careful: is it really obvious that the second frequency is not a side lobe leakage?

g) Construct a windowed version of $f(n)$ of length 200 with

```
fwb = f(1:200);
```

h) Compute the DFT and display the magnitude of the first 101 components.

i) Can you distinguish the two constituent frequencies? Compare the plot of `fwb` with the DFT of `fwa`.

j) Repeat the parts d–h using other window lengths such as 300, 100 or 50. How short can the time window be and still allow resolution of the two constituent frequencies?

k) Does it matter whether we treat the windowed signal as a vector of length 1000 as in part 4 or shorter vector as in part 7? Does the side lobe energy confuse the results?

Using reasonable resolution in frequency domain with zero padding in the time domain, determine the frequency of the periodic signal defined as

$$x_s = \sin(32.044245t) + \sin(37.070793t).$$

The discrete signal has only 32 samples x_n produced by sampling frequency $f_0 = 1/32$.

Matlab Session 3.3 — Zero padding (2/3)

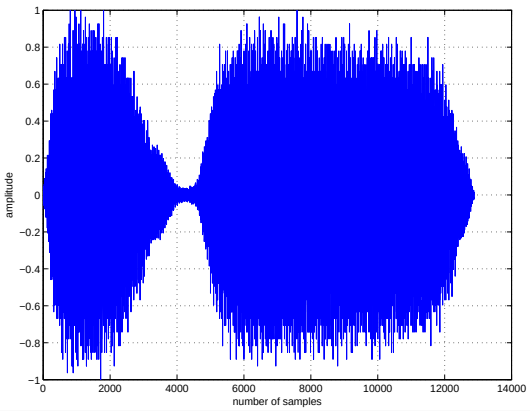

```
clear
t = linspace(0,1,1001);
xs = sin(32.044245*t)+sin(37.070793*t);
N = 32;
f0 = 1/N;
k = 0:1:N-1;
x1 = sin(32.044245*f0*k) + sin(37.070793*f0*k);
figure(1)
subplot(3,1,1)
plot(t,xs,'LineWidth',1.5,'Color',[1 0 0]);
```

Matlab Session 3.3 — Zero padding (3/3)

```
% Second row, make the lenghth 64 samples
subplot(3,1,2)
xfa_64 = abs(fft([x1 zeros(1,32)]));
plot(xfa_64); hold on;
stem(xfa_64); hold off;
% Third row, make the length 512 samples
xfa_512 = abs(fft([x1 zeros(1,32)]));
plot(xfa_512); hold on;
stem(xfa_512); hold off;
```

Matlab Session 3.4 — Steam whistle (1/3)

a) Start MATLAB. Load in the "train" signal with command `load('train')`. Note that the audio signal is loaded into a variable `y` and the sampling rate into `Fs`.



b) The sampling rate is 8192 Hz, and the signal contains 12 880 samples. If we consider this signal as sampled on an interval $(0, T)$, then $T = 12880/8192 \approx 1.5723$ seconds.

c) Compute the DFT of the signal with `Y=fft(y)`. Display the magnitude of the Fourier transform with `plot(abs(Y))`
The DFT is of length 12 880 and symmetric about center.

d) Since MATLAB indexes from 1, the DFT coefficient Y_k is actually `Y(k+1)` in MATLAB !

e) You can plot only the first half of the DFT with `plot(abs(Y(1:6441))`.

f) Compute the actual value of each significant frequency in Herz. Use the data cursor on the plot window to pick out the frequency and amplitude of three largest components.

g) Denote these frequencies f_1, f_2, f_3, and let A_1, A_2, A_3 denote the corresponding amplitudes. Define these variables in MATLAB.

h) Synthetize a new signal using only these frequencies, sampled at 8192 Herz on the interval $(0, 1.5)$ with

```
t=[0:1/8192:1.5];
ys=(A1*sin(2*pi*f1*t)+ ...
A2*sin(2*pi*f2*t)+A3*sin(2*pi*f3*t))/(A1+A2+A3);
```

i) Play the original train sound with `sound(y)` and the synthesized version `sound(ys)`. Compare the quality!

j) Can you explore another frequency components? If it is so, follow the steps g) – i) and listen to the result.

We can study a simple approach to compressing an audio signal:

The idea is to transform the audio signal in the frequency domain with DFT and then to eliminate the insignificant frequencies by thresholding, i.e. by zeroing out any Fourier coefficients below a given threshold. This becomes a compressed version of the signal. To recover an approximation to the signal, we use inverse DFT to take the limited spectrum back to the time domain.

k) Thresholding: Compute the maximum value of Y_k with `m=max(abs(Y))`. Choose a thresholding parameter $\in (0, 1)$, for example, `thresh=0.1`

l) Zero out all frequencies in `Y` that fall below a value `thresh*m`. This can be done with logical indexing or e.g. with

```
Ythresh=(abs(Y)>m*thresh).*Y;
```

Plot the thresholded transform with `plot(abs(Ythresh))`.

m) Compute the compression ratio as the fraction of DFT coefficients which survived the cut, `sum(abs(Ythresh)>0)/12880.`

n) Recover the original time domain with inverse transform `yt=real(ifft(Ytresh))` and play the compressed audio with `sound(yt)`. The `real` command truncates imaginary round-off error in the `ifft` procedure.

o) Compute the distortion (as a percentage) of the compressed signal using formula

$$\epsilon = \frac{\|\mathbf{y} - \mathbf{y}_t\|^2}{\|\mathbf{y}\|^2}$$

Note: The `norm(y)` command in MATLAB computes $\|\mathbf{y}\|$, the standard Euclidean norm of the vector $\mathbf{y}$.

p) Repeat the computation for threshold values `thresh=0.5`, `thresh=0.05` and `thresh=0.005`. In each case compute the compression ratio, the distortion, and play the audio signal and rate its quality.

Lectue Contents

Assignment — Analysis of audio signal

- Start MATLAB. Load in the "ding" audio signal with command `y=wavread('ding.wav');` The audio signal is stereo one and can be decoupled into two channels by `y1=y(:,1); y2=y(:,2);`. The sampling rate is 22 050 Herz, and the signal contains 20 191 samples. If we consider this signal as sampled on an interval $(0, T)$, then $T = 20191/22050 \approx 0.9157$ seconds.
- Compute the DFT of the signal with `Y1=fft(y1);` and `Y2=fft(y2);`. Display the magnitude of the Fourier transform with `plot(abs(Y1))` or `plot(abs(Y2))`. The DFT is of length 20 191 and symmetric about center.

- Since MATLAB indexes from 1, the DFT coefficient Y_k is actually `Y(k+1)` in MATLAB ! Also Y_k corresponds to frequency $k/T = k/0.9157$ and so `Y(k+1)` corresponds to $f_k = (k-1)/T = (k-1)/0.9157$.
- You can plot only the first half of the DFT with `plot(abs(Y1(1:6441)))` or `plot(abs(Y2(1:6441)))`. Use the data cursor button on plot window to pick out the frequency and amplitude of the two (obviously) largest components in the spectrum. Compute the actual value of each significant frequency in Herz.

Assignment — Analysis of audio signal

- Let f_1, f_2 denote these frequencies in Herz, and let A_1, A_2 denote the corresponding amplitudes from the plot. Define these variables in MATLAB.
- Generate a new signal using only these frequencies, sampled at 22 050 Herz on the interval $(0, 1)$ with

```
t = [0:1/22050:1];
y12 = (A1*sin(2*pi*f1*t) + A2*sin(2*pi*f2*t))/(A1+A2)
```

- Play the original sound with `sound(y1)` and the synthesized version `sound(y12)`. Repeat the experiment with sound of the second channel `sound(y2)`. Note that our synthesis does not take into account the phase information at these frequencies.
- Does the artificial generated signal reproduce `ding.wav` correctly? Compare the quality!

Assignment — Analysis of audio signal

1. Repeat the parts a)–k) from the lecture project, but this time using a triangular window.
2. A triangular window vector **w** of length $L = 201$ can be constructed using

```
L = 201;
w = triang(L);
```

3. Construct a windowed signal of the length 1000 as

```
fwc = zeros(size(f));
fwc(1:L)=f(1:L).*w;
```

and compute its spectrum using `fft(fwc)`.

4. Try varying the window length L. What is the shortest window that allows you to distinguish the two frequencies?
5. Repeat the previous parts 1–10 for the Hamming window.
6. Submit the answers for the several questions raised in parts 1–16 as a written Report on Window Functions by November 20, 2019.

Homework rules

Submit your results by Wednesday, October 21 2020 using the web page `http://zolotarev.fd.cvut.cz/mni`

Solution report should be formally correct (structuring, grammar).

Only `.pdf` files are acceptable. Handwritten solutions and `.doc` and `.docx` files will not be accepted.

Solutions written in TEX (using LyX, Overleaf, whatever) may receive small bonification.