Wavelets and Applications

Jhony H. Giraldo (Enseignant-Chercheur).
jhony.giraldo@telecom-paris.fr
October 20, 2023

Objectives of Today's Lecture

- Wavelets are termed as a "brief oscillation". The concepts of wavelets are pretty important in signal processing

Objectives of Today's Lecture

- Wavelets are termed as a "brief oscillation". The concepts of wavelets are pretty important in signal processing
- The two objectives of this lecture (plus the TP) are:

Fundamentals

Understand the basic concepts of wavelets in signal processing

Applications

Develop the ability to use wavelets in some applications

Objectives of Today's Lecture

- Wavelets are termed as a "brief oscillation". The concepts of wavelets are pretty important in signal processing
- The two objectives of this lecture (plus the TP) are:

Fundamentals

Understand the basic concepts of wavelets in signal processing

Applications

Develop the ability to use wavelets in some applications

- Identify scenarios where wavelets might be useful

A Brief History about Wavelets

- The history of wavelets goes back more than 100 years ago with the Haar wavelet proposed by Alfréd Haar

A Brief History about Wavelets

- The history of wavelets goes back more than 100 years ago with the Haar wavelet proposed by Alfréd Haar
- The term "wavelet" come until much later when the mathematics were formalized

A Brief History about Wavelets

- The history of wavelets goes back more than 100 years ago with the Haar wavelet proposed by Alfréd Haar
- The term “wavelet” come until much later when the mathematics were formalized
- People used wavelets in several fields of science like physics, geology, and other computer science disciplines without knowing a great deal about their mathematical properties

A Brief History about Wavelets

- The history of wavelets goes back more than 100 years ago with the Haar wavelet proposed by Alfréd Haar
- The term "wavelet" come until much later when the mathematics were formalized
- People used wavelets in several fields of science like physics, geology, and other computer science disciplines without knowing a great deal about their mathematical properties
- Wavelets have profound roots in harmonic analysis and applied mathematics, and have several applications and important consequences in signal analysis

A Brief History about Wavelets

- Some 20 to 30 years ago there was a huge hype about wavelets in the scientific community, everyone was using wavelets for everything (this is what our grandparents say, I was a baby back then)

A Brief History about Wavelets

- Some 20 to 30 years ago there was a huge hype about wavelets in the scientific community, everyone was using wavelets for everything (this is what our grandparents say, I was a baby back then)
- Does this sound familiar to you? Large Language Models now and Convolutional Neural Networks yesterday.

A Brief History about Wavelets

- Some 20 to 30 years ago there was a huge hype about wavelets in the scientific community, everyone was using wavelets for everything (this is what our grandparents say, I was a baby back then)
- Does this sound familiar to you? Large Language Models now and Convolutional Neural Networks yesterday.
- As with all hypes in computer science, the interest in wavelets has considerably decreased, but we keep using them for key (and important) applications

A Brief History about Wavelets

- Some 20 to 30 years ago there was a huge hype about wavelets in the scientific community, everyone was using wavelets for everything (this is what our grandparents say, I was a baby back then)
- Does this sound familiar to you? Large Language Models now and Convolutional Neural Networks yesterday.
- As with all hypes in computer science, the interest in wavelets has considerably decreased, but we keep using them for key (and important) applications
- Today, there are a couple of recent publications about wavelets in top conferences with exotic applications

-Hao Phung *et al.* "Wavelet Diffusion Models are Fast and Scalable Image Generators". In IEEE/CVF CVPR 2023.
-Daniel Rho *et al.* "Masked Wavelet Representation for Compact Neural Radiance Fields". In IEEE/CVF CVPR 2023.
-Sourav Pal *et al.* "Controlled Differential Equations on Long Sequences via Non-standard Wavelets". In ICML 2023.
-Florentin Guth *et al.* "Wavelet Score-Based Generative Modeling". In NeurIPS 2022.

A Brief History about Wavelets

- What I'm teaching today is a very intuitive introduction to wavelets

A Brief History about Wavelets

- What I'm teaching today is a very intuitive introduction to wavelets
- There are a lot of people who have devoted their entire careers to studying wavelets

A Brief History about Wavelets

- What I'm teaching today is a very intuitive introduction to wavelets
- There are a lot of people who have devoted their entire careers to studying wavelets
- Almost all scientists who developed the theory of wavelets are alive: Yves Meyer (Emeritus at ENS), Ingrid Daubechies (Prof. at Duke University), Stéphane Mallat (Prof. at ENS), and Pascal Auscher (Prof. at Université Paris-Saclay) among others.

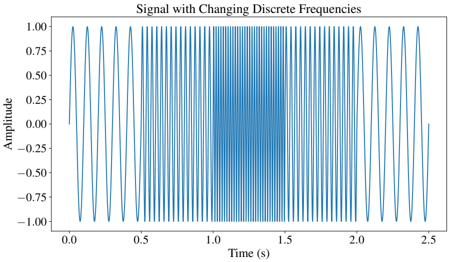
A Brief History about Wavelets

- What I'm teaching today is a very intuitive introduction to wavelets
- There are a lot of people who have devoted their entire careers to studying wavelets
- Almost all scientists who developed the theory of wavelets are alive: Yves Meyer (Emeritus at ENS), Ingrid Daubechies (Prof. at Duke University), Stéphane Mallat (Prof. at ENS), and Pascal Auscher (Prof. at Université Paris-Saclay) among others.
- There are plenty of resources if you want to study beyond what I'm presenting today

- **Stéphane Mallat "A Wavelet Tour of Signal Processing, The Sparse Way". Academic Press 2008.**
- Ingrid Daubechies "Ten Lectures on Wavelets". Society for Industrial and Applied Mathematics (SIAM) 1992.
- D. Lee Fugal "Conceptual Wavelets in Digital Signal Processing". Space & Signals Technical Publishing 2009.

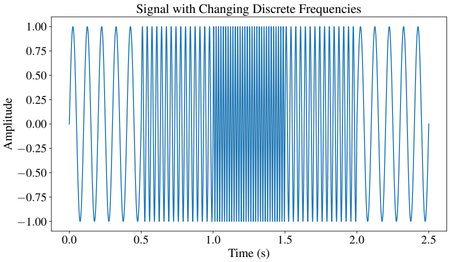
Motivation

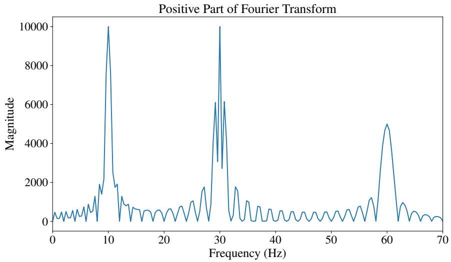
- Imagine you need to analyze the following signal



Motivation

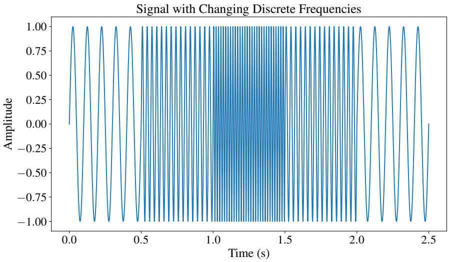
- Imagine you need to analyze the following signal
- You can compute the Fourier transform to discover what frequencies we have in this signal

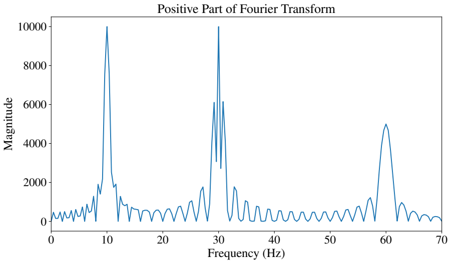




Motivation

- Imagine you need to analyze the following signal
- You can compute the Fourier transform to discover what frequencies we have in this signal

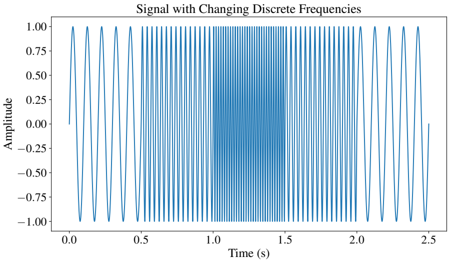


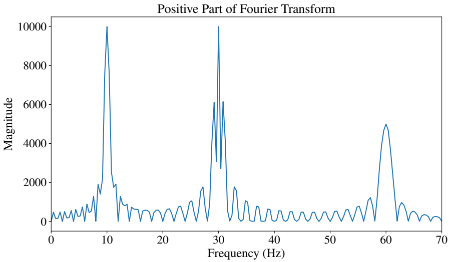


- Now you know we have components in 10, 30, and 60 Hz

Motivation

- Imagine you need to analyze the following signal
- You can compute the Fourier transform to discover what frequencies we have in this signal



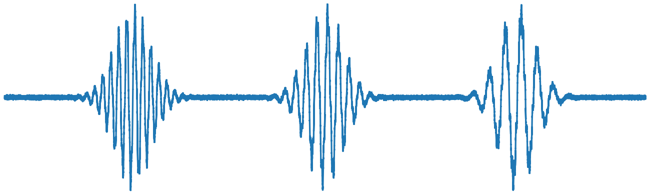


- Now you know we have components in 10, 30, and 60 Hz
- But the Fourier transform doesn't tell us when those frequencies happens... Easy-peasy compute the STFT

- What if you have more complex signals, like these we encounter in real life (This is a simulation of an electric signal in a mouse's brain):

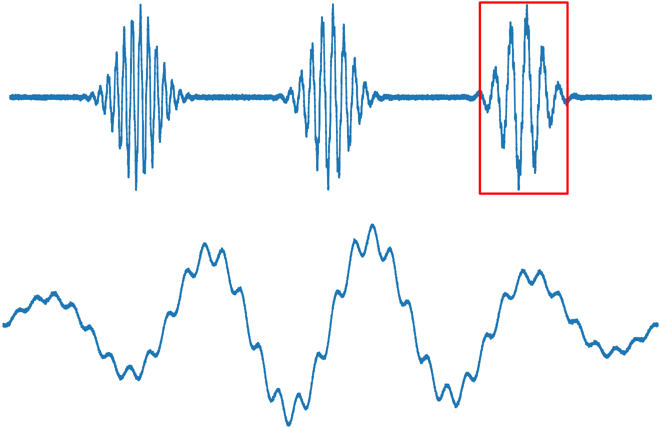
Motivation

- What if you have more complex signals, like these we encounter in real life (This is a simulation of an electric signal in a mouse's brain):



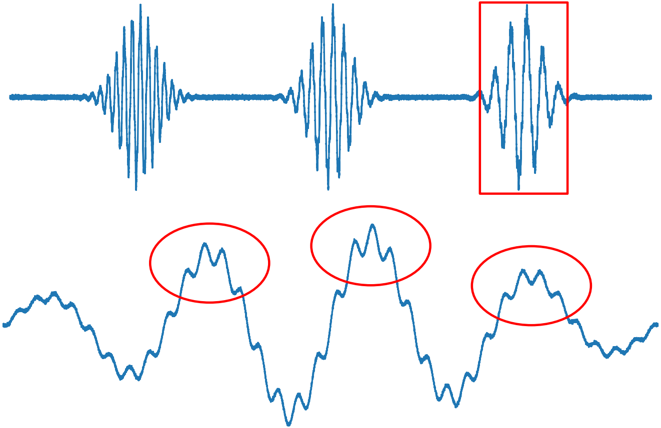
Motivation

- What if you have more complex signals, like these we encounter in real life (This is a simulation of an electric signal in a mouse's brain):



Motivation

- What if you have more complex signals, like these we encounter in real life (This is a simulation of an electric signal in a mouse's brain):



Motivation

- One of the limitations of the STFT is the inherent trade-off between time and frequency resolution. The window size used in the STFT affects the analysis resolution, where a smaller window provides better time resolution but poorer frequency resolution, and vice versa

- One of the limitations of the STFT is the inherent trade-off between time and frequency resolution. The window size used in the STFT affects the analysis resolution, where a smaller window provides better time resolution but poorer frequency resolution, and vice versa
- The STFT assumes that the frequency content of the signal does not change significantly within the window duration. This assumption may not hold for signals with rapidly changing frequencies, resulting in artifacts known as spectral leakage

Motivation

- One of the limitations of the STFT is the inherent trade-off between time and frequency resolution. The window size used in the STFT affects the analysis resolution, where a smaller window provides better time resolution but poorer frequency resolution, and vice versa
- The STFT assumes that the frequency content of the signal does not change significantly within the window duration. This assumption may not hold for signals with rapidly changing frequencies, resulting in artifacts known as spectral leakage
- Another weakness of the STFT is its inability to accurately localize frequency components in both time and frequency domains due to the uncertainty principle (more on this later)

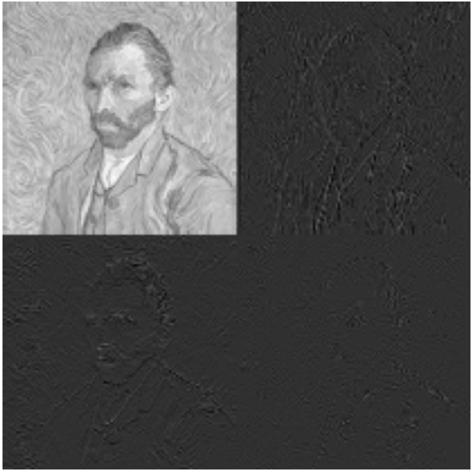
Motivation

- One of the limitations of the STFT is the inherent trade-off between time and frequency resolution. The window size used in the STFT affects the analysis resolution, where a smaller window provides better time resolution but poorer frequency resolution, and vice versa
- The STFT assumes that the frequency content of the signal does not change significantly within the window duration. This assumption may not hold for signals with rapidly changing frequencies, resulting in artifacts known as spectral leakage
- Another weakness of the STFT is its inability to accurately localize frequency components in both time and frequency domains due to the uncertainty principle (more on this later)
- The limitations of STFT can be mitigated to some extent using the wavelet transform!

Applications

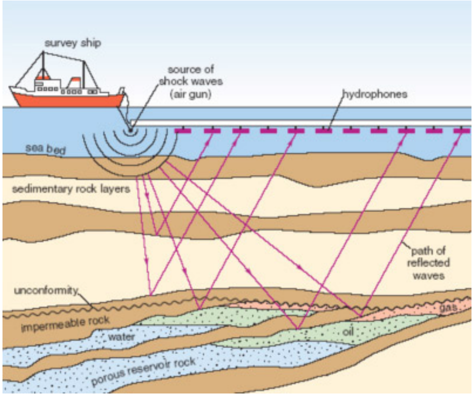
- Image compression

Wavelet coefficients



Applications

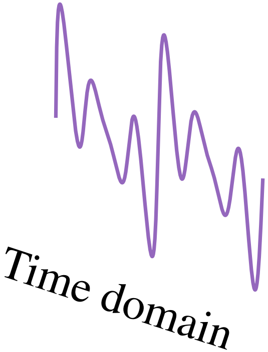
- Seismic exploration



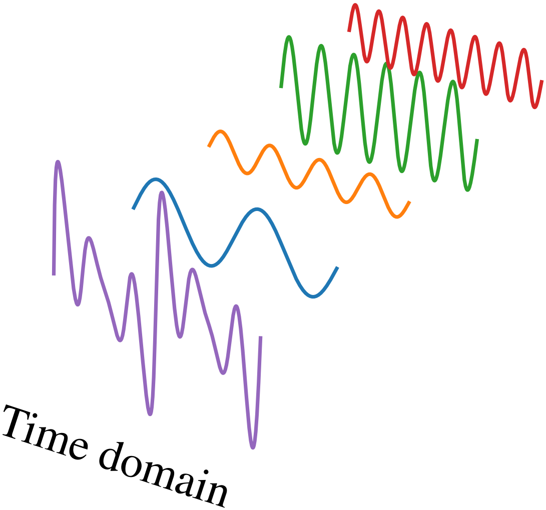
Outline

Fourier Transform (Recap)

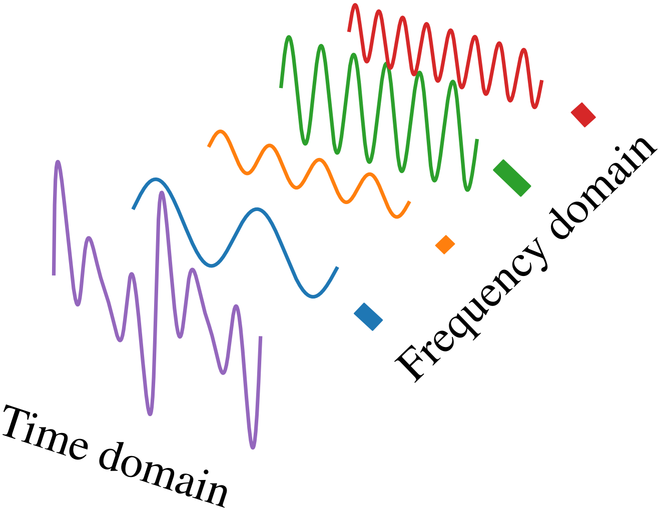
Fourier Transform (Recap)

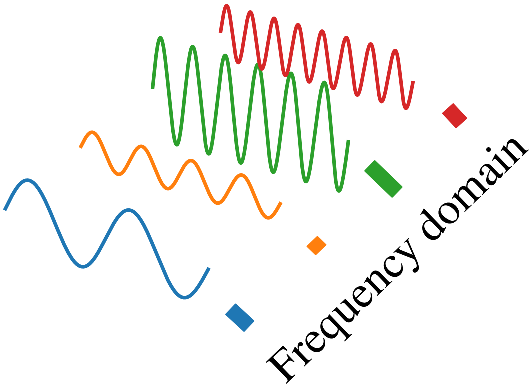


Fourier Transform (Recap)



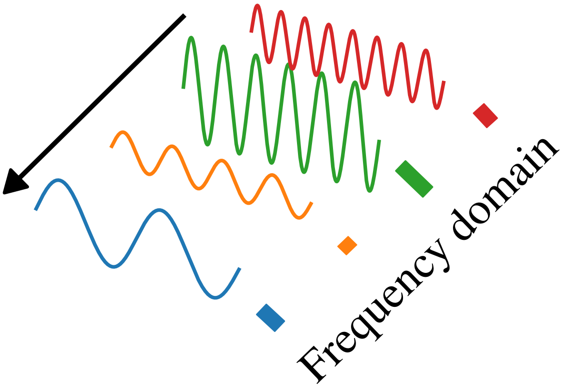
Fourier Transform (Recap)



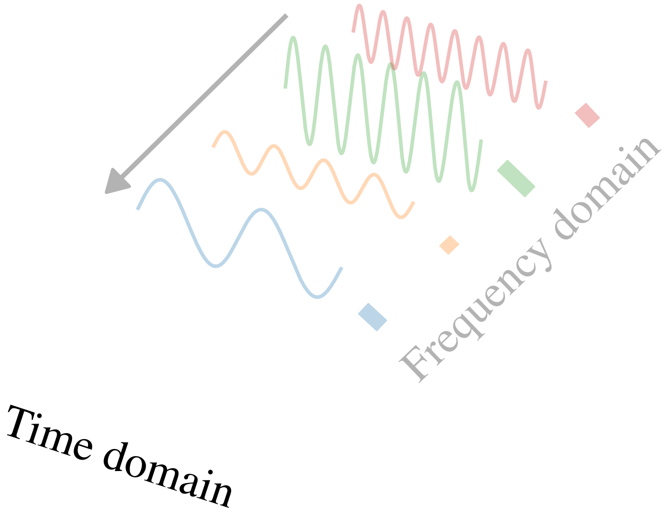

Frequency domain

Fourier Transform (Recap)

Inverse Fourier Transform

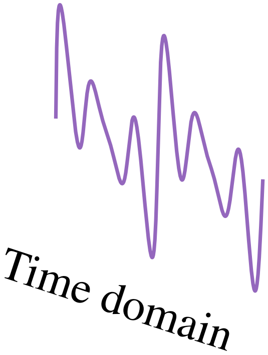


Inverse Fourier Transform



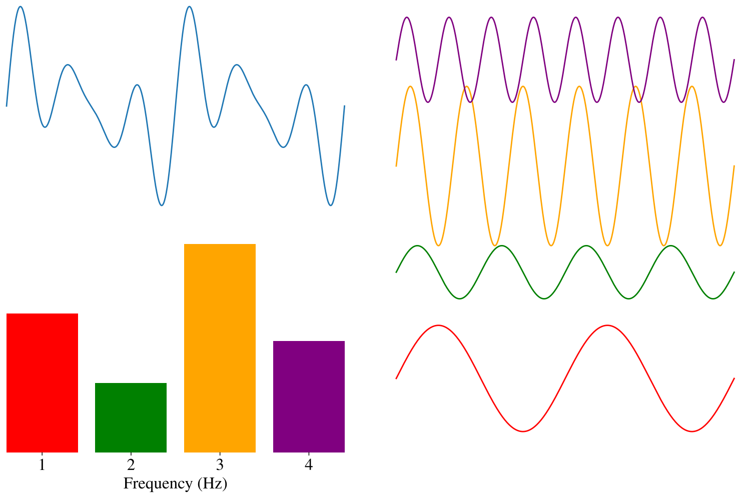
Fourier Transform (Recap)

Inverse Fourier Transform

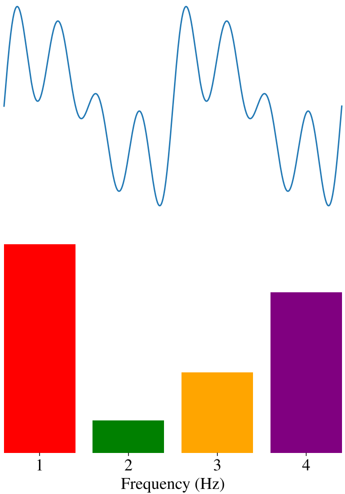


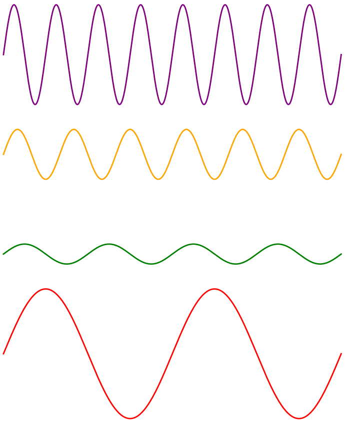
Fourier Transform (Recap)

Examples



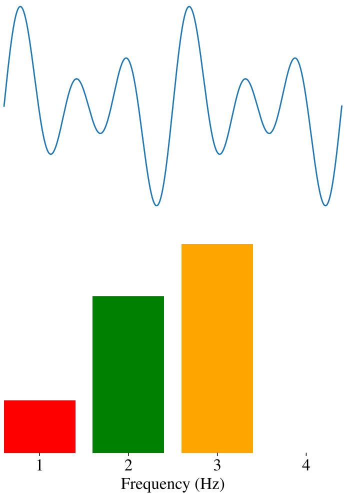
Examples

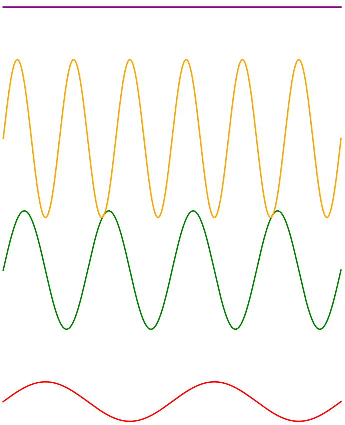




Fourier Transform (Recap)

Examples





Fourier Transform (Recap)

Uncertainty principle in signal processing

- Even though we have access to the information about the frequencies when performing the Fourier transform, we lose information about time

Uncertainty principle in signal processing

- Even though we have access to the information about the frequencies when performing the Fourier transform, we lose information about time
- It is impossible to have at the same time both perfect time and frequency resolution simultaneously. There is always a trade-off of information between the two

Uncertainty principle in signal processing

- Even though we have access to the information about the frequencies when performing the Fourier transform, we lose information about time
- It is impossible to have at the same time both perfect time and frequency resolution simultaneously. There is always a trade-off of information between the two
- This is a manifestation of the Heisenberg uncertainty principle

Fourier Transform (Recap)

Uncertainty principle in signal processing

- Time and frequency domains are two extremes of the uncertainty principle

Uncertainty principle in signal processing

- Time and frequency domains are two extremes of the uncertainty principle
- You can either know exactly what the value of a function is at every time but at the cost of being completely ignorant about what frequencies are unfolding at that time point

Uncertainty principle in signal processing

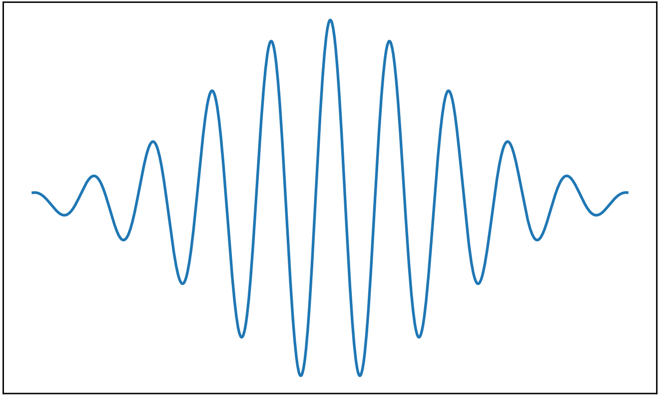
- Time and frequency domains are two extremes of the uncertainty principle
- You can either know exactly what the value of a function is at every time but at the cost of being completely ignorant about what frequencies are unfolding at that time point
- On the other hand, we can know exactly what frequencies are present in the signal, but we have no idea about the temporal dynamics of them

Fourier Transform (Recap)

Uncertainty principle in signal processing

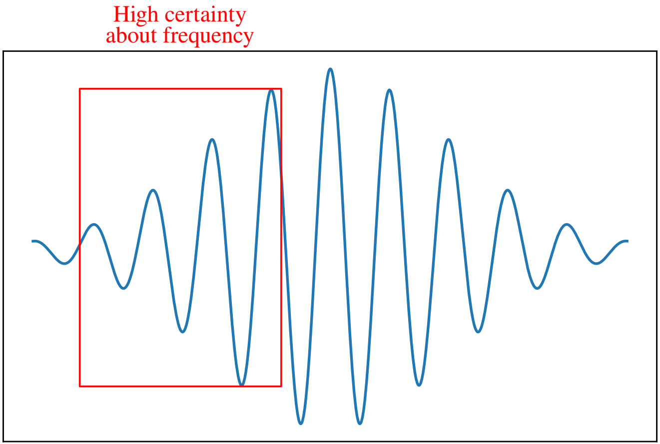
- Time and frequency domains are two extremes of the uncertainty principle
- You can either know exactly what the value of a function is at every time but at the cost of being completely ignorant about what frequencies are unfolding at that time point
- On the other hand, we can know exactly what frequencies are present in the signal, but we have no idea about the temporal dynamics of them
- How we can find the "sweet point" in this trade-off (we already saw that STFT is not an optimal solution)

Uncertainty principle in signal processing (example)

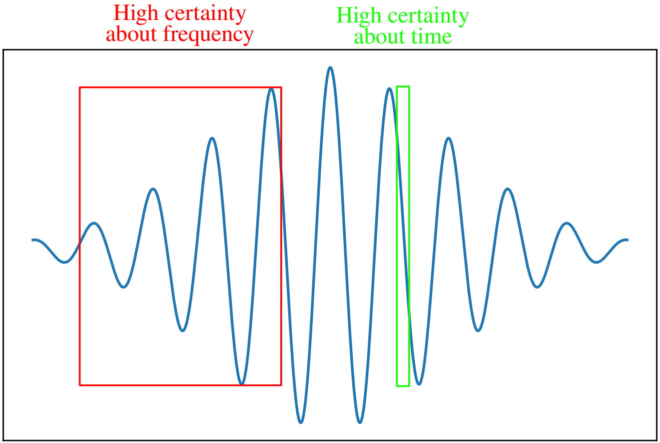


Fourier Transform (Recap)

Uncertainty principle in signal processing (example)



Uncertainty principle in signal processing (example)



Wavelets - Localized Functions

- What we do in Fourier transform is to decompose the function in sum of sines and cosines

Wavelets - Localized Functions

Analyzing Functions (in Fourier Transform)

- What we do in Fourier transform is to decompose the function in sum of sines and cosines
- These sines and cosines is what we called analyzing functions

Analyzing Functions (in Fourier Transform)

- What we do in Fourier transform is to decompose the function in sum of sines and cosines
- These sines and cosines is what we called analyzing functions
- The issue with sines and cosines is that the look the same everywhere.

Analyzing Functions (in Fourier Transform)

- What we do in Fourier transform is to decompose the function in sum of sines and cosines
- These sines and cosines is what we called analyzing functions
- The issue with sines and cosines is that the look the same everywhere.
- But what if we modify it a little bit? We still need oscillations up and down since this is the foundation of frequency representation

Wavelets - Localized Functions

Analyzing Functions (in Fourier Transform)

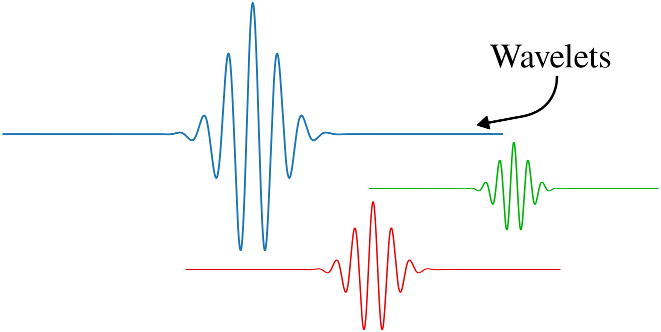
- What we do in Fourier transform is to decompose the function in sum of sines and cosines
- These sines and cosines is what we called analyzing functions
- The issue with sines and cosines is that the look the same everywhere.
- But what if we modify it a little bit? We still need oscillations up and down since this is the foundation of frequency representation
- Can we somehow restrain it in time?

Wavelets - Localized Functions

Wavelet Transform

- Restraining the analyzing function in time is precisely the idea of the wavelet transform

- Restraining the analyzing function in time is precisely the idea of the wavelet transform
- Wavelet transform is a mathematical tool that uses specialized functions called wavelets to analyze the signal



What's a Wavelet

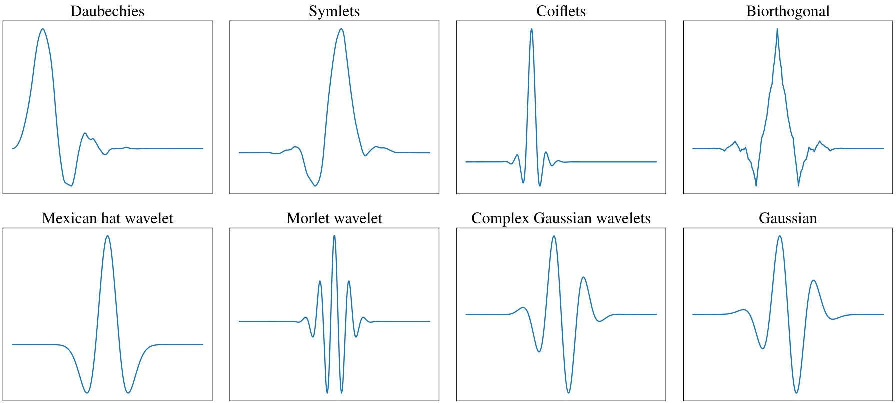
- A wavelet is short-lived wave-like oscillation that is localized in time

Wavelets - Localized Functions

What's a Wavelet

- A wavelet is short-lived wave-like oscillation that is localized in time
- Mathematically speaking, a wavelet is not just a function but a whole family of functions that satisfy certain requirements

- A wavelet is short-lived wave-like oscillation that is localized in time
- Mathematically speaking, a wavelet is not just a function but a whole family of functions that satisfy certain requirements



Mathematical Requirements for Wavelets

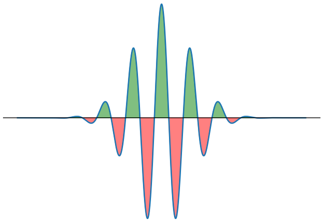
- To be considered a proper wavelet, a function $\Psi(t)$ should satisfy two main constraints:

- To be considered a proper wavelet, a function $\Psi(t)$ should satisfy two main constraints:
- It should have zero mean:

$$\int_{-\infty}^{\infty} \Psi(t)dt = 0$$

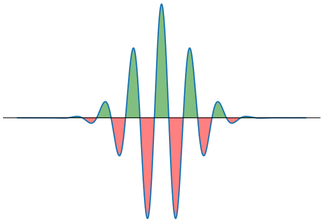
- To be considered a proper wavelet, a function $\Psi(t)$ should satisfy two main constraints:
- It should have zero mean:

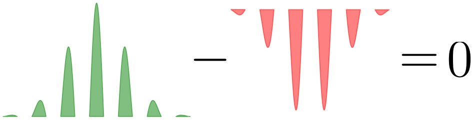
$$\int_{-\infty}^{\infty} \Psi(t)dt = 0$$



- To be considered a proper wavelet, a function $\Psi(t)$ should satisfy two main constraints:
- It should have zero mean:

$$\int_{-\infty}^{\infty} \Psi(t)dt = 0$$

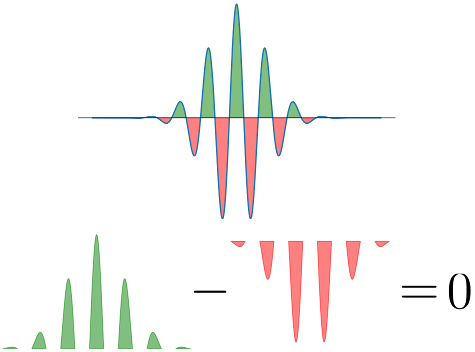




- To be considered a proper wavelet, a function $\Psi(t)$ should satisfy two main constraints:
- It should have zero mean:

$$\int_{-\infty}^{\infty} \Psi(t)dt = 0$$

- This is called the admissibility condition. More formally, it's phrased that the wavelet function should have no zero-frequency component (which is the average value of the function)



Wavelets - Localized Functions

Mathematical Requirements for Wavelets

- Please notice that a sine function also satisfies the admissibility condition, so we need something else.

Wavelets - Localized Functions

Mathematical Requirements for Wavelets

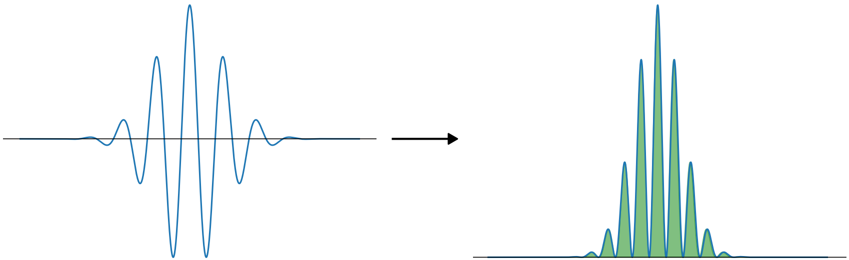
- Please notice that a sine function also satisfies the admissibility condition, so we need something else.
- The second condition is that the wavelet function should have finite energy:

$$\int_{-\infty}^{\infty} |\Psi(t)|^2 dt < \infty$$

Mathematical Requirements for Wavelets

- Please notice that a sine function also satisfies the admissibility condition, so we need something else.
- The second condition is that the wavelet function should have finite energy:

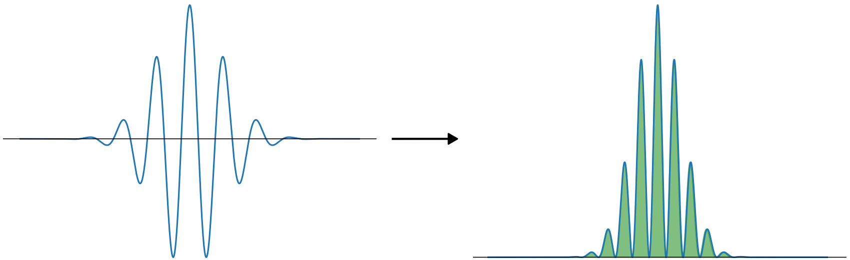
$$\int_{-\infty}^{\infty} |\Psi(t)|^2 dt < \infty$$



- Please notice that a sine function also satisfies the admissibility condition, so we need something else.
- The second condition is that the wavelet function should have finite energy:

$$\int_{-\infty}^{\infty} |\Psi(t)|^2 dt < \infty$$

- This is what makes the function localized in time

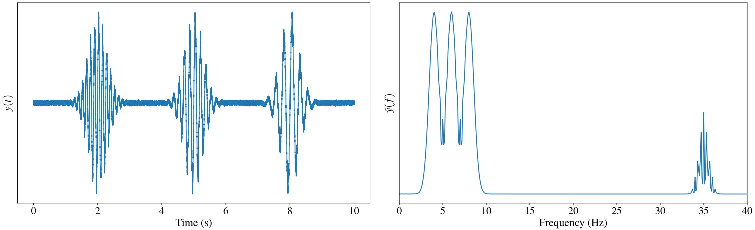


Wavelet Transform

Fourier Transform

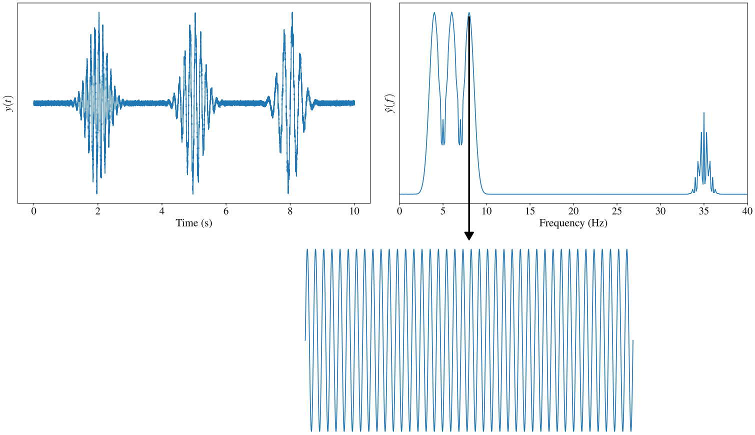
- With the FT we go from a one-dimensional function on time $y(t)$ to a one-dimensional function in frequency $\hat{y}(w)$

- With the FT we go from a one-dimensional function on time $y(t)$ to a one-dimensional function in frequency $\hat{y}(w)$



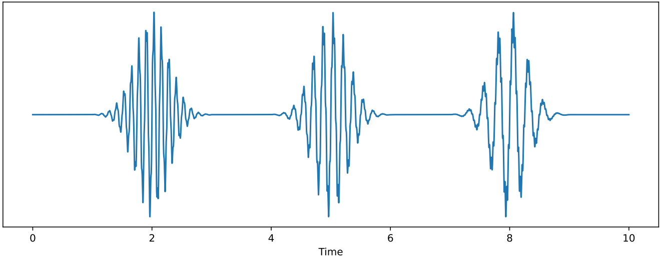
Wavelet Transform
Fourier Transform

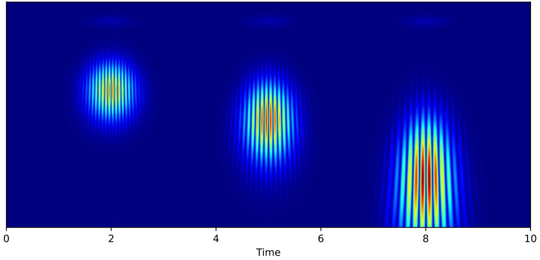
- With the FT we go from a one-dimensional function on time $y(t)$ to a one-dimensional function in frequency $\hat{y}(w)$



Wavelet Transform

Wavelet Transform

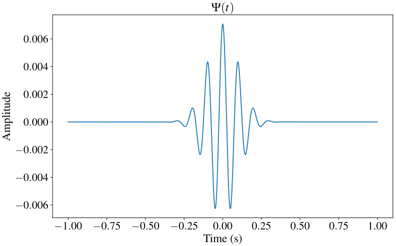




Wavelet Transform

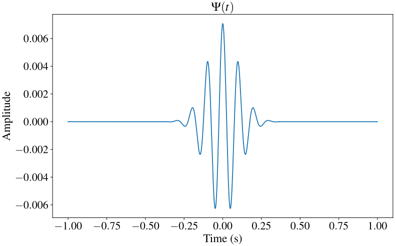
Mother Wavelet Modifications - Time and Frequency Shift

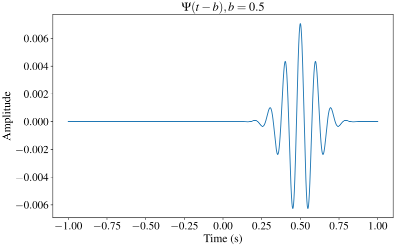
- To compute the wavelet transform we need to modify our initial wavelet function $\Psi(t)$ (called mother wavelet) to obtain a slightly modified version of the mother wavelet called data wavelets



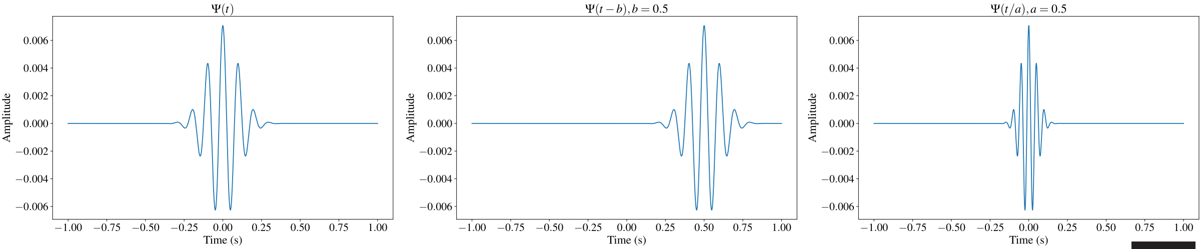
Mother Wavelet Modifications - Time and Frequency Shift

- To compute the wavelet transform we need to modify our initial wavelet function $\Psi(t)$ (called mother wavelet) to obtain a slightly modified version of the mother wavelet called data wavelets
- More precisely we require shifting in time $\Psi(t-b)$





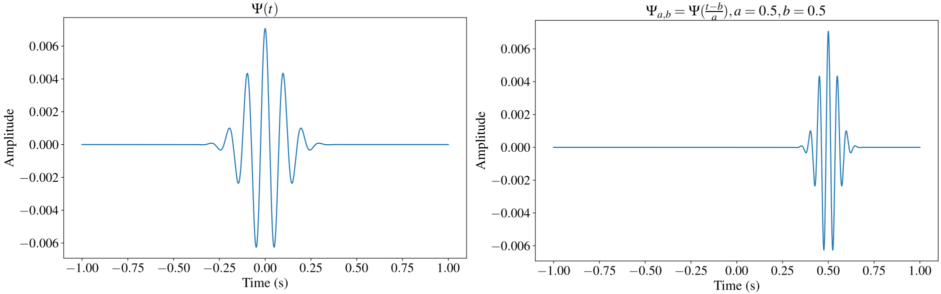
- To compute the wavelet transform we need to modify our initial wavelet function $\Psi(t)$ (called mother wavelet) to obtain a slightly modified version of the mother wavelet called data wavelets
- More precisely we require shifting in time $\Psi(t-b)$
- And frequency shift (or time scale) $\Psi(t/a)$



Wavelet Transform

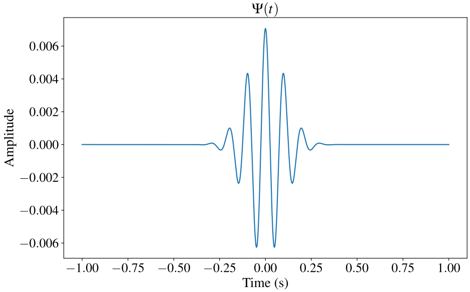
Scaled and translated wavelet

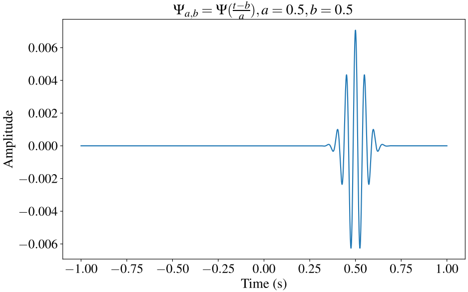
- We can now define the scaled and translated wavelet as $\Psi_{a,b} = \Psi\left(\frac{t-b}{a}\right)$



Scaled and translated wavelet

- We can now define the scaled and translated wavelet as $\Psi_{a,b} = \Psi\left(\frac{t-b}{a}\right)$
- Therefore, the wavelet transform of our signal at a particular scale a and time b will be equal to the "contribution" of $\Psi_{a,b}$ to our signal $y(t)$





Computing local similarity

- To compute the local similarity between our signal and the wavelet, we need to calculate the scalar product $\langle y(t), \Psi_{a,b}(t) \rangle$

- To compute the local similarity between our signal and the wavelet, we need to calculate the scalar product $\langle y(t), \Psi_{a,b}(t) \rangle$
- For those very good at maths, you might have understood already why we need to do this. For those who are not, let's unpack it a little bit

Computing local similarity

- To compute the local similarity between our signal and the wavelet, we need to calculate the scalar product $\langle y(t), \Psi_{a,b}(t)\rangle$
- For those very good at maths, you might have understood already why we need to do this. For those who are not, let's unpack it a little bit
- The scalar or inner product between functions is defined as an integral over some interval $[c, d]$:

$$\langle y(t), \Psi_{a,b}(t)\rangle = \int_c^d y(t)\Psi_{a,b}(t)dt$$

- Let's see why this integral is the inner product between functions, and why this is a measure of similarity

- Let's see why this integral is the inner product between functions, and why this is a measure of similarity
- In high school, we learned that the inner product between two vectors $\mathbf{a} \in \mathbb{R}^c$ and $\mathbf{b} \in \mathbb{R}^c$ is $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\intercal \mathbf{b}$

- Let's see why this integral is the inner product between functions, and why this is a measure of similarity
- In high school, we learned that the inner product between two vectors $\mathbf{a} \in \mathbb{R}^c$ and $\mathbf{b} \in \mathbb{R}^c$ is $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\top \mathbf{b}$
- We know that $\mathbf{a}^\top \mathbf{b} = \sum_{i=1}^{c} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots a_c b_c$, *i.e.*, we multiply point-wise both vector and sum everything

- Let's see why this integral is the inner product between functions, and why this is a measure of similarity
- In high school, we learned that the inner product between two vectors $\mathbf{a} \in \mathbb{R}^c$ and $\mathbf{b} \in \mathbb{R}^c$ is $\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^\top \mathbf{b}$
- We know that $\mathbf{a}^\top \mathbf{b} = \sum_{i=1}^{c} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots a_c b_c$, *i.e.*, we multiply point-wise both vector and sum everything
- We also know from high school that $\langle \mathbf{a}, \mathbf{b} \rangle = \|\mathbf{a}\| \|\mathbf{b}\| \cos(\theta)$, where θ is the angle between vector $\mathbf{a}$ and $\mathbf{b}$

- If both vectors are unitary, we have that $\langle \mathbf{a}, \mathbf{b} \rangle \in [-1, 1]$, where:

Wavelet Transform
Computing local similarity

- If both vectors are unitary, we have that $\langle \mathbf{a}, \mathbf{b} \rangle \in [-1, 1]$, where:
  1. $\langle \mathbf{a}, \mathbf{b} \rangle = 1$ when $\theta = 0$ so we have maximum alignment, *i.e.*, both vectors are parallel and pointing in the same direction

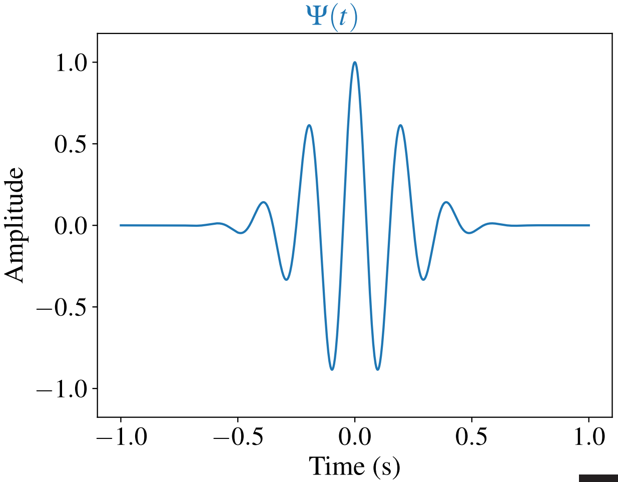
- If both vectors are unitary, we have that $\langle \mathbf{a}, \mathbf{b} \rangle \in [-1, 1]$, where:
  1. $\langle \mathbf{a}, \mathbf{b} \rangle = 1$ when $\theta = 0$ so we have maximum alignment, *i.e.*, both vectors are parallel and pointing in the same direction
  2. $\langle \mathbf{a}, \mathbf{b} \rangle = 0$ when the vectors are orthogonal

- If both vectors are unitary, we have that $\langle \mathbf{a}, \mathbf{b} \rangle \in [-1, 1]$, where:
  1. $\langle \mathbf{a}, \mathbf{b} \rangle = 1$ when $\theta = 0$ so we have maximum alignment, *i.e.*, both vectors are parallel and pointing in the same direction
  2. $\langle \mathbf{a}, \mathbf{b} \rangle = 0$ when the vectors are orthogonal
  3. $\langle \mathbf{a}, \mathbf{b} \rangle = -1$ when $\theta = 180$

- If both vectors are unitary, we have that $\langle \mathbf{a}, \mathbf{b} \rangle \in [-1, 1]$, where:
  1. $\langle \mathbf{a}, \mathbf{b} \rangle = 1$ when $\theta = 0$ so we have maximum alignment, *i.e.*, both vectors are parallel and pointing in the same direction
  2. $\langle \mathbf{a}, \mathbf{b} \rangle = 0$ when the vectors are orthogonal
  3. $\langle \mathbf{a}, \mathbf{b} \rangle = -1$ when $\theta = 180$
- So inner product gives us a measure of similarity in the vector domain, and this idea extends to the function domain as well

- Let's look at a graphical example in the function domain

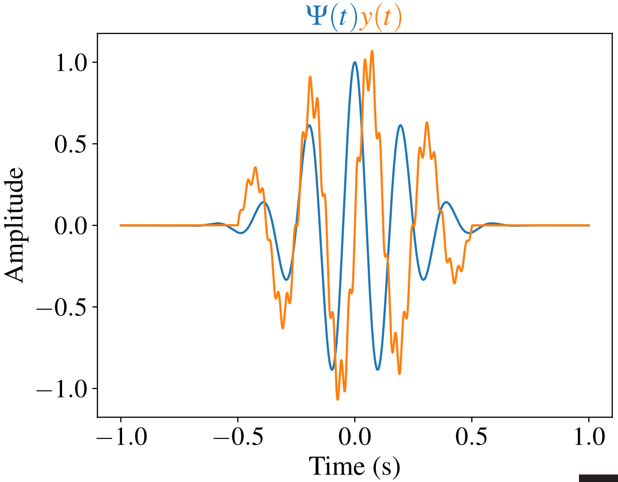
- Let's look at a graphical example in the function domain



Wavelet Transform

Computing local similarity

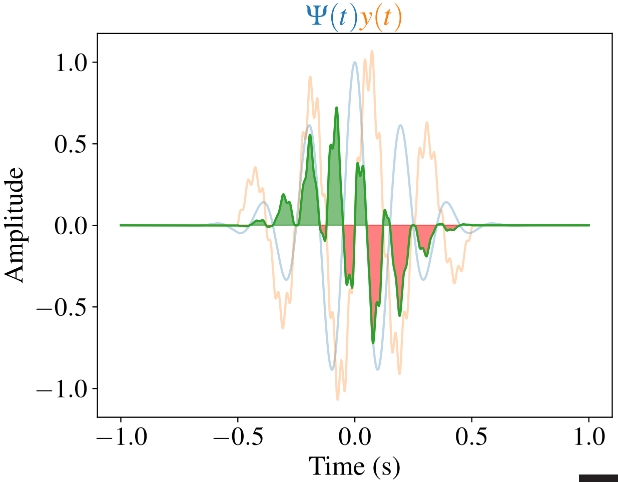
- Let's look at a graphical example in the function domain



Wavelet Transform

Computing local similarity

- Let's look at a graphical example in the function domain
- We multiply "point-wise" both functions

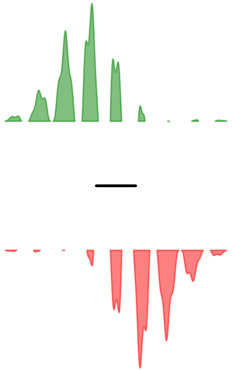


Wavelet Transform

Computing local similarity

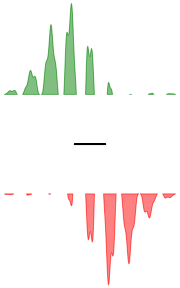
- Let's look at a graphical example in the function domain
- We multiply "point-wise" both functions
- And then we sum the results (the integral)

$$\int \Psi(t) y(t) =$$



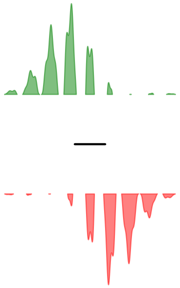
- Notice that this is precisely what we do in scalar products in vectors: we multiply point-wise and then sum the values.

$$\int \Psi(t) y(t) =$$



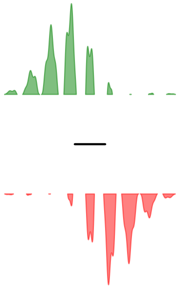
- Notice that this is precisely what we do in scalar products in vectors: we multiply point-wise and then sum the values.
- If you remember from high school, scalar products in vectors give us a measure of similarity between both vectors

$$\int \Psi(t)y(t) =$$



- Notice that this is precisely what we do in scalar products in vectors: we multiply point-wise and then sum the values.
- If you remember from high school, scalar products in vectors give us a measure of similarity between both vectors
- The intuition in functions is the same, we're computing a similarity between both functions (how good they match each other)

$$\int \Psi(t) y(t) =$$



- We know how to compute the local similarity between the signal and our wavelet

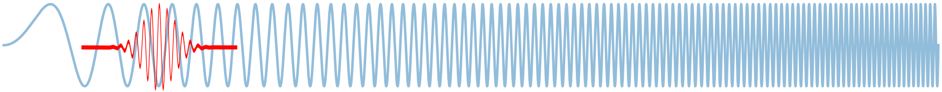
Wavelet Transform
Convolutions

- We know how to compute the local similarity between the signal and our wavelet
- Now we have to keep the frequency fixed and we start varying the time parameter b in $\Psi_{a,b}$

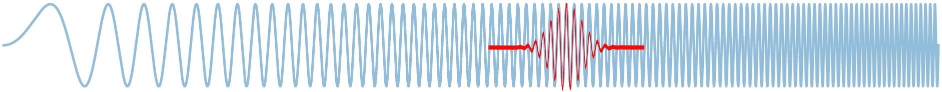
Convolutions

- We know how to compute the local similarity between the signal and our wavelet
- Now we have to keep the frequency fixed and we start varying the time parameter b in $\Psi_{a,b}$
- This slides the wavelet across the signal

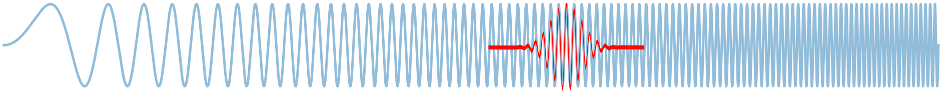
- We know how to compute the local similarity between the signal and our wavelet
- Now we have to keep the frequency fixed and we start varying the time parameter b in $\Psi_{a,b}$
- This slides the wavelet across the signal



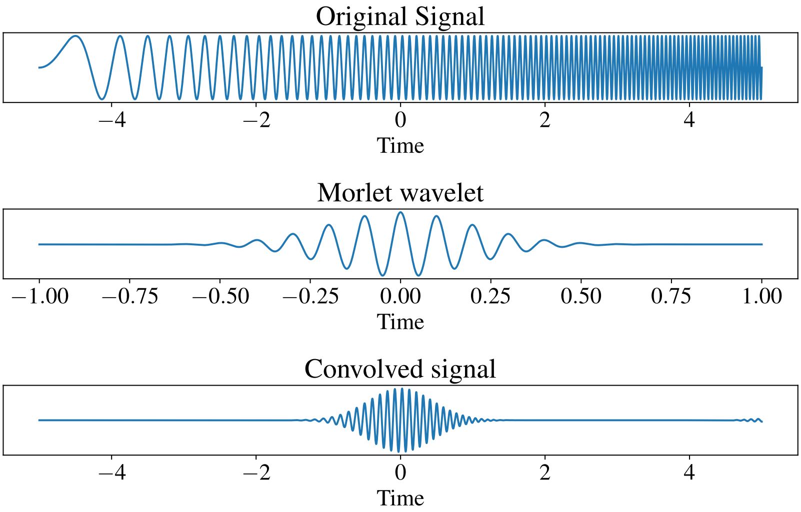
- We know how to compute the local similarity between the signal and our wavelet
- Now we have to keep the frequency fixed and we start varying the time parameter b in $\Psi_{a,b}$
- This slides the wavelet across the signal



- We know how to compute the local similarity between the signal and our wavelet
- Now we have to keep the frequency fixed and we start varying the time parameter b in $\Psi_{a,b}$
- This slides the wavelet across the signal
- So this is basically a convolution



Wavelet Transform
Convolutions

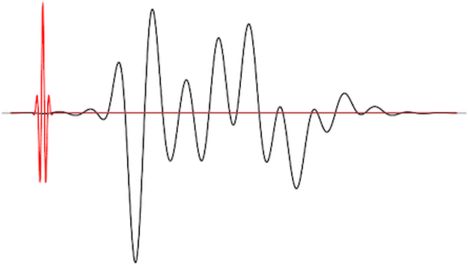


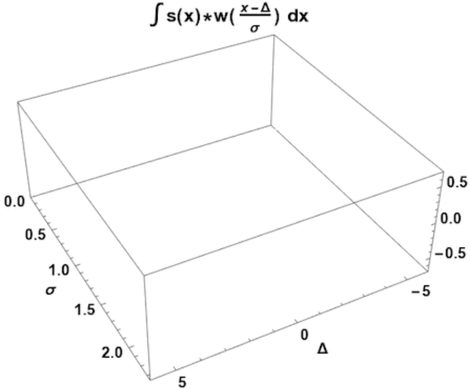
- Notice that the output of the convolution starts to oscillate when the signal and the wavelet are in synchrony

- Notice that the output of the convolution starts to oscillate when the signal and the wavelet are in synchrony
- To get the full wavelet transform, we need to compute the convolution for several values of a in $\Psi_{a,b}$ to scan our signal

- Notice that the output of the convolution starts to oscillate when the signal and the wavelet are in synchrony
- To get the full wavelet transform, we need to compute the convolution for several values of a in $\Psi_{a,b}$ to scan our signal
- However, notice that we get some zero values in the convolution before when we are in perfect synchrony

- Notice that the output of the convolution starts to oscillate when the signal and the wavelet are in synchrony
- To get the full wavelet transform, we need to compute the convolution for several values of a in $\Psi_{a,b}$ to scan our signal
- However, notice that we get some zero values in the convolution before when we are in perfect synchrony
- The reason for this is that we're only taking the real part of the Morlet wavelet to compute our convolution, so now it's time to include the complex values in our calculations





Source: Wikipedia

Wavelet Transform

Complex Morlet wavelet

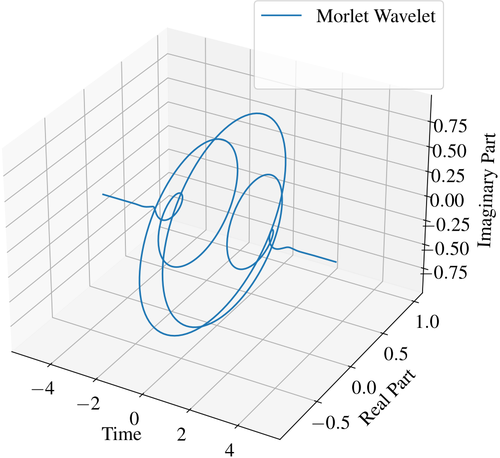
- We've been using only the real part of the Morlet wavelet previously

Wavelet Transform
Complex Morlet wavelet

- We've been using only the real part of the Morlet wavelet previously
- The "complete" Morlet wavelet is given by $\Psi(t) = ke^{iw_0 t}e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$

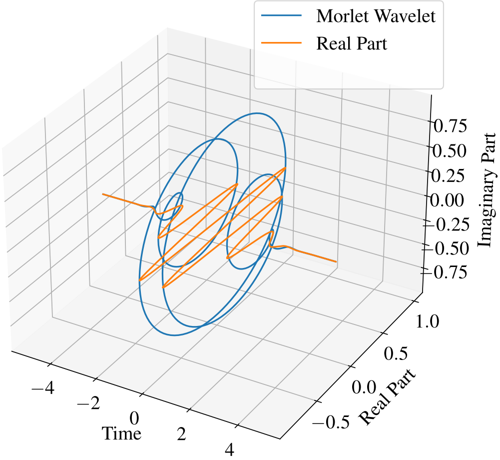
TELECOM Paris IP PARIS

- We've been using only the real part of the Morlet wavelet previously
- The "complete" Morlet wavelet is given by $\Psi(t) = k e^{i w_0 t} e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$



Complex Morlet wavelet

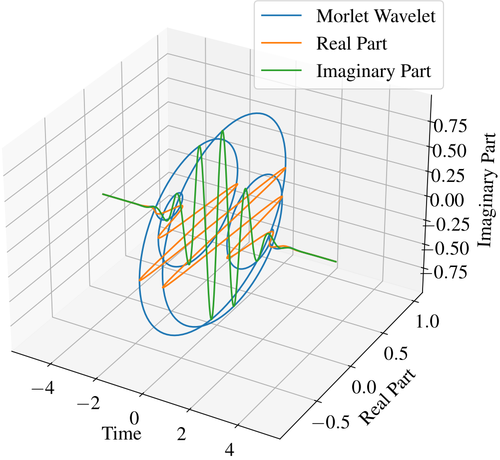
- We've been using only the real part of the Morlet wavelet previously
- The "complete" Morlet wavelet is given by $\Psi(t) = k e^{i w_0 t} e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$



Wavelet Transform

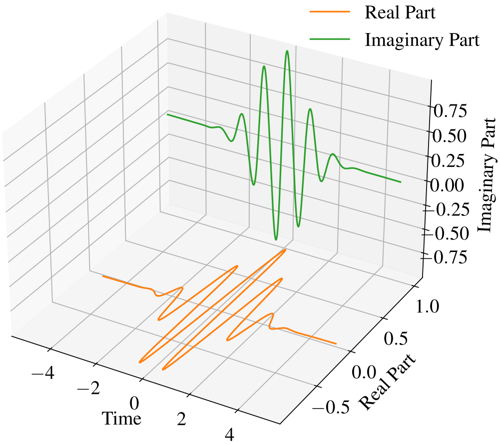
Complex Morlet wavelet

- We've been using only the real part of the Morlet wavelet previously
- The "complete" Morlet wavelet is given by $\Psi(t) = ke^{iw_0t}e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$



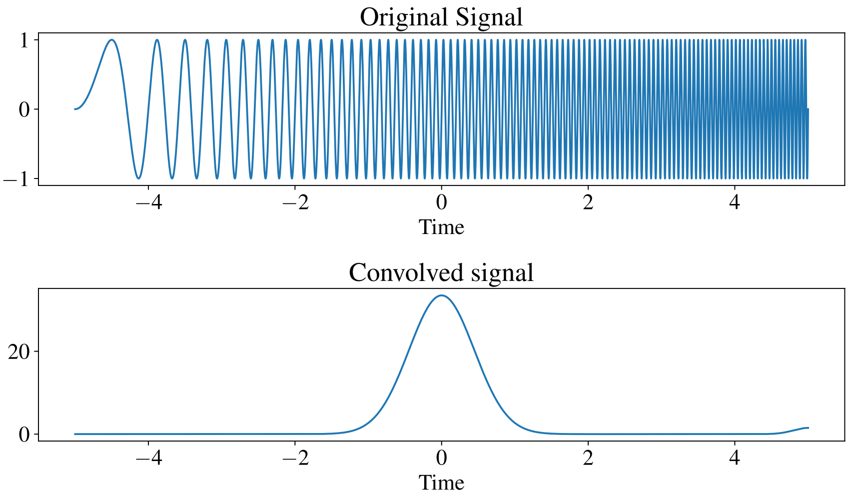
Wavelet Transform
Complex Morlet wavelet

- We've been using only the real part of the Morlet wavelet previously
- The "complete" Morlet wavelet is given by $\Psi(t) = ke^{iw_0t}e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$



Wavelet Transform

Convolution with the “complete” Morlet wavelet



- A wavelet dictionary is constructed from a mother wavelet Ψ of zero average

$$\int_{-\infty}^{+\infty} \Psi(t)dt = 0 \tag{1}$$

Wavelet Transform

Continuos wavelet transform

- A wavelet dictionary is constructed from a mother wavelet Ψ of zero average

$$\int_{-\infty}^{+\infty} \Psi(t)dt = 0 \tag{1}$$

- The mother wavelet is dilated with a scale parameter s, and translated by u

$$\mathcal{D} = \left\{ \Psi_{u,s}(t) = \frac{1}{\sqrt{s}} \Psi \left(\frac{t-u}{s} \right) \right\}_{u \in \mathbb{R}, s>0} \tag{2}$$

Continuos wavelet transform

- A wavelet dictionary is constructed from a mother wavelet Ψ of zero average

$$\int_{-\infty}^{+\infty} \Psi(t)dt = 0 \qquad (1)$$

- The mother wavelet is dilated with a scale parameter s, and translated by u

$$\mathcal{D} = \left\{ \Psi_{u,s}(t) = \frac{1}{\sqrt{s}} \Psi\left(\frac{t-u}{s}\right) \right\}_{u \in \mathbb{R}, s>0} \qquad (2)$$

- The continuous wavelet transform of f at any scale s and position u is the projection of f on the corresponding wavelet atom

$$Wf(u,s) = \langle f, \Psi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{s}} \Psi^{*}\left(\frac{t-u}{s}\right) dt \qquad (3)$$

- Notice that the wavelet transform can be rewritten as a convolution product

$$Wf(u,s) = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{s}} \Psi^* \left(\frac{t-u}{s} \right) dt = f * \bar{\Psi}(u), \tag{4}$$

with

$$\bar{\Psi}(t) = \frac{1}{\sqrt{s}} \Psi^* \left(\frac{-t}{s} \right) \tag{5}$$

- The inverse wavelet transform is important since it allows us to reconstruct signals from the wavelet domain

Wavelet Transform

Inverse wavelet transform

- The inverse wavelet transform is important since it allows us to reconstruct signals from the wavelet domain
- Let C_Ψ be the admissible constant of $\Psi(t)$ given by:

$$C_\Psi = \int_0^{+\infty} \frac{|\hat{\Psi}(w)|^2}{w} dw < +\infty, \tag{6}$$

where $\hat{\Psi}(w)$ is the Fourier transform of the wavelet

Inverse wavelet transform

- The inverse wavelet transform is important since it allows us to reconstruct signals from the wavelet domain
- Let C_Ψ be the admissible constant of $\Psi(t)$ given by:

$$C_\Psi = \int_0^{+\infty} \frac{|\hat{\Psi}(w)|^2}{w} dw < +\infty, \tag{6}$$

where $\hat{\Psi}(w)$ is the Fourier transform of the wavelet

- We then have that:

$$f(t) = \frac{1}{C_\Psi} \int_0^{+\infty} \int_{-\infty}^{+\infty} Wf(u,s) \frac{1}{\sqrt{s}} \Psi\left(\frac{t-u}{s}\right) du \frac{ds}{s^2} \tag{7}$$

Discrete Wavelet Transform

Discrete Wavelet Transform

- The Discrete Wavelet Transform (DWT) is an extensive topic and we could talk about it for a whole lecture and even more. But we're not doing so, if you're interested on it, I encourage you to read the book of Stéphane Mallat

Discrete Wavelet Transform

- The Discrete Wavelet Transform (DWT) is an extensive topic and we could talk about it for a whole lecture and even more. But we're not doing so, if you're interested on it, I encourage you to read the book of Stéphane Mallat
- If we want just to use the DWT for some application (like image compression) you just need to know that we can compute the DWT of a signal f by passing it through a series of low-pass and high-pass filters

Discrete Wavelet Transform

- The Discrete Wavelet Transform (DWT) is an extensive topic and we could talk about it for a whole lecture and even more. But we're not doing so, if you're interested on it, I encourage you to read the book of Stéphane Mallat
- If we want just to use the DWT for some application (like image compression) you just need to know that we can compute the DWT of a signal f by passing it through a series of low-pass and high-pass filters
- Let $f[n]$ be the discrete signal obtained by a low-pass filtering of some continous signal $\bar{f}(t)$ and uniform sampling at intervals N^{-1}

Discrete Wavelet Transform

- Let $\Psi(t)$ be a wavelet with a support included in $[-K/2, K/2]$. For $1 \leq a^j \leq NK^{-1}$, a discrete wavelet scaled by a^j is defined by

$$\Psi_j[n] = \frac{1}{\sqrt{a^j}} \Psi\left(\frac{n}{a^j}\right) \tag{8}$$

Discrete Wavelet Transform

- Let $\Psi(t)$ be a wavelet with a support included in $[-K/2, K/2]$. For $1 \leq a^j \leq NK^{-1}$, a discrete wavelet scaled by a^j is defined by

$$\Psi_j[n] = \frac{1}{\sqrt{a^j}} \Psi\left(\frac{n}{a^j}\right) \tag{8}$$

- To avoid border problems, we treat $f[n]$ and the wavelets $\Psi_j[n]$ as periodic signals of period N. The DWT can then be written as a circular convolution with $\bar{\Psi}_j[n] = \Psi^*[-n]$

$$Wf[n, a^j] = \sum_{m=0}^{N-1} f[m]\Psi^*[m-n] = f \circledast \bar{\Psi}_j[n] \tag{9}$$

Discrete Wavelet Transform

- A DWT computed up to a scale a^j is not a complete signal representation. We need to add low frequencies $Lf[n, a^j]$ corresponding to scales larger than a^j

Discrete Wavelet Transform

- A DWT computed up to a scale a^j is not a complete signal representation. We need to add low frequencies $Lf[n, a^j]$ corresponding to scales larger than a^j
- Let $\phi(t)$ be a scaling function defined as an aggregation of wavelets at scales larger than 1. I'm not going into the math details, but this scaling function can be interpreted as the impulse response of a low-pass filter

$$\phi_J[n] = \frac{1}{\sqrt{a^j}}\phi\left(\frac{n}{a^j}\right) \text{ for } n \in [-N/2, N/2] \tag{10}$$

Discrete Wavelet Transform

- A DWT computed up to a scale a^j is not a complete signal representation. We need to add low frequencies $Lf[n, a^j]$ corresponding to scales larger than a^j
- Let $\phi(t)$ be a scaling function defined as an aggregation of wavelets at scales larger than 1. I'm not going into the math details, but this scaling function can be interpreted as the impulse response of a low-pass filter

$$\phi_J[n] = \frac{1}{\sqrt{a^j}}\phi\left(\frac{n}{a^j}\right) \text{ for } n \in [-N/2, N/2] \tag{10}$$

- Let $\bar{\phi}_J[n] = \phi_J^*[-n]$, the low frequencies are carried by

$$Lf[n, a^j] = \sum_{m=0}^{N-1} f[m]\phi_J^*[m-n] = f \circledast \bar{\phi}_J[n] \tag{11}$$

Discrete Wavelet Transform

- The DWT of a signal f is calculated by passing it through a series of filters. First the samples are passed through a low-pass filter with impulse response g resulting in $y[n] = x * g[n]$

Discrete Wavelet Transform

- The DWT of a signal f is calculated by passing it through a series of filters. First the samples are passed through a low-pass filter with impulse response g resulting in $y[n] = x * g[n]$
- The signal is also decomposed simultaneously using a high-pass filter h. The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass)

Discrete Wavelet Transform

- The DWT of a signal f is calculated by passing it through a series of filters. First the samples are passed through a low-pass filter with impulse response g resulting in $y[n] = x * g[n]$
- The signal is also decomposed simultaneously using a high-pass filter h. The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass)
- It is important that the two filters are related to each other and they are known as a quadrature mirror filter

Discrete Wavelet Transform

- The DWT of a signal f is calculated by passing it through a series of filters. First the samples are passed through a low-pass filter with impulse response g resulting in $y[n] = x * g[n]$
- The signal is also decomposed simultaneously using a high-pass filter h. The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass)
- It is important that the two filters are related to each other and they are known as a quadrature mirror filter
- Since half the frequencies of the signal have now been removed, half the samples can be discarded according to Nyquist's rule

Discrete Wavelet Transform

- The DWT of a signal f is calculated by passing it through a series of filters. First the samples are passed through a low-pass filter with impulse response g resulting in $y[n] = x * g[n]$
- The signal is also decomposed simultaneously using a high-pass filter h. The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass)
- It is important that the two filters are related to each other and they are known as a quadrature mirror filter
- Since half the frequencies of the signal have now been removed, half the samples can be discarded according to Nyquist's rule
- The filter output of the low-pass filter g in is then subsampled by 2 and further processed by passing it again through a new low-pass filter g and a high-pass filter h with half the cut-off frequency of the previous one

Discrete Wavelet Transform

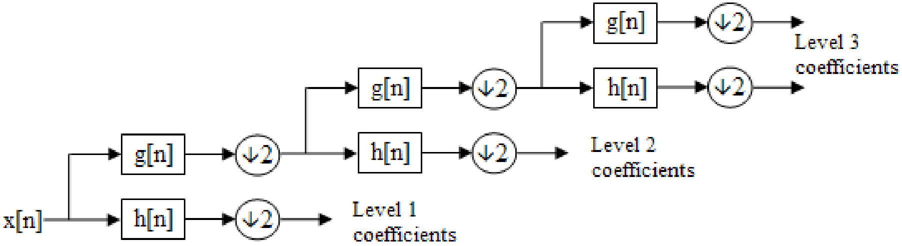
Filter bank

- In practice this means that the DWT is always implemented as a filter bank: a cascade of high-pass and low-pass filters

Discrete Wavelet Transform

Filter bank

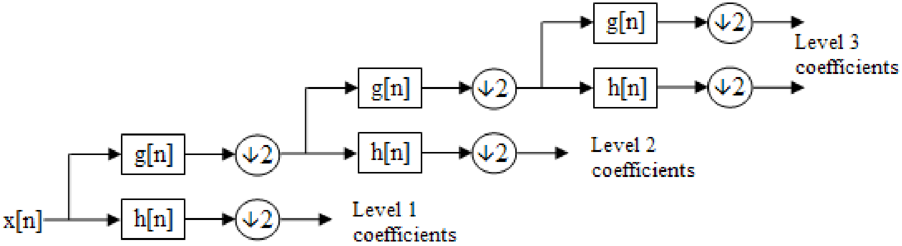
- In practice this means that the DWT is always implemented as a filter bank: a cascade of high-pass and low-pass filters



Discrete Wavelet Transform

Filter bank

- In practice this means that the DWT is always implemented as a filter bank: a cascade of high-pass and low-pass filters
- This is because filter banks are a very efficient way of splitting a signal into several frequency sub-bands



Discrete Wavelet Transform
Filter bank

- To apply the DWT on a signal, we start with the smallest scale

Discrete Wavelet Transform

Filter bank

- To apply the DWT on a signal, we start with the smallest scale
- Small scales correspond to high frequencies, so this means that we first analyze high-frequency behavior

- To apply the DWT on a signal, we start with the smallest scale
- Small scales correspond to high frequencies, so this means that we first analyze high-frequency behavior
- At the second stage, the scale increases with a factor of two (the frequency decreases with a factor of two), so we are analyzing behavior around half of the maximum frequency

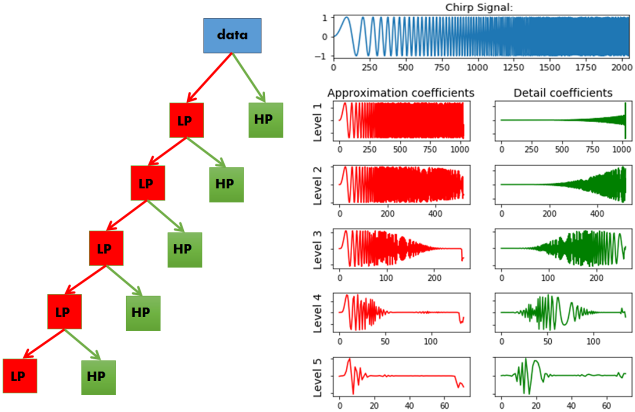
Discrete Wavelet Transform

Filter bank

- To apply the DWT on a signal, we start with the smallest scale
- Small scales correspond to high frequencies, so this means that we first analyze high-frequency behavior
- At the second stage, the scale increases with a factor of two (the frequency decreases with a factor of two), so we are analyzing behavior around half of the maximum frequency
- This goes on and on until we reach the maximum decomposition level

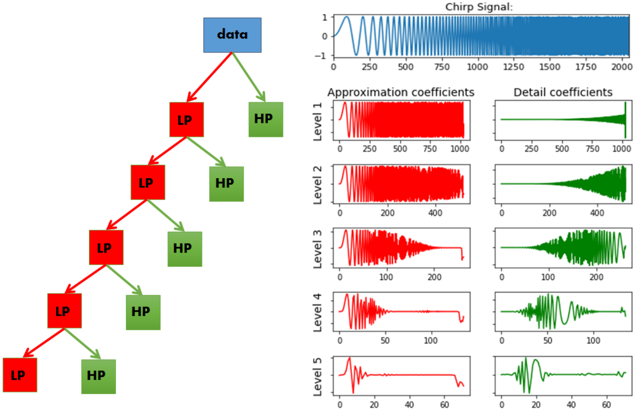
Discrete Wavelet Transform
Filter bank (example)

- This is the DWT of a chirp signal (function with a dynamic frequency spectrum, the frequency spectrum increases with time)



Discrete Wavelet Transform
Filter bank (example)

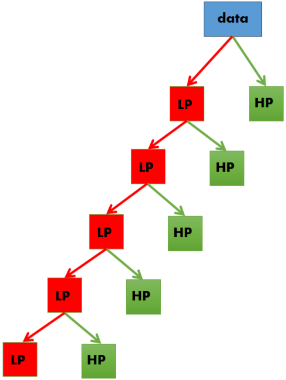
- This is the DWT of a chirp signal (function with a dynamic frequency spectrum, the frequency spectrum increases with time)
- The DWT returns two sets of coefficients: the approximation coefficients and detail coefficients

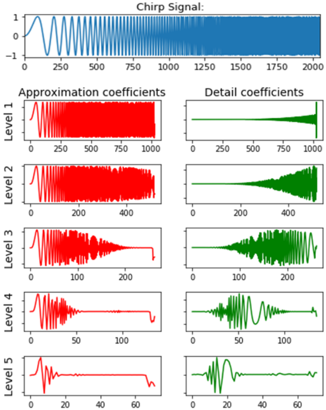


Discrete Wavelet Transform

Filter bank (example)

- This is the DWT of a chirp signal (function with a dynamic frequency spectrum, the frequency spectrum increases with time)
- The DWT returns two sets of coefficients: the approximation coefficients and detail coefficients
- The approximation coefficients represent the output of the low-pass filter (averaging filter) of the DWT

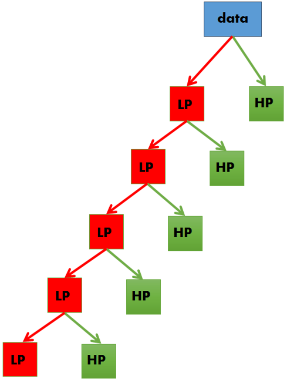


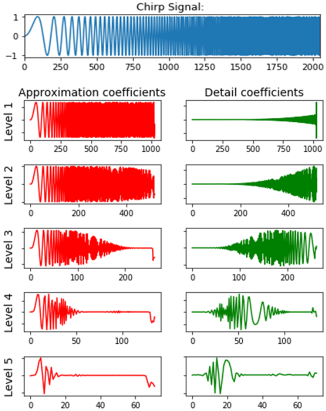


Discrete Wavelet Transform

Filter bank (example)

- The detail coefficients represent the output of the high-pass filter (difference filter) of the DWT

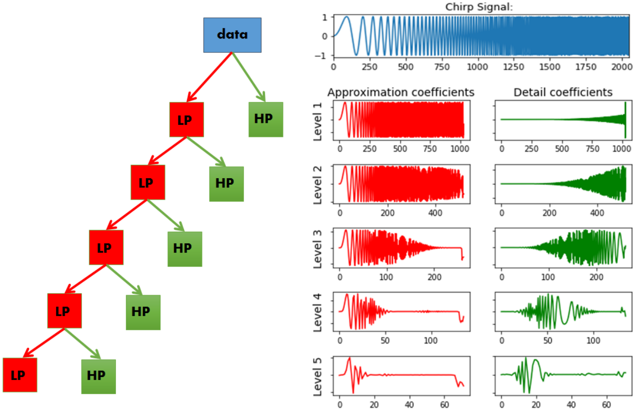




Discrete Wavelet Transform

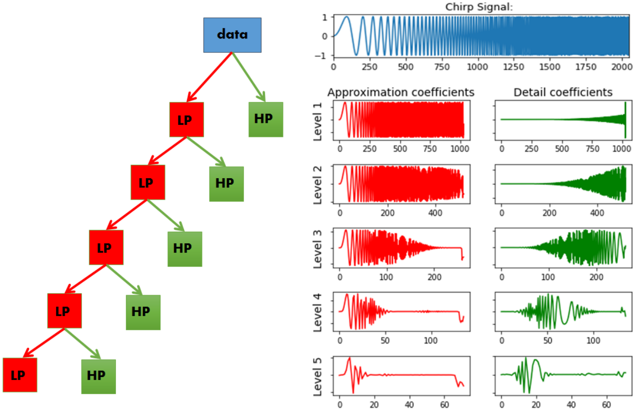
Filter bank (example)

- The detail coefficients represent the output of the high-pass filter (difference filter) of the DWT
- By applying the DWT again on the approximation coefficients of the previous DWT, we get the wavelet transform of the next level



Discrete Wavelet Transform
Filter bank (example)

- The detail coefficients represent the output of the high-pass filter (difference filter) of the DWT
- By applying the DWT again on the approximation coefficients of the previous DWT, we get the wavelet transform of the next level
- At each next level, the original signal is also sampled down by a factor of 2



Discrete Wavelet Transform
Filter bank (example)

- Now we know how the DWT is implemented as a filter bank

Discrete Wavelet Transform

Filter bank (example)

- Now we know how the DWT is implemented as a filter bank
- At each subsequent level, the approximation coefficients are divided into a coarser low-pass and high-pass part and the DWT is applied again on the low-pass part

Discrete Wavelet Transform

Filter bank (example)

- Now we know how the DWT is implemented as a filter bank
- At each subsequent level, the approximation coefficients are divided into a coarser low-pass and high-pass part and the DWT is applied again on the low-pass part
- At the end of this process, our original signal is converted to several signals, each corresponding to different frequency bands

Discrete Wavelet Transform
Filter bank (example)

- Now we know how the DWT is implemented as a filter bank
- At each subsequent level, the approximation coefficients are divided into a coarser low-pass and high-pass part and the DWT is applied again on the low-pass part
- At the end of this process, our original signal is converted to several signals, each corresponding to different frequency bands
- This idea of analyzing the signal on different scales is also known as multiresolution or multiscale analysis, and decomposing your signal in such a way is also known as multiresolution decomposition or sub-band coding

DWT in Images

Multilevel wavelet decomposition in images

- Multilevel (or multiresolution) decomposition in images is very similar to what we explained in 1D signals

DWT in Images

Multilevel wavelet decomposition in images

- Multilevel (or multiresolution) decomposition in images is very similar to what we explained in 1D signals
- However, we compute the high-pass filters in three directions: horizontal, vertical, and diagonal

DWT in Images

Multilevel wavelet decomposition in images

- Multilevel (or multiresolution) decomposition in images is very similar to what we explained in 1D signals
- However, we compute the high-pass filters in three directions: horizontal, vertical, and diagonal
- For the Haar wavelets is straightforward:

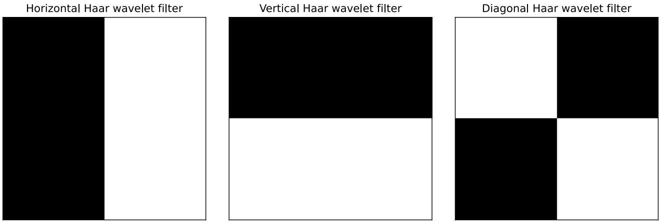
$$h = \begin{bmatrix} -0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix}, v = \begin{bmatrix} -0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix}, d = \begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \qquad (12)$$

DWT in Images

Multilevel wavelet decomposition in images

- Multilevel (or multiresolution) decomposition in images is very similar to what we explained in 1D signals
- However, we compute the high-pass filters in three directions: horizontal, vertical, and diagonal
- For the Haar wavelets is straightforward:

$$h = \begin{bmatrix} -0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix}, v = \begin{bmatrix} -0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix}, d = \begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \quad (12)$$

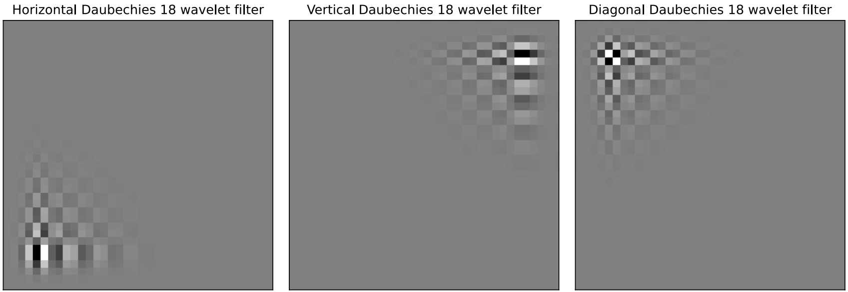


Multilevel wavelet decomposition in images

- We can have more complex wavelets

Multilevel wavelet decomposition in images

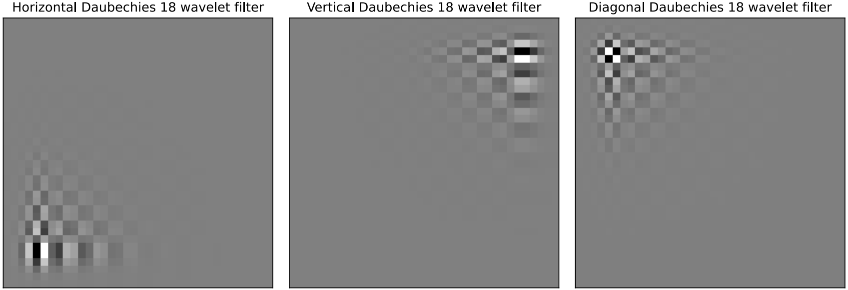
- We can have more complex wavelets



DWT in Images

Multilevel wavelet decomposition in images

- We can have more complex wavelets
- This is beyond the objectives of this lecture, but these filters are generated with the Daubechies wavelet with 18 vanishing moments



DWT in Images

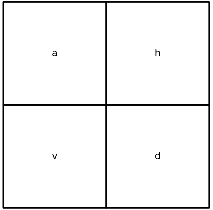
Multilevel wavelet decomposition in images

Image

DWT in Images

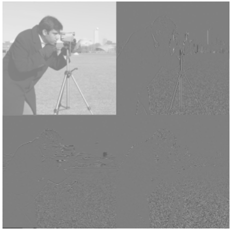
Multilevel wavelet decomposition in images

1 level
decomposition



Image

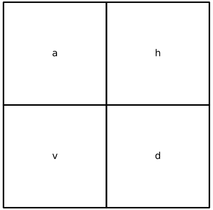
Coefficients
(1 level)



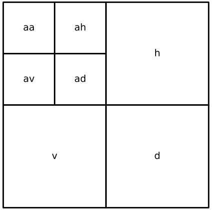
DWT in Images

Multilevel wavelet decomposition in images

1 level
decomposition

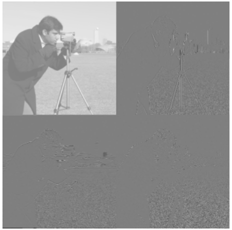


2 level
decomposition

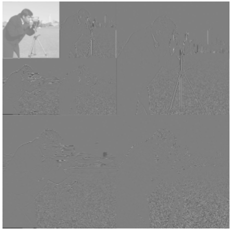


Image

Coefficients
(1 level)



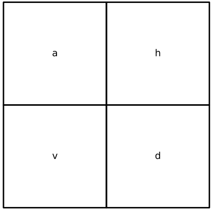
Coefficients
(2 level)



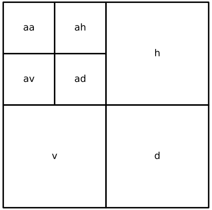
DWT in Images

Multilevel wavelet decomposition in images

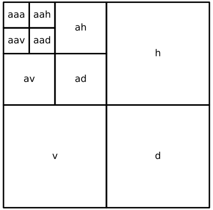
1 level
decomposition



2 level
decomposition

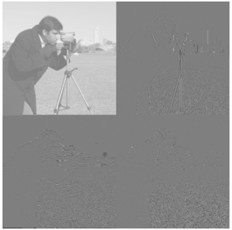


3 level
decomposition

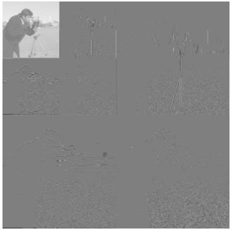


Image

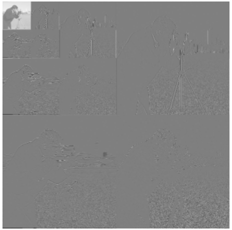
Coefficients
(1 level)



Coefficients
(2 level)



Coefficients
(3 level)



Sparsity

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness

What's sparsity?

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, where $N = 2'000,000$, where 95% of its entries are zero

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $N = 2'000,000$, where 95% of its entries are zero
- If we try to create a matrix of this dimension in Python, with float64 (64 bits) precision, what will be the amount of memory RAM required to store that matrix?

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $N = 2'000,000$, where 95% of its entries are zero
- If we try to create a matrix of this dimension in Python, with float64 (64 bits) precision, what will be the amount of memory RAM required to store that matrix?
- $N^2 \times 8$ bytes $= 32$ TB of memory RAM!!!

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $N = 2'000,000$, where 95% of its entries are zero
- If we try to create a matrix of this dimension in Python, with float64 (64 bits) precision, what will be the amount of memory RAM required to store that matrix?
- $N^2 \times 8$ bytes $= 32$ TB of memory RAM!!!
- This is such a waste of memory since we only need to store $N^2 \times 0.05$ numbers, so 1.6 GB of memory RAM

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $N = 2'000,000$, where 95% of its entries are zero
- If we try to create a matrix of this dimension in Python, with float64 (64 bits) precision, what will be the amount of memory RAM required to store that matrix?
- $N^2 \times 8$ bytes $= 32$ TB of memory RAM!!!
- This is such a waste of memory since we only need to store $N^2 \times 0.05$ numbers, so 1.6 GB of memory RAM
- In practice we also need to save the position of these numbers, so the real amount of memory occupied is slightly higher

- Sparsity is the property of being scanty or scattered; in other words, lacking denseness
- Imagine that we have a huge matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $N = 2'000,000$, where 95% of its entries are zero
- If we try to create a matrix of this dimension in Python, with float64 (64 bits) precision, what will be the amount of memory RAM required to store that matrix?
- $N^2 \times 8$ bytes $= 32$ TB of memory RAM!!!
- This is such a waste of memory since we only need to store $N^2 \times 0.05$ numbers, so 1.6 GB of memory RAM
- In practice we also need to save the position of these numbers, so the real amount of memory occupied is slightly higher
- Why this simple observation is important?

- The intuition behind image compression using wavelets is that, when we apply the DWT to an image, the output of the high-pass filter is usually sparse in the sense that we have a lot of wavelet coefficients that are zero or very close to zero

Intuition of image compression using wavelets

- The intuition behind image compression using wavelets is that, when we apply the DWT to an image, the output of the high-pass filter is usually sparse in the sense that we have a lot of wavelet coefficients that are zero or very close to zero
- This is a consequence of the smoothness that we usually encounter in images (we have regions where the intensities are almost constant)

Intuition of image compression using wavelets

- The intuition behind image compression using wavelets is that, when we apply the DWT to an image, the output of the high-pass filter is usually sparse in the sense that we have a lot of wavelet coefficients that are zero or very close to zero
- This is a consequence of the smoothness that we usually encounter in images (we have regions where the intensities are almost constant)
- Let's build first the intuitions, and then we'll move into the algorithm

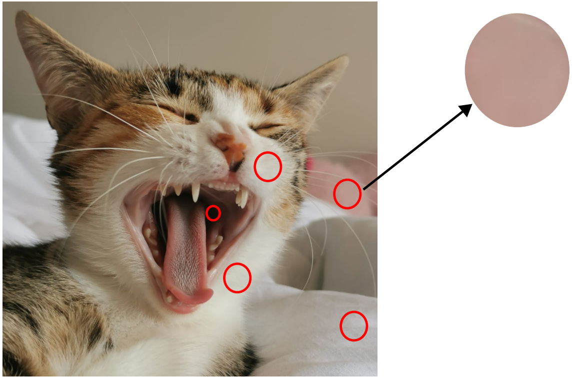
Sparsity

Intuition of image compression using wavelets

- This is my cat (Trudy), and this image is a good example to explain compression with wavelets

Sparsity

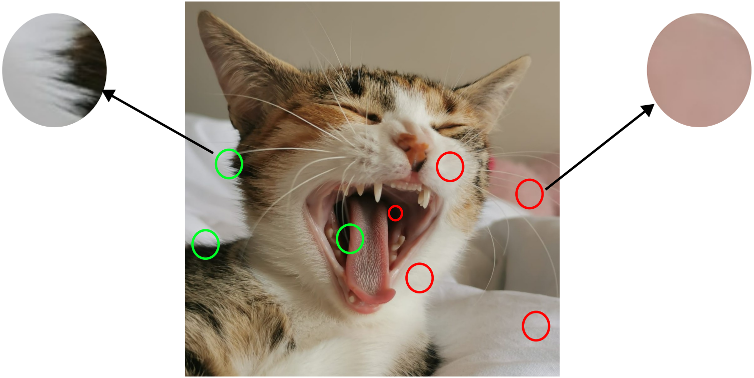
Intuition of image compression using wavelets



- This is my cat (Trudy), and this image is a good example to explain compression with wavelets
- We have plenty of regions where colors are uniform

Sparsity

Intuition of image compression using wavelets



- This is my cat (Trudy), and this image is a good example to explain compression with wavelets
- We have plenty of regions where colors are uniform
- There are only some specific parts where we need to save information

Sparsity

Intuition of image compression using wavelets

- These regions where we need to save information are the high frequencies of the image

Sparsity

Intuition of image compression using wavelets

- These regions where we need to save information are the high frequencies of the image
- If we find a mother wavelet such that the DWT of Trudy's image becomes sparse, we'll have a good compression algorithm

Intuition of image compression using wavelets

- These regions where we need to save information are the high frequencies of the image
- If we find a mother wavelet such that the DWT of Trudy's image becomes sparse, we'll have a good compression algorithm
- As we saw in the section of DWT, we need to compute low-pass (bottom left image) and high-pass filters (the other three images)

Wavelet coefficients

Sparsity

Intuition of image compression using wavelets

- These regions where we need to save information are the high frequencies of the image
- If we find a mother wavelet such that the DWT of Trudy's image becomes sparse, we'll have a good compression algorithm
- As we saw in the section of DWT, we need to compute low-pass (bottom left image) and high-pass filters (the other three images)
- In the image I already applied a threshold for visualization purposes. In the TP you'll find that when you apply the DWT, you get a lot of wavelet coefficients very close to zero

Wavelet coefficients

Sparsity

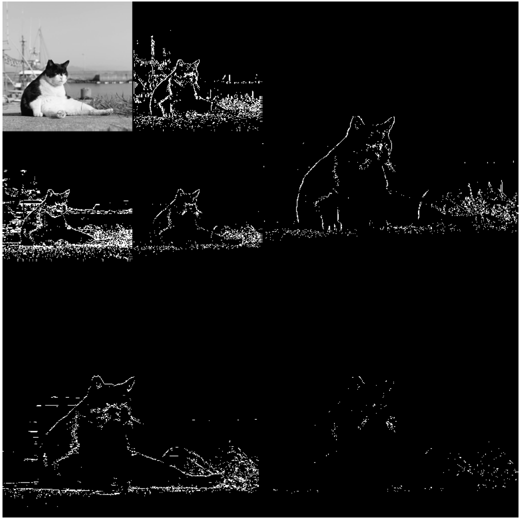
How to compute the 3-level DWT of this image?

Sparsity

How to compute the 3-level DWT of this image?

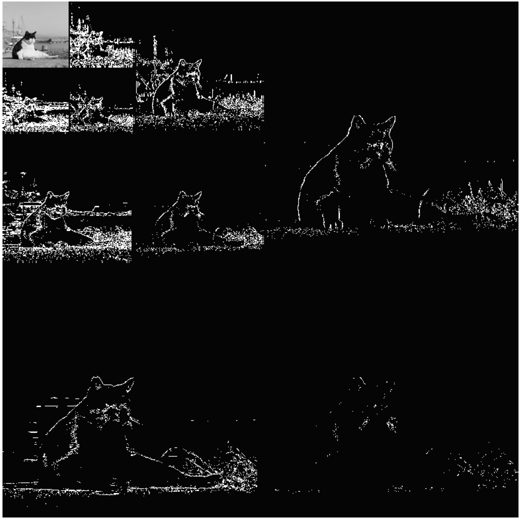
Sparsity

How to compute the 3-level DWT of this image?



Sparsity

How to compute the 3-level DWT of this image?



A very simple algorithm for compression

1. Compute the DWT of the image with a given amount of levels

A very simple algorithm for compression

1. Compute the DWT of the image with a given amount of levels
2. Choose a threshold number τ and set to zero all wavelet coefficients $|coeffs| \leq \tau$ (this doesn't give you control of the exact compression rate)

1. Compute the DWT of the image with a given amount of levels
2. Choose a threshold number τ and set to zero all wavelet coefficients $|coeffs| \leq \tau$ (this doesn't give you control of the exact compression rate)
3. Reconstruct the image applying the inverse DWT with the non-zero wavelet coefficients

- So now you know that some signals, like many kinds of images, are sparse in the wavelet domain

Sparsity
Denoising

- So now you know that some signals, like many kinds of images, are sparse in the wavelet domain
- Who can tell me one naive algorithm to do denoising with wavelets? (1 extra point for the TP)

Sparsity

Denoising

- So now you know that some signals, like many kinds of images, are sparse in the wavelet domain
- Who can tell me one naive algorithm to do denoising with wavelets? (1 extra point for the TP)
- Well, if our prior assumption happens to be true, you just need to put a threshold in the wavelet domain to recover the true signal and apply the inverse DWT

- So now you know that some signals, like many kinds of images, are sparse in the wavelet domain
- Who can tell me one naive algorithm to do denoising with wavelets? (1 extra point for the TP)
- Well, if our prior assumption happens to be true, you just need to put a threshold in the wavelet domain to recover the true signal and apply the inverse DWT
- This idea was introduce in 1994 (Donoho and Johnstone '94) in the wavelet shrinkage algorithm (hard-thresholding rule, keep or kill):

$$\hat{\theta}[i,j] = \begin{cases} Wf[i,j] & \text{if } |Wf[i,j]| > \tau \\ 0 & \text{if } |Wf[i,j]| \leq \tau, \end{cases} \tag{13}$$

where $\hat{\theta}[i,j]$ is the new wavelet coefficients and τ is the threshold

Sparsity

Denoising

Noisy
PSNR=3.146

Wavelet denoising
(VisuShrink, $\sigma = \sigma_{est}$)
PSNR=3.146

Wavelet denoising
(VisuShrink, $\sigma = \sigma_{est}/4$)
PSNR=3.146

Original

Wavelet denoising
(VisuShrink, $\sigma = \sigma_{est}/2$)
PSNR=3.146

Sparsity
Compressed sensing (motivation)

Raw: 900 KB

JPEG: 23.9 KB

Sparsity
Compressed sensing (motivation)

Raw: 900 KB

JPEG: 23.9 KB

- Data acquisition seems enourmously wasteful

Compressed sensing (motivation)

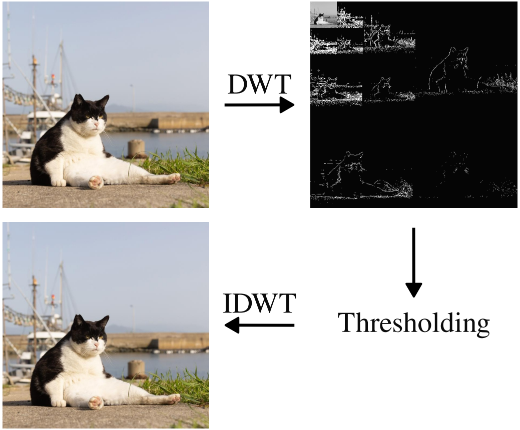
Raw: 900 KB

JPEG: 23.9 KB

- Data acquisition seems enourmously wasteful
- "*One can regard the possibility of digital compression as a failure of sensor design. If it's possible to compress measured data, one might argue that too many measurements were taken*" David Brady.

Sparsity
Compressed sensing

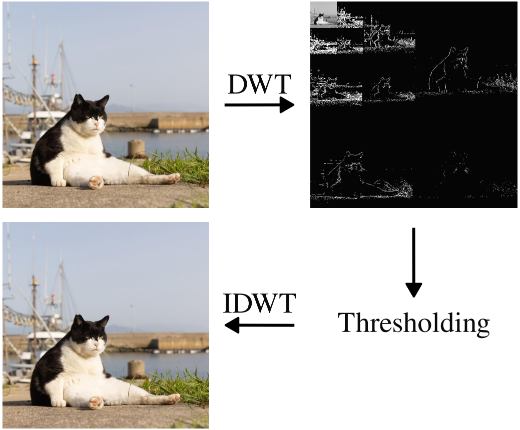
- Conventional approach: take all measurements and "keep" k largest



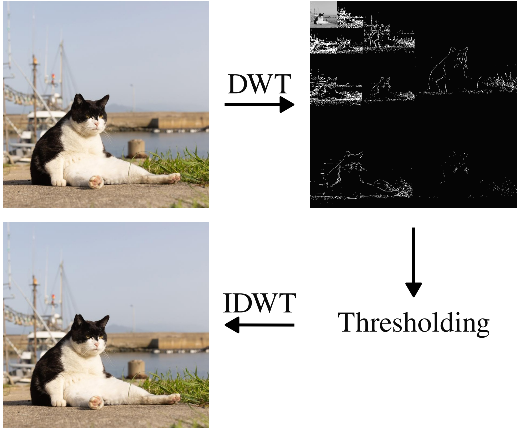
Sparsity

Compressed sensing

- Conventional approach: take all measurements and “keep” k largest
- Can’t we just measure the important pixels?

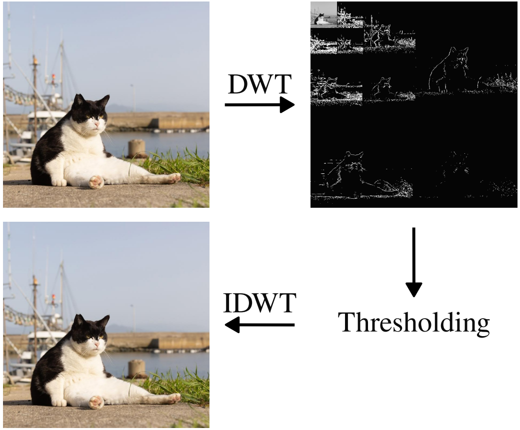


- Conventional approach: take all measurements and “keep” k largest
- Can’t we just measure the important pixels?
- To measure the important pixels we should know ahead of time where the edges are gonna be



Sparsity
Compressed sensing

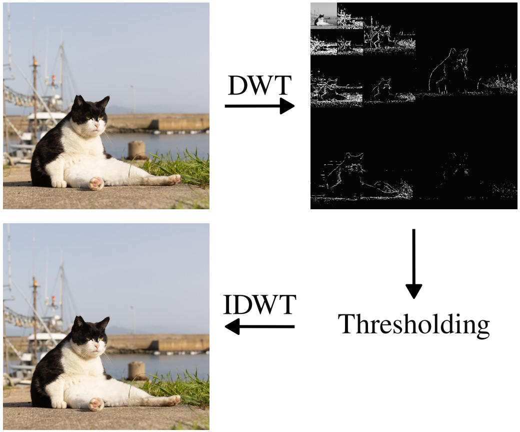
- Conventional approach: take all measurements and "keep" k largest
- Can't we just measure the important pixels?
- To measure the important pixels we should know ahead of time where the edges are gonna be
- In 2006 Candès *et al.* they said that we cannot measure the important bits (pixels) but we can exploit sparsity in the sensing mechanism acquisition itself



Sparsity

Compressed sensing

- Conventional approach: take all measurements and “keep” k largest
- Can’t we just measure the important pixels?
- To measure the important pixels we should know ahead of time where the edges are gonna be
- In 2006 Candès *et al.* they said that we cannot measure the important bits (pixels) but we can exploit sparsity in the sensing mechanism acquisition itself
- So we can take few measurements, assume sparsity, and reconstruct the original signal without much loose of information



- Why we should care about compressed sensing? Why is important?

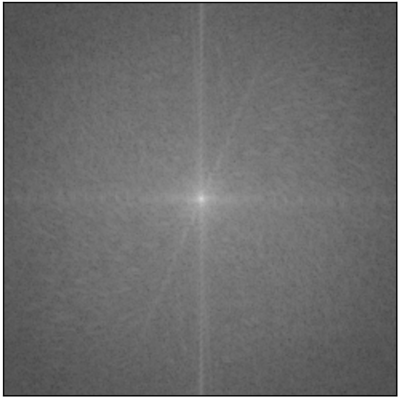
Sparsity
Compressed sensing

- Why we should care about compressed sensing? Why is important?
- One of the most important applications of compressed sensing is in MRI images

- Why we should care about compressed sensing? Why is important?
- One of the most important applications of compressed sensing is in MRI images
- In MRI what the sensing mechanism collect is indeed the Fourier transform of the image (this is given by the physics of the problem)

Sparsity

How do we form an MRI image?



IDFT $\longrightarrow$

Sparsity

Examples of MRI images

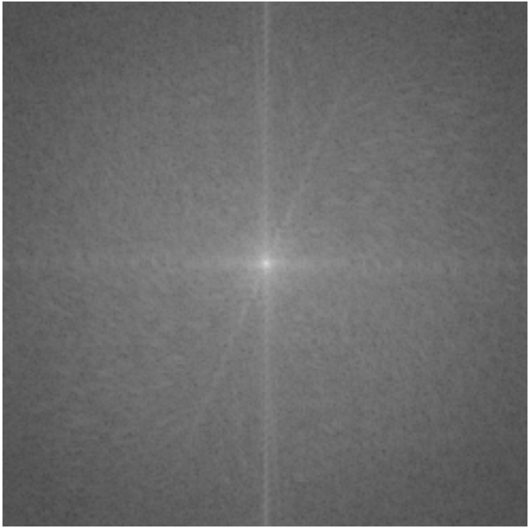
Knee

Abdominal blood vessels

Source: K. Pauly, G. Gold, RAD220

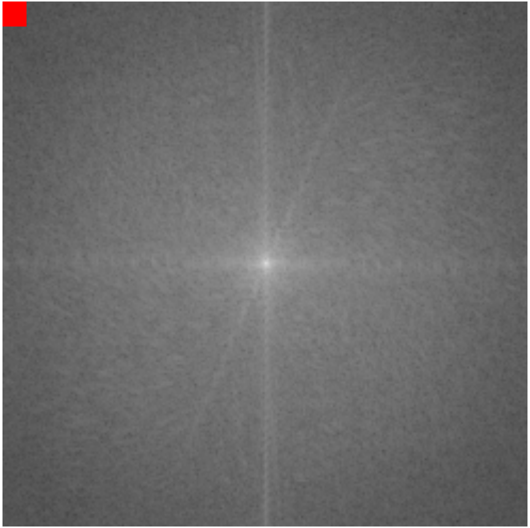
Sparsity

MRI data collection is inherently slow



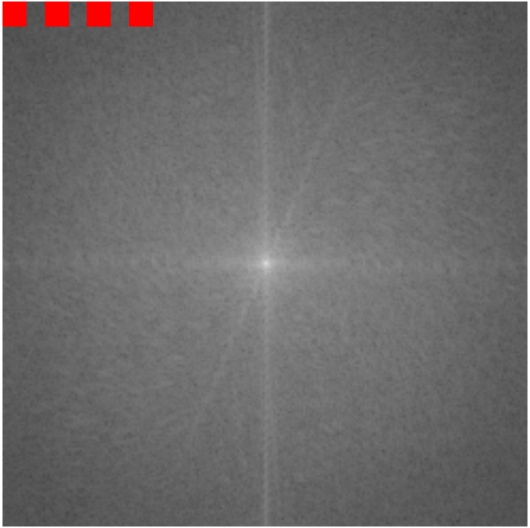
Sparsity

MRI data collection is inherently slow



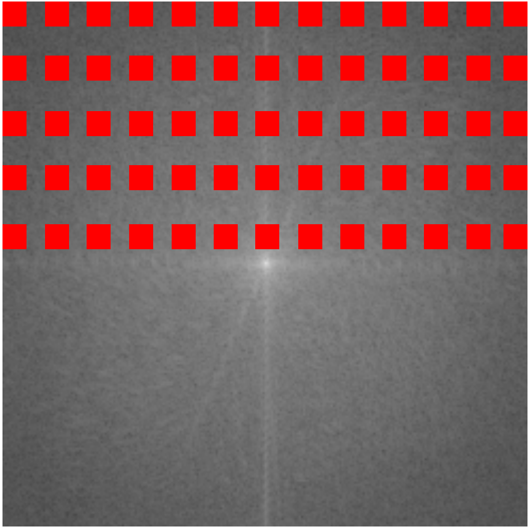
Sparsity

MRI data collection is inherently slow



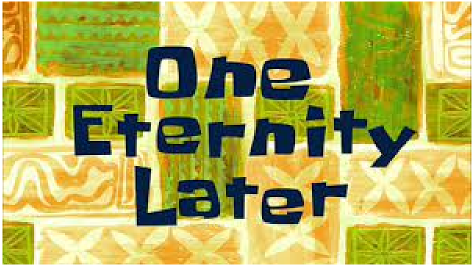
Sparsity

MRI data collection is inherently slow



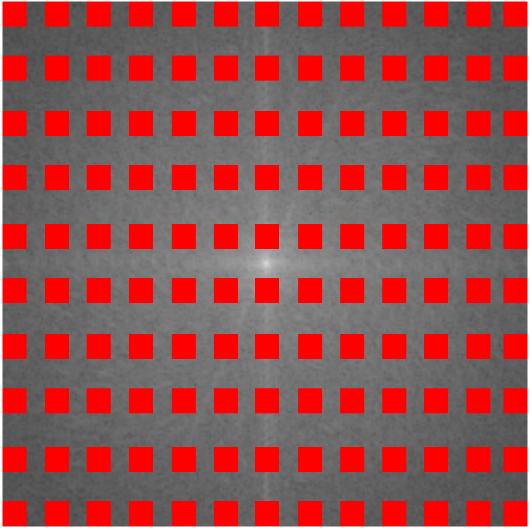
Sparsity

MRI data collection is inherently slow



Sparsity

MRI data collection is inherently slow



Sparsity

Fact: impact of MRI on children health is limited

- Children cannot stay still or breathhold!

Sparsity

Fact: impact of MRI on children health is limited

- Children cannot stay still or breathhold!
- Deep anesthesia required

Sparsity

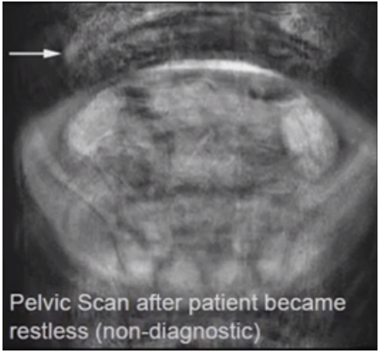
Fact: impact of MRI on children health is limited

- Children cannot stay still or breathhold!
- Deep anesthesia required
- Respiration suspension

Sparsity

Fact: impact of MRI on children health is limited

- Children cannot stay still or breathhold!
- Deep anesthesia required
- Respiration suspension



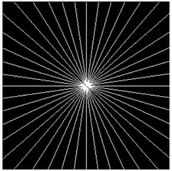
Source: Emmanuel Candès: Wavelets, sparsity and its consequences

Sparsity

A surprising experiment

A surprising experiment

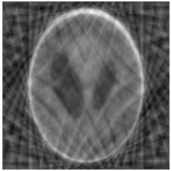
Fourier transform



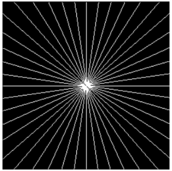
Highly subsampled

A surprising experiment

Classsical reconstruction



Fourier transform

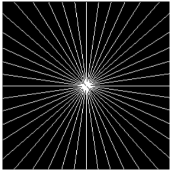


Highly subsampled

Sparsity

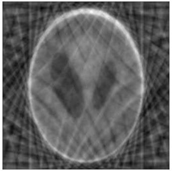
A surprising experiment

Fourier transform



Highly subsampled

Classsical reconstruction



Compressive sensing reconstruction

Source: C. Romberg and Tao 2004.

PyWavelets and TP

- To solve one part of the TP we're using the open-source library PyWavelets of Python (`https://pywavelets.readthedocs.io`)

PyWavelets and TP

- To solve one part of the TP we're using the open-source library PyWavelets of Python (`https://pywavelets.readthedocs.io`)
- It seems like PyWavelets is regularly updated with regular commits every week

PyWavelets and TP

- To solve one part of the TP we're using the open-source library PyWavelets of Python (`https://pywavelets.readthedocs.io`)
- It seems like PyWavelets is regularly updated with regular commits every week
- PyWavelets has several features, but the most important for us so far are:

PyWavelets and TP

- To solve one part of the TP we're using the open-source library PyWavelets of Python (`https://pywavelets.readthedocs.io`)
- It seems like PyWavelets is regularly updated with regular commits every week
- PyWavelets has several features, but the most important for us so far are:
  1. 1D, 2D, and nD Forward and Inverse Discrete Wavelet Transform (DWT and IDWT)
  2. 1D, 2D, and nD Multilevel DWT and IDWT
  3. 1D Continuous Wavelet Transform
  4. Over 100 built-in wavelet filters and support for custom wavelets

PyWavelets and TP

- One key function for the TP is wavedec2, which computes the multilevel 2D DWT

PyWavelets and TP

- One key function for the TP is wavedec2, which computes the multilevel 2D DWT
- The following code takes as input the “Trudy.jpeg” image and computes the DWT with 6 levels with a wavelet Daubechies 1

```
import numpy as np
import pywt
import cv2

# Load your image
input_image = cv2.imread('Trudy.jpeg', cv2.IMREAD_GRAYSCALE)

# Set the number of wavelet decomposition levels
num_levels = 6

# Define the wavelet transform method (e.g., 'haar', 'db1', 'bior1.3', etc.)
wavelet = 'db1'

# Perform the multi-level 2D wavelet decomposition
coeffs = pywt.wavedec2(input_image, wavelet, level=num_levels)
```

PyWavelets and TP

- One key function for the TP is wavedec2, which computes the multilevel 2D DWT
- The following code takes as input the “Trudy.jpeg” image and computes the DWT with 6 levels with a wavelet Daubechies 1
- I’ll let you explore in the TP the meaning of coeffs from this piece of code

```python
import numpy as np
import pywt
import cv2

# Load your image
input_image = cv2.imread('Trudy.jpeg', cv2.IMREAD_GRAYSCALE)

# Set the number of wavelet decomposition levels
num_levels = 6

# Define the wavelet transform method (e.g., 'haar', 'db1', 'bior1.3', etc.)
wavelet = 'db1'

# Perform the multi-level 2D wavelet decomposition
coeffs = pywt.wavedec2(input_image, wavelet, level=num_levels)
```

Usage of ChatGPT in the TP

- At Télécom we still don't have a clear policy regarding ChatGPT usage for homework

Usage of ChatGPT in the TP

- At Télécom we still don't have a clear policy regarding ChatGPT usage for homework
- In my view, it's OK to use it as long as you're **fully responsible for ChatGPT's outputs**. In practice, this means that you should held accountable for the hallucinations or violation of copyright if you use ChatGPT without being careful

- At Télécom we still don't have a clear policy regarding ChatGPT usage for homework
- In my view, it's OK to use it as long as you're **fully responsible for ChatGPT's outputs**. In practice, this means that you should held accountable for the hallucinations or violation of copyright if you use ChatGPT without being careful
- For our TP next week, I already asked ChatGPT about the solution, and it delivered a code that was wrong (at least in my case). If you use ChatGPT without reasoning about its response, you'll probably get a really bad score

TELECOM Paris
IP PARIS

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning
- However, to understand these applications you need to know concepts that you'll follow later this year

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning
- However, to understand these applications you need to know concepts that you'll follow later this year
- If you want to explore these topics in the future, I'm mentioning here some areas that I found personally interesting:

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning
- However, to understand these applications you need to know concepts that you'll follow later this year
- If you want to explore these topics in the future, I'm mentioning here some areas that I found personally interesting:
    1. Use of scalogram to represent audio signals that are later input to [put your favorite deep learning method here] to solve some task

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning
- However, to understand these applications you need to know concepts that you'll follow later this year
- If you want to explore these topics in the future, I'm mentioning here some areas that I found personally interesting:
  1. Use of scalogram to represent audio signals that are later input to [put your favorite deep learning method here] to solve some task
  2. Representation of EEG signals

Some remarks

- There are some interesting applications of wavelets to some problems in machine learning
- However, to understand these applications you need to know concepts that you'll follow later this year
- If you want to explore these topics in the future, I'm mentioning here some areas that I found personally interesting:
    1. Use of scalogram to represent audio signals that are later input to [put your favorite deep learning method here] to solve some task
    2. Representation of EEG signals
    3. Wavelets on graphs and graph wavelet networks

Some remarks

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images

Some remarks

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images
- The most direct and obvious application of this sparsity is compression as we saw in the lecture

Some remarks

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images
- The most direct and obvious application of this sparsity is compression as we saw in the lecture
- However, we also have compressed sensing

Some remarks

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images
- The most direct and obvious application of this sparsity is compression as we saw in the lecture
- However, we also have compressed sensing
- The idea behind compressed sensing is that, if we can have high compression ratios using wavelets, perhaps we don't need to capture the whole signal (image) in the first place

Some remarks

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images
- The most direct and obvious application of this sparsity is compression as we saw in the lecture
- However, we also have compressed sensing
- The idea behind compressed sensing is that, if we can have high compression ratios using wavelets, perhaps we don't need to capture the whole signal (image) in the first place
- The problem here, besides the reconstruction algorithm, is to know where and how we need to sample. This is still an active topic of research

- One of the most important consequences of wavelets is the intrinsic sparsity of some signals like images
- The most direct and obvious application of this sparsity is compression as we saw in the lecture
- However, we also have compressed sensing
- The idea behind compressed sensing is that, if we can have high compression ratios using wavelets, perhaps we don't need to capture the whole signal (image) in the first place
- The problem here, besides the reconstruction algorithm, is to know where and how we need to sample. This is still an active topic of research
- Compressed sensing is an active and interesting topic of research, not as famous as LLMs, but with interesting and challenging open questions

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals
- Wavelets are localized in time

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals
- Wavelets are localized in time
- In practice, we implement the DWT as a filter bank

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals
- Wavelets are localized in time
- In practice, we implement the DWT as a filter bank
- Wavelets are useful for image compression due to the nature of the images (piece-wise smoothness)

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals
- Wavelets are localized in time
- In practice, we implement the DWT as a filter bank
- Wavelets are useful for image compression due to the nature of the images (piece-wise smoothness)
- Wavelets have had a tremendous impact on signal processing

Main takeaways

- Wavelet analysis tries to find a better compromise in the trade-off between frequency and time resolution than the Fourier transform when studying 1D signals
- Wavelets are localized in time
- In practice, we implement the DWT as a filter bank
- Wavelets are useful for image compression due to the nature of the images (piece-wise smoothness)
- Wavelets have had a tremendous impact on signal processing
- The consequences of wavelets go beyond compression, and they are actively studied today (compressed sensing)

Thank you!

Jhony H. Giraldo: jhony.giraldo@telecom-paris.fr
(https://sites.google.com/view/jhonygiraldo)