Final Project

Transformations of Functions and Signals

Course: Fundamentals of Numerical Simulations

Name: Mina Shiri

Student-ID: 40011023

Table of Contents

1. Theoretical Background

1.1 Fourier Transformation

The Fourier transformation F of the function f on the real axis is defined as

$$F(\omega) = \mathcal{F}[f](\omega) = \int_{-\infty}^{\infty} f(x)\, \mathrm{e}^{-\mathrm{i}\omega x}\, \mathrm{d}x.$$

The sufficient conditions for the existence of F are that f is absolutely integrable, i.e. $\int_{-\infty}^{\infty} |f(x)|\, \mathrm{d}x < \infty$, and that f is piece-wise continuous or has a finite number of discontinuities. The inverse transformation is

$$f(x) = \mathcal{F}^{-1}[F](x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega)\, \mathrm{e}^{\mathrm{i}\omega x}\, \mathrm{d}\omega.$$

Changing the sign of the argument x of f (flipping the space or time coordinate) implies a change of the sign of the frequency ω in the transform:

$$g(x) = f(-x) \qquad \Longleftrightarrow \qquad G(\omega) = F(-\omega).$$

1.1.1 Fourier Theorem

If the function f is piece-wise continusous on the interval

$[-\pi,\pi]$, one can form the Fourier series

$$\frac{1}{2\pi} \sum_{n\in\mathbb{Z}} \int_{-\pi}^{\pi} f(x)\, \mathrm{e}^{\mathrm{i}n(\xi - x)}\, \mathrm{d}x = \frac{1}{2}\big[f(\xi + 0) + f(\xi - 0)\big], \quad \xi \in [-\pi, \pi]$$

.

It follows from this theorem that the function f which is continuous on R and its Fourier transform F for any a ∈ R, a > 0, are related by the *Poisson sum*

$$\sum_{n\in\mathbb{Z}} F(n\,a) = \frac{2\pi}{a} \sum_{m\in\mathbb{Z}} f\left(\frac{2\pi m}{a}\right)$$

1.1.2 Parseval's Equality

The integral of the square of the function f over x and the integral of the square of its transform F over ω are related by the Parseval's equality:

$$\int_{-\infty}^{\infty} |f(x)|^2 \, dx = \frac{1}{2\pi} \int_{-\infty}^{\infty} |F(\omega)|^2 \, d\omega.$$

This invariance of the norm means that power is preserved in the transition from the temporal to the frequency representation of a signal. Parseval's equality is valid for all function expansions in Hilbert spaces.

1.1.3 Fourier Uncertainty

Let f be normalized as $\int_{-\infty}^{\infty} |f(x)|^2 \, dx = 1$,Then $|f|^2$ is non-negative and can be understood as a probability distribution of the signal with respect to x, with the first and second moments (average and variation)

$$\langle x \rangle = \int_{-\infty}^{\infty} x |f(x)|^2 \, dx, \qquad \sigma_x^2 = \int_{-\infty}^{\infty} \left[x - \langle x \rangle\right]^2 |f(x)|^2 \, dx.$$

the Fourier transform of f is also normalized, $\int_{-\infty}^{\infty} |F(\omega)|^2 \, d\omega = 2\pi$.

Hence $|F(\omega)|^2/(2\pi)$ is the power distribution density with respect to frequencies, with the first and second moments

$$\omega\rangle = \frac{1}{2\pi} \int_{-\infty}^{\infty} \omega |F(\omega)|^2 \, d\omega, \qquad \sigma_\omega^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left[\omega - \langle \omega \rangle\right]^2 |F(\omega)|^2 \, d\omega.$$

The product of the variations in both representations of f is

$$\sigma_x^2 \sigma_\omega^2 \geq \frac{1}{4}$$

Equation above expresses the *Fourier uncertainty*: spatial or temporal variations of the signal are not independent of frequency variations. If variations decrease in the configuration space, they increase in the transform space, and vice versa. The lower bound of the uncertainty is achieved by Gaussian signals,

$$f(x) = \left(\frac{2s}{\pi}\right)^{1/4} e^{-s(x-a)^2}, \quad s > 0, \; a \in \mathbb{R}$$

We say that such signals have a *minimum possible width*.

1.1.4 Sampling Theorem

Assume that the function (signal) f is continuous on the whole real axis and that. The signal is uniformly sampled at the points spaced x apart, like

$f_n = f(n\,\Delta x)$ for $n \in \mathbb{Z}$. If the range of the frequencies corresponding to the signal f is bounded, i.e. if

$$F(\omega) = 0 \quad \forall |\omega| \geq \omega_c = \frac{\pi}{\Delta x},$$

where ω_c is the Nyquist (critical) frequency, the complete signal f can be reconstructed *without loss of information* from the discrete sample $\{f_n : n \in \mathbb{Z}\}$ by using the formula

$$f(x) = \sum_{n\in\mathbb{Z}} f_n \frac{\sin \omega_c (x - n\,\Delta x)}{\omega_c (x - n\,\Delta x)}.$$

1.2 Fourier Series

1.2.1 Continuous Fourier Expansion

The Fourier transform of the function f which is bounded and piece-wise continuous on the interval $[0,2\pi]$, is

$$\widehat{f_k} = \frac{1}{2\pi} \int_0^{2\pi} f(x)\, e^{-ikx}\, dx, \quad k = 0, \pm 1, \pm 2, \ldots.$$

The functions $\varphi_k(x) = e^{ikx}$ and $\varphi_{k^*}(x) = e^{-ikx}$ are orthogonal on $[0,2\pi]$,

$$\int_0^{2\pi} \phi_j(x)\phi_k^*(x)\, dx = 2\pi\delta_{j,k}.$$

The functions φ_k form the basis of the space T_N of trigonometric polynomials of degree $\leq N/2$. The Fourier cosine and sine transforms are the real and imaginary parts of the transform:

$$a_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) \cos kx\, dx, \qquad b_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) \sin kx\, dx,$$

where all three forms are related by $\widehat{f_k} = a_k - ib_k$. In general, the functions f and their Fourier coefficients $\widehat{f_k}$ are complex. Specific symmetry properties of the signals correspond to specific properties of their transforms. The following cases are the most relevant:

$$\begin{array}{lll} \text{if} & f \text{ is real}, & \widehat{f_{-k}} = \widehat{f_k^*}; \\ & f \text{ is real and even}, & \widehat{f_k} \text{ is real and even}; \\ & f \text{ is real and odd}, & \widehat{f_k} \text{ is imaginary and odd}. \end{array}$$

If $\widehat{f_k}$ ssis real, the coefficients of its cosine and sine transform a_k and b_k are also real, and $f_{-k} = f_k^*$. The function f can be represented by the Fourier series

$$Sf(x) = \sum_{k=-\infty}^{\infty} \widehat{f_k}\phi_k(x),$$

And we have:

$$S_N f(x) = \sum_{k=-N/2}^{N/2-1} \widehat{f_k}\phi_k(x).$$

The normalization factor $(2\pi)^{-1}$ has changed its place. In both cases the inverse transformation of the transform restores the original function. Other variants are widely used, e.g. swapping the roles of $\varphi_k(x) = e^{ikx}$ and $\varphi_{k*}(x) = e^{-ikx}$: the opposite phases cause only a sign change in the imaginary components. These conventions partly originate in different interpretations of the sign of the frequencies: the Schrödinger equation $i\hbar\partial\psi/\partial t = E\psi$ forces the physicist to associate $e^{i\omega t}$ with a quantum state with *negative* energy $E = -\hbar\omega$, while an electronics engineer will see it as a signal with s frequency.

If the function f is continuous, periodic, and has a bounded variation on $[0, 2\pi]$, i.e. $\int_0^{2\pi} |f'(x)|\,dx < \infty$, the series Sf(x) uniformly converges to f(x), so $\lim_{N\to\infty} \max_x |f(x) - S_N f(x)| = 0$. If f has a bounded variation on, then $_N$f(x) (f(x + 0) + f(x − 0))/2 for each x ∈ [0,2π]is continuous and periodic, the Fourier series does not necessarily converge at each x ∈ [0,2π] . The Fourier series of f ∈ $L^2(0,2\pi)$ converges to f in the sense

$$\lim_{N\to\infty} \int_0^{2\pi} \left|f(x) - S_N f(x)\right|^2 dx = 0$$

The truncation of Sf(x) to S_Nf(x) containing a finite number of terms implies an error. Its magnitude is determined by the asymptotics of the Fourier coefficients,

$$\max_{0\le x\le 2\pi} \left|f(x) - S_N f(x)\right| \le \sum_{k<-N/2} |\widehat{f_k}| + \sum_{k>N/2-1} |\widehat{f_k}|$$

If f is m-times (m ≥ 1) continuously differentiable on [0,2π] and its jth derivative is periodic for all j ≤ m − 2, we have

$$\widehat{f_k} = \mathcal{O}\left(|k|^{-m}\right), \quad k \to \infty$$

There are several methods to estimate the truncation error. In the norm in $L^2(0,2\pi)$, $S_N f$ is the best approximation for f among all functions from T_N. The estimate

$$\| f - S_N f \|_{L^2(0,2\pi)} \leq C N^{-m} \left\| f^{(m)} \right\|_{L^2(0,2\pi)} \quad \text{س}$$

is valid for m ≥ 0 for any m-times differentiable function f .

1.2.2 Discrete Fourier Expansion

The Fourier expansion of a function f for which the values on [0,2π] are known at the points (nodes)

$$xj \; = \; 2\pi j/N, j \qquad = \; 0,1,\ldots,N \; - \; 1, N \; even,$$

is represented by the coefficients of the *discrete Fourier transform* (DFT)

$$\widetilde{f_k} = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j)\, \mathrm{e}^{-\mathrm{i}kx_j}, \quad -N/2 \leq k \leq N/2 - 1$$

Since N is even, $f_{-N/2} = f_{N/2}$, and thus an expansion with $|k| \leq N/2$ is also sensible: for the coefficient at N/2 we then use the normalization 1/(2N) instead of 1/N. The index representing the frequency ω may also run from 0 to N. The lower portion of the spectrum ($1 \leq k \leq N/2 - 1$) then corresponds to the negative frequencies, $-\omega_c < \omega < 0$, while the upper portion ($N/2 + 1 \leq k \leq N - 1$) corresponds to the positive ones, $0 < \omega < \omega_c$, where $\omega_c = N/2$. The value at k = 0 belongs to frequency zero (the average of the signal) while the value at k = N maps to both ω_c and $-\omega_c$, and can be omitted so that we have precisely N distinct coefficients.

For real functions f it holds that

$$\widetilde{f}_{N-k} = \widetilde{f}_k^*, \quad \widetilde{f}_0 \in \mathbb{R}.$$

The function value at the point x_j obtained by the inverse DFT, that is, by the sum

$$f(x_j) = \sum_{k=-N/2}^{N/2-1} \tilde{f}_k \, e^{ikx_j}, \quad j = 0, 1, \ldots, N-1$$

s

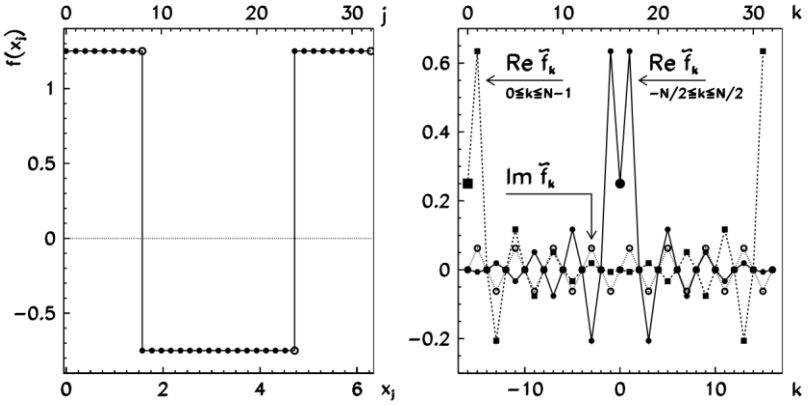


Fig. 1.1 Discrete Fourier transformation. [*Left*] The sum of the constant 0.25 and the square wave on [0,2π] is sampled at N = 32 points $x_j = 2\pi j/N$, $0 \le j \le N-1$ (the • *symbols*). At the points

the function is not sampled! [*Right*] The real and imaginary parts of the Fourier coefficients for two different ways of indexing ($-N/2 \le k \le N/2$ or $0 \le k \le N-1$). The *thick symbols* at k = 0 denote the zero-frequency component, which is the average of the signal—precisely the 0.25 vertical shift on the *left figure* (If we subtract the average of the signal, the zeroth component of its transform is zero)

Let $I_N f$ denote the interpolant of the function f at N points. The discrete Fourier series

$$I_N f(x) = \sum_{k=-N/2}^{N/2-1} \tilde{f}_k \, e^{ikx}$$

interpolates f , hence $I_N f(x_j) = f(x_j)$. The points x_j at which the value of f exactly coincides with the value of the interpolant $I_N f$ are known as the *Fourier collocation points*. The interpolant can also be written as

$$I_N f(x) = \sum_{i=0}^{N-1} f(x_j) g_j(x),$$

where

$$g_j(x) = \frac{1}{N} \sin \frac{N(x - x_j)}{2} \operatorname{ctg} \frac{x - x_j}{2}$$

are the characteristic Lagrange trigonometric polynomials with the property $g_j(x_i) = \delta_{i,j}$. The first three polynomials for N = 8 are shown in Fig. 1.2. An example of the trigono metric interpolation of a periodic function f(x) = 2/(4 − 3cosx) is shown in Fig. 1.3.

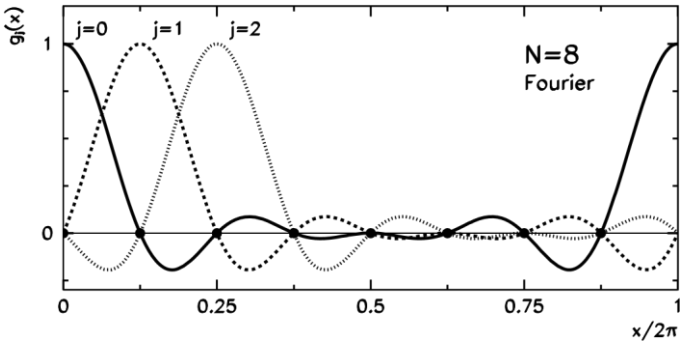


Fig. 1.2 The first three Lagrange trigonometric polynomials $g_j(x)$ in the case N = 8. The maxima are at the Fourier collocation points $x_j = 2\pi j/N$ (j = 0,1,...,N − 1)

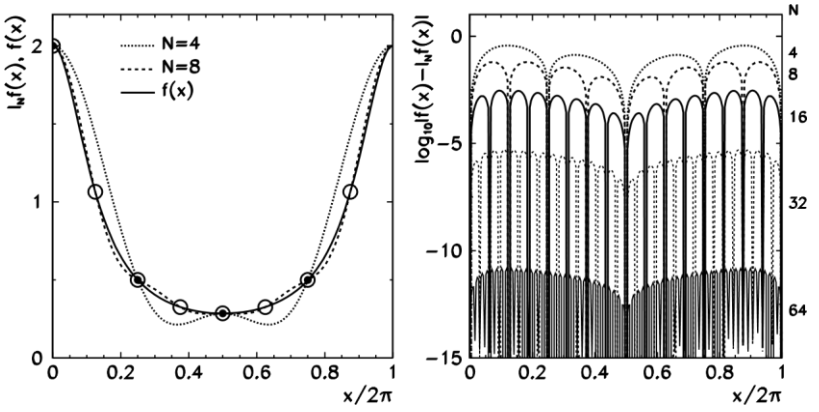


Fig. 1.3 Trigonometric interpolation of the function f(x) = 2/(4 − 3 cos x). [*Left*] The interpolants $I_4 f$ and $I_8 f$ (matching at eight points, symbols ◦). [*Right*] The error of the discrete Fourier series for = 4, 8, 16, 32, and 64. Note the exceptionally rapid drop-off of the error with the increasing number of points (*right vertical axis*): it is known as spectral convergence. The interpolation property $I_N f(x_j) = f(x_j)$ is seen in the characteristic spikes of vanishing error

The quadrature formula

$$\frac{1}{2\pi}\int_0^{2\pi} f(x)\,\mathrm{d}x = \frac{1}{N}\sum_{j=0}^{N-1} f(x_j)$$

This is exact for any function of the form e^{ikx} for $k \in Z$, $|k| < N$ or any trigonometric polynomial of degree less than N which is a linear combination of such functions. The DFT can therefore also be understood as an approximation of the continuous Fourier transform of a periodic function f on the interval $[0, 2\pi]$:

$$\frac{1}{2\pi}\int_0^{2\pi} f(x)\,\mathrm{e}^{-\mathrm{i}kx}\,\mathrm{d}x = \frac{1}{N}\sum_{j=0}^{N-1} f(x_j)\,\mathrm{e}^{-\mathrm{i}2\pi jk/N} + \mathcal{R}_N$$

Where $\mathcal{R}_N = -f''(\xi)(2\pi)^2/(12N^2)$ and $\xi \in [0, 2\pi]$

1.2.3 Aliasing

The coefficients of the discrete Fourier transform and the coefficients of the exact expansionss are related by

$$\widetilde{f}_k = \widehat{f}_k + \sum_{\substack{m=-\infty \\ m\neq 0}}^{\infty} \widehat{f}_{k+Nm}, \quad k = -N/2, \ldots, N/2-1$$

This can be written as $I_N f = S_N f + R_N f$. The remainder

$$R_N f = I_N f - S_N f = \sum_{k=-N/2}^{N/2-1} \left(\sum_{\substack{m=-\infty \\ m\neq 0}}^{\infty} \widehat{f}_{k+Nm} \right) \phi_k$$

is called the *aliasing error* and it measures the difference between the interpolation polynomial and the truncated Fourier series. Aliasing implies that the Fourier component with the wave-number (k + Nm) behaves like the component with the wave-number k. Because the basis functions are periodic,

$$\phi_{k+Nm}(x_j) = \phi_k(x_j), \quad m \neq 0,$$

such components are indistinguishable (Fig. 1.4). In other words, on a discrete mesh, the kth Fourier component of the interpolant $I_N f$ depends not only on the kth

component of f , but also on the higher components mimicking the kth. The aliasing error is orthogonal to the truncation error f − S_Nf , thus

$$\|f - I_N f\|^2 = \|f - S_N f\|^2 + \|R_N f\|^2$$

The interpolation error is therefore always larger than the truncation error.

1.2.4 Leakage

The discrete Fourier transformation of realistic signals of course involves only finitely many values. From an infinite sequence we pick (multiply by one) only a

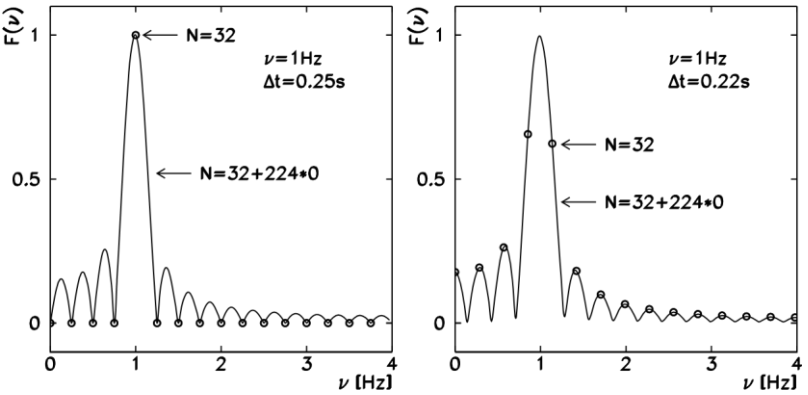


Fig. 1.6 Leakage in the frequency spectrum in the discrete Fourier transform of a sine wave with the frequency 1 Hz. [*Left*] Sampling at N = 32 points 0.25 s apart encompasses precisely four complete waves. The only non-zero component of the transform is the one corresponding to the frequency of 1 Hz. [*Right*] Sampling at N = 32 points 0.22s apart covers only 3.52 waves. Many non-zero-frequency components appear. The *curves* connect the discrete transform of the same signals, except that the 32 original samples of the signal are followed by 224 zeros (total of 256 points). Adding zeros in the temporal domain is known as *zero padding* and improves the resolution in the frequency domain. In the limit N → ∞ we approach the continuous Fourier transform

sample of length N, whereas the remaining values are dropped (multiplied by zero). Due to this restriction, known as *windowing*, the frequency spectrum exhibits the *leakage* phenomenon shown in Fig. 1.6. To some extent, leakage can be controlled by using more sophisticated *window functions* that engage a larger portion of the

signal and smoothly fade out instead of crude multiplication of the signal by one and the remainder by zero.

1.2.5 Fast Discrete Fourier Transformation (FFT)

The discrete Fourier transformation is a mapping between the vector spaces of dimensions N, $F_N : C^N \to C^N$. Let us rewrite it in a more transparent form,

$$F = \mathcal{F}_N[f], \quad F_k = \frac{1}{N}\sum_{j=0}^{N-1} f_j \, e^{-2\pi i jk/N}$$

where we have denoted $f = \{f_j\}_{j=0}^{N-1}$ and $F = \{F_k\}_{k=0}^{N-1}$. Note that the indices j and k run symmetrically, both from 0 to N − 1. The inverse transformation is

$$f = \mathcal{F}_N^{-1}[F], \quad f_j = \sum_{k=0}^{N-1} F_k \, e^{2\pi i jk/N}.$$

Then its can be written as

$$F_k = \frac{1}{N}\sum_{j=0}^{N-1} W_N^{kj} f_j, \quad W_N = e^{-2\pi i/N}.$$

To evaluate the DFT by computing this sum we need $O(N^2)$ operations. But precisely the same result can be achieved with far fewer operations by using the Cooley–Tukey algorithm. Let N be divisible by m. Then the sum can be split into m partial sums, and each of them runs over the elements f_j of the array f with the same modulus of the index j mod m:

$$F_k = \frac{1}{N}\sum_{l=0}^{m-1} W_N^{kl} \sum_{j=0}^{N/m-1} W_{N/m}^{kj} f_{mj+l}.$$

Let us denote $f^{(l)} = \{f_{mj+l}\}_{j=0}^{N/m-1}$ by the components of the array f which have the same modulus of the index with respect to m. We have thus recast the transform of the original array f of dimension N as a sum of m transforms of the shorter arrays $f^{(l)}$ of dimension N/m. This can be written symbolically as

$$\mathcal{F}_N[f]_k = \frac{1}{N}\sum_{l=0}^{m-1} W_N^{kl}\left(\frac{N}{m}\mathcal{F}_{N/m}\left[f^{(l)}\right]\right)_k.$$

This is a recursive computation of the DFT for the array f that follows the idea of *divide-and-conquer* algorithms. The array f for which the DFT should be computed is gradually broken down into sub-arrays, thus reducing the amount of necessary work. The optimal factorization is $N = 2^p$ ($p \in \mathbb{N}$) in which at each step the array is split into two sub-arrays containing elements of the original array with even and odd indices, respectively. This method requires $O(N \log_2 N)$ operations for the full DFT instead of $O(N^2)$ by direct summation, lending it the name *Fast Fourier Transformation*. Similar speeds are attainable by factorizing N to primes, e.g.

$$N = 2^{p_1} 3^{p_2} 5^{p_3} 7^{p_4}, \quad p_i \in \mathbb{N},$$

which is supported by all modern FFT libraries. Good implementations of the FFT are complicated, as the factorization should be carefully matched to the addressing of the arrays. The most famous library is the multiple-award winning FFTW (Fastest Fourier Transform in the West.

1.2.6 Multiplication of Polynomials by Using the FFT

Multiplication of polynomials is one of the tasks in computing with power bases, e.g. in expansions in powers of the perturbation parameters in classical and quantum mechanics. The multiplication of $p(x) = \sum_{i=0}^{n} a_i x^i$ and $q(x) = \sum_{i=0}^{m} b_i x^i$ in the form

$$q(x)p(x) = \sum_{i=0}^{n+m} c_i x^i, \quad c_i = \sum_{k=0}^{i} a_k b_{i-k},$$

requires $O((n + 1)(m + 1))$ operations to determine the coefficients c_i. If the number of terms is large ($n,m \gg 1$) this process is slow and prone to rounding errors. A faster and a more precise way is offered by the FFT. We form two arrays of length $N = m + n + 1$. The coefficients of the polynomials p and q are stored at the beginning of these arrays while the remaining elements are set to zero:

$$A = \{A_i\}_{i=0}^{N-1} = \{a_0, \ldots, a_n, \underbrace{0, \ldots, 0}_{m}\}, \qquad B = \{B_i\}_{i=0}^{N-1} = \{b_0, \ldots, b_m, \underbrace{0, \ldots, 0}_{n}\}$$

The coefficients of the product are given by the convolution

$$c_i = \sum_{k=0}^{N-1} A_k B_{i-k}, \quad i = 0, 1, \ldots, N-1,$$

$$A_k = A_{N+k}, B_k = B_{N+k}$$

where we assume periodic boundary conditions, .The convolution is then evaluated by first computing the FFT of the arrays B,

$$\hat{A} = \{\hat{A}_i\}_{i=0}^{N-1} = \mathcal{F}_N[A], \qquad \hat{B} = \{\hat{B}_i\}_{i=0}^{N-1} = \mathcal{F}_N[B],$$

multiplying the transforms component-wise into a new array $\hat{C} = \{\hat{A}_i \hat{B}_i\}_{i=0}^{N-1}$, and finally compute its inverse FFT,

$$C = \{C_i\}_{i=0}^{N-1} = N\mathcal{F}_N^{-1}[\hat{C}].$$

This procedure has a numerical cost of O(N $\log_2$ N) which, for n,m 1, is much smaller than the cost of directly computing the sums of the products.

1.2.7 Power Spectral Density

The Fourier transformation can be seen as a decomposition of a signal to a linear combination of the functions $A_\omega e^{i\omega x}$. The quantity $|A_\omega|^2$ is the *signal power* at the given frequency ω. If we are dealing with real signals, we are mostly interested in the total power at the absolute value of the frequency, $|A_\omega|^2 + |A_{-\omega}|^2$ for $\omega \geq 0$.

For a continuous signal f with the Fourier transform, we define the *doublesided power spectral density* (PSD) as

$$S(\omega) = \left|F(\omega)\right|^2, \quad \omega \in \mathbb{R},$$

while the single-sided power spectral density is

. $$S(\omega) = \left|F(-\omega)\right|^2 + \left|F(\omega)\right|^2, \quad \omega \in \mathbb{R}_+$$

Often, the double-sided spectral density of a signal is defined via the single-sided density in which the power of a component with the frequency ω is equal to the power of the component with the frequency −ω, and then $S(\omega) = 2|F(\omega)|^2$.

For discrete data $\{f_j\}$ with the transform the discrete double-sided power spectral distribution $\{S_k\}$ is defined in analogy to the continuous case,

$$S_k = |F_k|^2, k = 0, 1, \ldots, N - 1$$

The quantity S_k measures the power of the signal at the frequency $2\pi k/N$, while S_{N-k} corresponds to the frequency $-2\pi k/N$, where k = 0,1,...,N/2 − 1. For the single-sided distribution we sum over the powers of negative and positive frequencies. For odd N, the single-sided distribution $\{S_k\}$ is defined as

$$S_0 = |F_0|^2,$$
$$2S_k = |F_k|^2 + |F_{N-k}|^2, \quad k = 1, 2, \ldots, (N-1)/2,$$

while for even N it is given by

$$S_0 = |F_0|^2,$$
$$2S_k = |F_k|^2 + |F_{N-k}|^2, \quad k = 1, 2, \ldots, N/2,$$
$$S_{N/2} = |F_{N/2}|^2.$$

In the discrete case, Parseval's equality applies in the form

$$\frac{1}{N}\sum_{j=0}^{N-1} |f_j|^2 = \sum_{k=0}^{N-1} |F_k|^2 .$$

1.3 Transformations with Orthogonal Polynomials

Functions can also be expanded in terms of orthogonal polynomials [12]. Let us work in P_N, the space of polynomials of degree ≤ N, and restrict the discussion to Legendre and Chebyshev polynomials. The polynomials from these families are orthogonal on [−1,1] with respect to the weight function w. In the function space $L^2_w(-1,1)$ we define the scalar product

$$\langle u, v\rangle_w = \int_{-1}^{1} u(x)v(x)w(x)\,\mathrm{d}x$$

and the norm $\|u\|_w = \sqrt{\langle u, u\rangle_w}$, where w(x) = 1 for Legendre and $w(x) = (1 - x^2)^{-1/2}$ for Chebyshev polynomials. Orthogonality means that

$$\int_{-1}^{1} p_i(x)p_j(x)w(x)\,\mathrm{d}x = \begin{cases} \text{const.}; & i = j, \\ 0; & i \neq j. \end{cases}$$

The expansion of the function u into an infinite series of orthogonal polynomials has the form

$$u(x) \equiv Su = \sum_{k=0}^{\infty} \widehat{u}_k p_k(x), \quad \widehat{u}_k = \frac{1}{\|p_k\|_w^2}\int_{-1}^{1} u(x)p_k(x)w(x)\,\mathrm{d}x$$

Here $\widehat{u}_k$ are the expansion coefficients which may be understood as the transforms of u. For $N \in \mathbb{N}$ we have a finite series

$$S_N u \equiv \sum_{k=0}^{N} \widehat{u}_k \, p_k(x)$$

1.3.1 Relation to Quadrature Formulas

Orthogonal polynomials are closely related to quadrature formulas. For any quadrature we need the *collocation points* or *nodes* $x_j \in [-1,1]$ as well as the *quadrature weights* $w_j > 0$ for $j = 0,1,...,N$. With these quantities the integral of a continuous function can be approximated by the sum of the products of the weights and the function values at the nodes,

$$\int_{-1}^{1} f(x) w(x) \, \mathrm{d}x = \sum_{j=0}^{N} f(x_j) w_j + R_N \quad .$$

The remainder R_N should be minimal or zero if f is a polynomial of a certain maximum degree. These degrees depend on the choice of the nodes.

In *Gauss quadrature* the nodes $x_0,x_1,...,x_N$ are zeros of the orthogonal polynomial p_{N+1}, which lie in the interior of the interval $[-1,1]$. The quadrature formula is then exact for polynomials of degree at most $2N + 1$. In *Gauss–Radau quadrature* the nodes are zeros of the polynomial $q(x) = p_{N+1}(x) + ap_N(x)$, where a is chosen such that $q(-1) = 0$. Then $x_0 = -1$, $x_1,x_2,...,x_N$ are the zeros of q and the quadrature formula is exact for polynomials of degree at most 2N. In *Gauss– Lobatto quadrature* we also include the point $x_N = 1$. This time we are seeking the $N + 1$ zeros of the polynomial $q(x) = p_{N+1}(x) + ap_N(x) + bp_{N-1}(x)$, where a and b are chosen such that $q(-1) = q(1) = 0$. The quadrature is then exact for polynomials of degree at most $2N - 1$. The weights w_j depend on the class of the orthogonal polynomials.

1.3.2 Discrete Transformation

Following the notational conventions, we define the discrete transformation with orthogonal polynomials as

$$u(x_j) = \sum_{k=0}^{N} \widetilde{u}_k \, p_k(x_j)$$

while its inverse is

$$\tilde{u}_k = \frac{1}{\gamma_k} \sum_{j=0}^{N} u(x_j) p_k(x_j) w_j, \quad \gamma_k = \sum_{j=0}^{N} p_k^2(x_j) w_j$$

These equations for polynomial transforms are analogous to the discrete Fourier transforms with trigonometric polynomials.

In collocation methods (for example, for the solution of partial differential equations in Chap. 11) we can represent a smooth function u on the interval [−1,1] by its discrete values at the collocation points, while the derivatives of u are approximated by the derivatives of its interpolation polynomial. The interpolation polynomial is an element of P_N, and its values at the collocation points are equal to the function values, $I_N u(x_j) = u(x_j)$ for $0 \le j \le N$. It is formed by the sum

$$I_N u(x) = \sum_{k=0}^{N} \tilde{u}_k p_k$$

It can be shown—just like in Fourier expansions—that the interpolant $I_N u$ is the projection of u on the space P_N with respect to the scalar product

$$\langle u, v \rangle_N = \sum_{j=0}^{N} u(x_j) v(x_j) w_j$$

This means that $\langle I_N u, v \rangle_N = \langle u, v \rangle_N$ for any continuous function v. Expression can therefore also be read as $= u, p_k{}_N$, and the orthogonality relation as $\langle p_j , p_k \rangle_N = \gamma_k \delta_{j,k}$ for $0 \le k \le N$. The relation between the discrete polynomial coefficients and the coefficients of the continuous expansion is

$$\tilde{u}_k = \hat{u}_k + \frac{1}{\gamma_k} \sum_{j>N} \langle p_j, p_k \rangle_N \hat{u}_j$$

In general $\langle p_j , p_k \rangle_N = 0$ for $j > N$, so the kth component of the interpolant $I_N u$ depends on the th component of u and *all* components with indices $k > N$. We can again rewrite it in the form $I_N u = S_N u + R_N u$, where in

$$R_N u = I_N u - S_N u = \sum_{k=0}^{N} \left(\frac{1}{\gamma_k} \sum_{j>N} \langle p_j, p_k \rangle_N \hat{u}_j \right) p_k$$

we again recognize the aliasing error. The aliasing error is orthogonal to the series truncation error $u - S_N u$, hence

$$\|u - I_N u\|_w^2 = \|u - S_N u\|_w^2 + \|R_N u\|_w^2$$

1.4 Legendre Polynomials

Legendre polynomials P_k for k = 0,1,... are the eigenfunctions of the singular Sturm–Liouville problem

$$\left((1-x^2)P_k'(x)\right)' + k(k+1)P_k(x) = 0, \quad x \in [-1,1]$$

which is of the form (8.84) with $p(x) = 1 - x^2$, q(x) = 0, and w(x) = 1. The polynomials P_k for even (odd) k are even (odd). With the normalization $P_k(1) = 1$

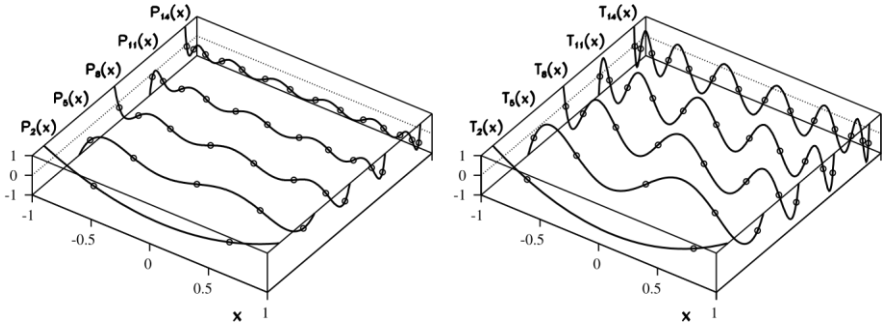


Fig. 1.7 Orthogonal polynomials used in the transformations of functions, collocation methods, and spectral methods for partial differential equations. [*Left*] Legendre polynomials

they can be generally written as

$$P_k(x) = \frac{1}{2^k} \sum_{l=0}^{\lfloor k/2 \rfloor} (-1)^l \binom{k}{l} \binom{2k-2l}{k} x^{k-2l}$$

where k/2 is the integer part of k/2. Legendre polynomials possess a three-term recurrence relation

$$(k+1)P_{k+1}(x) = (2k+1)xP_k(x) - kP_{k-1}(x), \quad P_0(x) = 1, \; P_1(x) = x$$

which is a great tool to actually compute the polynomial values. Some typical polynomials P_k are shown in Fig. 1.7 (left).

By using the polynomials P_k any function $u \in L^2(-1,1)$ can be expanded in a series

$$u(x) = \sum_{k=0}^{\infty} \widehat{u}_k P_k(x), \quad \widehat{u}_k = \frac{2k+1}{2} \int_{-1}^{1} u(x) P_k(x)\, dx.$$

For quadrature with Legendre polynomials there are no explicit formulas for the collocation points x_j; we must compute the zeros of the corresponding polynomials.

1.4.1 Gauss

For Legendre–Gauss quadrature the collocation points x_j (j = 0,1, ...,N) are the roots of the equation $P_{N+1}(x_j) = 0$, while the corresponding weights and normalization factors are

$$w_j = \frac{2}{(1-x_j^2)[P'_{N+1}(x_j)]^2}, \qquad \gamma_k = \frac{2}{2k+1}$$

1.4.2. Gauss–Radau

The collocation points for Legendre–Gauss–Radau quadrature are the roots of the equation $P_N(x_j) + P_{N+1}(x_j) = 0$, and the weights are

$$w_0 = \frac{2}{(N+1)^2}, \qquad w_j = \frac{1-x_j}{(N+1)^2[P_N(x_j)]^2}, \quad j = 1, 2, \ldots, N$$

The normalization factors are $\gamma_k = 2/(2k + 1)$.

1.4.3. Gauss–Lobatto

The most frequently used Legendre–Gauss–Lobatto quadrature explicitly includes both endpoints of the interval, $x_0 = -1$ and $x_N = 1$, while the interior points x_j are the roots of the equation $P'_N(x_j) = 0$:

$$x_j = \begin{cases} -1; & j = 0, \\ \text{solutions of } P'_N(x_j) = 0; & 1 \le j \le N-1, \\ 1; & j = N. \end{cases}$$

In this case the weights are

$$w_j = \frac{2}{N(N+1)} \frac{1}{[P_N(x_j)]^2}, \quad j = 0, 1, \ldots, N.$$

The normalization factors are $\gamma_k = 2/(2k + 1)$ for $k < N$ and $\gamma_N = 2/N$.

1.5 Chebyshev Polynomials

Chebyshev polynomials T_k for k = 0,1,... [13] are the eigenfunctions of the singular Sturm–Liouville problem

$$\left(\sqrt{1-x^2}\,T_k'(x)\right)' + \frac{k^2}{\sqrt{1-x^2}}\,T_k(x) = 0, \qquad x \in [-1, 1]$$

which is of the form (8.84) with $p(x) = (1 - x^2)^{1/2}$, $q(x) = 0$, and $w(x) = (1 - x^2)^{-1/2}$. The polynomials T_k for even (odd) k are even (odd). With the normalization $T_k(1) = 1$ they are defined as

$$T_k(x) = \cos(k \arccos x) = \sum_{i=0}^{\lfloor k/2 \rfloor} \binom{k}{2i} (x^2 - 1)^i x^{k-2i}$$

The polynomials are related by the three-term recurrence

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x), \qquad T_0(x) = 1, T_1(x) = x,$$

which allows us to compute the values $T_n(x)$ in just O(n) operations. Chebyshev polynomials are orthogonal on the interval [−1,1] with respect to the weight $(1 - x2)^{-1/2}$,

$$\int_{-1}^{1} T_k(x) T_l(x) \frac{dx}{\sqrt{1-x^2}} = \begin{cases} 0; & k \neq l, \\ \pi/2; & k = l \neq 0, \\ \pi; & k = l = 0, \end{cases}$$

but they also possess the peculiar property of *orthogonality by points*: on the discrete set of points

$$x_j = \cos\frac{(2j+1)\pi}{2N}, \qquad j = 0, 1, \ldots, N-1$$

corresponding to the N zeros of the polynomial T_N, Chebyshev polynomials are orthogonal in the sense of the sum

$$\sum_{j=0}^{N-1} T_k(x_j)T_l(x_j) = \begin{cases} 0; & k \neq l, \\ N/2; & k = l \neq 0, \\ N; & k = l = 0. \end{cases}$$

s

Any function u ∈ $L^2(-1,1)$ can be expanded in terms of Chebyshev polynomials T_k as

$$u(x) = \sum_{k=0}^{\infty} \widehat{u}_k T_k(x), \quad \widehat{u}_k = \frac{2}{\pi c_k} \int_{-1}^{1} u(x)T_k(x)w(x)\,dx,$$

where $c_0 = 2$ and $c_k = 1$ for $k \geq 1$. In contrast to quadrature involving Legendre polynomials, the nodes and the weights for Chebyshev quadrature are given by explicit formulas which are given in the following. Note that the nodes are indexed such that the values of x_j decrease when j increases.

1.5.1 Gauss

For Chebyshev–Gauss collocation the nodes and the weights are

$$x_j = \cos\frac{(2j+1)\pi}{2N+2}, \qquad w_j = \frac{\pi}{N+1}, \quad j = 0, 1, \ldots, N,$$

while the factors γ_k in this equation are equal to $\gamma_k = \pi c_k/2$ for $0 \leq k < N$ and $\gamma_N = \pi/2$.

1.5.2 Gauss–Radau

For Chebyshev–Gauss–Radau collocation we have

$$x_j = \cos\frac{2j\pi}{2N+1}, \qquad w_j = \begin{cases} \frac{\pi}{2N+1}; & j = 0, \\ \frac{2\pi}{2N+2}; & j = 1, 2, \ldots, N, \end{cases}$$

while $\gamma_k = \pi c_k/2$ for $0 \leq k < N$ and $\gamma_N = \pi/2$.

1.5.3 Gauss–Lobatto

The most frequently used Gauss–Lobatto collocation includes the endpoints $x_0 = 1$ and $x_N = -1$. Here, the nodes and the weights are given by

$$x_j = \cos\frac{j\pi}{N}, \qquad w_j = \begin{cases} \frac{\pi}{2N}; & j = 0, N, \\ \frac{\pi}{N}; & j = 1, 2, \ldots, N-1 \end{cases}$$

The normalization factors are $\gamma_k = \pi c_k/2$ for $0 \leq k < N$ and $\gamma_N = \pi$.

Discrete Chebyshev transformation on the interval x ∈ [−1,1] is most often associated with Gauss–Lobatto collocation, and this is the only case we discuss henceforth. The expansion of the function u in a finite series has the form

$$I_N u(x) = \sum_{k=0}^{N} \tilde{u}_k T_k(x), \quad \tilde{u}_k = \frac{2}{N\bar{c}_k} \sum_{j=0}^{N} \frac{1}{\bar{c}_j} u(x_j) T_k(x_j$$

where $\bar{c}_0 = \bar{c}_N = 2$ and $\bar{c}_j = 1$ for j = 1, 2, . . . , N − 1. This is a discrete equivalent of the continuous expansion. At the Gauss–Lobatto quadrature nodes, the expansion can be rewritten as

$$I_N u(x_j) = \sum_{k=0}^{N} \tilde{u}_k \cos\frac{\pi jk}{N}, \quad \tilde{u}_k = \frac{2}{N\bar{c}_k} \sum_{j=0}^{N} \frac{1}{\bar{c}_j} u(x_j) \cos\frac{\pi jk}{N}$$

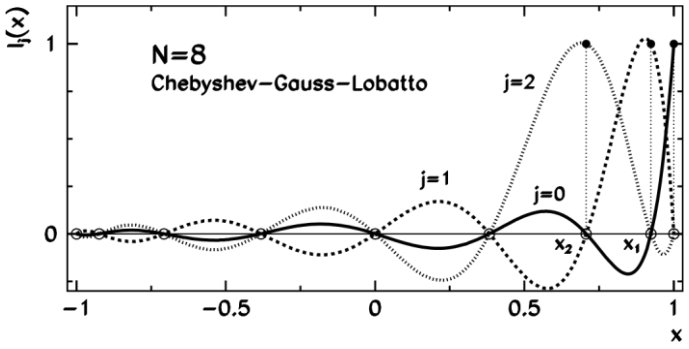


Fig. 1.8 The first three Lagrange interpolation polynomials $l_j(x)$ for Chebyshev–Gauss–Lobatto collocation with N = 8. The collocation points $x_j = \cos(\pi j/N)$, at which $l_j(x_k) = \delta_{j,k}$ holds, follow from right to left for indices j = 0,1,...,N

which is nothing but the cosine transform. The transliteration of the Chebyshev transformation to the Fourier transformation makes a great impact in spectral methods (Sect. 11.1.3).

Table 1.1 Laplace transforms of some common functions.

$f(t) = \mathcal{L}^{-1}[F]$	$F(s) = \mathcal{L}[f]$	Assumptions
t^p	$\Gamma(p+1)/s^{p+1}$	$p > -1, \mathrm{Re}\{s\} > 0$
e^{at}	$1/(s-a)$	$\mathrm{Re}\{s\} > a$
$\sin at$	$a/(s^2+a^2)$	$\mathrm{Re}\{s\} > 0$
$\cos at$	$s/(s^2+a^2)$	$\mathrm{Re}\{s\} > 0$
$\delta(t-c)$	e^{-cs}	$\mathrm{Re}\{s\} > 0$
$\theta(t-c)$	e^{-cs}/s	$\mathrm{Re}\{s\} > 0$

1.6 Laplace Transformation

The continuous Laplace transformation of the function f is defined as

$$F(s) = \mathcal{L}[f](s) = \int_0^\infty e^{-st} f(t)\,dt, \quad s \in \mathbb{C}$$

The sufficient conditions for the existence of the transform L [f] are that f is piecewise continuous on R and that f is of exponential order in the limit t → ∞ : this means that real constants+ C > 0, a, and T > 0 exist such that |f(t)| ≤ Ceat for ∀t > T. The transformation is linear, L [$c_1f_1 + c_2f_2$] = c_1L [f_1] + c_2L [f_2] . The transforms of some typical functions are enumerated in Table 1.1.

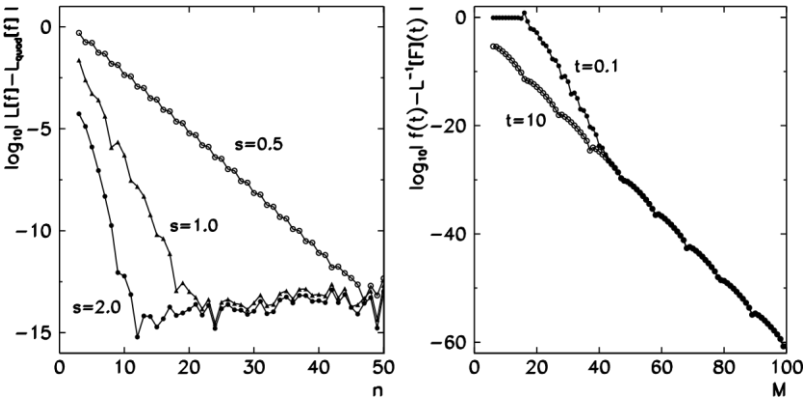


Fig. 1.9 [*Left*] The precision of the quadrature formula for the Laplace transformation of the function f(t) = cost (the exact transform is F(s) = s/(s2 + 1) *Right*] The precision of the FT algorithm to compute the inverse Laplace transform = s/(s^2 + 1) to arbitrary precision (the exact solution is f(t) = cost). Shown is the error

$\log_{10}|f(t) - L^{-1}[F](t)|$ as a function of the number of terms in the quadrature sum for two different t

Laplace transforms of real functions for which we do not know the suitable elementary integral, can be computed by using the Gauss–Laguerre quadrature of high order (see Fig. 1.9 (left)). Assuming that f can be sufficiently well described by a polynomial, the transform with s > 0 can be computed as

$$\mathcal{L}[f](s) = \frac{1}{s}\sum_{i=1}^{n} w_i\, f(x_i/s) + R_n, \qquad R_n = \frac{(n!)^2}{(2n)!} f^{(2n)}$$

whereare the zeros of the Laguerre polynomialare the quadrature weights, and $\xi \in \mathbb{R}$. From the zeros x_i we compute the weights as

$$w_i = \frac{x_i}{[(n+1)L_{n+1}(x_i)]^2}$$

Many modern numerical libraries include the generators of zeros and weights, and they are mostly based on the algorithms presented in. This approach allows for a stable and precise calculation of the zeros and weights in double precision up to order $n \approx 40$. The quadrature at small values of s fails, in particular with oscillatory functions f , because the value $f(x_i/s)$ changes too quickly; in such cases it is preferable to use the formula

$$\mathcal{L}[f](s) \approx \sum_{i=1}^{n} w_i\, e^{(1-s)x_i} f(x_i).$$

1.7 Use of Laplace Transformation with Differential Equations

One of the most fruitful application areas of the Laplace transformation is the study of electric circuits and mechanical systems where we wish to understand the solutions of linear differential equations with discontinuous or impulse forcing terms.

The generic example is the equation $x + \beta\dot{x} + \omega 20x = f(t)$ with the Heaviside (step) function $f(t) = \theta(t)$ or with the impulse . The kernel of the Laplace transformation e^{-st} determines the natural scale of the physical process

Laplace transformation draws its true strength from the relations between the transforms of the function f itself and the transforms of its derivatives. For a piecewise continuous f we have

$$\mathcal{L}[f'] = s\mathcal{L}[f] - f(0)$$

or, with similar assumptions for the higher derivatives ,

$$\mathcal{L}\left[f^{(n)}\right] = s^n \mathcal{L}[f] - s^{n-1} f(0) - \cdots - s f^{(n-2)}(0) - f^{(n-1)}(0).$$

By using the relation in the initial-value problem with constant coefficients

$$a\ddot{x} + b\dot{x} + cx = f(t)$$

with given initial conditions x(0) and x'(0), we obtain

$$a\left[s^2 X(s) - sx(0) - \dot{x}(0)\right] + b\left[sX(s) - x(0)\right] + cX(s) = F(s).$$

Instead of solving the differential equation for x(t) we have succeeded in rephrasing the problem in terms of an algebraic equation for X(s) into which the initial conditions have already been "built in". From the function X(s) we then obtain the solution x(t) by computing the inverse Laplace transform.

Formally, the inverse Laplace transform is given by the formula

$$\mathcal{L}^{-1}[F](t) = \frac{1}{2\pi \mathrm{i}} \int_C \mathrm{e}^{st} F(s)\, \mathrm{d}s,$$

where C is a vertical line in the complex plane, defined by C = ξ + iη, and ξ is chosen such that all singularities of the transform F(s) lie to the left of C. In other words, F is analytic on the half-plane Re{s } > ξ. The sufficient conditions for the existence of the inverse are

$$\lim_{s\to\infty} F(s) = 0, \qquad \lim_{s\to\infty} \left|sF(s)\right| < \infty$$

The inverse Laplace transformation is linear.

The world of the inverse Laplace transformation is not rosy: the expression for X(s), first has to be reshuffled such that in its various parts of the functions F(s) from Table 1.1 are identified, and these are then associated with the corresponding parts of the solution x(t).

1.8 Hilbert Transformation

The Hilbert transformation H of the function s : R → R is defined as the convolution of the function s with the function h(t) = 1/(πt):

$$\hat{s}(t) = \mathcal{H}[s](t) = -(s * h)(t) = \frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{s(\tau)}{\tau - t} \, \mathrm{d}\tau,$$

where $*$ denotes the convolution. The integral should be understood as the Cauchy principal value (symbol P). The principal value integral of the function f over the interval [a,b] , on which f has a singularity at c ∈ [a,b] , is defined as

$$P \int_a^b f(t) \, \mathrm{d}t = \lim_{\varepsilon \searrow 0} \left(\int_a^{c-\varepsilon} f(t) \, \mathrm{d}t + \int_{c+\varepsilon}^b f(t) \, \mathrm{d}t \right)$$

The value of the transform does not change if an arbitrary constant is added to the function, const] = H [s] . The transform of the function s ∈ $L^p(R)$ for exists only in the case that s tends to zero at positive and negative infinity , $\lim_{t \to \pm\infty} s(t) = 0$. The Hilbert transformation is linear and bounded:

$$\left\| \mathcal{H}[s] \right\|_p \leq C_p \|s\|_p, \quad C_p = \begin{cases} \tan(\pi/2p); & p = 1, 2 \\ \tan(\pi/2p)^{-1}; & p > 2. \end{cases},$$

The inverse Hilbert transformation is

$$s(t) = \mathcal{H}^{-1}[\hat{s}](t) = (\hat{s} * h)(t) = -\frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{\hat{s}(\tau)}{\tau - t} \, \mathrm{d}\tau,$$

hence $H^{-1} = -H$ and $H^2 = -id$.

1.8.1 Relation to the Fourier Transform

The Fourier transforms of the signal s, S(ω) = F[s](ω), and of the function h(t) = 1/(πt),

$$H(\omega) = \mathcal{F}[h](\omega) = -\mathrm{i} \operatorname{sign}(\omega), \quad \operatorname{sign}(\omega) = \begin{cases} 1; & \omega > 0, \\ 0; & \omega = 0, \\ -1; & \omega < 0, \end{cases}$$

where F^{-1} is the inverse Fourier transformation. The norm of the function s ∈ $L_2(R)$ with the Fourier transform S ∈ $L_2(R)$ is the same in all three representations,

$$\int_{-\infty}^{\infty} \left| \mathcal{H}[s](t) \right|^2 \mathrm{d}t = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left| \operatorname{sign}(\omega) S(\omega) \right|^2 \mathrm{d}\omega = \lim_{\varepsilon \searrow 0} \left(\int_{-\infty}^{-\varepsilon} + \int_{\varepsilon}^{\infty} \right) |s(t)|^2 \, \mathrm{d}t,$$

where we have used and Parseval's equality for the Fourier transform.

The Hilbert transformation therefore preserves the total power of the signal. A real signal s and its Hilbert transform sˆ = H [s] are orthogonal, i.e.

$$\int_{-\infty}^{\infty} s(t)\hat{s}(t)\,\mathrm{d}t = \frac{\mathrm{i}}{2\pi}\int_{-\infty}^{\infty} \mathrm{sign}(\omega)\left|S(\omega)\right|^2\,\mathrm{d}\omega = 0$$

if s, $\hat{s}$, and the Fourier transform S = F [s] are in $L_1(\mathbb{R})$ or $L_2(\mathbb{R})$ (we assumed $S(\omega)^* = S(-\omega)$). A similar conclusion can be made by examining for the Fourier modes, since for the functions cos ωt and sin ωt, which are orthogonal for all ω = 0, we have

$$\mathcal{H}[\cos\omega t] = -\mathrm{sign}(\omega)\sin\omega t, \qquad \mathcal{H}[\sin\omega t] = \mathrm{sign}(\omega)\cos\omega t.$$

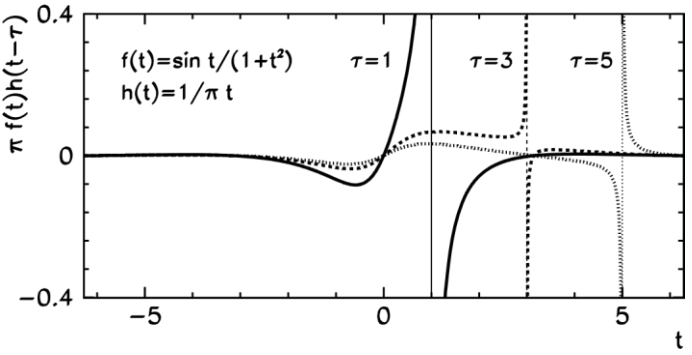


Fig. 1.10 The integrands for the computation of the Hilbert transform of f(t) = sint/(1+t²) for three values τ=1, τ =3, and τ=5. We integrate over the poles migrating along the real axis and "sample" f in a very sensitive manner. See also Sect.

1.9 Analytic Signal

Hilbert transformation is a commonly used tool in the analysis of signals where it is used for their "complexification". This means that to a real signal (function) s(t) is assigned a complex function

$$s^{\#}(t) = s(t) - \mathrm{i}\,\mathcal{H}[s](t)$$

known as the *analytic signal* (s and sˆ are real functions). If the signal is s(t) = cosωt, ω > 0, the analytic signal is $s^{\#}(t) = \cos \omega t + i \sin \omega t = \exp(i\omega t)$. The Fourier transform of the analytic signal is

$$S^{\#}(\omega) = S(\omega)\left[1 + \operatorname{sign}(\omega)\right]$$

where S(ω) = F [s](ω) is the Fourier transform of the signal s. The function $S^{\#}$ is zero for negative frequencies ω < 0: this is a fundamental property of the analytic signals. If the function $s^{\#}$ is analytic on the upper complex half-plane and satisfies the Cauchy integral equation

$$P \int_{-\infty}^{\infty} \frac{s^{\#}(\xi)}{\xi - x} \, \mathrm{d}\xi = \mathrm{i}\pi s^{\#}$$

we are referring to a *strongly analytic signal*. In addition to analyticity, the sufficient condition for the validity of the integral equation is that the function falls off quickly enough in the upper complex half-plane. If the signal $s^{\#}$ is strongly analytic, we may insert $s^{\#} = s - i\hat{s}$ and obtain the relations between the real and imaginary parts of the analytic signal,

$$\mathcal{H}[s] = \hat{s}, \qquad \mathcal{H}[\hat{s}] = -s.$$

The analytic signal can be represented in the complex plane as

$$s^{\#}(t) = A(t)\, \mathrm{e}^{\mathrm{i}\phi(t)},$$

where

$$A(t) = \left|s^{\#}(t)\right| = \sqrt{\left(s(t)\right)^2 + \left(\mathcal{H}[s](t)\right)^2}$$

is the analytic amplitude, and

$$s^{\#}(t) = A(t)\, \mathrm{e}^{\mathrm{i}\phi(t)},$$

is the analytic phase. In this representation we define various *instantaneous* quantities, like the *instantaneous signal power* $E_i(t) = |A(t)|^2$ and the *instantaneous complex phase* $\Phi_i(t) = \log s\ (t) = \log A(t) + i\phi(t)^{\#}$. By using the time derivative of the complex phase we also define the *instantaneous complex frequency*

$\Omega_i(t) = \dot{\Phi}_i(t) = \alpha_i(t) + i\beta_i(t)$, where $\alpha_i(t) = A(t)/A(t)^{\cdot}$ is the *instantaneous radial*

frequency and $\beta_i(t) = \varphi(^{\cdot}t)$ is the *instantaneous angular frequency*. These quantities help us in unraveling the characteristics of the signal s which are not apparent from observing just the time dependence of s(t).

1.10 Kramers–Kronig Relations

Relations between real and imaginary parts of the Fourier transforms play prominent roles in many areas of physics. These relations have the form of Hilbert transforms. Let s(t) be a real signal with the complex Fourier transform $S(\omega) = F\,[s\,](\omega)$ which is an analytic function on the upper half-plane of the complex variable ω.

$$\mathrm{Re}\ S(\omega) = \frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{\mathrm{Im}\, S(\omega')}{\omega' - \omega} \mathrm{d}\omega \quad ,\mathrm{Im}\ S(\omega) = -\frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{\mathrm{Re}\, S(\omega')}{\omega' - \omega} \mathrm{d}\omega'$$

in short, $\mathrm{ReS}(\omega) = H\,[\mathrm{ImS}\,](\omega)$ and $\mathrm{ImS}(\omega) = -H\,[\mathrm{ReS}\,](\omega)$. These equations are known as the *Kramers–Kronig (dispersion) relations*. Relations of this type can be formulated for all Fourier transforms of signals which appear in the descriptions of the system's response to an external perturbation. They are very general: their derivation requires only that the response of the system is causal. This means that the signal s at times $t > 0$ is a consequence of the events occurring at $t \leq 0$.

In the frequency representation this means $P(\omega) = \varepsilon_0 \chi(\omega) E(\omega)$, where $\chi(\omega)$ is the susceptibility which depends on the frequency of the external perturbation $E(\omega)$. Susceptibility χ satisfies the above Kramers–Kronig relations, which can be further simplified by using symmetry properties. We have $\chi(\omega)^* = \chi(-\omega)$, or $\mathrm{Re}\chi(-\omega) = \mathrm{Re}\chi(\omega)$ and $\mathrm{Im}\chi(-\omega) = -\mathrm{Im}\chi(\omega)$, so the integration range can be narrowed down to positive frequenciesonly,

$$\mathrm{Re}\,\chi(\omega) = \frac{2}{\pi} P \int_0^{\infty} \frac{\omega' \,\mathrm{Im}\,\chi(\omega')}{\omega'^2 - \omega^2} \mathrm{d}\omega',$$

$$\mathrm{Im}\,\chi(\omega) = -\frac{2}{\pi} P \int_0^{\infty} \frac{\omega \,\mathrm{Re}\,\chi(\omega')}{\omega'^2 - \omega^2} \mathrm{d}\omega'.$$

We need only one more step to find the relations between the real (dispersive) and the imaginary (absorption) part of the complex refraction index. In the limit of small susceptibilities we have

$$N(\omega) = \sqrt{1 + \mathrm{Re}\,\chi(\omega) + i\,\mathrm{Im}\,\chi(\omega)} \approx \underbrace{1 + \frac{1}{2}\mathrm{Re}\,\chi(\omega)}_{n(\omega)} + \underbrace{\frac{i}{2}\mathrm{Im}\,\chi(\omega)}_{i\kappa(\omega)}.$$

it then follows that

$$n(\omega) = 1 + \frac{2}{\pi} P \int_0^\infty \frac{\omega' \kappa(\omega')}{\omega'^2 - \omega^2} \, \mathrm{d}\omega'.$$

It is relatively hard to measure the frequency dependence of the real part of the refraction index n. But by using the derived relations it can be computed from the imaginary part κ which can be determined easily, just by measuring the ratio of the incident and transmitted wave intensities at different frequencies.

1.11 Numerical Computation of the Continuous Hilbert Transform

The numerical computation of the Hilbert transform is almost always a tough nut to crack, as the transformation involves the integral of a singular function. Here we mention a few strategies for a quick and stable computation.

1.11.1 Transforming the Integrand to a Sum of Orthogonal Polynomials

Let us first generalize the definition of the Hilbert transformation to integrals over the interval $I \subset \mathbb{R}$,

$$\mathcal{H}[f](y) = \frac{1}{\pi} P \int_I \frac{f(x)}{x - y} \, \mathrm{d}x.$$

If the interval I = [a,b] is finite, we can rewrite the integral as

$$\mathcal{H}[f](y) = \frac{1}{\pi} f(y) \log \left| \frac{b - y}{a - y} \right| + \frac{1}{\pi} \int_a^b \frac{f(x) - f(y)}{x - y} \, \mathrm{d}x,$$

which eliminates the singularity. We write the integrand as the product of a positive weight function w and the remaining factor, f(x) − f(y) = w(x)g(x ; y). With foresight, we would like to find a particular pair (interval I, weight function w) that corresponds to the definition domain and the weight function of some system of orthogonal polynomials. Then we could approximate the function g by the expansion

$$g(x; y) \approx \sum_{j=0}^{n} d_j(y) q_j$$

where $\{q_j\}_j \in \mathbb{N}_0$ are orthogonal polynomials, and compute the Hilbert transform as the weighted sum of the transforms of the individual terms in this expansion:

$$\mathcal{H}[wg](y) \approx \sum_{j=0}^{n} d_j(y)\psi_j(y), \quad \psi_j(y) = \mathcal{H}[wq_j]$$

All systems of orthogonal polynomials q_j possess three-term recurrence relations which also apply to the functions ψ_j, e.g. $\psi_{j+1}(x) = (A_j x + B_j)\psi_j(x) + C_j\psi_{j-i}(x)$, where A_j, B_j, and C_j are constants that do not depend on x. This relation between q_j and ψ_j is based on the property of the Hilbert transformation

$$\mathcal{H}[xf](y) = y\mathcal{H}[f](y) + \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)\,dx$$

and the orthogonality of the polynomials q_j to a constant, $\int_{-\infty}^{\infty} q_j(x)w(x)\,dx = 0$ for $j > 0$. Using the recurrence to compute ψ_j speeds up tremendously the computation of the transform H [wf](y). for the computation

with the Hermite polynomials ($w(x) = e^{-x^2}$ and $I = \mathbb{R}$) and the generalized Laguerre polynomials ($w(x) = x^{\alpha}e^{-x}$ and $I = \mathbb{R}_+$).

1.11.2 Quadrature Formulas

If our knowledge about the behavior of the integrand is very limited, the Hilbert transform can be computed by using quadrature formulas. A simple one is

$$\mathcal{H}[f](y) \approx \frac{2}{\pi}\sum_{n\in\mathbb{Z}} \frac{f(y + (2n+1)h)}{2n+1}$$

1.11.3 Collocation Method

In this approach

$$\rho_n(x) = \frac{(1+ix)^n}{(1-ix)^{n+1}}, \quad n \in \mathbb{Z}$$

which satisfy $H[\rho_n](x) = \text{isign}(n)\rho_n(x)$ and are therefore the eigenfunctions of the Hilbert transformation. The functions ρ_n on $L^2(\mathbb{R})$ form a complete orthogonal basis and fulfill. $\int_{-\infty}^{\infty} \rho_m^*(x)\rho_n(x)\,dx = \pi\delta_{m,n}$ An arbitrary function $f \in L^2(\mathbb{R})$ canbe expanded in terms of these basis functions as

$$f(x) = \sum_{n\in\mathbb{Z}} a_n \rho_n(x), \quad a_n = \frac{1}{\pi}\int_{-\infty}^{\infty} \rho_n(x)^* f(x)\,\mathrm{d}x.$$

The Hilbert transform of the function f is then

$$\mathcal{H}[f](x) = \mathrm{i}\sum_{n\in\mathbb{Z}} \operatorname{sign}(n) a_n \rho_n(x) \approx \mathrm{i}\sum_{n=-N}^{N-1} \operatorname{sign}(n) a_n \rho_n$$

If the function f is real, the expansion coefficients satisfy $a_n = a^*_{-n-1}$, so one needs to compute only half of the coefficients a_n for n = 0,1,...,N − 1. Alternatively, one can use the substitution x = tan(φ/2) to rewrite the very same expression as

$$a_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} g_n(\phi)\,\mathrm{d}\phi, \quad g_n(\phi) = \left(1 - \mathrm{i}\tan\frac{\phi}{2}\right) f\left(\tan\frac{\phi}{2}\right) \mathrm{e}^{-\mathrm{i}n\phi}$$

The function g is periodic on [−π,π] and $\lim_{\varphi \to \pm\pi} g(\varphi) = 0$ is assumed. The approximation of the integral in the above expression can be obtained by various integration methods, for example, by using the trapezoidal rule with the points $\varphi_j = \pi j/N$ for |j | < N. This results in the approximation

$$a_n \approx \frac{1}{2N}\sum_{j=-N+1}^{N-1} g_n(\phi_j).$$

By evaluating this formula with the fast Fourier transform we can compute the coefficients a_n for all n at once, requiring only O(N logN) operations. This approach is reliable in cases where the coefficients a_n rapidly decrease as n increases.

1.12 Discrete Hilbert Transformation

We sample a continuous signal s equidistantly in t in steps of and obtain the discrete signal $\{s_k\}_{k\in z}$ to which we assign a discrete Hilbert transform $\{\hat{s}_k\}_{k\in z}$. While there are several ways to do this, all of them reproduce the continuous transform in the limit → is smooth enough. The discrete Hilbert transform is an arbitrarily good approximation of the continuous one.

1.12.1 Time-Domain Approach

The discrete variant of the Hilbert transform for a finitely long signal $\{s_k\}$, specified at times $t_k = k\Delta$, can be cast in the form

$$\hat{s}_n = \mathcal{H}_N[s]_n = \frac{1}{\pi}\sum_{m=1}^{N}\frac{s_{n+m} - s_{n-m}}{m},$$

where N is the number of points in the sample to the left and right of t_n which are included in the sum. The discrete transform reverts to the continuous one when $N \to \infty$ and $\Delta \to 0$. "Infinitely long" signals occur when there is an incessant flow of data from the measuring apparatus and we wish to process them in real time, or if the amount of data is large compared to the available memory. We should therefore always estimate the extent of the signal with respect to some reference point that we still wish to use in the computation of the transform.

It is possible to compute the discrete Hilbert transform by numerically evaluating the integral in the continuous transform . We either use the interpolant of the signal s or a chosen model function is fitted to the signal: both can be simply and stably convoluted with the function $h(t) = 1/(\pi t)$. By linear interpolation of the signal values we obtain the discrete transform

$$\begin{aligned}\mathcal{H}_N[s]_n = -\frac{1}{\pi}\Bigg\{&\sum_{k=0}^{n-2} s_k\phi(k-n+1) + (s_k - s_{k+1})\big[1 + (n-k)\phi(n-k)\big] \\ &+ s_{n-1} - s_{n+1} + \sum_{k=n+2}^{N-1} s_k\phi(k-n) \\ &+ (s_{k-1} - s_k)\big[1 + (k-n)\phi(k-n)\big]\Bigg\},\end{aligned}$$

This computation requires $O(N^2)$ operations. Boche's approach which is tailored to signals of limited bandwidth, is also found in practice.

1.12.2 Fourier-Domain Approach

The Fourier approach is fruitful for periodic signals $\{s_k\}_{k=0}^{N-1}$ (for which $s_{k+N} = s_k$). The discrete Fourier transform of the signal is

$$S_n = \mathcal{F}_N[s]_n = \frac{1}{N}\sum_{k=0}^{N-1} s_k\, e^{-i2\pi nk/N}$$

We define the discrete Hilbert transform for such a signal as the convolution

$$\hat{s}_n = \mathcal{H}_N[s]_n = \sum_{k=0}^{N-1} h_{n-k} s_k = \sum_{k=0}^{N-1} h_k s_{n-k}$$

Where $\{h_k\}_{k=0}^{N-1}$ is the convolution kernel and we assume $h_{k+N} = h_k$, $s_{k+N} = s_k$. We determine the kernel by translating the property of the continuous Hilbert transform to discrete language, namely H [e](t) = isign(ω)e(t), where e(t) = exp(iωt). This means

$$\mathcal{H}_N\left[e^{(k)}\right]_n = \mathrm{i}\,\mathrm{sign}(N-2k)\, e_n^{(k)}, \quad e_n^{(k)} = \exp(\mathrm{i}\,2\pi kn/N).$$

We have already taken into account the definitions of the negative and positive frequencies in the discrete Fourier transform. By using the relation between the discrete Fourier transform and the convolution, we can express the Hilbert transform as

$$\mathcal{H}_N[s]_n = -N\mathcal{F}_N^{-1}[F]_n, \quad F = \{H_i S_i\}_{i=0}^{N-1}$$

we compute the components of the Fourier transform S_k of the signal s, multiply them by isign(N − 2k), and compute the inverse Fourier transform of the product:

$$\mathcal{H}_N[s]_n = \mathrm{i}\sum_{k=0}^{N-1} S_k\,\mathrm{sign}(N-2k)\,\mathrm{e}^{\mathrm{i}\,2\pi nk/N}$$

By analogy with the continuous case we define the complex analytic signal corresponding to the discrete Hilbert transform:

$$s_n^{\#} = s_n - \mathrm{i}\,\hat{s}_n, \quad \hat{s}_n = \mathcal{H}_N[s]_n.$$

Its real and imaginary parts can again be used to elegantly compute the instantaneous quantities. For example, the instantaneous radial frequency α_n and the instantaneous angular frequency β_n become

$$\alpha_n = \frac{s_n' s_n + \hat{s}_n' \hat{s}_n}{s_n^2 + \hat{s}_n^2}, \qquad \beta_n = \frac{\hat{s}_n' s_n - s_n' \hat{s}_n}{s_n^2 + \hat{s}_n^2},$$

where denotes the time derivative. Note that even these derivatives can be computed by using the discrete Fourier transformation. If the array $\{f_k\}_{k=0}^{N-1}$ corresponds to the discrete Fourier transform $\{F_n\}_{n=0}^{N-1}$, the time derivative of the component f_k is given by

$$f'_k = \sum_{n=0}^{N-1} \omega_n F_n \, e^{i2\pi nk/N}, \quad \omega_n = \frac{2\pi}{N}\left[\left(n - \frac{N}{2}\right)\text{sign}(N-2n) + \frac{N}{2}\right],$$

where the physical angular frequencies ω_k belong to the individual Fourier modes. Only O(N logN) operations are needed if FFT is used to compute the discrete Hilbert transform.

The Fourier-domain approach is the fastest among the possibilities described here, but it does have its deficiencies. The signal is periodic but rapid changes in the signal may be aliased in the frequency spectrum and therefore misinterpreted when the convolution is performed. It is not wise to use the Fourier approach if the Hilbert transform falls off too slowly at infinity as the finite Fourier series is not optimal for the description of the long tails.

1.13 Wavelet Transformation

We may think of the *wavelet transformation* of a signal as an extension of its Fourier analysis, through which not only the strengths of the signal's frequency components are determined, but also the times at which these components occur. The classic example in Fig. 1.11 illustrates this basic idea for the signal $\sin(t^2)$ whose frequency linearly increases with time. By using the wavelet transformation we can also locate changes in the signal that are not immediately apparent from its temporal behavior alone (Fig. 1.12).

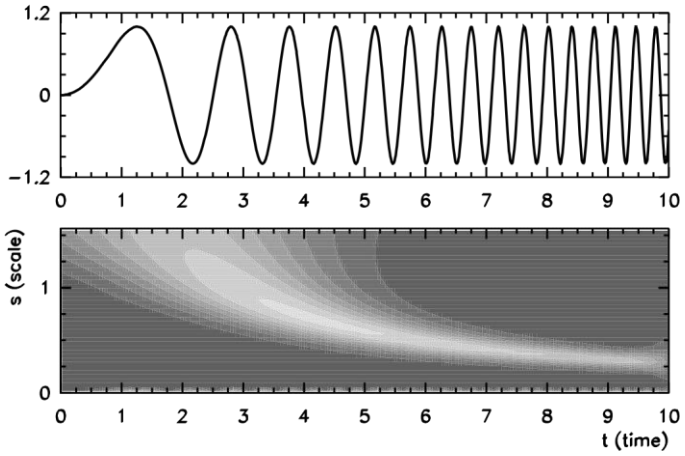


Fig. 1.11 The basic idea of the continuous wavelet transform. [*Top*] The signal f(t) = sinωt, ω ∝ t. [*Bottom*] In this portion of the signal, the continuous wavelet transformation detects large structures at short times (where the waves have a

typical scale s ≈ 1.5) and small structures at long times (scale s ≈ 0.3). The frequency and the scale of the oscillations are inversely proportional, which generates the typical curvature of the transform ($s \propto \omega^{-1} \propto t^{-1}$)

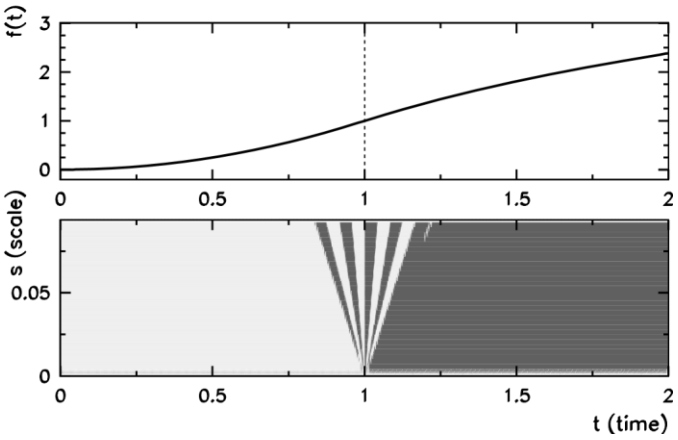


Fig. 1.12 Continuous transform of a real signal with a complex Morlet wavelet [*Top*] The signal f (t) = t2 (t < 1) or f (t) = 1 + 2 log t (t ≥ 1) is continuous and has a continuous first derivative at t = 1 while its second derivative is discontinuous. [Bottom] The phase of the wavelet transform in the vicinity of t = 1 oscillates a couple of times, and reveals the location of the critical point when the scale s is decreasedtransform in the vicinity of t = 1 oscillates a couple of times, and reveals the location of the critical point when the scale s is decreased

The continuous wavelet transform (CWT) of the function f is defined as

$$L_\psi[f](s,t) = \frac{1}{\sqrt{c_\psi s}} \int_{-\infty}^{\infty} f(\tau)\psi^*\left(\frac{\tau - t}{s}\right) \mathrm{d}\tau, \quad t \in \mathbb{R},\ s \neq 0$$

where t is the time at which a feature of scale s is observed in the function f , and c_ψ is the normalization constant. The function ψ, whose properties are given in the following, should allow us to change the parameter s (the typical scale of a structure in the signal f) as well as its shift t with respect to the signal f . By denoting

$$\psi_s(t) = \psi^*(-t/s)$$

we can rewrite the definition in the form of a convolution

$$L_\psi[f](s,t) = \frac{1}{\sqrt{c_\psi s}} \int_{-\infty}^{\infty} f(\tau)\psi_s(t-\tau)\,\mathrm{d}\tau.$$

The function ψ should satisfy certain conditions. Its "energy" should be bounded, and the weighted integral of its spectral density (the square of the Fourier transform $\hat{\psi}$) should be finite:

$$c_\psi = 2\pi \int_{-\infty}^{\infty} \frac{1}{|\omega|} \left|\hat{\psi}(\omega)\right|^2 \mathrm{d}\omega < \infty$$

The functions ψ found in the literature usually fulfill this *admissibility condition* by design. Moreover, we require the functions ψ to fulfill

$$\int_{-\infty}^{\infty} \psi(\eta)\,\mathrm{d}\eta = 0$$

The functions ψ therefore oscillate around the abscissa and fall off rapidly at large distances from the origin, giving them the appearance of small waves and the nickname *wavelets*. The simplest wavelet is the Haar function

$$\psi_{\mathrm{Haar}}(\eta) = \begin{cases} 1; & 0 \le \eta < 1/2, \\ -1; & 1/2 \le \eta < 1, \\ 0; & \text{otherwise.} \end{cases}$$

The derivative of Gaussian wavelets DOG(m) are also very simple to use. We obtain them by successive derivatives of the Gauss function,

$$\psi_{\mathrm{DOG}(m)}(\eta) = \frac{(-1)^{m+1}}{\sqrt{\Gamma(m+1/2)}} \frac{\mathrm{d}^m}{\mathrm{d}\eta^m}\left(\mathrm{e}^{-\eta^2/2}\right)$$

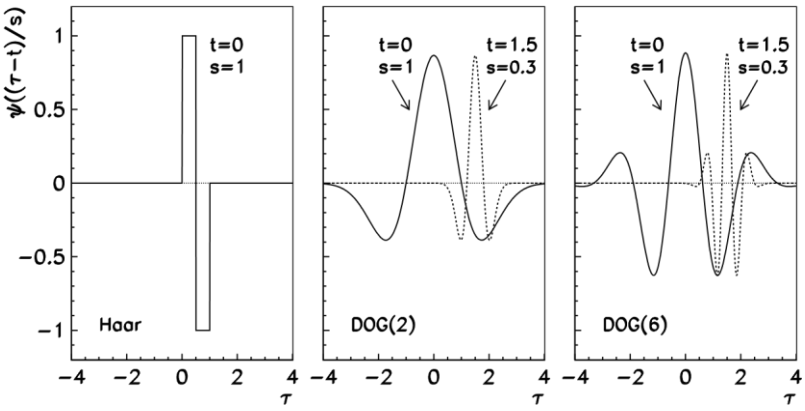


Fig. 1.13 Examples of wavelets used in the continuous wavelet transformation. By horizontal shifts and changes of scale the wavelet probes the features of the investigated signal and the times at which these features appear. [*Left*] Haar wavelet. [*Center*] The DOG(2) wavelet known as the "Mexican hat". [*Right*] The DOG(6) wavelet

Another useful wavelet is the complex Morlet wavelet

$$\psi_{\text{Morlet}}(\eta) = \pi^{-1/4}\, e^{i\omega_0\eta}\, e^{-\eta^2/2}, \quad \omega_0 \in [5, 6].$$

1.13.1 Numerical Computation of the Wavelet Transform

The continuous wavelet transform of the discrete values of the signal f_k with the chosen scale parameter s is evaluated by computing the convolution sum

$$L_\psi[f](s, t_n) = \frac{1}{\sqrt{c_\psi s}} \sum_{k=0}^{N-1} f_k \psi^*\left(\frac{(k-n)\Delta t}{s}\right), \quad n = 0, 1, \ldots, N-1$$

We take the values of s from an arbitrary set $\{s_m\}_{m=0}^{M-1}$ with some M < N. The most simple choice is $s_m = (m+1)\Delta t$ .The computation of the sum becomes unacceptably slow for large and as the time cost increases as $O(MN^2)$. Since the convolution of two functions in configuration space is equivalent to the multiplication of their Fourier transforms in the Fourier space, the CWT can be computed by using the fast Fourier transformation (FFT).

Wavelets Given as Continuous Functions

The procedure is simple when wavelet functions exist in closed forms and for which the analytic form of their Fourier transforms in known. First we use the FFT to compute the Fourier trans of the signal f which has been sampled at N points with uniform spacings

$$F_k = \mathcal{F}_N[f]_k = \frac{1}{N}\sum_{n=0}^{N-1} f_n \, \mathrm{e}^{-\mathrm{i}\,2\pi kn/N}$$

If the wavelet is given by the function ψ(t/s) in configuration space, its correlate in Fourier space (in the continuous limit) is the function ψ(sω). For example, the family of wavelets corresponds to the family of transforms

$$\hat{\psi}_{\mathrm{DOG}(m)}(s\omega) = \frac{-\mathrm{i}^m}{\sqrt{\Gamma(m+1/2)}}(s\omega)^m \, \mathrm{e}^{-(s\omega)^2/2}$$

The wavelet transform is then evaluated by using the inverse FFT to compute the sum

$$L_\psi[f](s,t_n) = \frac{1}{\sqrt{c_\psi s}}\sum_{k=0}^{N-1} F_k \hat{\psi}^*(s\omega_k)\, \mathrm{e}^{\mathrm{i}\,\omega_k n \Delta t}$$

where

$$\omega_k = \begin{cases} 2\pi k/(N\Delta t); & k \le N/2 \\ -2\pi k/(N\Delta t); & k > N/2. \end{cases}$$

The numerical cost of this procedure is O(MN logN).

Wavelets Given at Discrete Points

If the wavelet is not specified as an analytic function or if its Fourier representation is not known, we need to compute both the discrete Fourier transform of the sampled signal and the discrete Fourier transform of the wavelet at scale s, i.e.

$$Y_k = \mathcal{F}_N[\psi_s]_k = \frac{1}{N}\sum_{n=0}^{N-1} \psi^*(-n\Delta t/s)\, \mathrm{e}^{-\mathrm{i}\,2\pi kn/N}$$

The convolution is at the heart of this procedure and the way the sampling is done requires some attention. For even wavelets it makes sense to adopt the sampling

which preserves the symmetry of the wavelet about its origin. The wavelet and the signal are sampled as shown in Fig. 1.14.

We obtain the wavelet transform by multiplying the arrays $F_k \equiv F_N[f]_k$ and $Y_k \equiv F_N[\psi_s]_k$ component-wise, diving the result by $c_\psi s$, and computing the inverse Fourier transform of the product array W_k:

$$L_\psi[f](s) = N\mathcal{F}_N^{-1}[W], \qquad W_k \equiv \frac{1}{\sqrt{c_\psi s}} F_k Y_k$$

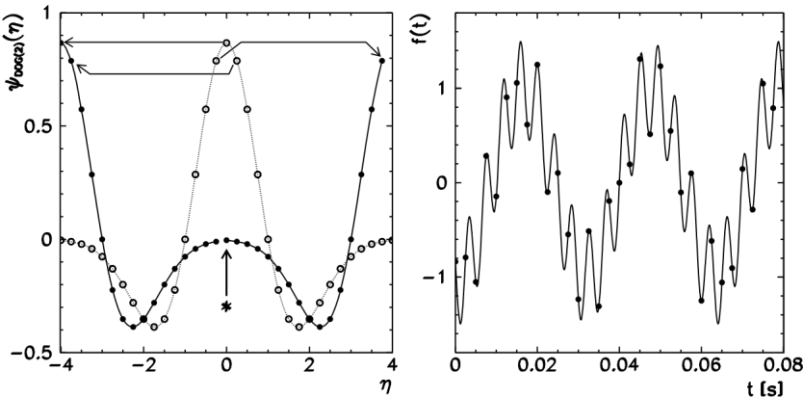


Fig1.14 Sampling at N = 32 points for the computation of the continuous wavelet transform. [Left] Periodic sampling of the wavelet in its natural (dimensionless) scale at the maximum scaling parameter s. The wavelet is sampled at N points on [ηmin,ηmax] = [−4,4]. When we wish to. reduce the parameter[Right] Sampling of the signal ats, we insert zeros at the location marked by the thick arrow and the symbolN points of the physical (time) scale or the interval $[0, N\Delta t]$. The relation between the dimensionless variable η and the physical time t is $\eta = t(\eta_{\max} - \eta_{\min})/(N\Delta t)$.

The inverse procedure is at hand: we reconstruct the original signal from the continuous wavelet transform by deconvolution, i.e. by dividing the Fourier representation of the transform by the Fourier representation of the wavelet. We form the arrays $L_k \equiv F_N[L_\psi[f](s)]_k$ and $Y_k \equiv F_N[\psi_s]_k$, divide them, multiply them by $\sqrt{c_\psi s}$, and compute the inverse Fourier transform of the quotient array:

$$f = N^{-1}\mathcal{F}_N^{-1}[F], \quad F_k \equiv \sqrt{c_\psi s}(L_k/Y_k).$$

1.13.2 Discrete Wavelet Transform

Even though we have sampled the signal and the wavelet at discrete points only, the transform described above can still be considered continuous, since the parameter s and the time axis span all M × N values. The continuous wavelet transform of a function sampled at N points has M × N values and is therefore highly redundant. In contrast, the wavelet transform that represents the signal uniquely at just ≈ N points, is known as the *discrete wavelet transform* (DWT). It lies at the heart of modern data (de)compression algorithms (an example with the CWT is given in Problems).

Problem 1: Fourier Spectrum of Signals

The discrete Fourier transformation (DFT) is the fundamental tool of signal analysis. First we confirm the formulas for known pairs of periodic signal $f = \{f_j\}_{j=0}^{N-1}$ and their transforms $F = \{F_k\}_{k=0}^{N-1} = \mathcal{F}_N[f]$, for example,

$$f_j = e^{iaj/N} \quad \Leftrightarrow \quad F_k = \frac{1}{N}\{(e^{ia} - 1)/(e^{i(a-2k\pi)/N} - 1)\},$$
$$\mathrm{Re}\, f_j, \mathrm{Im}\, f_j \sim \mathcal{N}(0,1) \quad \Leftrightarrow \quad \mathrm{Re}\, F_k, \mathrm{Im}\, F_k \sim \mathcal{N}(0,N),$$

where $N(\mu,\sigma^2)$ is the normal distribution with average μ and variance σ^2.

Question:

Compute the Fourier transforms of simple signals, in which several frequencies are represented. Compare the results in the case when the sample is periodic on the interval (the chosen frequencies are integer multiples of the fundamental frequency), to those cases when it is not. Observe the effect of aliasing for a sample that includes frequencies above the critical value. Show that for real signals:

$$F_{N-k} = F^*_k \text{ and that } F_N \text{ and } F_N^{-1} \text{ are truly inverse operations.}$$

Perform a Fourier analysis of the sound generated by tapping on a rectangular wood-box resonator, creating transient acoustic waves in its interior. Plot the power spectral density and identify the eigenmodes of the resonator from its most prominent peaks. Analyze the sound of the tuning fork attached to the resonator of a guitar.find out what happens when the sampling frequency is reduced, then try to confirm this by Fourier analysis.

Solution

For this we define our desired signals and variables:

```
[2]  import numpy as np
     import matplotlib.pyplot as plt

     # Define sampling parameters
     fs = 1000  # Sampling frequency (Hz)
     T = 1      # Duration (seconds)
     t = np.linspace(0, T, fs*T, endpoint=False)  # Time array
```

Create simple signals with multiple frequencies

```
[3]  # Case 1: Frequencies are integer multiples (harmonics)
     f1 = 50    # Fundamental frequency (Hz)
     f2 = 100   # 2nd harmonic
     f3 = 150   # 3rd harmonic
     signal_harmonic = np.sin(2*np.pi*f1*t) + 0.5*np.sin(2*np.pi*f2*t) + 0.3*np.sin(2*np.pi*f3*t)

     # Case 2: Frequencies are not integer multiples
     f4 = 60    # Not a harmonic of f1
     f5 = 125   # Not a harmonic of f1
     signal_nonharmonic = np.sin(2*np.pi*f1*t) + 0.5*np.sin(2*np.pi*f4*t) + 0.3*np.sin(2*np.pi*f5*t)
```

Then we compute discrete Fourier transform:

Compute DFT using NumPy's FFT function

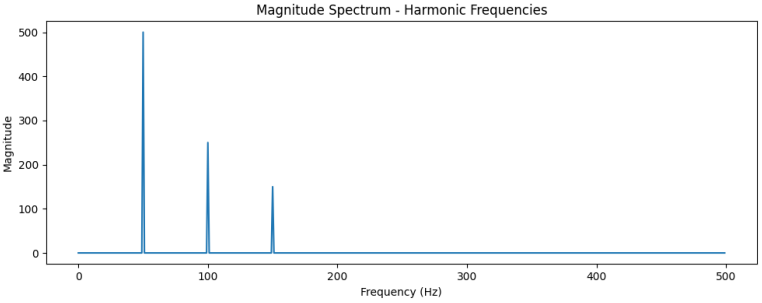
```
[4]  N = len(t)
     freqs = np.fft.fftfreq(N, d=1/fs)

     # FFT for harmonic signal
     fft_harmonic = np.fft.fft(signal_harmonic)
     # FFT for non-harmonic signal
     fft_nonharmonic = np.fft.fft(signal_nonharmonic)
```

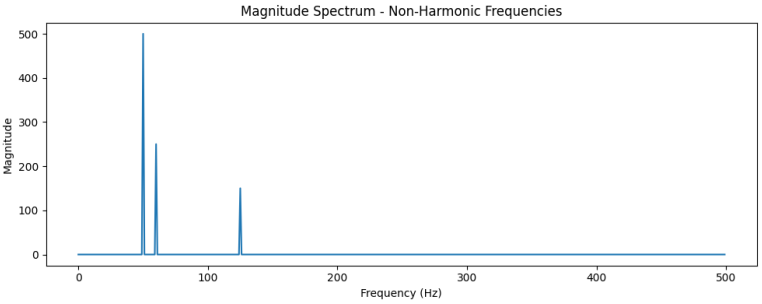
We plot the the magnitude spectrum:

Plot the magnitude spectrum

```
fig, axs = plt.subplots(2, 1, figsize=(10,8))

# Plot harmonic signal
axs[0].plot(freqs[:N//2], np.abs(fft_harmonic[:N//2]))
axs[0].set_title('Magnitude Spectrum - Harmonic Frequencies')
axs[0].set_xlabel('Frequency (Hz)')
axs[0].set_ylabel('Magnitude')

# Plot non-harmonic signal
axs[1].plot(freqs[:N//2], np.abs(fft_nonharmonic[:N//2]))
axs[1].set_title('Magnitude Spectrum - Non-Harmonic Frequencies')
axs[1].set_xlabel('Frequency (Hz)')
axs[1].set_ylabel('Magnitude')

plt.tight_layout()
plt.show()
```

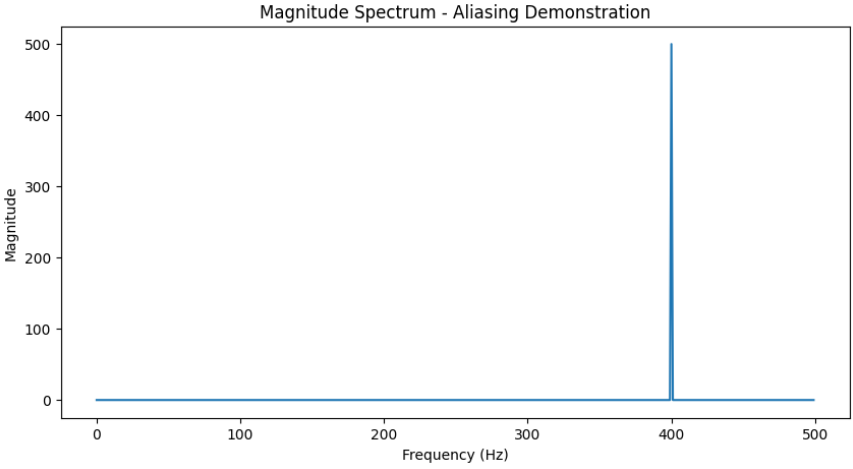
And we have:





Demonstrate Aliasing

```
[6]  # Frequency above Nyquist (e.g., 600 Hz, Nyquist is fs/2 = 500 Hz)
     alias_freq = 600
     signal_alias = np.sin(2*np.pi*alias_freq*t)
     fft_alias = np.fft.fft(signal_alias)

     plt.figure(figsize=(10,5))
     plt.plot(freqs[:N//2], np.abs(fft_alias[:N//2]))
     plt.title('Magnitude Spectrum - Aliasing Demonstration')
     plt.xlabel('Frequency (Hz)')
     plt.ylabel('Magnitude')
     plt.show()
```



Now we check the DFT properties:

Verify DFT properties for real signals

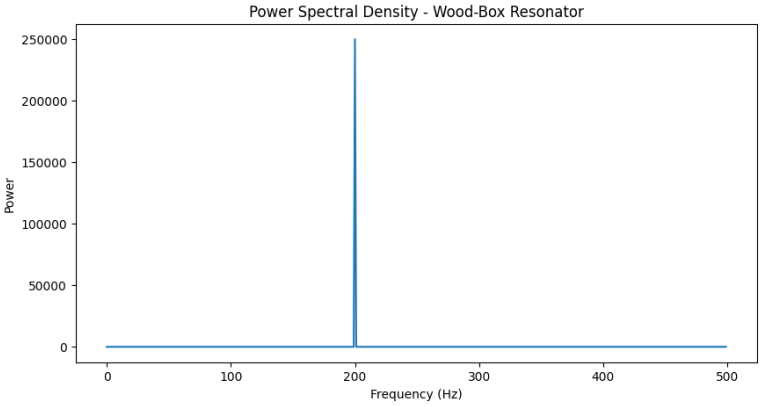
```
[8]  # Check F[N-k] = F_k^*

     # Using the harmonic signal as an example
     F = fft_harmonic
     N_freq = len(F)
     # Compute F[N-k] and F_k^*
     F_Nk = F[(N_freq - np.arange(N_freq)) % N_freq]  # Use modulo to wrap around indices
     F_k_star = np.conj(F)

     # Check if they are equal
     print("F[N-k] == F_k^*:", np.allclose(F_Nk, F_k_star))

     # Check that DFT and inverse DFT are inverses
     signal_recovered = np.fft.ifft(fft_harmonic)
     # Check if original and recovered signals are equal
     print("Signal recovered correctly:", np.allclose(signal_harmonic, signal_recovered, atol=1e-10))
```

```
F[N-k] == F_k^*: True
Signal recovered correctly: True
```

The result is fine, so we do the Fourier analysis of woodbox resonator sound:

Fourier analysis of wood-box resonator sound

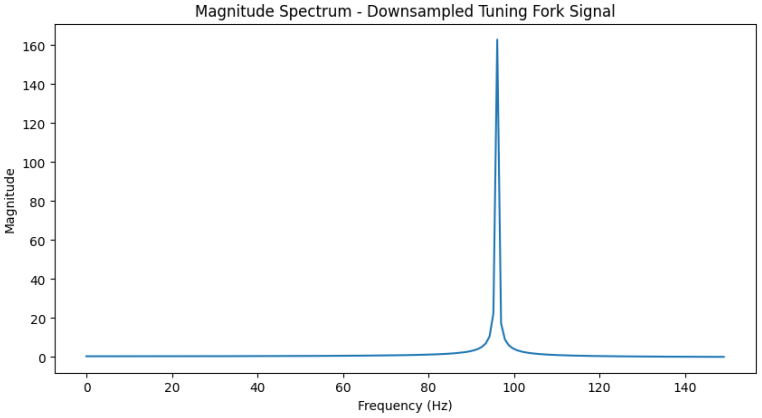
```
[9]  # For demonstration, simulate a signal with prominent eigenmodes
     # Assume eigenmodes at 200 Hz, 400 Hz, and 600 Hz
     eigenmode_freqs = [200, 400, 600]
     signal_resonator = np.sum([np.sin(2*np.pi*f*t) for f in eigenmode_freqs], axis=0)
     fft_resonator = np.fft.fft(signal_resonator)
     power_spectral_density = np.abs(fft_resonator)**2

     plt.figure(figsize=(10,5))
     plt.plot(freqs[:N//2], power_spectral_density[:N//2])
     plt.title('Power Spectral Density - Wood-Box Resonator')
     plt.xlabel('Frequency (Hz)')
     plt.ylabel('Power')
     plt.show()

     # Identify eigenmodes from peaks
     peaks = freqs[:N//2][np.argmax(power_spectral_density[:N//2])]
     print("Identified eigenmodes at:", peaks)
```



Now, we try to tuning fork signal:

Analyze tuning fork sound with reduced sampling frequency

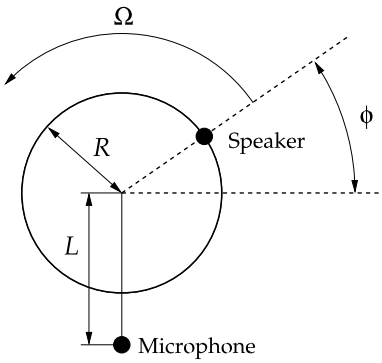
```
[10] # Original signal
     fork_freq = 440  # Tuning fork frequency (Hz)
     signal_fork = np.sin(2*np.pi*fork_freq*t)

     # Downsample to a lower sampling frequency, e.g., fs_down = 300 Hz
     fs_down = 300
     # Downsample by selecting every (fs/fs_down)th sample
     downsample_factor = int(fs / fs_down)
     signal_downsampled = signal_fork[::downsample_factor]
     t_down = t[::downsample_factor]

     # Perform FFT on downsampled signal
     fft_downsampled = np.fft.fft(signal_downsampled)
     freqs_down = np.fft.fftfreq(len(t_down), d=1/fs_down)

     plt.figure(figsize=(10,5))
     plt.plot(freqs_down[:len(freqs_down)//2], np.abs(fft_downsampled[:len(freqs_down)//2]))
     plt.title('Magnitude Spectrum - Downsampled Tuning Fork Signal')
     plt.xlabel('Frequency (Hz)')
     plt.ylabel('Magnitude')
     plt.show()

     # Observe the effect of reduced sampling frequency on the spectrum
     print("Original frequency:", fork_freq, "Hz")
     print("Downsampled frequency may show aliasing effects.")
```



Original frequency: 440 Hz

Problem 2: Fourier Analysis of the Doppler Effect

Here we discuss the experiment set up in Salzburg, the birthplace of C. Doppler (1805–1853). A speaker is attached to a rotating wheel with radius R, and there is a microphone in the plane of rotation at a distance L from the center of the wheel, as shown in the figure. The rotation angular frequency is $\Omega = v/R$, where v is the tangential velocity. The radius vector r(t) describes the relative position of the microphone with respect to the speaker. The speaker periodically approaches and recedes from the microphone (period $2\pi/\Omega$).



The microphone is used to measure the pressure differences δp(t) given by the formula

$$\delta p(t) = A\frac{\cos[\omega(t - x(\Omega t)/c)]}{x(\Omega t)}$$

Here A is the amplitude, ω is the angular frequency of the emitted sound and c is its velocity. The function x(t) is defined implicitly as the positive solution of the equation $l^2 + r^2 + 2lr\sin(t - x) = x^2$, where $l = \Omega L/c$ and $r = R/c$.

Question:

Let v = 100m/s and A = 1. Sample the signal δp(t) at time intervals $\Delta t = 1/(\Omega N)$ over two periods. This gives you the discrete signal $f_j = \delta p(j\,\Delta t)$

at j = 0,1,...,N − 1. Compute the DFT and the Hilbert transform of the discrete signal in the limits L << R, L >> R, and L ≈ R, and analyze its single-sided power spectral density (PSD), its instantaneous amplitude, and instantaneous frequency. An example of the signal and its spectral density is shown in Fig. 1.15. Compare the results to the classical Doppler prediction.

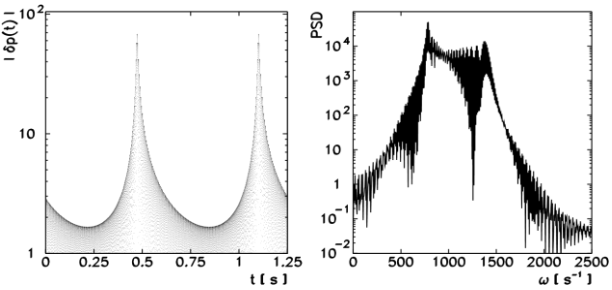


Fig. 1.15 [Left] An example of the signal at v = 100m/s, R = 10m, L = 10.5m, ω = 1000/s, and c = 340 m/s. [Right] The Fourier power spectral density. The peaks in the spectrum occur approximately at ω/(1 + v/c) = 773Hz and ω/(1 − v/c) = 1417Hz

Analyze the PSD of the signal at smaller time windows which open just slightly ahead of the time when the speaker is nearest to the microphone, and close just after that.

Solution

First, we deine variables and equations:

Define Variables

```
[ ]  import numpy as np
     from scipy.optimize import fsolve
     from scipy.fft import fft, fftfreq
     from scipy.signal import hilbert
     import matplotlib.pyplot as plt

     # Define parameters
     v = 100.0  # m/s
     A = 1.0
     R = 10.0  # m
     L = 10.5  # m
     omega = 1000.0  # rad/s
     c = 340.0  # m/s
```

Define equations

```
[ ]  # Compute Omega, I, r
     Omega = v / L
     I = Omega * L / c
     r = R / c

     # Define the implicit equation for x(t)
     def implicit_eq(x, t):
         return x**2 + r**2 + 2 * I * r * np.sin(t - x) - x
```

Then we solve it for x and do the Hilber and DFT:

solve for x

```
[ ]  # Number of samples over two periods
     N = 1024  # You can adjust this value for better resolution
     t_end = 2 * 2 * np.pi / Omega  # two periods
     delta_t = 1 / (Omega * N)
     t = np.arange(0, t_end, delta_t)

     # Solve for x(t) at each t_j
     x = np.zeros_like(t)
     for j in range(len(t)):
         x[j] = fsolve(implicit_eq, x0=0.0, args=(t[j]))

     # Compute delta_p(t)
     delta_p = A * np.cos(omega * (t - x / c)) / x
```

DFT and HIlbert transform

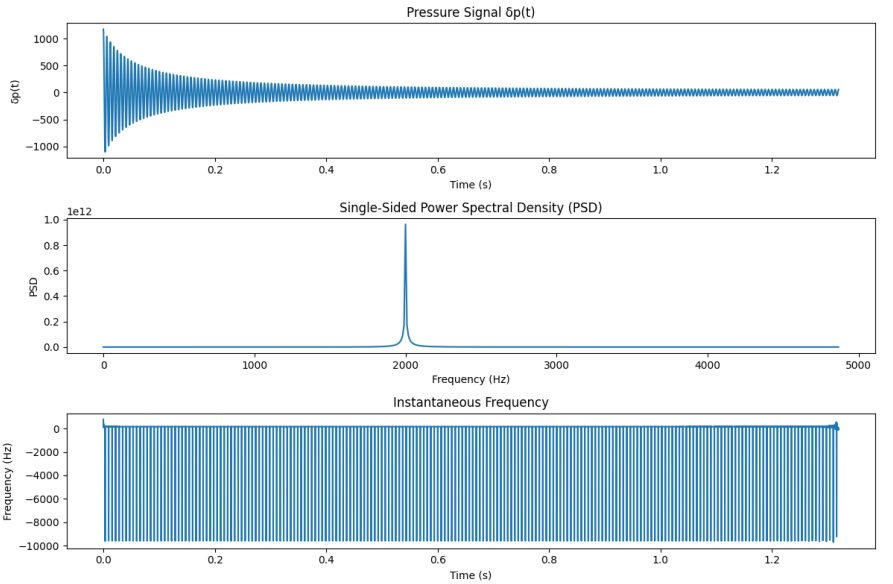
```
[ ]  # Compute DFT
     F = fft(delta_p)
     freq = fftfreq(N, delta_t)
     psd = np.abs(F) ** 2

     # Compute Hilbert transform
     analytic_signal = hilbert(delta_p)
     instantaneous_amplitude = np.abs(analytic_signal)
     instantaneous_phase = np.angle(analytic_signal)
     instantaneous_frequency = np.diff(instantaneous_phase) / (2 * np.pi * delta_t)
```

Here's the classical prediction:

Classical Doppler prediction

```
# At closest approach, relative velocity v_rel = v
f_emitted = omega / (2 * np.pi)
f_prime = f_emitted * c / (c - v)
```

And now we plot the results:



Problem 3: Use of Laplace Transformation and Its Inverse

The time dependence of the charge on a capacitor connected in series to a electric generator, a resistor, and a coil, is described by the differential equation

$$L\frac{\mathrm{d}^2Q}{\mathrm{d}t^2}+R\frac{\mathrm{d}Q}{\mathrm{d}t}+\frac{1}{C}Q=U_{\mathrm{g}}(t)$$

or, in dimensionless form, at given R, C and L, with $R\sqrt{\frac{C}{L}}=1$,

$$\ddot{x}+\dot{x}+x=g(t).$$

With initial conditions $x(0)=\dot{x}(0)=0$ the source generates a voltage pulse of the form

$$g(t)=1-\theta(t-\pi)=\begin{cases}1; & 0\le t<\pi,\\ 0; & t\ge\pi.\end{cases}$$

By taking the initial conditions into account, the Laplace transform of the differential equation is

$$s^2X(s)+sX(s)+X(s)=\mathcal{L}[1]-\mathcal{L}\left[\theta(t-\pi)\right]=\left(1-\mathrm{e}^{-\pi s}\right)/s$$

or

$$X(s)\equiv\left(1-\mathrm{e}^{-\pi s}\right)H(s),\quad H(s)=\frac{1}{s(s^2+s+1)}=\frac{1}{s}-\frac{(s+\frac{1}{2})+\frac{1}{2}}{(s+\frac{1}{2})^2+\frac{3}{4}}$$

Let us denote $h(t)=L^{-1}[H]$ and remember that the inverse Laplace transformation is linear. Then by looking at Table 1.1 once more

$$x(t)=\mathcal{L}^{-1}[X]=\mathcal{L}^{-1}[H]-\mathrm{e}^{-\pi s}\mathcal{L}^{-1}[H]=h(t)-h(t-\pi)\theta(t-\pi),$$

where

$$h(t)=\mathcal{L}^{-1}[H]=1-\frac{1}{3}\mathrm{e}^{-t/2}\left(3\cos\frac{\sqrt{3}}{2}t+\sqrt{3}\sin\frac{\sqrt{3}}{2}t\right)$$

Question:

Use the inverse Laplace transformation to find the solution x(t) from the function $X(s) = (1 - e^{-\pi s})H(s)$. Since a discontinuous function g(t) appears in the time domain, the numerical inverse should be computed by dedicated algorithms specially tailored to such functions.

Solution

We define the variables and Heaviside function, then solve the equation:

Define variables

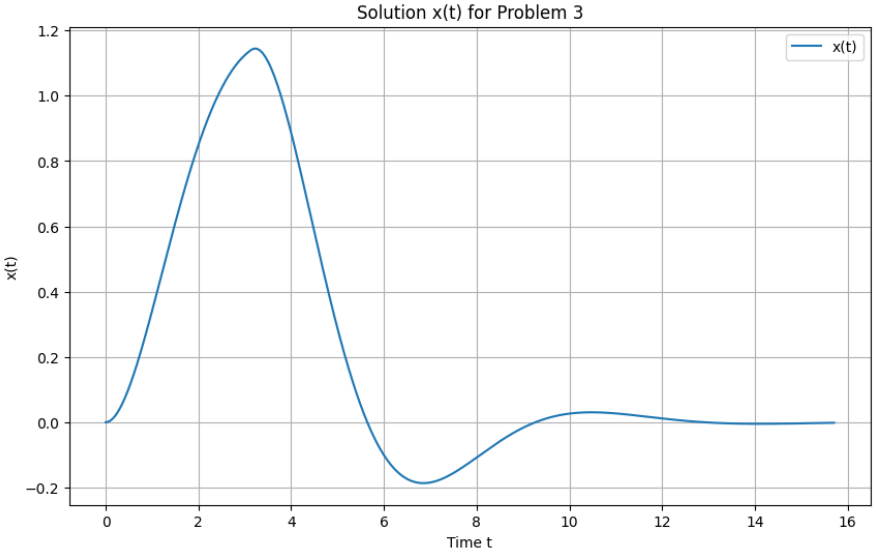
```
[3] import numpy as np
    import matplotlib.pyplot as plt

    # Define the Heaviside step function
    def heaviside(t, t0):
        return 1.0 * (t >= t0)

    # Define h(t) as given
    def h(t):
        if t < 0:
            return 0.0
        else:
            return 1 - (1/3) * np.exp(-t/2) * (3 * np.cos((np.sqrt(3)/2) * t) + np.sqrt(3) * np.sin((np.sqrt(3)/2) * t))

    # Vectorize h(t) for array input
    h_vec = np.vectorize(h)
```

Solve x(t)

```
[4] # Define x(t)
    def x(t, pi=np.pi):
        return h_vec(t) - h_vec(t - pi) * heaviside(t, pi)

    # Create time array
    t = np.linspace(0, 5 * np.pi, 1000)

    # Compute x(t)
    x_t = x(t)
```

Now we plot the results:

```
# Plot the solution
plt.figure(figsize=(10, 6))
plt.plot(t, x_t, label='x(t)')
plt.title('Solution x(t) for Problem 3')
plt.xlabel('Time t')
plt.ylabel('x(t)')
plt.legend()
plt.grid(True)
plt.show()
```



Problem 4: Use of the Wavelet Transformation

The continuous wavelet transformation is one of the basic tools in signal processing and analysis. We use it to determine the frequency components of signals and the instances in time when individual components actually occur: this is its main advantage over the usual Fourier transformation.

Question:

Use the continuous wavelet transformation on a simple periodic signal to which components with higher frequencies have been admixed at certain subinter vals, for example,

$$f(t) = \begin{cases} \sin t + 0.2 \sin 10t; & \pi \le t \le 2\pi, \\ \sin t + 0.1 \sin 20t; & 4\pi \le t \le 5\pi, \\ \sin t; & \text{otherwise} \end{cases}$$

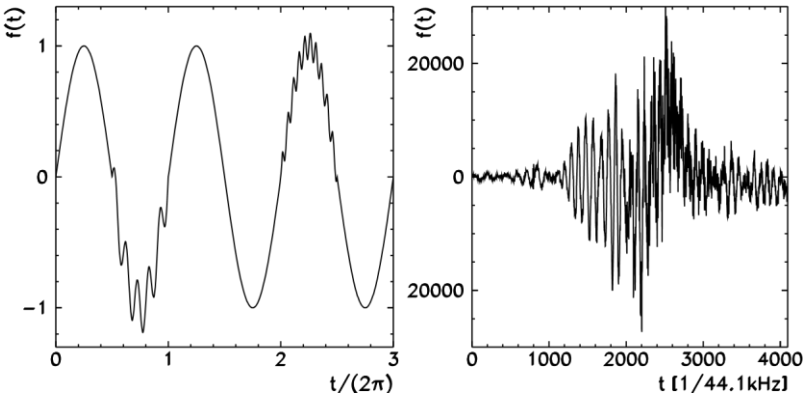


Fig. 1.16 [*Left*] The signal with three frequency components, two of which appear only occasionally. [*Right*] A recording of the intensity of sound caused by quickly waving a bamboo-stick through air

(see Fig. 1.16 (left)). You can additionally perturb the signal by mixing it with noise of various amplitudes. How do such admixtures influence the quality (the resolution capability) of the wavelet transform? Use different types of wavelet functions. What differences can you observe?

The continuous wavelet transformation also gives us some basic feeling for the properties of compression and decompression of data. Figure . 1.16 (right) shows a typical acoustic signal. By using the procedure described before, compute the continuous wavelet transform for this signal. Set to zero all components in the transform that are below a certain chosen threshold (for example, keep only 20, 10, or 5 % of the strongest components). Then perform the inverse wavelet transformation and compare the result to the original signal. How has it changed? Do you reach the same conclusion if you repeat the exercise with the Fourier transform? (Use the FFT to map the signal into Fourier space, keep 20, 10, or 5 % of the strongest components, and compute the inverse FFT.)

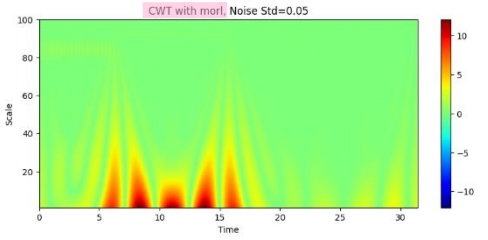
Solution

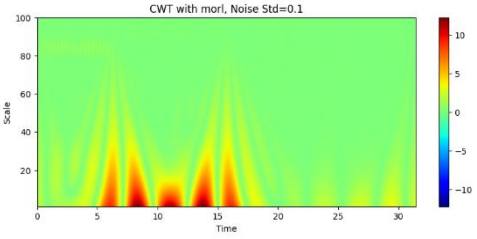
First we define our variables and functions:

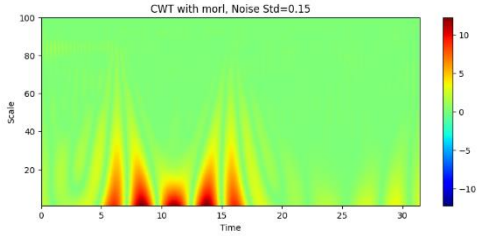
```
import numpy as np
import pywt
import matplotlib.pyplot as plt

# Define the signal function ft(t)
def ft(t):
    y = np.zeros_like(t)
    y[t < 2 * np.pi] = np.sin(t[t < 2 * np.pi]) + 0.2 * np.sin(10 * t[t < 2 * np.pi])
    mask = (t >= 2 * np.pi) & (t < 5 * np.pi)
    y[mask] = 4 * np.sin(t[mask]) + 0.1 * np.sin(20 * t[mask])
    y[t >= 5 * np.pi] = np.sin(t[t >= 5 * np.pi])
    return y

# Time array
t = np.linspace(0, 10 * np.pi, 1000)
signal = ft(t)

# Add noise with different amplitudes
noise_std = [0.05, 0.1, 0.15]
noisy_signals = [signal + np.random.normal(0, std, len(t)) for std in noise_std]
```

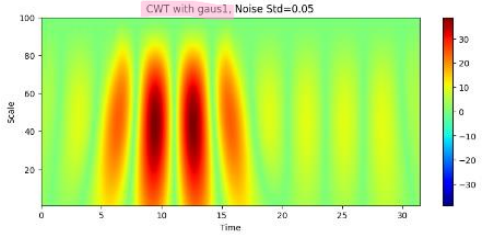
Perform CWT with different wavelets

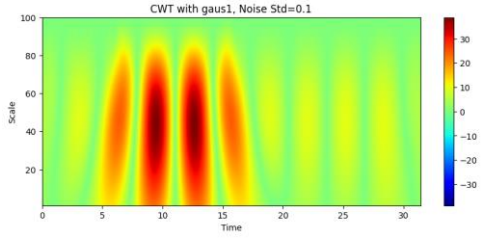
```
wavelets = ['morl', 'cmor', 'gaus1']
for wav in wavelets:
    for i, noise_sig in enumerate(noisy_signals):
        # CWT
        widths = np.arange(1, 100)
        cwtmatr, freqs = pywt.cwt(noise_sig, widths, wav)

        # Plot CWT coefficients
        plt.figure(figsize=(10, 4))
        plt.imshow(abs(cwtmatr), extent=[0, 10*np.pi, 1, 100], cmap='jet', aspect='auto', vmax=abs(cwtmatr).max(), vmin=-abs(cwtmatr).max())
        plt.title(f'CWT with {wav}, Noise Std={noise_std[i]}')
        plt.xlabel('Time')
        plt.ylabel('Scale')
        plt.colorbar()
        plt.show()
```

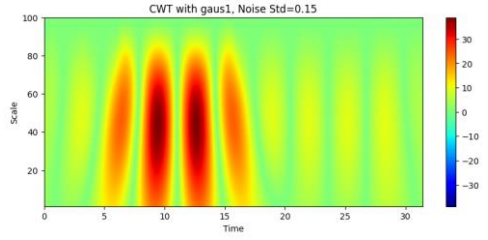


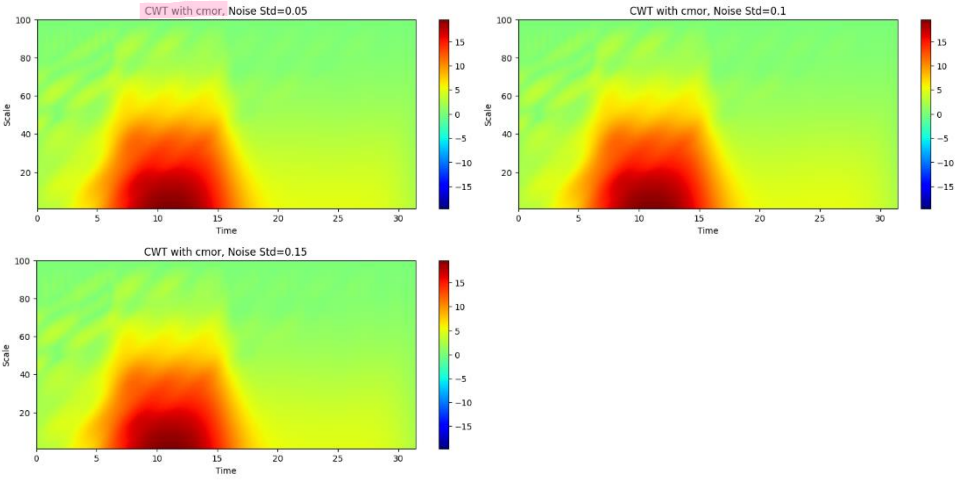










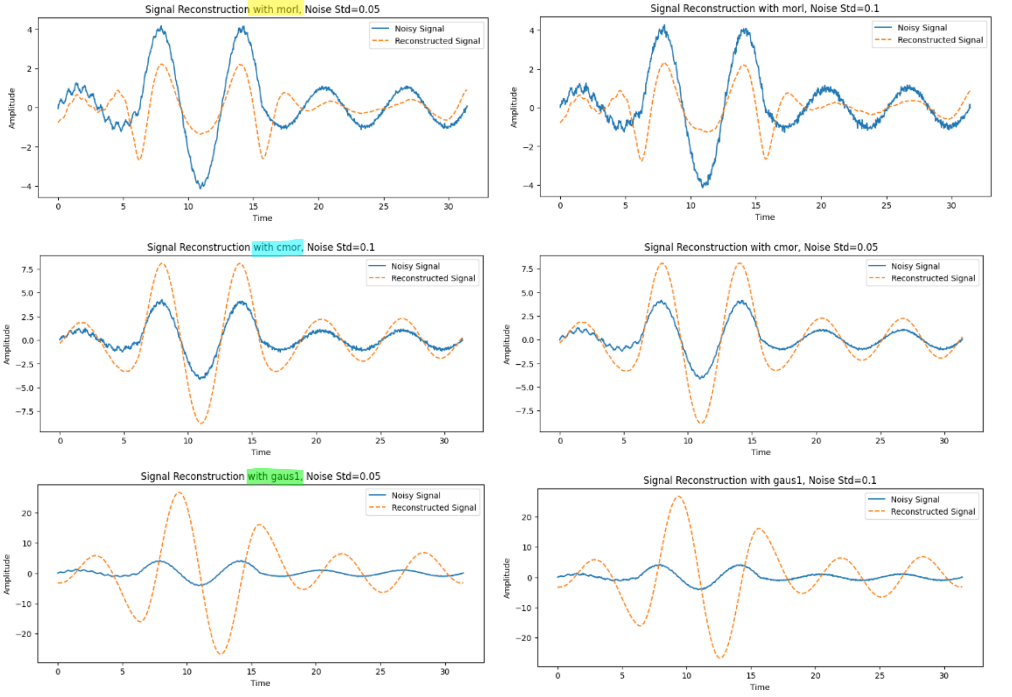


Now, we have:

Data compression via thresholding

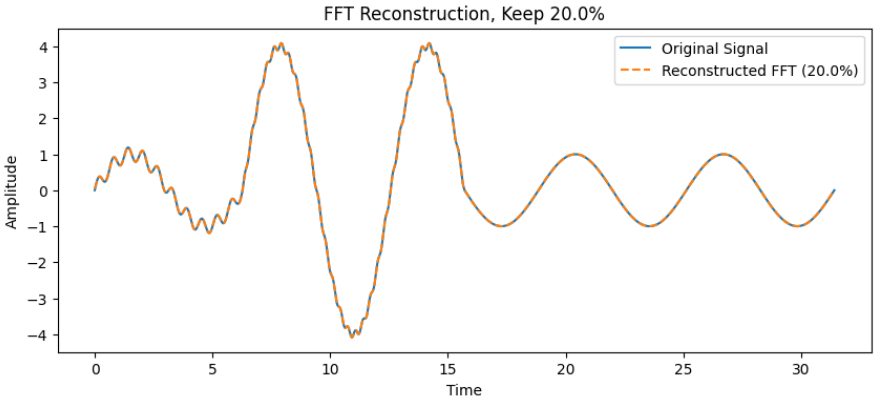
```
thresholds = [0.2, 0.1, 0.05]  # Keep 20%, 10%, 5% of coefficients
for wav in wavelets:
    for i, noise_sig in enumerate(noisy_signals):
        # CWT
        widths = np.arange(1, 100)
        cwtmatr, freqs = pywt.cwt(noise_sig, widths, wav)

        # Plot CWT coefficients
        plt.figure(figsize=(10, 4))
        plt.imshow(abs(cwtmatr), extent=[0, 10*np.pi, 1, 100], cmap='jet', aspect='auto', vmax=abs(cwtmatr).max(), vmin=-abs(cwtmatr).max())
        plt.title(f'CWT with {wav}, Noise Std={noise_std[i]}')
        plt.xlabel('Time')
        plt.ylabel('Scale')
        plt.colorbar()
        plt.show()

        # Reconstruct the signal manually
        # This is a simplified reconstruction and may not be exact
        signal_recon = np.sum(cwtmatr, axis=0) / len(widths)

        # Plot the original and reconstructed signals
        plt.figure(figsize=(10, 4))
        plt.plot(t, noise_sig, label='Noisy Signal')
        plt.plot(t, signal_recon, label='Reconstructed Signal', linestyle='--')
        plt.title(f'Signal Reconstruction with {wav}, Noise Std={noise_std[i]}')
        plt.xlabel('Time')
        plt.ylabel('Amplitude')
        plt.legend()
        plt.show()
```

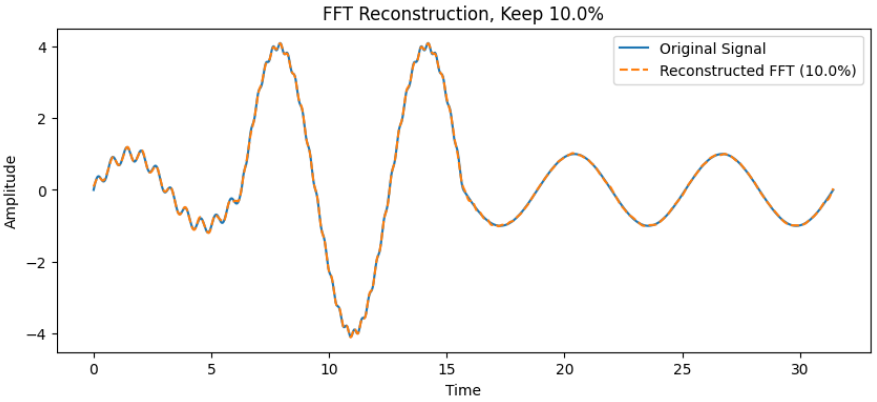
Here’s the output:

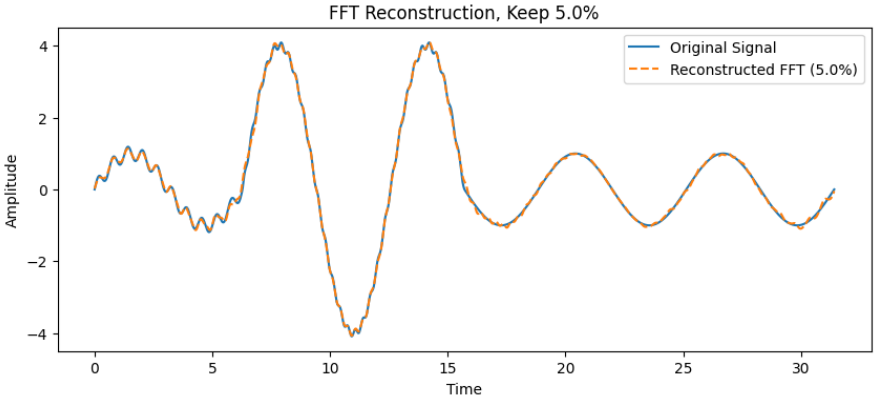


Fourier Transform Compression

```
for thresh in thresholds:
    fft_coef = np.fft.fft(signal)
    # Keep top thresh% coefficients
    fft_sorted_idx = np.argsort(abs(fft_coef))[::-1]
    keep = int(thresh * len(fft_coef))
    fft_threshold = np.zeros_like(fft_coef)
    fft_threshold[fft_sorted_idx[:keep]] = fft_coef[fft_sorted_idx[:keep]]
    # Inverse FFT
    signal_recon_fft = np.fft.ifft(fft_threshold)
    # Plot
    plt.figure(figsize=(10, 4))
    plt.plot(t, signal, label='Original Signal')
    plt.plot(t, signal_recon_fft.real, label=f'Reconstructed FFT ({thresh*100}%)', linestyle='--')
    plt.title(f'FFT Reconstruction, Keep {thresh*100}%')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()
    plt.show()
```


FFT Reconstruction, Keep 20.0%
Original Signal
Reconstructed FFT (20.0%)
Amplitude
Time


FFT Reconstruction, Keep 10.0%
Original Signal
Reconstructed FFT (10.0%)
Amplitude
Time


FFT Reconstruction, Keep 5.0%
Original Signal
Reconstructed FFT (5.0%)
Amplitude
Time

Conclusion

In conclusion, this report has explored various transformations of functions and signals, focusing on Fourier, Laplace, and wavelet transformations, as well as their applications in signal processing and analysis. The Fourier transformation was utilized to analyze frequency components and power spectral densities, demonstrating its effectiveness in identifying signal characteristics and eigenmodes. The Laplace transformation proved invaluable in solving differential equations, particularly in electrical circuits, by transforming complex time-domain problems into simpler algebraic equations in the frequency domain.

The wavelet transformation offered a unique advantage by providing both frequency and temporal information, making it particularly useful for analyzing signals with transient components and for data compression. Through practical examples and numerical computations, the report highlighted the strengths and limitations of each transformation method. Overall, these tools are essential for understanding and manipulating signals in various scientific and engineering contexts, enabling precise analysis and effective problem-solving.