3.3 Finite Signals

In practice we cannot store an infinite number of samples $\{f(n)\}_{n\in\mathbb{Z}}$ of a signal f; instead we can only keep a finite number of samples, say $\{f(n)\}_{0\le n<N}$. We thus must amend our definition of the Fourier transform as well as convolution, which will lead to the Discrete Fourier Transform (DFT) and circular convolution. One thing that will arise is that regardless of whether the original signal f is periodic, we will be forced to think of the finite sampling $(f(n))_{0\le n<N}$ as a discrete periodic signal with period N. This will lead to border effects which must be accounted for. However, the circular convolution theorem and Fast Fourier Transform will allow for fast computations of convolution operators.

Let $x, y \in \mathbb{C}^N$, which are vectors of length N, e.g., N samples of a signal f such that $x[n] = f(n)$ for $0 \le n < N$. The inner product between x and y is:

$$\langle x, y\rangle = \sum_{n=0}^{N-1} x[n]y^*[n]$$

We must replace the sinusoids $e^{it\omega}$ ($t \in \mathbb{R}$) and $e^{in\omega}$ ($n \in \mathbb{Z}$), which are continuous in the frequency variable ω, with discrete counterparts. The variable ω is replaced with an index k with $0 \le k < N$:

$$e_k[n] = \exp\left(\frac{2\pi ikn}{N}\right), \quad 0 \le n, k < N \tag{11}$$

The *Discrete Fourier Transform (DFT)* of x is defined as:

$$\widehat{x}[k] = \langle x, e_k\rangle = \sum_{n=0}^{N-1} x[n] \exp\left(-\frac{2\pi ikn}{N}\right), \quad 0 \le k < N$$

The following theorem shows that the set of vectors $\{e_k\}_{0\le k<N}$ is an orthogonal basis for $\mathbb{C}^N$.

Theorem 20. *The family of vectors $\{e_k\}_{0\le k<N}$ as defined in (11) is orthogonal basis for $\mathbb{C}^N$.*

Thus the DFT is a bijection and hence invertible. Since $\|e_k\|^2 = N$ for all k, it follows from Theorem 20 that x can be represented in the orthogonal basis $\{e_k\}_{0\le k<N}$ as:

$$x[n] = \sum_{k=0}^{N-1} \frac{\langle x, e_k\rangle}{\|e_k\|^2} e_k[n] = \frac{1}{N}\sum_{k=0}^{N-1} \widehat{x}[k] \exp\left(\frac{2\pi ikn}{N}\right)$$

This gives the inverse DFT.

We would like a convolution theorem for the DFT similar to the convolution theorem for $\mathbf{L}^1(\mathbb{R})$ functions and $\boldsymbol{\ell}^1(\mathbb{R})$ sequences. We define convolution of $x, y \in \mathbb{C}^N$ by first extending them to signals $\widetilde{x}, \widetilde{y} \in \boldsymbol{\ell}^1(\mathbb{R})$ defined as:

$$\widetilde{x}[n] = \begin{cases} x[n] & 0 \leq n < N \\ 0 & n < 0 \text{ or } n \geq N \end{cases}$$

We then define the convolution of x and y as the convolution of $\widetilde{x}$ and $\widetilde{y}$, and keep only the values with a chance of being nonzero:

$$x * y[n] = \sum_{m \in \mathbb{Z}} \widetilde{x}[m]\widetilde{y}[n-m], \quad 0 \leq n < 2N-1$$

In practice, there will be many times when you want to compute such convolutions. Indeed, if x and y are discrete samplings of non-periodic signals f and g, respectively, computing $x * y$ will give a discrete approximation for $f * g$. However, for computational reasons, we will often not want to compute discrete convolutions directly (more on this in a bit). Indeed, it will be better to compute such convolutions "in frequency," which will require a convolution theorem for the DFT. However, the discrete sinusoids $\{e_k\}_{0 \leq k < N}$ are not eigenvectors of discrete convolution operators $Lx = x * h$. The vectors e_k are periodic, but the standard convolution is not; indeed, it extends the vectors x, y to a twice longer vector $x * y$. We therefore define a periodic version of convolution, which is called *circular convolution*.

To define circular convolution, rather than extending x and y with zeros, we will extend them with a periodization over N samples:

$$\overline{x}[n] = x[n \mod N], \quad n \in \mathbb{Z}$$

The circular convolution is defined as:

$$x \circledast y[n] = \sum_{m=0}^{N-1} x[m]y[n-m]$$

Note that $x \circledast y \in \mathbb{C}^N$. One then has the following circular convolution theorem:

Theorem 21. *If $x, y \in \mathbb{C}^N$, then*

$$\widehat{x \circledast y}[k] = \widehat{x}[k]\widehat{y}[k]$$

The key to this theorem, and the DFT more generally, is that since the discrete sinusoids e_k are periodic vectors with period N, the DFT treats all vectors $x \in \mathbb{C}^N$ as periodic vectors with period N. This manifests in the convolution theorem by requiring us to utilize circular convolutions. However, it also means that when computing DFTs, we need always need to think of x as a periodic vector with period N. In particular, seemingly "smooth" vectors such as $x[n] = n$ actually have very sharp transitions once made periodic, since $x[N-1] = N-1$ and $x[N] = x[0] = 0$; see Figure 5.

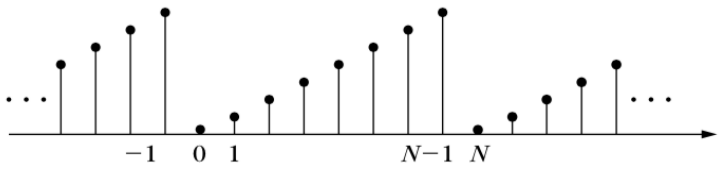

Figure 5: Periodization of the ramp vector $x[n] = n$ on $\mathbb{R}^N$. Taken from Figure 3.3 of *A Wavelet Tour of Signal Processing*.

The circular convolution theorem will be very important for opening up fast algorithms for computing $x \circledast y$. This will be made possible by the *Fast Fourier Transform (FFT)*, which we now describe. To motivate the algorithm, recall the DFT:

$$\widehat{x}[k] = \sum_{n=0}^{N-1} x[n] \exp\left(-\frac{2\pi i k n}{N}\right), \quad 0 \le k < N$$

and observe that it requires N^2 (complex) multiplications and additions (N for each k). The FFT algorithm reduces this to $O(N \log_2 N)$.

The FFT algorithm works through a divide and conquer approach; in these notes I will describe the radix-2 decimation in time (DIT) algorithm. This is a recursive algorithm. Given x we divide the DFT summation into two sums, one for the even indices of x and one for the odd indices of x:

$$\begin{aligned}\widehat{x}[k] &= \sum_{n=0}^{N/2-1} x[2n] \exp\left(\frac{-2\pi i k(2n)}{N}\right) + \sum_{n=0}^{N/2-1} x[2n+1] \exp\left(\frac{-2\pi i k(2n+1)}{N}\right) \\ &= \sum_{n=0}^{N/2-1} x[2n] \exp\left(\frac{-2\pi i k n}{N/2}\right) + e^{-2\pi i k/N} \sum_{n=0}^{N/2-1} x[2n+1] \exp\left(\frac{-2\pi i k n}{N/2}\right)\end{aligned}$$

The second line looks like the sum of two DFTs of length $N/2$ signals. Indeed, define $x_{\mathrm{e}}, x_{\mathrm{o}} \in \mathbb{R}^{N/2}$ as:

$$\begin{aligned} x_{\mathrm{e}}[n] &= x[2n], \quad 0 \le n < N/2 \\ x_{\mathrm{o}}[n] &= x[2n+1], \quad 0 \le n < N/2 \end{aligned}$$

and notice that we have

$$\begin{aligned} \widehat{x}_{\mathrm{e}}[k] &= \sum_{n=0}^{N/2-1} x[2n] \exp\left(\frac{-2\pi i k n}{N/2}\right), \quad 0 \le k < N/2 \\ \widehat{x}_{\mathrm{o}}[k] &= \sum_{n=0}^{N/2-1} x[2n+1] \exp\left(\frac{-2\pi i k n}{N/2}\right), \quad 0 \le k < N/2 \end{aligned}$$

This allows us to recover $\widehat{x}[k]$ for $0 \le k < N/2$ as:

$$\widehat{x}[k] = \widehat{x}_{\mathrm{e}}[k] + e^{-2\pi i k/N} \widehat{x}_{\mathrm{o}}[k], \quad 0 \le k < N/2 \tag{12}$$

For the frequencies $N/2 \le k < N$, we use the fact that the DFT is periodic and observe that

$$\widehat{x}_{\mathrm{e}}[k + N/2] = \widehat{x}_{\mathrm{e}}[k] \quad \text{and} \quad \widehat{x}_{\mathrm{o}}[k + N/2] = \widehat{x}_{\mathrm{o}}[k]$$

We thus obtain:

$$\begin{aligned} 0 \le k < N/2, \quad \widehat{x}[k + N/2] &= \widehat{x}_{\mathrm{e}}[k] + e^{-2\pi i (k+N/2)/N} \widehat{x}_{\mathrm{o}}[k] \\ &= \widehat{x}_{\mathrm{e}}[k] - e^{-2\pi i k/N} \widehat{x}_{\mathrm{o}}[k] \end{aligned} \tag{13}$$

Putting together (12) and (13) we obtain $\widehat{x}[k]$ for all $0 \le k < N$. Notice that already we have reduced computations. Indeed, the one step algorithm proceeds by first dividing x into even and odd indices signals, computing the two length $N/2$ DFTs, and recombining as above. The two length N DFTs cost $2(N/2)^2 = N^2/4$ multiplications and additions, and the combination costs N additions and multiplications. Thus we have replaced N^2 complex multiplications and additions with $N^2/2 + N$ complex multiplications and additions, which is already better for $N \ge 3$.

Now for simplicity, suppose $N = 2^p$. The $O(N \log_2 N)$ FFT algorithm is obtained by recursively subdividing the original signal x, according to the same procedure as outlined above into "even" and "odd" components, until

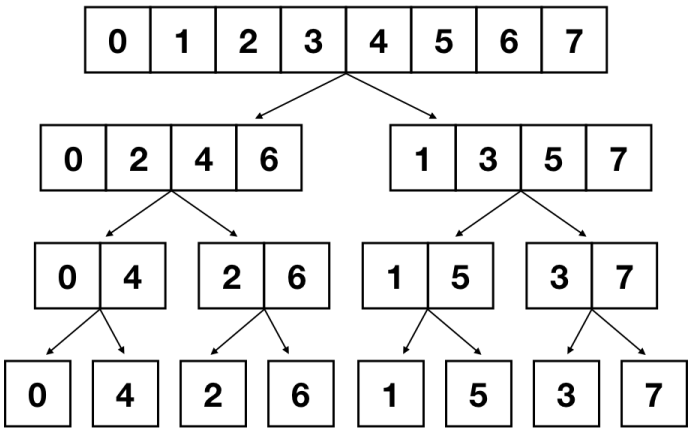

Figure 6: Recursive subdivision scheme of the FFT algorithm for $N = 8$.

we have N signals of length one; Figure 6 illustrates the idea for $N = 8$. The algorithm then "computes" N length one DFTs - notice that these just return the value of each length one signal, so no computation is actually performed. The algorithm then forms $N/2$ length two DFTs the next level up by combining the pairs of length one DFTs that have the same parent signal, and multiplying the "odd" length signal by the appropriate complex value before combination. At each level we incur a cost of $O(N)$, and there are $p = \log_2 N$ levels; thus the total cost of the algorithm is $O(N \log_2 N)$.

The FFT algorithm is remarkable for turning an $O(N^2)$ calculation into an $O(N \log N)$ calculation with no loss of accuracy. For this reason it is a pillar of digital signal processing. However, it is fundamentally an algebraic property of the DFT, based on symmetries. As such, the algorithm is "fragile," and in particular, if you do not uniformly sample your signal f, you cannot apply the FFT algorithm. That however is a discussion for another day (or class).

The FFT algorithm allows us to compute convolutions $x * y$ fast. Suppose the $x, y \in \mathbb{C}^N$ as usual; if we compute $x * y$ directly it will cost us $O(N^2)$ calculations. In order to calculate the non-circular convolution faster, we can use the circular convolution theorem 21, which will allow us to leverage the FFT algorithm. The main idea is that instead of computing $x * y$ directly, we compute $\widehat{x}$ and $\widehat{y}$, each costing $O(N \log N)$ calculations, then we compute the multiplication $\widehat{x}[k]\widehat{y}[k]$ for $0 \leq k < N$, costing $O(N)$ calculations, and then we compute the inverse Fourier transform of $(\widehat{x}[k]\widehat{y}[k])_{0 \leq k < N}$ with another FFT,

which costs $O(N \log N)$ calculations; the total run time of the algorithm if $O(N \log N)$, which in practice (depending upon the exact FFT algorithm you use) will be better for $N \geq 32$. One thing that we have not addressed though, is that the convolution theorem for finite length signal applies to circular convolution. If we do not account for this, we will run into border effects since we will be computing $x \circledast y$ instead of $x * y$. To fix this issue, we zero pad x and y by defining:

$$\widetilde{x}[n] = \begin{cases} x[n] & 0 \leq n < N \\ 0 & N \leq n < 2N \end{cases}$$

The signal $\widetilde{x} \in \mathbb{C}^{2N}$ and is just the signal x but with N zeros appended to the back of it. One can then verify that:

$$\widetilde{x} \circledast \widetilde{y}[n] = x * y[n], \quad 0 \leq n < 2N$$

Thus we apply the fast FFT based algorithm to $\widetilde{x}$ and $\widetilde{y}$ (rather than x and y) to obtain $x * y$ in $O(N \log N)$ time.

Exercise 21. Read Section 3.3 of *A Wavelet Tour of Signal Processing.*

Exercise 22. Read Section 3.4 of *A Wavelet Tour of Signal Processing.*

Exercise 23. Let $\widehat{x}[k]$ be the DFT of a finite signale $x \in \mathbb{C}^N$. Define a signal $y \in \mathbb{C}^{2N}$ by:

$$\widehat{y}[N/2] = \widehat{y}[3N/2] = \widehat{x}[N/2]$$

and

$$\widehat{y}[k] = \begin{cases} 2\widehat{x}[k] & 0 \leq k < N/2 \\ 0 & N/2 < k < 3N/2 \\ 2\widehat{x}[k-N] & 3N/2 < k < 2N \end{cases}$$

Prove that y is an interpolation of x that satisfies $y[2n] = x[n]$ for all $0 \leq n < N$.

Exercise 24. We want to compute numerically the Fourier transform of $f(t)$. Let $a[n] = f(n)$ for $n \in \mathbb{Z}$ be the countably infinite discrete sampling of f and let $x \in \mathbb{C}^N$ be the periodization of a over the period of length N:

$$x[n] = \sum_{p \in \mathbb{Z}} a[n - pN]$$

(a) Prove that the DFT of x is related to the Fourier series of a and to the Fourier transform of f by the following formula:

$$\widehat{x}[k] = \widehat{a}[2\pi k/N] = \sum_{\ell \in \mathbb{Z}} \widehat{f}\left(\frac{2\pi k}{N} - 2\pi\ell\right)$$

(b) Suppose that $|f(t)|$ and $|\widehat{f}(\omega)|$ are negligible when $t \notin [-t_0, t_0]$ and $\omega \notin [-\omega_0, \omega_0]$. Relate N to t_0 and ω_0 so that one can compute an approximate value of $\widehat{f}(\omega)$ for all $\omega \in \mathbb{R}$ by interpolating the samples $\widehat{x} \in \mathbb{C}^N$. It is possible to compute exactly $\widehat{f}(\omega)$ with such an interpolation formula?

Exercise 25. We are going to implement the FFT and fast convolution algorithms:

(a) Implement the DFT algorithm (programming language of your choice). Record the runtime for many values of N, and plot it as a function of N. Do you see the quadratic scaling? Turn in your code and plot(s).

(b) Make precise the $O(N \log N)$ FFT algorithm described above and implement it on your own for $N = 2^p$. Test the algorithm for accuracy by comparing its outputs to the outputs of your DFT algorithm. Test the algorithm for speed by comparing the runtime for numerous values of N to the runtimes your recorded for the DFT. For which value of N does your FFT algorithm become faster? Turn in your code, at least one output showing that the DFT and FFT codes produce the same results, and a plot of the FFT runtimes as a function of N (you can combine this plot with the DFT plot).

(c) Using either your own FFT and inverse FFT code, or built in code (in Matlab or Python, for example) since you are not required to write your own inverse FFT code, implement an algorithm to compute $x * y$ (for $x, y \in \mathbb{C}^N$) in $O(N \log N)$ time. Verify the accuracy by comparing against convolution code that computes $x * y$ directly (either your own code, or built in code), and compare the runtimes. For which N is your $O(N \log N)$ convolution code faster?

4 Time Meets Frequency

Chapter 4 of A Wavelet Tour of Signal Processing

4.1 Time Frequency Atoms

Section 4.1 of A Wavelet Tour of Signal Processing

A linear *time frequency transform* correlates the signal $f(t)$ with a dictionary of waveforms that are concentrated in time and frequency; these waveforms are called time frequency atoms. Denote a general dictionary of time frequency atoms by:

$$\mathcal{D} = \{\phi_\gamma\}_{\gamma\in\Gamma}, \quad \phi_\gamma \in \mathbf{L}^2(\mathbb{R}), \ \ \|\phi_\gamma\|_2 = 1$$

where Γ is a (multi)-index set. The time frequency transform of $f \in \mathbf{L}^2(\mathbb{R})$ in the dictionary $\mathcal{D}$ computes

$$\Phi f(\gamma) = \langle f, \phi_\gamma\rangle = \int_{-\infty}^{+\infty} f(t)\phi_\gamma^*(t)\,dt$$

Recall the definitions of the time mean u, frequency mean ξ, time variance σ_t^2, and frequency variance σ_ω^2 of a function $f \in \mathbf{L}^2(\mathbb{R})$, first defined when we studied the uncertainty principle in Section 2.4. Apply them to the dictionary $\mathcal{D}$ for each time frequency atom ϕ_γ, and denote the corresponding quantities by

$$u_\gamma, \ \omega_\gamma, \ \sigma_t(\gamma), \ \sigma_\omega(\gamma)$$

The waveform ϕ_γ is essentially supported in time on an interval of length $\sigma_t(\gamma)$, centered at u_γ, while its Fourier transform $\widehat{\phi}_\gamma$ is essentially supported in frequency on an interval of length $\sigma_\omega(\gamma)$, centered at ξ_γ. Thus the joint time frequency support of ϕ_γ in the time frequency plane (t,ω) is given by a Heisenberg box centered at (u_γ,ξ_γ) having time width $\sigma_t(\gamma)$ and frequency width $\sigma_\omega(\gamma)$; see Figure 7.

The Parseval formula (Theorem 6) proves that:

$$\Phi f(\gamma) = \int_{-\infty}^{+\infty} f(t)\phi_\gamma^*(t)\,dt = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \widehat{f}(\omega)\widehat{\phi}_\gamma^*(\omega)\,d\omega$$

Thus we see that $\Phi f(\gamma)$ only depends upon the values of $f(t)$ and $\widehat{f}(\omega)$ in the Heisenberg box of ϕ_γ. In particular, $\Phi f(\gamma)$ only measures the frequencies of f in a neighborhood of ξ_γ, and it only measures these frequencies in a neighborhood of the time u_γ. Because of the uncertainty principle (Theorem 14), we know that

$$\sigma_t(\gamma)\sigma_\omega(\gamma) \geq \frac{1}{2}$$

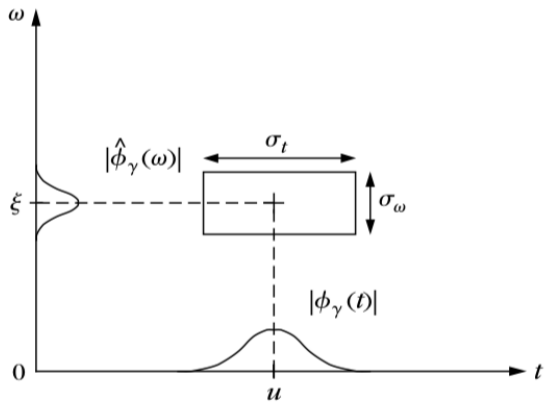

Figure 7: Heisenberg box representing the essential time frequency support of ϕ_γ

Thus it is impossible to measure precisely the frequency response $\widehat{f}(\omega_0)$ at the time t_0. The best we can do is measure the time frequency response of f in a Heisenberg box of area $1/2$. Theorem 14 proves that the time frequency atoms that achieve this optimal time frequency localization are given by Gabor functions; we will come back to this point shortly when we introduce the windowed Fourier transform.

For pattern recognition and machine learning tasks, it often important to construct time frequency representations that behave well with respect to translations of the signal $f(t)$ (and in 2D, rotations as well). Define $f_u(t) = f(t-u)$ as the translation of f by u, and notice that:

$$\Phi f_u(\gamma) = \int_{-\infty}^{+\infty} f(t-u)\phi_\gamma^*(t)\,dt = \int_{-\infty}^{+\infty} f(t)\phi_\gamma^*(t+u)\,dt = \langle f, \phi_{-u,\gamma}\rangle,$$

where $\phi_{u,\gamma}(t) = \phi_\gamma(t-u)$. This motivates the construction of translation invariant dictionaries. A translation invariant dictionary is obtained by starting with a family of generators $\{\phi_\gamma\}_{\gamma\in\Gamma}$, and augmenting this family with all translates of each time frequency atom ϕ_γ:

$$\mathcal{D} = \{\phi_{u,\gamma}\}_{u\in\mathbb{R},\gamma\in\Gamma}$$

Set:

$$\widetilde{\phi}_\gamma(t) = \phi_\gamma^*(-t)$$

The resulting time frequency transform with a translation invariant dictio-

Figure 8: Picture of a castle, taken from Wikipedia. Different regions of the picture have different patterns, such as the sky, the trees, and the castle itself. These patterns have different frequency responses, which are spatially localized.

nary is given by:

$$\Phi f(u,\gamma) = \langle f, \phi_{u,\gamma} \rangle = \int_{-\infty}^{+\infty} f(t)\phi_\gamma^*(t-u)\,dt = f * \widetilde{\phi}_\gamma(u)$$

It thus corresponds to a filtering of f by the time-frequency waveforms $\{\widetilde{\phi}_\gamma\}_{\gamma\in\Gamma}$.

Exercise 26. Read Section 4.1 of *A Wavelet Tour of Signal Processing.*

4.2 Windowed Fourier Transform

Section 4.2 of A Wavelet Tour of Signal Processing

The Fourier transform $\widehat{f}(\omega)$ tells us every frequency in the signal $f(t)$, but it does not tell us *when* such frequencies are present. For example, in music we hear the time variation of the sound frequencies. Similarly, images with vastly different patterns in them may correspond to different frequencies, localized not over time but space; see the picture of the castle in Figure 8 for an example.

A natural way to account for these localized structures is to localize the Fourier transform with a window function. Let g be a real symmetric window $g(t) = g(-t)$, which has support localized around $t = 0$ (e.g., a Gaussian $g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2\sigma^2}$). Translations of this window by $u \in \mathbb{R}$, and modulations of this window by the frequency $\xi \in \mathbb{R}$, yield a *Gabor type dictionary*:

$$\mathcal{D} = \{g_{u,\xi}\}_{u,\xi \in \mathbb{R}}, \quad g_{u,\xi}(t) = g(t-u)e^{i\xi t}$$

The window is normalized so that $\|g\|_2 = 1$, which implies that $\|g_{u,\xi}\|_2 = 1$ for all $(u, \xi) \in \mathbb{R}^2$. The resulting *windowed Fourier transform* (also known as the short time Fourier transform, or Gabor transform) is:

$$Sf(u,\xi) = \langle f, g_{u,\xi} \rangle = \int_{-\infty}^{+\infty} f(t)g(t-u)e^{-i\xi t}\, dt$$

Notice that $Sf(u, \xi)$ computes a localized version of the Fourier transform of $f(t)$, in which the Fourier integral is localized around u by the window $g(t - u)$.

The energy density of the windowed Fourier transform is *spectrogram*:

$$P_S f(u,\xi) = |Sf(u,\xi)|^2 = \left| \int_{-\infty}^{+\infty} f(t)g(t-u)e^{-i\xi t}\, dt \right|$$

The spectrogram removes the phase of $Sf(u, \xi)$ and measures the energy of f in a time frequency neighborhood of (u, ξ) specified by the Heisenberg box of $g_{u,\xi}$. The size of these Heisenberg boxes is in fact independent of (u, ξ), as we now show.

First note that since $g(t)$ is even, $g_{u,\xi}$ is centered at u. The variance around u is:

$$\sigma_t^2 = \int_{-\infty}^{+\infty} (t-u)^2 |g_{u,\xi}(t)|^2\, dt = \int_{-\infty}^{+\infty} t^2 |g(t)|^2\, dt$$

The Fourier transform $\widehat{g}$ of g is real and symmetric because g is real and symmetric. We also compute the Fourier transform of $g_{u,\xi}$ as (set $e_\xi(t) = e^{i\xi t}$):

$$\begin{aligned}
\widehat{g}_{u,\xi}(\omega) &= \widehat{g_u \cdot e_\xi}(\omega) \\
&= (2\pi)^{-1} \widehat{g}_u * \widehat{e}_\xi(\omega) \\
&= (2\pi)^{-1} e_{-u} \cdot \widehat{g} * 2\pi\delta_\xi(\omega) \\
&= e^{-iu(\omega-\xi)} \widehat{g}(\omega - \xi)
\end{aligned}$$

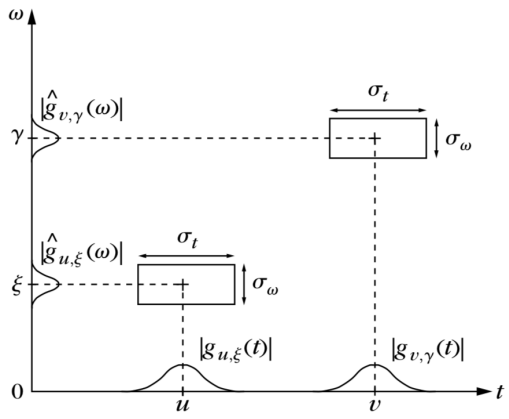

Figure 9: Heisenberg boxes of the windowed Fourier time frequency atoms.

It follows that $\widehat{g}_{u,\xi}$ is centered at ξ, and

$$\sigma_\omega^2 = \frac{1}{2\pi}\int_{-\infty}^{+\infty}(\omega-\xi)^2|\widehat{g}_{u,\xi}(\omega)|^2\,d\omega = \frac{1}{2\pi}\int_{-\infty}^{+\infty}\omega^2|\widehat{g}(\omega)|^2\,d\omega$$

These calculations show that the Heisenberg boxes of $g_{u,\xi}$ centered at (u,ξ) with an area $\sigma_t\sigma_\omega$ that is independent of the location (u,ξ). Thus the windowed Fourier transform has the same resolution across the time frequency plane; see Figure 9.

Exercise 27. Read Section 4.2 of *A Wavelet Tour of Signal Processing*, up to but not including Section 4.2.1.

Exercise 28. We are going to investigate further the windowed Fourier transform with a Gaussian window.

(a) Let $g_\sigma(t)$ be the Gaussian window:

$$g_\sigma(t) = \frac{1}{(2\pi\sigma^2)^{1/4}}e^{-t^2/4\sigma^2}$$

which is normalized so that $\|g\|_2 = 1$ and the time spread of $g(t)$ is $\sigma_t^2 = \sigma^2$. In practice, even though $g(t)$ has infinite support, we will have to sample it over a finite interval $[-N/2, N/2)$ of length N. Let $g_{\sigma,N}(t)$ be the restriction of $g_\sigma(t)$ to this interval:

$$g_{\sigma,N}(t) = \mathbf{1}_{[-N/2,N/2)}(t)g_\sigma(t)$$

Give an upper bound for the error $\|g_\sigma - g_{\sigma,N}\|_2$ in terms of σ and N. Recall our intuition that the "essential" width of $g_\sigma(t)$ is σ. If we take $N = \sigma$, how big is the bound on the error? Is this error acceptable?

(b) Compute σ_ω in terms of σ for $g_\sigma(t)$.

(c) Implement a discrete version of the windowed Fourier transform. Assume that your signal $x[n] = f(n)$ for $n \in \mathbb{Z} \cap [-N/2, N/2)$, and compute the discrete vectors:

$$g_{\sigma,k}[n] = g_\sigma(n) \exp\left(\frac{2\pi i k n}{N}\right), \quad -N/2 \le n < N/2, \;\; 0 \le k < N$$

with the same samples. Define $S_\sigma x[n,k]$ as:

$$S_\sigma x[n,k] = \exp\left(-\frac{2\pi i k n}{N}\right) x * g_{\sigma,k}[n], \quad -N/2 \le n < N/2, \;\; 0 \le k < N$$

(You should convince yourself this definition is consistent with the definition above for signals f). Use your work on the earlier exercises to get a fast implementation with $O(N^2 \log N)$ run time. Then analyze the signal $f(t)$ defined as:

$$f(t) = \exp\left[i\frac{\pi}{N}\left(t + \frac{N}{2}\right)^2\right]$$

with a sampling $x[n] = f(n)$ for $n \in \mathbb{Z} \cap [-N/2, N/2)$. Compute the power spectrum of x, which is $|\widehat{x}[k]|^2$, and the spectrogram $P_S x[n,k] = |S_\sigma x[n,k]|^2$. Plot them both and give an interpretation of your results.

References

[1] Stéphane Mallat. *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, 3rd edition, 2008.

[2] Yves Meyer. *Wavelets and Operators*, volume 1. Cambridge University Press, 1993.

[3] Elias M. Stein and Rami Shakarchi. *Fourier Analysis: An Introduction*. Princeton Lectures in Analysis. Princeton University Press, 2003.

[4] Richard L. Wheeden and Antoni Zygmund. *Measure and Integral: An Introduction to Real Analysis*. Marcel Dekker, 1977.

[5] Elias M. Stein and Guido Weiss. *Introduction to Fourier Analysis on Euclidean Spaces*. Princeton University Press, 1971.

[6] John J. Benedetto and Matthew Dellatorre. Uncertainty principles and weighted norm inequalities. *Contemporary Mathematics*, 693:55–78, 2017.