The Nyquist Theorem and Sinc Interpolation

Nicholas Dwork

1 Background

For this document, we will use the following definitions for the Fourier Transform and inverse Fourier Transform:

$$F(k) = \mathcal{F}\{f\}(k) = \int_{-\infty}^{\infty} f(x)\, \exp(-i\, 2\pi\, k\, x)\, dx$$
$$f(x) = \mathcal{F}^{-1}\{F\}(x) = \int_{-\infty}^{\infty} F(k) \exp(i\, 2\pi\, k\, x)\, dk,$$

where $i = \sqrt{-1}$.

The Discrete Fourier Transform (DFT) and the inverse Discrete Fourier Transform (DFT^{-1}) are linear operators; they can be derived as specific Riemann sums that approximate the Fourier Transforms where each element of the partition is equal in size, and the element in the summation (x_j or k_j) is the midpoint of the partition element. They are denoted as follows:

$$V[m] = \text{DFT}\{v\}[m] = \sum_{n=0}^{N-1} v[n] \exp\left(-i\, 2\pi\, \frac{mn}{N}\right)$$
$$v[n] = \text{DFT}^{-1}\{V\}[n] = \frac{1}{N} \sum_{m=0}^{N-1} V[k] \exp\left(i\, 2\pi\, \frac{mn}{N}\right).$$

The DFT and DFT^{-1} can be calculated very quickly using the Fast Fourier Transform (FFT) algorithm.

2 The Nyquist Theorem

Consider a function of interest $f : \mathbb{R} \to \mathbb{C}$. Suppose we are given a set of points $(x_1, x_2, \ldots, x_N)$ and the corresponding values $(f(x_1), f(x_2), \ldots, f(x_N))$. These two ordered sets together are called "samples of f".

In some cases, f can be recovered exactly form samples. The Nyquist Theorem provides one set of criteria where this is the case.

Suppose the support of $\mathcal{F}f$ is compact. Let B be the lowest upper bound on s such that $|\mathcal{F}\{f\}(k)| = 0$ for all $k \geq B$. The number B is called the bandwidth of the function f. A cartoon example of this is shown in Figure 1. Note that since $\mathcal{F}f$ has compact support, f can not have compact support by the uncertainty principle of the Fourier Transform.

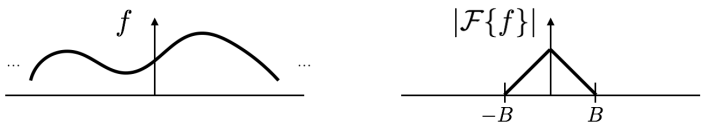

Figure 1: (Left) the original function; (Right) the original spectrum

Suppose one samples f with an infinite number of evenly spaced samples[1]: $\text{III}_{\Delta x}\, f$; then the Fourier Transform of the sampled data is

$$\mathcal{F}\left\{\text{III}_{\Delta x}\, f\right\} = \mathcal{F}f * \mathcal{F}\{\text{III}_{\Delta x}\} = \mathcal{F}f * \frac{1}{\Delta x}\text{III}_{1/\Delta x},$$

[1] One might think of a set of evenly spaced samples as $(\ldots, f(x_{-1}), f(x_0), f(x_1), \ldots)$. Sets of this form together with pointwise addition and vector multiplication are a vector space, and that vector space is isomorphic to $\text{III}_{\Delta x} f$.

where Ш is the comb (or Shah) function defined as $\text{Ш}_{\Delta x}(x) = \sum_{m=-\infty}^{\infty} \delta(x - m\,\Delta x)$. One can see a cartoon example of the sampled spectrum in Figure 2.

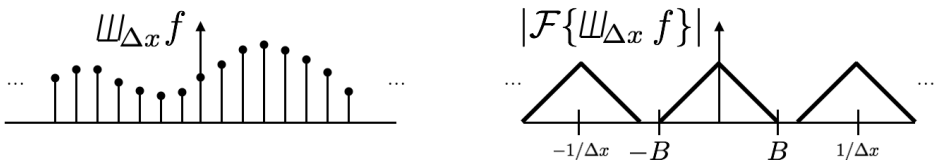

Figure 2: (Left) the sampled function; (Right) the spectrum of the sampled function

The fraction $1/\Delta x$ is called the sampling frequency. One notes that if the sampling frequency is greater than $2B$ then the repetitions of the spectrum don't overlap. That is, there is no aliasing.

If this condition is satisfied, one can recover the original spectrum by multiplying the sampled spectrum by Π_B where

$$\Pi_B(k) = \begin{cases} 1 & \text{if } |k| \leq B/2 \\ 0 & \text{otherwise} \end{cases}.$$

This corresponds to convolving the sampled function with a sinc. And thus, we have arrived at Nyquist's Theorem.

Nyquist's Theorem Consider a function f that is bandlimited with bandwidth B. If the sampling frequency $1/\Delta x$ is greater than twice the bandwidth B then f can be recovered from an infinite set of uniformly spaced samples as follows:

$$\boxed{f = \text{Ш}_{\Delta x} f * b,} \tag{1}$$

where $b(x) = B\,\text{sinc}(Bx)$ for all x.

3 Sinc Interpolation

In real world applications, a computer can not store an infinite set of samples. Instead, a computer will store a finite set of evenly spaced samples.

Consider a bandlimited function $f : \mathbb{R} \to \mathbb{C}$ with bandwidth B, and an N-tuple of evenly spaced samples $(f(x_1), f(x_2), \ldots, f(x_N))$.

Suppose we wish to approximate $f(x_i')$ for all $x_i' \in \{x_1', \ldots, x_M'\}$. With sinc interpolation, we approximate the values as

$$\hat{f} = (fS * b)\,S',$$

where $S = \sum_{i=1}^{N} \delta(x - x_i)$, and $S' = \sum_{i=1}^{M} \delta(x - x_i')$. Note the similarity to (1). In both cases, samples of f are convolved with b.

Note that $fS = f\text{Ш}_{\Delta x}\Pi_X$ where $fS \subset [-X/2, X/2]$. The spectrum of the sampled data

$$\mathcal{F}\{fS\} = \mathcal{F}f * b_X,$$

where $b_X(s) = X\,\text{sinc}(Xs)$. This shows that the spectrum $\mathcal{F}\{fS\}$ does not have compact support, and so there will be aliasing after sinc interpolation. We hope that the aliasing isn't very bad; in many cases, it isn't.

4 Zero Padding

A fast algorithm for performing sinc interpolation is with zero padding.

Consider a finite set of samples of f, $(f(x_0), f(x_1), \ldots, f(x_N))$. This set can be represented as $g = \text{Ш}_{\Delta x}\, f\, \Pi_\delta$. Equivalently, it can be represented by $h = g\Pi_\Delta$ where $\Delta > \delta$ since $\Pi_\delta\,\Pi_\Delta = \Pi_\delta$. Taking the Fourier Transform of h,

$$\begin{aligned} \mathcal{F}h &= \mathcal{F}\left\{\Pi_\Delta \Pi_\delta \text{Ш}_{\Delta x} f\right\} \\ &= \mathcal{F}\{\Pi_\Delta\} * \mathcal{F}\left\{\Pi_\delta \text{Ш}_{\Delta x} f\right\} \\ &= b_\Delta * g, \end{aligned} \tag{2}$$

where $b_\Delta(s) = \Delta\,\mathrm{sinc}(\Delta s)$. Evaluating (2) at $s = m/M$,

$$\mathsf{DFT}\{(f(x_0), \ldots, f(x_N), 0, \ldots, 0)\}[m] = \mathcal{F}\{h\}\,(m/M) = (b_\Delta * \mathcal{F}g)\,(m/M)\,.$$

This shows that zero-padding in the space domain corresponds to sinc interpolation in the Fourier domain. So, if one wanted to quickly perform sinc interpolation on a set of samples of a function of f, one could follow Algorithm 1. Note that this algorithm is appropriate when one desires a subset of evenly spaced points after sinc interpolation.

Algorithm 1: Fourier Sinc Interpolation

Inputs: $\{f(x_0), f(x_1), \ldots, f(x_M)\}$
Step 1: Compute $v = \mathsf{DFT}\,\{\,(\,(f(x_0), f(x_1), \ldots, f(x_M)\,)\,\}$
Step 2: Compute $v_z = Z_p\{v\}$, where Z_p is the zero-padding linear transformation.
Step 3: Compute $f_s = \mathsf{DFT}^{-1}\{v_z\}$