# Sampling Theory 101

This document is a short overview of some aspects of sampling theory which are essential for understanding the problems of Volume Rendering, which can be viewed as nothing but resampling a data set obtained from sampling some unknown function.

Prerequisite for this document is a basic understanding of Fourier Analysis on an intuitive level. You have to know that a function *f(x)* in the spatial (or time) domain has a counterpart *F(f)* in the frequency domain. Any function satisfying some simple properties can be written as a weighed sum of harmonic functions (shifted and scaled sine curves), and *(F(f))(s)*, called the *Fourier transform* or *spectrum* of *f*, gives the weight of the harmonic function of frequency *s* in *f*.

## Functions and Their Fourier Transforms

The process of finding the weighs of the harmonic components of a function is called *Fourier Analysis*, the opposite process of summing the harmonic components to reconstruct a function is called *Fourier Synthesis*. In our functional nomenclature, Fourier Analysis means applying the *F* function, Fourier Synthesis means applying its inverse $F^{-1}$. Figure 1 shows a function *f* and its Fourier transform *F(f)*.

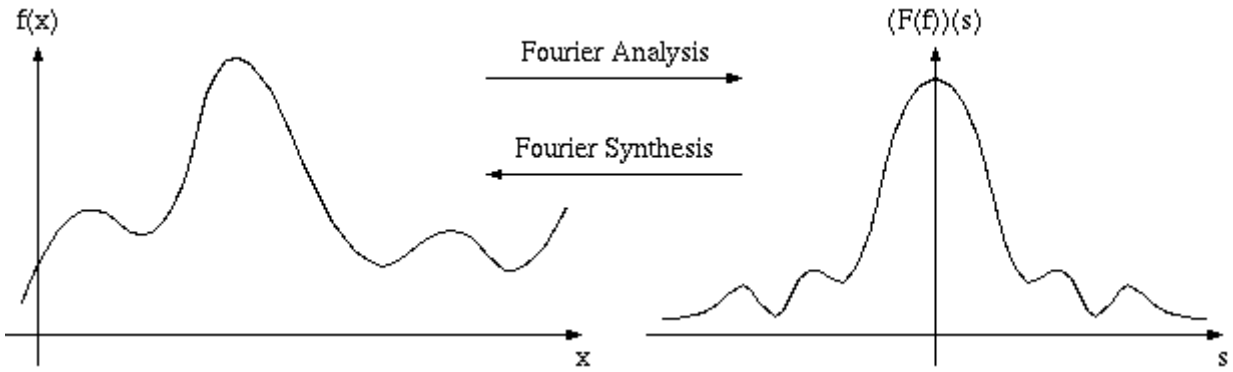


**Figure 1**: A function *f* and its Fourier transform *F(f)*. Both the function and its Fourier transform are complex-valued, but in graphs like this only the magnitudes of the functions are shown.

As it turns out, the operators *F* and $F^{-1}$ are identical up to a minus sign; thus, Fourier Analysis and Fourier Synthesis are almost symmetrical operators. This means, if a function of some "shape" has a certain Fourier transform, the Fourier transform of the Fourier transform (the latter one being interpreted as a spatial domain function again) has the same "shape" as the original function.

The following paragraphs introduce some basic functions which turn out to be important for sampling theory:

### The Dirac Impulse

The *Dirac impulse* $d_{x0}$ is not exactly a function, but it has a well-defined Fourier transform, and hence strong enough mathematical backing to be considered here. The Dirac impulse is a function that is zero for any *x != x0*, and its overall integral is one. This means, the function value $d_{x0}(x0)$ cannot be finite,

but we don't care too much about that here.

Mathematically, the Dirac impulse is defined as the limit of a series of narrower and narrower functions, constantly having an integral of one. In diagrams, a Dirac impulse $d_{x0}$ is drawn as a single peak at *x0* of height one, sometimes with an arrow to emphasize its "infiniteness". (We don't follow this convention here.)

The Dirac impulse (or, more precise, a pair of those) shows up as the Fourier transform of a single harmonic function (like a sine curve), see Figure 2. After all, a single harmonic consists of exactly one harmonic component; this fact can conveniently be expressed by Dirac impulses. (Why a pair? Because the magnitude of the Fourier transform of a real-valued function is always symmetrical with respect to the axis $s = 0$.)

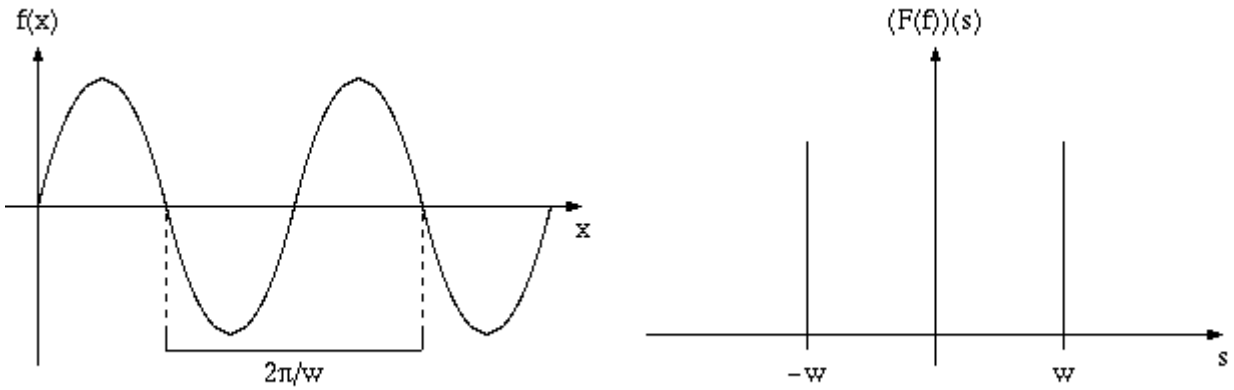


**Figure 2**: The Fourier transform of a harmonic of frequency *w* is a pair of Dirac impulses $d_{-w}$ and $d_w$.

**Comb Functions**

A *comb function* $c_w$ is an infinite series of equidistant Dirac impulses, where adjacent impulses are a distant of *w* apart. Surprisingly, the Fourier transform of a comb function $c_w$ is another (scaled) comb function $c_{w'}$, where *w' = 2*pi/w*, see Figure 3.

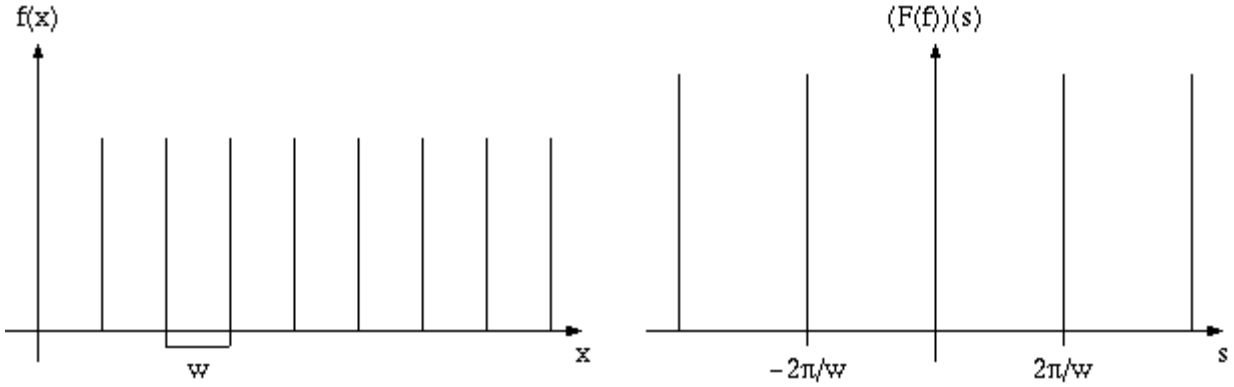


**Figure 3**: The Fourier transform of a comb function $c_w$ is a scaled comb function $c_{w'}$, where $w' = 2*pi/w$.

## Operators on Function Spaces

For sampling theory applications, especially Volume Rendering, functions and their Fourier transforms are not very interesting by themselves. In order to be applicable, we have to be able to express common sampling operations as operations on function spaces.

The two most interesting operations on functions are pointwise multiplication and convolution.

#### Pointwise Multiplication

Assuming that $f_1$ and $f_2$ are two functions defined over the same domain, the pointwise product $f = f_1 f_2$ is defined by $f(x) = f_1(x) * f_2(x)$. In other words, the pointwise product is defined by multiplying the two functions point-by-point (pretty straightforward, that!).

#### Convolution

The convolution operator is a bit more complicated. If $f_1$ and $f_2$ are two functions defined over the set of real numbers, the convolution $f = f_1 * f_2$ is defined as the integral $f(x) = S_{-oo}^{oo} f_1(t) * f_2(x - t)\, dt$.

Maybe this visualization helps understanding the process: For each point $x$, the function $f_1$ and a mirror-image of $f_2$ are overlayed with a displacement of $x$, and the pointwise product of these two functions is integrated over the whole real line.

On second thought, maybe this isn't that helpful after all. But in the special case of one of the functions being a sum of Dirac impulses (e.g., a comb function), it is easier: The convolution $f * c_w$ consists of infinitely many copies of $f$, each two adjacent ones being $w$ apart, and then all added up, see Figure 4.

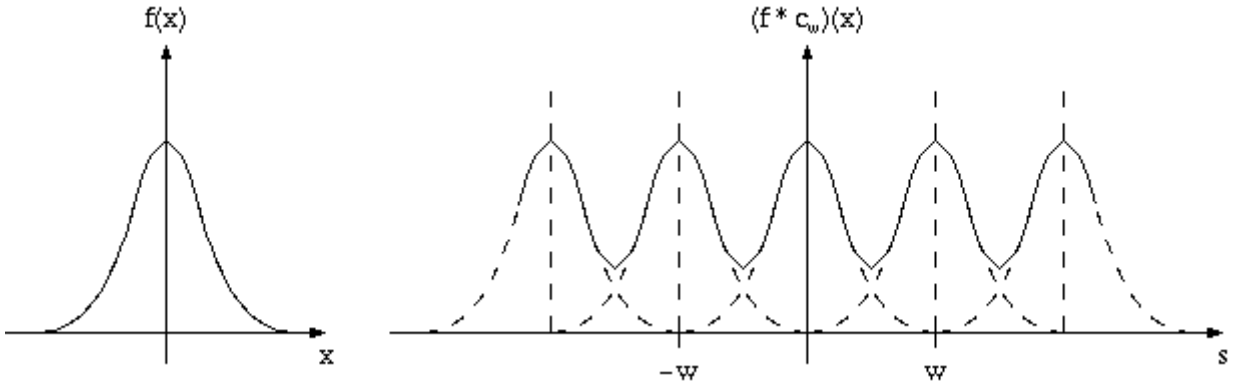


**Figure 4**: The convolution of a function *f* and a comb function $c_w$ consists of the sum of mirrored copies of *f*, shifted and scaled by the Dirace impulses defining $c_w$.

#### The Duality of Pointwise Multiplication and Convolution

The treatment of pointwise multiplication and convolution raises an interesting question: If the Fourier transforms of $f_1$ and $f_2$, $F(f_1)$ and $F(f_2)$, are known, what are the Fourier transforms of $f_1 f_2$ and $f_1 * f_2$?

The (hopefully not) surprising answer is: The Fourier transform of the pointwise product of two functions is the convolution of the two Fourier transforms, $F(f_1 f_2) = F(f_1) * F(f_2)$, and the Fourier transform of the convolution of the two functions is the pointwise product of the two Fourier transforms, $F(f_1 * f_2) = F(f_1) F(f_2)$.

In other words, convolution in the spatial domain is equivalent to pointwise multiplication in the frequency domain, and vice versa. The two operators are *dual* to each other.

## The Sampling Process Seen Mathematically

With the prerequisites out of the way, we can now express the process of sampling a continuous function in mathematical terms, and can hence understand the limitations of the process.

Sampling a function $f(x)$ on a regular grid means representing the continuous function by a discrete set of function values $f(x_1), f(x_2), f(x_3), \ldots$ If all pairs of adjacent sample position are a distance $w$ apart, this can be expressed by multiplying *f* pointwise with a comb function $c_w$, see Figure 5.

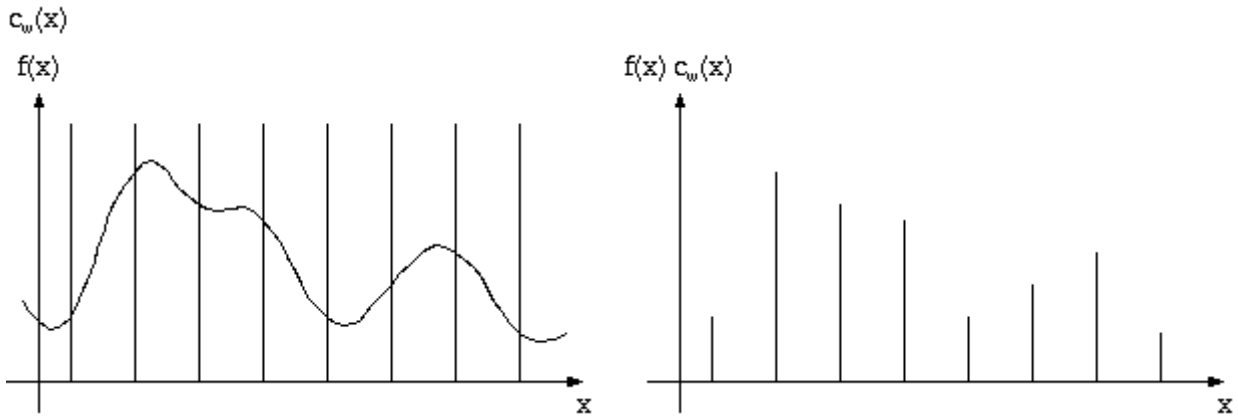


**Figure 5**: A function *f* is sampled by multiplying it with a comb function $c_w$. The result of sampling is a (discrete) sum of scaled Dirac impulses, which can be represented by a discrete set of weights - the sampled data set.

## Reconstructing a Sampled Function

Now that we have seen how to express the sampling process in terms of function space operators, how can the reconstruction process be formalized? To answer this, we have to understand how sampling a function affects its Fourier transform.

As we know, multiplying two functions point-by-point is equivalent to convolving the two functions' Fourier transforms. The Fourier transform of a comb function is another comb function, thus the Fourier transform of the sampled function is the Fourier transform of the original function, "replicated" infinitely often by the comb, see Figure 6.

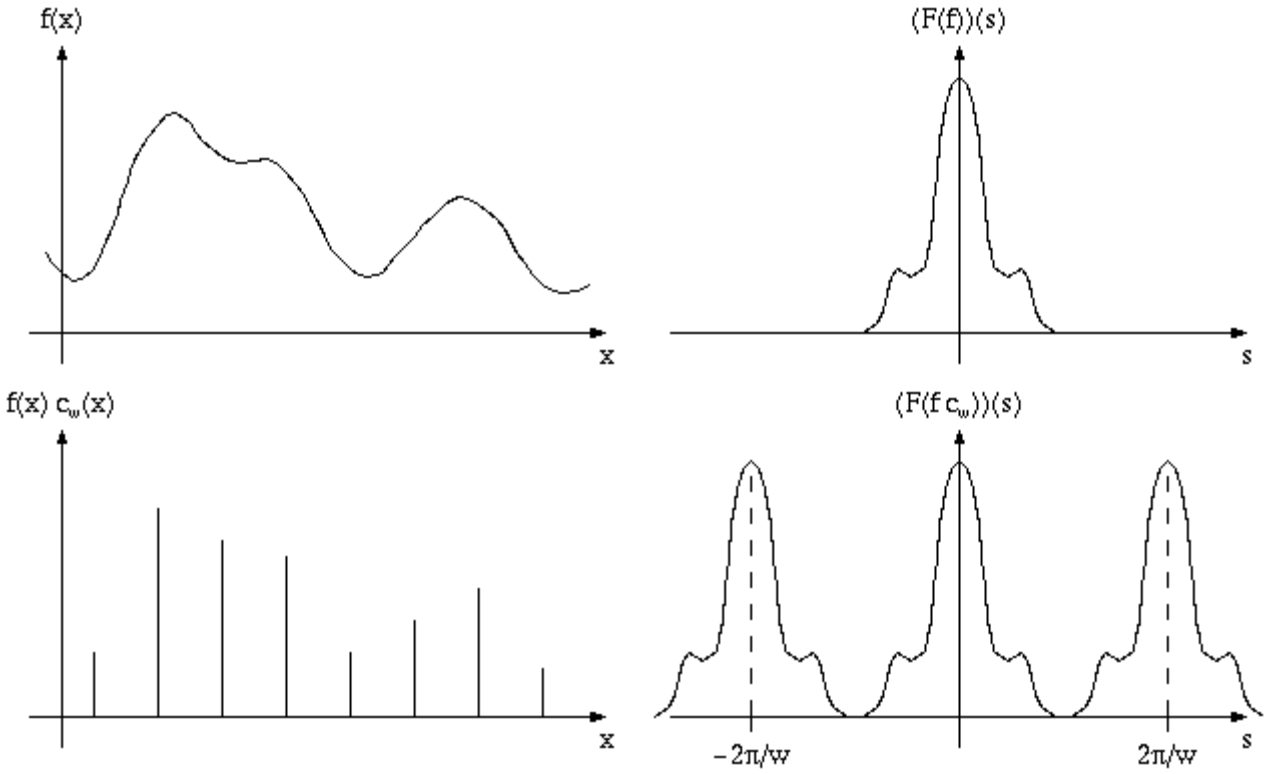


**Figure 6**: A function *f* and its Fourier transform *F(f)*. After sampling, the spectrum *F(f)* is replicated infinitely often in both directions.

**Aliasing and Shannon's Theorem**

The fact that the spectrum of a sampled function *f* is replicated has considerable impact on the sampling process. In order to be able to uniquely reconstruct the original function from its sampled version, the original spectrum and the replications must not overlap.

If the spectra do overlap, the Fourier transform only contains the sum of the two overlapping parts, see Figure 7. As it is impossible to uniquely reconstruct the values of $x$ and $y$ from the equation $x + y = 4$, it is impossible to reconstruct the original spectrum in case the replicated copies overlapped. This implies that it is also impossible to reconstruct *f* itself from the sampled data set - there is a one-to-one relationship between functions and their Fourier transforms; if one could be reconstructed, the other one could be as well.

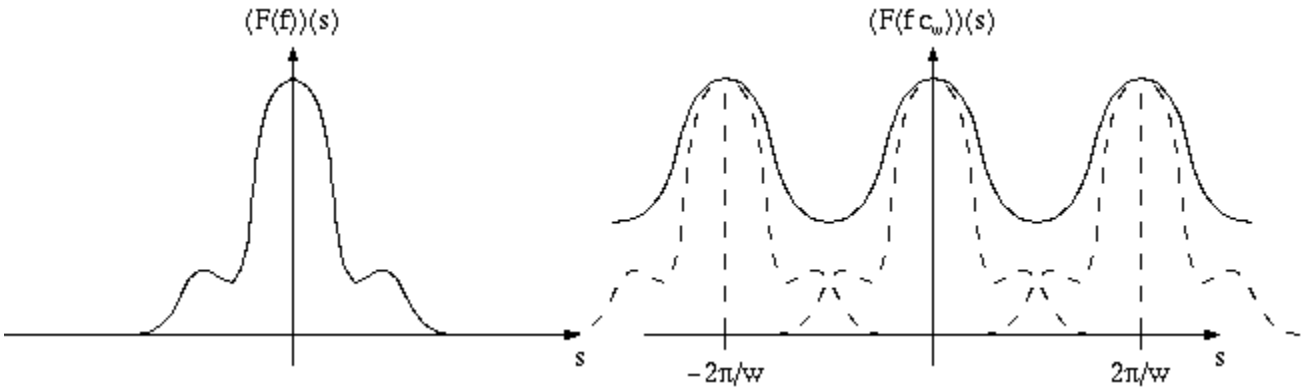


**Figure 7**: If the original spectrum of *f* and the copies introduced by the sampling process overlap, it becomes impossible to reconstruct the original spectrum, and hence it becomes impossible to reconstruct *f* itself.

Taking this effect into account, it follows that a function can be uniquely and accurately reconstructed from its sampled version, if no two adjacent replications of the original spectrum overlap.

This can be stated in a simpler way: The distance between two adjacent replications is *2*pi/w*, where *w* is the distance parameter of the comb function $c_w$ used for sampling *f*. Thus, two adjacent replications do not overlap, if all components of the original spectrum *F(f)* for frequencies larger than or equal to *pi/w* are equal to zero.

A function which has the above property, i.e., that has a certain *s0* such that all frequency components above the frequency *s0* are zero, is called *s0-frequency-limited*.

Noting that *2*pi/w* is the number of "cycles" of the sampling comb function per unit length, in other words, the *sampling frequency*, we conclude that in order to sample an *s0-frequency-limited* function, we have to sample with a frequency larger than *2 * s0*. This simple observation is called *Shannon's theorem*, and the critical sampling frequency of *2 * s0* is often called *Nyquist rate*.

In yet other words: In order to faithfully sample a function *f*, we have to sample with more than twice the frequency of the highest-frequency component of *f*.

If we sample below the Nyquist rate (referred to as *undersampling*), the replicated spectra will overlap; if we then go ahead and try to reconstruct the function anyway, the result will be different from the original, see Figure 8. This phenomenon is appropriately called *aliasing*.

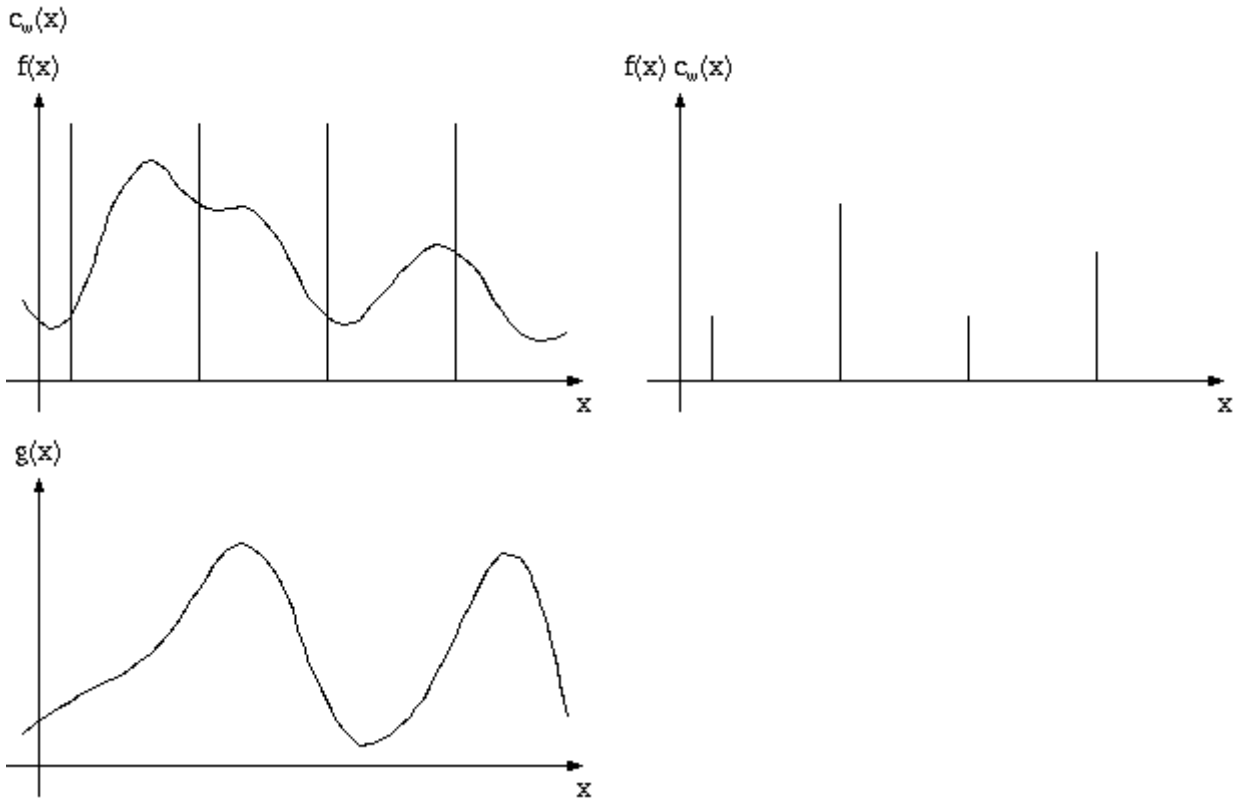


**Figure 8**: Sampling a function *f* below its Nyquist rate will result in aliasing when trying to reconstruct the function as *g*. Note that *f* and *g* have identical values at the sample positions.

**Optimal Reconstruction: The sinc Filter**

Knowing that a function *f* can be reconstructed from its sampled version if sampling above the Nyquist rate is nice, but how does one actually do it? The answer is quite simple: The only difference between *f* and its sampled version is that *f*'s spectrum has been replicated by the sampling process. Just getting rid of the replications will exactly reconstruct *f* in its original form!

The easiest way to erase the replicated spectra is to multiply the Fourier transform of the sampled version of *f* by a *box filter*. A box filter is a function $b_w(s)$ that is one for $-w <= s <= w$ and zero otherwise. If sampling was done (above the Nyquist rate) by a comb function $c_w$, the original spectrum of *f* will stretch at most from *-pi/w* to *pi/w*.

Multiplying the sampled spectrum point-by-point with the box filter $b_{pi/w}$ will extract the original spectrum from the sampled version; then we only have to calculate the inverse Fourier transform of the box-filtered spectrum, and, if we did in fact sample above the Nyquist rate, this will exactly reconstruct *f*, see Figure 9.

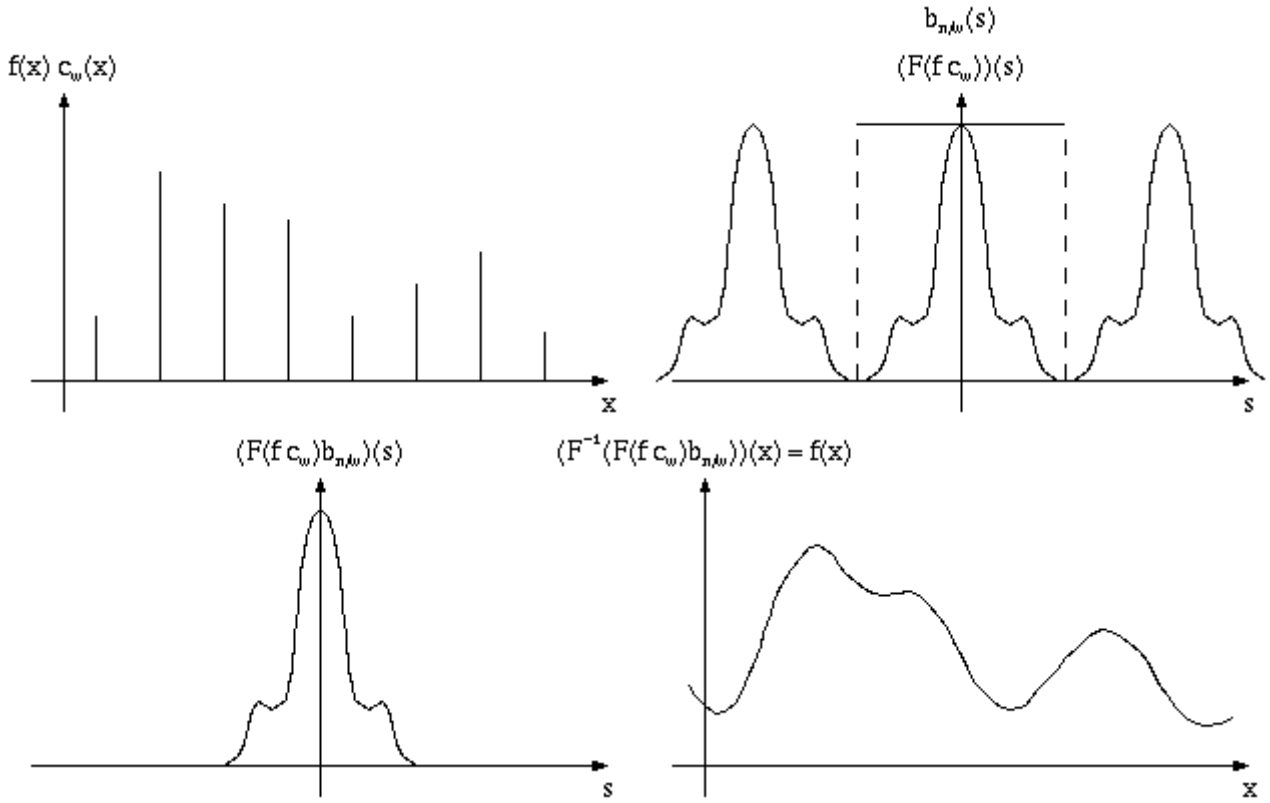


**Figure 9**: Reconstructing a function from its sampled version using a box filter.

There is a slight problem with this approach, though: Its result is purely theoretical. In real life, functions are not given analytically; this means, the original spectrum is also unknown, and the inverse Fourier transform cannot be calculated. Under this constraint, the only operations we can apply to our functions are in the spatial domain. But wait - we already know that multiplying a spectrum by a box filter is equivalent to convoluting a function with the box filter's inverse Fourier transform (a filter's spatial-domain version is often referred to as *filter kernel*)!

This is true, but it doesn't help much either. As it happens, the filter kernel of a box filter is a sinc function ($\sin(x) / x$), see Figure 10; a sinc function has infinite support, meaning that in order to calculate the convolution of the sinc function and the sampled version of *f*, even to reconstruct a single function value, would involve summing up infinitely many shifted and scaled sinc functions.

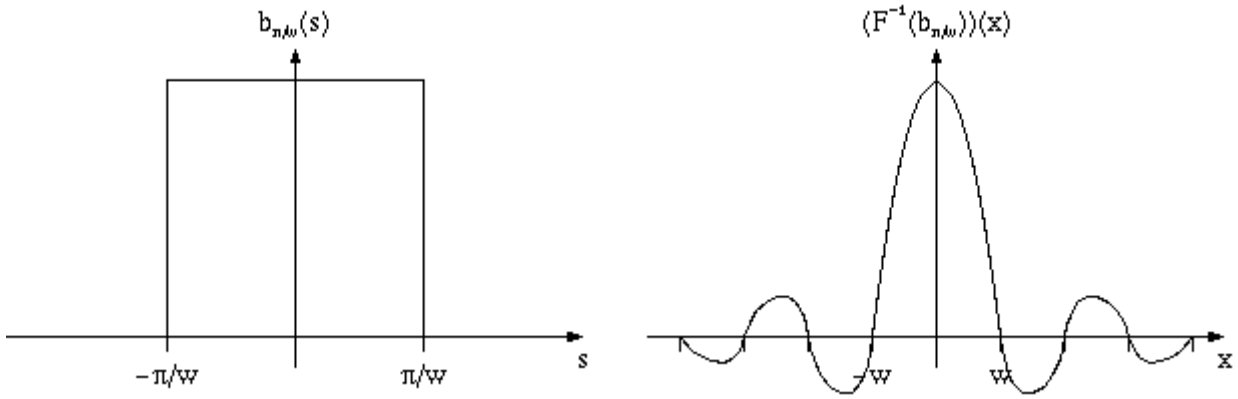

**Figure 10**: The box filter $b_w$ and its filter kernel, a scaled sinc function.

The bottom line is: Optimal reconstruction is possible, but not practical. Which other reconstruction filters could we use? The most obvious, and hence most often used, reconstruction method is linear interpolation. As we'll see, it is also the second-worst method possible - it is only surpassed by constant interpolation, i.e., setting values of points between samples to the value of the nearest sample, often called *nearest-neighbour reconstruction*.

**Why is Linear Interpolation Bad?**

To see why linear interpolation is a bad choice of a reconstruction filter, we have to find out how it affects the spectrum of the sampled function. In the language of function space operators, linear interpolation can be expressed as convolving the sampled function with a triangle function $t_w$, see Figure 11. The triangle function is zero for $x < -w$ or $x > w$, rises linearly to one for $x = 0$, and the drops to zero again for $x = w$. In works on sampling theory, the triangle filter is often referred to as *Bartlett filter*.

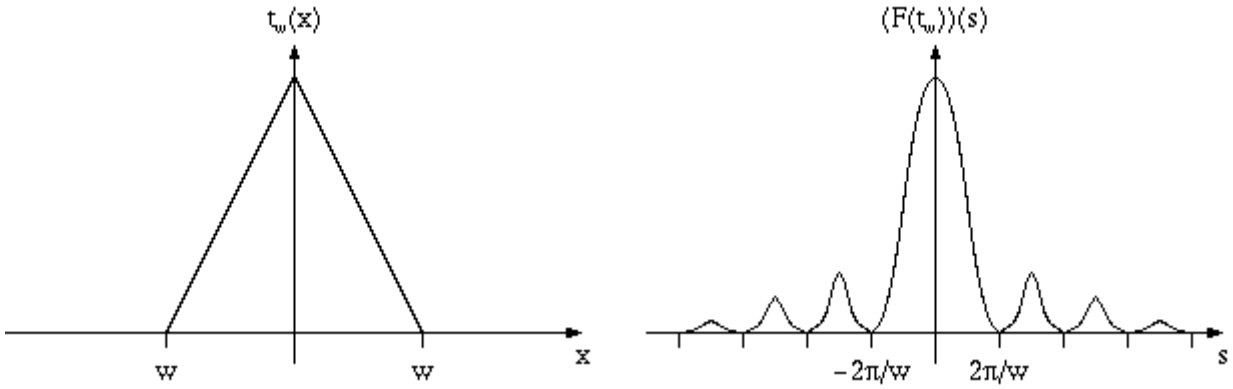


**Figure 11**: The triangle function $t_w$ and its spectrum, a scaled $\text{sinc}^2$ function.

When reconstructing a function from its sampled version using linear interpolation, the scaled comb function representing the sample set is convolved with the triangle function of the same distance parameter. This can be visualized as erecting scaled versions of the triangle function, centered at each sampled position, and summing them all up, see Figure 12.

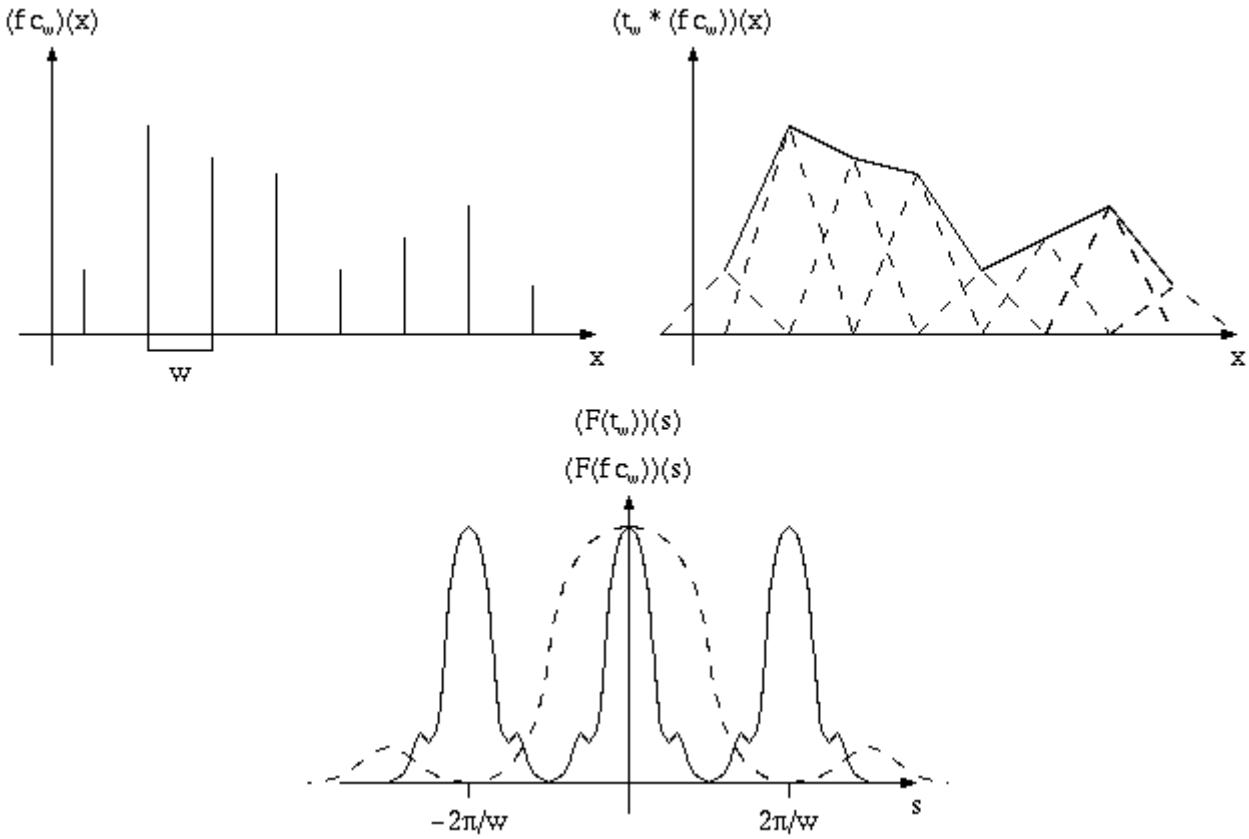


**Figure 12**: Reconstructing a function using linear interpolation. As visible from the spectrum graph, the Bartlett filter not only does not separate the original spectrum from the replications, it also aliases high-frequency components into the reconstruction due to its infinite support.

Convolving a sampled function with a triangle function in the spatial domain is equivalent to multiplying the functions' spectra. As can be seen from Figure 12, the triangle function's spectrum does not at all look like a box filter - the spectrum's magnitude at the intended "cut-off" frequency of *w/pi* is still 40% of its maximal magnitude, and the magnitude does not drop to zero until twice the cut-off frequency, at *s = 2*w/pi*.

This means, that reconstructing a function using a Bartlett filter does not separate the original spectrum from the replications, as would be necessary for faithful reconstruction. Furthermore, the triangle function's spectrum has infinite support, meaning that the reconstructed function has components of arbitrarily high frequencies, as obvious by the sharp "corners" between adjacent straight line segments.

Linear interpolation does an especially bad job of reconstructing a function when the function was sampled only slightly above its Nyquist rate. In that case, the original spectrum and the reconstruction almost overlap, and as the Bartlett filter cannot separate at the Nyquist rate, the reconstructed function is seriously distorted. An extreme example is sampling a single harmonic at slightly more than twice its frequency, see Figure 13.

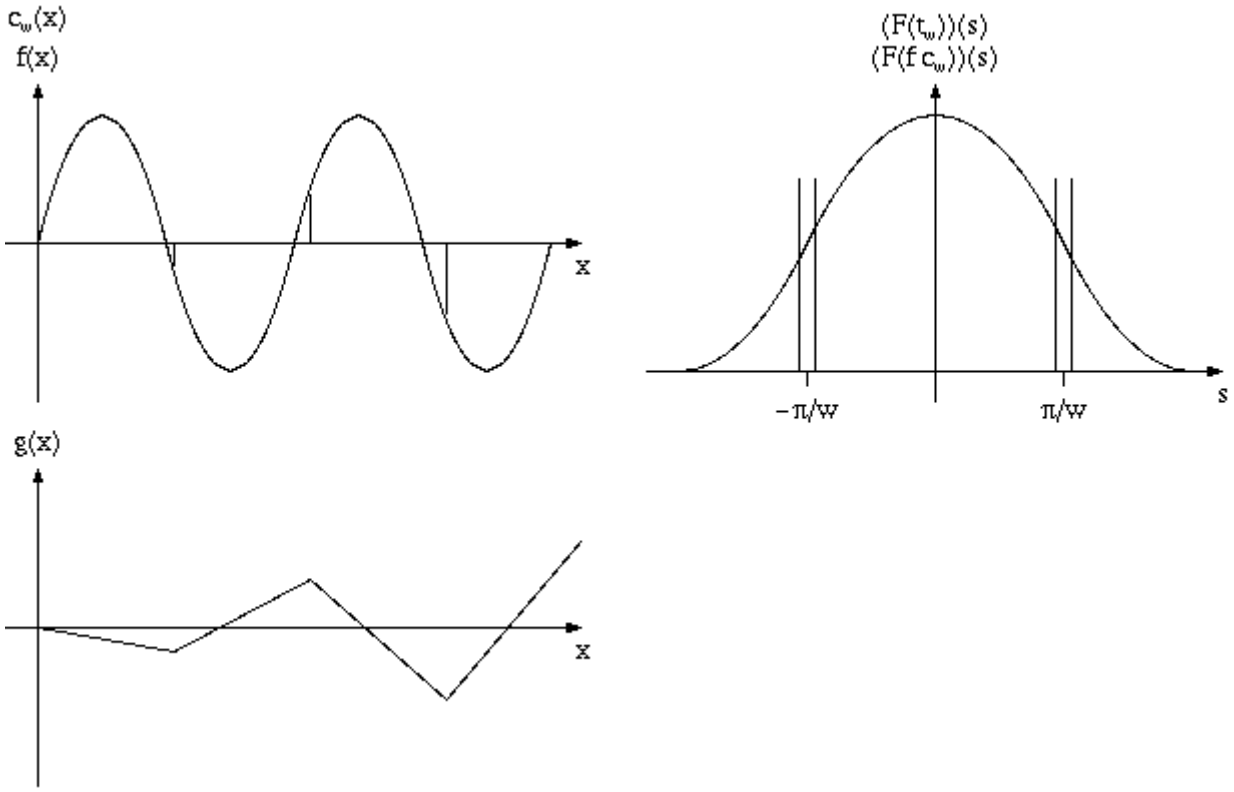


**Figure 13**: Sampling a sine function slightly above the Nyquist rate. The used linear interpolation reconstruction filter destroys the result, because the Dirac impulses making up the sine function's spectrum almost run into each other, being almost cancelled out in the process.

The distortions visible in Figure 13 are not due to aliasing - the function was correctly sampled above its Nyquist rate, so Shannon's theorem states that it could be reconstructed using a box filter. This fine point is lost on a lot of people though; some misunderstand these artifacts as aliasing and apply the seemingly obvious solution - cranking up the sampling rate by a factor of two.

Increasing the sampling rate does improve the reconstruction, there is no arguing about that, but this fact has nothing to do with sampling theory or Shannon's theorem, as often stated. It is the result of basic theorems of Fourier Analysis. If the sampling rate is increased, the triangle function becomes more and more narrow; this in turn makes the Bartlett filter wider and wider. Also, the original spectrum of the sampled function and its replications move farther and farther apart, see Figure 14. These combined facts increase the reconstruction quality. In theory, as the sampling rate approaches infinity, the reconstructed function approaches the original. But again, doubling the sampling rate does not solve the problem, it only makes it slightly less bad.

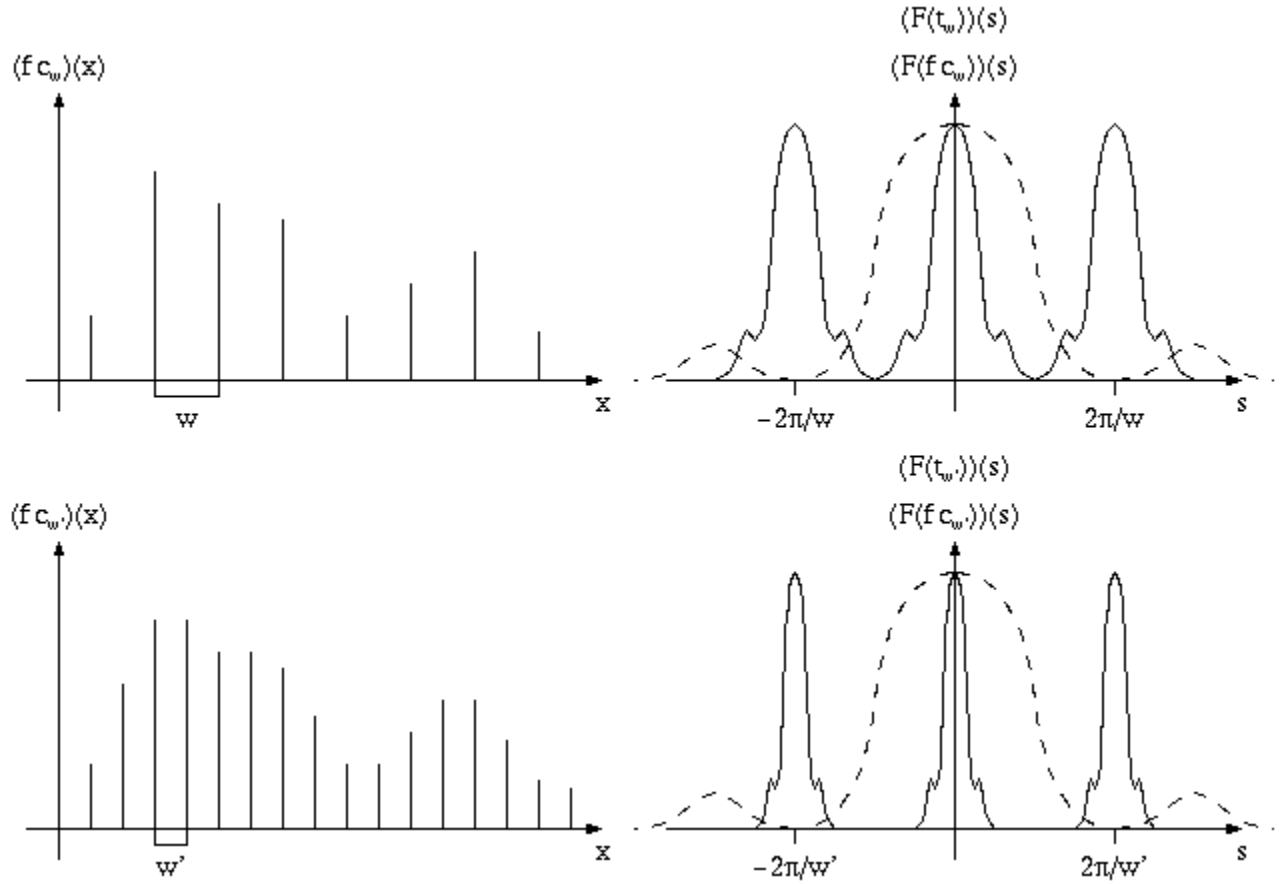


**Figure 14**: Doubling the sampling rate improves the reconstruction quality, but it is in no way a magic solution, as (wrongly) applying Shannon's theorem might suggest.

## Practical Reconstruction Filters

After this rant against linear interpolation and the common misconception of Shannon's theorem it induces, let us look at some working reconstruction filters. As we found out, the perfect solution (the box filter) is not practical; what else is there to use?

The main problem is the following: We need a filter that cuts off unwanted replications of spectra in the frequency domain, but distorts the wanted spectrum as little as possible. This means, we want a near-perfect low-pass filter. The problem lies in the theory: Fourier Analysis allows us to prove that any near-perfect low-pass filter has infinite support in the spatial domain, and is thus not computable.

A good compromise would be a filter that drops to zero "reasonably fast" beyond the *pi/w* cut-off line, and drops to zero reasonably fast in the spatial domain as well, to make calculating the convolution feasible.

### The Truncated Box Filter

One idea springs readily to mind: The box filter's problem is the infinite support of its filter kernel, the sinc function. This infiniteness renders computing the convolution with the sampled function impossible. Why not overcome this limitation by *truncating* the filter kernel, to limit it to finite support? After all, the values of the sinc function approach zero when moving in positive or negative *x*-direction -

truncating it at some point will not change the filter too much. Won't it?

What does truncating a filter kernel mean, in our mathematical model of Fourier Analysis? Truncating a function means multiplying it with a box filter, and this is equivalent to convolving the spectrum of the function with a sinc function. In the case of the box filter itself, truncating its filter kernel will convolute the box in the frequency domain with a sinc function. The result is a box filter with "wiggly" edges, see Figure 15. The truncated box filter suffers from *ringing*, also referred to as the *Gibbs phenomenon*.

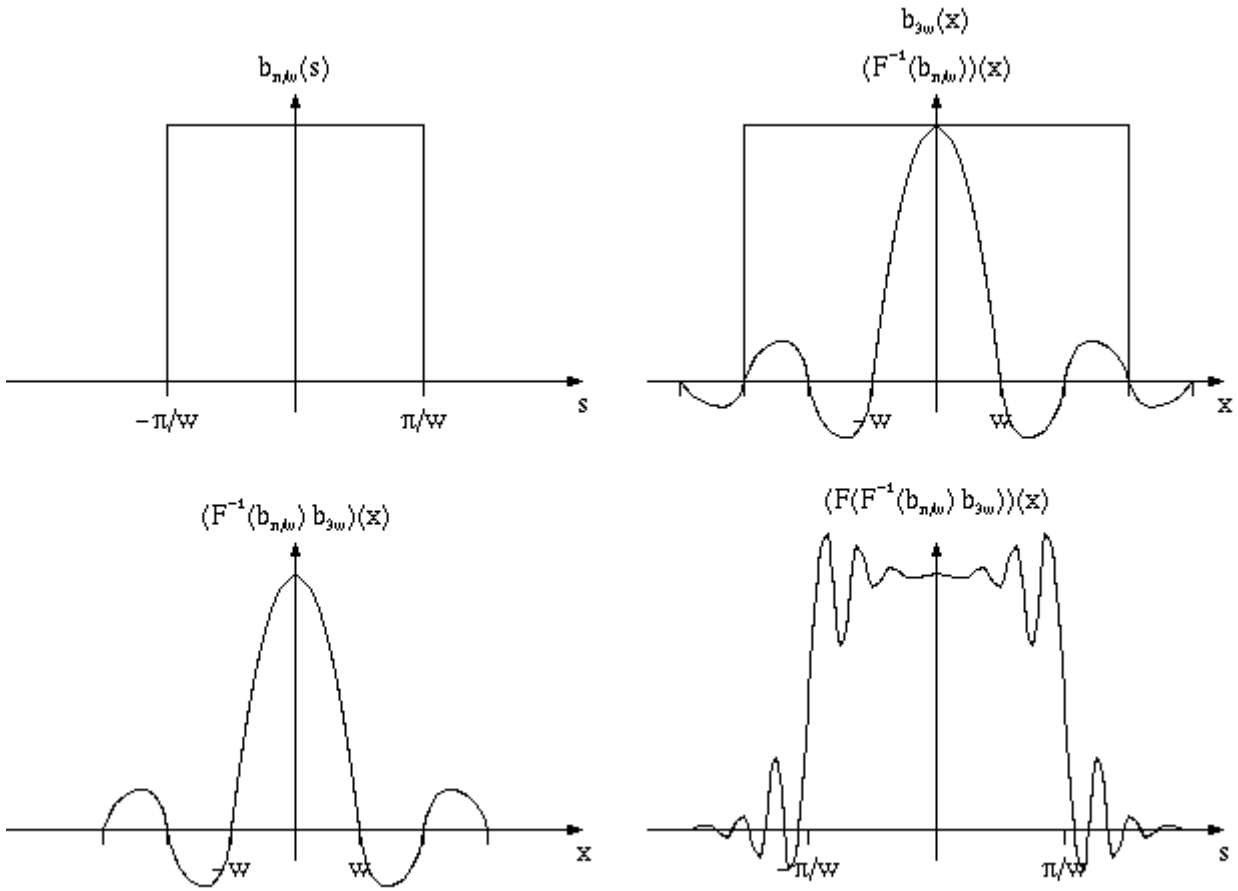


**Figure 15**: Truncating a box filter's kernel causes ringing in the filter.

A better approach is to multiply the box filter's kernel with a function that cuts it off more smoothly, resulting in less distortion in the frequency domain. One such filter is derived by Blinn in [BLIN89b].

Other practical reconstruction filters can be found in [GONZ87] or in any other book/article about signal processing or filtering - look for low-pass filters.

## Resampling a Sampled Data Set

After having discussed how to correctly sample a function and how to correctly (or incorrectly) reconstruct it, let us shift attention to the process of sampling an already sampled function for the second time.

Why would we ever want to do that? For example, to create an image of a function. A computer's frame buffer is organized as a finite set of pixels, thus rendering an image of a function effectively means sampling it over the regular pixel grid. If the function to be rendered was already sampled (as is most often the case), we are in fact sampling the function a second time.

Resampling a function is difficult, because it involves both steps discussed so far - sampling and reconstruction. If resampling a function, the two sampling grids used will hardly ever be identical. Because a sampled function is only represented by its values at the original sample positions, we have to reconstruct the values at the new sample positions first before we can sample it again, see Figure 16.

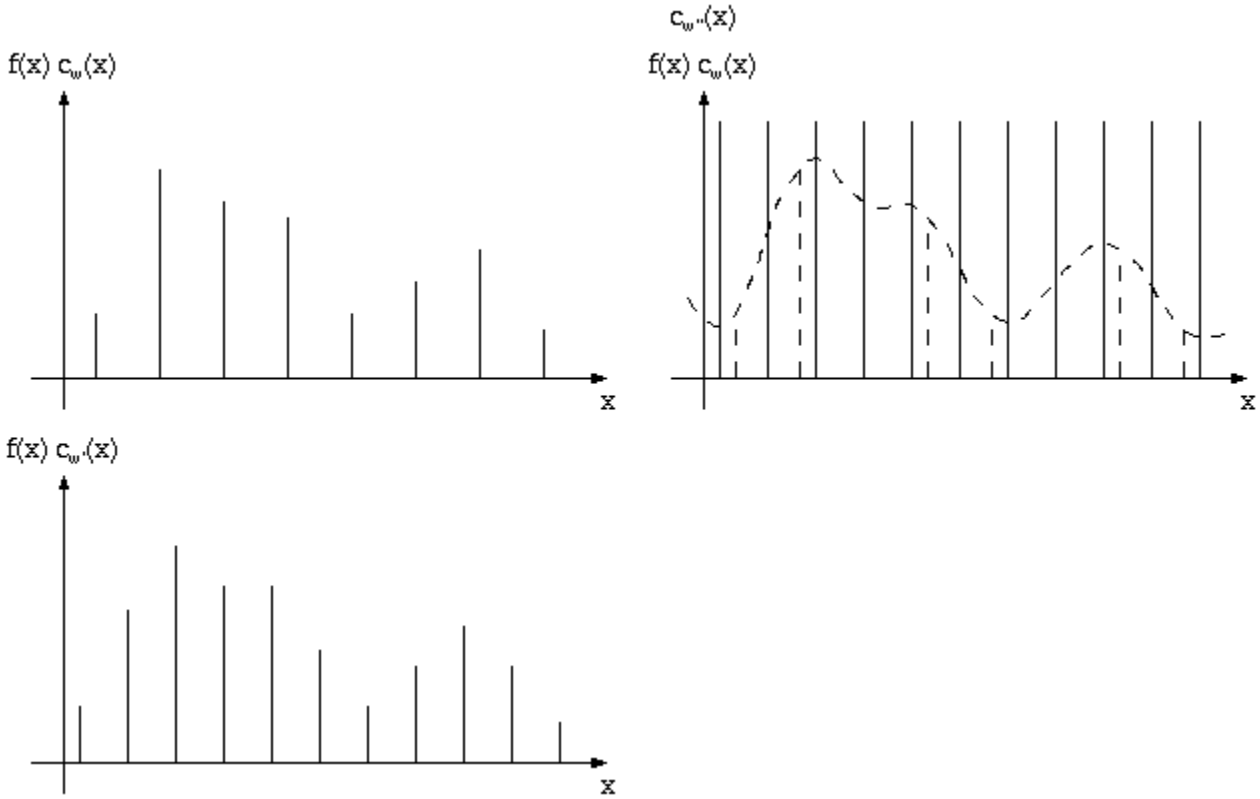


**Figure 16**: Resampling a sampled function involves reconstructing the function and then sampling it on the new grid. Because we are only interested in the reconstructed values at the new sample positions, we only have to reconstruct the function at those positions.

So what are the pitfalls of resampling? The most important one is using a bad reconstruction filter to calculate the original function values at the new sample positions. This mistake has already been discussed in the section on reconstruction.

**Resampling and Oversampling**

The second thing to remember is Shannon's theorem. When resampling a function, we have to make sure that we do not sample below the Nyquist rate; otherwise, faithful reconstruction of the resampled function would become impossible.

This problem has also been addressed earlier, but in the context of resampling it occurs in a different guise: When resampling, the original function is usually unknown - so how can we determine the

Nyquist rate?

The answer is: We do not have to. Assuming that the original sampling was done carefully, we know that the original function did not contain any frequency components above *pi/w*, where *w* is again the distance between adjacent sample positions. This means that we are safely sampling above the Nyquist rate, as long as the new sampling distance *w'* is not lower than the original one.

Wait a minute - don't we have to sample *twice as often* to satisfy Shannon's theorem? **No!** This is the same misunderstanding of sampling theory that lead to the common rule of thumb "sample twice as often to get rid of aliasing" that is often quoted when using linear interpolation as reconstruction filter.

Shannon's theorem states that sampling at the same distance *w* is safe. The only problem occurs when one uses a bad reconstruction filter. In that case, sampling is still safe, but by bad reconstruction one is sampling a completely different function! The obvious artifacts when resampling at similar frequencies are not based on Shannon's theorem, but on bad reconstruction. The same reasoning as in the reconstruction section applies here as well: Two times oversampling improves the result, but does not solve the problem. Using a better reconstruction filter does.

## Downsampling

So far, we have only been talking about sampling above the Nyquist rate. What can we do if we really *want* to sample below the Nyquist rate, for example to compress a sampled data set? In that case, we accept that we will not be able to reconstruct the function exactly, but we still want to reconstruct it as closely as possible.

Just sampling at a lower frequency would introduce aliasing, which could distort the sampled function beyond recognition. Let us reconsider Shannon's theorem: It states that a function can be sampled at a frequency *w*, if the function does not contain frequency components greater than or equal to *w/2*. Thus, if the sampling frequency is predefined, we have to change the function to satisfy this constraint.

This can be done by pre-filtering the function: Before the sampling process starts, the function is fed into a low-pass filter which will cut away all frequency components above *w/2 - e*, where *e* is some small positive value. This results in the function's Nyquist rate to drop to slightly less than *w*, and we can safely proceed to sample at the frequency *w*.

The problem of finding a good applicable low-pass filter is the same as finding a good reconstruction filter - after all, reconstruction is nothing else but low-pass filtering.

## Conclusion

The intent of this little document was to explain some of the basic ideas of sampling theory - too little to talk about it, but just enough to understand the impact sampling theory has on common applications like Volume Rendering. I also tried to correct some common misunderstandings about sampling theory, especially the "two-times-oversampling" rule of thumb, by showing that undersampling is not the major source of artifacts in sampled functions. Understanding the limitations of reconstruction is essential to getting it right some day.

Another intention was to point out some sampling and reconstruction methods that actually work; alas, discussion of the really good filters is beyond the scope of this document. See [BLINN89b] for a readable derivation of a "nice" reconstruction filter, and [MARS94] for a comparison of some existing

filters (alas, Marschner and Lobb didn't include Blinn's filter in their evaluation).

And thanks for reading this far!

## References

The following books/articles offer a more in-depth coverage of the topics touched in this document:


- **[BLIN89b]** Blinn, J.F., *Return of the Jaggy*, CG&A, 9(2), March 1989, pp. 82-89
- **[GONZ87]** Gonzales, R. and Wintz, P., *Digital Image Processing*, 2nd edition, Addison-Wesley, Reading, MA, 1987
- **[MARS94]** Marschner, S.R. and Lobb, R.J., *An Evaluation of Reconstruction Filters for Volume Rendering*, in Proceedings of Visualization '94, Washington, DC (1994), pp. 100-107