DSP Notes

Jeremy Neal Kelly
www.anthemion.org
August 28, 2015

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/4.0/.

Contents

1 Statistics and probability

The variable representing the *input* in some data series is known as the **independent variable**, the **domain**, or the **abscissa**; the variable representing the *output* is known as the **dependent variable**, the **range**, or the **ordinate**.

If the mean of samples x_0 through x_N is μ, the **deviation** of each sample is $|x_i - \mu|$. Given:

$$\sigma^2 = \frac{1}{N-1}\sum_{i=0}^{N-1}(x_i - \mu)^2$$

σ^2 and σ estimate the **variance** and the **standard deviation** of the *population*. Dividing by N rather than $N-1$ gives the exact variance of the *sample*, but that less accurately describes the population.

The variance measures the *power* of the sample variation. When independent random signals are summed, their variances also add to produce the variance of the combined signal.

The mean gives the DC offset of a signal, while the standard deviation measures the AC component. The **root mean square** amplitude:

$$A_{RMS} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1} x_i^2}$$

measures the DC and AC components together.

The mean changes continually as a running series is measured. To avoid recalculating the entire sum at each accumulated point, the variance can also be calculated with:

$$\sigma^2 = \frac{1}{N-1}\left[\sum_{i=0}^{N-1} x_i^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} x_i\right)^2\right]$$

In some cases, the mean represents a value being measured, and the standard deviation, noise. When this is true, the **signal-to-noise ratio (SNR)** equals μ/σ. Conversely, the **coefficient of variation (CV)** is σ/μ.

Non-stationary processes have statistical properties that change as they are sampled.

A **probability mass function** gives the likelihood of each possible outcome for a discrete random variable. A **probability density function** does the same for a continuous variable, with the understanding that the probability at a single point is infinitely small, since the domain contains an infinite range of values. To use a density function, the area under a segment must be calculated. This can be done with the **cumulative distribution function**, which is the integral of the probability density function.

The **normal** or **Gaussian distribution**:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma}\,\mathrm{e}^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

Though $P(x)$ is never zero, the function approaches zero very quickly as x moves away from μ. The normal cumulative distribution function is represented by $\Phi(x)$.

The **Central Limit Theorem** guarantees that, when a set of random values are added, the distribution of their sum approaches a normal distribution as the number of values increases, regardless of their individual distributions. Alternatively, given random numbers R_1 and R_2 that are evenly distributed over $(0, 1]$, the **Box-Muller transform**:

$$R_N = \sqrt{-2\ln R_1}\cos(2\pi R_2)$$

produces values that are normally distributed.

Accuracy describes the proximity of the sample mean to the true value; **precision** describes the proximity of sample values to each other. Poor accuracy is caused by systematic errors; poor precision, by noise.

2 ADC and DAC

Sampling changes time from a continuous variable to a discrete variable; quantization does the same with amplitude.

Quantization produces errors that range from $-1/2$ to $1/2$ bits; the errors generally have an even distribution, and a mean of zero. The standard deviation over this range is $1/\sqrt{12}$, so the resulting noise has RMS amplitude equal to $\sqrt{12}/2^b$ of the full range, where b is the bit depth.

When the errors are not evenly distributed, as happens when signal variations are small relative to the bit depth, the output can be improved by **dithering**, which adds noise to the signal before it is quantized. Small input values which would otherwise be rounded appear in the quantized output as biases toward the positive or negative range of the noise. Since the noise has a mean of zero, this brings the output mean at each point closer to the continuous value than would otherwise be possible.

2.1 Sampling Theorem

An **impulse train** is a series of equally-spaced impulses. Sampling is equivalent to the multiplication of a continuous signal by an impulse train with unit amplitude, which implicitly convolves the two signal spectra. An impulse train with frequency f_s contains an infinite series of components at integer multiples of f_s. Signal multiplication creates output containing the sum and difference of every component pair in the signals. Adding the components produces copies of the source spectrum at multiples of f_s; these are called **upper sidebands**. Subtracting produces mirror images of the spectrum that end at multiples of f_s; these are called **lower sidebands**. The distance between each peak is f_s; when components in the source signal exceed half this distance, the sidebands overlap, and aliasing results. The presence of high-frequency sidebands requires low-pass filtering at the Nyquist frequency when the signal is returned to a continuous form; this is performed by a **reconstruction filter**.

After a frequency f is sampled at rate f_s, the samples are indistinguishable from those of frequency $|f - Nf_s|$, for all integer N.

In practice, impulses are difficult to generate electronically, so DACs use **zero-order hold** components that hold each sample value for one increment. This essentially convolves the impulse train with a rectangular pulse, which in turn scales each output component by:

$$H[f] = \left| \frac{\sin(\pi f/f_s)}{\pi f/f_s} \right| = |\operatorname{sinc}(f/f_s)|$$

This effect must also be corrected by the DAC.

Aliasing always changes the frequency of components that exceed the Nyquist frequency. It can also change the phase of such components, but the only change that is possible is a 180° shift.

2.2 Analog filters for data conversion

Three common analog filters are the **Chebyshev**, **Butterworth**, and **Bessel** designs, each of which optimizes a particular filtering characteristic.

The sharpest roll-off is offered by the Chebyshev filter, but this design also produces amplitude variations in the passband called **passband ripple**. Butterworth filters offer the greatest roll-off achievable without passband ripple.

Step response describes the way a filter behaves after the input changes abruptly from one level to another. After a sudden change, filters exhibiting **overshoot** will briefly pass the target level in the time domain, and then ring, varying above and below the target until the steady state is reached. Chebyshev and Butterworth filters both produce significant overshoot. Bessel filters produce a flat passband and no overshoot, and a maximally linear phase response that creates relatively symmetrical output in response to symmetrical input. Their roll-off is very low, however.

Many devices use **multirate** data conversion. Instead of sampling and processing at the same rate, these devices first sample at a much higher rate, increasing the usable bandwidth relative to the required bandwidth, and allowing the use of simpler and cheaper antialiasing hardware. The samples are then filtered in software and decimated to reach the lower processing rate. After processing, the data is upsampled to a high rate by padding with zeros and filtering digitally. Per the sampling theorem, the sidebands produced by the sampling process are centered around multiples of the sample rate; by increasing this rate, it is possible to use simpler components during reconstruction. In addition to lowering costs, the use of digital filters improves output quality.

2.3 Single-bit data conversion

Single-bit conversion digitizes continuous signals without sampling. Most single-bit techniques use **delta modulation**. In the simplest designs, the analog signal is routed to an IC containing a comparator, a capacitor, and a latch. The capacitor starts with zero voltage. When the signal voltage exceeds that of the capacitor, the latch is set; when it does not, the latch is unset. Output is generated by reading the latch state at a high rate, typically several hundred kilohertz. Every time the latch is read, the capacitor's voltage is increased or decreased, depending on whether the latch was set. The result is a stream of distinct bits, each of which represents an increase or decrease in input voltage at that point. The data is returned to a continuous signal in a similar manner. Single-bit output cannot represent abrupt changes in level; instead, new values are approached incrementally at the **slew rate**, defined by the quantization size and the bit rate. Steady signal levels are approximated by alternating set and unset bits.

Simple single-bit implementations cannot represent audio data effectively without extremely high bit rates. **Continuously Variable Slope Delta modulation** improves fidelity by increasing the step size (and thus the slew rate) when many set or unset bits are read consecutively.

Neither of these techniques produce representations that can be used for general DSP, and neither captures the DC offset of the source signal, if any. More complex designs like **delta-sigma conversion** can be converted to sample representations.

3 Linear systems

In DSP, time domain signals are typically represented with lowercase letters, and frequency domain data with uppercase. Discrete signals are indexed with square brackets, and continuous signals with parentheses.

In this context, a **system** is a process that returns an output signal $y[n]$ in response to an input signal $x[n]$; in this sense, it is a function of signals rather than one of time or sample indices.

A system is **linear** if it exhibits both *homogeneity* and *additivity*. Assuming $x[n] \Rightarrow y[n]$, **homogeneity** requires that:

$$kx[n] \Rightarrow ky[n]$$

If $x_1[n] \Rightarrow y_1[n]$ and $x_2[n] \Rightarrow y_2[n]$, **additivity** requires that the signals pass through without interacting, so that:

$$(x_1[n] + x_2[n]) \Rightarrow (y_1[n] + y_2[n])$$

Linear systems **commute**, so when they are connected in series, changing their order does not affect the final output.

A system exhibits **shift invariance** if, given $x[n] \Rightarrow y[n]$, it is also the case that:

$$x[n + s] \Rightarrow y[n + s]$$

This ensures that the system does not change over time, and though this property is not a requirement for linearity, it is necessary for most DSP techniques. Note that adding positive s shifts the signal *left* relative to its original graph.

When shift invariance is assumed, linear systems demonstrate *static linearity* and *sinusoidal fidelity*. If the system receives an unvarying DC input, **static linearity** requires that it produce a steady output that is equal to the input multiplied by some constant. If the input is a sinusoidal wave, **sinusoidal fidelity** requires that the output be a sinusoidal wave with the same frequency, though possibly one with a different phase or amplitude, including an amplitude of zero. It follows from this that amplitude modulation, frequency modulation, clipping, and slewing are not linear systems. It also follows that non-sinusoidal inputs are likely to change in shape, since they contain sinusoidal components which may be phase-shifted or scaled by different amounts.

3.1 Decomposition

In linear systems, signals can be combined only by shifting, scaling, and then summing them, this process being known as **synthesis**. Separating a signal into two or more additive components is called **decomposition**. By decomposing a complex input signal into simple components, and then understanding the output produced by the components separately, it is possible to determine the output produced by the original complex input.

Impulse decomposition divides a signal of N samples into N components, each containing a single distinct sample from $x[n]$. So, given components $u_i[n]$ for $0 \leq i \leq N-1$, every component sample is zero except for $u_i[i] = x[i]$. This supports *convolution*, which characterizes the system according to how it responds to impulses.

Step decomposition also produces N components, but the first has all samples set to $x[0]$, and the rest contain i zero samples followed by $N - i$ samples equal to $x[i] - x[i - 1]$. In all components, $u_i[i]$ gives the difference between the corresponding sample in x and its predecessor. Because each component contains at most two values, this allows the system to be described in terms of its response to changes in input.

Even/odd decomposition divides the input into two components, one having *even* or *reflective* symmetry about a vertical line at the center of the signal, and one having *odd* or *rotational* symmetry about a point at the center. The even component:

$$u_E[n] = \frac{x[n] + x[N - n]}{2}$$

while the odd component:

$$u_O[n] = \frac{x[n] - x[N - n]}{2}$$

Note that the center is implicitly defined as $N/2$, not $(N - 1)/2$, and the input is assumed to repeat, such that $u[N] = u[0]$. These choices allow Fourier analysis of the signal.

Interlaced decomposition also divides the input into two components, one containing the *even* input samples,

with zeros between them, the other containing the *odd* samples, also with zeros. This decomposition is used during the fast Fourier transform.

Fourier decomposition produces $N+2$ components, half of them sine waves, and half cosines. The first sine and cosine components complete zero cycles over the N samples, so they both constitute DC offsets. The second sine and cosine components complete one cycle over N, the third complete two cycles, et cetera. The component amplitudes vary as necessary to produce the original input. This characterizes the system according to its effect on the amplitude and phase of sinusoidal inputs.

3.2 Non-linear systems

Non-linear systems are not readily analyzed. If the amount of non-linearity is small, the system can be analyzed as if it were linear, with the difference being treated as noise. In particular, many non-linear systems approximate linearity when amplitudes are small. Sometimes it is possible to transform the system into a linear equivalent; **homomorphic processing** uses logarithms to convert non-linear signal products into linear signal sums.

4 Convolution

Non-causal or **acausal** systems allow the output to be affected by sample values that have not yet been received. In **causal** systems, no output sample $y[i]$ is affected by any input sample $x[j]$ where $j > i$; as a result, the impulse response is zero for all sample indices less than zero.

The **delta function** $\delta[n]$ has value one at sample zero, and zeros everywhere else. This is also known as the **unit impulse**. An impulse with sample index s and amplitude a is represented with $a \cdot \delta[n-s]$.

The **impulse response** $h[n]$ is the signal produced by a system in response to the delta function:

$$\delta[n] \Rightarrow h[n]$$

The impulse response of a filter is sometimes known as the **filter kernel** or **convolution kernel**; the response of an image processing system is the **point spread function**. Given a linear, shift-invariant system, and an impulse with any position or amplitude, the output can be represented as a shifted and scaled copy of the impulse response.

The convolution of input $x[n]$ with impulse response $h[n]$ produces output $y[n]$:

$$x[n] * h[n] = y[n]$$

During this process, a copy of $h[n]$ is superimposed at each point i in the output after being scaled by $x[i]$:

$$y[i] = \sum_{j=0}^{N_h-1} x[i-j] \cdot h[j]$$

In this equation, the first sample of the impulse response is scaled by the current sample of the input, while later response samples are scaled by earlier input values, representing the continuation of previous response iterations.

If the input contains N_x samples, and the impulse response N_h samples, the output will contain $N_y = N_x + N_h - 1$ samples. Because the first and last $N_h - 1$ output samples use only part of the impulse response, discontinuities and other distortions may be found at the edges, unless the input is padded with zeros.

Convolution is linear. It is also *commutative*:

$$a[n] * b[n] = b[n] * a[n]$$

associative:

$$(a[n] * b[n]) * c[n] = a[n] * (b[n] * c[n])$$

and *distributive*:

$$a[n] * b[n] + a[n] * c[n] = a[n] * (b[n] + c[n])$$

The distributive property allows a group of parallel systems to be represented by one impulse response that is the sum of the individual responses.

The delta function acts as an identity, so:

$$x[n] * \delta[n] = x[n]$$

and, by extension:

$$x[n] * k\delta[n] = kx[n]$$
$$x[n] * \delta[n-s] = x[n-s]$$

Given the impulse response:

$$h_D[n] = \begin{cases} 0, & \text{for } n < 0 \\ 1, & \text{for } n = 0 \\ -1, & \text{for } n = 1 \\ 0, & \text{for } n > 1 \end{cases}$$

$y_D[n] = x[n] * h_D[n]$ gives the **first difference** or 'discrete derivative' of $x[n]$, this showing the slope at each point of the input. Given:

$$h_I[n] = \begin{cases} 0, & \text{for } n < 0 \\ 1, & \text{for } n \geq 0 \end{cases}$$

$y_I[n] = x[n] * h_I[n]$ produces the **running sum** or 'discrete integral' of $x[n]$. As expected, $h_D[n] * h_I[n] = \delta[n]$.

The same operations can be represented with **recursion equations**, which are also called **difference equations**:

$$y_D[n] = x[n] - x[n-1]$$
$$y_I[n] = x[n] + y[n-1]$$

In general, the impulse response of a low-pass filter contains a series of adjacent positive values, these averaging and smoothing the output. The cutoff frequency is adjusted by changing the width of the series. To produce a filter with unity gain at zero hertz, it is necessary that the sum of the response values equal one.

Since $\delta[n]$ leaves input unchanged, subtracting the values of a low-pass impulse response from $\delta[n]$ produces the response for a high-pass filter. This is analogous to filtering with the original response to isolate the low frequencies, and then subtracting from the original signal. Such a response contains a series of negative values with a single positive discontinuity. To produce a filter with zero gain at zero hertz, it is necessary that the response values add up to zero.

If a roughly pulse-shaped signal is convolved with itself one or more times, a signal with a Gaussian-shaped profile quickly results.

Given $a[n]$ and **target signal** $b[n]$, the **correlation** with $b[n]$ at all points within $a[n]$ can be determined with **matched filtering**, which aligns $b[0]$ with $a[i]$, multiplies corresponding points in the signals, and sums them to produce point $c[i]$:

$$c[i] = \sum_{j=0}^{N_b-1} a[i+j] \cdot b[j]$$

This is equivalent to superimposing the *end* of the *reversed* target signal at each point, after scaling.

This process is equivalent to convolution after reversing $a[n]$ or $b[n]$ around the zero sample, with values before that sample implicitly equal to zero. This is represented as:

$$c[n] = a[n] * b[-n]$$

$c[n]$ is the **cross-correlation** between $a[n]$ and $b[n]$. Correlating a signal with itself produces an **autocorrelation**. Because the signal is convolved with a reversed image of the target, a perfect match produces a symmetrical peak with twice the target width. Given white background noise, this technique produces the greatest possible contrast between output values where a match is found and the signal background where it is not.

5 Discrete Fourier transform

The Fourier transform converts an input signal into a set of cosine and sine waves of varying amplitudes. Sinusoids are useful as components because linear systems are guaranteed to exhibit sinusoidal fidelity. A combination of cosine and sine functions are needed at each point to establish the phase at that frequency.

There are four general types of Fourier transform, one for each combination of *continuous* or *discrete* and *periodic* or *aperiodic* input:

- The **Fourier Series** applies to *continuous, periodic* signals;
- The general **Fourier transform** applies to *continuous, aperiodic* signals;
- The **discrete Fourier transform (DFT)** applies to *discrete, periodic* signals;
- The **discrete time Fourier transform** applies to *discrete, aperiodic* signals.

A discrete signal in one domain is associated with a periodic signal in the other. A continuous signal in one domain is associated with an aperiodic signal in the other. If the time domain signal is periodic, it is analyzed over one period; if it is aperiodic, it is analyzed from negative to positive infinity. When real-number transforms are used for synthesis, only positive frequencies are considered, and these are processed from zero to one half of a cycle for periodic time domain signals, or from zero to positive infinity for aperiodic signals. When complex transforms are used, the negative frequencies are also included.

The time domain signal is assumed to run from negative to positive infinity; this follows from the fact that the sinusoids used to describe the signal themselves cover this

range. Decomposing an aperiodic signal produces an infinite series of sinusoid frequencies, so, in practice, the input buffer is assumed to represent one cycle of an infinite periodic series, and the DFT is used to process it.

All four transforms can be implemented with real or complex numbers. The real DFT converts an N point input $x[n]$ into two $N/2+1$ point outputs, $\operatorname{Re} X[k]$ and $\operatorname{Im} X[k]$. $\operatorname{Re} X[k]$ is the **real part** of the output, and each of its values gives the unnormalized amplitude of one cosine output component. $\operatorname{Im} X[k]$ is the **imaginary part**, and it gives the unnormalized amplitudes of the sine components.

The unscaled components are called **basis functions**:

$$c_k[n] = \cos(2\pi kn/N)$$
$$s_k[n] = \sin(2\pi kn/N)$$

for $0 \le k \le N/2$.

In each function, the number of complete cycles over the N input points is given by k. The basis for the zero-frequency DC offset:

$$c_0[n] = 1$$

At the other end of the spectrum:

$$c_{N/2}[n] = \cos(\pi n)$$

produces one cycle for every two samples, which is the Nyquist frequency, regardless of the rate at which the input is ultimately played. The DC offset and the Nyquist frequency are always represented in the output, and frequencies between them are added as N increases. $s_0[n]$ equals zero and (because its phase causes all samples to coincide with zero crossings) so does $s_{N/2}[n]$. For this reason, both these functions can be ignored.

The frequency variable in a graph of DFT output may be labeled in one of four ways. When integers are displayed, they give the indices of the amplitude functions, $\operatorname{Re} X[k]$ and $\operatorname{Im} X[k]$. When a range from zero to one-half is given, it may be understood as a fraction of the sample rate; this is written as $\operatorname{Re} X[f]$ and $\operatorname{Im} X[f]$, where $f = k/N$. A range from zero to π is the same range using the **natural frequency**, which expresses the frequency in radians per second:

$$\omega = 2\pi f = \frac{2\pi k}{N}$$

This is written as $\operatorname{Re} X[\omega]$ and $\operatorname{Im} X[\omega]$. Finally, the output may be labeled in Hertz, though this is only meaningful relative to a fixed sample rate. Otherwise the DFT is independent of the sample rate, and produces meaningful results regardless of the rate at which the input is actually played.

Given the *normalized* component amplitudes $\operatorname{Re} \overline{X}$ and $\operatorname{Im} \overline{X}$, the input can be recreated with the **DFT synthesis equation**:

$$x[i] = \sum_{k=0}^{N/2} \operatorname{Re} \overline{X}[k] \cdot \cos\left(\frac{2\pi k}{N} i\right) + \sum_{k=0}^{N/2} \operatorname{Im} \overline{X}[k] \cdot \sin\left(\frac{2\pi k}{N} i\right)$$

This process is called the **inverse DFT**. For a given real or imaginary component, it is most easily understood as the summation of a number of sinusoids that have been scaled by values in the spectrum; in this reading, each sinusoid spans the range in the time domain covered by i, and the summation occurs between $N/2+1$ sinusoids having frequency k/N of the sampling rate. However, it can also be read as a series of correlations between the spectrum itself and N sinusoids associated with points in the time domain. In this reading, each sinusoid spans the range in the frequency domain covered by k, and has a frequency equal to i/N of the sample rate.

The normalized amplitudes:

$$\operatorname{Re} \overline{X}[k] = \begin{cases} \frac{1}{N} \operatorname{Re} X[k], & \text{for } k = 0,\ k = N/2 \\ \frac{2}{N} \operatorname{Re} X[k], & \text{for } 0 < k < N/2 \end{cases}$$

$$\operatorname{Im} \overline{X}[k] = -\frac{2}{N} \operatorname{Im} X[k]$$

The **spectral density** at a point in some frequency range is the amount of amplitude at that point per unit of bandwidth. The continuous functions $\operatorname{Re} X$ and $\operatorname{Im} X$ – which are merely sampled by the DFT – describe the spectral density of the input. To convert the density near each point to a sinusoidal amplitude, it is necessary to multiply the density by the bandwidth associated with that point.

$N/2+1$ bands are defined by the DFT. The first and last bands are centered around the zero frequency and the Nyquist frequency, so their widths are half that of the other bands; this gives the inner bands a width of $2/N$ of the total bandwidth, and the outer bands a width of $1/N$. $\operatorname{Im} \overline{X}$ is negated for consistency with the complex DFT.

5.1 Calculating the DFT

$\operatorname{Re} X$ and $\operatorname{Im} X$ can be calculated in any of three ways: by solving simultaneous equations, with correlation, or by using the FFT.

Though there are $N+2$ values in $\operatorname{Re} X$ and $\operatorname{Im} X$ together, the first and last values of $\operatorname{Im} X$ are already known, so N equations are sufficient to solve with simultaneous equations. These are produced by equating the values of $x[n]$ with values from the synthesis function. Because the basis functions are linearly independent, the resultant equations are independent as well. This method is not used in practice.

The DFT is described and calculated in the most general sense with the **DFT analysis equations**:

$$\operatorname{Re} X[k] = \sum_{i=0}^{N-1} x[i] \cdot \cos\left(\frac{2\pi i}{N} k\right)$$
$$\operatorname{Im} X[k] = -\sum_{i=0}^{N-1} x[i] \cdot \sin\left(\frac{2\pi i}{N} k\right)$$

For a given real or imaginary component, this is most easily understood as a series of correlations between the time domain signal and $N/2+1$ sinusoids associated with points in the spectrum. In this reading, each sinusoid spans the range in the time domain covered by i, and has a frequency equal to k/N of the sample rate. However, it can also be read as the summation of a number of sinusoids that have been scaled by values in the time domain; in this reading, each sinusoid spans the range in the spectrum covered by k, and the summation occurs between N sinusoids having frequency i/N of the sampling rate.

More generally, either the synthesis and the analysis equations can be understood as the summation of a group of sinusoids, as scaled by samples in the opposing domain, or as a set of correlations between frequencies associated with points in one domain and a signal in the other.

Two functions are **orthogonal** if they are **uncorrelated**, that is, if the sum of their products over some range is zero. Just as simultaneous equations are solvable only if each is linearly independent, the correlation technique requires that each basis function be orthogonal relative to all others. Other orthogonal functions, including square and triangle waves, can theoretically serve as basis functions.

5.2 Duality

These synthesis and analysis functions are very similar in structure, and in the complex DFT, they are even more similar. This symmetry between domain translations is called **duality**.

Given an impulse input $x[i] = a$:

$$\operatorname{Re} X[k] = a\cos(2\pi ki/N)$$
$$\operatorname{Im} X[k] = -a\sin(2\pi ki/N)$$

When i is non-zero, $\operatorname{Re} X[k]$ and $\operatorname{Im} X[k]$ are sinusoids. When i is zero, $\operatorname{Re} X[k] = a$ and $\operatorname{Im} X[k] = 0$. Since constant values are, in effect, zero-frequency sinusoids, and since each point in the output also represents a sinusoid in the input, it can be said that a single point on one side of the process represents a sinusoid on the other.

Multiplication in the time domain represents convolution in the frequency domain, as in AM synthesis. Conversely, convolution in the time domain represents multiplication in the frequency domain, as demonstrated by any filter and the amplitude response it applies to the input spectrum.

5.3 Polar notation

Because:

$$\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

it is seen that:

$$M\cos(\omega t+\phi) = a\cos\omega t - b\sin\omega t$$

with:

$$a = M\cos\phi$$
$$b = M\sin\phi$$

Since a and b are constant with respect to t, any linear combination of same-frequency sinusoids will produce another same-frequency sinusoid with a different magnitude and phase.

Because:

$$M = \sqrt{a^2+b^2}$$
$$\phi = \arctan(b/a)$$

any DFT basis pair $\operatorname{Re} X[k]$ and $\operatorname{Im} X[k]$ can be represented by a single **polar form** component having:

$$\operatorname{Mag} X[k] = \sqrt{\operatorname{Re} X[k]^2 + \operatorname{Im} X[k]^2}$$
$$\operatorname{Ph} X[k] = \arctan\left(\frac{\operatorname{Im} X[k]}{\operatorname{Re} X[k]}\right)$$

This is analogous to converting a rectangular vector with coordinates $\operatorname{Re} X[k]$ and $\operatorname{Im} X[k]$ to a polar vector. Conversely, results in polar form can be converted to **rectangular coordinates** with:

$$\operatorname{Re} X[k] = \operatorname{Mag} X[k] \cos(\operatorname{Ph} X[k])$$
$$\operatorname{Im} X[k] = \operatorname{Mag} X[k] \sin(\operatorname{Ph} X[k])$$

The polar representation is often easier to understand; $\operatorname{Mag} X$ provides a single amplitude for each frequency k, and the phase graph provides useful information.

By convention, the magnitude in polar coordinates is not allowed to be negative; when a negative value would otherwise be necessary, the phase is increased or decreased by π. This can produce discontinuities in DFT phase output.

6 DFT applications

Increasing the sample count improves the frequency resolution of DFT output, but it does not remove noise from the results; for this, it is necessary to process the output with a low-pass filter. Alternatively, the input can be divided into a number of shorter segments, each of these can be processed with the DFT, and their results averaged; this reduces noise by the square root of the segment count. In both cases, noise is reduced at the cost of frequency resolution.

White noise is uncorrelated from sample to sample, and contains all frequencies at the same amplitude. It appears in DFT output as a relatively flat feature running across the frequency range. Near the Nyquist frequency, antialiasing filter roll-off will be seen. **Pink noise** or **1/f noise** also contains all frequencies, but its spectral density is $1/f$. It is frequently found in natural systems.

To distinguish components that are very near in frequency, it is first necessary that enough input be processed to produce distinct basis functions near the components. It is also necessary that the input cover a sufficient length of time, since similar frequencies present similar profiles when the span is short.

DFT input is theoretically infinite in length, and if it could be processed as such, the output would contain infinitely narrow peaks at each input component. Processing a finite sample implicitly multiplies the infinite signal by a finite window. When signals are multiplied, their spectra are convolved; this replaces the narrow peaks with images of the window spectrum. The finite sample count also quantizes the spectrum. Increasing the sample count improves the resolution, even when the additional samples are outside the window, and thus zero. Though this adds no information to the calculation, it increases the number of basis functions, and decreases their spacing. Of course, the zero samples do not need to be correlated with the basis functions; this is merely a way of increasing resolution within the framework as generally defined.

When an input component fails to align with a single basis function, the output contains a shorter, wider peak between the neighboring basis frequencies, with rounded *tails* surrounding it. The tails represent *spectral leakage*, and their shape and relative amplitude is determined by the spectrum of the window. A *rectangular window* produces the narrowest peak, but it also produces tails with the greatest amplitude. The *Blackman window* produces low-amplitude tails, but it also creates a wide peak. The *Hamming window* produces tails of moderate amplitude and a peak of moderate width.

6.1 Frequency response

Just as the effect of a linear system $x[n] \Rightarrow y[n]$ is defined by its impulse response, $h[n]$:

$$x[n] * h[n] = y[n]$$

it is also defined by its **frequency response** $H[f]$, which describes the way the system changes the amplitude and phase of cosine input components:

$$X[f] \times H[f] = Y[F]$$

The frequency response is the Fourier transform of the impulse response. As a result, convolution in the time domain is equivalent to multiplication in the frequency domain, and vice versa.

Although the impulse response is a discrete signal, a system's frequency response is necessarily continuous, since any frequency might be input to the system; a finite-length DFT merely samples the actual response. Padding the impulse response with zeros before the DFT produces a smooth curve that approaches the actual shape.

In polar form, the product of two spectra is found by multiplying magnitudes and adding phase values:

$$\operatorname{Mag} Y[f] = \operatorname{Mag} X[f] \cdot \operatorname{Mag} H[f]$$
$$\operatorname{Ph} Y[f] = \operatorname{Ph} X[f] + \operatorname{Ph} H[f]$$

Conversely, a quotient is produced by dividing and subtracting:

$$\begin{aligned}\operatorname{Mag} H[f] &= \frac{\operatorname{Mag} Y[f]}{\operatorname{Mag} X[f]} \\ \operatorname{Ph} H[f] &= \operatorname{Ph} Y[f] - \operatorname{Ph} X[f]\end{aligned}$$

In rectangular form, the product:

$$\begin{aligned}\operatorname{Re} Y[f] &= \operatorname{Re} X[f] \cdot \operatorname{Re} H[f] - \operatorname{Im} X[f] \cdot \operatorname{Im} H[f] \\ \operatorname{Im} Y[f] &= \operatorname{Im} X[f] \cdot \operatorname{Re} H[f] + \operatorname{Re} X[f] \cdot \operatorname{Im} H[f]\end{aligned}$$

and the quotient:

$$\begin{aligned}\operatorname{Re} H[f] &= \frac{\operatorname{Re} Y[f] \operatorname{Re} X[f] + \operatorname{Im} Y[f] \operatorname{Im} X[f]}{\operatorname{Re} X[f]^2 + \operatorname{Im} X[f]^2} \\ \operatorname{Im} H[f] &= \frac{\operatorname{Im} Y[f] \operatorname{Re} X[f] - \operatorname{Re} Y[f] \operatorname{Im} X[f]}{\operatorname{Re} X[f]^2 + \operatorname{Im} X[f]^2}\end{aligned}$$

6.2 Convolution with the DFT

Convolution can be performed by multiplying $X[f]$ by $H[f]$ and then resynthesizing with the inverse DFT; when the FFT is used, this can be much faster than direct convolution. **Deconvolution** produces $x[n]$ from $y[n]$ and $h[n]$; it can be performed by dividing $Y[f]$ by $H[f]$ and then resynthesizing.

Convolving a signal of N samples with one of M samples produces an output of $N + M - 1$ samples. Using the DFT to perform convolution produces an output of $\max(N, M)$ samples. If N_u and M_u are the unpadded lengths of the two signals, and if $N_u + M_u - 1$ is greater than $\max(N, M)$, the inverse DFT will be too short to show the convolved signal accurately. As seen from the synthesis function, the inverse DFT repeats after N samples, since the basis functions themselves repeat. If the output length is too short to accommodate $N_u + M_u - 1$, **circular convolution** will occur; the end of the ideal convolved signal will overlap the beginning to produce a periodic signal of length $\max(N, M)$. This is avoided by padding the input and the impulse response with zeros until $\max(N, M)$ equals or exceeds $N_u + M_u - 1$.

7 Properties of the Fourier transform

Using the Fourier transform, if $x[n] \Rightarrow X[f]$, it must be true that $kx[n] \Rightarrow kX[f]$, since all input components are scaled evenly by k. From this it follows that the transform is *homogeneous*. In rectangular form, both the real and imaginary values are multiplied by k; in polar form, only the magnitude is.

If $a[n] \Rightarrow A[f]$, $b[n] \Rightarrow B[f]$, $c[n] \Rightarrow C[f]$, and:

$$a[n] + b[n] = c[n]$$

it follows that:

$$\begin{aligned}\operatorname{Re} A[f] + \operatorname{Re} B[f] &= \operatorname{Re} C[f] \\ \operatorname{Im} A[f] + \operatorname{Im} B[f] &= \operatorname{Im} C[f]\end{aligned}$$

since the cosine and sine components at each frequency combine without affecting the others. This shows that the Fourier transform is *additive*. Only in rectangular form can the real and imaginary values be added; this is not possible in polar form because their phases might differ.

Since the Fourier transform is both homogeneous and additive, it is also *linear*. It is not shift invariant, however. If f is the frequency as a fraction of the sample rate, and:

$$x[n] \Rightarrow \operatorname{Mag} X[f] \text{ and } \operatorname{Ph} X[f]$$

it must be true that:

$$x[n + s] \Rightarrow \operatorname{Mag} X[f] \text{ and } \operatorname{Ph} X[f] + 2\pi f s$$

This follows from the fact that, for frequency F in cycles per second, the angular frequency, in radians per second, is $2\pi F$. If F_s is the sample rate, then the time represented by s:

$$t = s/F_s$$

Multiplying the angular frequency by time produces the angular displacement:

$$\theta = 2\pi F t = \frac{2\pi F}{F_s} s$$

Since $F/F_s = f$:

$$\theta = 2\pi f s$$

As s increases, the signal shifts to the left, and the slope of the phase graph $\operatorname{Ph} X[f] + 2\pi f s$ increases. The change in slope is consistent with the fact that, for a given time displacement, the phase change is greater for high frequency components, since they have shorter periods.

By definition, all basis functions complete a whole number of cycles within the span covered by the DFT; therefore, all $\operatorname{Ph} X$ slopes produced by various $s = kN$ are equivalent when k is a whole number. In particular, at each frequency

in the DFT output, the phase of these graphs will differ by an integer multiple of 2π. Alternatively, because DFT input is implicitly periodic, increasing s causes samples near the beginning of the input to be wrapped to the end, and when k is a whole number, the input is wrapped back to its original position. It would seem that points between the DFT frequencies differ by non-integer multiples, but it must be remembered that the DFT produces point values, not functions, and that graphs of DFT output are merely interpolations.

A signal with left-right symmetry at any point is said to be a **linear phase** signal, and its phase graph is a straight line over f. A signal that is symmetric about the zero sample is called a **zero phase** signal, and the slope of its phase graph is zero. Because DFT input is periodic, a signal that is symmetric about sample $N/2$ is necessarily symmetric about zero as well. Signals without even symmetry have **non-linear phase**, and their phase graphs are not straight.

The spectral characteristics that produce sharp rising or falling edges are concentrated in the phase, since edges are created when multiple components rise or fall at the same time.

Given:

$$\begin{aligned} X[f] &= \operatorname{Re} X[f] \text{ and } \operatorname{Im} X[f] \\ &= \operatorname{Mag} X[f] \text{ and } \operatorname{Ph} X[f] \end{aligned}$$

the **complex conjugate** of $X[f]$:

$$\begin{aligned} X^*[f] &= \operatorname{Re} X[f] \text{ and } -\operatorname{Im} X[f] \\ &= \operatorname{Mag} X[f] \text{ and } -\operatorname{Ph} X[f] \end{aligned}$$

Negating the phase values reverses the direction of the signal in the time domain, so if $x[n] \Leftrightarrow X[f]$, then $x[-n] \Leftrightarrow X^*[f]$. This relates the convolution $a[n]*b[n] \Leftrightarrow A[f]\times B[f]$ with the correlation $a[n] * b[-n] \Leftrightarrow A[f] \times B^*[f]$.

When spectra are multiplied, their magnitudes are multiplied and their phases added. Given any signal $x[n]$, a zero phase signal can be produced with $X[f]\times X^*[f]$, since this cancels all phase values. The new signal must equal $x[n]*x[-n]$, so convolving any signal with its reverse image produces a signal that is symmetric about the zero sample.

Time domain aliasing results during the inverse DFT when modifications to the frequency domain produce a new ideal signal with length greater than N; because the signal is implicitly periodic, the end overlaps the beginning. Circular convolution is an example of this type of aliasing.

Mathematically, the frequency range from zero to the Nyquist frequency is mirrored around the zero sample, and this symmetrical image is repeated in both positive and negative directions. Proceeding from zero in the positive direction, the audible spectrum is repeated once in the forward direction, once in reverse, again in the forward direction, and so on. The frequency spectrum as a whole is symmetric about zero, giving it **even symmetry**. When a component is decreased below zero or increased above the Nyquist frequency, its mirror image in the audible range moves in the opposite direction, making it seem that the frequency has been 'reflected'. The curves of the negative and higher positive frequencies fit the input samples with the same precision that the audible frequencies do.

The phase range from zero to the Nyquist frequency also repeats this way, but the reversed images are also negated in sign. This gives the phase spectrum rotational or **odd symmetry**. Phase components also reflect from the zero frequency and the Nyquist frequency.

When two spectra are convolved, frequency zero in one of them is superimposed over frequencies in the other; this transposes the entire spectrum, causing negative frequencies to enter the audible range. This explains why amplitude modulation produces the sums and differences of the input frequencies: the sums are created when positive frequencies are shifted to new locations relative to a frequency in the other signal, while the differences are created when negative frequencies are shifted this way. The region in the new spectrum corresponding to previously negative frequencies is called a **lower sideband**; the region corresponding to positive frequencies is called an **upper sideband**.

If a continuous signal is 'expanded' in time, the spectrum will be compressed within the frequency range by a like amount; specifically, given $x(t) \Leftrightarrow X(f)$:

$$x(kt) \;\Leftrightarrow\; \frac{1}{k} \times X(f/k)$$

An analogous relationship applies to discrete signals. Expanding the signal relative to the sample rate is comparable to sampling the original signal at a higher sample rate.

More generally, events that happen faster are composed of higher frequencies. In the extreme case, the spectrum of an impulse is found to be a constant amplitude covering all frequencies. Compressing a signal in the time domain can cause aliasing in the frequency domain; conversely, compressing a signal in the frequency range can cause aliasing in the time domain.

Just as the resolution of the spectrum can be improved by

padding the time domain with zeros before the DFT, the resolution of the signal can be improved by padding the end of the frequency domain with zeros before the inverse DFT. Because the synthesis function always produces frequencies that run from zero to f_s, padding lowers the effective frequencies of the non-zero values. The new signal can be interpreted as a spectrum-perfect resampling of the original input at a higher sample rate. As when DFT input is padded, no information is introduced; instead, the existing components are sampled with greater precision. Interpolation can also be performed by inserting zeros between existing samples and then low-pass filtering.

Since the time and frequency domain representations are equivalent, they must have the same energy. This yields **Parseval's Relation**:

$$\sum_{i=0}^{N-1} x[i]^2 = \frac{2}{N} \sum_{k=0}^{N/2} \operatorname{Mag} X[k]^2$$

7.1 Discrete time Fourier transform

The discrete time Fourier transform processes aperiodic discrete signals. Padding DFT input with zeros increases the input length and the number of basis functions while decreasing the distance between each function; by extension, padding until the signal has infinite length turns it aperiodic and makes the output continuous. This produces the **DTFT analysis equations**:

$$\operatorname{Re} X(\omega) = \frac{1}{\pi} \sum_{i=-\infty}^{\infty} x[i] \cos(\omega i)$$

$$\operatorname{Im} X(\omega) = -\frac{1}{\pi} \sum_{i=-\infty}^{\infty} x[i] \sin(\omega i)$$

The input remains discrete, and the output periodic. In the DFT analysis equations, frequency is represented by $2\pi k/N$, with k ranging from zero to $N/2$. For brevity, the frequency is here represented with the natural frequency ω, which ranges from zero to π.

The **DTFT synthesis equation**:

$$x[i] = \int_0^{\pi} \operatorname{Re} X(\omega) \cos(\omega i) - \operatorname{Im} X(\omega) \sin(\omega i) \, \mathrm{d}\omega$$

The DFT characterizes both domains with samples. If the time domain is described with an equation, the DTFT allows the frequency domain to be described in like manner. The DTFT does nothing to reduce aliasing, however, as the input remains in discrete form.

8 Fourier transform pairs

If $x[n] \Leftrightarrow X[f]$, then $x[n]$ and $X[f]$ are **Fourier transform pairs**. Unless aliasing interferes, if waveform $a[n]$ in the time domain produces $b[f]$ in the frequency domain, then $b[n]$ in the time domain will produce something very similar to $a[f]$.

8.1 Delta function

An impulse in one domain produces a sinusoid with possibly zero frequency in the other.

An impulse at sample zero in the time domain produces a spectrum with constant magnitude and zero phase across all frequencies. This conforms with the observation that compression in one domain causes expansion in the other; an impulse is a maximally compressed signal, and a flat line is a maximally expanded spectrum. As the impulse is shifted to the right, the slope of the phase decreases, while the magnitude remains unchanged.

At sample zero, an impulse produces a spectrum with constant non-zero real values and imaginary values equal to zero. As the impulse is shifted to the right, the real values take the form of a cosine wave, and the imaginary values, that of a sine. In both cases, the number of cycles spanning the frequency range from zero to the sampling rate is equal to the sample number where the impulse occurs. This is consistent with the way the analysis equations work; just as the synthesis function mixes a number of sinusoids with amplitudes equal to values in the spectrum, the analysis equation mixes sinusoids with amplitudes equal to successive values in the signal and frequencies proportional to the sample number.

8.2 Sinc function

The **normalized sinc function**:

$$\operatorname{sinc}(x) = \begin{cases} \dfrac{\sin(\pi x)}{\pi x}, & \text{for } x \neq 0 \\ 1, & \text{for } x = 0 \end{cases}$$

A rectangular pulse in the time domain produces a sinc waveform in the frequency domain, and vice versa. When the pulse is centered around sample zero, the phase alternates regularly between zero and π; this represents the

negative ranges in the sinc function, since the magnitude is meant to remain positive.

The sinc function has infinite width, so aliasing always results. Given an N point signal with a zero-centered unit amplitude rectangular pulse M samples wide:

$$\operatorname{Mag} X[k] = \begin{cases} \left|\dfrac{\sin(\pi k M/N)}{\sin(\pi k/N)}\right|, & \text{for } k \neq 0 \\ M, & \text{for } k = 0 \end{cases}$$

The sine term in the denominator is the result of aliasing; without aliasing, the denominator would be $\pi k/N$. $\sin(x)$ is very close to x when x is near zero, so at low frequencies, the aliasing is minimal; at the Nyquist frequency, the magnitude is approximately 57% greater.

Using the DTFT:

$$\operatorname{Mag} X(f) = \begin{cases} \left|\dfrac{\sin(\pi f M)}{\sin(\pi f)}\right|, & \text{for } f \neq 0 \\ M, & \text{for } f = 0 \end{cases}$$

The zero values in the magnitude are found at frequencies that fit an integer number of cycles within the pulse width; because the sum of a sinusoid over one cycle is zero, these frequencies have no correlation with the pulse. By the same token, an impulse must contain all frequencies, since a single sample can be correlated with any frequency.

When performing the DFT, selecting a finite set from the theoretically infinite range of input samples implicitly convolves the signal spectrum with the sinc function. Increasing the number of input samples lengthens the rectangular window, which compresses the sinc function and causes the spectrum at each component frequency to approach the impulse ideal. Padding with zeros merely increases the resolution.

A rectangular pulse in the frequency domain corresponds to a sinc function in the time domain, and when the inverse DFT is used, time domain aliasing necessarily results. Given a unit-amplitude pulse covering frequencies zero through $M-1$, the aliased time domain signal:

$$x[i] = \begin{cases} \dfrac{2M-1}{N}, & \text{for } i = 0 \\ \dfrac{1}{N} \cdot \dfrac{\sin(2\pi i(M-1/2)/N)}{\sin(\pi i/N)}, & \text{for } i \neq 0 \end{cases}$$

Using the inverse DTFT eliminates aliasing, since the time domain is infinite. If the pulse has unit amplitude and runs from zero to frequency f_c:

$$x[i] = \begin{cases} 2f_c, & \text{for } i = 0 \\ \dfrac{\sin(2\pi f_c i)}{\pi i}, & \text{for } i \neq 0 \end{cases}$$

This is the impulse response of an ideal low-pass filter, and is used to implement the *windowed-sinc filter*.

8.3 Other transform pairs

Convolving a rectangular pulse of length M with itself produces a triangular pulse of length $2M-1$. Multiplying in the frequency domain produces a spectrum that is the square of the sinc function representing the original pulse.

When aliasing is ignored, a Gaussian curve in the time domain produces a zero-centered Gaussian in the frequency domain. If σ_t and σ_f are the standard deviations in the time and frequency domains, then $1/\sigma_t = 2\pi\sigma_f$.

A **Gaussian burst** is the product of a Gaussian curve and a sine wave. Because the sine wave produces an impulse within the spectrum, the implicit convolution moves the Gaussian to a new position equal to the frequency of the sine.

8.4 Gibbs effect

The **Gibbs effect** is the overshoot and ringing that occurs near sharp edges in the time domain when an ideal waveform is approximated with additive synthesis. As frequency components are added, the width of the overshoot decreases, but the amplitude remains approximately constant. In a continuous signal, the overshoot never decreases significantly in height, but its width approaches zero, giving it zero energy.

8.5 Harmonics

In a periodic signal with fundamental frequency f, all component frequencies must be integer multiples of f, since any other frequency would produce a period that does not fit evenly within that of the signal. Conversely, adding two signals can only produce a period equal to or longer than the source periods, and a fundamental frequency equal to or lower than the source frequencies.

If a recurring waveform has been modified with clipping or any other waveshaping function, any new frequencies in the spectrum must be harmonics, since the fundamental frequency has not changed. If the waveform has odd symmetry, such that the peaks and troughs present identical profiles, the signal will contain only odd harmonics.

A discrete signal in either domain necessarily represents harmonics in the other, since the synthesis and analysis functions use only harmonics, and there is no way to represent between-sample frequencies. This explains why the DFT is periodic in the time domain, while the DTFT is not. The DFT represents the signal as a finite number of harmonics that necessarily repeat when the fundamental repeats. By contrast, the DTFT represents the signal as an infinite number of frequencies. If this signal had a period, it would be the least common multiple of the component periods. Since there is no finite multiple of all possible periods, so there is no fundamental period or frequency.

8.6 Chirp signals

In the time domain, a **chirp signal** is a short oscillating pulse that increases in frequency and then rapidly fades out. Its spectrum has unit magnitude, like that of a unit impulse, with a parabolic phase curve:

$$\operatorname{Ph} X[k] = \alpha k + \beta k^2$$

The value α determines the slope of the phase graph, and thus the position of the chirp. α and β must be chosen such that the phase at the zero and Nyquist frequencies is a multiple of 2π.

In radar systems, the power required to produce a pulse varies inversely with the pulse length; longer signals, like the chirp, thus require less power than would a single impulse. When signals are convolved, their magnitudes are multiplied and their phases added. Convolving a chirp with its own complex conjugate thus produces a unit magnitude and a constant zero phase, which is the spectrum of an impulse. A radar system can broadcast a chirp and then convolve the echo to produce impulses representing the targets of the pulse.

9 Fast Fourier transform

Calculating the DFT with correlation produces $O(n^2)$ time complexity; the same results are produced by the FFT in $O(n \log n)$. This relationship holds for the inverse operations as well.

The complex DFT accepts N complex numbers, with the real parts set to the signal values, and the imaginary parts set to zero. It also returns N complex numbers, with the first $N/2+1$ of these corresponding to the values produced by the real DFT, and the remaining values representing negative frequencies.

The FFT derives from the complex DFT. The analysis function in the complex DFT:

$$X[k] = \sum_{i=0}^{N-1} x[i] \cdot \mathrm{e}^{\frac{-2\pi j}{N} ik}$$

can be divided into two sums, one that covers the even elements, and one that covers the odd:

$$X[k] = \sum_{i=0}^{N/2-1} x[2i] \cdot \mathrm{e}^{\frac{-2\pi j}{N}(2i)k} + \sum_{i=0}^{N/2-1} x[2i+1] \cdot \mathrm{e}^{\frac{-2\pi j}{N}(2i+1)k}$$

If $E[k]$ is the DFT of the even elements, and $O[k]$ that of the odd, it follows that:

$$X[k] = E[k] + \mathrm{e}^{\frac{-2\pi j}{N}k} \cdot O[k]$$

If N is a power of two, the process can be applied recursively to produce $N/2$ DFTs of length two, which can then be calculated directly.

9.1 Real FFT

Ordinarily, the real parts of the complex DFT input are used to store the time domain values, while the imaginary parts are set to zero; this produces even symmetry in the real output and odd symmetry in the imaginary output. If the time values are instead stored in the imaginary part, the imaginary output displays even symmetry, while the real output displays odd.

The **real FFT** exploits this relationship by storing the even input samples in the real parts of the input, and the odd samples in the imaginary parts; this halves the FFT length and produces spectra that are the sum of the even and odd sample spectra. Even/odd decomposition splits a signal into two parts, one with even symmetry, and one with odd. Applying this to the FFT output produces the spectra of the original even and odd inputs; these can then

be joined the way even and odd sample spectra are joined in the FFT, producing finished output with almost twice the speed of a normal FFT.

10 Continuous signal processing

Linear electronic components include resistors, capacitors, and inductors. When a linear circuit accepts a very short pulse, the shape of the output is determined by the construction of the circuit, not the shape of the pulse, and the amplitude varies with the net positive area of the pulse. Any input short enough to produce this behavior can be called an **impulse**.

The **continuous delta function** $\delta(t)$ is an impulse at time zero with an infinitely short length and an area of one. Because the width is infinitesimal, the amplitude is theoretically infinite.

10.1 Convolution

At each point, the effect of convolution can be visualized with a reversed image of the impulse response. After aligning the end with the current input sample, all samples are multiplied with the corresponding input values and then summed; this accounts for the way that later samples in the response are scaled by previous input values. Convolution between continuous signals can be understood in a like manner:

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(t-\tau)h(\tau)\ \mathrm{d}\tau$$

Because convolution is commutative, this can also be written:

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)\ \mathrm{d}\tau$$

Continuous convolution can be visualized with a reversed image of the continuous impulse response, its end advancing through the input as t increases, with the output at each point equal to the area under the product of the signals. Typically, the integral must be solved by dividing the problem into regions, one for the range where the impulse response overlaps the beginning of the input, one where the signals completely overlap, and one where the response overlaps the end. In all cases the integration range is chosen to cover the intersection of the signals, where both are defined.

Complex convolution problems can be solved by applying a linear process that simplifies one of the signals, convolving, and then reversing the simplifying operation. Integration and differentiation are themselves linear processes, since they are both homogeneous and additive. Output can therefore be calculated by convolving with the derivative of one of the signals, and then computing the integral of the convolution. The derivative of a rectangular pulse is a single positive impulse followed by an offsetting negative impulse. Convolving an input with this response produces one image of the input combined with a time-shifted and negated image; integrating this output over two ranges produces the final result. When this is done, the DC offset must be calculated by other means, since it is lost during differentiation.

10.2 Fourier transform

The Fourier transform applies to signals that are continuous and aperiodic, like the impulse response of a filter. The **Fourier transform synthesis equation**:

$$x(t) = \int_0^{\infty} \operatorname{Re} X(\omega)\cos(\omega t) - \operatorname{Im} X(\omega)\sin(\omega t)\ \mathrm{d}\omega$$

The **Fourier transform analysis equations**:

$$\operatorname{Re} X(\omega) = \frac{1}{\pi}\int_{-\infty}^{\infty} x(t)\cos(\omega t)\ \mathrm{d}t$$
$$\operatorname{Im} X(\omega) = -\frac{1}{\pi}\int_{-\infty}^{\infty} x(t)\sin(\omega t)\ \mathrm{d}t$$

10.3 Fourier Series

The Fourier series applies to signals that are continuous and periodic; these contain only harmonic frequencies, since enharmonic cycles would not fit within the signal's period.

Given fundamental frequency f, the **Fourier series synthesis equation**:

$$x(t) = \operatorname{Re} X[0] + \sum_{k=1}^{\infty} \operatorname{Re} X[k]\cos(2\pi f k t) - \sum_{k=1}^{\infty} \operatorname{Im} X[k]\sin(2\pi f k t)$$

Given signal period $T = 1/f$, the **Fourier series analysis equations**:

$$\operatorname{Re} X[0] = \frac{1}{T}\int_0^T x(t)\ \mathrm{d}t$$
$$\operatorname{Re} X[k] = \frac{2}{T}\int_0^T x(t)\cos\left(\frac{2\pi kt}{T}\right)\ \mathrm{d}t$$
$$\operatorname{Im} X[k] = \frac{-2}{T}\int_0^T x(t)\sin\left(\frac{2\pi kt}{T}\right)\ \mathrm{d}t$$

$\operatorname{Re} X[0]$ gives the DC offset, while $\operatorname{Im} X[0]$ is always zero. Since the signal is periodic, the correlations need only be calculated over a single period.

Given a **pulse train** with amplitude A, pulse width w, and **duty cycle** $d = w/T$:

$$\operatorname{Re} X[0] = Ad \qquad \operatorname{Re} X[k] = \frac{2A}{\pi k}\sin(\pi kd)$$
$$\operatorname{Im} X[k] = 0$$

The pulses rise from zero, so the signal has a DC offset that is proportional to the duty cycle. The first pulse is centered around time zero, and the resultant symmetry produces a zero phase spectrum, which itself yields a zero imaginary spectrum, since $\operatorname{Im} X(t) = \operatorname{Mag} X(t) \cdot sin(\operatorname{Ph} X(t))$ is zero for zero $\operatorname{Ph} X(t)$. This can also be understood geometrically: since the waveform is symmetrical about the time axis, it must consist of even components, such as cosine waves. If the waveform had odd symmetry, it would instead contain odd components, like sine waves.

Given a **square wave**:

$$\operatorname{Re} X[k] = \frac{2A}{\pi k}\sin\left(\frac{\pi k}{2}\right) \qquad \operatorname{Im} X[k] = 0$$

Given a **triangle wave**:

$$\operatorname{Re} X[k] = \frac{4A}{(\pi k)^2}\sin\left(\frac{\pi k}{2}\right) \qquad \operatorname{Im} X[k] = 0$$

Given a **sawtooth wave**:

$$\operatorname{Re} X[k] = 0 \qquad \operatorname{Im} X[k] = \frac{A}{\pi k}$$

Given a **rectified sine wave**:

$$\operatorname{Re} X[0] = \frac{2A}{\pi} \qquad \operatorname{Re} X[k] = \frac{-4A}{\pi(4k^2-1)}$$
$$\operatorname{Im} X[k] = 0$$

In electronics, the Fourier series is used to implement **frequency multiplication**. A lower-frequency component like a crystal is used to produce a sinusoidal output, which can be clipped or squared to introduce harmonics that are precise multiples of the fundamental. These are then isolated with a band-pass filter.

11 Digital filters

Analog filters are fast, and can be made to handle a wide range of amplitudes and frequencies, but their filtering characteristics are limited by the accuracy and stability of their components. Digital filters can produce vastly superior filtering characteristics.

A filter's **step response** or **edge response** is the output produced by a step input; it shows how time domain information is modified by the filter. Because an impulse is the derivative of a step, the step response is equal to the integral of the impulse response. A filter can be completely described by its impulse response, step response, or frequency response, and if any one of these is known, the others can be calculated.

Finite impulse response filters are implemented with processes equivalent to convolution. A filter can also be implemented with **recursion**, which weights and sums input values the way convolution does, but also includes weighted output values. This creates an **infinite impulse response** filter, with an impulse response containing exponentially-decaying sinusoids. The characteristics of an IIR filter are defined by its **recursion coefficients**.

11.1 Filter characteristics

In the time domain, an ideal filter exhibits fast step response and no overshoot, and has linear phase. In the frequency domain, it exhibits fast roll-off, strong stopband attenuation, and has a flat passband. Good characteristics in the time domain produce poor results in the frequency domain, and vice versa.

The speed of a step response is described by its **rise time**, often defined as the time to transition from 10% to 90% of the rise. To discern short time domain events in a filtered signal, the step response must be shorter than the events themselves.

Overshoot in the step response distorts amplitudes in the time domain.

An impulse response with even symmetry produces a step response with odd symmetry about its middle point; a filter

with such a response has **linear phase**. Given a fixed time displacement, angular displacement varies linearly with frequency. Therefore, a linear phase filter is one that displaces all frequencies by the same amount of time, causing no phase distortion.

11.2 Manipulating filters

A filter of one type can be transformed into another with **spectral inversion**; this is performed by negating the values in a symmetrical impulse response $h[n]$ and then adding one to the sample at the center of symmetry. Negating $h[n]$ reverses the original output vertically; adding one introduces the delta function, so that an unprocessed image of the input is included:

$$y[n] = x[n] * (\delta[n] - h[n])$$

This causes frequencies that would have been passed to be cancelled instead. The frequency response of the new filter is reversed vertically relative to the original, so that passbands become stopbands over their original ranges, and stopbands become passbands. For this to be effective, the original filter must not alter the phase of low frequency components, or they will be incompletely cancelled when combined with the input. The delta impulse is added to the middle of the response to maintain the even symmetry that this implies.

Another technique is **spectral reversal**; it is performed by negating the sign of alternating samples in the impulse response. This is equivalent to multiplying the impulse response by a Nyquist-frequency sinusoid, which implicitly convolves the frequency response with that frequency, causing the original response to be replaced by its own negative frequency range. This reverses the frequency response horizontally, causing passbands on one side to become passbands of like size and position on the other.

A band-pass filter can be constructed by *convolving* a low-pass filter and a high-pass filter with *overlapping* passbands, such that their effects are processed in series:

$$y[n] = x[n] * (h_L[n] * h_H[n])$$

A band-stop filter can be constructed by *adding* a low-pass filter and a high-pass filter *without* overlapping passbands, such that their effects are processed in parallel:

$$y[n] = x[n] * (h_L[n] + h_H[n])$$

Filters are selected according to their intended use and the desired implementation. In the time domain, filters are used to smooth or shape waveforms, or remove DC offsets; in the frequency domain, they are used to isolate components. Certain special applications also exist, such as deconvolution:

Domain	**Convolution**	**Recursion**
Time	Moving average	Single pole
Frequency	Windowed-sinc	Chebyshev
(other)	Custom FIR	Iterative design

Convolution produces superior filtering characteristics, but it is slower.

12 Moving average filters

The **moving average filter** optimally removes random noise while maintaining a fast step response:

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j]$$

To produce a filter that does not shift the output relative to the input, the averaging window can be made symmetric about the input sample.

The sum within the average implicitly convolves the input with a unit-area rectangular pulse, producing a flattened aliased sinc function in the frequency response:

$$H[f] = \frac{1}{M} \frac{\sin(\pi f M)}{\sin(\pi f)}$$

Input noise is attenuated by a factor equal to the square root of M. The rise time from 0% to 100% is equal to M, since that is the time for the output to become stable after a step transition. Because averaging optimally removes random variations, this filter produces the lowest possible noise for a filter with that rise time. The filter is very good at smoothing, but it has a low roll-off and very poor stopband attenuation.

The moving average filter can be greatly optimized with a recursive implementation. After calculating the first output sample:

$$y[i] = y[i-1] + \frac{x[i+M-1] - x[i-1]}{M}$$

Unlike most recursive implementations, this does not produce an infinite impulse response.

12.1 Similar filters

A **multiple-pass moving average filter** is implemented by passing the input two or more times through a moving average filter; alternatively, the filter kernel can be convolved with itself to produce the same effect with a single operation. Convolving a rectangular pulse with itself produces a triangular pulse with length $2M - 1$; as the pulse is further convolved, its length increases, and its shape approaches a Gaussian curve. Each iteration smooths the corners of the step response, and causes the frequency response to be multiplied by the original moving average frequency response.

Compared to the moving average filter, the multiple-pass moving average, the Gaussian filter, and the filter produced by the Blackman window produce similar noise reduction for a given rise time, but offer better stopband attenuation.

13 Windowed-Sinc filters

Windowed-sinc filters have good frequency domain characteristics, but they produce significant overshoot in the step response, and they are slow when implemented with convolution. Performance can be improved with FFT convolution.

In the frequency domain, the ideal low-pass filter is a rectangle centered around zero. Applying the inverse DFT produces an impulse response containing the sinc function:

$$h[i] = \frac{\sin(2\pi f_C i)}{i\pi}$$

with f_C specifying the middle of the transition band, where amplitude is one-half. The sinc function is infinite in length, so it must be truncated or windowed, which produces a **windowed-sinc filter**. Truncation convolves the frequency response with a sinc function, producing ripple in the pass and stop bands and reducing the stopband attenuation to -21dB (8.9% of amplitude). The frequency response of a windowed-sinc filter has odd symmetry, so passband ripple, as a percentage of amplitude, is equal to the stopband level.

If M is an even number, and the sinc function is symmetric about $M/2$, and truncated below sample zero and above sample M, the kernel can be multiplied in the time domain by a **Blackman window**:

$$w[i] = 0.42 - 0.5\cos\left(\frac{2\pi i}{M}\right) + 0.08\cos\left(\frac{4\pi i}{M}\right)$$

This largely eliminates overshoot and stopband ripple, and improves the stopband attenuation to -74bB (0.02% of amplitude). It also rounds the corners in the frequency response, and decreases the roll-off to 40% of the value produced by truncation.

The **Hamming window** has a similar effect:

$$w[i] = 0.54 - 0.46\cos\left(\frac{2\pi i}{M}\right)$$

The roll-off is 20% faster than the Blackman window, but the stopband attenuation drops to -53bB (0.2% of amplitude).

The **Bartlett window** is a simple triangle covering the sinc range. Its roll-off is similar to the Hamming window, and its stopband attenuation is -25dB (5.6% of amplitude).

The **Hanning** or **raised cosine window**:

$$w[i] = 0.5 - 0.5\cos\left(\frac{2\pi i}{M}\right)$$

This also has roll-off similar to the Hamming window, and stopband attenuation of -44dB (0.63% of amplitude).

In any windowed-sinc filter, expanding the sinc function and its containing window compresses the transition band, which increases roll-off. Though the exact width depends on the choice of window, the width of the transition band, as a fraction of the sample rate:

$$BW \approx \frac{4}{M}$$

The shape and width of the transition do not vary with the cutoff.

In general, the kernel for a windowed-sinc filter:

$$h[i] = \begin{cases} K\dfrac{\sin(2\pi f_C(i - M/2))}{i - M/2} \cdot w[i], & \text{for } i \neq 0 \\ \\ 2\pi f_C K \cdot w[i], & \text{for } i = 0 \end{cases}$$

K is selected to provide unity gain at the zero frequency; this is done by summing the unnormalized values in the kernel, and then dividing the values by the sum.

Stopband attenuation can be improved by passing the signal through the filter more than once; with each pass the effective roll-off is lowered, but another increment of the original attenuation is achieved. This can also be accomplished by convolving the kernel with itself.

14 Custom filters

The inverse DFT allows a filter to be constructed for almost any frequency response. First, the spectrum is sampled in magnitude and phase or in its real and imaginary parts. As always, when sampling the phase, the first and last values must be multiples of 2π. After the inverse DFT, the impulse response can be shifted, truncated, and windowed. Depending on the original frequency response, time domain aliasing may result in the new kernel. Truncation and windowing can reduce the effect of this aliasing, and shorten the required convolution.

14.1 Deconvolution

If events in some signal are contaminated by an unwanted convolution, **deconvolution** can be used to restore them.

To start with, the unwanted convolution must be known; then a pulse must be chosen to represent the reconstructed events. After finding the frequency response of the convolution and the new pulse, the response of the pulse is divided by that of the convolution to produce the frequency response of a correcting filter. The inverse DFT is then used to obtain an impulse response for this filter, which can be truncated and windowed to produce a kernel.

The pulse used to represent the events must not be too short, because short pulses require high frequency components that presumably have low levels in the contaminated signal. This being the case, the correcting frequency response would have to amplify these frequencies very strongly, causing discrepancies between the estimated convolution and the actual convolution to produce large errors in the output.

If frequencies needed to reconstruct the signal have been attenuated below the noise floor, adequate deconvolution may not be possible. If frequency components have been lost altogether, division by the contaminated frequency response will produce bands with infinite gain; these areas must be adjusted, or a longer pulse chosen.

Blind deconvolution is used when the unwanted convolution is not known; it is generally performed by estimating the convolution.

14.2 Optimal filters

Optimal filtering is used to separate a target signal from noise. In the frequency domain, the noise spectrum will overlap that of the target, making low-pass and high-pass filters less effective.

A *moving average filter* provides the fastest step response for a given amount of noise reduction.

A *matched filter* returns the correlation of the target pulse with the input at each point; this optimizes the difference between the peaks in the target signal and their background. It also changes the shape of the target pulse, but the shape must have been known already to perform the correlation.

If $S[f]$ is the frequency response of the target, and $N[f]$ that of the noise, a **Wiener filter** has frequency response:

$$H[f] = \frac{S[f]^2}{S[f]^2 + N[f]^2}$$

Creating a custom filter from this response optimizes the ratio of the target signal power to the noise power, over the length of the signal.

15 FFT convolution

Real-time or other segmented input can be convolved with the **overlap-add method**, which accepts a segment, pads it with zeros to make room for the convolution, convolves, and then adds the processed segment to the output. Each segment has the same position in the output that it held in the input, such that the beginning overlaps the tail added to the previous segment by the convolution.

FFT convolution uses this technique to process long signals in less time than ordinary convolution. The desired convolution is first translated to the frequency domain with the FFT; the input is then segmented, and each segment is also processed with the FFT. After multiplying the frequency responses, the inverse FFT is used to produce an output segment, which is made to overlap the segment before. For simplicity, the spectra are represented and multiplied as real and imaginary parts.

It may be impractical to multiply frequency responses unless their sample counts are identical, since interpolation would otherwise be needed. To produce identical sample counts, and to avoid circular convolution, the convolution

kernel and the input segment must both be padded with zeros before the FFTs until their lengths equal the same number, a power of two greater than or equal to the length of the segment plus that of the kernel, less one.

The time to process an ordinary convolution varies linearly with the length of the kernel; the time for FFT convolution varies logarithmically. The implementations are said to be equivalent when the kernel contains forty to eighty samples; above that, the FFT is faster.

16 Recursive filters

A **recursive filter** is implemented with a **recursion equation**, which incorporates one or more past outputs in each current output value. Recursion creates a long impulse response without requiring a lengthy convolution:

$$\begin{aligned} y[n] =& a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + \cdots \\ & + b_1 y[n-1] + b_2 y[n-2] + \cdots \end{aligned}$$

As always, passing the delta function gives the impulse response of the filter, which in this case is typically an exponentially-decaying oscillation. Because the response never settles at zero, this is called an **infinite impulse response (IIR) filter**. Among other methods, the coefficients can be calculated with the z-transform, which translates them to or from the frequency domain.

16.1 Single-Pole recursive filters

Single-pole recursive filters are equivalent to simple RC networks; they are appropriate for DC removal, smoothing, or other basic filtering operations. Given $0 < \lambda < 1$, the **single-pole low-pass filter** has coefficients:

$$\begin{aligned} a_0 &= 1 - \lambda \\ b_1 &= \lambda \end{aligned}$$

This produces the same effect as a first-order low-pass RC circuit. In the output:

$$\begin{aligned} y[n] &= (1-\lambda)x[n] + \lambda y[n-1] \\ &= (1-\lambda)\sum_{i=0}^{\infty} \lambda^i x[n-i] \end{aligned}$$

λ represents the decay constant. At any point, $y[n]$ approaches a fixed value as $x[n]$ is held constant:

$$L = \lim_{n \to \infty} y[n]$$

If a is the new value of $x[n]$, L is produced by the sum of a geometric series:

$$\begin{aligned} L &= (1-\lambda)(a + \lambda a + \lambda^2 a + \cdots) \\ \frac{L}{1-\lambda} &= a + \lambda a + \lambda^2 a + \cdots \end{aligned}$$

Multiplying both sides by λ and subtracting the new expression truncates the series:

$$\frac{L}{1-\lambda} - \lambda \frac{L}{1-\lambda} = a$$

This yields:

$$\begin{aligned} (1-\lambda)\frac{L}{1-\lambda} &= a \\ L &= a \end{aligned}$$

The **single-pole high-pass filter** is also equivalent to a first-order RC circuit:

$$\begin{aligned} a_0 &= \frac{1}{2}(1+\lambda) \\ a_1 &= -\frac{1}{2}(1+\lambda) \\ b_1 &= \lambda \end{aligned}$$

Just as RC is the time for the corresponding RC circuit to decay to $1/e$ of the starting voltage, d gives number of samples for output to decay to $1/e$ of the original amplitude, after the input drops from a steady state to zero:

$$d = -\frac{1}{\ln \lambda}$$

This gives:

$$\lambda = \mathrm{e}^{-1/d}$$

Alternatively:

$$\lambda = \mathrm{e}^{-2\pi f_C}$$

These relationships allow the coefficients to be set to achieve a given rise time or a given cutoff. By equating them, it is seen that the rise time increases as the cutoff decreases:

$$d = \frac{1}{2\pi f_C}$$

Single-pole filters are very fast, but in general, they have high roll-offs and very poor stopband attenuation. These can be improved somewhat by passing the signal through

the filter more than once, which effect can also be implemented with coefficients drawn from the z-transform. For a four-stage low-pass filter:

$$a_0 = (1-\lambda)^4$$
$$b_1 = 4\lambda$$
$$b_2 = -6\lambda^2$$
$$b_3 = 4\lambda^3$$
$$b_4 = -\lambda^4$$

This is comparable to a Blackman or Gaussian filter, but much faster.

16.2 Band-pass and band-stop filters

Given center frequency f, bandwidth BW at -3dB amplitude, and:

$$R = 1 - 3BW$$
$$K = \frac{1 - 2R\cos(2\pi f) + R^2}{2 - 2\cos(2\pi f)}$$

it is possible to implement a band-pass filter with:

$$a_0 = 1 - K$$
$$a_1 = 2(K - R)\cos(2\pi f)$$
$$a_2 = R^2 - K$$
$$b_1 = 2R\cos(2\pi f)$$
$$b_2 = -R^2$$

For a band-stop or **notch filter**:

$$a_0 = K$$
$$a_1 = -2K\cos(2\pi f)$$
$$a_2 = K$$
$$b_1 = 2R\cos(2\pi f)$$
$$b_2 = -R^2$$

These filters have somewhat rounded corners in the frequency domain; this can be partially amended by processing the signal more than once. The step responses exhibit moderate ringing.

16.3 Phase response

An impulse response that is symmetric about any sample has linear phase; convolving a symmetrical pulse with such a response will produce another symmetrical pulse. An impulse response that is not symmetrical will have non-linear phase, and convolution with a symmetrical pulse will necessarily produce an asymmetrical pulse. While it is trivial to produce FIR filters with symmetry, the impulse response of a recursive filter is inherently asymmetrical, since it approaches each steady state exponentially. For this reason, analog filters cannot maintain the symmetry of input pulses.

Bidirectional filtering is used to produce recursive filters with zero phase. After a signal is filtered in the forward direction, it is filtered again in reverse, as though the signal had been reversed, processed, and reversed again. To implement this reverse filtering:

$$\begin{aligned} y[n] =& a_0 x[n] + a_1 x[n+1] + a_2 x[n+2] + \cdots \\ &+ b_1 y[n+1] + b_2 y[n+2] + \cdots \end{aligned}$$

Conceptually, the two filter convolutions cause the forward and reversed filters to be multiplied in the frequency domain, which entails multiplying their magnitudes and adding their phase values. Since reversing an impulse response causes its phase spectrum to be negated, this cancels all phase values, producing a zero-phase process.

If the input is segmented, the segments can be processed and combined with the overlap-add method. Since the impulse responses are technically infinite, the segments must be truncated on both sides when output values reach sufficiently low levels.

17 Chebyshev filters

The **Chebyshev filter** is a recursive filter that improves roll-off by allowing ripple in the frequency domain. A **Type I** Chebyshev filter allows no ripple outside the passband, while a **Type II** filter allows no ripple outside the stopband. **Elliptic filters** allow ripple in either band. Allowing no ripple at all produces a **maximally flat** or **Butterworth filter**.

The coefficients of a Chebyshev filter are determined with the z-transform. In Type I filters, increasing the roll-off increases the amount of passband ripple, but good roll-off can be achieved with as little as 0.5% ripple, this being comparable to the precision of analog electronics. Setting the cutoff near the middle of the frequency range produces lower roll-offs.

In the z-transform, a filter's characteristics are defined by a rational expression of two complex polynomials. The roots

of the numerator in this expression are known as **zeros**, while the roots of the denominator are **poles**. The pole count is always an even number. The filter's roll-off increases with the number of poles, as does the amount of overshoot in the step response, this typically ranging from 5% to 30% in Type I. Step response overshoot is also affected somewhat by the cutoff frequency.

As the pole count increases, the magnitude of the input coefficients $a_0, a_1, a_2, \cdots$ can decrease far below that of the output coefficients, $b_1, b_2, b_3, \cdots$. Eventually the input values can be lost in the noise produced by rounding error in the other calculations; when this happens, filter performance degrades and the filter ultimately becomes unstable. A filter with more than two poles is mathematically equivalent to a series of two-pole filters, and explicitly processing input with such a series produces the same output with less rounding error.

18 Comparing filters

18.1 Digital and analog filters

The filtering characteristics of analog systems are limited by the accuracy of their components; even a maximally flat analog filter may produce passband ripple near 1%. Digital filters can produce much flatter passbands, steeper roll-offs, and better stopband attenuation; moreover, with additional computing resources, their performance can be improved to almost arbitrarily high levels. Digital filters can produce symmetrical step responses and linear phase output.

Analog systems generally have much higher dynamic ranges, with a typical op amp producing noise of 2μV and having a saturation level of 20V, for a range of ten million. They can also operate at high frequencies that would necessitate very high bit rates if sampling were used.

18.2 Windowed-Sinc and Chebyshev filters

Windowed-sinc filters and Chebyshev filters provide good performance in the frequency domain, but the windowed-sinc uses convolution, while the Chebyshev uses recursion.

Type I Chebyshev filters allow ripple in the passband, though this can be eliminated by decreasing the roll-off. Windowed-sinc filters provide similar roll-offs without passband ripple, and when very precise frequency separation is needed, it can be obtained by lengthening the impulse response. Recursive filters, by contrast, are eventually constrained by rounding error. For a given roll-off, windowed-sinc filters provide somewhat better stopband attenuation. Recursive filters use much less CPU time, even when the windowed-sinc is implemented with FFT convolution.

Both filters produce significant overshoot in the step response, but the Chebyshev also has non-linear phase, though this can be corrected with bidirectional filtering.

18.3 Moving average and single-pole filters

Moving average and single-pole filters operate well in the time domain and much less well in the frequency domain. The moving average filter produces a fast linear step response. The single-pole filter produces an asymmetrical step with non-linear phase, unless bidirectional filtering is used, in which case two smooth corners are created. Both filters require minimal CPU time.

19 Audio processing

Sound power level (SPL) is a measure of loudness relative to the weakest level discernible to human ears, at 0dB SPL. The loudest distinct level is approximately 120dB SPL, and damage can occur above 85dB SPL. Speech occurs at roughly 60dB SPL.

After decoding, CD audio is represented with 16-bit samples at a 44.1KHz sampling rate. If this were immediately converted to an analog signal, the antialiasing filter would have to block frequencies above 22.05KHz while passing those below 20KHz, which would be difficult with analog components. This filtering cannot be performed digitally, because the aliasing is inherent to the sampling process; it is the negative frequencies above 22.05KHz, and the other iterations above that which must be removed. Instead, most systems interpolate to a 176.4KHz sampling rate by inserting three zeros between each of the original samples; because of the increased quality this offers, the bit depth can also be reduced to 15. Next, frequencies between 22.05KHz and 88.2KHz are removed with a digital filter. It is still necessary to use an analog antialiasing filter, but now the filter can have a wider transition band that ranges from 22.05KHz to 88.2KHz. The distortion produced by

zero-order hold in the DAC can be corrected in either filter.

Although conventional quantization produces a linear relationship between sample values and output amplitudes, the human perception of loudness varies logarithmically relative to amplitude. **Companding** exploits this fact by using a non-linear quantization scale; this allows the subjective quality of a 12-bit telephone signal to be produced with only 8 bits. Companding can be implemented either by passing the analog signal through a waveshaper before the ADC, by sampling with a specialized non-linear ADC, or by sampling at the higher bit depth and then converting with a lookup table.

Linear predictive coding (LPC) produces a simplified representation of human speech. The speech is sampled at around 40 points per second, and at each point, parameters are stored representing a sound source, the pitch of the source, if it is pitched, and the filter coefficients of a vocal formant. The sound source can be noise or a harmonically-rich waveform. The data is used for speech synthesis or speech recognition.

19.1 Non-linear processes

Sometimes non-linear processes are needed to produce desired results. When a signal is contaminated by wideband noise, the noise can be reduced by segmenting the signal, translating each segment to the frequency domain, and then modifying the resultant frequency response such that high-magnitude components are retained, low-magnitude components are discarded, and intermediate components are attenuated smoothly between those extremes. The modified frequency response is used to produce a custom filter, which is then applied to the input segment. To avoid abrupt changes in frequency content, segments are typically made to overlap, and are windowed after filtering and then recombined. Unlike the Wiener filter, the correction varies from moment to moment, and there is no need to know the signal and noise spectra in advance.

Homomorphic signal processing can be used to process signals that result from non-linear operations. If two signals have been multiplied, the logarithm of their product $\ln xh = \ln x + \ln h$. Since $\ln h$ is another periodic signal, it can perhaps be removed with a conventional filter, though the logarithm will add harmonics that also must be filtered. Because negative values are found in the input, the complex logarithm must be used. Also, because the logarithm causes aliasing, the signal is often oversampled before being processed. After, the process is reversed by calculating the exponent.

When signals are convolved, such that $y = x * h$, the DFT produces $Y = X \times H$. Calculating the logarithm of the spectrum $\ln Y = \ln X + \ln H$ can allow $\ln H$ to be removed by filtering the spectrum itself. The corrected spectrum is produced by calculating the exponent, and the signal by performing an inverse DFT.

20 Complex numbers

Representing some number $a+bj$ as a point on the complex plane allows that number to be expressed in polar terms, with a vector that stretches from the origin to the point. The **magnitude** of this vector:

$$M = \sqrt{a^2 + b^2}$$

while the **phase angle** or **argument**:

$$\theta = \arctan(b/a)$$

As expected:

$$a = M \cos\theta$$
$$b = M \sin\theta$$

This allows the point's rectangular representation to be expressed in terms of its polar coordinates:

$$a + bj = M \cos\theta + (M \sin\theta)j$$

20.1 Euler's formula

Euler's formula gives:

$$e^{j\theta} = \cos\theta + j \sin\theta$$

Therefore, the point's polar representation can also be expressed as a **complex exponential** that gives the signal's **complex amplitude**:

$$a + bj = Me^{j\theta}$$

Where complex numbers in the rectangular form are easily added and subtracted, numbers in this form are easily multiplied and divided:

$$M_1 e^{j\theta_1} \times M_2 e^{j\theta_2} = M_1 M_2 e^{j(\theta_1+\theta_2)}$$

Multiplying a complex number by j causes the coordinates to be switched, with the real coordinate negated relative

to the original imaginary coordinate. This produces a 90° counter-clockwise rotation within the complex plane, around the origin. This is to be expected, since it represents multiplication by a unit vector with a 90° phase angle.

Note that:

$$\frac{1}{j} = \frac{j}{j^2} = -j$$

Also, because $\cos(\pi/2) = 0$ and $\sin(\pi/2) = 1$:

$$j = \cos\left(\frac{\pi}{2}\right) + j\sin\left(\frac{\pi}{2}\right) = \mathrm{e}^{j\pi/2}$$

Because $\sin(-\alpha) = -\sin(\alpha)$:

$$-j = \cos\left(-\frac{\pi}{2}\right) + j\sin\left(-\frac{\pi}{2}\right) = \mathrm{e}^{-j\pi/2}$$

21 Phasor transform

In the time domain, $M\cos(\omega t+\phi)$ produces a sinusoid with frequency ω. If M and *starting* phase ϕ are interpreted as polar coordinates, the point they reference:

$$\begin{aligned} M\cos\phi + (M\sin\phi)j &= a + bj \\ &= M\mathrm{e}^{j\phi} \end{aligned}$$

is called a **phasor**, and it identifies a specific signal within the complex plane of sinusoids with frequency ω. Note that the exponent is $j\phi$, not $j(\omega t+\phi)$, as might be obtained by applying Euler's relation in the time domain; the point *represents* the signal, it does not reproduce it, and in fact it cannot, since it is not a function of t.

Returning to the time domain, since:

$$\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

it follows that:

$$\begin{aligned} M\cos(\omega t+\phi) &= M(\cos\omega t\cos\phi - \sin\omega t\sin\phi) \\ &= A\cos\omega t + B\sin\omega t \end{aligned}$$

This is the same sinusoid, so it is represented by the same point. Where the polar representation gives the magnitude and phase of the sinusoid, this rectangular representation defines it as a linear combination of zero-phase cosine and sine waves with the same frequency ω. The phase of the sinusoid is determined by the sign and relative weights of the two rectangular components. Since:

$$\begin{aligned} A &= M\cos\phi = a \\ B &= -M\sin\phi = -b \end{aligned}$$

the coefficients in the time domain relate directly to the coordinates in the complex plane. Notice that the sine component weight $B = -b$; for this reason, a time-domain sine wave is represented in the phasor domain with $-j$.

Any sinusoid can be decomposed this way, and when two of the same frequency are summed, their coefficients are summed as well:

$$\begin{aligned} &M_1\cos(\omega t+\phi_1) + M_2\cos(\omega t+\phi_2) \\ &\quad = (A_1+A_2)\cos\omega t + (B_1+B_2)\sin\omega t \end{aligned}$$

Therefore, the sum of two sinusoids with frequency ω is given in the complex plane by the sum of their vector representations. This **phasor transform** simplifies many operations by replacing time-domain representations like $M\cos(\omega t+\phi)$ with phasor-domain representations like $a+bj$ and $Me^{j\phi}$. All sinusoids must have the same frequency, and the operations must be linear.

Alternatively, since $\cos\theta = \mathrm{Re}(e^{j\theta})$:

$$\begin{aligned} M\cos(\omega t+\phi) &= \mathrm{Re}(M\mathrm{e}^{j(\omega t+\phi)}) \\ &= \mathrm{Re}(M\mathrm{e}^{j\phi}\cdot\mathrm{e}^{j\omega t}) \end{aligned}$$

Since $\mathrm{e}^{j\omega t}$ is fixed for all sinusoids in the plane, phasor $M\mathrm{e}^{j\phi}$ again uniquely identifies $M\cos(\omega t+\phi)$.

22 Circuit analysis

Because a linear system exhibits sinusoidal fidelity, so that only the amplitude and phase of sinusoidal inputs are allowed to change, its effect on input components of a particular frequency can be represented by a single polar-form phasor that is multiplied by the input phasor to produce a change in magnitude and a shift in the phase. A set of such phasors can describe the amplitude and phase response of the system in general. If the input and output signals are known, these phasors can be determined by dividing the output by the input at each frequency.

22.1 Inductance and capacitance

An **inductor** is typically constructed by winding a conductor into a coil, often around a magnetic core. When current flows through the coil, a magnetic field is created; if the current changes, this field changes with it, inducing

a voltage that *opposes* the change in current. The component's **inductance** L relates this voltage to the current change:

$$v_L(t) = L\frac{\mathrm{d}i_L(t)}{\mathrm{d}t}$$

Inductors pass direct current while *opposing* alternating current.

A **capacitor** is constructed from two conductive plates separated by a thin insulator called a **dielectric**. When there is a difference in potential across the plates, a negative charge accumulates on one of them, and an equivalent positive charge on the other. If the voltage remains constant, the accumulated charge eventually comes to offset the potential difference, and the current flow stops; if the voltage changes in either direction, the current resumes. The component's **capacitance** C relates the current flow to the change in voltage:

$$i_C(t) = C\frac{\mathrm{d}v_C(t)}{\mathrm{d}t}$$

Capacitors pass alternating current while *blocking* direct current.

The most general model of a circuit is produced by combining these expressions for each component and solving the resultant differential equation; if the input is assumed to be sinusoidal, however, a much simpler solution can found using the phasor transform. If the current flowing through an inductor:

$$i_L(t) = \sin(\omega t)$$

then the induced voltage:

$$v_L(t) = \omega L\cos(\omega t)$$

Expressing these as phasors gives:

$$I_L = -j$$
$$V_L = \omega L$$

22.2 Impedence

The ratio between the complex voltage amplitude and the complex current is known as the **impedence**:

$$Z = \frac{V}{I}$$

When the values are expressed as complex exponentials, this shows how the magnitude and phase of the voltage relate to those of the sinusoidal current, just as Ohm's law relates voltage to a direct current. Therefore:

$$Z_L = \frac{V_L}{I_L} = j\omega L$$

As expected, the magnitude of the impedence produced by an inductor increases with the input frequency. Therefore, as frequency increases, a higher voltage is needed to maintain a given flow of current.

Similarly, if:

$$v_C = \sin(\omega t)$$

then:

$$i_C = \omega C\cos(\omega t)$$

and:

$$V_C = -j$$
$$I_C = \omega C$$
$$Z_C = \frac{V_C}{I_C} = -\frac{j}{\omega C}$$

The magnitude of the impendence produced by a capacitor decreases as the input frequency increases. Therefore, as frequency increases, a lower voltage is needed to maintain a given level of current.

Together, inductance and capacitance are known as **reactance**, this being the component's opposition to changes in current or voltage. Reactance is the imaginary part of the impedence:

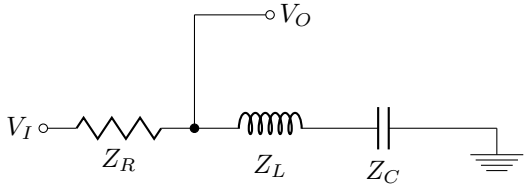
$$X = \omega L - \frac{1}{\omega C}$$

As shown, capacitance *decreases* total reactance. Given resistance R:

$$Z = R + jX$$

In fact, resistance can be understood as impedence with a zero phase shift.

This circuit implements a **notch filter**, which is a band-stop filter with a narrow stop band:



If the components were all resistors, V_O would be related to V_I by a voltage divider formula containing the associated resistances. As it happens, this formula also works when impedences are used, though a complex ratio is produced:

$$\frac{V_O}{V_I} = \frac{Z_L + Z_C}{Z_R + Z_L + Z_C}$$

Substituting for Z_R, Z_L, and Z_C, and separating the result into real and imaginary parts gives the frequency response of the filter, in rectangular coordinates:

$$H(\omega) = \frac{V_O}{V_I} = \frac{k^2}{R^2 + k^2} + j\frac{Rk}{R^2 + k^2}, \quad \text{for } k = \omega L - \frac{1}{\omega C}$$

Converting this to polar coordinates gives the amplitude and phase response:

$$\operatorname{Mag} H(\omega) = \frac{k}{\sqrt{R^2 + k^2}} \quad \operatorname{Ph} H(\omega) = \arctan\left(\frac{R}{k}\right)$$

23 Complex DFT

In the **complex DFT analysis equation**, both $x[n]$ and $X[k]$ represent complex numbers:

$$X[k] = \frac{1}{N}\sum_{n=0}^{N-1} x[n]\, \mathrm{e}^{-j2\pi k\cdot n/N} = \frac{1}{N}\sum_{n=0}^{N-1} x[n]\Big[\cos\left(2\pi k\frac{n}{N}\right) - j\sin\left(2\pi k\frac{n}{N}\right)\Big]$$

Correlation with the cosine basis function produces the real part of $X[k]$, while correlation with the sine basis produces the imaginary part.

In the real DFT, k runs from zero to $N/2$, so only positive frequencies are evaluated; in the *complex* DFT, both positive *and* negative frequencies are evaluated. Because $\cos(-\alpha) = \cos\alpha$ and $\sin(-\alpha) = -\sin\alpha$, DFT value $X[-k]$ is seen to be the complex conjugate of $X[k]$. In the time domain, because $\mathrm{e}^{j\theta} = \cos\theta + j\sin\theta$, it can be shown that:

$$\cos\theta = \frac{\mathrm{e}^{j\theta} + \mathrm{e}^{-j\theta}}{2} \qquad \sin\theta = \frac{\mathrm{e}^{j\theta} - \mathrm{e}^{-j\theta}}{2j}$$

This allows:

$$\cos\omega t = \frac{1}{2}\,\mathrm{e}^{j(-\omega t)} + \frac{1}{2}\,\mathrm{e}^{j(\omega t)}$$

$$\sin\omega t = \frac{1}{2}j\mathrm{e}^{j(-\omega t)} - \frac{1}{2}j\mathrm{e}^{j(\omega t)}$$

which expresses either sinusoid as the sum of one complex exponential at the negative frequency, and one at the positive. As a result, the basis functions of frequency $-k$ and k sum to produce a set of positive-frequency functions, as expected.

Ordinarily, the real values of $x[n]$ contain the time domain signal, while the imaginary values are set to zero. When this is done, the spectrum produced by $\operatorname{Re} x[n]$ displays even symmetry in the real part of $X[k]$, and odd symmetry in the imaginary part. When $\operatorname{Im} x[n]$ contains time domain data, its spectrum displays odd symmetry in the real part of $X[k]$, and even symmetry in the imaginary part.

Spectral values must be normalized before being processed with the inverse DFT. After the real DFT, values $0 < k < N/2$ are scaled by a factor of two, but this is not necessary for the complex DFT, as each of these is associated with two sinusoids, one in the positive frequency range, and one in the negative. By contrast, the $k = 0$ and $k = N/2$ values represent a single frequency each; therefore, there is no need to scale these differently, and all values are normalized with division by N.

The **complex DFT synthesis equation**:

$$x[n] = \sum_{k=0}^{N-1} X[k]\, \mathrm{e}^{j2\pi k\cdot n/N}$$

Because $X[k] = \operatorname{Re} X[k] + j \operatorname{Im} X[k]$, this produces:

$$x[n] = \sum_{k=0}^{N-1} \operatorname{Re} X[k]\Big[\cos\left(2\pi k\frac{n}{N}\right) + j\sin\left(2\pi k\frac{n}{N}\right)\Big] + \sum_{k=0}^{N-1} \operatorname{Im} X[k]\Big[j\cos\left(2\pi k\frac{n}{N}\right)\Big] - \sin\left(2\pi k\frac{n}{N}\right)$$

As demonstrated, each value in the frequency domain produces both a real and an imaginary sinusoid in the time domain. The values between zero and $N/2$ represent positive frequencies, and each is matched by another value between $N/2$ and N with an effectively negative frequency. The real values of $X[k]$ produce sinusoids $\cos\alpha + j\sin\alpha$, and since $\sin(-\alpha) = -\sin\alpha$, the positive and negative frequencies cancel the imaginary part to produce a single cosine component. The imaginary values of $X[k]$ produce sinusoids $j\cos\alpha - \sin\alpha$. Because the imaginary spectrum has odd symmetry, the values associated with negative frequencies are themselves negated, and the two combine to produce a single sine component. The cosine and sine components together define a single sinusoid with the necessary amplitude and phase.

23.1 Other complex transforms

If T is the period of some periodic input, the **complex Fourier Series analysis equation** gives:

$$X[k] = \frac{1}{T}\int_0^T x(t)\,\mathrm{e}^{-j2\pi k\cdot t/T}\,\mathrm{d}t$$

In the **complex Fourier Series synthesis equation**:

$$x(t) = \sum_{k=-\infty}^{\infty} X[k]\,\mathrm{e}^{j2\pi k\cdot t/T}$$

positive k represent positive frequencies, and negative k negative frequencies.

The **complex DTFT analysis equation**:

$$X(\omega) = \frac{1}{2\pi}\sum_{n=-\infty}^{\infty} x[n]\,\mathrm{e}^{-j\omega n}$$

In the **complex DTFT synthesis equation**:

$$x[n] = \int_0^{2\pi} X(\omega)\,\mathrm{e}^{j\omega n}\,\mathrm{d}\omega$$

ω values between zero and π represent positive frequencies, while those between π and 2π represent negative.

The **complex Fourier transform analysis equation**:

$$X(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} x(t)\,\mathrm{e}^{-j\omega t}\,\mathrm{d}t$$

In the **complex Fourier transform synthesis equation**:

$$x(t) = \int_{-\infty}^{\infty} X(\omega)\,\mathrm{e}^{j\omega t}\,\mathrm{d}\omega$$

positive ω represent positive frequencies, and negative ω negative frequencies.

24 Laplace transform

The **Laplace transform** associates continuous time-domain signals with signals in the **Laplace domain**, a complex plane with frequency ω on the imaginary axis and exponential weight σ on the real. This allows any point in the Laplace domain to be identified with:

$$s = \sigma + j\omega$$

Each such point has a value that is also a complex number. Given time-domain signal $f(t)$, the value at s:

$$F(s) = \int_{-\infty}^{\infty} f(t)\,\mathrm{e}^{-st}\,\mathrm{d}t$$

Because $\mathrm{e}^{-st} = \mathrm{e}^{-\sigma t}\cdot\mathrm{e}^{-j\omega t}$ the values for any fixed σ are seen to equal the Fourier transform of $f(t)\,\mathrm{e}^{-\sigma t}$. Therefore, for negative σ, the time-domain signal is weighted by an *increasing* exponential function that equals one where t is zero, and has a steeper slope as σ becomes more negative. For positive σ, the signal is weighted by a *decreasing* exponential function. Where σ is zero, the Laplace domain values equal the complex Fourier transform of $f(t)$.

Despite its similarity to the Fourier transform, the Laplace transform is primarily used to solve differential equations. Time domain functions are associated with functions in the Laplace domain by solving Laplace integrals to produce **Laplace pairs**. Given function $f(t)$ that is zero for $t \geq 0^-$:

$$f(t) \quad \Leftrightarrow \quad F(s) = \int_{0^-}^{\infty} f(t)\,\mathrm{e}^{-st}\,\mathrm{d}t$$

From this, it can be shown that:

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} \quad \Leftrightarrow \quad sF(s) - f(0)$$

These and other Laplace pairs allow systems of differential equations to be represented as expressions of $F(s)$. These expressions can be solved algebraically, and Laplace pairs or the **inverse Laplace transform** can then be used to return the solutions to the time domain.

24.1 Transfer functions

Like the phasor transform, the Laplace transform can be used to analyze circuits. Given input signal $x(t)$ and a linear system with impulse response $h(t)$, the output:

$$y(t) = x(t) * h(t)$$

As with the Fourier transform, moving the functions to the Laplace domain produces:

$$Y(s) = X(s)\cdot H(s)$$

with $H(s)$ being known as the system's **transfer function**. So, if the current flowing through an inductor:

$$i_L(t) = \sin(\omega t)$$

then, because $v_L(t) = L\frac{\mathrm{d}i_L(t)}{\mathrm{d}t}$, the induced voltage:

$$v_L(t) = \omega L \cos(\omega t)$$

If the signal is assumed to start at $t = 0$, the Laplace pairs for sine and cosine give:

$$I_L(s) = \frac{\omega}{\omega^2 + s^2} \qquad V_L(s) = \frac{sL\omega}{\omega^2 + s^2}$$

Alternatively, applying the Laplace pair for differentiation:

$$v_L(t) = L\frac{\mathrm{d}i_L(t)}{\mathrm{d}t} \;\Leftrightarrow\; V_L(s) = sLI_L(s)$$

Therefore, the inductor is represented in the Laplace domain by transfer function:

$$\frac{V_L}{I_R} = sL$$

Similarly, a capacitor is represented by $1/sC$, and a resistor by R. The phasor transform is in fact seen to be a subset of the Laplace transform, since $\sigma = 0$ produces $s = j\omega$.

Just as impedences can be combined in a voltage divider to characterize an entire circuit, so can Laplace representations. The transfer function for a simple notch filter:

$$H(s) = \frac{V_O}{V_I} = \frac{sL + 1/sC}{R + sL + 1/sC}$$

This extends the frequency response $H(\omega)$ produced by the phasor transform into the Laplace domain.

If the system is defined by a set of differential equations (as any RLC circuit will be) the transfer function can be expressed as a ratio of polynomials of s:

$$H(s) = \frac{Ls^2 + 1/C}{Ls^2 + Ls + 1/C}$$

Factoring these polynomials produces an equation of the form:

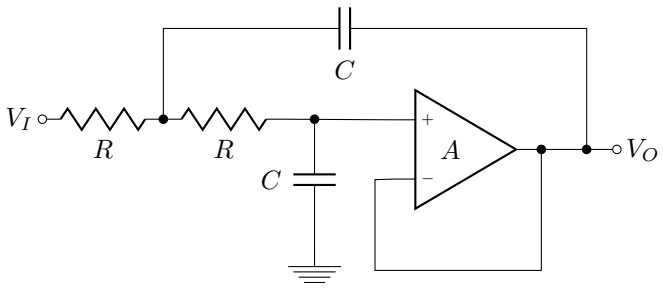
$$H(s) = \frac{(s - z_1)(s - z_2)(s - z_3)\cdots}{(s - p_1)(s - p_2)(s - p_3)\cdots}$$

The complex roots z_n give the **zeros** of the filter, while the roots p_n give the **poles**. The transfer function for an RLC circuit will contain one pole for each inductor or capacitor, and a number of zeros equal to or less than the number of poles. Inductors and capacitors create poles because they store energy.

24.2 Filter design

Plotting the Laplace domain magnitude in three dimensions shows the poles to be points where the function rises to infinity, while the zeros are points where it drops to zero. The placement of these points is often represented with a **pole-zero diagram** that gives a top-down view of the domain, with poles marked as X's and zeros as O's. Immediately it is seen that the system's frequency response $H(\omega)$ is determined by the placement of these structures along the frequency axis, along with their proximity thereto, since $H(\omega)$ is the two-dimensional cross-section of the function at $\sigma = 0$. The compromises inherent in filter design are evident as well. The greatest stopband attenuation is created by placing zeros on or near the frequency axis. Filter roll-off is maximized by placing a pole very near some zero, but if the zeros are near the frequency axis, this places the pole near that axis as well, creating a sharp peak in the passband.

Because second-order polynomials can be factored with the quadratic equation, complex filters are often constructed by combining multiple **biquad filters**, these being recursive filters with two poles and two or fewer zeros. One common biquad design is the **Sallen-Key** circuit:



After factoring this system's transfer function, poles are found at:

$$\sigma = \frac{A - 3}{2RC} \qquad \omega = \frac{\pm\sqrt{-A^2 + 6A - 5}}{2RC}$$

Because:

$$\sigma^2 + \omega^2 = \left(\frac{1}{RC}\right)^2$$

it is seen that both poles lie on a circle around the origin with radius $1/RC$. The circle intersects the vertical ω-axis at the cutoff frequency, so that:

$$\omega_C = \pm\frac{1}{RC}$$

When A is one, the poles meet on the σ-axis, at the left edge of the circle, where ω is zero; this creates a low-pass

filter with a slow roll-off. As A increases, the poles separate, approaching the ω-axis from either side of the circle; the roll-off increases, and eventually a peak forms at ω_C. When A reaches three, the poles coincide with the ω-axis, producing an infinite peak at the cutoff. Beyond this point the filter becomes unstable, as does any filter with poles in the right half of the Laplace domain.

When multiple Sallen-Key circuits are connected so that their poles distribute evenly around the left half of the circle, a Butterworth filter is formed, producing the sharpest possible roll-off without allowing ripple in the passband. Because all poles fall on the same circle, all the circuits use the same values of R and C.

If this even pole distribution is elongated along the ω-axis to create an ellipse, a Chebyshev filter is formed, increasing the rolloff but producing ripple in the passband. This requires different values of R and C in the component circuits.

If zeros are placed on the ω-axis just past the cutoff frequency, an elliptic filter is formed. This creates the sharpest possible roll-off, but it produces ripple in both the stopband and the passband. Filters of this type are designed with elliptic rational functions.

A low-pass filter is converted to a high-pass filter by replacing all instances of s in the transfer function with $1/s$. In a Sallen-Key circuit, this is produced by exchanging the resistors with the capacitors, which moves the poles and places two zeros at the origin.

25 Z-transform

The **z-transform** is a refinement of the Laplace transform that applies to discrete signals. The Laplace transform of continuous signal $x(t)$:

$$X(s) = \int_{-\infty}^{\infty} x(t)\, \mathrm{e}^{-st}\, \mathrm{d}t$$

with $s = \sigma + j\omega$. Therefore:

$$\mathrm{e}^{-st} = \mathrm{e}^{-\sigma t} \cdot \mathrm{e}^{-j\omega t}$$

If the same exponential weight is represented by:

$$r = \mathrm{e}^{\sigma}$$

and if:

$$z = r\, \mathrm{e}^{j\omega}$$

then a similar construction can serve in the z-domain, after replacing time variable t with sample number n:

$$\mathrm{e}^{-\sigma n} \cdot \mathrm{e}^{-j\omega n} = (\mathrm{e}^{\sigma} \cdot \mathrm{e}^{j\omega})^{-n} = (r\, \mathrm{e}^{j\omega})^{-n} = z^{-n}$$

This gives the *z-transform* of discrete signal $x[n]$:

$$X(z) = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$$

Note that where s is interpreted in the Laplace domain as a set of complex *rectangular* coordinates, z is interpreted as a set of complex *polar* coordinates, giving the z-domain very different properties. The magnitude coordinate r is the base of the exponential curve. The argument ω, when divided by 2π, is the frequency as a fraction of the sample rate.

When e^{σ} is one, the Laplace transform is equivalent to the complex Fourier transform; the frequency response therefore follows the vertical ω-axis, and extends indefinitely in both directions, since the frequency in a continuous signal could have any value. The z-transform is equivalent to the DTFT when r is one; the frequency response follows the unit circle, and repeats as ω grows in either direction. When the input consists entirely of real values, the top and bottom halves of both domains are symmetrical; in each case, this produces a frequency response that is symmetrical for positive and negative frequencies.

Placing poles in the right half of the Laplace domain produces an unstable filter. In the z-domain, this happens when poles are placed *outside* the unit circle.

25.1 Analyzing recursive systems

Just as continuous systems are described by differential equations, discrete recursive systems are described by *difference* equations. In particular, recursive filters are implemented with *recursion equations*:

$$\begin{aligned} y[n] =& a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + \cdots \\ &+ b_1 y[n-1] + b_2 y[n-2] + b_3 y[n-3] + \cdots \end{aligned}$$

Calculating the z-transform of both sides eventually allows the transfer function to be expressed in terms of the filter coefficients:

$$H[z] = \frac{Y[z]}{X[z]} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots}{1 - b_1 z^{-1} - b_2 z^{-2} - b_3 z^{-3} - \cdots}$$

Note that this equation is sometimes written so that the b terms are *added*, and the coefficients are negated to account

for this. If negated coefficients are used in the original recursion equation, the filter will unstable.

The **order** of a filter is the difference between the number of the current sample and the oldest sample used in its difference equation. Negative exponents are used in the general form of the transfer function because the order is not known. When the order *is* known, the equation is commonly expressed with positive exponents. For a third-order filter, this produces:

$$H[z] = \frac{a_0 z^3 + a_1 z^2 + a_2 z + a_3}{z^3 - b_1 z^2 - b_2 z - b_3}$$

Filters are placed in series by multiplying their transfer functions, or in parallel by adding them. New coefficients can then be calculated to implement the combination.

Factoring the transfer function allows the filter to be described as a collection of poles and zeros:

$$H[z] = \frac{(z - z_1)(z - z_2)(z - z_3)\cdots}{(z - p_1)(z - p_2)(z - p_3)\cdots}$$

Conversely, a filter can be *designed* by placing poles and zeros in the z-domain, expressing the transfer function in terms of $(z - z_n)$ and $(z - p_n)$, multiplying these expressions, and then collecting the z^n terms and converting to negative exponents to find the coefficients.

In some cases it is possible to relate the filter coefficients directly to pole and zero positions. Given a *biquad* filter with poles $(r_p, \pm\omega_p)$ and zeros $(r_z, \pm\omega_z)$:

$$\begin{aligned} a_0 &= 1 \\ a_1 &= -2r_z \cos\omega_z \\ a_2 &= r_z^2 \\ b_1 &= 2r_p \cos\omega_p \\ b_2 &= -r_p^2 \end{aligned}$$

Because it is found where the unit circle intersects the z-domain, the frequency response $H(\omega)$ can be expressed mathematically by setting r to one and then solving the transfer function.

To produce a graph of the frequency response, the transfer function is sometimes sampled along the unit circle; this method does not account for the rounding error that accumulates as values are cycled through recursive equations, however, and the resulting noise can make the filter unstable. As an alternative, the recursion equation can be used to generate an impulse response, and the Fourier transform can be applied to find its spectrum. A sufficiently large number of samples must be used; if a *larger* sample length produces a similar spectrum, it can be assumed that the original length is adequate.

25.2 Manipulating filters

If their passbands overlap, a low-pass filter can be placed *in series* with a high-pass filter to create a band-pass filter. If they do not overlap, they can be combined *in parallel* to create a band-stop filter. These combinations are implemented by *multiplying* or *adding* transfer functions, respectively.

Filters can be modified with *spectral inversion*, which inverts the frequency response by adding a copy of the source signal to a negated copy of the original filter output. When applied to a recursive filter, the b coefficients remain unchanged, while the new a coefficients:

$$\begin{aligned} a'_0 &= 1 - a_0 \\ a'_1 &= -a_1 - b_1 \\ a'_2 &= -a_2 - b_2 \\ a'_3 &= -a_3 - b_3 \\ &\vdots \end{aligned}$$

This typically produces poor results with recursive filters, however, because of the phase shift they produce.

A filter's gain can be adjusted by multiplying each of the a coefficients by a common gain factor. To produce a unity-gain low pass filter, the filter's natural gain must be determined at the zero frequency. This is done by setting all input values in the recursion equation to one, and all output values to the gain g, so that:

$$g = a_0 + a_1 + a_2 + \cdots + b_1 g + b_2 g + b_3 g + \cdots$$

This yields:

$$g = \frac{a_0 + a_1 + a_2 + \cdots}{1 - b_1 - b_2 - b_3 - \cdots}$$

To produce a unity-gain high pass filter, the gain at the Nyquist frequency must be determined. This is done by setting alternating input values to one or negative one, so that:

$$g = a_0 - a_1 + a_2 - \cdots - b_1 g + b_2 g - b_3 g + \cdots$$

This yields:

$$g = \frac{a_0 - a_1 + a_2 - \cdots}{1 + b_1 - b_2 + b_3 - \cdots}$$

25.3 Filter transforms

The **bilinear transform** is used to change continuous-time analog filters into discrete-time digital filters. The stability of the original filter is preserved, and each point on the vertical frequency response in the Laplace domain is mapped to a point on the circular z-domain response. Features in the original response are increasingly compressed as the frequency rises, since that response is infinite and the z-domain response is finite. This *frequency warping* shifts points to lower frequencies in the new response.

The transform is effected by replacing instances of s in the original transfer function so that:

$$s \to \frac{2(z-1)}{T(z+1)} \qquad T = 2\tan\left(\frac{1}{2}\right)$$

The poles of a Butterworth filter are equally spaced around a circle in the Laplace domain, and the intersection of this circle with the ω axis gives the cutoff frequency ω_C of the filter, in radians per second. Starting with a recursive low-pass filter with $\omega_C = 1$, a new filter with cutoff ω'_C can be created by applying a **low-pass to low-pass transform** to the transfer function:

$$z^{-1} \to \frac{z^{-1}-k}{1-kz^{-1}} \qquad k = \frac{\sin\left(\frac{1}{2}(1-\omega'_C)\right)}{\sin\left(\frac{1}{2}(1+\omega'_C)\right)}$$

Given a biquad filter with coefficients a_0, a_1, a_2, b_1, and b_2, this produces new coefficients:

$$a'_0 = \frac{a_0 - a_1 k + a_2 k^2}{D}$$
$$a'_1 = \frac{-2a_0 k + a_1(1+k^2) - 2a_2 k}{D}$$
$$a'_2 = \frac{a_0 k^2 - a_1 k + a_2}{D}$$
$$b'_1 = \frac{2k + b_1(1+k^2) - 2b_2 k}{D}$$
$$b'_2 = \frac{-k^2 - b_1 k + b_2}{D}$$

with:

$$D = 1 + b_1 k - b_2 k^2$$

Similarly, a **low-pass to high-pass transform** can be used to change the cutoff *and* create a high-pass filter:

$$z^{-1} \to \frac{-z^{-1}-k}{1+kz^{-1}} \qquad k = -\frac{\cos\left(\frac{1}{2}(\omega'_C+1)\right)}{\cos\left(\frac{1}{2}(\omega'_C-1)\right)}$$

When applied to a biquad filter, this produces the same coefficients as the low-pass to low-pass transform, except that k is defined differently and a'_1 and b'_1 are negated.

Sources

The Scientist and Engineer's Guide to DSP, First Edition
Steven W. Smith
California Technical Publishing

Wikipedia
`http://en.wikipedia.org`