Linear Filters, Sampling, & Fourier Analysis

Goal: Mathematical foundations for digital image analysis, representation and transformation.

Outline:

- **Sampling Continuous Signals**
- **Linear Filters and Convolution**
- **Fourier Analysis**
- **Sampling and Aliasing**

Suggested Readings: "Introduction to Fourier Analysis" by Fleet and Jepson (2005), Chapters 1 and 7 of Forsyth and Ponce.

Matlab Tutorials: linSysTutorial.m, samplingTutorial.m, upsample.m and imageTutorial.m.

Sampling

Approximate continuous signals with a discrete sequence of *samples* from the continuous signal, taken at regularly spaced intervals.

Useful approximations require that the coninuous signal be sufficiently smooth relative to the sampling interval so that one can approximately reconstruct the continuous signal (cf., the sampling theorem).
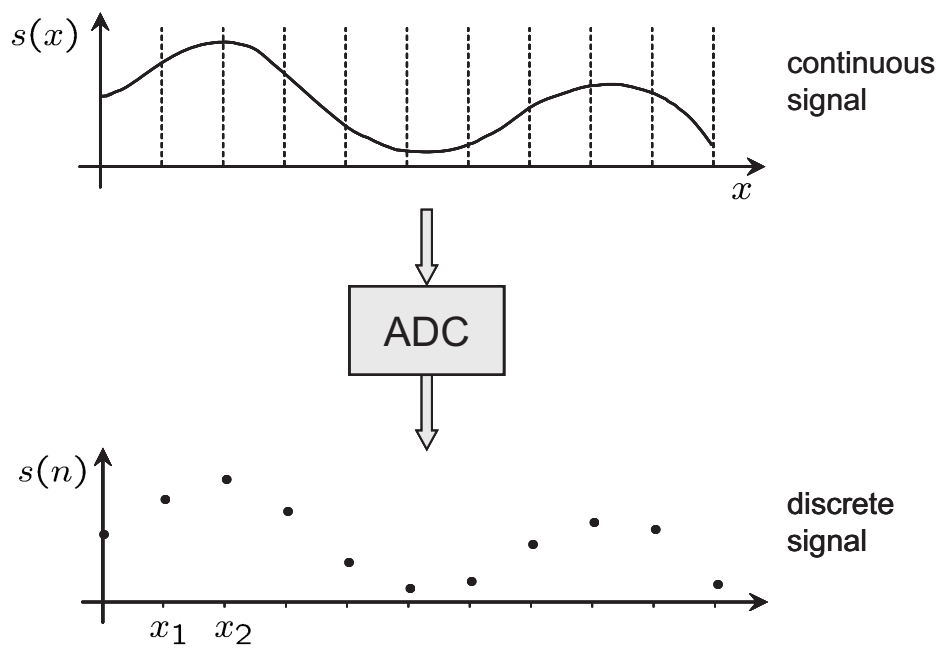

Notation: We will often express a 1D discrete signal, $s(n)$, for $n = 0...N-1$, as real-valued vector, $\vec{s} \in \mathcal{R}^N$.

Our signals of interest are (1) irradiance as a function of image position, and (2) sampled and quantized versions of it (i.e., arrays of numbers that represent the colour at each 2D image location). Initially we'll work with 1D signals; think of a column of pixels from an image. We'll generalize to 2D after the basics are introduced.

Introduction to Linear Filters

A filter transforms one signal into another, often to enhance certain properties (e.g., edges), remove noise, or compute signal statistics.
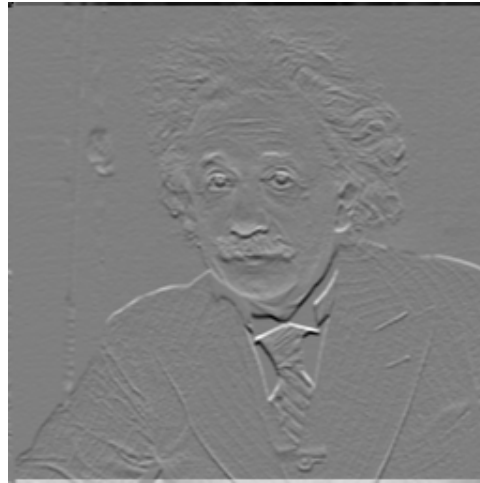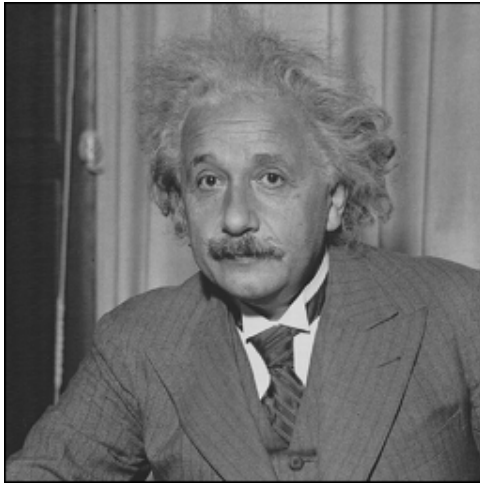
A transformation T, is linear iff, for inputs $s_i(n)$, responses $r_i(n) = T[s_i(n)]$, and scalars a and b, T satisfies **superposition:**

$$T[as_1(n) + bs_2(n)] = a\,T[s_1(n)] + b\,T[s_2(n)] \quad \forall a, b \in \mathcal{C}$$

In 1D, a linear filter can be represented by a matrix, A, and its response $\vec{\mathbf{r}}$ to input $\vec{\mathbf{s}}$ is given by matrix multiplication:

$$\vec{\mathbf{r}} \;=\; A\,\vec{\mathbf{s}}$$

The mth element of $\vec{\mathbf{r}}$ is the inner product of the mth row of A and $\vec{\mathbf{s}}$.

Shift Invariance

Often we want to apply the same operation to every point in an image (e.g., smoothing). If T is shift-invariant, then $\forall m \in \mathcal{I}$

$$r(n) = T\left[s(n)\right] \quad \text{iff} \quad r(n-m) = T\left[s(n-m)\right]$$

Linear, shift-invariant filters can be expressed as Toeplitz matrices (i.e., constant along diagonals):

- so each row is equal to the previous row, but shifted right by one, and each column is a shifted version of every other column.

- E.g.: Let's smooth a signal by computing a weighted average of each input sample and its two neighbours with weights 0.25, 0.5, and 0.25 (i.e., with a sliding window). The corresponding matrix has the form:

$$A = \frac{1}{4} \begin{bmatrix} \ldots & 0 & 1 & 2 & 1 & 0 & \ldots & & \\ & \ldots & 0 & 1 & 2 & 1 & 0 & \ldots & \\ & & \ldots & 0 & 1 & 2 & 1 & 0 & \ldots \end{bmatrix}$$

Local filters compute responses using only small neighborhoods of pixels from the input, like the smoothing filter. For 1D signals this produces a banded Toeplitz matrix. The width of nonzero entries in a row is called the filter's *support*.

Boundary Conditions

With finite length signals we need to handle boundaries carefully.

1. Shift-invariance is preserved if we assume *periodic signals and cyclical shifts*. For local filters this introduces nonzero entries in the upper-right and lower-left corners of the matrix. E.g.:

$$\frac{1}{4}\begin{bmatrix} 2 & 1 & 0 & \dots & & 1 \\ 1 & 2 & 1 & 0 & \dots & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \ddots \\ & & \dots & 0 & 1 & 2 & 1 \\ 1 & & & \dots & 0 & 1 & 2 \end{bmatrix}$$

2. We could instead assume that the input is always zero beyond its endpoints. In practice, the number of zeros we use to *pad* the input depends on the filter's support width. If the support is M samples, then we need $M-1$ zeros on each end.

   The response is then longer than the input by $M-1$ samples, so people often just truncate the response. The transform is then:

$$\frac{1}{4}\begin{bmatrix} 2 & 1 & 0 & \dots & \\ 1 & 2 & 1 & 0 & \dots \\ & & \ddots & \ddots & \ddots \\ & & & \ddots & \ddots & \ddots \\ & & \dots & 0 & 1 & 2 & 1 \\ & & & \dots & 0 & 1 & 2 \end{bmatrix}$$

   But then this is no longer shift-invariant.

3. Often it is more desirable to assume a constant signal beyond the boundary, i.e., pad the ends by repeating the two end samples.

Impulse Response

One can also characterize a linear, shift-invariant operator with its impulse response, i.e., the response to an impulse, δ:

- Kronecker delta function (discrete)

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$$

- Dirac delta function (continuous)

$$\delta(x) = 0 \quad \forall x \neq 0, \quad \text{and} \quad \int \delta(x) f(x)\, dx = f(0)$$

for sufficiently smooth $f(x)$

In the discrete case, multiplying A by the delta function $\delta(n)$ simply extracts the first column from A:

$$\vec{\mathbf{h}} = A\, \vec{\mathbf{e}}_1 \;, \quad \vec{\mathbf{e}}_1 = [1, 0, \ldots, 0]^t$$

$h(n)$ is called the *impulse response*. (If we pad the input boundaries with zeros and truncate the result, then the origin should be near the middle of the vector so we don't get a truncated impulse response!)

- Applying A to a shifted impulse signal, $\delta(n - m)$, gives us the mth column of A, ie., a shifted version of the impulse response.

- Therefore, from the impulse response we can construct the matrix; i.e., it contains all the information needed to define the filter.

Convolution

The conventional way to express a linear shift-invariant filter mathematically is with the convolution operator. Let the scalar w_p denote the signal value at position p, i.e., $w_p = s(p)$. Then we write $s(n)$ as

$$s(n) = \sum_{p=-\infty}^{\infty} w_p\, \delta(n-p)$$

For a linear, shift-invariant operator T it follows that

$$\begin{aligned}
T\left[s(n)\right] &= T\left[\sum_{p=-\infty}^{\infty} w_p\, \delta(n-p)\right] \\
&= \sum_{p=-\infty}^{\infty} w_p\, T\left[\delta(n-p)\right] \\
&= \sum_{p=-\infty}^{\infty} w_p\, h(n-p)
\end{aligned}$$

where h is the impulse response. Thus the response $r(n) = T[s(n)]$ is just a weighted sum of shifted impulse responses, that is:

$$r(n) = \sum_{p=-\infty}^{\infty} s(p)\, h(n-p) \tag{1}$$

Eqn (1) is called convolution, and is expressed as a binary operator (often with $*$):

$$s * h \equiv \sum_{p=-\infty}^{\infty} s(p)\, h(n-p)\,.$$

For continuous signals, $h(x)$ and $s(x)$, convolution is written as

$$s * h = \int_{-\infty}^{\infty} s(\xi)\, h(x-\xi)\, d\xi$$

Properties of Convolution

Commutativity: $s * h = h * s$

Not all matrix operations commute, but this does.

Associativity: $(h_1 * h_2) * h_3 = h_1 * (h_2 * h_3)$

This is true of all matrix multiplication.

Distributivity over Addition: $(h_1 + h_2) * h_3 = h_1 * h_3 + h_2 * h_3$

This is true of all matrix multiplication.

Local Support: Often the support of the filter is limited. If $h(m)$ is only nonzero for $-M/2 \leq m \leq M/2$, then we rewrite Eqn (1) as

$$s * h = \sum_{p=-M/2}^{M/2} s(n + p)\, h(-p)\,.$$

In words, for each signal position, n, center the reflected impulse response at position n, and then take its inner product with the image. This is a better way to implement the filter than matrix multiplication!

Inverse: One way to find the inverse of a convolution operator is to create the corresponding Toeplitz matrix and invert it. One can show that the inverse of a cyclic Toepliz matrix is also cyclic and Toepliz. In other words, the inverse of a discrete linear shift-invariant operator, if it exists, is also linear and shift-invariant. A better way to find the inverse uses the Fourier transform.

2D Image Convolution

In 2D the convolution equation is given by

$$r(n,m) \ = \ \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} s(p,q)\, h(n-p, m-q)$$

Computational Expense: In general, 2D convolution requires $O(N^2 M^2)$ multiplications and additions where N^2 is the number of image pixels, and M^2 is the 2D support of the impulse response.

Separability: If a 2D impulse response can be expressed as $h(x,y) = h_1(x)\, h_2(y)$ for some $h_1(x)$ and some $h_2(y)$, then h is said to be separable. In the discrete case, the impulse response is separable if it can be expressed as an outer product:

$$\begin{bmatrix} & & \\ & h[n,m] & \\ & & \end{bmatrix} \ = \ \begin{pmatrix} | \\ h_1[n] \\ | \end{pmatrix} \begin{pmatrix} -\!\!-\ h_2[m]\ -\!\!- \end{pmatrix}$$

With separability, 2D convolution can be expressed as a cascade of 1D convolutions, first along the rows, and then along the columns. Because convolution is commutative you could convolve along the columns and then the rows.

- Each 1D convolution, and hence the separable 2D filter, is $O(N^2 M)$. This is important if the filter support is more than 4 or 5 pixels.

2D Image Convolution

Examples:

- The $m \times m$ constant matrix (a crude 2d averaging operator) can be expressed as an outer product of two 1d constant vectors. (But this is not isotropic.)

- In continuous terms, the Gaussian is the only 2d isotropic function that can be decomposed into a separable product of two 1d Gaussians:
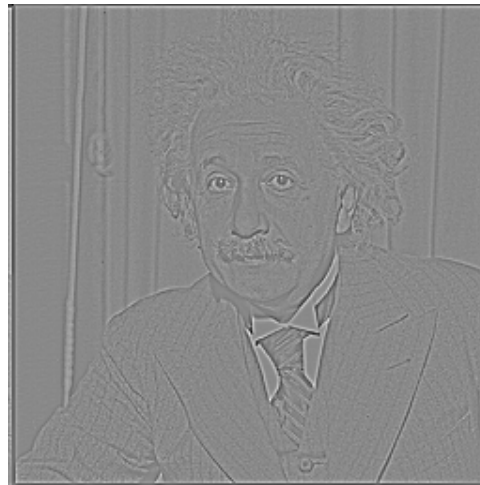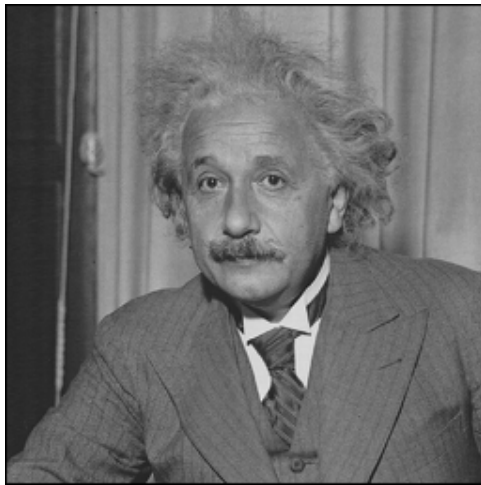
$$\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2} \;=\; \frac{1}{\sqrt{2\pi}\sigma}e^{-x^2/2\sigma^2} \; \frac{1}{\sqrt{2\pi}\sigma}e^{-y^2/2\sigma^2}$$

- Discrete approximations to Gaussians are often given by by binomial coefficients, (e.g, (1, 4, 6, 4, 1)/16)).
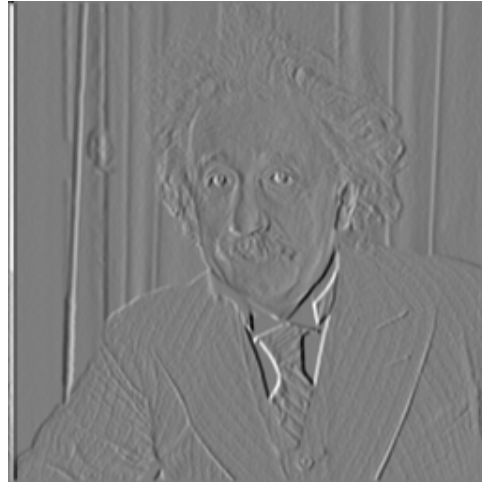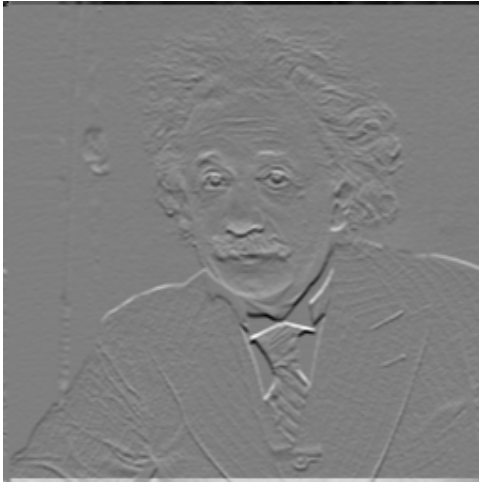
Example: Smoothing Filters
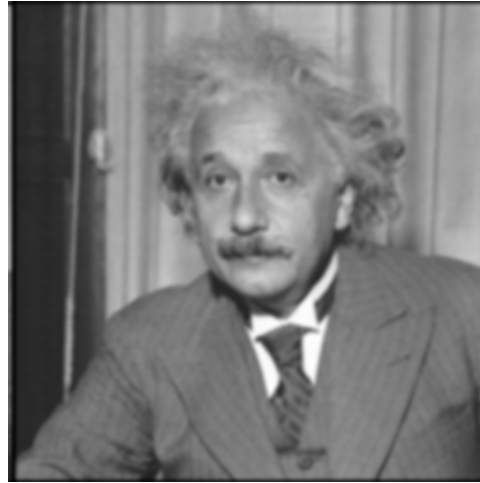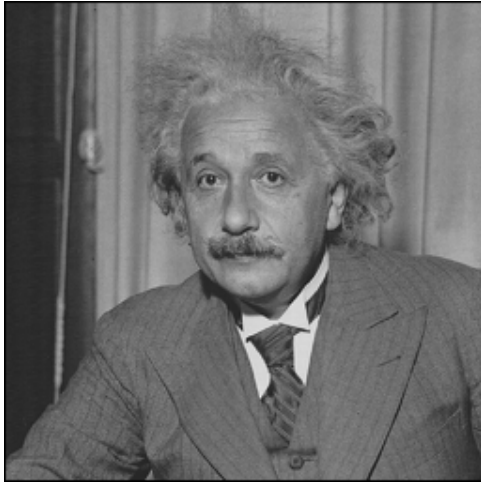
Al and a blurred version of Al are shown. The impulse response was separable, composed of the same horizontal and vertical 5-tap 1D impulse response, that is, $\frac{1}{16}(1, 4, 6, 4, 1)$.


This shows Al and the difference between Al and the blurred version of Al. The image is only non-zero where the blurred version is different from the original, i.e., where there are significant local intensity variations. The impulse response for this filter is $\delta(n, m) - h(n, m)$ where $h(n, m)$ is the separable blurring filter used above.

Example: Derivative Filters

Derivative filters are common in image processing. Here we use crude separable approximations to horizontal and vertical derivatives. They are composed of a smoothing filter in one direction (i.e., $\frac{1}{4}(1, 2, 1)$) and a first-order central difference (i.e., $\frac{1}{2}(-1, 0, 1)$) in the other.
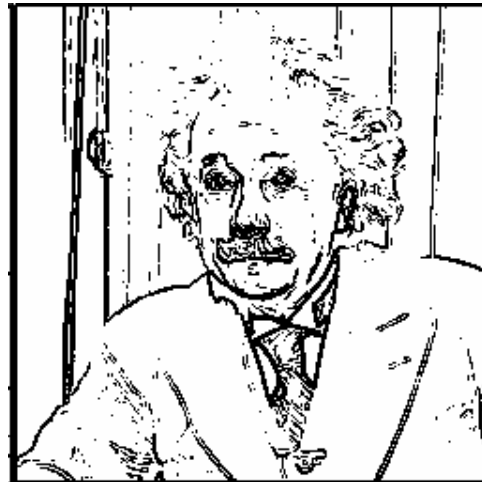
Sum of squared derivative responses (the squared magnitude of the image gradient at each pixel). When clipped, this gives a rough idea of where edges might be found.

Example: Down-Sampling and Up-Sampling

Down-sampling (or decimation) is the process of collapsing a signal by removing every n^{th} sample.

Up-sampling refers to the expansion of a signal by adding new samples to make it longer. One introduces n zeros in between every pair of adjacent samples in the original signal.

Both of these operators are linear.

Example: down-sampling a signal $\vec{s}$ by a factor of 2 to create $\vec{s}_2$.

$$\vec{s}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ & & & & \vdots & & & \end{bmatrix} \vec{s}$$

Example: up-sampling a signal $\vec{s}$ by a factor of 2 to create $\vec{s}_1$.

$$\vec{s}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ & & & & \vdots & & & \end{bmatrix} \vec{s}$$

Up-sampling is often a precursor to smoothing for signal interpolation.

Introduction to Fourier Analysis

Overview

- Fourier transform of an signal is a decomposition of the signal into a weighted sum of sinusoids.
- We'll concentrate on Fourier transforms for discrete signals (see detailed notes for continuous transforms).

Discrete Sinusoids

- For now, consider sinusoids that are periodic on N samples.

$$I(n) = A\sin(\omega n + \phi)$$

   - A is the amplitude
   - ϕ is the phase offset
   - ω is frequency, with wavelength $\lambda = 2\pi/\omega$.

- Two results:

   1. Frequency ω is only unique between 0 and 2π.
      Proof: because n is an integer $\sin((\omega + 2\pi)n) = \sin(\omega n)$.

   2. Assuming periodic signals of length N, then there are only N distinct frequencies: $\omega_k \equiv 2\pi k/N$ for $0 \le k < N$.

- Euler's formula: $e^{i\omega n} = \cos(\omega n) + i\,\sin(\omega n)$ where $i^2 = -1$.
  Conversely, one can write $\cos(\omega n) = \frac{1}{2}\left[e^{i\omega n} + e^{-i\omega n}\right]$

Eigenfunctions of Convolution

Convolution of a sinusoid of frequency ω_k, $e^{i\omega_k n}$, and filter $h(n)$ is:

$$r(n) = \sum_{m=0}^{N-1} e^{i\omega_k(n-m)} h(m) = e^{i\omega_k n} \sum_{m=0}^{N-1} e^{-i\omega_k m} h(m)$$

This response is just a scaled version of the original sinusoid, i.e.,

$$r(n) = e^{i\omega_k n} H_k$$

where the scalar, H_k, is the inner product of $h(n)$ and a sinusoid $f_k(n) = e^{-i\omega_k n}$. In vector notation, $H_k = \vec{\mathbf{f}}_k^{\,t} \vec{\mathbf{h}}$.

We can collect the complex-valued scalars associated with sinusoids at all N frequencies into a vector, $\vec{\mathbf{H}} = [H_0, ..., H_{N-1}]^t$, and then:

$$\vec{\mathbf{H}} = \mathbf{F}\,\vec{\mathbf{h}}, \quad \text{where} \quad \mathbf{F} = \begin{bmatrix} \vec{\mathbf{f}}_0^{\,t} \\ \vdots \\ \vec{\mathbf{f}}_{N-1}^{\,t} \end{bmatrix} \tag{2}$$

$\mathbf{F}$ is a unitary matrix (up to a scalar); its inverse is given by

$$\mathbf{F}^{-1} = \frac{1}{N} \mathbf{F}^{*t} = \frac{1}{N} \left[\vec{\mathbf{f}}_0^{\,*}, ..., \vec{\mathbf{f}}_{N-1}^{\,*} \right]$$

where $\mathbf{F}^{*t}$ is the conjugate transpose of $\mathbf{F}$.

- To prove this, show that $(1/N)\mathbf{F}\mathbf{F}^{*t} = Identity(N)$. Use the fact that sinusoidal signals of different frequencies ω_k are orthogonal, while the product of a complex sinusoid with its complex conjugate is simply $e^{i\omega n} e^{-i\omega n} = e^{i0} = 1$.
- one can also show that $\mathbf{F}$ is symmetric.

Discrete Fourier Transform

If we multiply both sides of the forward equation,

$$\vec{\mathbf{H}} = \mathbf{F}\,\vec{\mathbf{h}}\;, \quad \text{or equivalently}\,, \quad H(k) \;=\; \sum_{n=0}^{N-1} e^{-i\,\omega_k\,n} h(n) \qquad (3)$$

by $\mathbf{F}^{-1}$ we then obtain:

$$\vec{\mathbf{h}} \;=\; \frac{1}{N}\,\mathbf{F}^{*t}\,\vec{\mathbf{H}}\;, \quad \text{or equivalently,}\;\; h(n) \;=\; \frac{1}{N}\sum_{k=0}^{N-1} e^{i\,\omega_k\,n}\,H(k) \quad (4)$$

So what have we shown?

1. Eqn. (4) expresses h as a sum of N sinusoids. No more than N are required. The weights are called Fourier coefficients, and they are obtained by multiplying the signal $\vec{\mathbf{h}}$ with the matrix $\mathbf{F}$.

2. Eqn. (3) is the discrete Fourier transform (DFT) equation. $\mathbf{F}$ is the DFT matrix, and $H(k)$ is called the DFT of $h(n)$.

3. Eqn. (4) is called the inverse DFT equation.

4. Although we derived the DFT for an impulse response $h(n)$, the derivation can be applied to any discrete signal $s(n)$ of length N.

Fast Fourier Transform: Forward and inverse DFT can be computed in $O(N \log N)$. (Don't use matrix multiplication with $\mathbf{F}$)

Inverse DFT Matrices
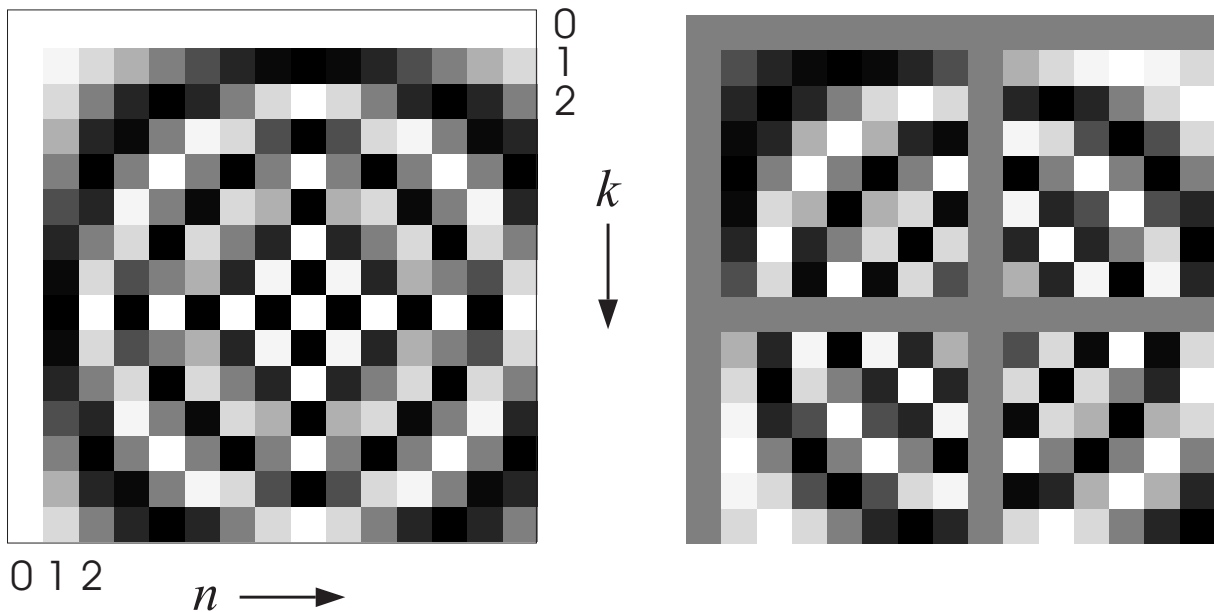
Left: This shows the real part of the inverse DFT matrix, $\cos(2\pi nk/16)$, with frequencies $\omega_k = 2\pi k/16$ for $0 \le k \le 15$ from top to bottom. The spatial variable n increases from 0 to 15. The first row and column are filled with ones.

Right: This depicts the imaginary part of the inverse DFT matrix, $-\sin(2\pi nk/16)$, with the same dependence on k and n. In this case, the first row and column are filled with zeros.

DFT Example
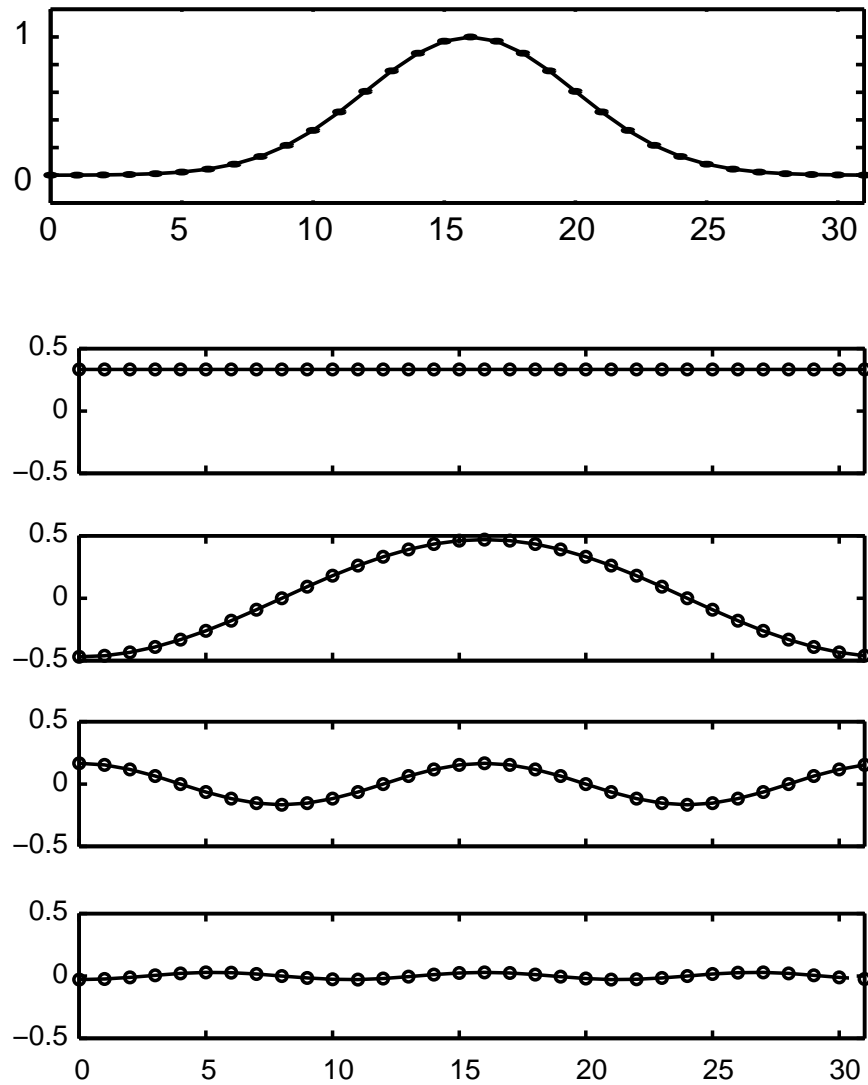
A Gaussian signal is shown on top with 32 samples. The first 4 terms of its Fourier decomposition are shown below, with frequencies $2\pi k/N$ for $0 \le k \le 3$. The amplitudes of these first 4 Fourier coefficients are 0.3333, 0.47, 0.166, and 0.029.

DFT Examples

E.g. 1: DFT of $s(n) = A\cos(\omega_3 n + \phi)$ for some real-valued A and ϕ.

- first note that $s(n) = \frac{1}{2}\left(e^{i(\omega_3 n + \phi)} + e^{-i(\omega_3 n + \phi)}\right)$

- because frequencies are unique within 2π, also note that $\omega_{N-3} = -\omega_3$.

- Finally, rows of DFT matrix are orthogonal, and therefore the matrix-vector product of $\mathbf{F}$ with $s(n)$ will have nonzero values at positions $k = 3$ and $k = N - 3$.

- what will these values be? (Check your answer with linSysTutorial.m)

E.g. 2: DFT of $\delta(n - n_0)$.

- first note that $\mathbf{F}$ is symmetric, so rows as well as columns of $\mathbf{F}$ are sinusoids, i.e., $e^{-i\omega_k n}$.

- from this it's clear that the product of $\mathbf{F}$ with an impulse at position n_0 is simply the column corresponding to that position, i.e., $e^{-i\omega_k n_0}$.

Related Transforms

Discrete-Time Fourier Transform (DTFT): For signals of infinite length the spectrum becomes continuous on $\omega \in (0, 2\pi]$. The transform and its inverse are given by:

$$\hat{s}(\omega) = \sum_{n=-\infty}^{\infty} s(n) \, e^{-i\omega n} \quad , \quad s(n) = \frac{1}{2\pi} \int_0^{2\pi} \hat{s}(\omega) \, e^{i\omega n} d\omega$$

Fourier Series: For (periodic) continuous signals on a finite interval $(0, 1)$, the transform becomes an infinite Fourier series:

$$\hat{s}(k) = \int_0^1 s(x) \, e^{-i\omega_k x} dx \quad , \quad s(x) = \sum_{k=-\infty}^{\infty} \hat{s}(k) \, e^{i\omega_k x}$$

for $\omega_k = 2\pi k$.

Amplitude and Phase Spectra: Fourier transforms $\hat{f}(\omega) = \mathcal{F}[f(x)]$ are complex-valued in general. Accordingly, as is common with complex numbers z, it is often convenient to express them in terms of magnitude and phase, as in $\rho e^{i\phi}$. It is therefore common to factor Fourier spectra into amplitude and phase spectra, that is,

$$\rho(\omega) = |\mathcal{F}[f(x)]| \quad , \quad \phi(\omega) = \arg(\mathcal{F}[f(x)])$$

Discrete-Time Fourier Transform (DTFT)

As length of the discrete signal grows the number of Fourier coefficients grows, since the number of Fourier coefficients in the DFT is equal to the number of samples in the signal. In the limit the sampling of frequencies between 0 and 2π becomes dense, and the Fourier transform becomes a continuous function of frequency.

For discrete signals of finite length we can implicitly increase the length of the signal by padding the ends with zeros. In the limit we also obtain the discrete-time Fourier transform, continuous over the same range of unique frequecies, from 0 to 2π, or equivalently from $-\pi$ to π.

Here, the DTFT, for $0 \leq \omega < 2\pi$, is given by

$$\hat{I}(\omega) \;=\; \sum_{n=-\infty}^{\infty} I(n)\, e^{-i\omega n}$$

The inverse DTFT, with which we reconstruct the signal, is

$$I(n) \;=\; \frac{1}{2\pi} \int_0^{2\pi} \hat{I}(\omega) e^{i\omega n} d\omega$$

One can show that, when we take a signal and pad it with zeros out to infinity to obtain the DTFT, the DFT of the unpadded signal is simply a sampled version of DTFT.

Uses:

- For an impulse response, the DTFT tells us how the filter behaves when applied to any frequency. We can then understand the filter's behaviour with signals of arbitrary length.

- One way to compute an approximation to the DTFT is to add more rows to the DFT matrix, at additional frequencies. That is, there is no reason to acutally pad with zeros; rather, we simply construct a non-square transform matrix.

- One can also find analytic solutions. E.g., let $h(n) = \frac{1}{4}(1, 2, 1)$. Then,

$$\hat{I}(\omega) \;=\; \sum_{n=-\infty}^{\infty} h(n)\, e^{-i\omega n} \;=\; \sum_{n=-1}^{1} h(n)\, e^{-i\omega n} \;=\; \frac{1}{4}\left(e^{i\omega} + 1 + e^{-i\omega}\right) \;=\; \frac{1}{2}(1 + \cos(\omega))$$

Fourier Series

For continuous signals there are corresponding Fourier transforms. The main case of interest in practice concerns the transforms of continuous signals of finite length (such as images). Suppose $s(t)$ is a bounded, complex-valued signal for $x \in [0, 1]$. Then $s(t)$ can be represented as a sum of basis functions:

$$s(x) =_{almost\ everywhere} \sum_{k=-\infty}^{\infty} a_k b_k(x). \qquad \text{(Fourier Series)}$$

In particular, the Fourier basis is given by

$$b_k(x) \equiv e^{i2\pi kx} = \cos(2\pi kx) + i\sin(2\pi kx),\, k \in \mathcal{Z}.$$

Like the discrete case, the basis functions in the Fourier series have several desirable properties:

- **Orthogonality:**

$$
\begin{aligned}
\langle b_k(x), b_l(x) \rangle &\equiv \int_0^1 b_k^*(x) b_l(x) dx \\
&= \int_0^1 e^{-i2\pi kx} e^{i2\pi lx} dx = \int_0^1 e^{i2\pi(k-l)x} dx \\
&= \delta_{k,l}.
\end{aligned}
$$

- **Periodicity:**

$$b_k(t) = b_k(t+1).$$

  The signal $\sum_{k=-\infty}^{\infty} a_k b_k(x)$ is periodically extended for x beyond $[0, 1]$.

- **Shift Invariance:**

$$b_k(x+c) = \lambda(c) b_k(x),$$

  with $\lambda(c) = e^{i2\pi kc}$.

Fourier Transform and Its Inverse (i.e., the Fourier Series):

$$S(k) = \langle b_k(x), s(x) \rangle \equiv \int_0^1 b_k^*(x) s(x) dx \quad, \qquad s(x) =_{a.e.} \sum_{k=-\infty}^{\infty} S(k)\, b_k(x)$$

2D Fourier Transforms

In 2D, for signals $h(n, m)$ with N columns and M rows, the idea is exactly the same:

$$\hat{h}(k, l) \;=\; \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} e^{-i(\omega_k\, n + \omega_l\, m)}\, h(n, m)$$

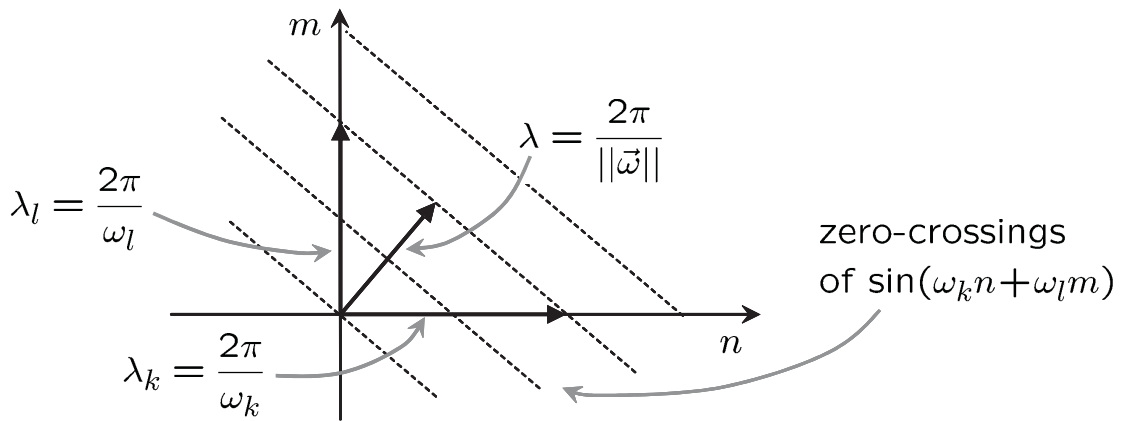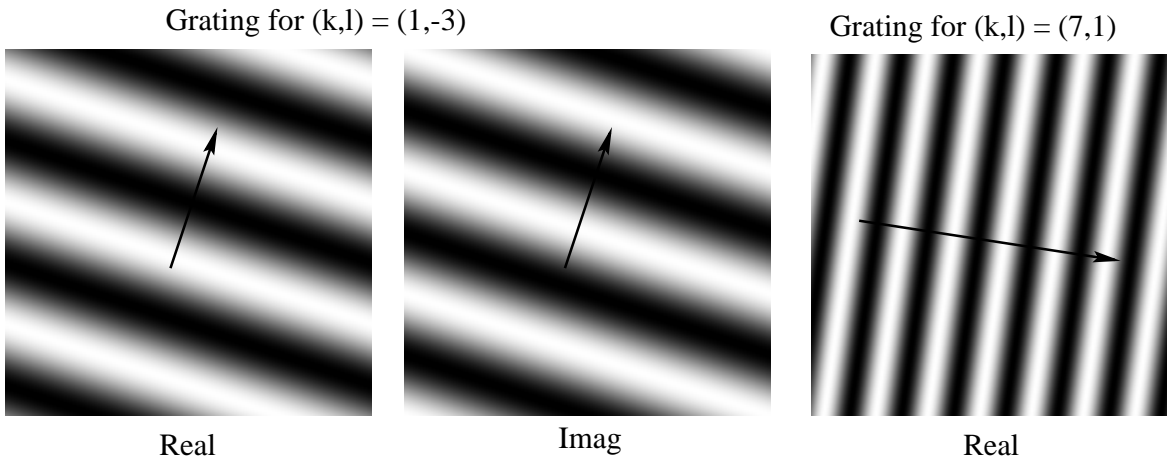$$h(n, m) \;=\; \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} e^{i(\omega_k\, n + \omega_l\, m)}\, \hat{h}(k, l)$$

Often it is convenient to express frequency in vector notation with $\vec{k} = (k, l)^t$, $\vec{n} = (n, m)^t$, $\vec{\omega}_{kl} = (\omega_k, \omega_l)^t$ and $\vec{\omega}^t \vec{n} = \omega_k\, n + \omega_l\, m$.

2D Fourier Basis Functions: Sinusoidal waveforms of different wavelengths (scales) and orientations. Sinusoids on $N \times M$ images with 2D frequency $\vec{\omega}_{kl} = (\omega_k,\, \omega_l) = 2\pi(k/N,\, l/M)$ are given by:

$$e^{i\,(\vec{\omega}^t \vec{n})} \;=\; e^{i\,\omega_k n}\, e^{i\,\omega_l m} \;=\; \cos(\vec{\omega}^t \vec{n}) + i\sin(\vec{\omega}^t \vec{n})$$

Separability: If $h(\vec{n})$ is separable, e.g., $h(n, m) = f(n)\, g(m)$, then, because complex exponentials are also separable, so is the Fourier spectrum, $\hat{h}(k, l) \;=\; \hat{f}(k)\, \hat{g}(l)$.

2D Fourier Basis Functions

Grating for (k,l) = (1,-3)

Grating for (k,l) = (7,1)

Real   Imag   Real

$$\lambda_l = \frac{2\pi}{\omega_l}$$

$$\lambda = \frac{2\pi}{||\vec{\omega}||}$$

$$\lambda_k = \frac{2\pi}{\omega_k}$$

zero-crossings
of $\sin(\omega_k n + \omega_l m)$

Blocks image and its amplitude spectrum
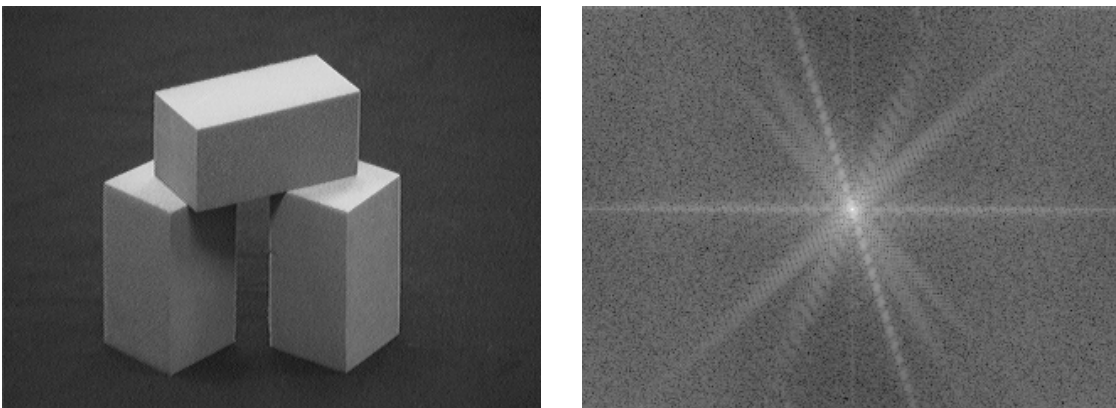
Properties of the Fourier Transform

Some key properties of the Fourier transform, $\hat{f}(\vec{\omega}) = \mathcal{F}[f(\vec{\mathbf{x}})]$.

Symmetries:

For $s(x) \in \mathcal{R}$, the Fourier transform is symmetric, i.e., $\hat{s}(\omega) = \hat{s}^*(-\omega)$.

For $s(x) = s(-x)$ the transform is real-valued, i.e., $\hat{s}(\omega) \in \mathcal{R}$.

For $s(x) = -s(-x)$ the transform is imaginary, i.e., $i\,\hat{s}(\omega) \in \mathcal{R}$.

Shift Property:

$$\mathcal{F}\left[f(\vec{\mathbf{x}} - \vec{\mathbf{x}}_0)\right] = exp(-i\,\vec{\omega}^t\vec{\mathbf{x}}_0)\,\hat{f}(\vec{\omega}) \tag{5}$$

The amplitude spectrum is invariant to translation. The phase spectrum is not. In particular, note that $\mathcal{F}[\delta(\vec{\mathbf{x}} - \vec{\mathbf{x}}_0)] = exp(-i\,\vec{\omega}^t\vec{\mathbf{x}}_0)$. Proof: substitution and change of variables.

Differentiation:

$$\mathcal{F}\left[\frac{\partial^n f(\vec{\mathbf{x}})}{\partial x_j{}^n}\right] = (i\,\omega_j)^n\,\hat{f}(\vec{\omega}) \tag{6}$$

For intuition, remember that $\frac{\partial e^{i\,\omega x}}{\partial x} = i\,\omega e^{i\,\omega x}$ and $\frac{\partial \sin(\omega x)}{\partial x} = \omega \cos(\omega x)$.

Linear Scaling: Scaling the signal domain causes scaling of the Fourier domain; i.e., given $a \in \mathcal{R}$, $\mathcal{F}[s(ax)] = \frac{1}{a}\hat{s}(\omega/a)$.

Parseval's Theorem: Sum of squared Fourier coefficients is a constant multiple of the sum of squared signal values.

Convolution Theorem

The Fourier transform of the convolution of two signals is equal to the product of their Fourier transforms:

$$\mathcal{F}[f * g] \;=\; \mathcal{F}[f]\,\mathcal{F}[g] \;\equiv\; \hat{f}(\omega)\,\hat{g}(\omega)\,. \tag{7}$$

Proof in the discrete 1D case:

$$
\begin{aligned}
\mathcal{F}[f * g] = \sum_n f * g\, e^{-i\omega n} &= \sum_n \sum_m f(m)\, g(n-m)\, e^{-i\omega n}\\
&= \sum_m f(m) \sum_n g(n-m) e^{-i\omega n}\\
&= \sum_m f(m)\, \hat{g}(\omega)\, e^{-i\omega m} \quad \text{(shift property)}\\
&= \hat{g}(\omega)\, \hat{f}(\omega)\,.
\end{aligned}
$$

Remarks:

- This theorem means that one can apply filters efficiently in the Fourier domain, with multiplication instead of convolution.

- Fourier spectra help characterize how different filters behave, by expressing both the impulse response and the signal in the Fourier domain (e.g, with the DTFT). The filter's amplitude spectrum tells us how each signal frequency will be attentuated. The filter's phase spectrum tells us how each sinusoidal signal component will be phase shifted in the response.

- Convolution theorem also helps prove properties. E.g. prove:

$$\frac{\partial}{\partial x}\,(h * g) \;=\; \frac{\partial h}{\partial x} * g \;=\; h * \frac{\partial g}{\partial x}$$

Common Filters and their Spectra
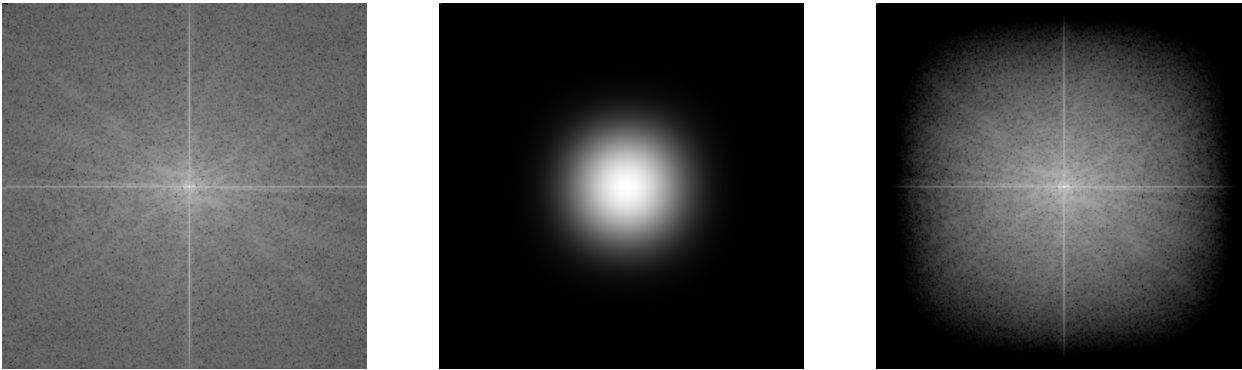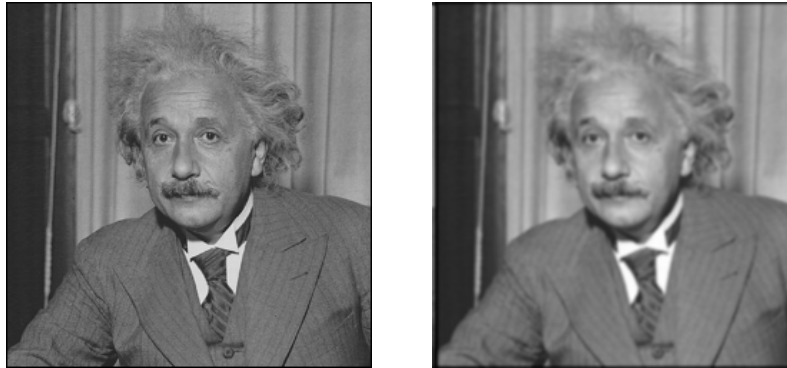
Top Row: Image of Al and a **low-pass** (blurred) version of it. The low-pass kernel was separable, composed of 5-tap 1D impulse responses $\frac{1}{16}(1, 4, 6, 4, 1)$ in the x and y directions.

Bottom Row: From left to right are the amplitude spectrum of Al, the amplitude spectrum of the impulse response, and the product of the two amplitude spectra, which is the amplitude spectrum of the blurred version of Al. (Brightness in the left and right images is proportional to log amplitude.)

Ideal Low-Pass Spectrum

The ideal low-pass filter is one that attenuates to zero all frequencies higher than a certain cut-off frequency, ω_s. For low frequencies, below the cut-off frequency, the signal components remain unchanged. This filter is central to signal reconstruction (of both discrete and continuous signals).

Let $|H(\omega)| = 1$ for $|\omega| < \omega_s$, and 0 otherwise.
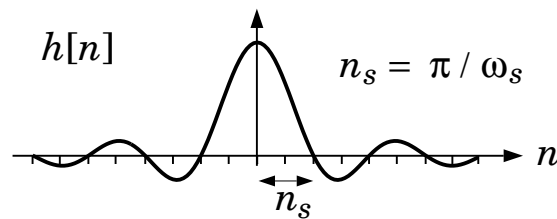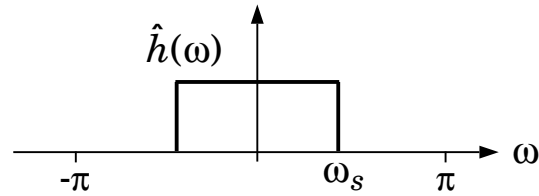
The impulse response can be shown to be given by:

$$h(n) \;=\; \frac{\sin(\pi n/n_s)}{\pi n/n_s}$$

where $n_s = \pi/\omega_s$.

Note:

- broad support of impulse response (not localized in space)

- causes ringing in the response to many simple image features such as lines and step edges.

# Common Filters and their Spectra (cont)
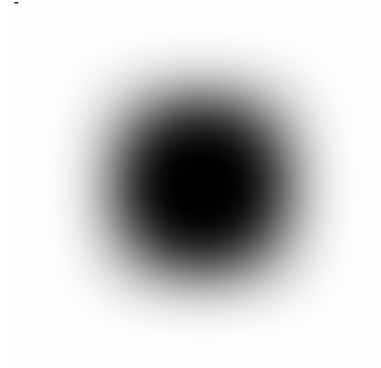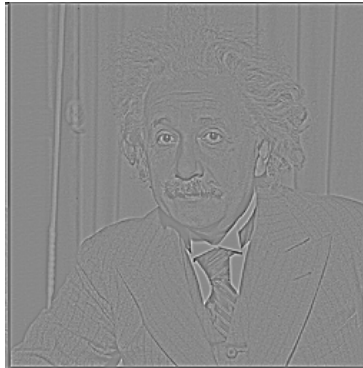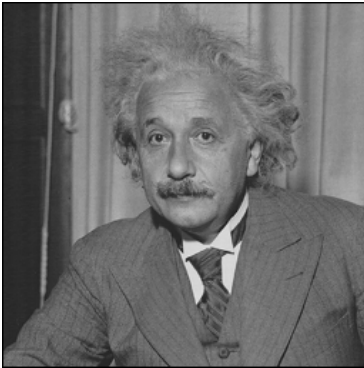
From left to right is the original Al, a **high-pass** filtered version of Al, and the amplitude spectrum of the filter. This impulse response is defined by $\delta(n) - h(n, m)$ where $h[n, m]$ is the separable blurring kernel used in the previous figure.
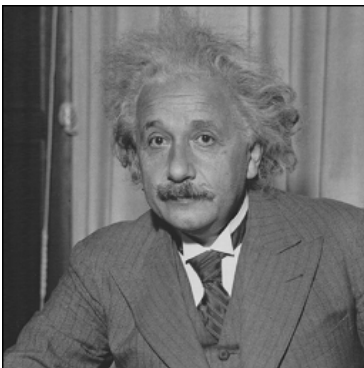
From left to right is the original Al, a **band-pass** filtered version of Al, and the amplitude spectrum of the filter. This impulse response is defined by the difference of two low-pass filters.

Common Filters and their Spectra (cont)
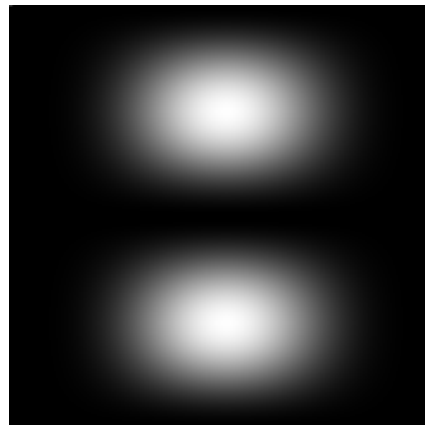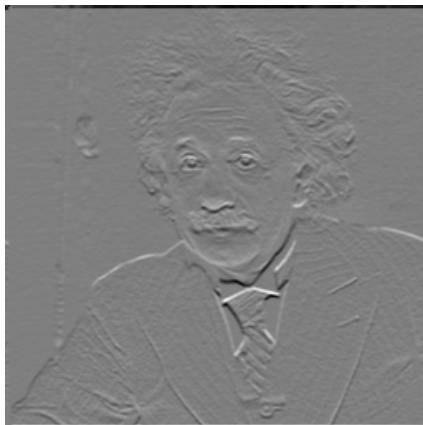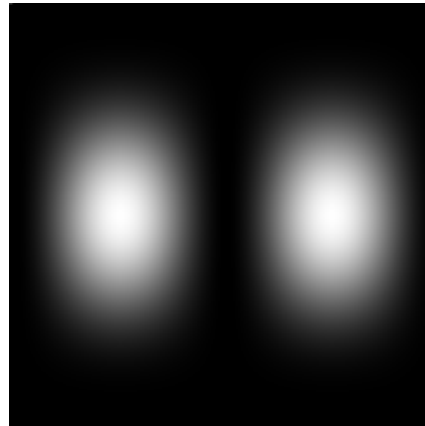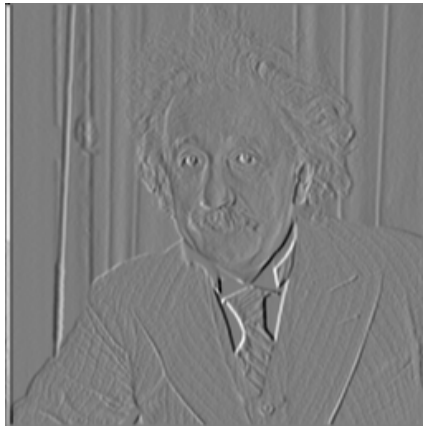
Top Row: Convolution of Al with a horizontal derivative filter, along with the filter's Fourier spectrum. The 2D separable filter is composed of a vertical smoothing filter (i.e., $\frac{1}{4}(1, 2, 1)$) and a first-order central difference (i.e., $\frac{1}{2}(-1, 0, 1)$) horizontally.

Bottom Row: Convolution of Al with a vertical derivative filter, and the filter's Fourier spectrum. The filter is composed of a horizontal smoothing filter and a vertical first-order central difference.

Quadrature-Pair Filters

Real-valued band-pass filters, like derivative filters, have symmetric amplitude spectra. Another important class of filters is complex-valued with non-zero Fourier power only in one half-plane of the Fourier domain.

The real and imaginary parts of such filters $f(x) \in \mathcal{C}$ have symmetric amplitude spectra, but they are out of phase so that one half of the Fourier spectrum is cancelled when they are combined as a complex-valued filter. The real and imaginary parts are often called a quadrature pair (or Hilbert transform pairs).


real part   imaginary part
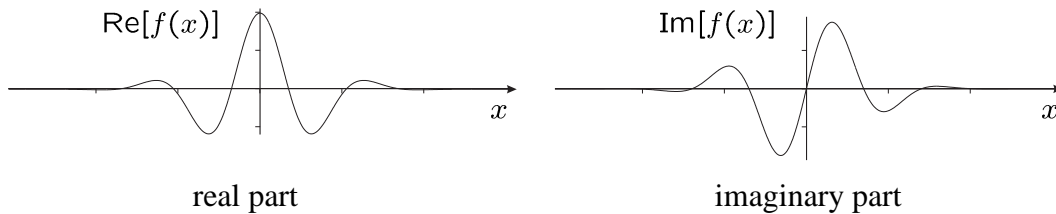
One of their useful properties is that one can easily separate the response amplitude from the phase of the response (effectively separating structure from amplitude). That is, because the filter is complex-valued, its response will also be complex valued. Hence the response can be written:

$$r(x) \;=\; a(x)\,e^{i\phi(x)}$$

For filters with somewhat narrow bandwidths (say less than 1.5 octaves), the amplitude is slowly varying and the phase is typically linear. The derivative of phase with respect to position, $d\phi/dx$ is often called the instantaneous frequency of the signal.

Discrete Cosine Transform

Another variation is the Discrete Cosine Transform. Here, instead of assuming the signal is periodic with period N, it is extended by successively reversing the signal from left to right so that it has the period $2N$. This avoids the large step discontinuities at the borders, which can dominate the Fourier transform.
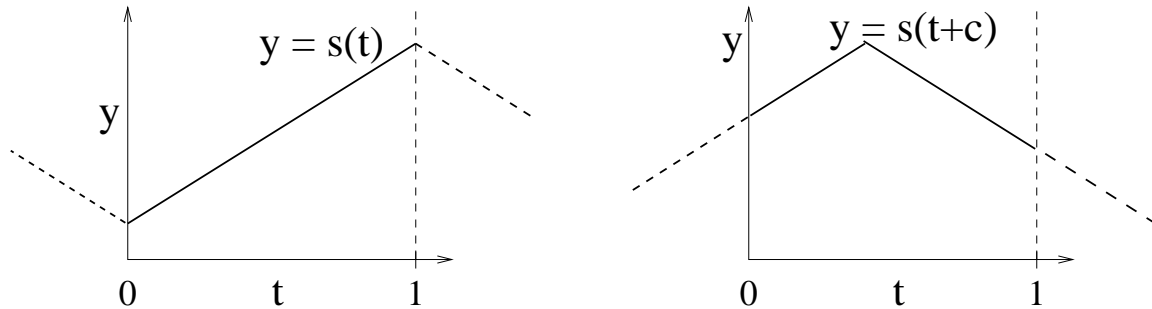

Here the signal is twice as long, but it is flip symmetric around the middle. As a consequence:

- The anti-symmetric sine terms cannot contribute, and only the cosine terms remain in the Fourier transform.

- The matrix for the transform is orthogonal and can be evaluated in $O(N \log N)$ time.

- The advantage of using the DCT is that the artifacts due to the borders of the image are decreased.

- JPEG compression uses the DCT within small image regions.

Example: Discrete Cosine Transform

The standard Fourier transform provides an amplitude spectrum consistent with the periodic signal:
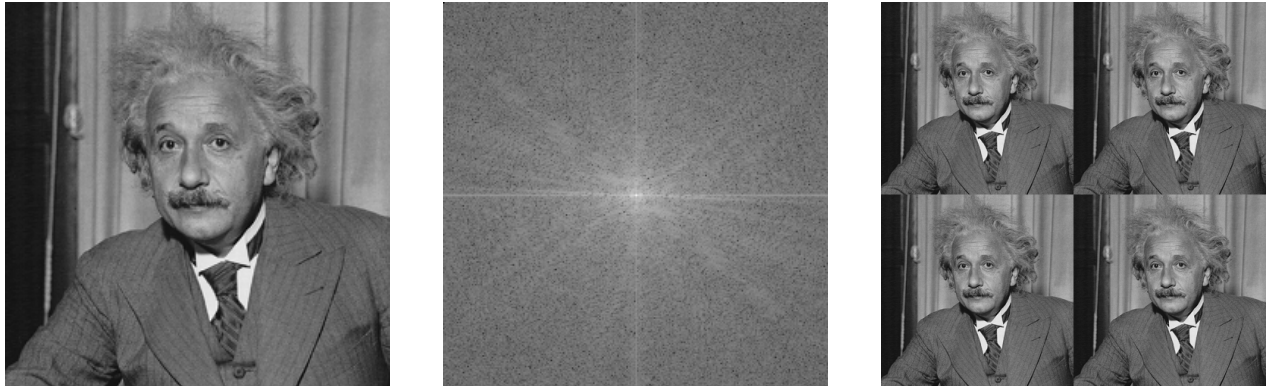
Note the significant response for horizontal and vertical gratings, which are artifacts due to the image borders.

The DCT transform tiles the image in a doubly periodic fashion which helps to reduce the large steps around the borders.
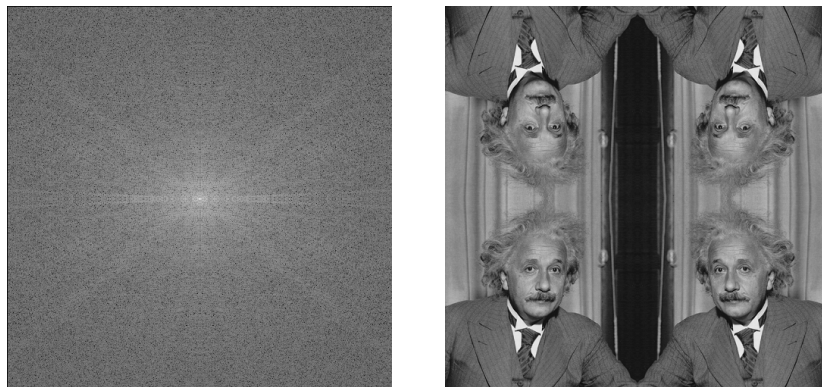

The DCT requires only one quadrant in this response. (We show all four for comparison purposes.) Note the responses for horizontal and vertical gratings have been significantly reduced.

Introduction to Sampling and Aliasing

*Sampling* – creating a discrete signal from a continuous signal.

Down-sampling (decimation) – subsampling a discrete signal

Up-sampling – placing zeros between existing samples
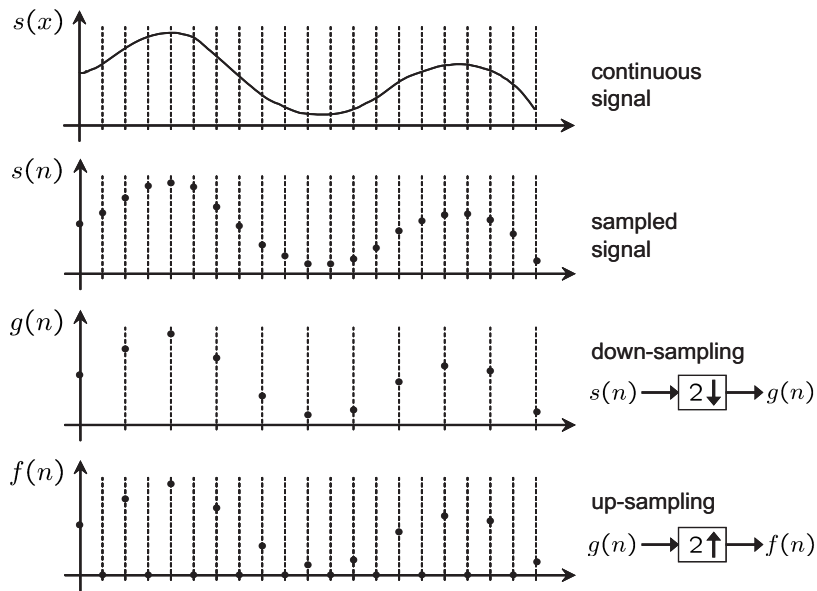
Aliasing – corruption of signal information caused by *under-sampling*.

When a signal is sampled too sparsely (e.g., at the image points denoted by ∘) then we will not be able to reconstruct the original signal because of aliasing.

Undersampled sensors
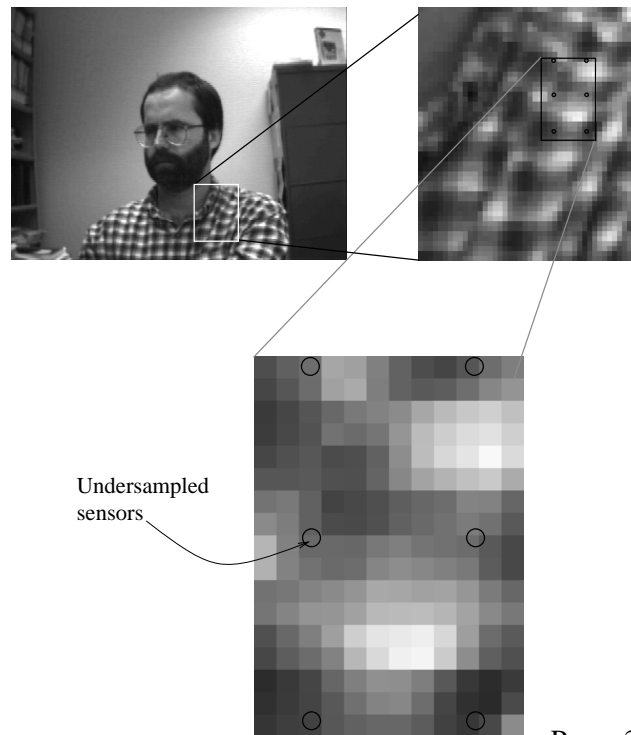
Down-Sampling

Let $f(n)$ be a signal of length N.

Assume we wish to reduce the number of samples by factor of n_s from N to $M = N/n_s$, where n_s is a divisor of N.
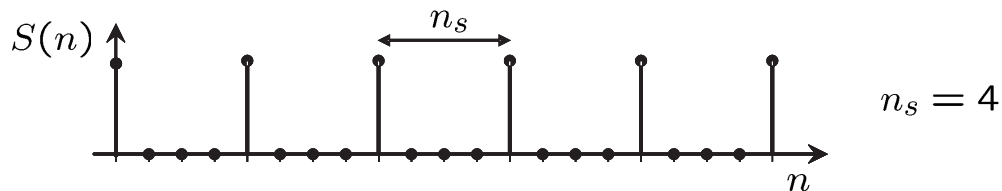
Let the new signal be $g(m)$.

We first multiply the original signal $f(n)$ with a sampling (comb) function that is only non-zero every n_s^{th} sample, i.e.:

$$f_s(n) = S(n) \, f(n)$$

where

$$S(n) = \sum_{m=0}^{(N/n_s)-1} \delta(n - m \, n_s)$$

The sampling function zeros out all but the values of f we wish to keep. It can be depicted by

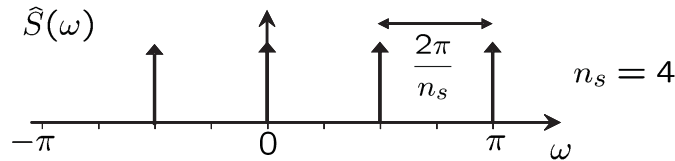Finally, the down-sampled signal $g(m)$ is given by

$$g(m) = f_s(m \, n_s)$$

Down-Sampling in the Fourier Domain

One can show that the Fourier transform of the sampling function is (note that $w = 2\pi k/N$):

$$\hat{S}(\omega) = \frac{N}{n_s} \sum_{m=0}^{n_s-1} \delta\left(\omega - m\frac{2\pi}{n_s}\right) .$$

This transform is defined uniquely on an interval of length 2π, but we can choose which interval. Here we center the spectrum at the origin:

Because multiplication in space is convolution in the Fourier domain, the Fourier transform of f_s is given by

$$\hat{f}_s(\omega) = \frac{1}{N}\hat{S}(\omega) * \hat{f}(\omega) = \frac{1}{n_s}\sum_{m=0}^{n_s-1}\hat{f}\left(\omega - m\frac{2\pi}{n_s}\right)$$

So, down-sampling introduces replicas of $\hat{f}(\omega)$ spaced every $2\pi/n_s$ (solid curves depict spectra & the dotted curves depict replicas):
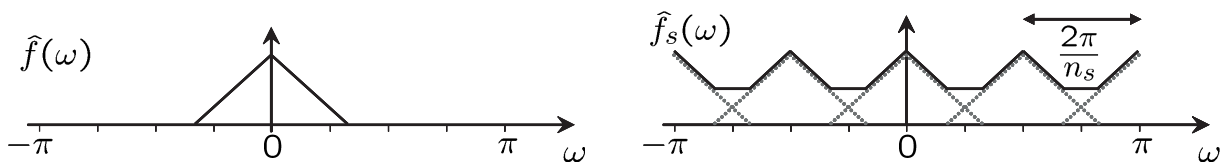
Finally, one can show that by removing the zeros in $f_s(n)$ we are simply contracting the signal, i.e., $g(m) = f_s(m\,n_s)$, so:

$$\hat{g}(\omega) = \hat{f}_s(\omega/n_s) = \frac{1}{n_s}\sum_{m=0}^{n_s-1}\hat{f}\left(\frac{\omega}{n_s} - m\frac{2\pi}{n_s}\right)$$

Up-Sampling

Up-Sampling refers to the introduction of zeros between the existing samples of a signal $g(n)$ to form a new signal $f(m)$. Typically this is followed by some form of smoothing in order that the resulting signal be a reasonable approximation to $g(n)$ (but at a higher resolution).

The introduction of $n_s - 1$ zeros between every two samples of $g(n)$ can be expressed as

$$f(n) = \sum_{m=0}^{M-1} g(m)\, \delta(n - m\, n_s)$$

One can then show that the spectrum of $f(n)$ is given by

$$\hat{f}(\omega) = \hat{g}(\omega\, n_s)$$
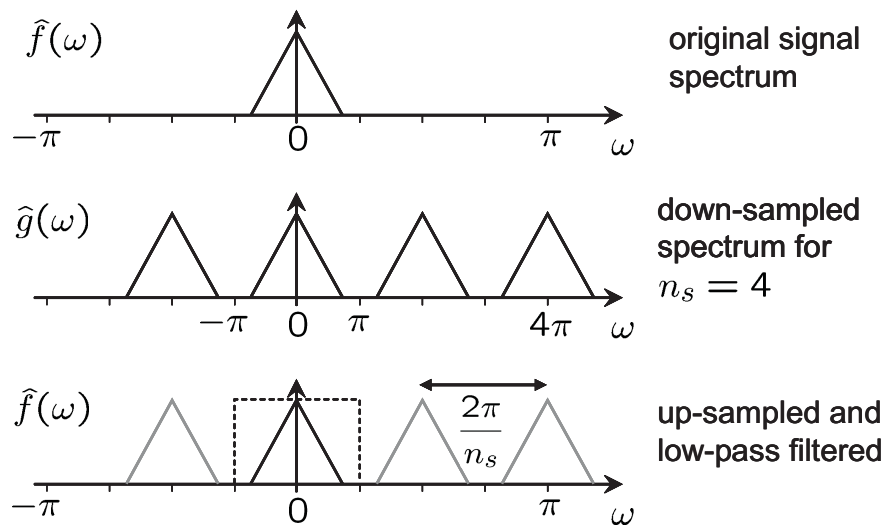
Nyquist Sampling Theorem

Theorem: Let $f(x)$ be a band-limited signal such that

$$\hat{f}(\omega) = 0 \quad \text{for} \quad |\omega| > \omega_0$$

for some ω_0. Then $f(x)$ is uniquely determined by its samples $g(m) = f(m\,n_s)$ when

$$\frac{2\pi}{n_s} > 2\omega_0 \quad \text{or equivalently} \quad n_s < \frac{\lambda_0}{2}$$

where $\lambda_0 = 2\pi/\omega_0$. In words, the distance between samples must be smaller than half a wavelength of the highest frequency in the signal.


$\hat{f}(\omega)$ — original signal spectrum

$\hat{g}(\omega)$ — down-sampled spectrum for $n_s = 4$

$\hat{f}(\omega)$ — up-sampled and low-pass filtered

Here the replicas can be isolated by an ideal low-pass filter (the dotted pass-band), so the original signal can be perfectly reconstructed.

Corollary: Let $f(x)$ be a single-sided band-pass signal with band-width $2\omega_0$. Then $f(x)$ is uniquely determined if sampled at a rate such that $n_s < \frac{\lambda_0}{2}$.

Aliasing

Aliasing occurs when replicas overlap:
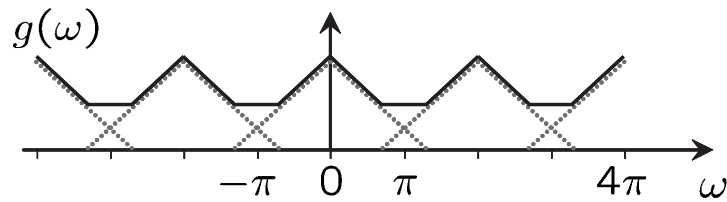
Consider a perspective image of an infinite checkerboard. The signal is dominated by high frequencies in the image near the horizon. Properly designed cameras blur the signal before sampling, using

- the point spread function due to diffraction,
- imperfect focus,
- averaging the signal over each CCD element.

These operations attenuate high frequency components in the signal. Without this (physical) preprocessing, the sampled image can be severely aliased (corrupted):
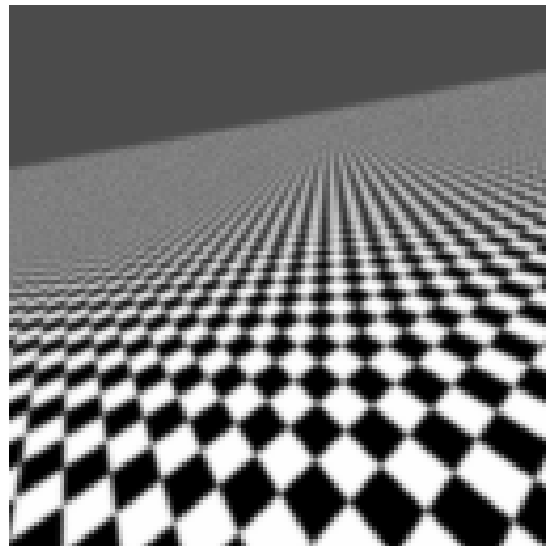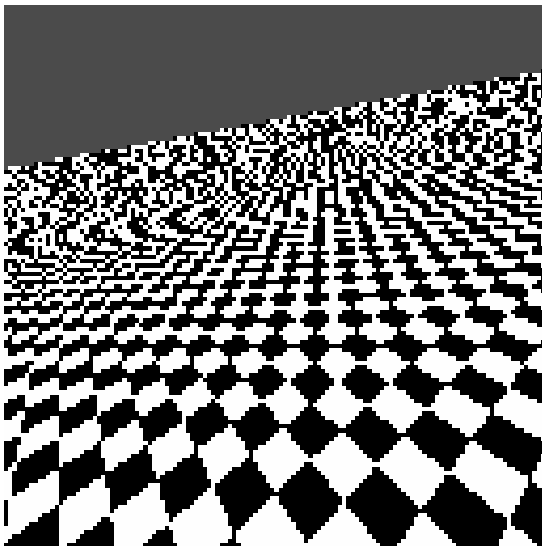
Sampling Continuous Signals

Consider the simple 1D signal $s(t)$, for $t \in [0, 1]$.

Sample Points:

$$t_j = j/N, \quad \text{for } j \in \{0, \ldots, N-1\}.$$

Discrete Sampling:

$$\begin{aligned}
\vec{s} &\equiv (s(t_0), s(t_1), \ldots, s(t_{N-1}))^T = D_N s(t) \\
\vec{b}_n &\equiv (b(t_0), b(t_1), \ldots, b(t_{N-1}))^T = D_N b(t).
\end{aligned} \tag{8}$$

Now, from the Fourier series for $s(t)$, we find

$$\begin{aligned}
\vec{s} &= D_N s(t) = D_N \sum_{n=-\infty}^{\infty} a_n b_n(t) = \sum_{n=-\infty}^{\infty} a_n D_N b_n(t) \\
&= \sum_{n=-\infty}^{\infty} a_n \vec{b}_n.
\end{aligned} \tag{9}$$

Aliasing:

- Note $\vec{b}_n \in \mathcal{C}^N$, so at most N of these vectors can be linearly independent.

- Any maximal linearly independent set $\{\vec{b}_n\}$ can serve as a basis.

- Often choose $n \in (-N/2, N/2]$ for the basis.

- The other sampled basis functions, say $\vec{b}_m$ for $|m| > \lfloor N/2 \rfloor$, must be linear combinations of the chosen basis $\{\vec{b}_n\}_{n=-\lfloor (N-1)/2 \rfloor}^{\lfloor N/2 \rfloor}$. These vectors $\vec{b}_m$ are said to be *aliased*.

Details of Aliasing

Fourier Analysis of Sampling:

$$\vec{b}_n = D_N b_n(t) == (b_n(t_0), b_n(t_1), \ldots, b_n(t_{N-1}))^T.$$
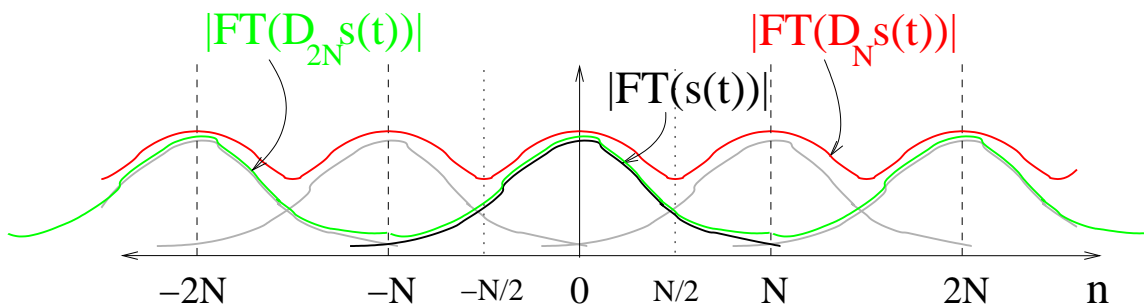
Note that

$$
\begin{aligned}
b_n(t_j) &= e^{i2\pi n t_j} = e^{i2\pi n j/N} = e^{i(2\pi n/N)j} \\
&= e^{i\omega_n j} = (z_n)^j, \quad \text{for } z_n = e^{i\omega_n}.
\end{aligned}
\tag{10}
$$

Here $\omega_n = 2\pi n/N$ is called the *frequency* of $b_n(t)$. Also,

$$
\begin{aligned}
z_n &= z_n e^{i2\pi k}, && (\text{since } e^{i2\pi} = 1) \\
&= e^{i2\pi(n+kN)/N} = z_{n+kN}.
\end{aligned}
\tag{11}
$$

Therefore, for all integers n and k, $\vec{b}_n = \vec{b}_{n+kN}$. The Fourier series becomes

$$\vec{s} = \sum_{n=-\lfloor (N-1)/2 \rfloor}^{\lfloor N/2 \rfloor} \left[\sum_{k=-\infty}^{\infty} a_{n+kN} \right] \vec{b}_n = \sum_{n=-\lfloor (N-1)/2 \rfloor}^{\lfloor N/2 \rfloor} \hat{a}_n \vec{b}_n.$$

Aliasing *destroys* information about the original continuous signal whenever there are significant nonzero coefficients, a_n, for any frequency larger than the *Nyquist frequency* $\omega_{N/2} = \pi$ (i.e. after sampling we only have access to $\hat{a}_n$ not a_n).

Dimensionality

A guiding principal throughout signal transforms, sampling, and aliasing is the underlying dimension of the signal, that is, the number of linearly independent degrees of freedom (dof). This helps clarify many issues that might otherwise appear mysterious.

- Real-valued signals with N samples have N dof. We need a basis of dimension N to represent them uniquely.

- Why did the DFT of a signal of length N use N sinusoids? Because N sinusoids are linearly independent, providing a minimal spanning set for signals of length N. We need no more than N.

- But wait: Fourier coefficients are complex-valued, and therefore have $2N$ dofs. This matches the dof needed for complex signals of length N but not real-valued signals. For real signals the Fourier spectra are symmetric, so we keep half of the coefficients.

- When we down-sample a signal by a factor of two we are moving to a basis with $N/2$ dimensions. The Nyquist theorem says that the original signal should lie in an $N/2$ dimensional space before you down-sample. Otherwise information is corrupted (i.e. signal structure in multiple dimensions of the original N-D space appear the same in the $N/2$-D space).

- The Nyquist theorem is not primarily about highest frequencies and bandwidth. The issue is really one of having a model for the signal; that is, how many non-zero frequency components are in the signal (i.e., the dofs), and which frequencies are they.

Further Readings

Texts on Image Processing and Computer Vision

Castleman, K.R., **Digital Image Processing**, Prentice Hall, 1995

Gonzalez, R.C. and Wintz, P., **Digital Image Processing 2nd ed.**, Addison-Wesley, 1987

Rosenfeld, A. and Kak, A., **Digital Picture Processing 2nd ed.**, Academic Press, 1982

Wolberg, G., **Digital Image Warping**, IEEE Computer Society Press, 1990

Wandell, B.A., **Foundations of Vision**, Sinauer Press, 1995