Enhancing signal gradient for edge detection:
A concern with computing signal intensity gradients for edge detection is that random noise can present as a strong gradient. For this reason, calculation of gradients is often accompanied by Gaussian filtering to remove high-frequency noise. Most people first prepare their signal by convolving it with a Gaussian filter and then proceed with the gradient calculation. Recall that a simple gradient calculation can be achieved by creating a gradient kernel and convolving this with the image. Thus both filtering and taking the gradient can be achieved as independent convolution steps. However, convolution is associative as well as commutative, thus the result of filtering and computing the gradient is the same whether the gradient operation is performed before or after the Gaussian filter. Moreover, one can compute the gradient of the Gaussian kernel and then convolve the result with the original signal and still get the same result.

Commutative and Associative laws for convolution.	Gaussian Filter: F Gradient Kernel: G
$R = G*(F*S) = F*(G*S) = (F*G)*S$	Original, unfiltered signal: S Resultant, filtered signal: R

Utility of signal gradient information for feature segmentation: Snakes
Segmentation is the process of identifying and outlining the border of an image feature object of interest. Edge detection is a primary means of identifying feature borders, however edge information can be problematic when noise is present and/or gaps in the edge information is missing.

The classic problems in automatic segmentation of medical images:

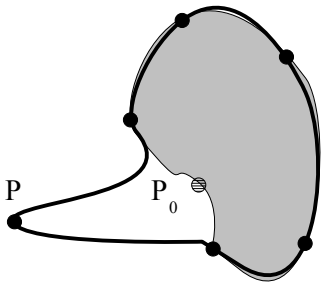
1. variable shape: the shape of interest can change over time and vary from person to person, so a rigid shape model (ala template matching) won't work
2. image noise: because of image noise the edges of the shape of interest can be difficult to pick out, especially when computing the edge using a gradient, so need to smooth over jagged edges
3. poor image quality: blurring and/or lack of image contrast can leave gaps in the edge, so need to be able to interpolate across gaps
4. curvilinear shapes: organs tend to be curvilinear or at least smoothly varying so need a base model that is curvilinear.

Snakes:
The snake concept is attributed to Kass, Witkin and Terzopoulos who presented it as a conference proceeding: "*Snakes: Active contour models*", International Journal of Computer Vision, 1987. Snakes are a series of points (*snake points*) connected by splines (or similar smooth interpolation) that has associated with it an internal force/energy that constrains the snake to behave in a smooth and 'reasonable' manner. The snake is also acted upon by external forces/energies provided by the image and/or user that encourage it to move toward image features of interest. We start by performing an edge-enhancement of the image via Gaussian filtering and application of a gradient operator. We then associate a large positive energy with bright pixels in the edge image. The high energy then will attract the snake to strong edges.

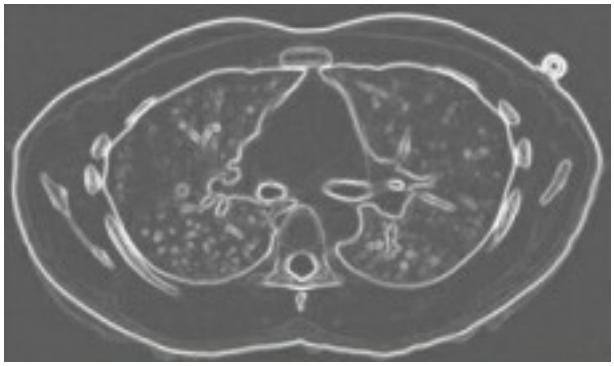
The internal energy term:
The usual way to impose constraints on the snake shape and motion is to impose a penalty on excessive bending between adjacent points along the curve, and impose a penalty on excessive stretching between points. These prevent the snake from forming kinks and from having one point shoot out very far away from the rest in search of a strong but distant image feature. It is reasonable to penalize kinks and excessing stretching if we can assume that the image object of interest is smoothly varying. Figure 1 depicts a point 'p' shifted from its original Po location. This is unfavorable

because it results in excessive bending and excessive stretching from its neighbors.

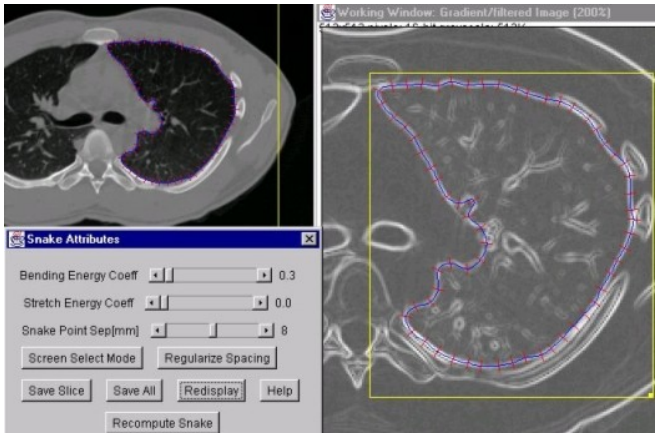
The penalty term is represented mathematically by defining an internal energy as: $E_{\text{int}} = \alpha \left|\left(\frac{dv}{ds}\right) - d_o\right|^2 + \beta \left|\frac{dv^2}{ds^2}\right|$



The term d**v**/ds represents stretching/shortening of the snake (change in distance along its length), thus the penalty term α is a measure of the **elasticity** of the snake, ala a rubber band. If $\boldsymbol{d_o}$ is set to zero, the default round shape will try to become as small as possible, so in the absence of strong image features the snake will converge down to an infinitesimally small point. $\boldsymbol{d_o}$ establishes a baseline distance between points. Squaring the term achieves two things: it increases the penalty substantially when you move farther away from equilibrium; and it absolves you from having to worry about a negative energy when compressing instead of stretching. Alpha can be made to vary along the snake, $\alpha(s)$, but is usually made constant. The term $d^2\mathbf{v}/ds^2$ represents bending of the snake, thus the penalty term β is a measure of the **stiffness** of the snake. β can also be made to vary along the snake, $\beta(s)$, but is usually made constant. If β is set to zero, then the snake is allowed to develop a kink or corner (a discontinuity in the first derivative). One way to intentionally create a kink is to set β = 0 for only the snake point of interest. Another way to create a kink is to define two snake points right on top of each other. Since the segment length between the two is zero, the direction is essentially undefined, thus the change in direction is effectively null. The default shape of a snake is a circle -- to minimize bending of a closed-loop. I



A



B

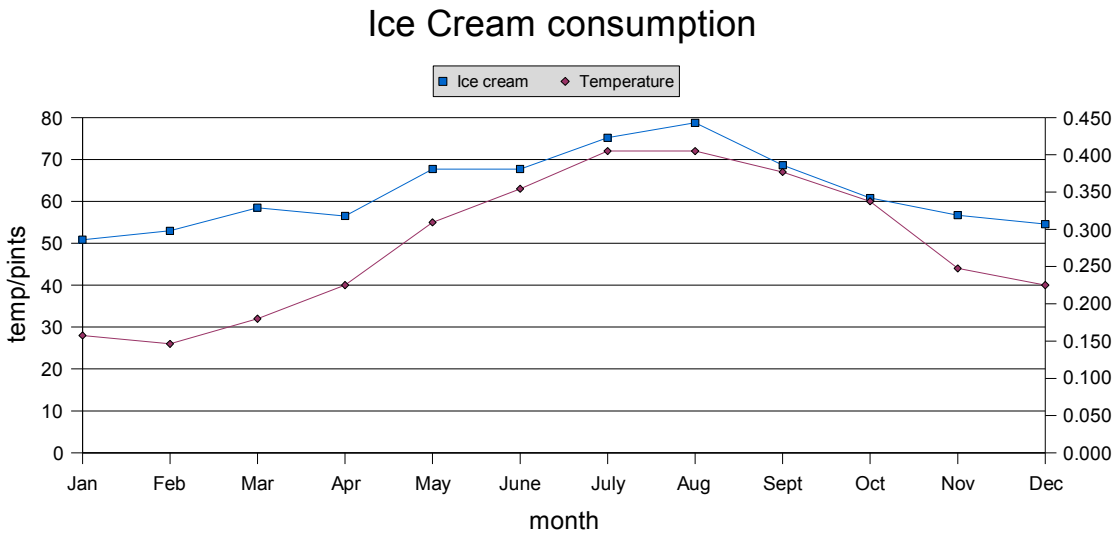
Figure 2. [A] Lung CT image slice after Gaussian filtering and application of a gradient. [B] Snake manipulation interface. Each red line segment represents a snake point and its local direction of edge search. The blue line is the spline-fitted contour of the lung superposed over the edge image. Bright pixels in the edge image attract the snake. Bending and stretching energies can be adjusted on the fly. The appearance of the snake on the original CT image is shown in the upper left of [B].

Snake optimization:

It is the battle between the internal forces that try to maintain the snake in a reasonable shape, and the external forces that try to pull it in every which way to conform to local image features, that then dictate the motion and eventual equilibrium state of the snake. This can be interpreted as a balance of forces where the stronger force will act to move the snake points toward the equilibrium state, or you can interpret it as a desire to minimize the total energy of the system. In this latter view, the image features are represented as energy wells that the snake points are trying to fall into, while excessive bending and stretching add to a large positive the internal energy component

Statistics: correlation between two sets of observations

example: Ice cream consumption in pints per person per month versus the average temperature per month for the year 1951 in Aberdeen, Scotland.



	Jan	*Feb*	*Mar*	*Apr*	*May*	*Jun*	*July*	*Aug*	*Sept*	*Oct*	*Nov*	*Dec*
Ice cream	0.286	0.298	0.329	0.318	0.381	0.381	0.423	0.443	0.386	0.342	0.319	0.307
Temp	28	26	32	40	55	63	72	72	67	60	44	40

A simple (and weak) statistical measure of a correlation between two sets of samples, X and Y, is their covariance, S_{xy}:

$$\bar{x} = \frac{1}{n}\sum x_i \;, \qquad \bar{y} = \frac{1}{n}\sum y_i$$

$$S_{xx}^2 = \frac{1}{n-1}\sum (x-\bar{x})^2 \;, \qquad S_{yy}^2 = \frac{1}{n-1}\sum (y-\bar{y})^2$$

$$S_{xy} = \frac{1}{n-1}\sum (x-\bar{x})\cdot(y-\bar{y})$$

S_{xx}^2 is the standard deviation of the values in set X. S_{xx} is the variance of set X. S_{xy} is a sample by sample multiplication of each sample in X with its corresponding time- or space matched sample in Y after each sample has been offset by the mean for each set. The covariance between ice cream consumption and temperature is 0.73. Except for the correction for sample means and the 1/(n-1) normalization out front the covariance calculation is very similar to the convolution equation. Indeed, there is a equation for ***simple image correlation*** which is identical to the convolution equation except that the kernel indices do not have a negative sign in front of them.

$$\text{Simple Correlation (1D)}: \quad \boldsymbol{S * K} = \sum_{s=0}^{N_S} \sum_{t=-k_w/2}^{k_w/2} S[s+t]\cdot K[\frac{k_w}{2}+t]$$

In simple correlation, *S* is the reference signal and the kernel, *K*, is called the test signal. Each sample in the convolved signal represents the simple correlation between the kernel and the portion of the signal that is aligned with the kernel and centered at the sample of interest. To better see what is

happening, consider Figure 2 below which diagrams a signal with a central hump convolved with a similar looking hump as the kernel.

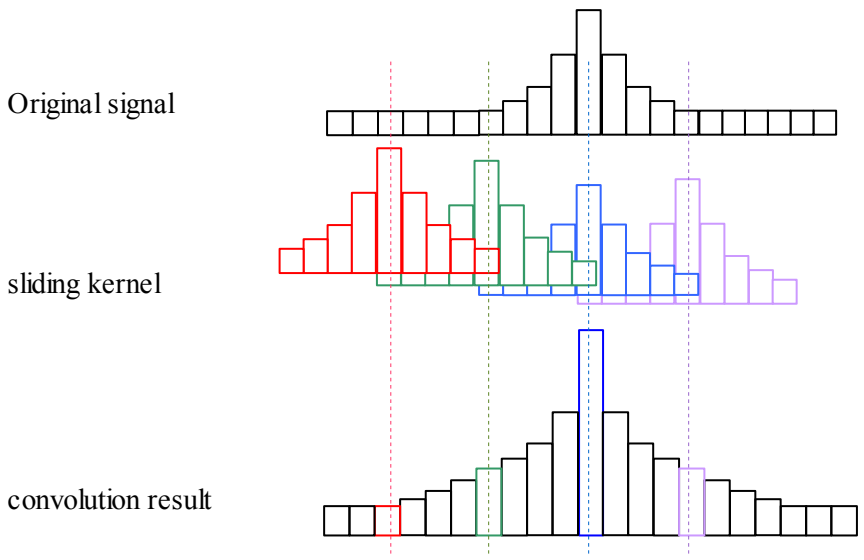
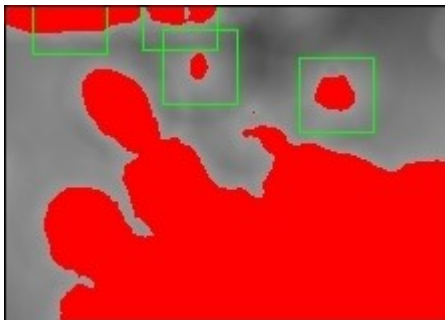
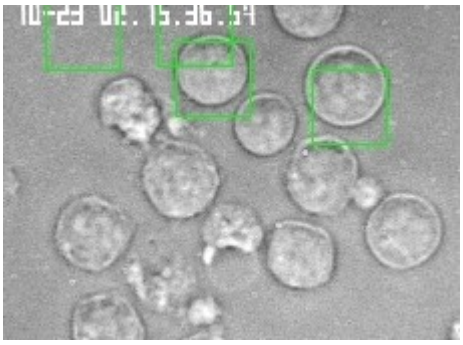


Figure 2: Diagram of simple convolution/correlation of a signal with a kernel consisting of a bump feature where the signal contains a similar feature. The original signal (top, black samples) is multiplied sample by sample with the kernel. Highlighted in red, green, blue and purple are the locations of the kernel as it slides along, and the resulting simple correlation values. When the kernel is aligned with a similar-shaped feature in the signal, a large correlation is found.

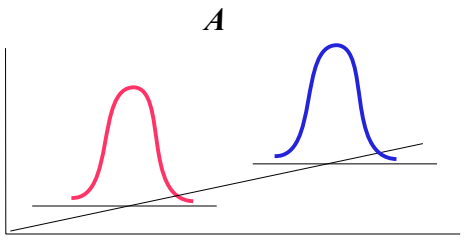


A

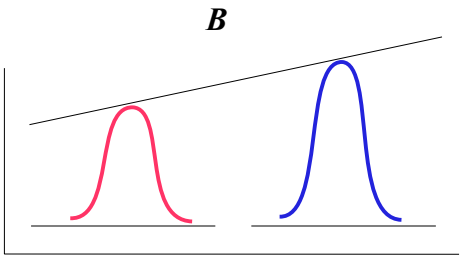


B

Figure 3: The correlation map [A] resulting from convolution of an image of red blood cells on a microscope [B] convolved with a disk kernel of similar size to the cells. Local maxima [green boxes] mark each cell, but uneven background lighting leads to large correlation values over the bottom right of the correlation image.



A



B

Figure 4: Increased simple correlation values with increased background signal intensity [A] and with increased object contrast [B]. Both of these effects occur in Figure 3B due to uneven lighting of the microscope slide.

A problem with convolution for finding correlations is that fluctuations in background signal intensity or in contrast will contribute strongly to the sample correlations, as observed in figure 3 and diagrammed in Figure 4. The statistical covariance index corrects for uneven background signal intensities

between the reference signal and the test signal but subtracting the mean signal intensity from each sample. Correction for background signal intensity greatly improves the detection of signal features. This is diagrammed in Figure 5.

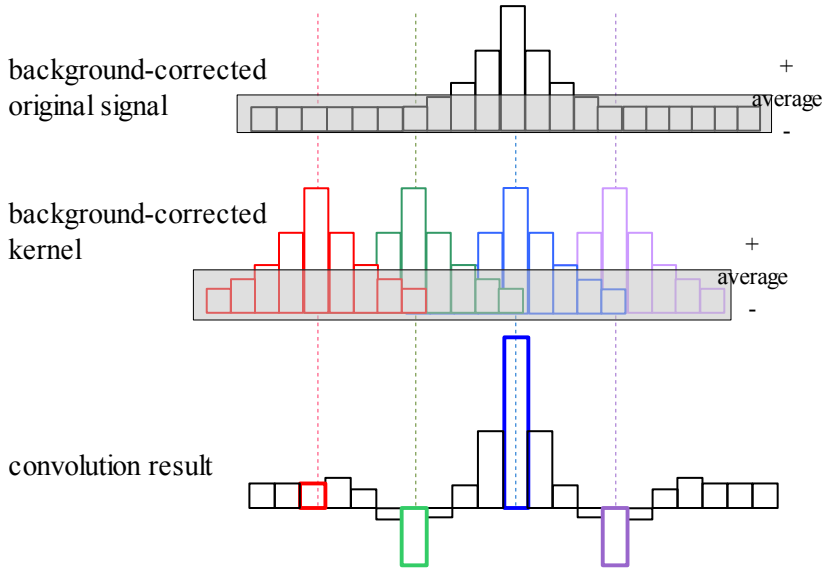


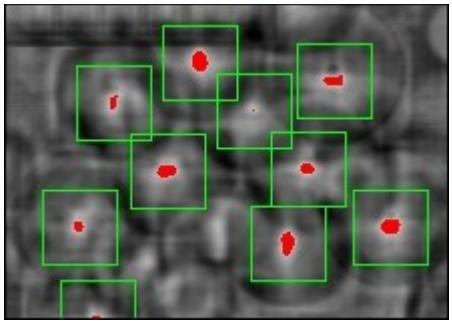
Figure 5. Improved feature detection using background-correction of original signal and of kernel. Red: BG-corrected original signal is all negative-valued over red kernel. Kernel is half positive and half negative, giving near-zero result. Green: Where kernel samples are large positive, original signal is negative, giving a negative contribution. Where kernel is negative, signal is mostly positive, giving a negative contribution. Overall green result is large negative. Blue: Kernel and signal are perfectly matched. Negative samples matched with negative samples, positive samples matched with positive samples, giving large positive result.

However, statisticians do not rely on the covariance but instead normalize this by the sample variances to achieve the **normalized cross-correlation coefficient** (NCCC), otherwise known as the Pearson Product-moment correlation coefficient and commonly called simply the **r-value.** R-values range from -1 to +1. R-value magnitudes greater than 0.7 are generally considered indicative of a strong correlation, and magnitudes less than 0.3 are considered indicative of a very weak (i.e. non-) correlation. Moreover, from the R-value one can compute a *t-statistic* (aka t-test) and from the t-statistic one can use a look-up table to find the *p-value* = the probability that observed similarity between the 2 sets of observations occurred by chance.

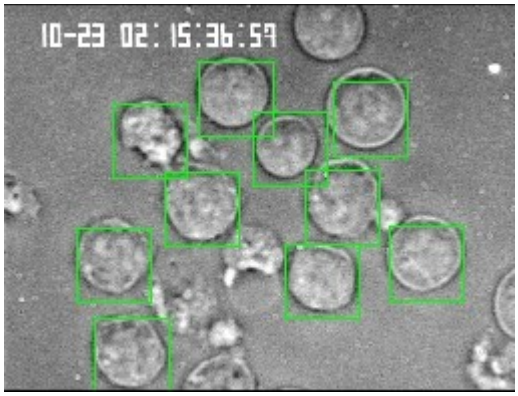
$$r = \frac{S_{xy}}{\sqrt{S_{xx}^2 \cdot S_{yy}^2}}, \quad t_{statistic} = \frac{r}{SE_r}, \quad SE_r = \sqrt{\frac{1-r^2}{n-2}}$$

SE_R is the standard error in the r-value and is the means to convert from the statistics from a small random sample to those of the population. The r-value corrects not only for background intensity variations but because of normalization with the individual variances it corrects also for variations in contrast such as occurred with the uneven microscope lighting. The normalized cross-correlation value therefore matches image features by general size and shape, independent of background intensity and contrast level. Because of these features of the NCCC, it is not necessary to spend a lot of effort in the microscope image acquisition to correct for these, however NCCC will still be affected greatly by the signal SNR and CNR.

Using the statistical normalized cross-correlation coefficient, we can use our kernel to select objects in the signal that closely match the kernel's shape, size and intensity features. We refer to this selection kernel as the **template**, and the process of using a template to select out objects from the signal is called **template matching**. This approach is the method of choice for many signal detection applications, such as detecting when the QRS complex of an ECG signal has passed a sensor, as part of a heart rate monitor and/or implanted defibrillator. It is also the method my labs uses to detect metastatic tumors in the lung and brain.

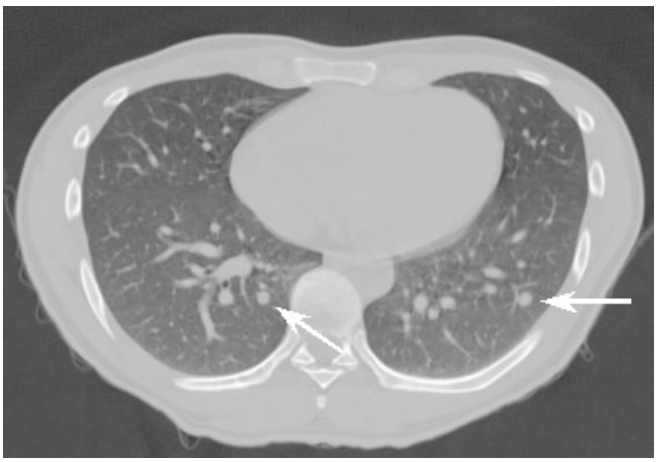


A



B

Figure 6: [A] R-value map of cell microscope slide image compared with disk kernel of similar size to cells. An r-value threshold of 0.4 was used. [B] Detected cells are marked with green boxes.



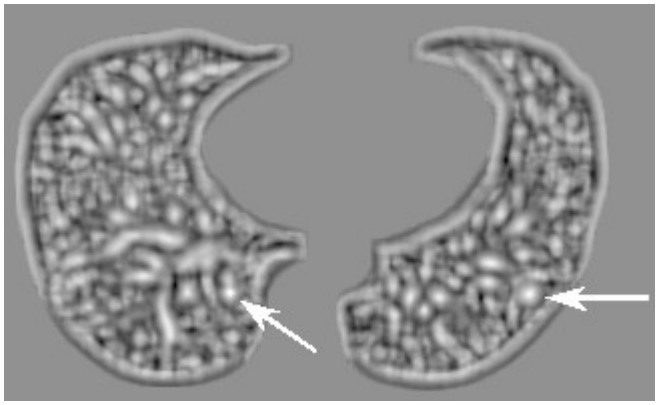


Figure 7. Normalized Cross-correlation in 3D of lung CT series with a spherical kernel of 10 mm diameter. [Left] One slice through the lung CT series, showing two metastatic lung tumors indicted by the two arrows. [Right] NCCC map through the same slice. The two tumors have the highest r-values over the entire slice.

group	*# subjects*	*# slices*	**slice thickness [mm]**	*# nodules*	*nodule diameters [mm]*	*% nodules detected*	*# false positives per nodules*	*# false positives per slice*
O'Dell/ Wang	7	479	3	40	4-20	100	0.30	0.025
Lee	20	557	10	98	5-30	72	6.28	1.10
Zhao	1	~30	7	4	~10	75	2.25	0.3
Brown 2001	17	*NA*	5-10	36	5-30	86	5.19	*NA*
Brown 2003	15	~300	1	77	1-10	78	2.81	0.72

Table 1. Detection rate and false positive rate for the author's method in comparison with recently published results from the literature (*NA* = data not available). All results shown are for human lung CT datasets.

Undersampling and Aliasing

The Nyquist theorum states that in order to detect the presence of signal at a frequency of interest (ν_o) you must sample the signal with at least twice that frequency and this is called the Nyquist sampling frequency for this signal ($\nu_n = 2\nu_o$, or $\nu_o = \nu_n/2$). We now want to consider what happens when there are components of the signal that have frequency greater than $\nu_n/2$. We will see that even though we have sufficient sampling to detect the the signal components at ν_o, the presence of signal components at higher frequencies leads to errors in our measurement of the magnitude of the contribution of the ν_o component, and every other component. Consider Figure 8 below. The top line depicts a signal at frequency ν_o that is sampled exactly at $\nu_n = 2\nu_o$ and where we got lucky and our samples occur at the peaks and troughs of the sinusoid. The middle line shows a signal with frequency equal to that of the Nyquist rate and the resulting samples all have the same value. The reconstructed signal is constant valued, i.e. at a frequency of zero.

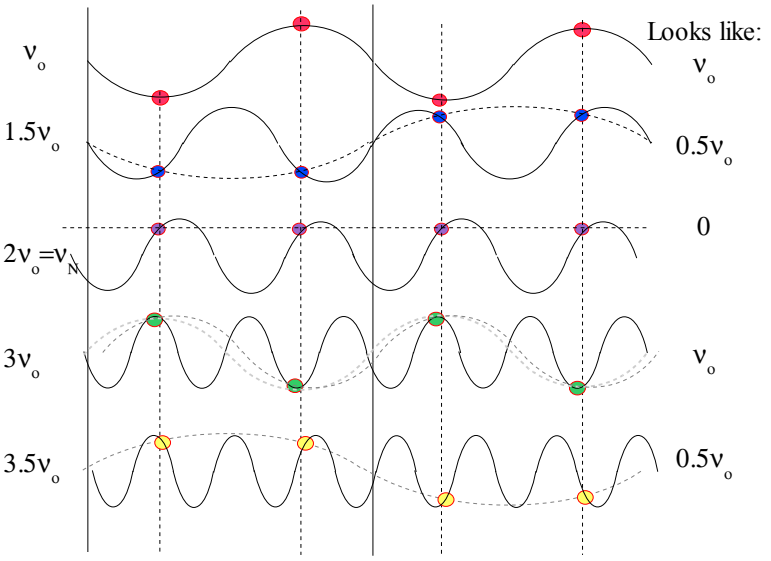


Figure 8. Undersampling examples. In each case where the frequency of the signal is > ν_o, the signal reconstructed from the samples is at a frequency <ν_o.

The second line shows a signal at $1.5\,\nu_o$ and here the samples generate a signal that appears to be at a frequency of $0.5\nu_o$. In each case where the frequency of the signal is > ν_o, the signal reconstructed from the samples is at a frequency < ν_o. Indeed it turns out that a signal occurring at a frequency of ($\nu_o + \varepsilon$) is reconstructed instead at a frequency of ($\nu_o - \varepsilon$), where ν_o is one-half the sampling rate. This phenomenon is called **Aliasing** and represents wrap-around in the Fourier (i.e. frequency) domain.

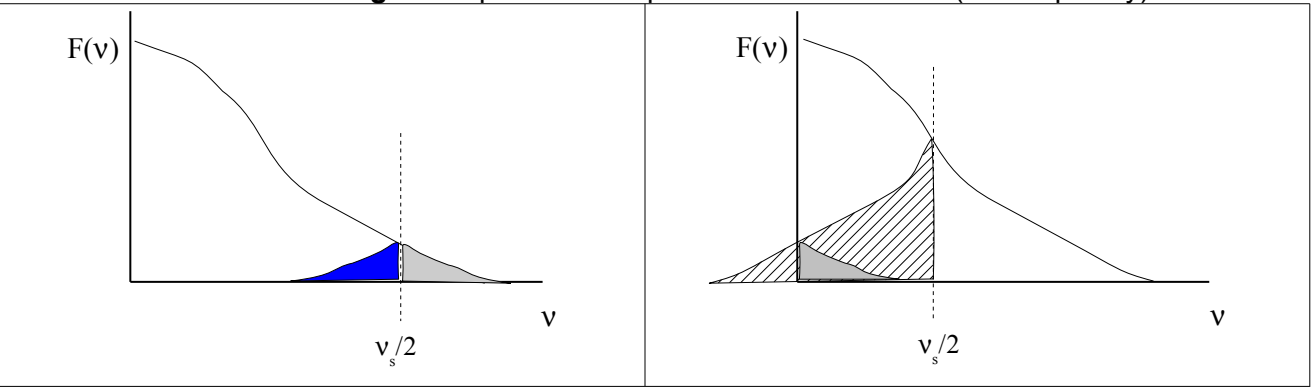


Figure 9: Wrap around of the frequency spectrum due to undersampling. Wrap-around occurs at ½ the sampling frequency (ν_s) [left] and also at 0 if the undersampling is severe enough [right].

Components of the signal at frequencies higher than ½ the sampling frequency (½ ν_s) are reflected back to signal frequency below ½ ν_s. The mathematical basis of this phenomenon stems from the observation that $\sin(\theta) = \sin(\theta + 2\pi m)$ where m is any positive or negative integer. For a sampling interval of $t_s = 1/2\pi\nu_s$, the above equation leads to the result for the value at sample n, x(n):

$$x(n) = \sin(2\pi\nu_o n t_s) = \sin(2\pi(\nu_o + m\nu_s) n t_s)$$

Another way to visualize aliasing is to repeat the Fourier spectrum every interval of ν_s along the frequency axis in the spectrum plot:

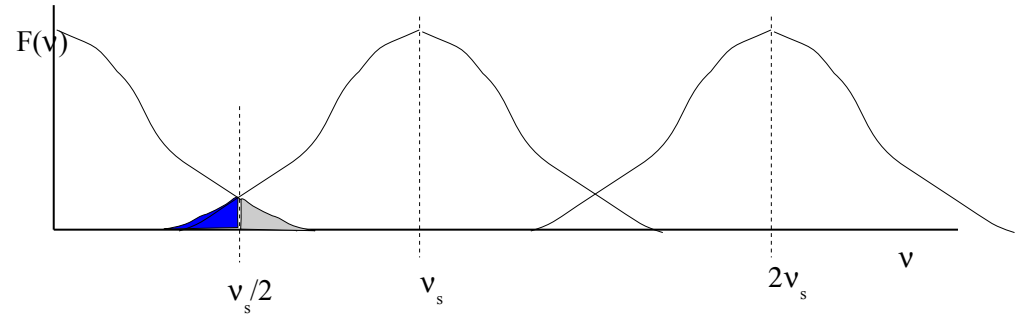


Figure 10: Alternative representation of aliasing in the frequency domain as a repeated +/- spectrum at every interval of the sampling frequency, ν_s. Aliasing occurs where the tails of adjacent spectra overlap across $\nu_s/2$.

From the samples themselves we can no longer unambiguously determine which frequency components the sample values came from. The theorum is:
When sampling at a rate of ν_s samples/s, if *k* is any positive or negative integer, we cannot distinguish between the sampled values of a sine wave of ν_o Hz and a sine wave of (ν_o+kν_s) Hz.
As a check, the signal at ($2\nu_o = \nu_o + \nu_o$) appears to come from ($\nu_o - \nu_o = 0$) zero frequency, as demonstrated in the middle line of Figure 8. An interactive demo of this phenomenon is available at: http://www2.egr.uh.edu/~glover/applets/Sampling/Sampling.html

Correcting Aliasing Artifacts:
Oversampling: The easiest way to eliminate aliasing is with oversampling. If frequency spectrum for a signal has negligible values above a certain maximum frequency (i.e. the signal is frequency limited) then you simply need to sample at a rate that is greater than twice that frequency maximum. This ensures that the tails of the spectrum (Figure 11) do not overlap. Thus you need to consider not only the frequency components of interest but all the frequency components that are in your signal. Modern commercially-available hardware for digital sampling is very cheap and very fast such that achieving a sufficient sampling rate is not a problem. The disadvantage is that you accumulate tons of data that is taxing on computer memory and requires more time to process.

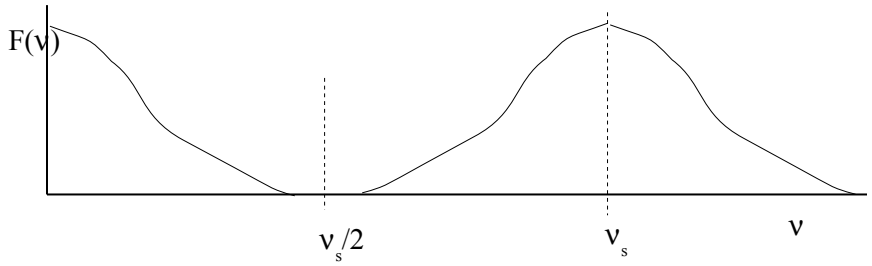
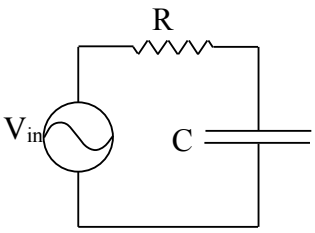
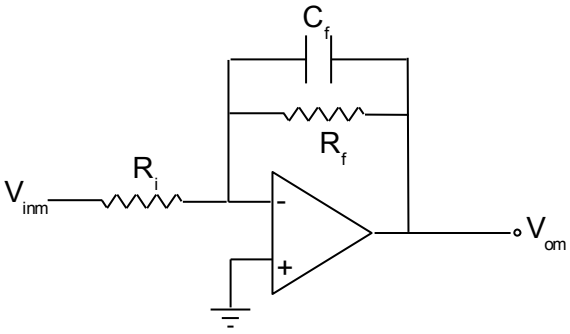


Figure 11. Oversampling to ensure that the frequency spectrum tails do not overlap.

Analog filtering prior to sampling:
Unfortunately in many cases the high frequency components are due to noise and noise is both unpredictable in its frequency content and is often present at very high frequencies. (Noise that spans all frequencies equally is called White Noise.) Thus even if the signal of interest is band limited the addition of noise makes it less so. Thus a common solution to aliasing of high frequency noise is to perform filtering of the analog signal prior to sampling. This is often done using a low-pass circuit such as a simple resistor-capacitor circuit. The frequency content of the signal is cut-off at approximately the corner frequency of the circuit (ν_c = 1/sqrt(RC)). Digital sampling without aliasing artifact can proceed when applied with a sampling frequency that is comfortably above $2\nu_c$.



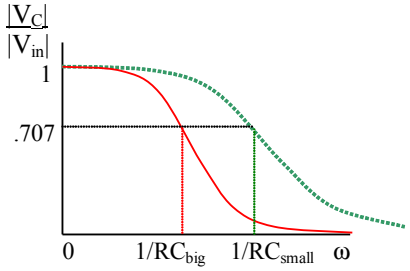


$$H(\omega) = \frac{V_c}{V_{in}} = \frac{Z_c}{Z_c + Z_R} = \frac{1}{1 + j\omega RC}$$

$$|H(\omega)| = \frac{1}{\sqrt{1-(\omega^2/\omega_c^2)}} \quad \text{where} \quad \omega_c = \frac{1}{\sqrt{RC}}$$

$$H(\omega) = \frac{V_o}{V_{in}} = \frac{-Z_f}{Z_i} = -\frac{R_f // Z_c}{R_i}$$

$$|H(\omega)| = \frac{R_f}{R_i} \frac{1}{\sqrt{1-(\omega^2/\omega_c^2)}}$$



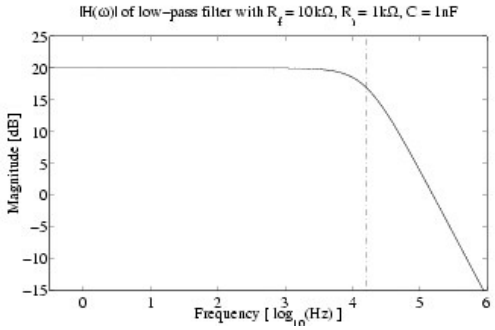


Figure 12: Passive [left] and active [right] low-pass analog filters.