Matched Filters

1 Basics

Let's say you have a long stretch of noisy data. There might (or might not) be a signal in there. You'd like to be able to find it. We'll assume you know the shape of the expected signal, but its amplitude, or when it might have happened.

The simplest way to go about searching is to do a least-squares fit to the data using the template, for all possible times the signal could have happened. To maintain consistency, we'll let the template profile be $\mathrm{A}(t-\tau)$ for a signal centered at time τ, the amplitude of that signal be m, the data d and the noise N. In that case, we have

$$\chi^2 = (d - \mathrm{A}m)^T \mathrm{N}^{-1} (d - \mathrm{A}m)$$

The maximum likelihood value for m is then:

$$\mathrm{A}^T \mathrm{N}^{-1} \mathrm{A} m = \mathrm{A}^T \mathrm{N}^{-1} d$$

For a single template (*i.e.* A only has one column), we can write the best fit amplitude for the signal occurint at time τ as:

$$m(\tau) = \left(\mathrm{A}(t-\tau)^T \mathrm{N}^{-1} d\right) / \left(\mathrm{A}(t-\tau)^T \mathrm{N}^{-1} A(t-\tau)\right)$$

2 Stationary Noise, one (possibly more) Template

The simplest case is when the noise is stationary - *i.e.* $n_{i,j} = f(i-j)$. In this case, let $t' = t - \tau$ and the denominator becomes

$$\mathrm{A}(t')^T \mathrm{N}(i+\tau, j+\tau)^{-1} \mathrm{A}(t')$$

Since the noise is stationary, $\mathrm{N}(i+\tau, j+\tau) = \mathrm{N}(i,j) = \mathrm{N}$. The denominator is therefore constant, and we can ignore it if we're just looking to find *when* a signal happend. It is a very useful number to have, though, so you might want to calculate it anyways. We'll call it H since it is the Hessian (curvature) of our problem.

Once again taking advantage of the stationarity of the noise, we can write the right-hand side in two ways:

$$f \equiv \left(\mathrm{N}^{-1} \mathrm{A}(t-\tau)\right)^T d(t) = \mathrm{A}(t-\tau)^T \left(\mathrm{N}^{-1} d(t)\right)$$

These ought to be identical up to round-off error, so you are free to calculate either (or both, as a bug check). Writing out the entries in the dot product explicitly, we have:

$$f_\tau = \sum (\mathrm{N}^{-1}\mathrm{A})_{i+\tau} d_i = \sum A_{i+\tau} (\mathrm{N}^{-1} d)_i$$

This is just the *cross-correlation* of two vectors, which we can calculate efficiently with a DFT. Since the noise has to be diagonal in Fourier space (see the note on stationary noise if this is unfamiliar to you), we can write this quite neatly. Let tildes denote DFTs and * the complex conjugate, then we have:

$$\tilde{f} = \tilde{\mathrm{A}}^* \tilde{\mathrm{N}}^{-1} \tilde{d}$$

Three forward DFTs and one inverse DFT, and we can fit an arbitrary function to arbitrary data with arbitrary (but stationary) noise, which is rather remarkable. These are called *matched filters*. Peaks in f tell you where to look for your signal.

While the output of the matched filter is not normalized, we can go back to the χ^2 equations and get properly normalized quantities of interest. The maximum likelihood amplitude m is just:

$$m = \mathrm{H}^{-1} f$$

The value for χ^2 is

$$(d - \mathrm{A}m)^T \mathrm{N}^{-1} (d - \mathrm{A}m) = d^T \mathrm{N}^{-1} d - 2m^T \mathrm{A}^T \mathrm{N}^{-1} d + m^T \mathrm{A}^T \mathrm{N}^{-1} \mathrm{A} m$$

Our original value of χ^2 was just $d^T \mathrm{N}^{-1} d$, so the *improvement* in χ^2 is

$$-2m^T \mathrm{A}^T \mathrm{N}^{-1} d + m^T \mathrm{A}^T \mathrm{N}^{-1} \mathrm{A} m$$

We can simpify this by expressing in terms of H and f (and noting that the Hessian is symmetric):

$$-2f^T \mathrm{H}^{-1} f + f^T \mathrm{H}^{-1} \mathrm{H} \mathrm{H}^{-1} f = -f^T \mathrm{H}^{-1} f$$

If we're fitting a single amplitude, H is a scalar, and this reduces to $-f^2/\mathrm{H}$. Do note that up to this point, we never actually assumed that we had a single parameter, so you could just as easily use this framework to fit multiple components simultaneously. In that case, you have to Fourier transform each column of A, and f and m are 2-dimensional. The improvement in χ^2 at each value of τ is a scalar, so you can look at that to see if your matched filter found anything.

3 Case 2: Non-stationary, Uncorrelated Noise

Another common case is when the data points are independent, but do not all have the same noise. In this case, N will be diagonal, but not constant. We can fortunately still use the power of the matched filter. Writing down the Hessian, we have:

$$\mathrm{H} = \mathrm{A}(t - \tau)^T \mathrm{N}^{-1} \mathrm{A}(t - \tau) = \mathrm{A}^2 (t - \tau)/\mathrm{N}$$

where in the last step we have used the diagonality of N. This is now just the cross-correlation of A^2 with N^{-1}, which we can do with the usual Fourier techniques. The right-hand side becomes

$$f = \mathrm{A}(t - \tau)^T \, (d/\mathrm{N})$$

Again, we can do this with cross-correlations, only this time we have to apply N to the data and not to A. Unlike the simplest case, H is now a vector, but otherwise we proceed as before. The best-fit amplitude remains $\mathrm{H}^{-1}f$ and the improvement in χ^2 is f^2/H.

One particularly common case is when you are missing data. If you look at the equations, the only way N shows up is as N^{-1}, which acts like a set of weights applied to the data. If we set the noise in the missing data to an arbitrarily large number, then the weight goes to zero. So, if the noise is diagonal, we can still happily use our matched filters just by multiplying N^{-1} by the mask (1 where we have data, 0 where we don't).

4 Case 3: Correlated, Non-Stationary Noise

The most general case is when we have non-stationary noise with correlations between noise values. I don't know of any clever ways of applying matched filters in this case. Is this a disaster? Well, how much work does it take to do the brute-force fit at each point? If we have N^{-1}, then for every shift, the left-hand side is a vector dotted with a matrix times a vector, which takes $\sim n^2 + n$ operations. Since we have n shifts, the total work scales like n^3. For the right-hand side, we have to dot n copies of A against $\mathrm{N}^{-1}d$, which is a total work of n^2. So, our work, once we have N^{-1} has gone from $n\log(n)$ to n^3. This is a pretty big loss. However, how much work did it take to make N^{-1} in the first place? Well, the general correlated, non-stationary case usually means there aren't many tricks to inverting N, so that was an n^3 operation to start with. So even though there aren't any particularly clever tricks, the *additional* work we have to do to fit a model at every point is about the same as the work we had to do to fit a single model, so we perhaps aren't as badly off as it might seem.

Having said that, in practice there are often tricks one can play to get an approximate matched filter in faster than n^3 time. If you find yourself facing an extremely large and/or complicated dataset, you might want to try some of them.

First off, you can often in practice *solve* linear systems faster than n^3 - usually n^2 and often even $n\log(n)$ - though the prefactor can be dozens, hundreds, or even thousands. Basically, you can check if a guess b at $b = \mathrm{N}^{-1}d$ is correct by checking if $\mathrm{N}b = d$. Techniques such as conjugate-gradient tell you how to make increasingly clever guesses at b. If the noise is "approximately" stationary, then you can just look at $\mathrm{A}^T\left(\mathrm{N}^{-1}d\right)$, either by brute-force multipication of all the A's (done in n^2 time), or by cross-correlating your standard one-column version of A against $\mathrm{N}^{-1}d$ (done in $n\log(n)$ time).

In short, although an exact solution to the non-stationary, correlated problem may require n^3 operations, you might be able to play tricks to get a "good enough" solution in n^2 or even $n\log(n)$ time. It all depends on what your noise looks like, and how approximate a matched filter would still be useful for you.

5 Statistical Significance and Correlation

After you have run your matched filter, you have to decide if you saw anything. This can be slightly tricky. Sadly, we never know with absolute certainty that we have seen something. Instead, we can know that it is extremely unlikely (but not impossible) that we would have seen the observed data in the absence of a signal. Say someone gave us a number drawn from a standard Gaussian, and asked "is there signal?" We know that 95% of the time, we'll get something within 2σ of zero if there's no signal, 99.99994% of the time we get something within 5σ *etc.*. It's your job as the physicist to decide at what probability you think you have a signal, but once you've decided that, the threshold is well-determined. Now let's say you get 1,000 Gaussian numbers, and have to decide if you saw something. Let's say our threshold for significance is 95%. Now, if we want to say we saw something, we need to set our threshold so that 95% of the time, *all* of the numbers fall below our threshold. That means we set our threshold at a rarity of one in 20,000 instead of 1 in 20, which we can look up for a Gaussian to be just over 4σ.

How does this work for matched filters? You might think that you could just set the number of independent trials to be the length of the data, but that's not quite correct. Say our template is a symmetric Gaussian (so the DFT is strictly real) - then the output of the matched filter is:

$$\tilde{f} = \tilde{G}\tilde{\mathrm{N}}^{-1}\tilde{d}$$

If the noise is white, then the output of our matched filter is just the data convolved with a Gaussian. If our template is very wide, then χ^2 has to be very smooth, and so you would expect that the number of truly independent measurements is the number of data points divided by the width of the template. For the more general case, we want to know $\langle f_i f_{i+\tau}\rangle$. Where this falls by a factor of ~ 2 from the value at $\tau = 0$, then we know our matched filter outputs are not very correlated. This is just the cross-correlation of the matched filter with itself, so we can go back and look at what this would be. We need the IDFT of $\left\langle \tilde{f}^* \tilde{f} \right\rangle$ which is:

$$\left\langle \tilde{f}^* \tilde{f} \right\rangle = \left\langle \tilde{\mathrm{A}}\mathrm{N}^{-1}\tilde{d}\tilde{d}^\dagger \mathrm{N}^{-1}\tilde{\mathrm{A}}^\dagger \right\rangle = \tilde{\mathrm{A}}\mathrm{N}^{-1}\tilde{\mathrm{A}}^\dagger$$

where † denotes the conjugate transpose, and we have used the fact that $\left\langle dd^\dagger \right\rangle = \mathrm{N}$. You can look at the width of this to decide how many actually independent samples you have.

6 Mismatches and Filter Banks

The preceding discussion has assumed we know the shape of the signal we're looking for. This is often not quite true. For instance, if we're looking for galaxies in an image from a telescope, the galaxies could come in a range of

sizes. Since we don't know the object size, we could try out a range of trial sizes, and run the matched filter for each of those sizes. This is a *filter bank*, and is a common application of matched filters.

The question immediately arises, how closely spaced to our trial sizes need to be? More generally, how much do we lose if our template doesn't quite match the signal? Let's assume we have a true signal A (where we have assumed $m = 1$, and we use a wrong template A' to search for it. To quantify the penalty we pay, we will compare the expected SNR when we search for A when using A' as the template and the ideal case when using $\mathrm{A}' = \mathrm{A}$ as the template. For further simplicity, we'll set N to be the identity matrix.

With these assumptions, the noise in our matched filter output is $\mathrm{H}'^{-1/2}$ (see the least-squares note for a derivation). The fit amplitude is $\mathrm{H}'^{-1}\left(\mathrm{A}'^{T}\mathrm{A}\right)$, so the overall SNR is

$$\frac{\mathrm{A}'^{T}\mathrm{A}}{\sqrt{\mathrm{A}'^{T}\mathrm{A}'}}$$

In the ideal case, we set A' to A, which gives the ideal SNR $\sqrt{\mathrm{A}^{T}\mathrm{A}}$. That leaves us with the following simple relation for the fractional loss in SNR from using the wrong template:

$$\frac{\mathrm{A}'^{T}\mathrm{A}}{\sqrt{\mathrm{A}'^{T}\mathrm{A}'\ \mathrm{A}^{T}\mathrm{A}}}$$

Consider the case where the template/signal are Gaussians, with standard deviations σ' and σ. As long as the Gaussians are wide compared to the data spacing, the dot products turn into integrals. The numerator becomes $\exp\left(-\frac{x^2}{2}\left(\frac{1}{\sigma^2}+\frac{1}{\sigma'^2}\right)\right)$. The area under this is $\sqrt{2\pi\sigma_{eff}^2}$ where σ_{eff} is the effective standard deviation of the numerator, $\frac{1}{\sigma_{eff}^2} = \frac{1}{\sigma^2}+\frac{1}{\sigma'^2}$. The terms in the denominator pick up factors of 2 in their σ^2 from the multiplication so, after cancelling the $\sqrt{2\pi}$ factors, we have

$$\frac{\sqrt{\frac{2}{\sigma\sigma'}}}{\sqrt{\frac{1}{\sigma^2}+\frac{1}{\sigma'^2}}} = \sqrt{\frac{2\sigma\sigma'}{\sigma^2+\sigma'^2}}$$

If we let $\sigma' = \alpha\sigma$, then we get

$$\sqrt{\frac{2\alpha}{1+\alpha^2}}$$

It may not be obvious, but this is a surprisingly forgiving (at least to me) result. If you space your widths by factors of 2, then the furthest off you'll be in α is $\sqrt{2}$. Plug that in, and you see that the expected SNR you get is $(8/9)^{1/4}$, or 97.1% of the ideal. A Gaussian filter bank with factor-of-2 width spacings will find peaks with *any* with (at least within the range you searched) with at least 97% efficiency. This is one of the really powerful features of matched filters - they still tend to perform quite well even when you've given them pretty poor inputs. Of course, you might want to go and do a proper, possibly non-linear, fit at each of your matched filter detections, but you can get an awfully good

picture of where those peaks are and roughly how broad they are very quickly, before starting your expensive fitting.