The basic version of the code if FullEfflength.f. It calculates radial profiles of the intensity by a several step process:

1) Calculate the electron distribution as a function of radial coordinate and energy. Thankfully, since we are assuming spherical symmetry, we only have to do this once and then we can find the distribution at points in between the tabulated grid through interpolation
2) Calculated the electron emissivity by convolving the above particle distribution with the standard one-particle emissivity (read in from fglists.dat)
3) Integrating the emissivity over lines of sight (called x in the code) to get the intensity

Adding that all up, since 1) requires an integral, that's a triple integral that you have to compute at each point in radius to find the intensity. This can get computationally demanding depending in the choices of grid sizes. Speaking of which, there is not really any rigorous reason for my choices of resolutions…I mostly just made them as high as possible without slowing down calculations too much. I think the most important place to have high resolution is in 1), but other than that it gets a little wishy-washy. You do want to make sure that the radial resolution for the intensity is high enough to have a significant number of grid cells in the FWHM, and to make sure the resolution is fairly consistent when you plot over a range of frequencies or diffusion coefficients.

The most complicated part of the code is calculating the electron distribution. The solutions for arbitrary diffusion coefficients are given by integrals over very ill-behaved Green's functions (note the singularities in the expressions in the paper), so a change of variables is required. For each case (mu=1, mu>1, mu>1) this change of variables is different, each chosen to eliminate the singularity. For instance, for mu=1, the new variable is t = sqrt(ln(n)), so you can prove to yourself by doing the substitution that this works. When the diffusion coefficient is very small, integrating over the Green's function is time-wasting since we know the solution analytically for advection-only transport, so if l_ad/l_diff >30 the code just uses that solution. You can play around with this condition if you want. There's also some different normalizations in the two distributions involved, but since the overall normalization of the distribution is irrelevant for FWHMs I just added in the correct numerical factor by hand so that they exactly match up for low diffusion coefficients.

Fitting Data

Right now, there is no fitting algorithm employed by the code. In fact, this is the aspect of the calculation that is in the most dire need of improvement. At some point I actively tried to add a Levenberg-Marquardt scheme that would treat the code as a vector function of two variables (B_0 and eta) that returns three values (FWHM at .7 keV, 1 keV, and 2 keV) and find the best fit parameters to best fit to the data, but something wasn't working quite right. I suspect it might be because this function is not the most well-behaved in the world, but it could have just been a bug. The reason I never completed this task was for two reasons: lack of time, and the fact that with only three data points (FWHM at each energy)we don't have amazing statistics anyway.

So, what I did was to manually plug in values of B_0 and eta until I found the right match for the FWHM and m_E value at 2 keV. This scheme simply ignores the FWHM at .7 keV. This sounds more painful than it actually is, because eta mostly determines m_E, so once you find the right eta you just have to vary B_0 to get the right FWHM. Then you can vary the parameters about this best fit value to estimate uncertainties (also manually).

I've found that a good resolution for this is irmax = 400 (intensity grid spacing), iradmax = 100 (spacing in the particle distribution table), and ixmax = 500 (spacing in line of sight). You just have to make sure you change rminarc to something close to the FWHM you are trying to fit.

Magnetic Damping Model

This version of the code is less cleaned up because I haven't used it in a while. There are declared variables that are never used, lots of stuff could be condensed, etc., but to the best of my knowledge, it works. Apparently I thought it might be a good idea to interpolate the electron distribution using splines instead of simple linear interpolation, so the code is more confusing because of that, but is basically the same as the above. The basic calculations work in the same way, except tabulating is more complicated because now the energy-frequency relation for synchrotron emission depends on space, so you have to be careful about the values of energy at which you calculate the distribution. Also, the analytic solution for the distribution for the case of lad>>ldiff is different because of the spatial dependence of B, which you can check by hand.

Now we have an extra parameter, so to prevent the hopeless situation of having more parameters than data points we assume Bohm diffusion. The new parameter is ab, which is the scale length of the field (i.e. B ~ exp(-r/ab)). I don't have any experience fitting this model to the data since SN1006 had FWHMs that depended strongly on energy while the damped model's signature is a very weak dependence on photon energy. I'd imagine that this model is much more difficult to fit to data, since now you can have both losses and damping be important.

If you could possible get a non-linear fitting algorithm working, then you should at least be able to extend this to eta!=1 since then you are incorporating all three data points and can have three free parameters (B,eta,ab).