Introduction

This document briefly covers how images are analysed and processed in the Matlab image analysis program `ImageAnalyser`. Derivations for formulas can be found in the green and yellow ring binder labelled “Image Analysis” and in references listed in this document.

Calculating Optical Depth

In each imaging cycle, three images are acquired by the camera and saved as ASCII files by Andor Solis. These files are overwritten if ImageAnalyser doesn’t process them. The three images are loaded into ImageAnalyser and stored in the following matrices:

`A` = probe light + atoms
`B` = probe light only
`C` = dark background image

There are now three possible methods for calculating optical depth selectable from the drop-down menu in the “Analysis Options” box, plus two methods for improving image quality, which will be discussed later. They are computed in `ana_sp1.m` and `ana_sp2.m` for species 1 and 2, respectively.

“Classic” Mode

A processed image `A` is generated like so: $A = ln[(A_{cam} - C_{cam})/(B_{cam} - C_{cam})]$
The values in the matrix `A` are now in units of optical depth instead of camera counts. We assume that the probe intensity is much less than the saturation intensity. From here, we can calculate the atom number in three main ways:

1. From the fits, we know the dimensions of the cloud and can determine the atom number using
   $$N_{OD} = 2\pi\sigma_x\sigma_z OD_{peak}/\sigma_{tot},$$
   where OD_{peak} is the peak optical depth from the fit, σ_x and σ_x are the cloud widths along each axis, and σ_{tot} is the total absorption cross-section. This gives us “Number from OD” in the results box.
2. We can also calculate the atom number by summing over the horizontal and vertical fits. This gives us the *x* and *y* atom numbers in the “Results” box.
3. If fits are poor, atom number can be extracted by summing over the pixels:
   $$N_{px} = \Sigma Ap^2/\sigma_{tot},$$
   where *p* is the pixel size. Note that because this sums over all of the pixels, best results are achieved when a small region of interest is used around the atoms. This gives us “N px sum” in the “Results” box.

Pixel-by-Pixel

In practice, the probe light we use is not uniform across the atomic cloud. We can more accurately extract atom number and density if we can determine the probe intensity on a

pixel-by-pixel basis, and therefore I/I_{sat} for each pixel. The method described below can be found in Robert Wild's PhD thesis (JILA) in section 5.1. We begin with the measured optical depth $A_{meas} = ln(B/A)$ where $A = A_{cam} - C_{cam}$ and $B = B_{cam} - C_{cam}$ (unfortunately, this notation is a remnant from the earlier versions of the image analysis program, and A_{meas} here is identical to the A calculated in classic mode).

The next step is to correct for saturation of optical depth. This is where the OD saturates in your system due to, for example, off-resonant light. Typically, it is around 3, but should be experimentally verified for a given imaging system. Once determined, this value goes into "OD_sat" in the configuration file editor. The correction to the optical depth is made as follows:

$$A_{mod} = ln(\frac{1 - e^{-OD_{sat}}}{e^{-A_{meas}} - e^{-OD_{sat}}})$$

This is something that can be corrected for even in classic mode, though it has not been implemented presently.

Finally, we correct for the variation in probe intensity across the image. This can have a significant effect on the atoms due to saturation of the atomic transition! Because it's difficult to exactly determine the probe intensity at the atoms, an experimentally-determined effective saturation intensity I_{eff} is used instead. We then have:

$$A_{actual} = A_{mod} + (1 - e^{-A_{mod}})\frac{B}{I_{eff}}.$$

To determine I_{eff}, take a series of thermal cloud images with low optical depths at different probe intensities, and then fit the above equation for A_{actual}. The fitting parameters are A_{actual} and I_{eff}. Figure 5.1 in Robert Wild's thesis shows an example of a measurement of I_{eff}. I_{eff} can now be entered in the configuration file editor as "I_sat_eff".

High Intensity Imaging

"But what if we really really want to image clouds with optical depths more than three??" you may ask. There is a solution (with caveats) that may let you image optical depths up to 8. "Wow!" you say, amazed at this unexpectedly positive turn of events. Again, Robert Wild's thesis (section 5.1.3) has the answers. You will need the measured value for I_{eff}, as described above, high probe power (>10 x I_{sat}), and a neutral density filter for the camera. This time, the full definition of optical depth is used:

$$A_{full} = ln(\frac{B}{A}) + \frac{B - A}{I_{eff}}.$$

At high intensities, photon re-absorption becomes significant, but this actually saturates once you get to high enough intensity.

A potential problem is atoms being Doppler-shifted out of resonance. To avoid this, probe pulses must be kept short. The number of photons absorbed can be calculated, and this can be used to work out the error in the optical depth. See Robert Wild's thesis for details.

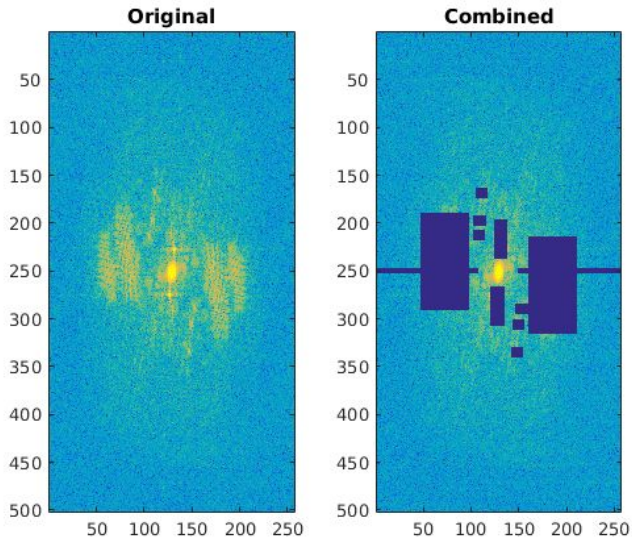
High intensity imaging is no good for thermal clouds due to poor signal to noise, so keep this in mind should you choose to implement it. At JILA, they used a lens on a flipper mount to focus their probe beam when they wanted high intensities.

Improving Image Quality Without Actually Improving Your Imaging System

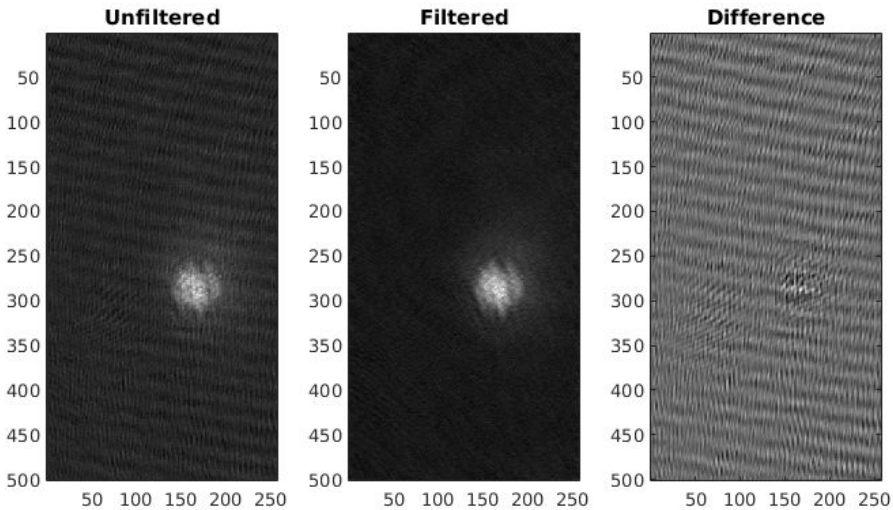
The new image analysis program has two options for image post-processing to get rid of unwanted fringes that result from imperfect background subtraction. You can use one or the other, or both at the same time.

Fourier Filtering

By taking a Fourier transform of your image, it's possible to identify frequency components that correspond to interference fringes coming from your imaging system. A Fourier filter is a mask of the same dimensions as your image that simply blocks out those frequency components. The filter is applied to the Fourier transformed image, and then an inverse Fourier transform is applied, leaving you with with a cleaner image.



The example on the right shows a Fourier transform of an absorption image of a BEC without (left) and with (right) a Fourier filter applied. The Fourier filter is composed of several rectangles that effectively mask parts of the Fourier transformed image. In practice, you will probably have to experiment with masks until you come up with one that works. If you look at the m-file `applyFourierFilter_sp1.m` for guidance, you can adapt it to try out different Fourier filters offline. There is a bit of code there that's superfluous to the image analysis program; only the last three lines actually do the Fourier transforms and apply the filter. As you can see below, the results are pretty good!



Things to note:

- If you change your binning or region of interest (in Andor Solis, not in the image analysis program), you will need to create a new Fourier filter. Otherwise, you should be able to use the same Fourier filter for all your images, but do be careful.
- It's easy to accidentally filter out features of importance in a BEC if they are a similar size to the fringes being filtered out. For example, it was very difficult to filter out fringes in BEC images containing vortices without filtering out the vortices themselves!
- `ImageAnalyser` does not save the Fourier filtered image. It only saves the raw `A`, `B`, and `C` images.

Fringe Removal by Creating the Perfect Background Image

Most of the time, the fringes seen in a series of processed images are of the same character, and the only thing that changes is their contrast and position. Using a collection of probe images (more is better, but can be computationally intensive), one can come up with a linear combination of images that results in the "perfect" probe background image (`B`) using an LU decomposition. This method comes from Caspar F. Ockeloen's masters thesis (University of Amsterdam). The maths is described in section 2.2 and there is a sample script in Appendix B.

You do not need to create a mask. Instead, just click "Remove Fringes" and you will be able to select the atoms as you would a region of interest.

Fitting

Two types of fitting may be performed. The first is a standard Gaussian fit and the second is a bimodal fit, which fits a Thomas-Fermi profile (parabola) to the BEC component and a Gaussian to the thermal component.

Gaussian Fitting

The distribution of atoms in a thermal cloud is described by $f(x) = A_0 e^{-(x-x_0)^2/2\sigma^2} + y_0$.
Slightly confusingly, this isn't quite the fitting function used in the analysis program (I have left this part of the code as I found it). Instead, the fitting function is given by

```
c(1) + c(2)*exp(-(x - c(3)).^2/(c(4).^2))
```

where `c(1)` corresponds to the vertical offset y_0, `c(2)` corresponds to the amplitude A_0, `c(3)` corresponds to the horizontal offset x_0, and σ = `c(4)`$/\sqrt{2}$. This is the σ that is displayed in the "Results" box.

This is the fit that is performed when "Gaussian" is selected for the Fit Type in the "Analysis Options" box.

Bimodal Fitting

If "Thomas-Fermi" is selected from the drop-down Fit Type menu in the "Analysis Options" box, the program will perform a bimodal fit. The code for the fitting routine is based on the

2D double Gaussian fit from the previous version of the image analysis program, hence why it seems convoluted. I have left it this way in case people want the 2D fit display back.

The double Gaussian fit fits a Gaussian to the thermal cloud and another Gaussian to the BEC component. It has to make an educated guess about where the BEC component is spatially in order to have good initial guess parameters. The code for this is found in `bimodal_findstart.m`, and you may need to modify it if the fit is bad. You will most likely need to adjust the initial guesses for cloud widths (the appropriate lines in the code are annotated). After the double Gaussian fit is performed, the BEC Gaussian fit parameters are used as input parameters for a parabolic fit (Thomas-Fermi). Several iterations of parabolic fitting are performed for best results.

The parabolic fitting function is of the form $f(x) = y_0 - A_0(x - x_0)^2$. In the image analysis program, this function is provided by `para_fit.m` and is written as

```
y = params(1) - params(2)*(x - params(3)).^2 .
```

The Matlab function `lsqcurvefit` (a least squares fit) is used to perform the fitting, whereas `fminsearch` is used for the other fitting. For practical purposes, there isn't really a difference.

Please don't use the bimodal fitting as the be all and end all for extracting useful information about your cloud. The choice of where to draw the line between the thermal component and the BEC can be tricky, so relying completely on the automated fitting routine is not recommended! If you need to accurately determine quantities such as condensate fraction and aspect ratio, I recommend analysing your images separately.

Output Parameters

A summary of how all of the output parameters in the "Results" box are calculated. Most of the outputs are based on the fitting routines described above.

- Top section: (almost) raw Gaussian fitting parameters. A potential future modification is to display the thermal cloud results here for the Thomas-Fermi bimodal fitting case. This can be changed based on feedback. Distance units are in pixels.
    - Background: `cx(1)` for *x*, `cz(1)` for *z*.
    - Amplitude: `cx(2)` for *x*, `cz(2)` for *z*.
    - Centre: `cx(3)` for *x*, `cz(3)` for *z*.
    - Sigma: `cx(4)/sqrt(2)` for *x*, `cz(4)/sqrt(2)` for *z*.
- Top section calculated parameters:
    - Aspect ratio TOF:
        - If the view angle is zero, then the aspect ratio after a time of flight is `cx(4)/cz(4)`, i.e., σ_x/σ_z.
        - If the view angle is a non-zero angle θ, then the horizontal size (*x*) in the trap must be corrected as follows:

$$cx_{corr} = \frac{1}{cos\theta}\sqrt{cx(4)^2 - cz(4)^2 sin^2\theta}$$

Then the aspect ratio is $cx_{corr}/cz(4)$. Note that this assumes radial symmetry of the cloud, i.e, that the y and z sizes are the same, which is reasonably for an optical dipole trap propagating in the horizontal direction. For a derivation of this, see the yellow and green ring binder.

- Optical Depth TOF: the optical depth after a time of flight is calculated from the fits to the *z* crosscut of the cloud and is equal to $A_0 + y_0$ from the Gaussian fitting formula above. In the code, this is calculated in `ana_sp1.m` and `ana_sp2.m`.
- PSD: phase space density. This is calculated as follows. We first start with the peak density n_{pk}, since this is a useful output parameter:

$$n_{pk} = N\varpi_{mean}{}^3\left(\frac{m}{2\pi k_B T_{mean}}\right),$$

where N is the number of atoms, $\varpi_{mean} = 2\pi(f_x f_y f_z)^{1/3}$ is the mean trapping frequency (frequencies f_j are in Hz), m is the atomic mass, k_B is the Boltzmann constant, and T_{mean} is the mean temperature from the x and z temperatures (see how temperature is calculated below). In the case of a bimodal fit, T_{mean} comes from the Gaussian fit to the thermal component. The phase space density is then

$$PSD = n_{pk}\left(\frac{h}{\sqrt{2\pi k_B m T_{mean}}}\right).$$

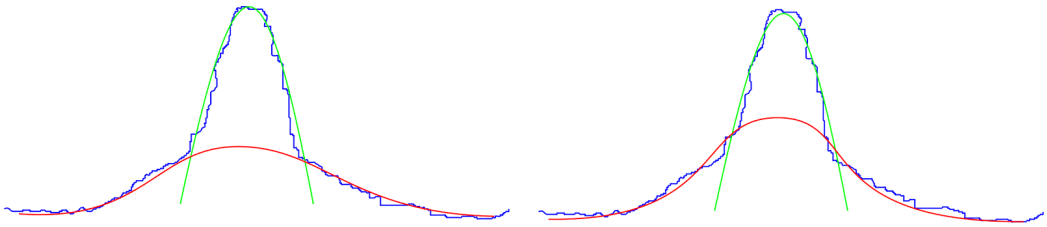
- Middle section:
  - Sigma trap: size of the cloud in-trap in μm along direction j, calculated from the trapping frequency f_j, time of flight t, and size after time of flight σ_j:

$$\sigma_{trap\,j} = \frac{\sigma_j}{1 + (2\pi f_j t)^2}.$$

    This is calculated from the Gaussian fit parameters.
  - Sigma TOF: size of the cloud after a time of flight in μm.
  - Number: atom number as calculated by taking the area under the x and z Gaussian fits.
  - Temperature: to calculate temperature, we use $T_j = \frac{m}{k_B}(2\pi f_j \sigma_{trap\,j})^2$. If bimodal fitting is being used, $\sigma_{trap\,j}$ for the thermal component is used.
  - Number from OD: this comes from the peak optical depth from the crosscut fits in the x and z directions as described in the Classic Mode section.
    - When the pixel-by-pixel or high intensity modes are used, the formulae for calculating atom number are the same, but the absorption cross-section σ_{tot} is applied pixel by pixel.
  - N px sum: this calculation does not rely on any fits at all, as described above in the Classic Mode section. We just sum over A/σ_{tot} scaled by the pixel area.
- Bottom section: bimodal fitting output parameters.
  - Centre: comes directly from `params(3)` i.e. x_0 in the parabolic fit.
  - TF Radius: the Thomas-Fermi radius. This is the spatial extent of the condensate. It is calculated by determining where the parabola fitted to the BEC component crosses zero.
  - TFn Radius (μm): same as the Thomas-Fermi radius above, but in pixels.
  - Condensate Fraction: in `fittingTF_sp1.m` (and `sp2`), the two parabolic fits (one along each axis) are used to generate a parabolic surface to complement the 2D Gaussian fit to the thermal cloud. Summing over the thermal fit gives us the number in the thermal component, N_{therm}.To get the

number in the condensate, we first subtract the 2D thermal fit from the 2D parabolic surface and retain only the positive values and sum over those, giving us the atom number in the condensate, N_{cond}. Subtracting the thermal component here is important so we don't sum over the same atoms twice. The condensate fraction is then $N_{cond}/(N_{therm} + N_{cond})$.

- Note that determining the condensate fraction this way is **inherently unreliable!** This is why:



In this sketch, the Gaussian part of the fit is only done to the wings of the distribution, so fitting an amplitude is unreliable. As the atom number is quite sensitive to the amplitude of the Gaussian, this leads to a lot of uncertainty in the condensate fraction.

- Aspect Ratio: this is the ratio of the *x* to the *z* Thomas-Fermi radii, giving the aspect ratio of the condensate.