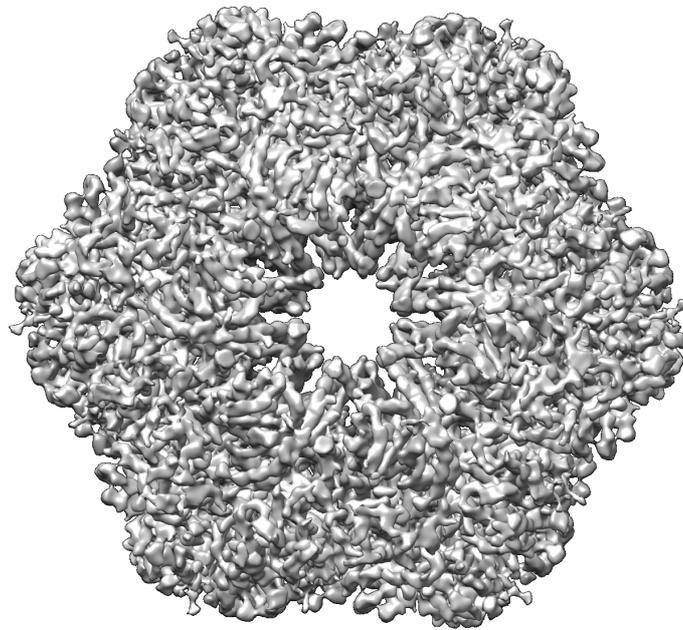


THE SIXTH BRAZIL SCHOOL FOR SINGLE PARTICLE CRYO-EM

HANDS-ON



Version 29-Sep-2014

© www.ImageScience.de

PART 1: [Introduction to IMAGIC](#)

PART 2: [The Fourier Transform](#)

PART 3: [The Data Set](#)

PART 4: [Single Particles Image Analysis](#)

[Content](#)

1. Introduction to IMAGIC

This chapter is on how to work with the **IMAGIC** software.

An **IMAGIC** image file actually consists of two files: the header file (".hed") and the image file (".img"):

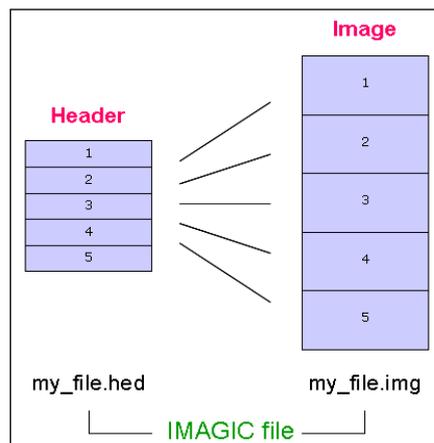


Fig. 1: IMAGIC file

The image file contains the actual image density values whereas the header file contains information about the images ("meta data") as a set of records that can be accessed through different labels. For example:

IMN	image location number (1,2,3,...)
IXLP	number of lines per image
IYLP	number of pixels per line
IZLP	number of sections if input is a 3-D volume
REF	multi-reference number
CLASSNO	class number
ALPHA	Euler alpha angle
BETA	Euler beta angle
GAMMA	Euler gamma angle
Etc...	

An additional PLT text file can be associated to an **IMAGIC** file to store further meta-data like:

- coordinates of particles
- contour of masks
- image numbers
- Euler angles
- graphics (curves)
- Etc...

The PLT file can contain a maximum of five numerical values per line, separated by blanks or by a comma.

A few other **IMAGIC** text (ASCII) files can be generated during processing:

- a CLS file is a classification file containing classes and their members

- a LIS file contains information printed during execution of a program

- a LOG file output of programs when running as batch job (script)

- a DAT file containing data for various purposes

- DFF (deFault Files) are used to store your last answers

The **IMAGIC** coordinate system is a right-handed system with its (1,1) origin in the top-left corner of the image. The length of the lines (number of rows/columns) is **NY** and the number of lines is **NX**:

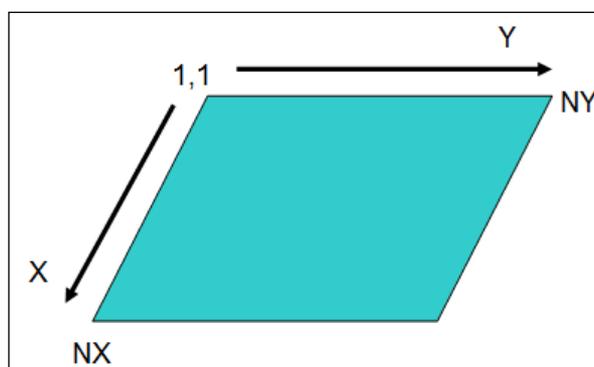


Fig. 2: IMAGIC 2-D coordinate system

The **IMAGIC** coordinates for a 3-D volume are the following:

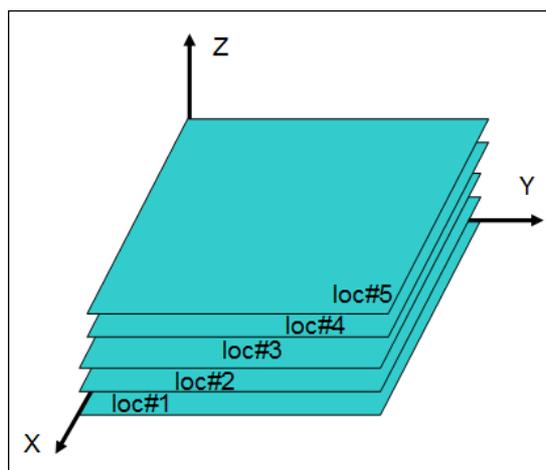


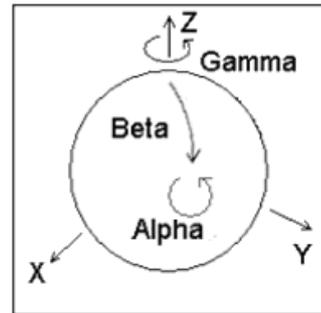
Fig. 3: IMAGIC 3-D coordinate system

Note that $\mathbf{Z} = \mathbf{X} \times \mathbf{Y}$ as required for a right-handed co-ordinate system.

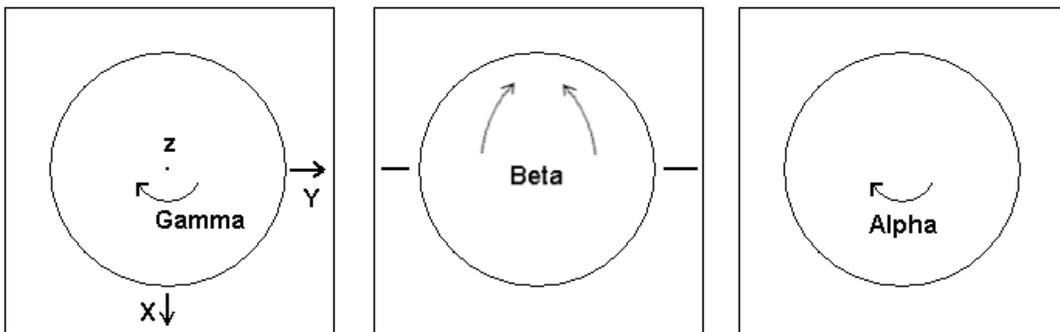
In **IMAGIC** 3-D orientations are defined by three Euler angles Alpha, Beta and Gamma.

From the perspective of an external viewer (like every IMAGIC image used/ created in commands **ANGULAR-RECONSTITUTION**, **THREED-SURFACE**, **THREED-FORWARD**, etc.) the Euler angles are defined as follows:

The first rotation is a rotation around the Z-axis by **GAMMA**, followed by a rotation **BETA** around the new Y-axis and a rotation **ALPHA** around the new Z-axis.



But normally a user does not think in this way but tries to imagine how the particle would look like "in his hands":



- Look at the particle along the Z-axis ("north pole")
- Rotate the particle clockwise by Gamma
- Rotate the particle into the plane clockwise by Beta
- Rotate the particle clockwise by Alpha

PLEASE NOTE:

The important angles to define a 3-D orientation are Beta and Gamma. Alpha is only the final in-plane rotation.

IMAGIC is started in one (or various) command window(s). The commands are interactive and are followed by specific questions. Every question also has an associated help, which can be accessed by typing "?"

IMAGIC command questions will often have a default value which appears in brackets [default]. You can use the default value by just hitting ENTER/CR.

IMAGIC remembers the last values you have entered for a specific command. These values (store in the DFF files) become the default values the next time command is used in that working directory. In general, if you do not know how to answer a question, the default values serve as an intelligent first guess.

MPI refers to parallel processing. If your notebook has multiple cores commands which are using parallel processing will ask you if you want to run the command in parallel or not. When using many images your answer will be **YES**. Note that the number of processors to be used should be at least the number of nodes PLUS 1:

Use MPI parallelisation [YES]	: yes
Number of processors to be used	: 3

Throughout this hands-on, words that appear in **GREEN** refer to **IMAGIC** commands. Words in **red** are required/suggested input values. Suggested file names are in **blue**.

File names are only suggested. You are free to choose whatever names you wish. However, bear in mind you will have to remember what you've chosen for the next commands.

YOUR NOTES:

1.1. Commands CREATE-IMAGE and DISPLAY

1. To start, open a command window and run **IMAGIC** by typing **i** or **imagic**.

```
my_notebook> imagic

IMAGIC-COMMAND:
```

2. Use **CREATE-IMAGE** to create a (test) image. First use the default options, i.e., just hit the ENTER button.

```
IMAGIC-COMMAND: create-image

** TESTIM welcomes you **

Output filename, image loc#s           : my_image
Image dimensions X,Y                   : 256,256
IMAGIC data formats you can choose     : real
Currently you can choose                : blobs      you choose
```

3. Use a separate command window to **DISPLAY** the image on the screen:

```
my_notebook> imagic

IMAGIC-COMMAND: display

Input image file, image loc#s          :
```

4. The first question that will appear on the screen concerns the choice of the file you wish to display. Get the test image (image), which you just created. If you have forgotten the names of the images type:

```
Input image file, image loc#s          : $dir      MS Windows
```

or

```
Input image file, image loc#s          : $ls      Linux
```

which is just an operating system call to get a list of the files you own, and look for files with the extension ".img".

Now specify the name of the image file you want to display:

```
Input image file, image loc#s      : my_image
```

DISPLAY first shows the current settings:

```
Current DISPLAY settings:

Input image FILE name           : my_image
LOCATION numbers                  : 1,1
Output DEVICE                    : XWINDOWS
DEVICE window size              : 800,1024
SCALE factor                     : 1.0
MINX, MAXX                      : 1,256
MINY, MAXY                      : 1,256
GREYVALUES                      : 2D local survey
ERASE screen before display     : no
STARTING point (top left)       : 1,1
Display of NAME & information    : file name and location
Video lookup table (VLT)        : linear black/white
...
Parameters to be changed:
NO_CHANGES(=DISPLAY), SETTINGS, OPTIONS [NO]:
```

5. Hit the ENTER key, which means that the default **NO CHANGES** is used and the image will be displayed.
6. If you want to change certain settings go for the words written in capitals. For example, to change the scaling factor:

```
Parameters to be changed:
NO_CHANGES(=DISPLAY), SETTINGS, OPTIONS [NO] : scale

Image size is: 128 x 128

Give scale factor for display           : 2
```

7. Hit ENTER to apply the changes and to go back to the DISPLAY parameter settings.
8. Then use option **GREYVALUES** to give different grey levels. Start with option **INTERACTIVE** and black, white levels **-10,10** and display. Use other black, white levels and display to see how brightness and contrast of the displayed image changes. After, use option **SURVEY, 2D_LOCAL**.

NOTE:

If you are displaying a gallery of images (aligned images, class averages, 3-D sections etc. you should always use the **GREYVALUES** options **SURVEY**, **GLOBAL**.

9. Play with **CREATE-IMAGE** again. Create **REAL** images, but of different sizes, and of different options.
10. **DISPLAY** the images. After this, create a **256,256** image of a **SIEMENS** star.
11. Use the **COARSE** command with factors of **6** to give coarsened images. **DISPLAY** them and see the effects of sampling size on your resolution. What is the size of your image now? How much detail can you see?
12. Use the **BLOW-UP-IMAGE** command (option **BLOWUP**) to blow up the coarsened images back to the original size. **DISPLAY** the results and compare them with the original and the coarsened images. Compare the output images of **COARSE** and **BLOW-UP-IMAGE**.

TIP:

You can open multiple **DISPLAY** windows from different command windows.

13. Use **CREATE-IMAGE** to create a new test image using the option **BLOBS**. Use **MOVE-IMAGE** to **ROTATE**, **SHIFT** and **COMBINE (ROT&SHIFT)** the image. **DISPLAY** the results. Remember the IMAGIC coordinate system (chapter 1).

YOUR NOTES:

1.2. Noise

You will create a sequence of test images and add noise to them.

1. Use **CREATE-IMAGE** to create 256 images of **CHECKERS**. Make the images **REAL** and of size **128,128**. Note: To create a file with multiple images you specify the start and final location numbers, like **my_image,1,256** where **my_image** is the file name, **1** is the start location and **256** is the final location.

```
IMAGIC-COMMAND: create-image
Output filename, image loc#s      : my_image,1,256
Image dimensions X,Y              : 128,128
IMAGIC data formats you can choose : real
Currently, you can choose         : checkers
Checker size                       : 16
```

2. Use **ADD-NOISE** to add noise to the images:

```
IMAGIC-COMMAND: add-noise
Option used                        : ADD_NOISE
Input filename, image loc#s       : my_image
Output filename, image loc#s      : my_image_noise
Mode of operation                  : noise
Mean, sigma of Gaussian noise     : 0,20
Random number seed                 : 0
```

3. **DISPLAY** the results.

1.3. Noise Reduction by Image Averaging

You will get some impression what image averaging means and why image processing can enhance the resolution of noisy images.

1. Use **SUM-IMAGES** (option **SOME_SUM**) to make sums of **2**, **8**, **64**, and **256** of the images from the file you added noise to. Input file is **my_image_noise**. Suggested output file names are: **my_sum_2**, **my_sum_8...** and **my_sum_256**. In "Location number(s) wanted:" you should specify **1-2**, or **1-8**, or **1-64**, or **1-256** accordingly:

```
IMAGIC-COMMAND: sum-images

Mode of summing           : some_sum
Input file, NO loc#s     : my_image_noise
Output file, ONE loc#    : my_sum_2      etc.
Variance file, ONE loc#  : none
Location number(s) wanted : 1-2          etc.
Numbers wanted           : all
```

2. **DISPLAY** each result and see the effects of image averaging on the signal to noise ratio.

YOUR NOTES:

1.4. MODE Commands

MODE commands allow the creation of batch/script files with a collection of commands that can be run from the command window.

1. Use command **MODE-ACCUMULATE**:

```
IMAGIC-COMMAND: mode-acc  
IMAGIC-COMMAND (ACC.) :
```

2. Now use commands **CREATE-IMAGE** and **SURVEY-DENSITIES**. The input file for **SURVEY-DENSITIES** is the output file of **CREATE-IMAGE**.

Remember: You can find out about a command using **HELP**, and about a question using **?** .

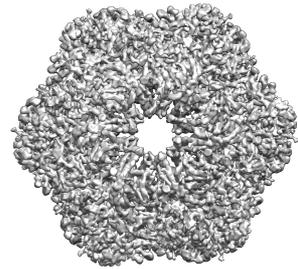
Stop accumulating commands with **MODE-STOP**. You can run the file with the accumulated commands in the command, e.g. **bigjob.b** (Linux) or **bigjob.bat** (MS Windows), respectively:

```
IMAGIC-COMMAND (ACC) : mode-stop  
Filename for batch/script file [bigjob] : bigjob  
  
Command (batch/script) file bigjob.b  
is available now  
  
To run the job on the monitor please use  
...
```

3. Use command **MODE-ACCUMULATE** again and accumulate command **TEST-IMAGE** and some other commands like **ARITHM-WITH-IMAGE**, **SURVEY...**
4. Stop accumulating commands and run the script file with **MODE-SEND**.
5. Use **MODE-PROTOCOL** to create a protocol file.

Again use command **CREATE-IMAGE** with some other commands.

Stop the protocol mode with command **MODE-STOP**. Edit the protocol file using a text editor of your choice.



2. The Fourier Transform

This is an exercise that will provide some basic insight into the Fourier transform (FT). Fourier transforms will be covered in the lectures; the aim of this exercise is to familiarize you with the principles of the Fourier transform and the associated **IMAGIC** commands.

Start with one-dimensional (1D) Fourier transforms and later play around with 2-D images.

2.1. Test Curves

1. Open a command window. Create a curve (a 1D image) using command **CREATE-CURVE**. Create an **IMAGE** with amplitude **1** and wavelength **0.1**:

```
IMAGIC-COMMAND: create-curve
Mode of output                : image
Output file, curve loc#s     : my_curve
Length of curve              : 512
Curve option                  : sine
Amplitude of signal          : 1
Wavelength of periodic signal : 0.1
```

2. Create additional curves into the same file, sequential locations:

```
IMAGIC-COMMAND: create-curve
Mode of output                : image
Output file, curve loc#s     : my_curve,2
Length of curve              : 512      as before
Curve option                  : sinc    you choose
Amplitude of signal          :        you choose
Wavelength of periodic signal :        you choose
```

3. Open a second command window and use the command **PLOT** to display the curves. Like the command **DISPLAY**, the **PLOT** command first displays the current settings, which you can change by typing the names in capitals. Giving **ENTER** means "**NO CHANGES**" and the curve is displayed. You want to compare all curves so it is a good idea to fix the vertical scaling of the plot according to the chosen amplitudes with the option **VERTICAL**:

```
Change settings (MULT,HOR,VER,SURVEY...) [NO]: vert
Minimum, maximum for vertical scaling      : -10,10
```

Plot the curves.

4. If you want to display all curves at the same time use option **MULTIPLE**:

```
Change settings (MULT,HOR,VER,SURVEY...) [NO]: mult
Number of curves per plot                    : you choose
```

2.2. Fourier Transform

1. Now calculate the Fourier transforms of the curves (**curve**) with the command **CURVE-FORWARD-FT**. The suggested output file name is **my_curve_ft**.

```
IMAGIC-COMMAND: curve-forw
Input file, curve loc#s      : my_curve
Output file, curve loc#s    : my_curve_ft
Options you can choose      : FORWARD_FT
```

2. **PLOT** the Fourier transforms (May be, you want to set **MULTIPLE** back to 1. You can also use option **VERTICAL** (values like -1000,1000) to use the same vertical scaling for all curves.

2.3. Curves and their Fourier Transforms

1. There is a interactive command to create images/curves and to display the related Fourier transforms: **PLAY-WITH-FOURIER-TRANSFORMS**.

```
IMAGIC-COMMAND: play-with-fourier

Play with                : 1d_image
Mode of input            : create
Choose curve             : triangle
Length of curve          : 512
Amplitude of the curve   : 1
Width of signal          : 0.25
```

Both, the curve and the related Fourier transform will be displayed.

You can create the next curve by typing **NEXT_IMAGE**:

```
How to continue          : next
...
```

or leave the command by giving **STOP_PLAYING**.

```
How to continue          : stop
```

NOTE:

In command **PLAY-WITH-FOURIER-TRANSFORMS** you can also use a curve/image from an input file. Use Mode of input : **FILE**

2. Now create various sine curves using different amplitudes **2, 4, 6, and 8**. Always use wavelength **0.1**:

```
IMAGIC-COMMAND: play-with-f
Play with                : 1d_image
Mode of input            : create
Choose curve             : sine
Length of curve         : 512
Amplitude of the curve   :                you choose
Wavelength of periodic signal : 0.1
```

Look at the vertical scaling of the plots and notice how the amplitudes are related to the height of the peaks in the Fourier transforms.

3. Continue using **SINE** curves now with fixed amplitude but changing the periodicity. Use amplitude **1** and wavelength of periodic signal **0.1, 0.25** etc.

```
IMAGIC-COMMAND: play-with-f
Play with                : 1d_image
Mode of input            : create
Choose curve             : sine
Length of curve         : 512
Amplitude of the curve   : 1
Wavelength of periodic signal :                you choose
```

Notice how the periodicity changes the Fourier transforms.

NOTE:

The sine (or cosine) curve in the image space is a peak in Fourier space. The amplitude and wavelength of the sine (or cosine) curve are "related" to the height and position in Fourier space.

2.4. Relationship between Image Space and Fourier Space

1. Create a new curve file (`my_curve`) with a number of **SINE** waves with various amplitudes and wavelengths. Create these into the same file, sequential locations (`my_curve,1` , `my_curve,2` , ... `my_curve,20`):

```
IMAGIC-COMMAND: create-curve
Mode of output: image
Output file, curve loc#s           : my_curve
Length of curve                     : 512
Curve option                       : sine
Amplitude of the curve             :          you choose
Wave length of periodic signal     :          you choose

IMAGIC-COMMAND: create-curve
Mode of output: image
Output file, curve loc#s           : my_curve,2
Length of curve                     : 512
Curve option                       : sine
Amplitude of the curve             :          you choose
Wave length of periodic signal     :          you choose

IMAGIC-COMMAND: create-curve
...

IMAGIC-COMMAND: create-curve
Mode of output: image
Output file, curve loc#s           : my_curve,20
Length of curve                     : 512
Curve option                       : sine
Amplitude of the curve             :          you choose
Wave length of periodic signal     :          you choose
```

2. Calculate the Fourier transforms of the new curves with **CURVE-FORWARD-FT** and store them in file (**my_curve_ft**).

```
IMAGIC-COMMAND: curve-forw
Input file, curve loc#s      : my_curve
Output file, curve loc#s    : my_curve_ft
Option used for current command : FORWARD_FT
```

3. As before **PLOT** both, the curves (**my_curve**) and the related Fourier transforms (**my_curve_ft**).
4. Now, sum all curves (**curve**) with command **SUM-CURVE**:

```
IMAGIC-COMMAND: sum-curve
Choose summing option      : total_sum
Input file, NO loc#s       : my_curve
Output file, ONE loc#     : my_curve_sum
Variance file, ONE loc#   : none
```

5. **PLOT** the result (**my_curve_sum**).

NOTE:

a) Summing a huge number of sine (and cosine) curves with different amplitudes and wavelengths creates a non-periodic curve.

b) And even more: one can say that any (real) curve can be constructed by a combination of sine and cosine waves of different wavelengths and amplitudes.

6. Calculate the Fourier transform (**my_curve_sum_ft**) of the new curve (**my_curve_sum**) using command **CURVE-FORWARD-FT**. **PLOT** the Fourier transform (**my_curve_sum_ft**).

NOTE:

The sum of sine (or cosine) curves in the image space relates to the sum of the sine (or cosine) peaks in Fourier space.

7. Now calculate a reverse Fourier transform with command **CURVE-REVERSE-FT**. Input is the Fourier transformed curve (**my_curve_sum_ft**).
8. **PLOT** the reverse Fourier transform. Note that it is the same as the original curve (**curve**) before the Fourier transformation.

NOTE:

The information in a curve/image ("Image Space" or "Real space") and in the Fourier transform ("Fourier Space") is equivalent. The curves/images in both spaces contain all curve/image information.

This means that in image processing it is possible to go from one space to the other without losing any image information!

NOTE:

Mode of Fourier transforms: FORWARD means going from a curve/image ("Image Space" or "Real space") to its Fourier transform ("Fourier Space"). The transformation going from the Fourier transform to the curve/image is called REVERSE (sometimes also called "inverse").

YOUR NOTES:

2.5. Fourier Space and Filtering

1. Let us once again create sine curves: one with a long wavelength and another with a short wavelength (**0.5** and **0.002**) using **CREATE-CURVE**. Remember to save these two curves to the same file using the same file name and different location numbers, like in **my_curve** and **my_curve,2** .
2. Calculate the Fourier transforms with **CURVE-FORWARD-FT** and compare the results using command **PLOT** (use option **MULTIPLE**).

NOTE:

(a) The first sine curve shows "large" details, which in Fourier space are represented by densities close to the centre of the Fourier transform.

(b) The second sine curve shows "small" details, which in Fourier space are located far away from the centre.

(c) Usually the very "large" details (density ramps, for example) and the very "small" details (mostly noise) are hiding the motif, which you are interested in.

(d) As seen in your two test curves Fourier space offers a nice possibility to remove this unwanted information: filter the Fourier transform close to the centre ("low frequencies") and at the borders ("high frequencies").

3. Create a new curve (**my_curve**) with **CREATE-CURVE**, with **BLOCKWAVE** for curve option, **500** for the amplitude and **0.25** for the wave length
4. Next create a ramp in location #2 (**my_curve,2**). Run **CREATE-CURVE** with option **RAMP**, and use **1** for the amplitude and **3** for the inclination.
5. Now add the block-wave and the ramp curves with **SUM-CURVE** using summing option **TOTAL_SUM**. Do not calculate a standard deviation (give **none**). Use the output name **my_curve_ramp**.
6. **PLOT** the sum (**my_curve_ramp**). Notice how the signal (block-wave) is disturbed by the ramp.
7. Remove these "large" unwanted details (low frequencies) with a high-pass filter in Fourier space using command **CURVE-FILTER** and option **HIGH_PASS** and low-frequency cut-off **0.01**.

```
IMAGIC-COMMAND: curve-filt
Input file, curve loc#s           : my_curve_ramp
Output file, curve loc#          : my_curve_ramp_hp
Filter option                     : high_pass
Low frequency cut-off            : 0.01
Remaining LF transmission         : 0,0
```

8. PLOT the high-pass filtered curve (`my_curve_ramp_hp`).

NOTE:

Large details can be suppressed by high-pass filtering in Fourier space. But, be aware that the low-frequency component of the original curve can also be affected.

Errors/artefacts can occur at the edges.

9. Next create Gaussian noise in location #3 (`my_curve,3`). Run `CREATE-CURVE` with option `NOISE`, and use `0, 40` for the MEAN and SIGMA.
10. Add the noise to the block-wave signal with `CURVE-SUM`.

```
IMAGIC-COMMAND: sum-curve
Choose summing option            : some_sum
Input file, NO loc#s            : my_curve
Output file, curve loc#         : my_curve_noise
Output standard deviation file   : none
Location number(s) wanted       : 1;3
```

11. PLOT the sum (`my_curve_noise`). Notice that the signal is disturbed by noise.

12. Remove these "small" unwanted details (high frequencies) by low-pass filtering in Fourier space with command **CURVE-FILT** using option **LOW_PASS** and high-frequency cut-off of **0.1**.

```
IMAGIC-COMMAND: curve-filt
Input file, curve loc#s      : cmy_curve_noise
Output file, curve loc#     : my_curve_noise_lp
Filter option                : low_pass
High frequency cut-off      : 0.1
```

13. **PLOT** the low-pass filtered curve (**my_curve_noise_lp**).

NOTE:

Noise (small details / high frequencies) can be removed by a (Fourier space) low-pass filter. But, of course, also fine details of the original curve are affected.

14. Finally, create a curve with disturbing low (ramp) and high frequencies (noise). Add all three curves (**my_curve**) with **CURVE-SUM**, using the option **TOTAL_SUM** (to get **my_curve_ramp_noise**).
15. **PLOT** the curve (**curve_ramp_noise**) to see how the curve is disturbed by the ramp and by Gaussian noise.
16. To remove both unwanted information call **CURVE-FILTER** again now using a **BAND_PASS** filter, which is a combination of a high-pass and a low-pass filter:

```
IMAGIC-COMMAND: curve-filt
Input file, curve loc#s      : my_curve_ramp_noise
Output file, curve loc#     : my_curve_ramp_noise_bp
Option to choose            : band
Low frequency cut-off       : 0.01
Remaining LF transmission   : 0
High frequency cut-off      : 0.1
```

17. As usual **PLOT** the filtered curve (**my_curve_ramp_noise_bp**) and see how a band-pass can remove unwanted information.

TIP:

During image processing, it is a good idea to have your own naming convention, so that in a list of files you can easily understand what each file is from its name. For example in this case "sine" is your input file containing a sine wave and "sine_mask" is your output file with the masked sine wave.

YOUR NOTES:

2.6. 2-D Images and Fourier Transforms - First Steps

We now want to play with 2D images and their Fourier transforms.

1. Start by using **PLAY-WITH-FOURIER-TRANSFORMS** using 2D images (option **2D_IMAGE**). First, create a **SINE**-wave image (**my_sine**). When asked for a **PERIODICITY** choose **0.1**:

```
IMAGIC-COMMAND: play-with-f
Play with                : 2d_image
Mode of input            : create
Choose curve             : my_sine
Image dimensions X,Y     : 512,512
Wavelength of periodic signal : 0.1
Direction of wave       : horizontal
Mask radius, drop-off (0: no mask) : 0          no mask
Grey values to scale display (0: auto) : 0          automatic
```

PLAY-WITH-FOURIER-TRANSFORMS will display the create images in the first display window and the related Fourier transform in the second display window.

2. You should now simply see two "points" in the Fourier transform of the input image. This is the Fourier space representation of a sine wave, with the periodicity you have specified (**0.1** is suggested above). When "in" Fourier space, information about higher frequencies (i.e. when the wave length is small) is given further towards the edge of the Fourier transform image, whilst information about the lower frequencies is given towards the centre.
3. As you are examining a 2-D image, the sine waves also have a direction. In this case the sine wave travels horizontally. In the Fourier transform, if you were to join the points with a line it would also go horizontally.
4. To demonstrate this effect continue with **NEXT_IMAGE** and create the same image again but now using option **VERTICAL**. Note that how the direction has changed in both, the image and the related Fourier transform.
5. Finally continue with **NEXT_IMAGE** now using the direction option **ANGLE**. Use **45** (degrees). As before you should find that the direction of the frequency space points has rotated by 45 degrees.

```
IMAGIC-COMMAND: play-with-f
Play with                : 2d_image
Mode of input            : create
Choose curve             : sine
Image dimensions X,Y     : 512,512
Wavelength of periodic signal : 0.1
Direction of wave       : angle
Rotatation angle        : 45
Mask radius, drop-off (0: no mask) : 0          no mask
Grey values to scale display (0: auto) : 0          automatic
```

6. However when using angles, which are not multiples of 45 (let's say: 30) there will no longer be just points. This is because the rotated sine waves are no longer continuous, and also have interpolation effects from the rotation. In other words the images are no longer pure sine waves
7. Use command `PLAY-WITH-FOURIER-TRANSFORMS` to play around with other images. Learn how the related Fourier transforms look like.
8. You can use command `CREATE-IMAGE` to create 2-D image files (`my_image`), which you can sum with command `SUM-IMAGES` (`my_image_sum`). Use command `PLAY-WITH-FOURIER-TRANSFORMS` to visualise images and Fourier transforms.

NOTE:

Like in the 1D case for curves, any (real) 2-D image can be seen as a combination of sine and cosine waves of different frequencies, and in different directions. A Fourier transform decomposes a real image into its constituent sine / cosine waves. Only sine waves that fit perfectly on the sampling grid, have perfect diffraction peaks in Fourier space.

2.7. 2-D Images and Fourier Transforms - Masks

1. The motif in your images is normally close to the centre of the frame and you are normally not interested in the information near the frame edge. To minimize the influence of background you may want to mask out the edges.
2. The effect of masking can also be visualised using **PLAY-WITH-FOURIER-TRANSFORMS**. Create a **SINE** image rotated by **30** degrees. First use no mask. To better visualise the result adapt the display grey-value scale:

```
IMAGIC-COMMAND: play-with-f
Play with                : 2d_image
Mode of input            : create
Choose curve             : sine
Image dimensions X,Y     : 512,512
Wavelength of periodic signal : 0.1
Direction of wave        : angle
Rotatation angle         : 30
Mask radius, drop-off (0: no mask) : 0          no mask
Grey values to scale display (0: auto) : 1,8
```

3. Subsequently mask out the centre of the image with a soft drop off circular mask. You should see the peaks more clearly now.

```
IMAGIC-COMMAND: play-with-f
Play with                : 2d_image
Mode of input            : create
Choose curve             : sine
Image dimensions X,Y     : 512,512
Wavelength of periodic signal : 0.1
Direction of wave        : angle
Rotatation angle         : 30
Mask radius, drop-off (0: no mask) : 0.7,0.1    soft mask
Grey values to scale display (0: auto) : 1,8        as before
```

NOTE:

You may have noticed that after applying the soft-circle, as well as the points becoming clearer, they also become larger. This demonstrates a very important image space / Fourier space relationship. A multiplication in image space (the application of a soft-circle is effectively a multiplication) leads to a convolution in Fourier space. This relationship occurs in both directions i.e. if you were to multiply the Fourier space image by a circular mask, you would get a convolution in image space (this is what filtering is), and similarly if you were to convolute in one space, you will get a multiplication in the other.

2.8. 2-D Images and Fourier Filters

1. As was done for the 1-D curves you can also use filters in Fourier space to remove unwanted information in 2-D images.
2. Create a new test-image showing a **RECTANGLE** and add some noise to it:

```
IMAGIC-COMMAND: create-im
Putput filename           : my_rectangle
Image dimension           : 256,256
...

IMAGIC-COMMAND: add-noise
Mode of operation         : ADD_NOISE
Input file                : my_ectangle
Output file               : rmy_rectangle_noise
Mwan, sigma of Gaussian noise : 0,5
Random number seed       : 0                your choice
```

3. Apply low-pass filters to the images ([my_rectangle_noise](#)) with the command **LOW-PASS-FILTER**:

```
IMAGIC-COMMAND: low-pass
Mode of operation           : LOWPASS
Input file                  : my_rectangle_noise
Output file                 : my_rectangle_noise_lp
High frequency cut-off     : 0.2                your choice
```

Play with different values for “High frequency cut-off” and always **DISPLAY** both, the original image ([my_rectangle_noise](#)) and its low-pass filtered version ([my_rectangle_noise_lp](#)).

4. Also apply high-pass filters onto the images ([my_rectangle_noise](#)):

```
IMAGIC-COMMAND: high-pass
Mode of operation           : HIGHPASS
Input file                  : my_rectangle_noise
Output file                 : my_rectangle_noise_hp
Low frequency cut-off      : 0.2                your choice
Remaining transmission      : 0
```

Play with different values for the “Low frequency cut-off” and **DISPLAY** both, the original image ([my_rectangle_noise](#)) and its high-pass filtered version ([my_rectangle_noise_hp](#)).

NOTE:

- (a) “Large” details are represented by low frequencies.
- (b) “Small” details are represented by high frequencies
- (c) Usually the very “large” details (density ramps, for example) and the very “small” details (mostly noise) are hiding the motif, which you are interested in.
- (d) Fourier filters offers a nice possibility to remove this unwanted information: filter the Fourier transform close to the centre (“low frequencies”) and at the borders (“high frequencies”). Such a filter is called a BAND-PASS FILTER.

- Now apply band-pass filters to a "real" image. In the data directory "Dataset_Wormhemoglobin/Particles_5" on the Brazil School network drives you can find an **IMAGIC** image file with five "worm hemoglobin" particles (`h_test`). Copy `h_test.hed` and `h_test.img` to your working directory.
- Apply the band-pass filter with command **BAND-PASS-FILTER**:

```
IMAGIC-COMMAND: band-pass

Mode of operation           : BANDPASS
Input file                  : h_test
Output file                 : h_test_bp
Low frequency cut-off      : 0.2           your choice
Remaining transmission     : 0
High frequency cut-off     : 0.8           your choice
```

- Open a second terminal window and **DISPLAY** the output file (`h_test_bp`) to check the result. Do NOT exit **DISPLAY** but call option **WATCHDOG**.
- Now play with different values of "Low frequency cut-off" and "High-frequency cut-off". Always use the same output file name (`h_test_bp`). **DISPLAY/WATCHDOG** will automatically display the new images (`h_test_bp`).

Use "extreme" band-pass parameters so that only high frequencies (`0.2,0,0.9`, for example) or only low frequencies (`0.05,0.005,0.1`, for example) remain are retained

- Next, adjust the low frequency and high frequency cut-offs to the particle size. Remember, a band-pass filter is combination of low-pass and a high-pass filter:

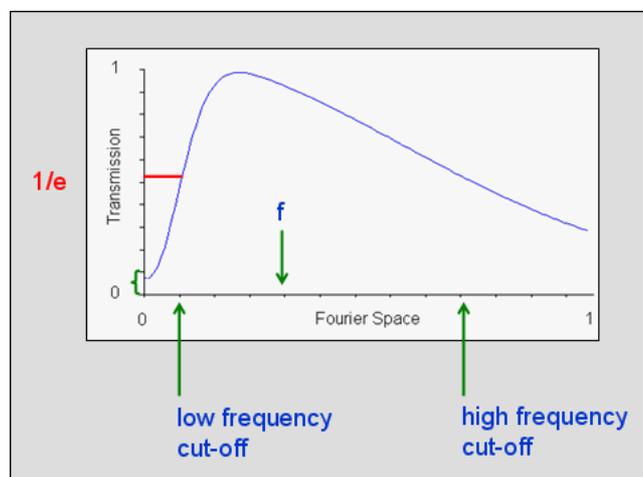


Fig. 4: Band-Pass Filter

Say that the image is scanned such that each pixel is of size

pixel size

The best resolution, which can (theoretically) be achieved for this sampling is given by the right edge of the Fourier transform image. It is the so-called Nyquist frequency

$2 \times$ pixel size

Which corresponds to the maximum spatial frequency

$$\frac{1}{2 \times \text{pixel size}}$$

Remember that the centre of the transform is zero spatial frequency.

Any cut-off value asked by **IMAGIC** filtering commands is a fraction f between 0 and 1 and corresponds to a spatial frequency

$$\frac{f}{2 \times \text{pixel size}}$$

The low-frequency cut-off: to remove all those low frequencies, which contain information larger than the size of your particle. These could be density ramps or other low-frequency information coming from the background of the images. You can adapt the low-frequency cut-off ("large patterns") to the size of the particle

$$\frac{2 \times \text{pixel size}}{\text{particle size}}$$

High frequency cut-off: to remove high frequencies containing mostly noise and little signal, thus increasing the overall SNR (signal to noise ratio) of the images. Adapt the high frequency cut-off ("small patterns, noise") to the expected resolution:

$$\frac{2 \times \text{pixel size}}{\text{expected resolution}}$$

The size of the test particle is 200 Angstrom and the pixel size is 5.43 Angstrom. Expecting a resolution of 15 Angstrom you get:

$$\text{LF cut-off} = \frac{2 \times \text{pixel size}}{\text{particle size}} = \frac{2 \cdot 5.43}{200} = 0.0543$$

$$\text{HF cut-off} = \frac{2 \times \text{pixel size}}{\text{exp. resolution}} = \frac{2 \cdot 5.43}{15} = 0.724$$

So you can use:

LF cut - off: 0.05

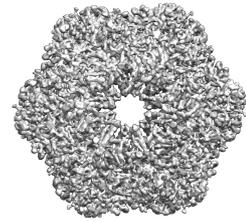
HF cut - off: 0.75

The **IMAGIC** filter "cut-off" parameters are very gradual Gaussian drop-off values and do **not** correspond to sharp masks in Fourier space!

NOTE:

2 x pixel size is the Nyquist frequency which is the theoretical limit to the resolution that can be achieved.

YOUR NOTES:



3. The Data Set: Worm Hemoglobin

Hemoglobin (Hb) is the iron-containing oxygen-transport metalloprotein present in the red blood cells of vertebrates. In earth worms (*Lumbricus terrestris*), the hemoglobin (sometime spelt as haemoglobin; also known as erythrocrucorin) is extracellular, freely dissolved in the blood as a 3.6 MDa dodecameric assembly.

Point-group symmetry: Dodecameric assembly D6 (622)

Data was collected under the following conditions:

Data collection: micrographs were collected on an FEI Titan KRIOS TEM with a FEG operated at 300 kV.

Spherical aberration: 2.7 mm

Focal distance: 3.4 mm

Pixel Size: 1.36 Å

Number of TEM images: 297

Size of a single TEM image: 4096 x 4096

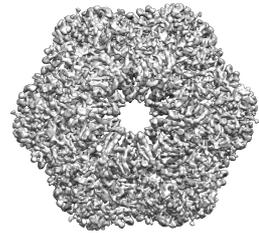
In the directory "Dataset_Wormhemoglobin" on the Brazil School network drive you can find the following data sets:

Directory "Raw_Micrographs_15":
15 raw micrographs (MRC format)

Directory "Micrographs_297":
All 297 coarsened/filtered micrographs (**IMAGIC** files)

Directory "Micrographs_297_raw":
All 297 raw and coarsened raw micrographs (**IMAGIC** files)

Directory "Particles_5":
Five worm hemoglobin particles (**IMAGIC** file)



4. Image Analysis: Short Overview

The hands-on processing of the worm hemoglobin dataset:

- Prepare and CTF correct the Micrographs
- Automatically Pick Particles
- Extract/box and pre-treat the Single Particle Images
- Alignment-by-Classification
- Multi-Reference Alignment
- MSA-Classification
- Angular Reconstitution
- 3-D Reconstruction
- Re-Projections and Iterative Refinements
- Fourier Shell Correlation
- More...

5. Micrographs

The raw data will first be organized as a “stack” of images, which will be treated as your initial, raw data. To achieve this, the micrographs will be appended together into one single file.

1. Before you begin appending the micrographs you should look at the original micrographs to get an idea what is the output of the FEI Titan Krios TEM (or the related EPU software, respectively).
2. You will find 15 of these TEM images on the data directory “Dataset_Wormhemoglobin/Raw_Micrographs_15” on the Brazil School network drive. The microscopical images are stored in MRC format in sub-directory “mrc”, TEM information in sub-directory “xml”. To visualise the micrographs with a display programs of your notebook you can use sub-directory “jpg”. Here the TEM images are stored in JPG format (size 512x512).
3. Copy the 15 MRC images to your working directory.
4. To be able to process the TEM images in **IMAGIC** you need to convert them into **IMAGIC** format. First you need to write the file names of all micrographs you want to process into a text file. Create this text file ([filenames.txt](#)) with your text editor and add the micrograph file names
[GridSquare_0000000032_FoilHole_0000000002_Data_0000000003_20120416_172245.mrc](#)
...
[GridSquare_0000000032_FoilHole_0000000016_Data_0000000003_20120416_181011.mrc](#)
one name per line.
5. You maybe intend to refine with **Frealign** in a later stage. That program requires a micrograph identification number, which we can already write into the header of the **IMAGIC** files. To be able to do this you should create an additional text file ([filenumbers.txt](#)) with the micrograph numbers:
[172245](#)
...
[181011](#)
one number per line. (You can also insert this numer, and many other general parameters) at a later processing stage using one of the many options of the HEADERS-INFORMATION command).
6. Convert the wanted TEM images with the command **IMPORT-EXPORT** (same command as **EM2EM**). Remember that the pixel size of these micrographs is 1.36 Å:

```
IMAGIC-COMMAND: import-ex
Convert 2D images or 3D volumes           : 2d
Data format of the input to be converted: mrc
MRC version                               : fei_epu
Type of input file                         : set_of_files
Export to which data format               : imagic
How to get import file names              : file_of_filenames
File of input file names                  : filenames.txt
Output image file                         : micrograph
Use standard em2em coordinate conversion: yes
In case of type/format conflicts,
      which preference: keep_densities
How to get the image names/titles         : name_of_import
Set some additional output header values: yes      we want to
                                                    store the
                                                    EM data
How to get the defocus values             : no_defocus
                                                    not yet
                                                    determined
How to get the EM parameters              : interactive
Microscope acceleration voltage           : 300
Focal distance of objective               : 3.4
Spherical aberration                      : 2.7
How to get the micrograph numbers         : file_of_numbers
Text file with numbers                    : filenumbers.txt
```

NOTE:

Remember that every **IMAGIC** command provides detailed help. You can access it by typing `HELP <command>`. Also every command question provides help that can be accessed by typing a `?` next to it.

7. Suppress extreme low frequencies by applying a band-pass filter. Use command **PREPARE-IMAGE**. Set the remaining low frequency transmission to **0**. One can also use this command to remove extreme density values in the micrographs (such as dead pixel in the CCF camera, dust on a scanned micrograph etc.). Do not use a mask:

```
IMAGIC-COMMAND: prep-im
Mode of operation           : PREPARE_IMAGES
Input file, image loc#s    : micrograph
Output file, image loc#s   : micrograph_filt
Low frequency cut off      : 0.01
Remaining low-freq. transm. : 0.
High frequency cut off     : 0.99
Mask radius, drop-off (0: no mask) : 0          0 means no mask
Desired new sigma          : 10
Remove (dust) outliers     : yes
Outliers off beyond which sigma : 5
Invert the image densities  : yes
```

NOTE:

If you have a huge number of micrographs you can use command **CAMERA-NORM** to normalize the micrograph images and to correct for camera pixel errors etc.

8. To reduce the computing time of all subsequent image processing coarsen the micrographs (**micrograph_filt**) by a factor of **4** with the command **COARSE-IMAGE**:

```
IMAGIC-COMMAND: coarse-im
Input file, image loc#s    : micrograph_filt
Output file, image loc#s   : micrograph_filt_coarse
Summing parameter          : 4
```

Note that the resolution (Angstrom/pixel) of the coarsened images ([micrograph_filt_coarse](#)) has changed!

Also note that you can later use the un-coarsened images ([micrograph_filt](#)) for refinements.

NOTE:

We very often use command **COARSE-IMAGE** because it is a very fast operation. In command **RESIZE-IMAGE** you can use the more sophisticated options **BLOW-UP/DOWN** or **RESIZE_IN_FS**.

NOTE:

Due to time constraints, you currently only used 15 TEM images. But now you should continue the practical using all 297 TEM images available. To avoid the time and CPU consuming commands, which you have used until now, we have prepared an **IMAGIC** image file ([micrograph_filt_coarse](#)) containing the filtered and coarsened images of all 297 TEM images. The image analysis will be continued with these 297 micrograph images.

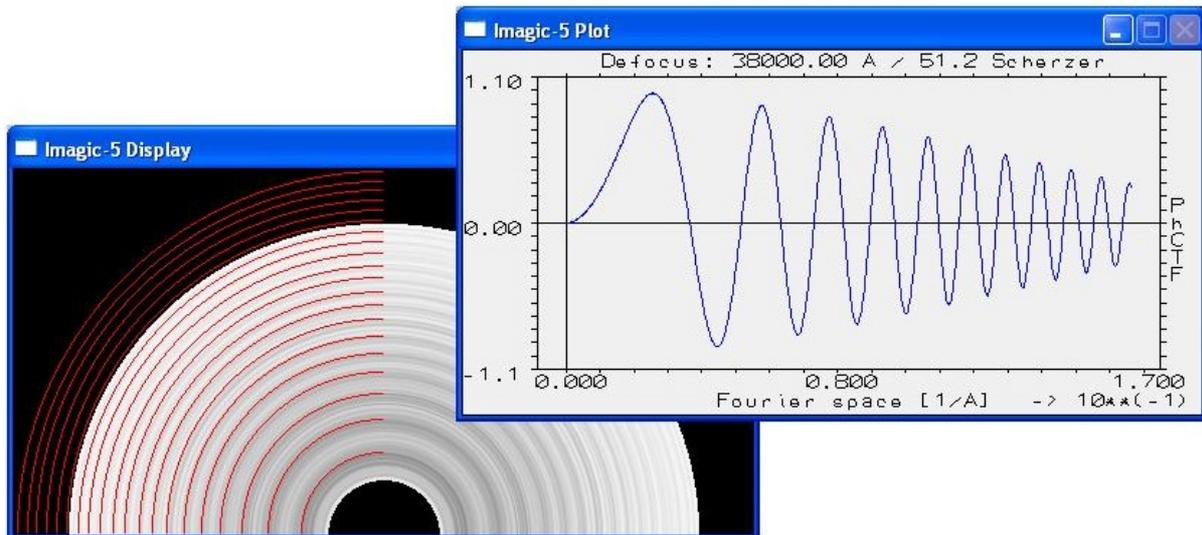
9. Copy the files [micrograph_filt_coarse.hed](#) and [micrograph_filt_coarse.img](#), which you can find on the on the Brazil School network drive (data directory "Dataset_Wormhemoglobin/Micrographs_297") to your working directory.
10. To check if the copy was done correctly you can use the command **HEADER** option **HOWMANY**:

```
IMAGIC-COMMAND: header
Specify option           : howmany
Input (header) file, image loc#s : micrograph_filt_coarse
```

The command **HEADER** should tell you that there are 297 images, which have a size of 1024x1024.

NOTE: The pixel size of the coarsened images is 5.43 Angstrom.

6. Contrast Transfer Function (CTF)



As was shown in the lectures an electron microscope unfortunately does not image all frequencies in the same way. This CTF exercise is meant to play around with command **TRANSFER**, which is an interactive program to calculate the (rotationally symmetric) CTF according to chosen microscope parameters.

6.1. Playing around with EM Parameters and their Influence on the CTF

1. First open a command window and call command **TRANSFER**. Note that this command is an interactive command with many parameters like in commands **PLOT** and **DISPLAY**, which you already know. You can use keywords written in capitals to change important parameters. **TRANSFER** shows the parameter settings until you give CR/ENTER, which means NO CHANGES, i.e. go ahead and display the CTF curve:

```
IMAGIC-COMMAND: transfer
```

TRANSFER displays the settings:

```
Current TRANSFER settings:
=====
Desired TRANSFER function      : Phase CTF
Acceleration VOLTAGE          : 200          kV
Relativistic WAVE length in Angstrom: 0.025045  Angstrom
CHROMATIC aberration          : None
SPHERICAL aberration constant : 2.2          mm
FOCAL length of objective     : 1.6          mm
APERTURE of objective lens    : 50.0        micro m
Coherent illumination SOURCE/ANGLE : 0.0
DEFOCUS value                 : 890.7438965 Angstrom
GENERAL defocus values        : 1.2          Scherzer
OBJECT size defocus envelope   : Off
LENGTH of transfer function    : 640         pixel
PIXEL size in curve           : 1.0         Angstrom
-----
MODE of operation             : Calculation of CTF
Output DESTINATION for plot(s) is : IMAGIC plot
Change options (VOLT,DESTIN.,MODE, etc. ...) [NO] :
```

2. First play around with different pixel sizes:

```
Change options (VOLT,DESTIN.,MODE, etc. ...) [NO] : pixel
Pixel size measured in Angstrom                   : 5.43
```

3. Next change the **GENERAL** defocus value to **1** Scherzer:

```
Change options (VOLT,DESTIN.,MODE, etc. ...) [NO] : general
General defocus value                             : 1
```

4. Now give CR/ENTER to display the CTF curve.
5. Now you can play around with other **GENERAL** defocus values (**2, 5, 10, 30...**) and notice their influence on the CTF.

NOTE:

As you already learned in the lectures: In Scherzer focus (GENERAL defocus value = 1.0 Scherzer) you have good image contrast over a large range of frequencies but, unfortunately, you have very little image contrast in the low frequencies and, as a result, you cannot recognize your particles. Using higher Scherzer foci lower frequencies are better transferred, thus, enabling us to see the particles. But, unfortunately, you now get more frequencies, which are not imaged at all (the "zeroes") and even worse, some frequencies are imaged with reversed contrast.

6. Also play around with other parameters (**VOLTAGE** etc.) and examine the related CTF curves.
7. For a low **VOLTAGE** parameter also define a **CHROMATIC** aberration. Examine the related CTF curves.

NOTE:

A large amount of chromatic aberration creates an envelope function, which is imposed onto the CTF so that the very high frequencies are not transferred any more. Even CTF correction cannot restore these higher frequencies.

YOUR NOTES:

6.2. Interactive CTF Correction

Although there are **IMAGIC** commands to automatically correct for CTF it is a good idea to use command **TRANSFER** for an interactive determination of the CTF. Use **DISPLAY** to choose one of the filtered micrographs (`micrograph_filt`) to play around with. Remember its location number. As before use command **TRANSFER**.

1. First specify the important EM parameters. In this case:

```
Change options           : voltage
Acceleration voltage in kV : 300
...
Change options           : spherical
Spherical aberration in mm : 2.7
...
Change options           : focal
Focal distance           : 3.4
...
Change options           : pixel
Pixel size measured in Angstroms : 1.36
...
```

2. To estimate the CTF call option **FIND_CTF**:

```
Change options           : mode
Dimension of the data set : 2d
Choose mode of operation : find
Input file, image loc#s  : micrograph_filt, your
                           location number
Default filter parameters : no
Dimensions of checkers   : 640,640
LF cut-off, HF cut-off   : 0.015,0.2
Inner and outer radius of mask : 0.1,0.5
```

These calculations can take some time. At the end **TRANSFER** displays the CTF curve (bottom)

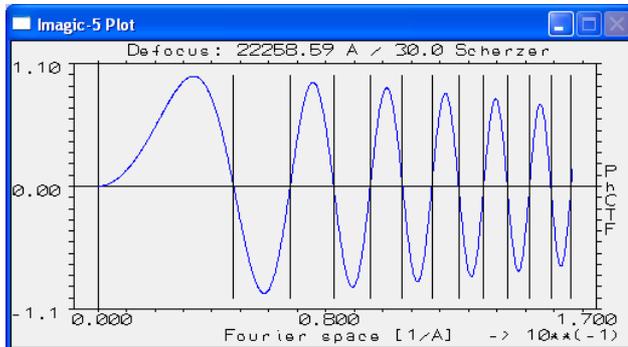


Fig. 5: CTF plot in TRANSFER

Profile of the rotational power spectrum (in the middle)

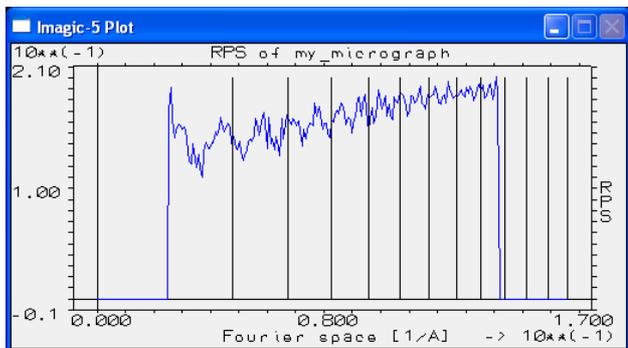


Fig. 6: RPS plot in TRANSFER

as well as the rotational power image (top)

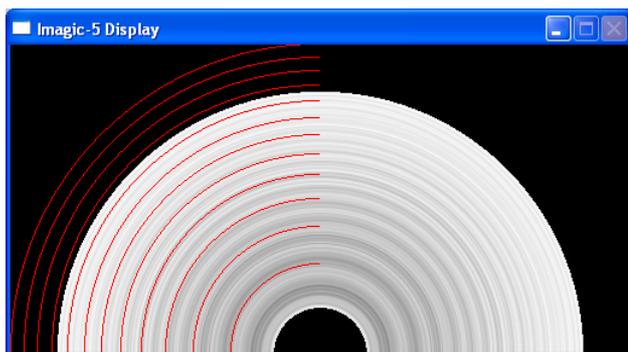
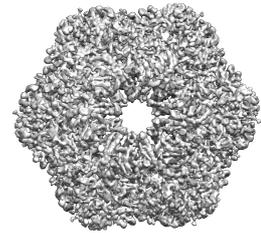


Fig. 7: RPS image in TRANSFER

In the CTF curve vertical lines mark the zeroes. The positions of these zeroes are also shown in the rotational power spectrum profile (vertical lines) and the rotational power spectrum image (red lines).

3. Play around with various defocus values until the zeroes in the CTF curve (lower curve) and the zeroes/Thon rings in the micrograph images (images above) are the same.

Defocus found:



7. (Automatic) CTF Correction

Coming back to your stack of images, you will proceed by CTF correcting the stack rather than the individual micrographs. This is one of the advantages of having all the data in one single file.

7.1. Calculate pre-treated Amplitude Images

1. Calculate the amplitude images ([micrograph_ampl](#)) of the micrographs ([micrograph_filt_coarse](#)). Before the amplitudes are calculated the micrographs will be masked and once more band-pass filtered. These are the first filter parameters, which you are asked to specify. The amplitudes itself will also be masked and band-pass filtered (especially the background has to be removed by reducing the low frequencies). Having applied this filter the Thon rings should be better visible. The command to do all this is [CREATE-PRETREATED-AMPLITUDES](#). Output will be the pre-treated amplitudes ([micrograph_ampl](#)):

```
IMAGIC-COMMAND: create-pre-ampl

Option used                : AMP_PRETREATED
Input file, image loc#s    : micrograph_filt_coarse
Output file, image loc#s   : micrograph_ampl

  Before the calculation of the amplitudes the images
  will be band-pass filtered to remove low frequencies.
  Please specify (0,0: no filter):

Low frequency cut off      : 0.25    High: data is coarsened
Remaining low-freq. transm. : 0
High frequency cut off     : 0.99

  The image will be masked by a soft circle.
  Please specify:resize-im

Mask radius, drop-off (0: no mask) : 0.9,0.05
Apply which arithmetic operation  : nothing
```

You can cut out the inner part of the amplitudes.

Please specify (0,0: no cut):

Cut amplitudes into which size : 0,0

Finally also the amplitudes will be band-pass filtered to better visualize the Thon rings. Please specify the band-pass for the amplitudes (0,0: no filter):

Low frequency cut off : 0.013 *very small*

Remaining low-freq. transm. : 0.01 *NEVER use 0*

High frequency cut off : 0.4 *no HF*

Resize the final amplitude images : yes

Coarse factor : 4

2. Now it is necessary to check if the filter parameters were chosen correctly. First we average all pre-treated amplitudes (`micrograph_ampl`) with the command `SUM-IMAGE`:

```
IMAGIC-COMMAND: sum-image
```

Mode of summing : total

Input image file, No loc#s : micrograph_ampl

Output image file, ONE loc# : micrograph_ampl_sum

Variance file, ONE loc# : none

3. `DISPLAY` this sum (`micrograph_ampl_sum`):

You can (but you don't have to) adjust the `DISPLAY` parameter `GREVVALUES` to better visualise the Thon rings:

Generate a `PROFILE` of the central line to check the band-pass parameters:

```
Change options (VOLT,DESTIN.,MODE, etc. ...) : profile
```

```
Use cursor to position profile : no
```

```
Starting point (IMAGE coordinates X,Y) : 129,129
```

```
End point (IMAGE coordinates X,Y) : 129,256
```

```
...
```

```
Change options (VOLT,DESTIN.,MODE, etc. ...) : no
```

If the CTF curve does not converge to zero, the low frequencies are not yet reduced enough and the parameter in **CREATE-PRETREATED-AMPLITUDES** should be enhanced:

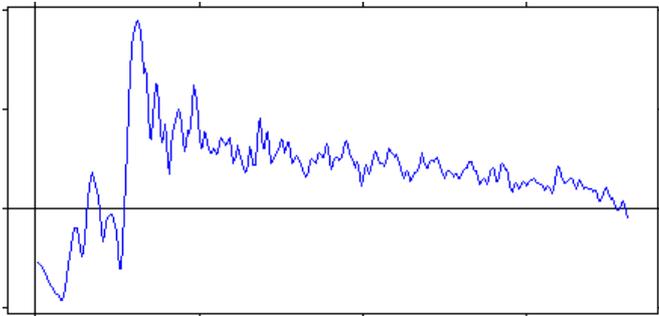


Fig. 8a: Profile in DISPLAY

If the CTF curve approaches zero for high frequencies the band-pass parameters were chosen correctly:

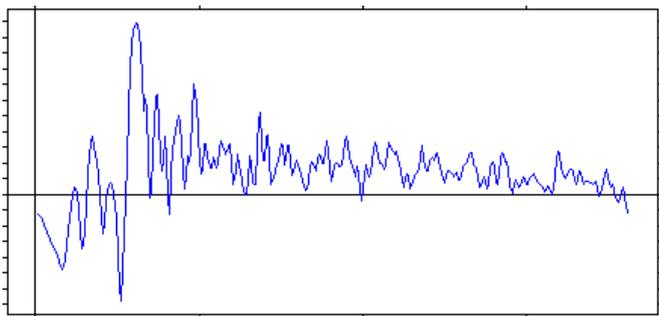


Fig. 8b: Profile in DISPLAY

NOTE:

If the checks are done, do not forget to reset your **DISPLAY** parameters **GREYVALUE** to **SURVEY** and **2D_IMAGE** or **GLOBAL**.

YOUR NOTES:

7.2. MSA of Amplitude Images

1. Certain areas (like the central part as well as the vertical and horizontal line central lines) are not of interest and should not be taken into consideration for MSA and classification. So create a **TOTAL_SUM** mask for MSA:

```
IMAGIC-COMMAND: msa-mask

Option used                : MSAMASK
Mode of mask                : total_sum
Input file, image loc#s    : micrograph_ampl
Output filename, image loc#s : micrograph_ampl_msamask
Additional mask             : ring_mask
Ring mask radius1, radius2 : 0.33,0.9
```

2. **DISPLAY** the MSA mask (`micrograph_ampl_msamask`) and check if the ring mask correctly masks out the unwanted inner and outer parts. If not redo the command **MSA-MASK** using other radii.
3. Run **MSA-RUN** on the amplitude images (`micrograph_ampl`).

```
IMAGIC-COMMAND: msa-run

Choose mode of operation    : fresh
MSA distances                : modulation
Input (= output) file       : micrograph_ampl
Input MSA mask file         : micrograph_ampl_msamask
Eigenimages output file     : micrograph_ampl_eigen
Use default answers for
other MSA options           : yes
Number of iterations        : 40
Number of eigenimages       : 10
Rootname for results file   : micrograph_ampl_msa
```

NOTE:

MSA-RUN is a CPU intensive command when using large data sets (even when running in MPI parallel). So it can be a good idea to use commands **MODE-ACCUMULATE** (and later **MODE-STOP**) to create a batch job and run it over night or during lectures.

7.3. Estimate CTF using MSA Eigen-Filtering

In this approach the eigenimages, which show Thon rings are used to MSA eigen-filter the amplitude images ([micrograph_ampl](#)). After this MSA eigen-filtering the amplitude images ([micrograph_ampl_eigenfilt](#)) should also clearly show Thon rings.

1. **DISPLAY** the eigenimages ([micrograph_ampl_eigen](#)) generated by **MSA-RUN**. Check which is the highest eigenimage that still contains clearly visible Thon rings. This will be the number of "Active eigenimages" which you have to specify in the next command (**MSA-EIGEN-FILTER**).
2. Apply **MSA-EIGEN-FILTER** to the MSA treated amplitude images ([micrograph_ampl](#)). Specify the highest eigenimages as explained above:

```
IMAGIC-COMMAND: msa-eigen-filt

Input file (treated by MSA)      : micrograph_ampl
Input eigenimage file, NO loc#s  : micrograph_ampl_eigen
Output file                      : micrograph_ampl_efilt
Output quality (PLT) file       : micrograph_ampl_qual
First and last eigenimage       : 1,2   the last eigenimage must still
                                     contain Thon rings (see above)
Weight of the first eigenvector  : 0.5
Normalise by variance           : no
Full output                     : no
```

3. **DISPLAY** (some of) the eigen-filtered micrographs ([micrograph_ampl_efilt](#)) and compare these images with the unfiltered ones ([micrograph_ampl](#)).
4. Next, call command CTF-FIND. Input is the file with the eigen-filtered micrographs ([micrograph_ampl_efilt](#)). Note that this a CPU intensive, time consuming command:

```
IMAGIC-COMMAND: ctf-find

Input amplitude image file, loc#s : micrograph_ampl_efilt
Output CTF "check" file, loc#s   :
                                   micrograph_ampl_half_half
Show orrelation area in half-half : yes
PLT output file with defocus values:
                                   micrograph_ampl_defocus
All EM data in input header       : yes
Inner and outer correlation radius : 0.33,0.9
                                   inner: radius of MSA mask
Defocus search range              : 15000,50000
Step size for search              : 1000
Maximum astigmatism level expected : 3000
Use partial coherence             : no
Generic envelope function halfwidth: 0.4
Did you cut the amplitudes        : no
Full output                       : no
```

NOTE:

Like **MSA-RUN** command **CTF-FIND** is a CPU intensive and time-consuming command (even when running in MPI parallel). So think about running this command in batch mode and create the related script/batchjob with commands **MODE-ACCUMULATE** and later **MODE-STOP**.

Important settings of command **CTF-FIND** are explained here:

- Input amplitude images (**micrograph_ampl_efilt**): The input file consists of the MSA eigen-filtered amplitude images. One should be able to identify a number of "zero" transfer (Thon) rings, and it is by fitting a theoretical CTF to these rings that the CTF parameters are estimated.
- Output "found" CTF (**micrograph_ampl_half_half**): Each image in this output file will contain a) in the left half: the input amplitude image, and b) in the right half: the "estimated" CTF. These "half_half" images should be used to check the accuracy of the CTF estimation. The Thon rings of both half should fit.

- PLT output file ([micrograph_ampl_defocus.plt](#)): This text file will contain the estimated CTF parameters, such as defocus #1, defocus #2 and the direction of astigmatism (defocus angle). This file can be used as an input to command **CTF-FLIP** for flipping the phases, and can also be opened with a text editor to view the results of the fitting.
 - Inner and outer correlation radius: A normalized cross correlation is used to compare the filtered experimental amplitude image to the theoretical CTFs. The centre and periphery of the amplitude image do not contain rings and are not important in the estimation. Therefore, the cross-correlation is only computed over a ring area specified by two radii. You can play with these parameters to obtain the best estimation - or simply try the suggested values.
 - Defocus range and step size: Here you can set the parameters for the initial brute force search. The first parameter is the start of the search, the second is the end of the search and the third is the step size over which the search is conducted. You can play with these parameters to obtain the best estimation - or simply try the suggested values.
5. The CTF parameters determined by **CTF2D-FIND** are stored in the PLT output file ([micrograph_ampl_defocus.plt](#)), in the headers of the output images ([micrograph_ampl_half_half](#)) or in the input=output MSA eigen-filtered amplitude images [micrograph_ampl_efilt](#)) but not in the headers of the micrograph images ([micrograph](#)), which should be CTF corrected. Call the command **HEADER** to take over the CTF/defocus parameters:

```
IMAGIC-COMMAND: headers

Options available           : takeover
Takeover options available : defocus/ctf
Input MSA-SUM or CTF file  : micrograph_ampl_efilt
Input=output (header) file : micrograph_filt_coarse
```

6. Before doing the CTF correction you first have to evaluate the CTF estimation done in **CTF-FIND** by comparing the amplitude images against the estimated theoretical amplitude images:

DISPLAY the "half_half" output images ([micrograph_ampl_half_half](#)). The left half of each image shows the MSA filtered amplitude image, the right half the estimated CTF. Compare how well the zeros match between them.

Use **DISPLAY** option **SELECT** to store the locations of the "bad" micrographs (for which the zeros in the "half_half" images do not match). "Bad" amplitude images are also the ones, which do not show any Thon rings. The selected locations are stored in a PLT file ([bad_micrographs.plt](#)):

```
Change settings (MULT,HOR,VER,SURVEY...) [NO]: select
Output (PLT) file for loc#s                :
                                             bad_micrographs
...
Change settings (MULT,HOR,VER,SURVEY...) [NO]: no
```

Select a "bad" image by clicking into the image on the screen. To cancel this selection, click into the image once more. A red border indicates that the image is selected; a black box indicates that the selection was cancelled. To get the next series of images click into the green NEXT button in the upper left corner of the display window. To stop option **SELECT** click into the red STOP button.

You can also write down the "bad" amplitude images and use the **INTERACTIVE** option in the subsequent command **EXCLUSIVE-COPY**.

"BAD" MICROGRAPH IMAGES:

7. If you found bad micrographs exclude these bad images with command **EXCLUDE-IMAGE**. Input is the file with the micrographs:

```
IMAGIC-COMMAND: exlude-im

What should be copied                : 2d
Excusive copy operation              : EXCLUDE
Input file, NO loc#s                 : micrograph_filt_coarse
Output file, image loc#s             : good_micrographs

  The location numbers you specify will be
  EXCLUDED from the output file

Source of image loc#s                 : plt  or INTERACTIVE (see above)
PLT file containing loc#s            : bad_micrographs
```

8. Use **CTF2D-FLIP** to correct for the CTF. Inputs to **CTF2D-FLIP** are the file with the "good" micrographs (**good_micrographs**):

```
IMAGIC-COMMAND: ctf-flip
Use classification results to read CTF : no
Original images/patches, imageloc#s   : good_micrographs
Output file name, NO loc#s             : micrograph_flip
ALL defocus and EM values in headers   : yes
Aperture of objective                  : 70           you have to know
Full output                             :
```

NOTE:

You can alternatively estimate the defocus parameters with **CTFFIND3** (Linux or Mac OS X).

CTFFIND3 is a program provided by the Grigorieff lab (<http://emlab.rose2.brandeis.edu/ctf>) and is licensed under the terms of the GNU Public License version 3 (GPLv3). It is not part of **IMAGIC** but can be used within the **IMAGIC** environment with command **CTFFIND3**, if the **CTFFIND3** program is implemented in the FREALIGN directory of **IMAGIC**.

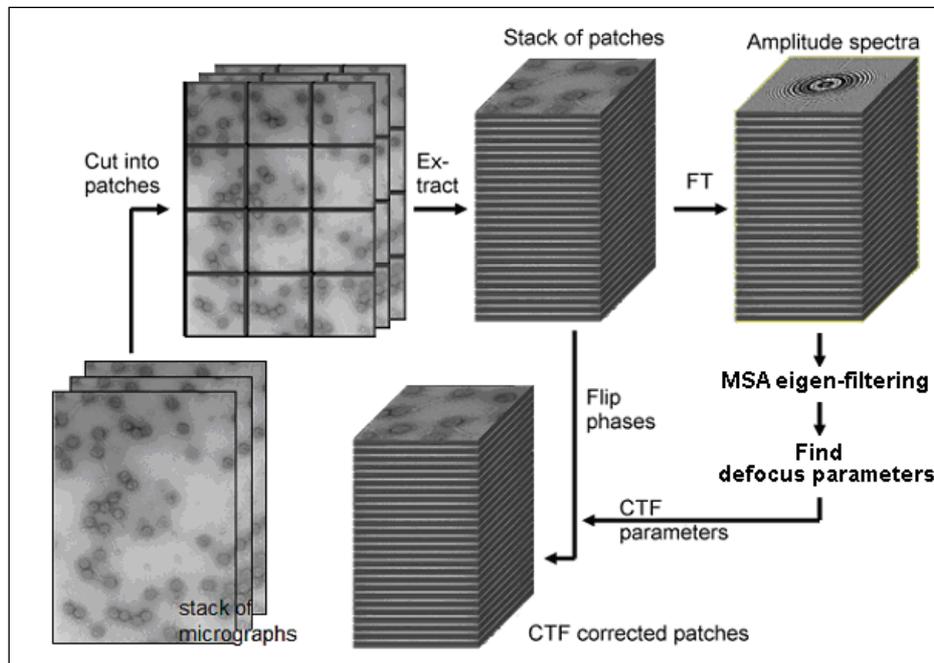
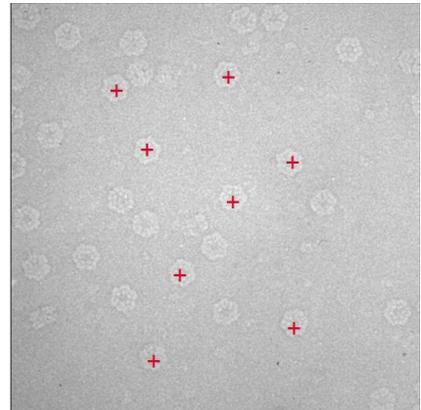


Fig. 9: Automatic CTF correction using MSA eigen-filtering

8. Particle Picking

After CTF correction you are ready to select the particles from the stack of CTF-corrected micrographs using command **PICK-PARTICLES**.

Particle selection can be carried out in several ways, using reference free VARIANCE/MODULATION picking or using correlation (CCF_MATCHING) to a given (set of) reference(s).



8.1. Modulation Picking

Particle selection using modulation can be very useful if

- you have not (much) a priori knowledge of the particles you are trying to select
- you want to avoid introducing bias by template matching to references (i.e. using correlation functions CCF/MCF)

The variance image and the modulation image are very closely related (refer to the lectures), and although you will be using the modulation image here, the same ideas and procedures can be followed when using the variance image.

Here, we will pretend worm hemoglobin is a completely new particle, which you have never seen before. You start to select particles from a few CTF corrected micrographs ([micrograph_flip](#)).

1. **DISPLAY** the micrographs. Some of the particles are very close to each other and there are some contaminating ice crystals. Think about the number of decisions you would have to make if you were picking this manually: do you keep particles that are touching each other? How close is "touching"? What makes a particle "good" or "bad"? Are you sure you are noticing all the views? And importantly, think about how long it would take you.
2. To calculate a meaningful modulation image, you will need to have a rough idea of the dimensions of the particle you are looking for. Using option **COORD** of the **DISPLAY** command, measure the largest and smallest "diameters" of the views you can see. You only need very rough measurements here so don't spend too long on it.

YOUR NOTES:

Dimension small:

Dimension large:

3. Now pick particles using the **PICK-ALL-PARTICLES** command. Use the **MODULATION_IMAGE** mode of particle detection.

Choose to store the peaks functions to file, so you can look at them later. It will not be necessary to extract boxed particles for now, so answer **NO** to that question.

Also, answer **NO** to "Set modulation parameters manually". This is only useful if you are familiar with the steps to calculate the modulation image.

Next, the command needs to know the band-pass filter parameters (refer to the related lecture).

Then, for "Max. number of particles per location wanted", enter your estimate of how many particles are to be found in each patch (try out).

The "Min. distance between peaks" parameter defines how close two picks are "allowed" to be to each other.

```
IMAGIC-COMMAND: pick
```

```
Mode of particle detection           : modulation
Input raw images file, loc#s       : micrograph_flip
Store pick functions to file       : yes
Output modulation peaks file      : mod_var
Output (PLT) file with peaks      : coord_mod
Extract found particles            : yes
Output single particles file      : particles_mod
Output particle image size        : 128,128
Set filter params in Fourier space : no
Unit of dimension measurements    : pixels
```

```
Typical (lower) size of object      : 55          your choice
Typical (upper) size of object     : 75          your choice
Max. number of particles per loc   : 100         your choice
Max. overall number expected       : 0
Minimum distance between peaks     : 32
Minimum distance X,Y from edges    : 32,32
Full output of all peak parameters :
```

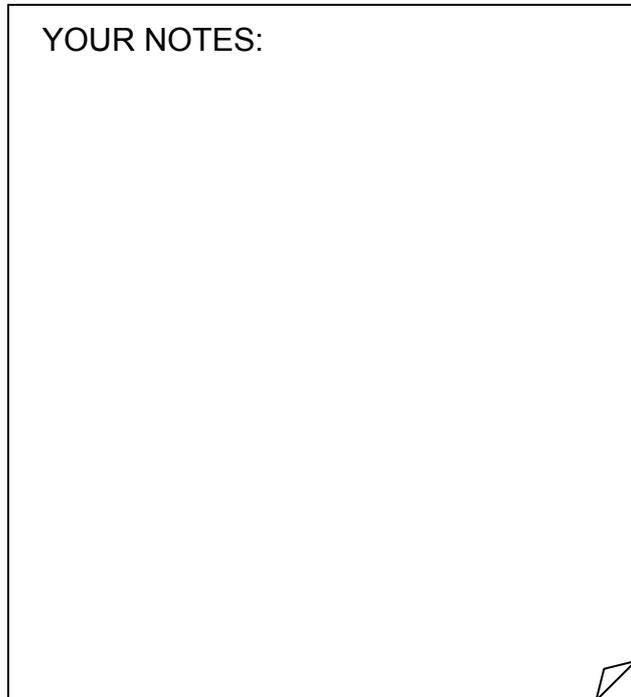
4. Check the results: **DISPLAY** the micrographs (location by location), and use option **PLOT** to display the coordinates **PICK-ALL-PARTICLES** found (`coord_mod.plt`).

```
IMAGIC-COMMAND: disp
Input image file, image loc#s      : micrograph_flip
...
Current DISPLAY settings:
...
NO_CHANGES(=DISPLAY), SETTINGS, OPTIONS [NO]: location
Input location number wanted       : 1  location wanted
                                       ONE SINGLE number
...
Current DISPLAY settings:
...
NO_CHANGES(=DISPLAY), SETTINGS, OPTIONS [NO]: plot
Plot (PLT) file to be displayed    : coord_mod
Current DISPLAY settings:
...
NO_CHANGES(=DISPLAY), SETTINGS, OPTIONS [NO]: no
```

Did it miss any particles (false negatives)? Did it choose things, which were not particles (false positives)? Notice how the edges of ice crystals or carbon foil are picked. How accurate would you say the particle selection was?

5. Re-do **PICK-ALL-PARTICLES** and play around with other input parameters. Check the results (refer to 4.) and find out how the results change.

YOUR NOTES:

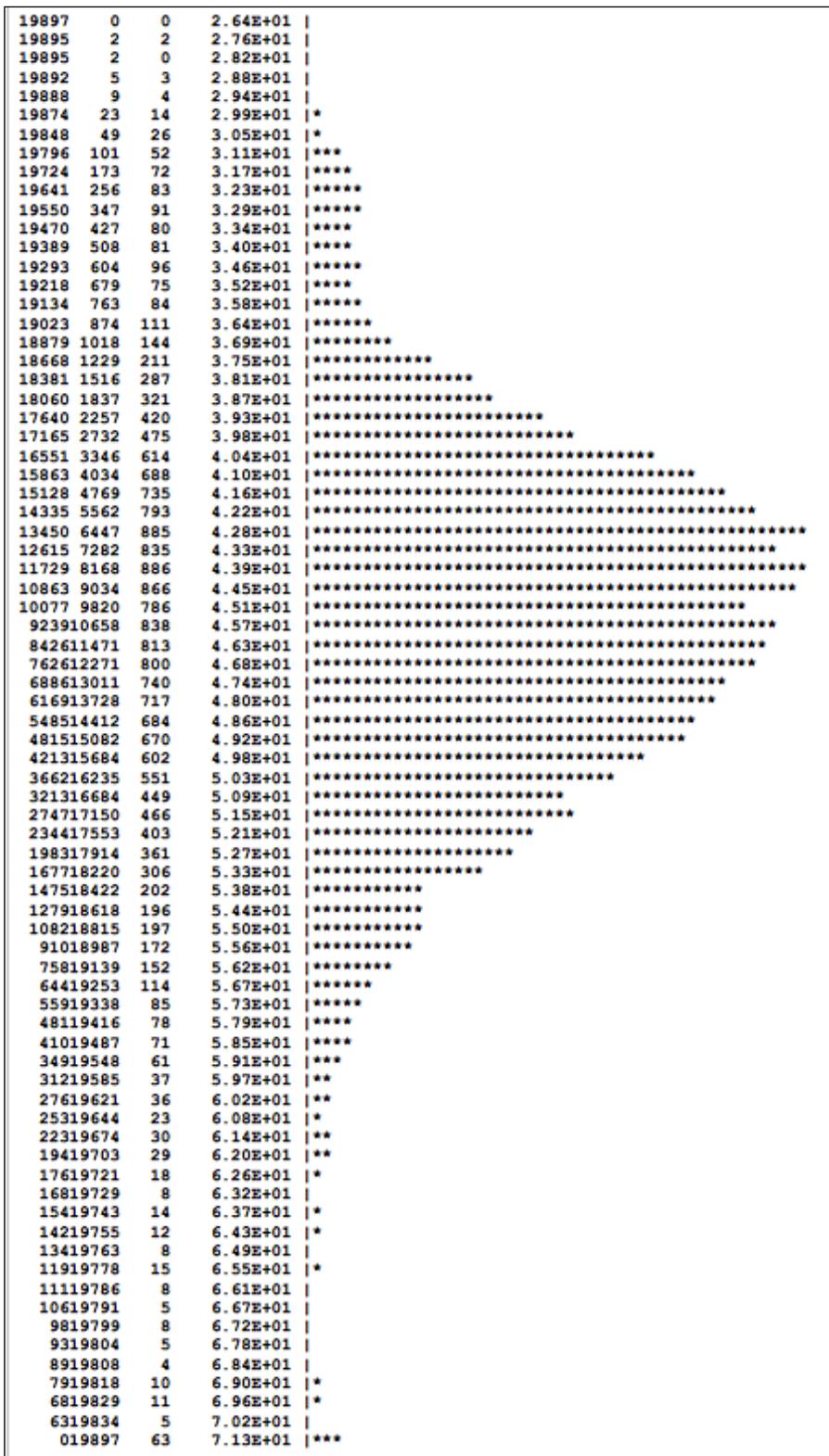


8.2. Extract "good" Images after Modulation Picking

1. **PICK-PARTICLES** usually also picks some unwanted objects, which you should remove. First, generate a histogram of the cross-correlation coefficients (**CCC**) with the command **HEADERS**:

```
IMAGIC-COMMAND: head
Option available           : histogram
Histogram options available : ccc
Number of bins for histogram : 56
Width of histogram        : 79
Input header file         : particles_mod
```

The resulting histogram should more or less look like this:



The command **PICK-PARTICLES** dumps the picked particles in decreasing order of the correlation coefficient. The histogram curve shown by **HEADERS** therefore has a bump-like structure. The first images usually contain edges

and relate to the upper part of the histogram curve. The last images normally contain ice particles and relate to the lower part of the histogram curve. The “good” images can therefore be easily removed. Check the histogram to find out the locations numbers (second row of numbers) where to cut the histogram:

19897	0	0	2.64E+01	
19895	2	2	2.76E+01	
19895	2	0	2.82E+01	
19892	5	3	2.88E+01	
19888	9	4	2.94E+01	
19874	23	14	2.99E+01	*
19848	49	26	3.05E+01	*
19796	101	52	3.11E+01	***
19724	173	72	3.17E+01	****
19641	256	83	3.23E+01	*****
19550	347	91	3.29E+01	*****
19470	427	80	3.34E+01	****
19389	508	81	3.40E+01	****
19293	604	96	3.46E+01	*****
19218	679	75	3.52E+01	****
19134	763	84	3.58E+01	*****
19023	874	111	3.64E+01	*****
18879	1018	144	3.69E+01	*****
18668	1229	211	3.75E+01	*****
18381	1516	287	3.81E+01	*****
18060	1937	321	3.87E+01	*****
17640	2257	420	3.93E+01	*****
17165	2732	475	3.98E+01	*****
16551	3346	614	4.04E+01	*****
15863	4034	688	4.10E+01	*****
15128	4769	735	4.16E+01	*****
14335	5562	793	4.22E+01	*****
13450	6447	885	4.28E+01	*****
12615	7282	835	4.33E+01	*****
11729	8168	886	4.39E+01	*****
10863	9034	866	4.45E+01	*****
10077	9820	786	4.51E+01	*****
923910658	838		4.57E+01	*****
842611471	813		4.63E+01	*****
762612271	800		4.68E+01	*****
688613011	740		4.74E+01	*****
616913728	717		4.80E+01	*****
548514412	684		4.86E+01	*****
481515082	670		4.92E+01	*****
421315684	602		4.98E+01	*****
366216235	551		5.03E+01	*****
321316684	449		5.09E+01	*****
274717150	466		5.15E+01	*****
234417553	403		5.21E+01	*****
198317914	361		5.27E+01	*****
167718220	306		5.33E+01	*****
147519432	202		5.38E+01	*****
127918618	196		5.44E+01	*****
108918815	197		5.50E+01	*****
91018987	172		5.56E+01	*****
75819139	152		5.62E+01	*****
64419253	114		5.67E+01	*****
55919338	85		5.73E+01	****
48119416	78		5.79E+01	****
41019487	71		5.85E+01	****
34919548	61		5.91E+01	***
31219585	37		5.97E+01	**
27619621	36		6.02E+01	**
25319644	23		6.08E+01	*
22319674	30		6.14E+01	**
19419703	29		6.20E+01	**
17619721	18		6.26E+01	*
16819729	8		6.32E+01	
15419743	14		6.37E+01	*
14219755	12		6.43E+01	*
13419763	8		6.49E+01	
11919778	15		6.55E+01	*
11119786	8		6.61E+01	
10619791	5		6.67E+01	
9819799	8		6.72E+01	
9319804	5		6.78E+01	
8919808	4		6.84E+01	
7919818	10		6.90E+01	*
6819829	11		6.96E+01	*
6319834	5		7.02E+01	
019897	63		7.13E+01	***

2. **DISPLAY** the related images and find out the start and end location numbers of the "good" images.

YOUR NOTES:

First good location:

Last good location:

3. Extract the good images with the command **EXTRACT-IMAGES**:

```
IMAGIC-COMMAND: extract-im

What should be copied           : 2d_images
Exclusive copy operations       : EXTRACT
Input file                      : particles_mod
Output file                    : particles_mod_2
Source of image locations       : interactive
Location numbers wanted        : 1550-23098      your choice
Numbers wanted                 : all
```

NOTE:

You can use other parameters to exclude "bad" images. If time is restricted you can skip the following SORT and EXCLUDE parts and continue with **BAND-PASS-FILTER** (9.)

4. To calculate the image statistics (average density, minimum, maximum, sigma) call command **SURVEY-DENSITIES**. Use option **UPDATE_HEADER** to store the results in the image headers.

```
IMAGIC-COMMAND: survey

Mode of survey                  : 2d_local
Mode of output                  : update_header
Input file                     : particles_mod_2
```

5. Subsequently, sort the images using **SIGMA** as criterion:

```
IMAGIC-COMMAND: sort-image
What should copied           : 2d
Exclusive copy operation     : SORT
Input file                   : particles_mod_2
Output file                  : particles_mod_3
Source of SORT values       : header
Criteria for SORT           : sigma
Sort Up or DOWN             : up
How many of sorted images wanted : 0                                0: all
```

6. Again look at the histogram in **HEADERS** option **HISTOGRAM** now using **SIGMA** and check if there are bad images (refer to 1.).

YOUR NOTES:

First good location:

Last good location:

7. As before (3.) extract the good images with **EXCLUSIVE-COPY**.
8. Further criteria to exclude bad images are: **AVDENS** (average density), **MINDENS** (minimal density value) and **MAXDENS** (manimal density value). Sort (5.) and extract as before (3). Note that usually **SIGMA** is the most important parameter at this stage of the analysis. Finally your best images will be stored in file **particles_mod_best**.

9. At the current state of the analysis we do not yet need references with fine details: the low frequencies (overall shape etc.) are more important. So you will first apply a **BAND-PASS-FILTER** removing high frequencies:

```
IMAGIC-COMMAND: band-pass

Mode of operation           : BANDPASS
Input file, image loc#s    : particles_mod_best
Output file, image loc#s   : particles_mod_filt
Low frequency cut-off      : 0.05           your choice
Remaining low transmission : 0.             your choice
High frequency cut-off     : 0.6           your choice
ASQ filter the images      : no
```

10. After modulation picking the particles (`particles_mod_best`) are not well centred. Call command **CENTER-IMAGE**:

```
IMAGIC-COMMAND: center-image

Input file, image loc#s    : particles_mod_filt
Output file, image loc#s   : particles_mod_cent
Options for centering      :
```

Use option **SELF_CENTER** if the particles are already well centred:

```
Options for centering      : self
Correlation functions available : ccf
Max shift (pixels or as fraction) : 0.5
Number of centering iterations : 3
```

Or, if the particles are not well centred, **ROTATE_SELF**. Use **?** to get more information:

```
Options for centering      : rotate
Full output                : no
```

8.3. MSA Classification to get References for Correlation Picking

1. Subsequently MSA classify the best particles. This is the same procedure as you did with the amplitudes. Remember: **MSA-RUN** is time consuming.

```
IMAGIC-COMMAND: msa-mask

Mode of operation                : circular
Input header file (no loc#s)    : particles_mod_cent
Output filename, image loc#s    : particles_msamask
MSA mask radius                 : 0.5

IMAGIC-COMMAND: msa-run

Choose mode of operation        : fresh
MSA distance                    : modulation
Input (= output) aligned "images" : particles_mod_cent
Input MSA mask file             : particles_msamask
Eigenimages output file, NO loc#s : particles_mod_eigen
Use default answers for other options : yes
Number of iterations            : 100
Number of eigenimages           : 50
Rootname for results files, no ext. : particles_mod_msa

IMAGIC-COMMAND: msa-classify

Input to be classified          : images
Input (= output) file (treated by MSA) : particles_mod_cent
Percentage of images to be ignored : 0
Active eigenimages for classification : 50
Use default classification options : yes
What number of classes do you wish : 200          your choice
Rootname of output files        :
                                particles_mod_classify
```

```
IMAGIC-COMMAND: msa-sum

Input images to be summed      : particles_mod_cent
Rootname of MSA-CLASSIFY results : particles_mod_classify
Output class averages         : particles_mod_classsums
Downweight small classes      : yes
Fract. of worst members to ignore : 20
Mode of summing statistics    : none
```

2. May be or may be not, you want to sort the class averages by the overall quality of the classes (command **SORT-IMAGE** with options **HEADER** and **OVERALL_QUALITY_CLS**).
3. Visualize the class averages (**particles_mod_classsums**) with the command **DISPLAY** and select some (close-to-) top views (circular like shape), some intermediate views and also some (close-to-) side views (rectangular like shape). The chosen particles **MUST** be well centred.

LOCATION NUMBERS:

Intermediate views:

Side views:

Top views:

4. Extract the chosen images with the command **EXCLUSIVE-COPY**:

```
IMAGIC-COMMAND: extract-im

What should be copied           : 2d
Excusive copy operation         : EXTRACT
Input file, NO image loc#s     : particles_mod_classums
Output file, image loc#s       : particles_ref
Source of image locations      : interactive
Location(s) of image(s) wanted :           your location numbers
                                (note: the syntax is:
                                12;1-5;91;76-81)
Numbers wanted                  : all
```

The chosen images (which will be your references) are stored in the output file **particles_mod_ref**.

Subsequently, the references have to be averaged rotationally. Use the command **AVERAGE-ROTATIONAL**:

```
IMAGIC-COMMAND: average-rotat

Input file, images locations    : particles_ref
Mode of output                  : image
Output file, image loc#s       : particles_ref_avrot
```

8.4. Prepare Micrographs and References for Correlation Picking

1. Next you will re-do particles picking now using the chosen typical views as references. Remember: you are NOT interested in correlations of fine details. To find the particles the low frequencies describing the shapes, for example, are the important once. So we low-pass filter both, reference and micrographs reducing the high frequencies with a band-pass filter. Use command **PREAPRE-IMAGE**:

```
IMAGIC-COMMAND: prep-im
Mode of operation           : PREPARE_IMAGES
Input file, image loc#s    : particles_ref_avrot
Output file, image loc#s   : particles_ref_blobby
Low frequency cut off      : 0.1
Remaining low-freq. transm. : 0.
High frequency cut off     : 0.2
Mask radius, drop-off (0: no mask) : 0.5,0.05
Desired new sigma          : 10
Remove (dust) outliers     : no
Invert the image densities  : no

IMAGIC-COMMAND: prep-im
Mode of operation           : PREPARE_IMAGES
Input file, image loc#s    : micrograph_flip
Output file, image loc#s   : micrograph_blobby
Low frequency cut off      : 0.05
Remaining low-freq. transm. : 0.
High frequency cut off     : 0.1
Mask radius, drop-off (0: no mask) : 0
Desired new sigma          : 10
Remove (dust) outliers     : no
Invert the image densities  : no
```

8.5. Correlation Picking

1. Run **PICK-ALL-PARTICLES** now using the option **CCF_MATCHING**:

```
IMAGIC-COMMAND: pick

Mode of particle detection           : ccf
Input raw images file, loc#s       : micrograph_blobby
Store peaks functions to file      : no
Output (PLT) file with peaks       : coord_ccf
Extract found particles            : no
Input reference file               : particles_ref_blobby
Ref. already rotationally symmetric: yes
Max. number of particles per loc   : 100          your choice
Max. overall number expected       : 0
Minimum distance between peaks     : 32
Minimum distance X,Y from edges    : 32,32
Full output of all peak parameters : no
```

2. You used the blobby micrograph images (**micrograph_blobby**) but, of course, the particles will be extracted from the original high-resolution micrograph images (**micrograph_flip**)!!!! Use the command **CUT-IMAGE**, option. Using the **CORRELATION** option, the picked particles are better centred than with option **MODULATION**. Therefore, the image size of the boxed particles can now be smaller (**96,96**). The input coordinates file is the PLT file generated in **PICK-ALL-PARTICLES** (**coord_ccf.plt**).

```
IMAGIC-COMMAND: cut-im

Mode of operation                   : aperiodic
Input file, image loc#s            : micrograph_flip
Output file, image loc#s          : particles_ccf
Output image dimensions X,Y       : 96,96
Coordinates (plt) file            : coord_ccf
```

3. **DISPLAY** and check the extracted particles (**particles_ccf**).

- The extracted particles are sorted by CCC (cross correlation coefficient). This can be used to exclude “bad” particles. As was done in modulation picking (8.2.) create a histogram of the CCC values with command **HEADERS**:

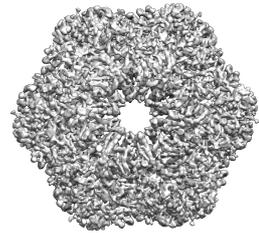
```
IMAGIC-COMMAND: head
Option available           : histogram
Histogram options available : ccc
Number of bins for histogram : 56
Width of histogram        : 79
Input header file         : particles_ccf
```

YOUR NOTES:

First good location:

Last good location:

- Extract the “good” images as you did in chapter 8.2. (1.-3.)
- In “real science” you may want to also use some other criteria to exclude “bad” images. Such criteria are **SIGMA**, **AVERAGE**, **MINDENS**, and **MAXDENS**.). Note that **SIGMA** usually is the most important parameter here.
- If you want to use **SIGMA** call the command **SURVEY** with option **UPDATE_HEADER**. Sort the images by **SIGMA** using **SORT-IMAGES** and create a histogram with the command **HEADER**, option **HISTOGRAM**. As before use extract the best images (chapter 8.2. (1.-3.)) now using index **SIGMA**.



9. Pre-Treatment

1. The file `particles_ccf_best` finally contains the “best” picked (boxed) particles. Very often at this stage of the analysis you would call the command `PREPARE-IMAGES` to pre-treat the boxed particles (`particles_ccf_best`). As you already know `PREPARE-IMAGES` band-pass filters, normalizes and zero-float the data set. And finally, it masks the images.
2. You have already applied `PREPARE-IMAGES` onto the micrographs. So this pre-treatment is no more needed here. But it usually is a good idea to normalize and mask the particle image (`particles_ccf_best`). Call the command `PRETREAT-IMAGES` with options `NORM_VARIANCE` and `MASK`:

```
IMAGIC-COMMAND: pretreat-im
Mode of operation           : PRETREAT
Please specify option       : norm
Type of variance mask     : circle
Input file, loc#s         : particles_ccf_best
Output file, loc#s        : particles_filt
Mask radius,drop-off      : 0.85,0.05
Desired new sigma         : 10
```

10. Alignment-by-Classification

Alignment by classification is a method by which you can obtain classes by aligning the particles to references generated from the data set itself. Neither external references nor references generated from 3-D volumes are used at this stage. It consists of the following steps:

- Centring (chapter 11)
- MSA classification (chapter 12)
- Multi-reference alignment (MRA) of the particles against the (selected) class averages (chapter 13).

11. Centring

The first step in alignment-by-classification is to do centring. This means all the images will be aligned translationally (but not rotationally) to the total sum of the data set. The command to do this is **CENTER-IMAGE**. In this command, you specify how many rounds (iterations) of summing and aligning against the total sum you would like to perform. Usually, after 4 rounds the alignment stops changing (converges). Try the following settings:

```
IMAGIC-COMMAND: center-image
Input file, image loc#s      : particles_filt
Output file, image loc#s    : particles_cent
Options for centering        : totsum
Correlation functions available : ccf
Max shift (pixels or as fraction) : 0.2
Number of centering iterations : 4
Options to filter the total sum : low
Halfwidth value for low-pass filter : 0.1
```

12. Multivariate Statistical Analysis (MSA) Classification

1. First, you need to create a mask for MSA (`msamask`) that will specify the area of the image used to calculate the statistics. The MSA classification will be based on this area alone. As this is the first classification and you do not know much about your object yet, you will use a large circular mask taking almost the entire image area into account for MSA classification. In later runs you might want to use a different mask.

Use the command `MSA-MASK` to create this mask. Choose option `TOTAL_SUM` option, with a radius of `0.7`:

```
IMAGIC-COMMAND: msa-mask

Mode of operation           : total_sum
Input file name, image loc#s : particles_cent
Output file name, image loc#s : msamask
Additional mask             : circular
Circular mask radius        : 0.7
```

2. `DISPLAY` the mask (`msamask`) and check if the radius of the circular was chosen correctly.
3. Now, run `MSA-RUN` on the centred particles using this mask. Bear in mind that this will take some time to run:

```
IMAGIC-COMMAND: msa-run

Choose mode of operation   : fresh
MSA distance                : modulation
Input (= output) aligned "images" : particles_cent
Input MSA mask file        : msamask
Eigenimages output file, NO loc#s : eigenim
Use defaults for other MSA options : yes
Number of iterations       : 150
Number of eigenimages      : 69      "real science": more
Rootname for results files, no ext. : msa
```

4. In another command window you can use option **WATCHDOG** of the command **DISPLAY** to continuously display the eigenimages (**eigenim**) as they are being updated throughout the **MSA-RUN** iterations.
5. When **MSA-RUN** had finished look for symmetry components in the eigenimages. What symmetry eigenimages can you observe? What information can you extract from them about the symmetry of the molecule being studied?

The first eigenimage always shows the average of all images. If your particles are symmetric you can find this point-group symmetry in the next eigenimages. The images were not yet rotationally aligned, so the 2nd and 3rd (or higher) eigenimages are rotated to each other (like a sine and cosine wave). In case of cyclical symmetry, both usually clearly show an n-fold (cyclical) symmetry.

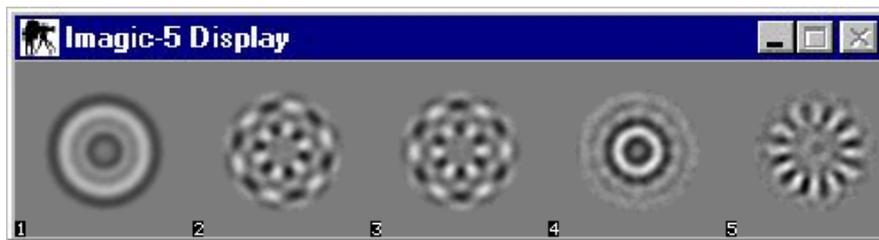


Fig. 11: First eigenimages of the (rotational) unaligned worm hemoglobin data set. (Remember that worm hemoglobin has D6 symmetry)

Once **MSA-RUN** has finished you can classify the particles.

6. The classification is performed with the command **MSA-CLASSIFY**. The number of classes you choose to create is related to the average number of images per class you would like. You can play with this value to see how the quality of the classes is affected. Ideally, you would have as few members per class as possible whilst still obtaining high contrast class averages.

```
IMAGIC-COMMAND: msa-classify

Input to be classified                : images
Input (= output) file (treated by MSA) : particles_cent
Percentage of images to be ignored    : 0
Active eigenimages for classification : 69 "real science": more
Use default classification options    : yes
What number of classes do you wish   : 500      your choice
Rootname of output files              : classify1
```

7. Once the classification has finished, average all the particles that belong to the same class using the command **MSA-SUM**.

```
IMAGIC-COMMAND: msa-sum

Input images to be summed             : particles_cent
Rootname of MSA-CLASSIFY results      : classify
Output class averages                 : classsums
Downweight small classes               : yes
Fraction of worst class members to ignore : 20
Mode of summing statistics             : none
```

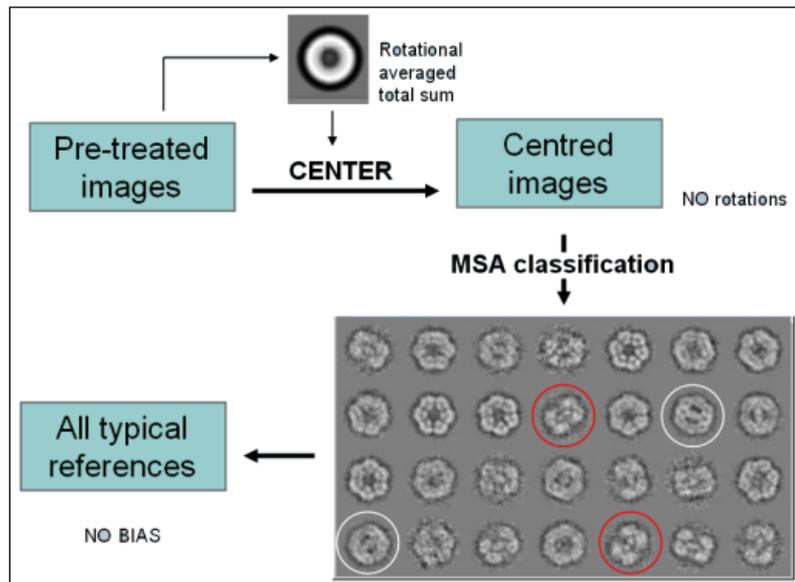


Fig. 12: Alignment by Classification

8. **DISPLAY** the class averages created in **MSA-SUM**.

REMEMBER:

You should **DISPLAY** the class averages using the same grey value for all images. Either use option **GREYVALUE** with **SURVEY** and **GLOBAL** or **GREYVALUE** with option **INTERACTIVE**.

You will see that there are a lot of "good" showing particles views with high resolution of also a number of "bad" class averages showing classes with low resolution.

9. Good criterions to exclude "bad" the classes are the number of members per class and the overall class quality (refer to chapter (8.3))

First sort by number of class members (**NUMBER_OF_CLASS_MEMBERS**) with command **SORT-IMAGE**:

```
IMAGIC-COMMAND: sort-image
What should copied           : 2d
Exclusive copy operation     : SORT
Input file                   : classsums
Output file                  : classsums_sort_num
Source of SORT values        : header
Criteria for SORT            : number
Sort UP or DOWN              : down
How many of sorted images wanted : 0                                0: all
```

10. Again look at the histogram of the number of class members in **HEADERS** option **HISTOGRAM** now using **INDEX**, **LABEL**, **NUMCLS** and check if there are bad classes (refer to chapter (8.3)). **DISPLAY** the sorted class averages (**classsums_sort_num**) and find out the "bad" classes.

YOUR NOTES:

Last class location with enough members:

11. As usual, extract the good class averages with the command **EXTRACT-IMAGE**.
12. Next use the command **SORT-IMAGE** to once more sort the good averages, now by classification over all quality (refer to 9.). As before (10.) create a histogram (**HEADERS** with options **HISTOGRAM** and **OVERALL_QUALITY**) and check if there are bad classes.

YOUR NOTES:

Last class location with good quality:

13. As before (3.) extract the good classes with **SORT-IMAGES**. When displaying (**DISPLAY**) the extracted images.

NOTE:

Depending on you data set you will continue with multi-reference alignment (chapter 13) or immediately continue with angular reconstitution and 3-D reconstruction (chapter 16).

In a "real science" analysis of worm hemoglobin you would usually skip the multi-reference part here, because the particle is well centred and highly symmetric. But here in the course you will continue with multi-reference alignment.

14. .

YOUR NOTES:

13. Multi-Reference-Alignment (MRA)

1. You need to select a number (let's say five) of your "best" class averages, which you believe to be characteristic of your data set. You will hopefully be able to select a number of different views, and so it is a good idea to take the "best" images from each of the different views to use as references for your first alignment.

The term "best" is obviously subjective but "by eye" is the best way to select the class averages. Try and select views which contain the most internal detail, also avoid class averages which are very round, usually if a class average contains artefacts or looks "strange" to you, then you shouldn't take it. It can cause some problems if you do not have a reference corresponding to each type of view. Almost all data sets have some views, which are very common, and some views, which are much rarer. If you miss references for the rare views at this stage, it can be difficult to regain them at a later stage, and so it is important to bear this in mind when choosing references. But you also should not select too many references at this stage of the analysis. Create a PLT file containing the locations of the best class averages ([classums_best.plt](#)). You can use a text editor or **DISPLAY** with option **SELECT**.

2. Extract the chosen best class averages using the command **EXTRACT-IMAGE**:

```
IMAGIC-COMMAND: extract-image

What should be copied           : 2d
Mode of copy operation          : EXTRACT
Input file, NO loc#s           : classums
Output file, image loc#s        : classums_best
Source of image locations        : plt
Plt file containing image loc#s : classums_best
```

3. It may or may be not a good idea at this stage to use the command **PREPARE-MRA-REFERENCES**, which will align the references to each other.

```
IMAGIC-COMMAND: prep-mra-ref
Alignment modes available      : both
Start option                   : rotation_first
Correlation functions available : ccf
Reference input file           : classums_best
Use contours on the references  : no
Radius for circular mask       : 0.8
MRA references output file     : mraref_1
Density for thresholding.      : -999
Max shift (pixels or as fraction) : 0.1
Min, max rotation angle       : -180,180
Precision for rotational alignment : high
Min,max radius for rot. alignment : 0,0.7
Create mirrors of references    : no
Max. number of alignment iterations : 5
Full output of all parameters   : no
```

4. **DISPLAY** the aligned references (**mraref_1**) and check if they are well aligned. If not use the non-aligned references (**classums_best**) in the subsequent multi-reference alignment.

NOTE:

Multi-reference alignment is very time consuming. During this practical, you should not run it interactively for all images.

To get an idea how a multi-reference alignment is calculated you will work with only a number of input images.

To align all images you will use a batch job running over night or during lectures.

5. Run **MULTI-REFERENCE-ALIGNMENT**. First, you will use only 1000 input images:

```
IMAGIC-COMMAND : m-r-a

MRA options:                : fresh
4D options:                  : all_references
Methods available           : align
Alignment modes available   : both
Start options available     : rotation_first
Correlation functions available : ccf
Input file, loc#s          : particles_cent,1,1000
Output file, loc#s         : particles_mra_1
Original (pretreated) file, loc#s : particles_filt
Reference file, loc#s       : mraref_1
Option to filter the reference(s) :
```

NOTE:

There are a number of options to filter the reference(s). Very often you can use **NO_FILTER**, but especially for elongated particles it can be useful to apply a **LOWPASS_FILTER**. To avoid over fitting it can be helpful at a later stage of the analysis to use a sharp cut-off of the high frequencies (**CUT_OFF_HIGH_FREQUENCIES**).

```
Option to filter the reference(s) : no_filter
Max shift (compared to originals) : 0.2
Max shift (during this alignment) : 0.1
Min,max rot. angle
  (compared to originals)         : -180,180
Min,max rot. angle
  (for this alignment)            : -180,180
```

```
Precision for rotational alignment : high
Min,max radius for rot alignment  : 0,0.7
Number of alignment iterations    : 5
Full output of all parameters     : no
```

6. **DISPLAY** the aligned images (`particles_mra_1`). Especially have a look at the side views and see that they are aligned to each other now.
7. Now you should align ALL images. As already mentioned this is very time consuming and the alignment should run overnight or during lectures.
8. Create a script file using the **MODE** commands. Start accumulating commands with **MODE-ACCUMULATE**.

```
IMAGIC-COMMAND : mode-acc
```

and create a script file for **MULTI-REFERENCE-ALIGNMENT**: Do not forget to use all images now.

```
IMAGIC-COMMAND (ACC.): m-r-a

MRA options:                : fresh
4D options:                  : all_references
Methods available           : align
Alignment modes available   : both
Start options available     : rotation_first
Correlation functions available : ccf
Input file, loc#s           : particles_cent  no loc#s
Output file, loc#s          : particles_mra_1
Original (pretreated) file, loc#s : particles_filt
Reference file, loc#s       : mraref_1
Option to filter the reference(s) : no_filter
Max shift (compared to originals) : 0.2
Max shift (during this alignment) : 0.1
```

```
Min,max rot. angle
  (compared to originals)           : -180,180
Min,max rot. angle
  (for this alignment)              : -180,180
Precision for rotational alignment  : high
Min,max radius for rot alignment   : 0,0.7
Number of alignment iterations     : 5
Full output of all parameters      : no
```

Note that the command is not yet running. If wanted to can accumulate more commands (the MSA commands of chapter 14, for example).

9. To finally create the script file give **MODE-STOP**:

```
IMAGIC-COMMAND (ACC.): mode-stop

Filename for batch/script file [bigjob] : my_script

...

                                some information
                                and suggestions are
                                printed here

END OF IMAGIC RUN

my_norebook>
```

10. Now you can run the script file ([my_script](#)). Read the related suggestions printed on the monitor. The way on how to start a script file depends on your operating system.

REMEMBER:

MODE-ACCUMULATE:

Start accumulating commands

MODE-STOP:

Stop accumulating commands and finalise the script file

MODE-ABORT:

In command **MODE-ACCUMULATE** abort everything and do not create any script file.

14. MSA Classification

1. After the alignment has completed, you should run a new round of MSA classification (**MSA-RUN** / **MSA-CLASSIFY** / **MSA-SUM**) on the aligned data set. The images are aligned, so you can choose a smaller number of classes.

Usually, it is helpful to use **MODE-ACCUMULATE** to accumulate the three mentioned commands and run the related script file.

```
IMAGIC-COMMAND: mode-acc
IMAGIC-COMMAND (ACC.): msa-run
Choose mode of operation           : fresh
MSA distance                       : modulation
Input (= output) aligned "images"  : particles_mra_1
...
IMAGIC-COMMAND (ACC.): msa-classify
Input to be classified              : images
Input (= output) file (treated by MSA) : particles_mra_1
...
IMAGIC-COMMAND (ACC.): msa-sum
Input images to be summed           : particles_mra_1
...
Output class averages               : classsums_mra_1
...
IMAGIC-COMMAND (ACC.): mode-stop
Filename for batch/script file [bigjob]: my_msa_job
...
```

2. Refer to the instructions given by **MODE-STOP** and start the script (the "batch job") over night or during lunch/afternoon/lectures. Do not forget to create a protocol (log) file so that you can check the output.

3. **DISPLAY** the class averages (`classums_mra_1`) and **SELECT** the good class averages (creating the output PLT file `classums_mra_1_best.plt`).
4. Extract these new "best" class averages using command **EXTRACT-IMAGE**. Input files are the new class averages (`classums_1_mra`) and the **PLT** file `classums_1_mra_best.plt`. Output will be `classums_1_mra_best`.

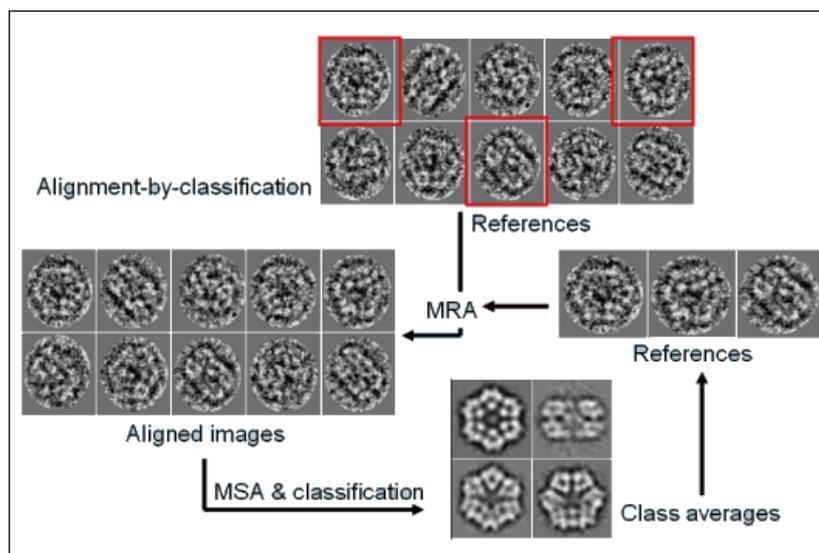


Fig. 13: Alignment by Classification and Multi-Reference Alignment

15. Iteration Cycle(s) of Multi-Reference Alignment and MSA Classification

As mentioned in the lectures you can iterate this MRA / MSA classification cycle (13) and (14) until you feel your class averages are of sufficient quality.

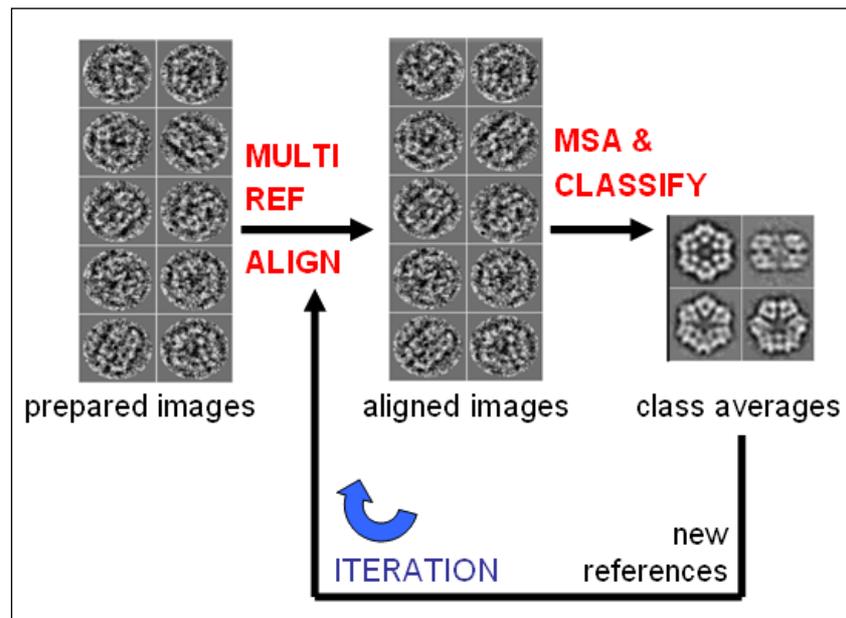
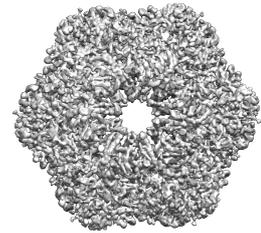


Fig. 14: Iteration of Multi-Reference Alignment and MSA Classification

Due to time constraints however, you will use the current class averages and continue straight on with angular reconstitution and 3-D reconstruction.



16. Angular Reconstitution - Initial Angular Assignment

Once you have selected the best class averages you need to find their relative orientation (Euler angles).

Worm hemoglobin is a molecule with D₆ (622) point-group symmetry. This high degree of symmetry makes the initial angular assignment much easier than with lower degrees of symmetry.

The images ([classums_mra_1_best](#)) should be well centred and masked tightly before you start with angular reconstitution. Use option **SELF** or **ROTATE_SELF** (refer to help in chapter 16):

```
IMAGIC-COMMAND: center-image

Input file, image loc#s      : classums_1_mra_best
Output file, image loc#s    : classums_1_mra_best_cent
Options for centering       :          SELF or ROTATE_SELF
...

IMAGIC-COMMAND: mask-image

Mode of operation:          : MASK_IMAGE
Mode of mask                : soft
Input file, image loc#s    : classums_1_mra_best_cent
Output file, image loc#s   : classums_1_mra_best_masked
Mask radius, drop-off      : 0.6,0.05          a tight mask
```

DISPLAY the centred images ([classum_mra_best_cent](#)) to check if the chosen centring option worked correctly.

16.1. Angular Reconstitution - Self Search

The next step is to check how well each single class average conforms to the given point-group symmetry of the particles.

The idea of option `SELF_SEARCH` in command `ANGULAR-RECONSTITUTION` is to sort the class-averages with the smallest residual for the given point-group symmetry to start up a 3-D reconstruction of the object. Each class-average image is examined exclusively with respect to itself. This option works best for highly symmetric point-groups like the D6 symmetry of worm hemoglobin.

Remember that the class average images (`classums_1_mra_best_masked`) should contain good classes of all typical views.

1. Call `ANGULAR-RECONSTITUTION`, option `SELF_SEARCH`:

```
IMAGIC-COMMAND: ang-rec

Point-group symmetry           : d6
Minimal stay-away from equator : 10
Option for angular reconst.    : self
Mode of output                  : update_header
Input (classum) images         : classums_1_mra_best_masked
Sinogram file, image loc#s     : none
Output sinecorr file, NO loc#s : none
ASQ filter the sinogram lines  : yes
Linear mask radius of sinograms: 0.6
Wanted angular increment       : 2
Full output of the results     :
```

At the end of the SELF_SEARCH calculations you will find a list like this:

```

=====
SELF_SEARCH Euler angles (sorted list)
=====

#          ERROR      LOC
#          (%)        #      ALPHA      BETA      GAMMA

1 :>    e.eeeee      nn      aaa.aaa      bbb.bbb      ggg.ggg
2 :>    e.eeeee      nn     -aaa.aaa      bbb.bbb      ggg.ggg
...

Average error of the set = e.eeeee %
    
```

2. Now **DISPLAY** the class averages (`classums_1_mra_best_masked`) with options **NAME** and **LEULER** so that the images are displayed with location number and Euler angles as just estimated (do not forget to later set this option back to **LOCATION!**).
3. For the subsequent 3-D reconstruction you should select three good class-averages with very different Euler angles:
 - a) First select two intermediate view. All views should look like intermediate views and should have "intermediate" Euler angles Beta (around 40°-60°)
 - a) Next select a (close-to-) side view, which are the ones that have rectangular like shape. Try to find such a view, whose Euler angle Beta is around 70-80°.
 - c) Finally select a (close-to-) top view (round shape) with a Beta angle around 10°.
 - d) Select 2-3 other intermediate or side views.

LOCATION NUMBERS:

Intermediate:

Close to side:

Close to top:

IMPORTANT NOTE:

SELF-SEARCH is NOT an Euler angles determination but rather a consistency check to see how well each single image conforms to the given point-group symmetry.

Here SELF-SEARCH is only used to find intermediate views with intermediate Euler angles, which you can use as input to command **ANGULAR-RECONSTITUTION** option **NEW_IMAGE**.

16.2. Angular Reconstitution - New projections

To assign Euler angles to all "best" class averages ([classums_1_mra_best_masked](#)) use option **NEW** of **ANGULAR-RECONSTITUTION**. In contrast to **SELF_SEARCH**, the Euler angles of each new image are now calculated in relation to the Euler angles of all images, which already have assigned Euler angles:

1. Call **ANGULAR-RECONSTITUTION**, option **NEW**:

```
IMAGIC-COMMAND: ang-rec
Point-group symmetry           : d6
Minimal stay-away from equator : 10
Option for angular reconst.    : new
Option of NEW                  : fresh
Input (classum) images, NO loc#s: classums1_mra_best_masked
Location numbers wanted        :          you choose:
                                three of our selected
                                class loc#s
                                seperated by ";"s
Output (selected) image file   : select_1
Output sinogram file, NO loc#s : sino
ASQ filter the sinogram lines  : yes
Linear mask radius for sinograms: 0.6
Output sinecorr file, NO loc#s : sinecorr
Wanted angular increment       : 2
Full output of the results     :
```

2. The output file ([select_1](#)) contains all selected images with Euler angles stored in their headers. This file is created because Euler angles were not assigned one-by-one to each image in the input file.
3. Remember that in option **SELF_SEARCH** each image was assigned an Euler angle exclusively with respect to itself. In contrast, option **NEW** Euler angles are assigned to each image with respect to itself and to all other previous images with assigned Euler angles.
4. After **ANGULAR-RECONSTITUTION** has finished **DISPLAY** the selected classsums ([select_1](#)) with option NAME and LEULER, in which case loc#s and the Euler angles will be printed. Check that the angles obtained make sense. The beta and gamma angles should be different and apart from each other for at least 50 degrees. If all Beta angles are the same (usually close to 90° - ("Minimal stay-away from equator") your three class averages are too similar and you should try another combination of three images, always starting with an intermediate view.
5. Assign Euler angles to the next selected good class: Call **ANGULAR-RECONSTITUTION**, option **NEW** with suboption **ADD**:

```
IMAGIC-COMMAND: ang-rec

Point-group symmetry           : d6
Minimal stay-away from equator : 5
Option for angular reconst.    : new
Option of NEW                   : add
Input (classsum) images, NO loc#s: classsums1_mra_best_masked
Location numbers wanted        : 10;16;13-14;5   you choose:
                                                    next selected
                                                    class loc#s

Output (selected) image file    : select_1
Output sinogram file, NO loc#s  : sino
ASQ filter the sinogram lines   : yes
Linear mask radius for sinograms: 0.6
Output sinecorr file, NO loc#s  : sinecorr
Wanted angular increment        : 2
Full output of the results      : no
```

6. Assign Euler angles to the next selected good class averages by giving YES to the next question:

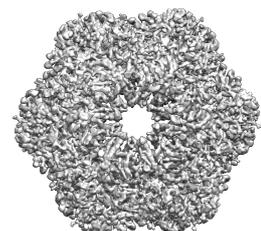
Continue with option NEW	: yes	
Option of NEW	: add	
Location number(s) wanted	:	<i>remaining selected class loc#s</i>

NOTE:

There are is another option, which should be mentioned here, although you will normally not use it during this practical.

You can remove bad images (with a too high ERROR) by using **ANGULAR-RECONSTITUTION** with options **NEW** and **REMOVE**:

Continue with option NEW	: yes	
Option of NEW	: remove	
Re-calculate Euler angles	: yes	
Location number(s) wanted	: 3;5	<i>loc#s in the selected file, you choose</i>



17. Initial 3-D Reconstruction

1. Once you have assigned angles to your (best) class averages you are ready to build your first 3-D using the command **THREED-RECONSTRUCTION**:

```
IMAGIC-COMMAND: th-reconst
Mode of 4D operation           : all
Point-group symmetry          : d6
Use default 3D reconstruction options : yes
Input 2D (classum) images     : select_1
Source of Euler angles        : angrec_header
Update output header          : no
Output file for 3D reconstruction : 3d_1
Output file for re-projections  : rep_1
Output file for error projections : err_1
Spherically mask the reconstruction : yes
Radius of the mask             : 0.6
Also create a normalized 3D volume : yes
Give new sigma                 : 1      helpful in Chimera
Use Hamming window factor      : 0.5
Object size as fraction of image size : 0.8
```

HAMMING WINDOW:

This is a filtering factor and affects the resolution. Here we use a relatively low Hamming window factor due to the low number of class averages used to build the 3-D. It will be increased in future rounds.

2. It is important to check "by eye" how well the re-projections match the class averages. To do this, **DISPLAY** the class averages (`select_1`) you used to create the 3-D in one window and the re-projections (`rep_1`) in another.

The **DISPLAY** settings such as **SCALE** should be the same in both windows.

By flicking back and forth between the two overlaid windows compare how well these two match. If they do not match, the angular assignment was not correct and you should re-run **ANGULAR-RECONSTITUTION** and **THREED-RECONSTRUCTION** with other class averages (chapters 16.2 and 17).

3. Use the error listing at the end of **THREED-RECONSTRUCTION** to remove "bad" class averages from your selected images ([select_1](#)).

BAD CLASS AVERAGES:

As before, exclude the "bad" class averages with the command **EXCLUDE-IMAGE**. Input files are the last class averages ([select_1](#)). Output will be [select_1_best](#). Redo **THREED-RECONSTRUCTION** with [select_1_best](#).

4. If you did not calculate a normalised 3-D volume within **THREED-RECONSTRUCTION** you can still normalise it with the command **THREED-NORM-VARIANCE**:

```
IMAGIC-COMMAND: th-norm

Mode of operation           : NORMVAR
Input file, 3D loc#s       : 3d_1
Output file, 3D loc#s     : 3d_1_norm
Desired new sigma         : 10
```

NOTE:

You can visualize a 3-D volume with the command **DISPLAY**. The display will show slices through the 3-D from bottom to top.

To look at surface views of the 3-D volume use the command **THREED-SURFACE** and **MOVIE** or the program **CHIMERA**.

Please refer to chapter 19.

18. Angular Reconstitution – Random Start-Up

In chapter 16 you had chosen a number “good” images, which were used to assign Euler angles, using three images as a starting point and adding new images.

This procedure can be automated. The option **RANDOM_STARTUP** in the command **ANGULAR-RECONSTITUTION** each image is assigned Euler angles by using the other images as a reference (“anchor-set”). The starting Euler angles are assigned randomly to all images. Please refer to the lectures.

Remember that the input images (**classums1_mra_best_masked**) are the “best” and most typical images, which should be well centred.

1. Call the command **ANGULAR-RECONSTITUTION**, option **RANDOM_STARTUP**:

```

IMAGIC-COMMAND: ang-rec

Point-group symmetry           : d6
Minimal stay-away from equator : 10
Option for angular reconst.    : random_startup
How are the images available    : images
Input (=output) image file     : classums1_mra_best_masked
Sinogram file                  : sino
ASQ filter the sinogram lines  : no
Linear mask radius for sinograms: 0.5                tight
Delete output sinograms        : yes
Wanted angular increment       : 5
Random number generation seed  : 1                (later: old value + 1)
Number of iteration steps      : 25
Full output of the results     : yes

```

2. Reconstruct the 3-D volume with the command **THREED-RECONSTRUCTION** (refer to chapter 17). Output file will be **3d_random_startup_1**, for example. Normalize the 3-D volume (command **THREED-NORM-VARIANCE**).
3. Compare the new reconstruction (**3d_random_startup_1**) with the previous results (**3d_1**). Use **THREED-SURFACE** and **MOVIE** or **Chimera** to visualize the results (chapter 19.).

4. If necessary, redo angular-reconstitution / random-start-up with another combination of input images and another seed for the random number generator.

YOUR NOTES:

19. 3-D Visualization

Before you continue, a few explications on how to visualize a 3-D volume.

19.1. 2-D Sections of the 3-D volume

The 3-D volume is stored as a stack of 2-D sections stored in one IMAGIC image file.

Usually the command **DISPLAY** is used to visualize these 2-D sections.

19.2. 3-D Surface Views

Instead of looking at the sections of the 3-D volume you can also create surface representations of the 3-D volume.

1. Use command **THREED-SURFACE**:

```
IMAGIC-COMMAND: th-surf

Input 3D file, ONE 3D loc#s           : 3d_x
Output file for 2D surface view(s)    : 3d_x_surf
Threshold 3D density value           : 0.1

                                                                    important!! see below
                                                                    you select

Choose projection option:
  FILE          ANOTHER          INTERACTIVE    ORTHOGONAL
  SPIRAL        TETRAHEDRON      TOMOGRAPHY     STEREO
  UNIFORM       ICOSAHEDRON      ASYM_TRIANGLE  RANDOM
Please specify option                  :

                                                                    you select
                                                                    SPIRAL for example

...

Blow-up 3D volume before 3D rendering : yes
3D blow-up dimension                  : 512

...

Default rendering parameters          : yes

...
```

It is very difficult to define the correct surface of an object. Playing around with the threshold (surface rendering) value will result in different looking surface views. A very high threshold value would wipe out the particle; a very low one would keep all the sensible and non-sensible parts. The threshold value found in **THREED-AUTO-MASK** may also be used here.

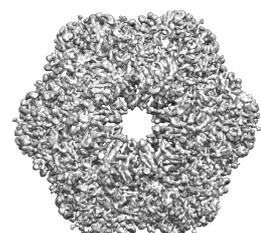
THREED-SURFACE prints a protein mass value (in kDalton), which was calculated according to the specified threshold value:

```
Threshold value used for depicting 3D volume      :      0.250
Number of voxels with density > threshold        :      7962

With a scale (Angstrom per pixel) of             :      10.500
  this corresponds to cubic Angstrom > threshold : 9217010.000
Assuming a protein density (Dalton/cub.Angstrom ) :      0.844
  this corresponds to a protein mass (kDalton) of :      7779.157
Scale (Angstrom per voxel) was specified by      : user
```

You can play around with the threshold value until the kDalton value for your particle is correct. Please don't take this value too seriously; it is only a helping hint!

2. Use **DISPLAY** to look at the surface views.
3. If you have created a sequence of surface view images (with option **SPIRAL**, **TOMOGRAPHY** etc.) you can use the command **MOVIE** to display the surface representation images in an endless loop.
4. Call **MOVIE** and answer the related questions. When the movie is displayed move the cursor into the image and click with the right mouse button to get the control panel. If you have created **STEREO IMAGES** click the "●●" switch in the control window to get moving stereo images. Now roll your eyes and try to see neighbored images in stereo (3-D). Use the "●" switch to leave **MOVIE**.



19.3. Use Chimera

Chimera is a nice non-**IMAGIC** program to visualise 3-D volumes.

Unfortunately, **Chimera** does not (yet) have an **IMAGIC** plugin. So you need an intermediate MRC file to export/import the 3-D volume.

If you did NOT create a normalised 3-D volume in **THREED-RECONSTRUCTION** it is usually helpful to normalize the 3-D volume to a sigma of 1. Use command **THREED-NORM**:

```
IMAGIC-COMMAND: th-norm
Mode of operation           : NORMVAR
Input file, 3D loc#s       : 3d_x
Output file, 3D loc#s      : 3d_x_norm
Desired new sigma          : 1
```

Convert the normalised 3-D volume to MRC format:

```
IMAGIC-COMMAND: em2em
Convert 2D images or 3D volumes      : 3d
Data format of the input to be converted: imagic
How to store output 3D volume        : 3d_volume
Input 3D image file                  : 3d_x_norm
Output 3D image file                 : 3d_x.mrc
Use standard em2em coordinate conversion: yes
In case of type/format conflicts,
                                     which preference: keep_densities
How to get the image names/titles     : name_of_import
```

Run **Chimera** with **3d_x.mrc** as input file.

For further details please refer to the **Chimera** demonstration.

20. 3-D Reconstitution – First Refinements

20.1. Align Input Images to their Re-Projections

To improve the 3-D reconstruction you can apply the following refinement step:

Align the input class averages used for 3-D reconstruction to the related re-projections created in **THREED-RECONSTRUCTION**.

NOTE:

The re-projections were created from the same 3-D volume which means that they are perfectly "3-D aligned".

Call **ALIGN-PARALLEL**:

```
IMAGIC-COMMAND : ali-para

Alignment modes available      : both
Start option                   : trans
Correlation functions available : ccf
Input file, image loc#s       : select_1_best
Output file, image loc#s      : select_1_best_alipara
Reference file, image loc#s    : rep_1_best
Max shift                      : 0.1
Min, max rotation angle       : -180,180
Precision for rot. alignment   : med
Min,max radius for rot. align  : 0,0.7
Maximum number of iterations   : 5
```

Re-do the **THREED-RECONSTRUCTION** now using the aligned selected images (**selected_1_alipara**) as input file. Refer to chapter 17.

20.2. 3-D (Automatic) Masking

Use the command **THREED-AUTO-MASK** to automatically generate a mask for the 3-D volume.

Such a mask removes the influence of the noise when generating references for alignments or anchor sets for angular reconstitution. **THREED-AUTO-MASK** calculates a modulation volume is, which will be binarised. Use **MODULATION** as opposed to **VARIANCE** (check the help for this question if you are curious why to do this).

1. Call **THREED-AUTO-MASK**.

The filter parameters for the modulation calculation need to be specified manually. The filter parameters were explained in the lectures. Note that the low-pass filter parameter needs to be greater-equal the low bound of the band-pass. The higher this filter is set, the "finer/sharper" the mask will be. Use the **?** before answering any question.

```
IMAGIC-COMMAND: th-auto-mask

Automasking options           : do_it_all
Input 3D volume file         : 3d_1_alipara
Output file containing masked input 3D: 3d_1_alipara_masked
Output modulation/variance volume : 3d_1_alipara_modvar
Output file with binary 3D mask  : 3d_1_alipara_mask
Masking based on local modulation : yes
Band-pass parameters         : 0.05,0.25
Low-pass filter parameter     : 0.04
Threshold options            : automatic
Auto-threshold percentage     : 16
```

2. **DISPLAY** the output modulation volume (**3d_1_alipara_mod_var**) and the masked 3-D (**3d_1_alipara_masked**). Also use **THREED-SURFACE** and **MOVIE** or **Chimera** to visualize the results (chapter 19.).
3. Then, start **THREED-AUTO-MASK** again, change the filter parameters (for example, change the low-pass filter to **0.1**), and observe how the modulation and the masked 3-D volume is affected.

4. An ideal mask removes noise outside of the object, but leaves your object completely intact.

YOUR NOTES:

21. Angular Reconstitution - Anchor set

You have created a 3-D volume, which can be used to refine all previous image-processing steps.

First you will use the 3-D volume to get all “typical” views with command **THREED-FORWARD-PROJECTION**. The resulting “2-D forward projection images” have well-defined Euler angles and can serve as references (a so called “anchor set”) to refine the Euler angles of the class averages.

1. You will create these forward projections with command **THREED-FORWARD-PROJECTION**:

```
IMAGIC-COMMAND: th-forward

Option used for current IMAGIC command: FORWARD
Input 3D image file                   : 3d_1_alipara_masked
Output file for forward projections    : arset_1
Threshold 3D density value            : -9999
Use default interpolation mode         : yes
Choose Euler angles option            : asym_triangle
Point-group symmetry to be used       : d6
Option to chose Euler angles          : rand
Number of projections wanted           : 20
Random number generator seed          : 0
Option for Euler angle alpha          : zero
Full output of all parameters         : no
```

2. Assign angles to ALL class averages ([select_1_best_alipara](#)) using the created anchor set ([arset_1](#)). Call the command **ANGULAR-RECONSTITUTION** with the option **ANCHOR_SET**:

```

IMAGIC-COMMAND: angular-reconst

Point-group symmetry           : d6
Minimal stay-away from equator : 5
Option for angular reconstitution : anchor_set
Option of ANCHOR_SET           : fresh
Anchor set options              : single
How are the input images available : images
Input(=output) (classsum) images : select_1_best_alipara
Sinogram file, image loc#s      : sino
ASQ filter the sinogram lines    : yes
Linear mask radius for sinograms : 0.6
How is the anchor set available  : images
Input anchor set IMAGES          : arset_1
Output anchor set sinograms      : ar_sino_1
Output sinecorr file, NO loc#s   : sinecorr
Delete output sinecorr file(s)   : yes
Wanted angular increment in search : 2
Criterion for peak search         : fisher
Output of results                 : final_out
Print histograms                  : no
    
```

You will find error values (printed on the screen) to give you an idea about the quality of the Euler angle assignment of each class average:

```

=====
      EULER results (sorted list)
=====

#          ERROR          LOC          ALPHA          BETA          GAMMA
#          (%)           #
1 :>    e.eeee         nn     aaa.aaa     bbb.bbb     ggg.ggg
2 :>    e.eeee         nn    -aaa.aaa     bbb.bbb     ggg.ggg
...

Average error of the set = e.eeeee %
    
```

3. You will now look for the class averages, which Euler assignments are best. In general, class averages with a low angular error tend to be better.

Use the angular reconstitution error to select some 100 - 400 of the "best" class averages using the command `SORT-IMAGE` with criteria `ANGREC_ERROR` and `UP`, so that the class averages with the lowest angular reconstitution error will be at the beginning of the output file.

```
IMAGIC-COMMAND: ex-copy

What should be copied           : 2d_images
Exclusive copy operation        : SORT
Input file, NO loc#s           : select_1_best_alipara
Output file, image loc#s       : select_1_best_alipara_sort
Source of SORT values          : header
Criteria for sort               : angrec_error
Sort up or down                : up
How many of images wanted      : you select
```

4. You should always check the selected/sorted images "by eye" with the command `DISPLAY` as those "best" class averages are not necessarily the best ones. Also make sure that you did not miss a typical view.

YOUR NOTES:

22. Refined 3-D Reconstruction

This chapter describes a number of iterations to refine your 3-D reconstruction in “real science”. In this practical you will probably do not have the time for these refinements. In this case continue with chapter 25 (Fourier Ring Correlation).

1. Use these best class averages ([select_1_best_sort](#)) to calculate a 3-D reconstruction with the command [THREED-RECONSTRUCTION](#) (chapter 17) and create a new 3-D volume ([3d_2](#)). In this case, use a Hamming window factor of **0.9**.

The command re-projects the 3-D in the same direction as the input class averages. It then calculates an error based on the correlation between the re-projections and the input class averages. After 3-D volume is calculated [THREED-RECONSTRUCTION](#) displays an error list:

```

Error in input images (sorted list)
=====

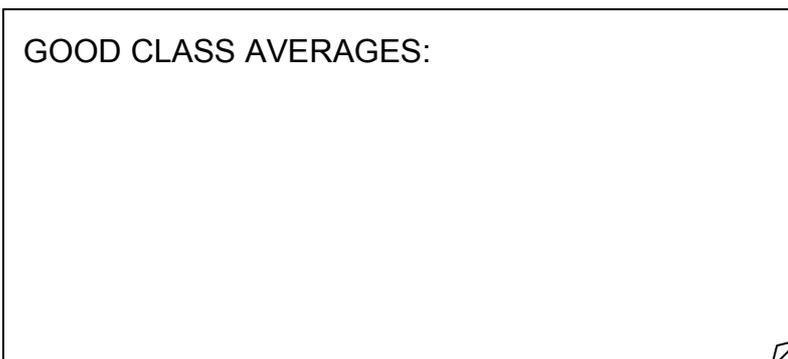
  Loc   class  alpha  beta   gamma  error
  #     #           beta   gamma  in 3-D
1:>  nn     cc     aa.aa  bb.bb  gg.gg  x.xx %
2:>  nn     cc     aa.aa  bb.bb  gg.gg  x.xx %
3:>  nn     cc     aa.aa  bb.bb  gg.gg  x.xx %
      .
      .
      .

Average error in the set of input images:  e.ee %

```

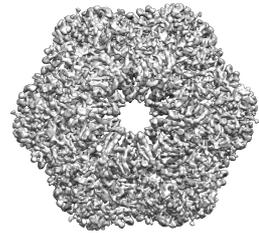
2. If one re-projection has a much higher error than the rest, i.e. if the error suddenly in the sorted list jumps then you should exclude all the following “bad” class averages using the command [EXCLUDE-IMAGE](#).

But, as before, [DISPLAY](#) and check “by eye” and make sure not to miss “typical” views.



3. Calculate a new **THREED-RECONSTRUCTION** without using the excluded bad class-averages (chapter 17).
4. Use **THREED-AUTO-MASK** to mask the filtered 3-D volume (chapter 20.2).

YOUR NOTES:



23. Iterate Angular-Reconstitution and 3-D Reconstruction

1. Forward project with **THREED-FORWARD** the last filtered masked 3-D volume to create a new anchor set. Use this anchor set to run **ANGULAR-RECONSTITUTION** again to refine the Euler angles assignment (chapter 21) and get a new 3-D reconstruction with the command **THREED-RECONSTRUCTION** (chapters 17 and 22). See Figure 15.
2. Repeat these steps until the angles are stable.

YOUR FILE NAMES:

YOUR NOTES:

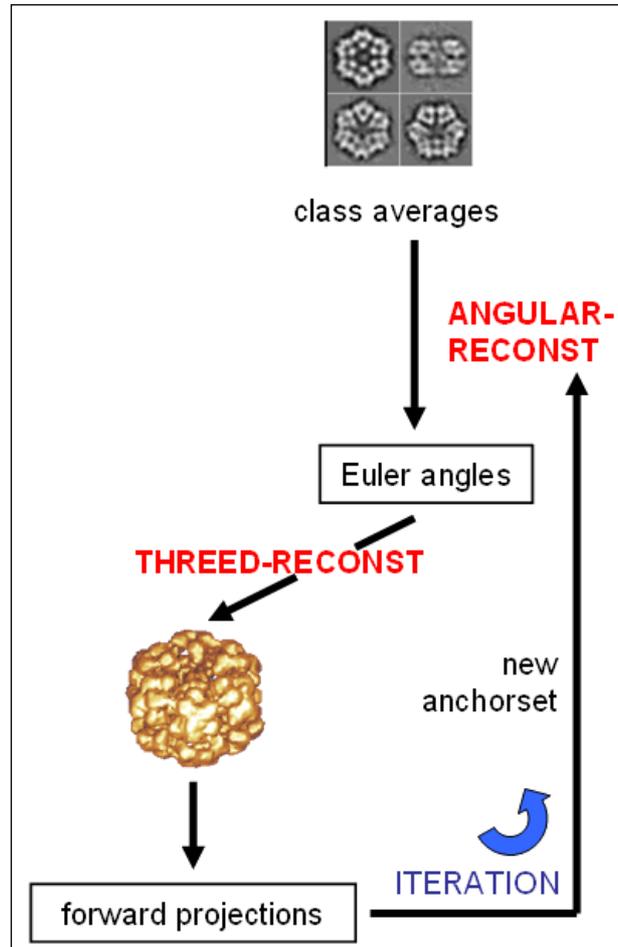
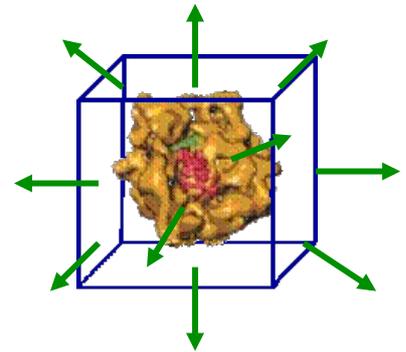


Fig. 15: Iteration of Angular Reconstitution and 3-D Reconstruction



24. Iterate MRA/MSA Classification and Angular Reconstitution/3-D Reconstruction

1. Forward project your last masked 3-D volume into the **ASYM_TRIANGLE** with an angular increment of **7.5** degrees (command **THREED-FORWARD**).

YOUR FILE NAMES:

2. Use these forward projection images as references for a new round of **MULTI-REFERENCE-ALIGNMENT** (chapter 13).

The input file is the aligned output of your last multi-reference alignment (the first time: [particles_mra1](#)). The original (pre-treated) file does not change ([particles_filt](#)).

IMPORTANT NOTE:

Do NOT use the commands **PREPARE-IMAGES**, **NORM-VARIANCE** and **PREPARE-MRA** any more because the references come from one single 3-D and are already centred, i.e. perfectly "3-D aligned".

YOUR FILE NAMES:

3. Run a new MSA classification cycle on the new aligned images and generate a new set of class averages (chapters 14 and 12, respectively).

YOUR FILE NAMES:

4. Iterate M-R-A and MSA classification as before (chapter 15 and figure 14).

YOUR FILE NAMES:

5. Use your last anchor set (the first time: [anchor_set](#)) to assign angles to these class averages with the command **ANGULAR-RECONSTITUTION** option **ANCHOR_SET** (chapter 21).

YOUR FILE NAMES:

6. Re-calculate and refine the 3-D reconstruction as before (chapters 22 and chapter 23).

YOUR FILE NAMES:

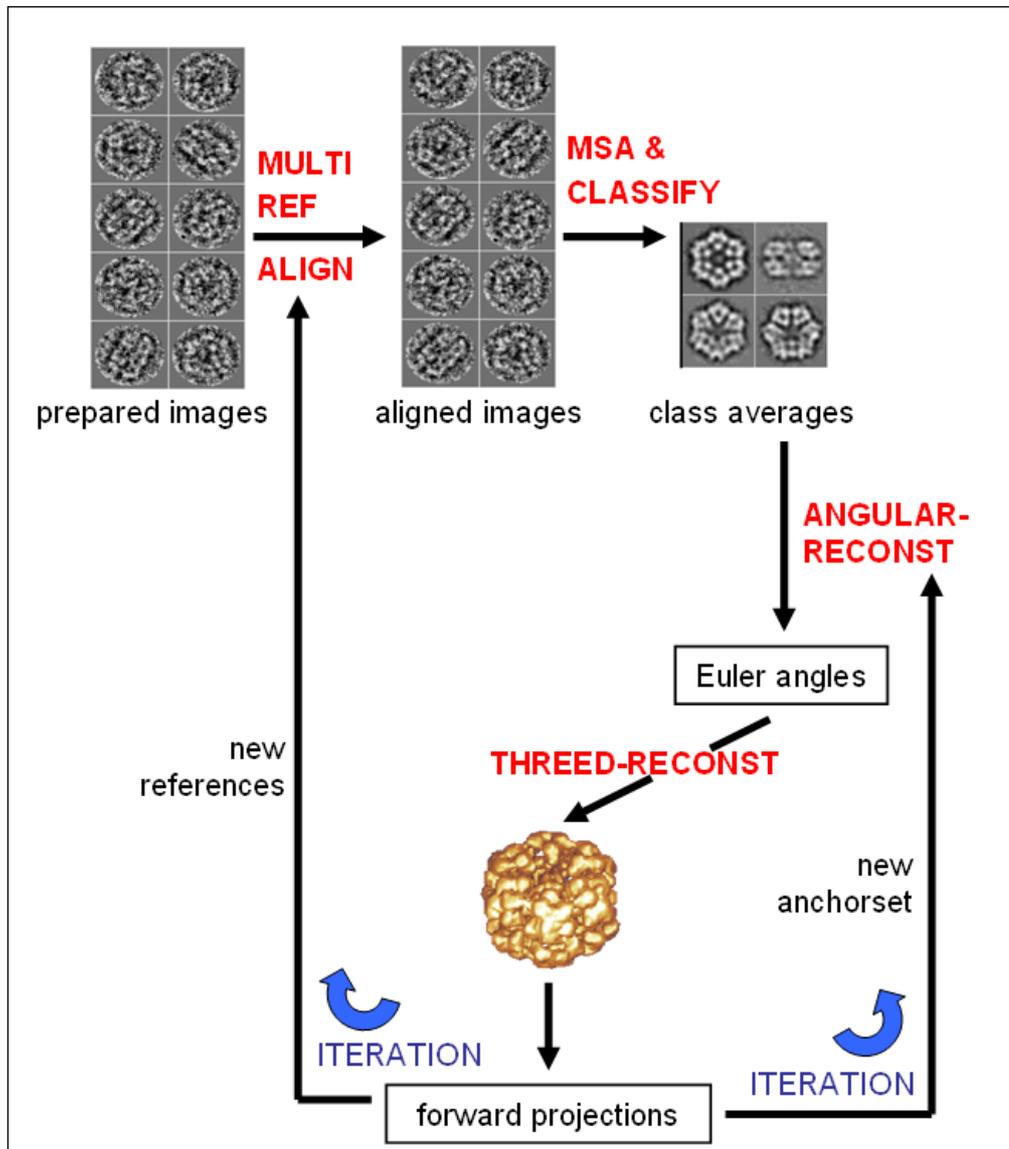


Fig. 16: Iterate MRA / MSA Classification and Angular Reconstitution. / 3D Reconstruction

This refinement loop is repeated many times until you reach your desired resolution or convergence. This iterative refinement is the most time consuming process. As the quality of your reconstruction increases you can use a finer angular increment for forward projecting your M-R-A references.

Also you can go back to particle picking (chapter 8.5) now using the **THREED-FORWARD** images as (better) references for a **CORRELATION** picking of particles (**PICK-IMAGES**) and repeat the MRA / MSA classification (chapter 15) and angular reconstitution / 3-D reconstruction iterations (chapter 23).

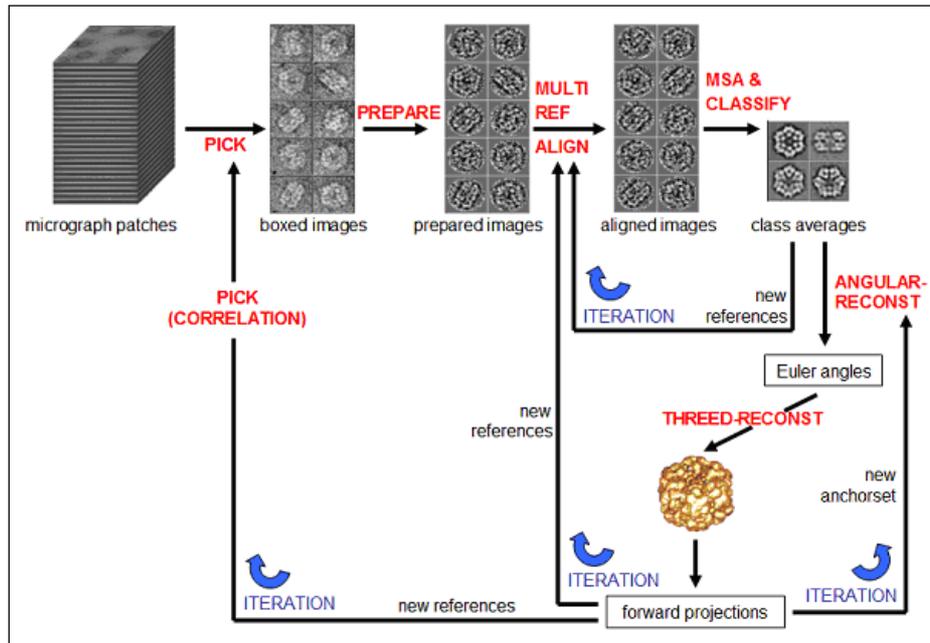


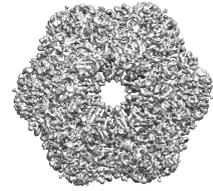
Fig. 17: Re-do Particle Picking / Iterations

Of course, these iterations (chapter 24) are very time-consuming and are usually not done in this practical.

You can also go back to the micrographs and use the un-coarsened data set if the limit of the current sampling is reached. Again, this is not part of this practical.

NOTE:

All these iteration procedures are very important. Practice extreme patience!



25. Fourier Shell Correlation (Resolution Estimation)

In nearly all publications the Fourier shell correlation (FSC) is used to estimate the resolution of a 3-D reconstruction.

Remember, that the FSC is not really a resolution measure but a criterion to compare the similarity of two 3-D reconstructions. If it is used to estimate the resolution of a 3D reconstruction one has to make sure that the two 3D subsets do not contain artificial similarities or the same systematic errors. The best approach would be to calculate two 3-D volumes completely independently.

You cannot do this in this practical. To get an idea how the FSC can be calculated and interpreted you will calculate 3-D volumes from two-subsets, which we assume to be the "two independent data sets".

NOTE:

This is not a "real science" approach

1. Split the class averages (used to create the last 3-D volume) with the **EXCLUSIVE-COPY** command, option **EXTRACT**. Select all the locations in your file and use options **SPLIT_FILE** and **ODD_EVEN**.

```
IMAGIC-COMMAND: ex-copy

What should be copied           : 2d
Mode of copy operation          : split_file
SPLIT_FILE option available     : odd_even
Input file, NO loc#s           :
                                your class averages used to
                                create last best 3D volume
Output root name, NO loc#s     : sub
```

IMPORTANT NOTE:

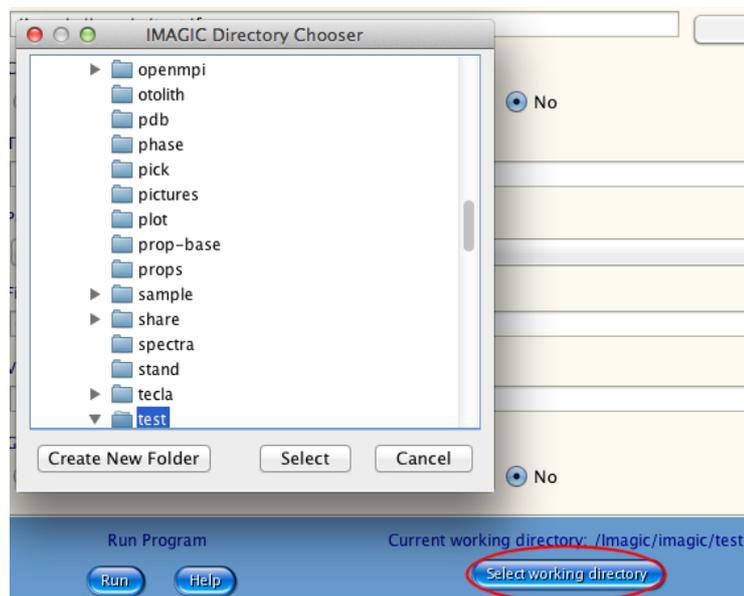
Do NOT use any mask!!

- Then, build two 3-D volumes with the two subsets created. In this case you do not need the re-projections:

```
IMAGIC-COMMAND: th-reconst
...
Input 2D (classum) images, loc#s : sub_odd
...
Output file for 3D reconstructions : fsc_3d_odd
Output file for re-projections : none
Output file for error projections : none
Spherically mask the reconstruction: no
...

IMAGIC-COMMAND: th-reconst
...
Input 2D (classum) images, loc#s : sub_even
...
Output file for 3D reconstructions : fsc_3d_even
...
```

- Use command FSC-GUI to start the GUI version of command **FOURIER-SHELL-CORRELATION**. Change to your working directory (button "Change working directory")



and answer all questions:

The screenshot shows the IMAGIC software interface with the following settings:

- Mode of operation:** FSC
- Mode of operation:** ONE_REFERENCE
- Input 3D file, 3D loc#s:** /Imagic/imagic/test/sub_odd
- 3D reference file, ONE 3D loc:** /Imagic/imagic/test/sub_even
- Output (PLT) filename:** /Imagic/imagic/test/fsc
- Create additional CSV output file:** No
- Threshold for FSC curve (sigma):** 3.0
- Point-group symmetry to be used:** D6
- Filling degree:** 0.66
- Voxel size measured in Angstroms:** 1.0
- Graphics output also in terminal window:** No

At the bottom, the interface shows a "Run Program" section with buttons for "Run", "Help", "Select working directory", and "Exit IMAGIC". The current working directory is set to /Imagic/imagic/test.

Press the "Run" button to calculate the FSC. In the results window use the buttons "Next" and "Previous" to get the FSC/3 Sigma and the FSC/1/2-Bit curve.

The crossing of the Fourier Shell Correlation and the (modified) 3-sigma curve (in output plot #1) indicates where the FSC systematically emerges above the expected random correlations of the background noise. This criterion indicates at which spatial frequency you are systematically gaining information significantly above the random noise level. When you continue collecting information by adding more data of the same quality to the data set you would certainly improve the data set up to - and maybe even somewhat beyond - this point.

The crossing of the Fourier Shell Correlation and the 1/2-bit information threshold curve (in output plot #2) expresses where you have already collected a sufficient amount of data in the final 3-D reconstruction to allow a direct structural interpretation at that resolution level. The 1/2-bit curve is calibrated to approximately yield resolution values comparable to resolution values in use in X-ray crystallography (FOM).

4. If you do not want (or cannot use) the FSC wrapper use **FOURIER-SHELL-CORRELATION** and **PLOT** to visualise the FSC curves:

```
IMAGIC-COMMAND: four-shell

Option used for current IMAGIC command : FSC
Mode of operation                       : one_ref
Input 3D file, 3D loc#s                 : fsc_3d_odd
3D reference file, ONE 3D loc           : fsc_3d_even
Output (PLT) filename                   : fsc
Create additional CSV output file       : no
Threshold for FSC curve (sigma)        : 3

Pointgroup symmetry to be used         : d6
Filling degree                          : 0.66

IMAGIC-COMMAND: plot

Input file (image or plot)              : fsc
...
```

Note that when using **PLOT** the horizontal axis shows 1/Resolution. Also do not miss to consider the exponent! Estimate your resolution based on where the 1/2 bit curve crosses the FSC curve.

NOTE:

The FSC is a measure to compare the similarity of two 3-D data sets. If it is used to estimate the resolution of a 3-D reconstruction you have to make sure that the two 3-D subsets do not contain artificial similarities (like masks, for example).

It is good practise not to interpret resolution curves, which are too close to the high end of the resolution curve (the right hand side of the FSC curve). In other words: you should never claim any resolution level beyond 2/3rd of the Nyquist frequency.

If the sampling size is 5.2 Angstrom per pixel/voxel then the attainable resolution is about 15.6 Angstrom rather than the theoretical Nyquist frequency of 10.4 Angstrom. If the resolution is better than 3x the sampling size your data set is under-sampled and you should re-scan your micrographs with a higher resolution and re-do the image analysis.

Whilst the 1/2 bit curve provides a single figure for your resolution it is important to always take into account the curve as a whole when judging the quality of the reconstruction.

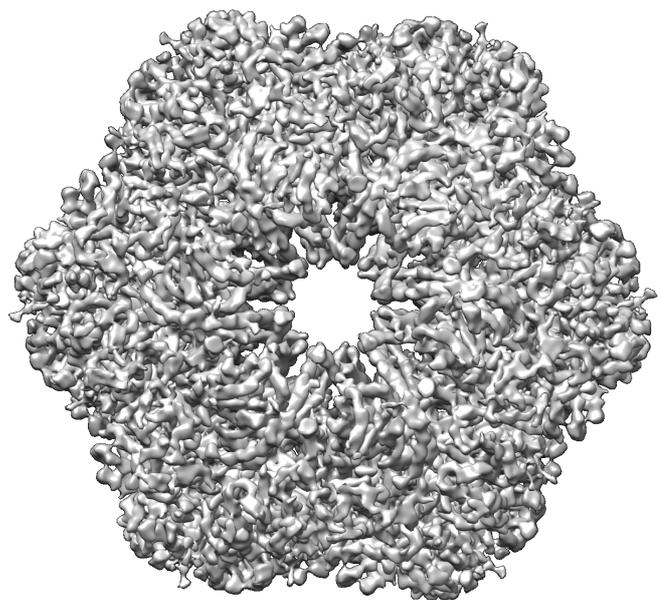
The best resolution measure is still the resolution of the biological details, which you can see in your 3-D reconstruction.

WEBSITES:

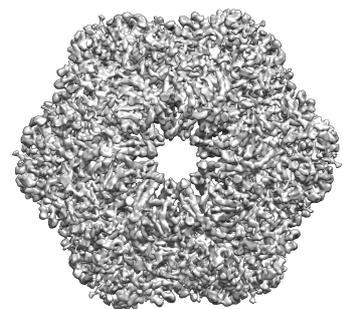
www.single-particles.org

ERROR HINTS:

We tried to find and correct all errors and typos during the Brazil School. If you still find some mistakes please send your error hints to michael@ImageScience.de so that we can improve this tutorial. Thank you very much.



YOUR NOTES:



CONTENT

1. INTRODUCTION TO IMAGIC	3
1.1. COMMANDS CREATE-IMAGE AND DISPLAY	7
1.2. NOISE	10
1.3. NOISE REDUCTION BY IMAGE AVERAGING	11
1.4. MODE COMMANDS	12
2. THE FOURIER TRANSFORM	13
2.1. TEST CURVES	13
2.2. FOURIER TRANSFORM	14
2.3. CURVES AND THEIR FOURIER TRANSFORMS	15
2.4. RELATIONSHIP BETWEEN IMAGE SPACE AND FOURIER SPACE	17
2.5. FOURIER SPACE AND FILTERING	20
2.6. 2-D IMAGES AND FOURIER TRANSFORMS - FIRST STEPS	24
2.7. 2-D IMAGES AND FOURIER TRANSFORMS - MASKS	26
2.8. 2-D IMAGES AND FOURIER FILTERS	27
3. THE DATA SET: WORM HEMOGLOBIN	32
4. IMAGE ANALYSIS: SHORT OVERVIEW	33
5. MICROGRAPHS	34
6. CONTRAST TRANSFER FUNCTION (CTF)	38
6.1. PLAYING AROUND WITH EM PARAMETERS AND THEIR INFLUENCE ON THE CTF	38
6.2. INTERACTIVE CTF CORRECTION	41
7. (AUTOMATIC) CTF CORRECTION	43
7.1. CALCULATE PRE-TREATED AMPLITUDE IMAGES	43
7.2. MSA OF AMPLITUDE IMAGES	46
7.3. ESTIMATE CTF USING MSA EIGEN-FILTERING	47
8. PARTICLE PICKING	53
8.1. MODULATION PICKING	53
8.2. EXTRACT "GOOD" IMAGES AFTER MODULATION PICKING	56
8.3. MSA CLASSIFICATION TO GET REFERENCES FOR CORRELATION PICKING	62
8.4. PREPARE MICROGRAPHS AND REFERENCES FOR CORRELATION PICKING	65
8.5. CORRELATION PICKING	66

9. PRE-TREATMENT	68
10. ALIGNMENT-BY-CLASSIFICATION	69
11. CENTRING	69
12. MULTIVARIATE STATISTICAL ANALYSIS (MSA) CLASSIFICATION	70
13. MULTI-REFERENCE-ALIGNMENT (MRA)	76
14. MSA CLASSIFICATION	81
15. ITERATION CYCLE(S) OF MULTI-REFERENCE ALIGNMENT AND MSA CLASSIFICATION	83
16. ANGULAR RECONSTITUTION - INITIAL ANGULAR ASSIGNMENT	84
16.1. ANGULAR RECONSTITUTION - SELF SEARCH	85
16.2. ANGULAR RECONSTITUTION - NEW PROJECTIONS	87
17. INITIAL 3-D RECONSTRUCTION	90
18. ANGULAR RECONSTITUTION - RANDOM START-UP	92
19. 3-D VISUALIZATION	94
19.1. 2-D SECTIONS OF THE 3-D VOLUME	94
19.2. 3-D SURFACE VIEWS	94
19.3. USE CHIMERA	96
20. 3-D RECONSTITUTION - FIRST REFINEMENTS	97
20.1. ALIGN INPUT IMAGES TO THEIR RE-PROJECTIONS	97
20.2. 3-D (AUTOMATIC) MASKING	98
21. ANGULAR RECONSTITUTION - ANCHOR SET	100
22. REFINED 3-D RECONSTRUCTION	103
23. ITERATE ANGULAR-RECONSTITUTION AND 3-D RECONSTRUCTION	105
24. ITERATE MRA/MSA CLASSIFICATION AND ANGULAR RECONSTITUTION/3-D RECONSTRUCTION	107
25. FOURIER SHELL CORRELATION (RESOLUTION ESTIMATION)	111
CONTENT	118