Introductory session
Basic Formats

- Opening *Dynamo*
- Getting help
- Manipulating particles and metadata

General GUI

In matlab:

```
>>dynamo
```

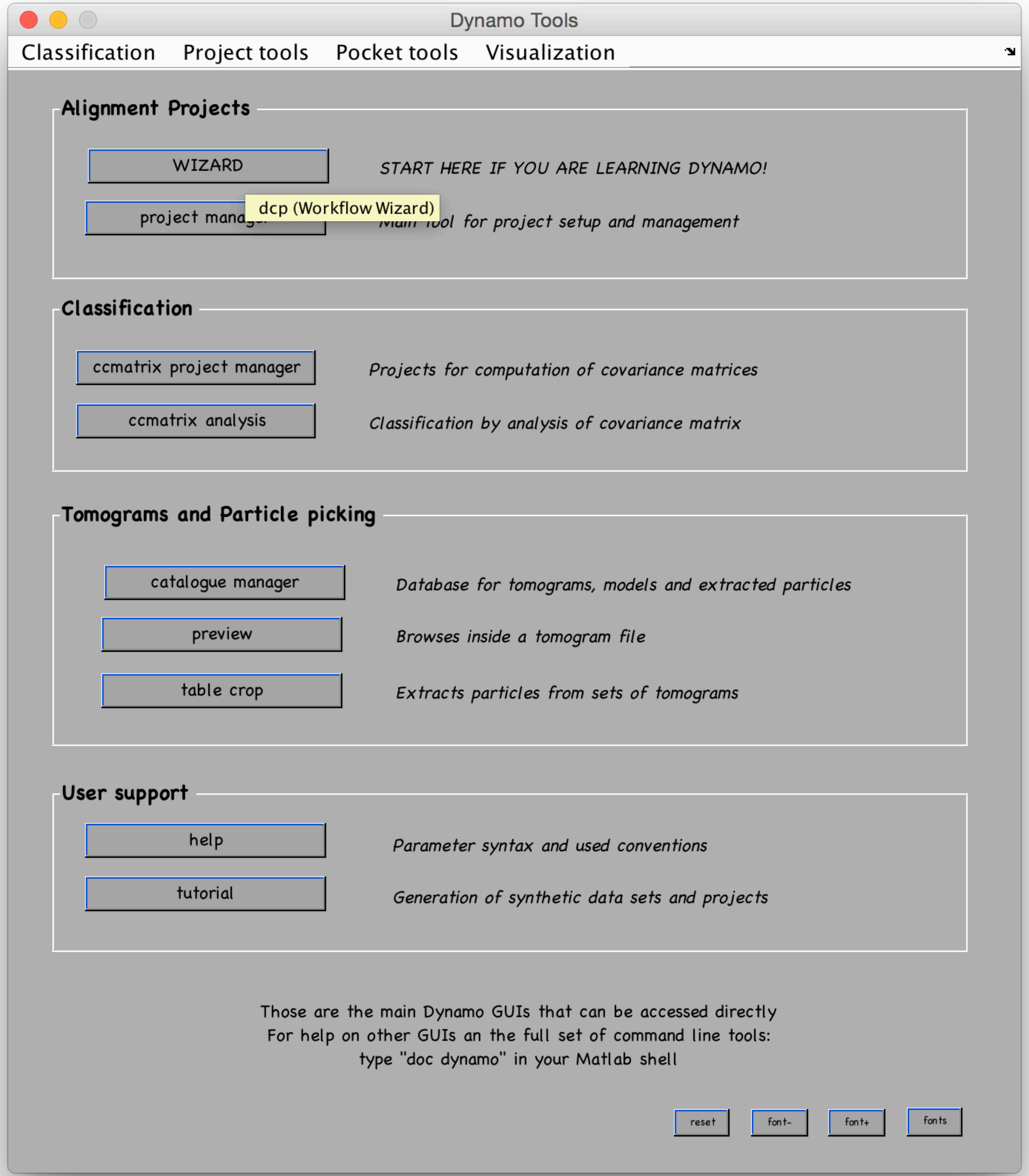
Or

```
>>dgui
```

In the standalone console:

```
Dynamo >> gui
```

All the modules are independently accessible through the command line.
Keep the cursor on the corresponding button to find out which command Invokes the corresponding functionality.



START HERE IF YOU ARE LEARNING DYNAMO!

Main tool for project setup and management

Projects for computation of covariance matrices

Classification by analysis of covariance matrix

Database for tomograms, models and extracted particles

Browses inside a tomogram file

Extracts particles from sets of tomograms

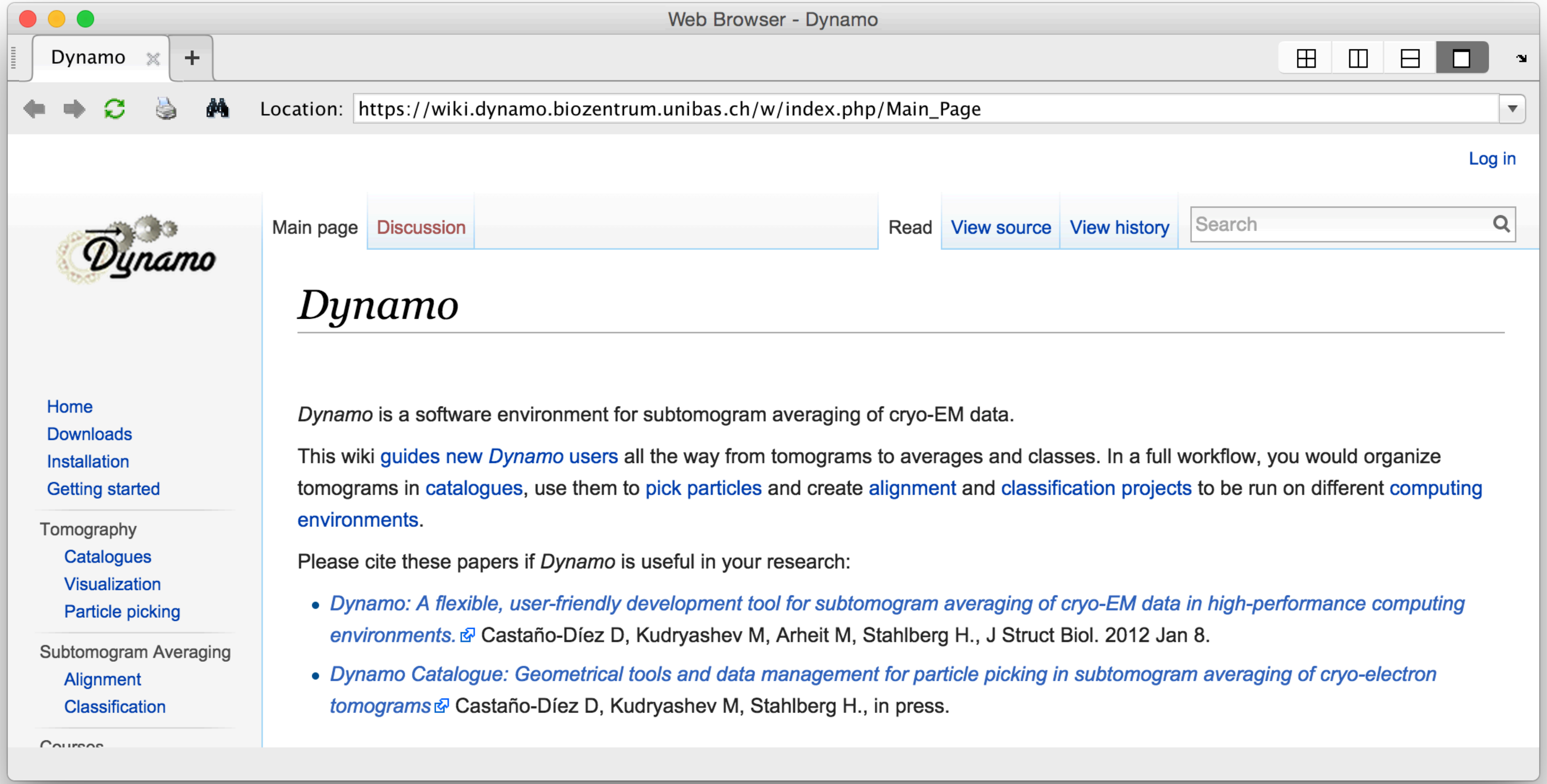
Parameter syntax and used conventions

Generation of synthetic data sets and projects

Those are the main Dynamo GUIs that can be accessed directly
For help on other GUIs an the full set of command line tools:
type "doc dynamo" in your Matlab shell

Online help is available through the wiki:

```
>> dwiki
```



Dynamo

Dynamo is a software environment for subtomogram averaging of cryo-EM data.

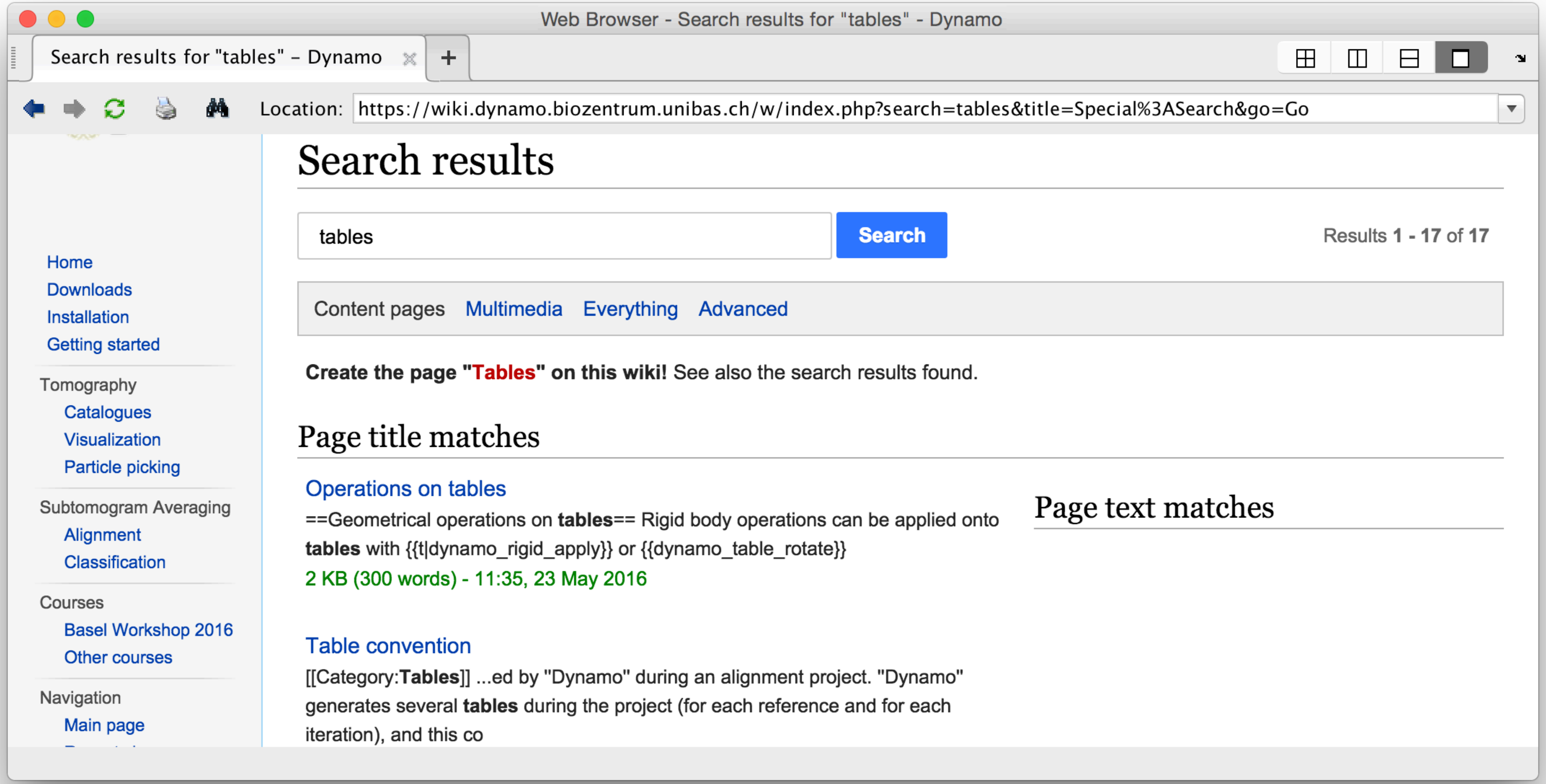
This wiki guides new *Dynamo* users all the way from tomograms to averages and classes. In a full workflow, you would organize tomograms in catalogues, use them to pick particles and create alignment and classification projects to be run on different computing environments.

Please cite these papers if *Dynamo* is useful in your research:

- *Dynamo: A flexible, user-friendly development tool for subtomogram averaging of cryo-EM data in high-performance computing environments.* Castaño-Díez D, Kudryashev M, Arheit M, Stahlberg H., J Struct Biol. 2012 Jan 8.
- *Dynamo Catalogue: Geometrical tools and data management for particle picking in subtomogram averaging of cryo-electron tomograms* Castaño-Díez D, Kudryashev M, Stahlberg H., in press.

... or with arguments

```
>> dwiki table
```



https://wiki.dynamo.biozentrum.unibas.ch/w/index.php?search=tables&title=Special%3ASearch&go=Go

Search results

tables

Results **1 - 17** of **17**

Content pages Multimedia Everything Advanced

Create the page "Tables" on this wiki! See also the search results found.

Page title matches

Operations on tables

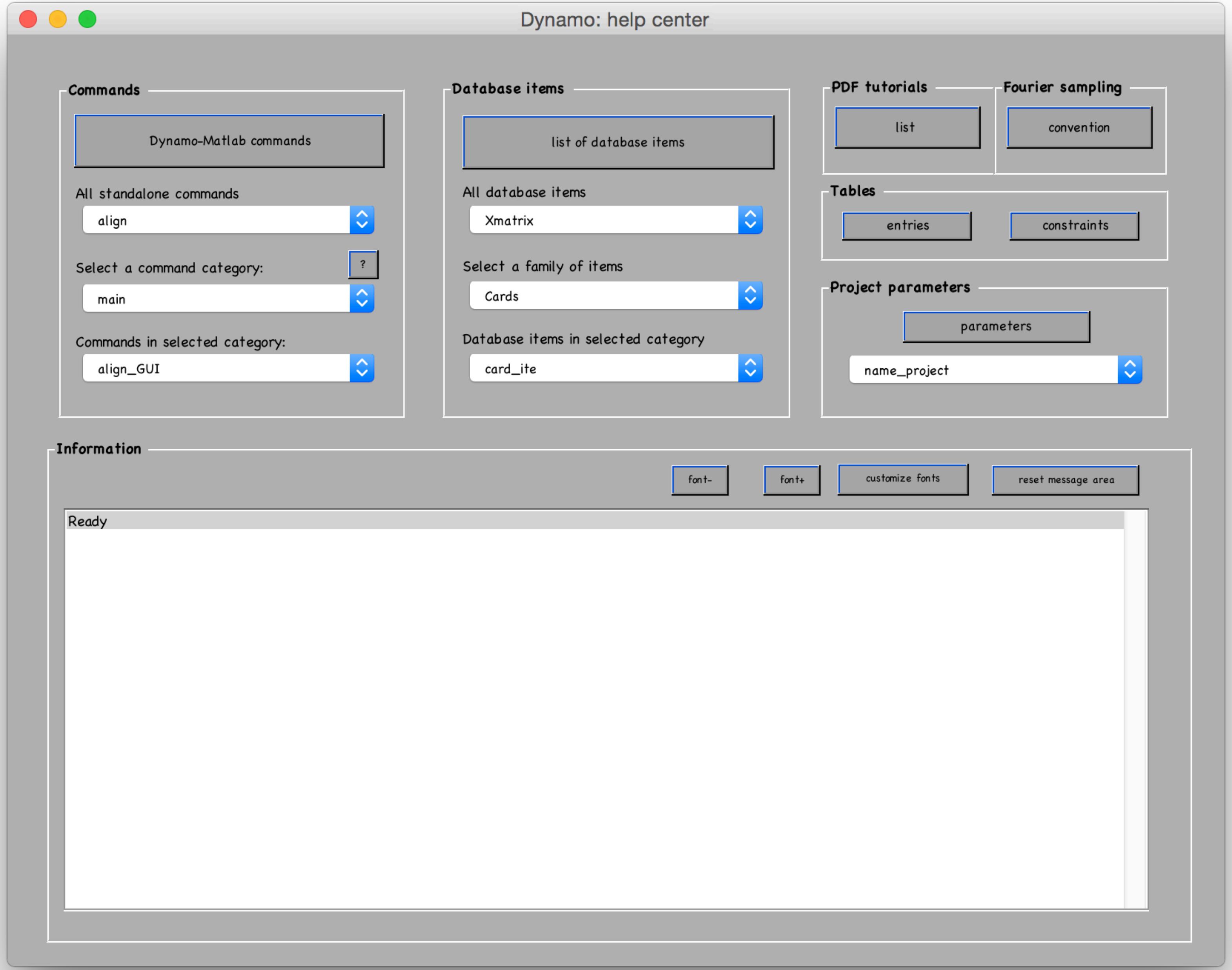
==Geometrical operations on **tables**== Rigid body operations can be applied onto **tables** with {{t|dynamo_rigid_apply}} or {{dynamo_table_rotate}}

2 KB (300 words) - 11:35, 23 May 2016

Table convention

[[Category:**Tables**]] ...ed by "Dynamo" during an alignment project. "Dynamo" generates several **tables** during the project (for each reference and for each iteration), and this co

Page text matches



All standalone commands: align

Select a command category: main

Commands in selected category: align_GUI

All database items: Xmatrix

Select a family of items: Cards

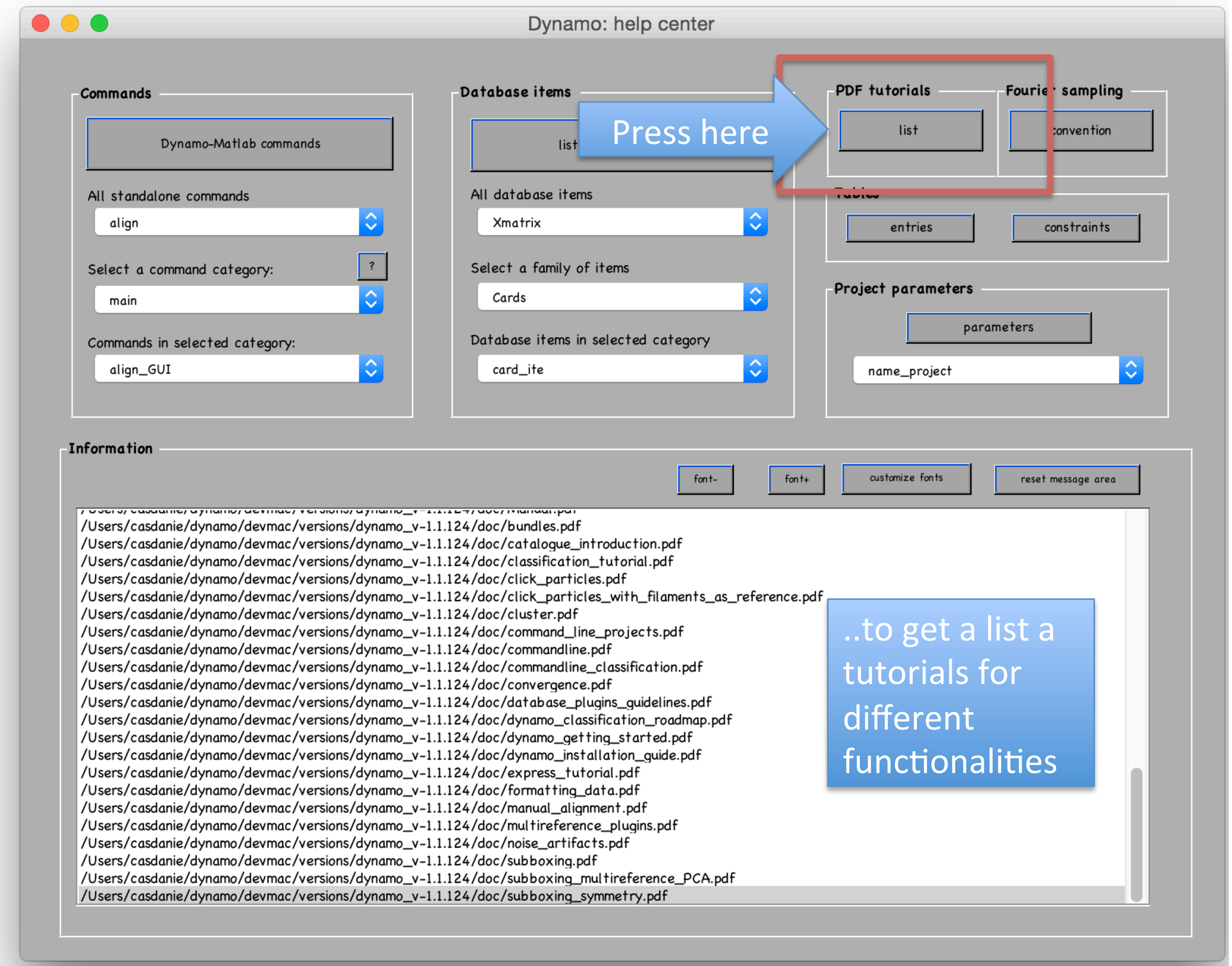
Database items in selected category: card_ite

name_project

Ready

For instance the help center an also be invoked by the command line with dhelp

The help center offers access to many tutorials, including this one:



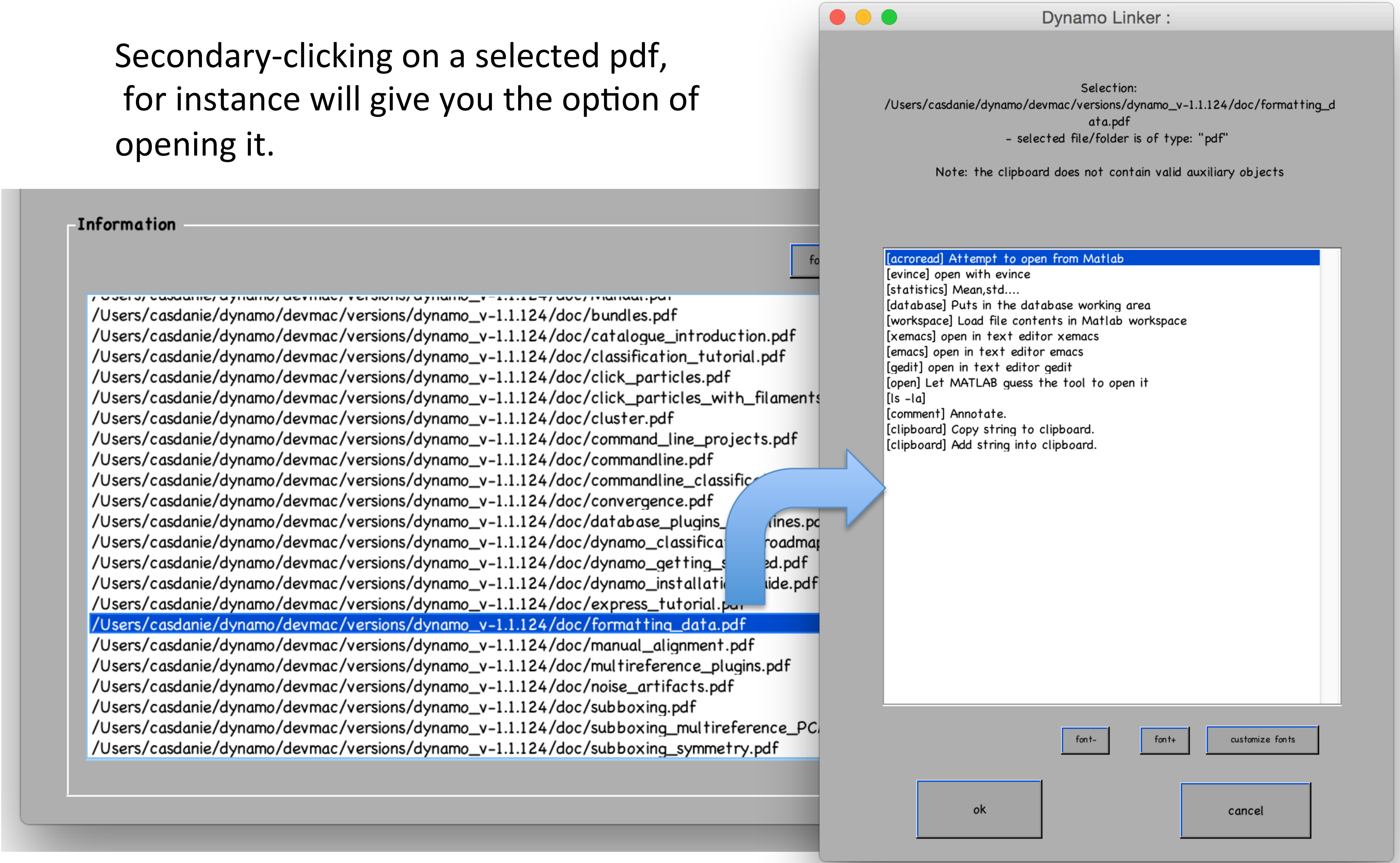
Press here

```
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/Manual.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/bundles.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/catalogue_introduction.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/classification_tutorial.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/click_particles.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/click_particles_with_filaments_as_reference.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/cluster.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/command_line_projects.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/commandline.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/commandline_classification.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/convergence.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/database_plugins_guidelines.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_classification_roadmap.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_getting_started.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_installation_guide.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/express_tutorial.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/formatting_data.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/manual_alignment.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/multireference_plugins.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/noise_artifacts.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing_multireference_PCA.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing_symmetry.pdf
```

..to get a list a tutorials for different functionalities

In any Dynamo GUI, file and folder names are selectable
[secondary click or ctrl+main click on a Mac]
Selecting an object will get you a menu of options that will vary for the different objects.

Secondary-clicking on a selected pdf,
for instance will give you the option of
opening it.



Information

/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/manual.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/bundles.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/catalogue_introduction.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/classification_tutorial.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/click_particles.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/click_particles_with_filaments
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/cluster.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/command_line_projects.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/commandline.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/commandline_classific
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/convergence.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/database_plugins_ ines.pd
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_classifica roadmap
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_getting_s d.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/dynamo_installati ide.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/express_tutorial.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/formatting_data.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/manual_alignment.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/multireference_plugins.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/noise_artifacts.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing.pdf
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing_multireference_PC
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/subboxing_symmetry.pdf

Dynamo Linker :

Selection:
/Users/casdanie/dynamo/devmac/versions/dynamo_v-1.1.124/doc/formatting_data.pdf
- selected file/folder is of type: "pdf"

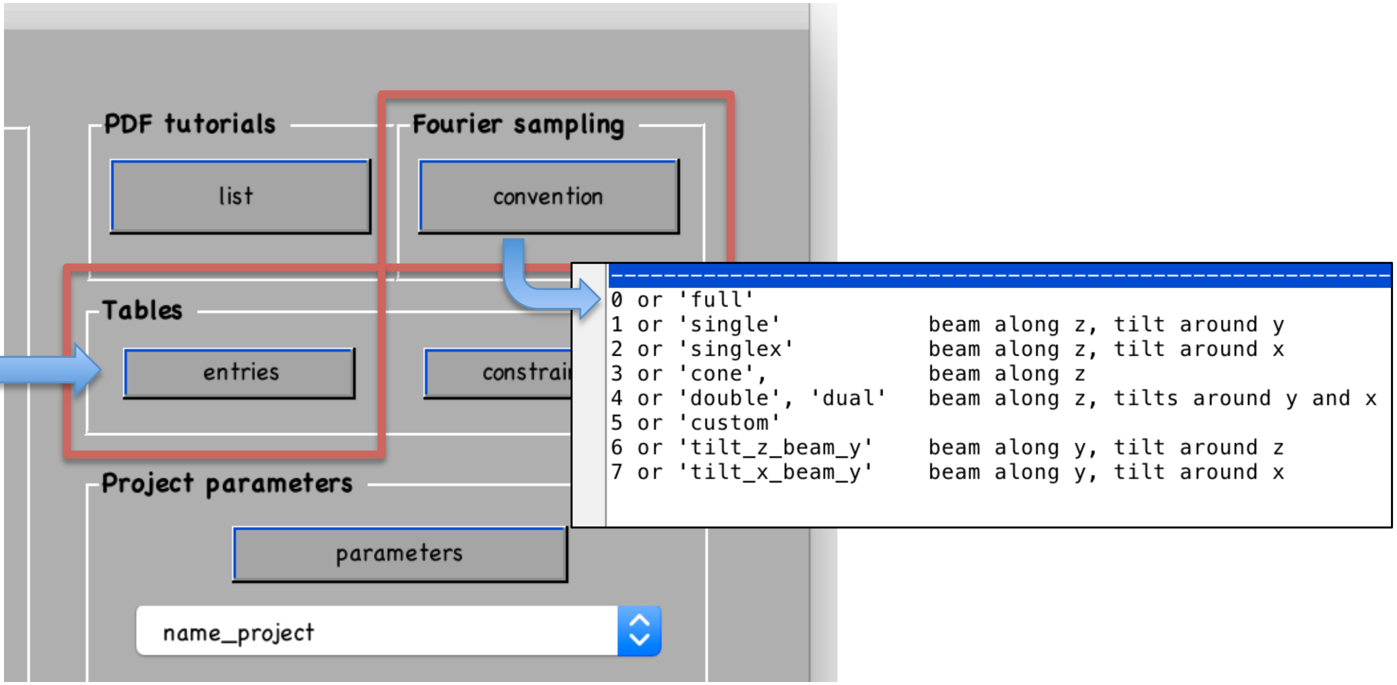
Note: the clipboard does not contain valid auxiliary objects

[acroread] Attempt to open from Matlab
[evince] open with evince
[statistics] Mean,std....
[database] Puts in the database working area
[workspace] Load file contents in Matlab workspace
[xemacs] open in text editor xemacs
[emacs] open in text editor emacs
[gedit] open in text editor gedit
[open] Let MATLAB guess the tool to open it
[ls -la]
[comment] Annotate.
[clipboard] Copy string to clipboard.
[clipboard] Add string into clipboard.
font-
font+
customize fonts
ok
cancel

Dynamo uses several conventions: metadata, data, fourier sampling

In the Help Center is the place to look when you need a quick reminder.

Metadata description



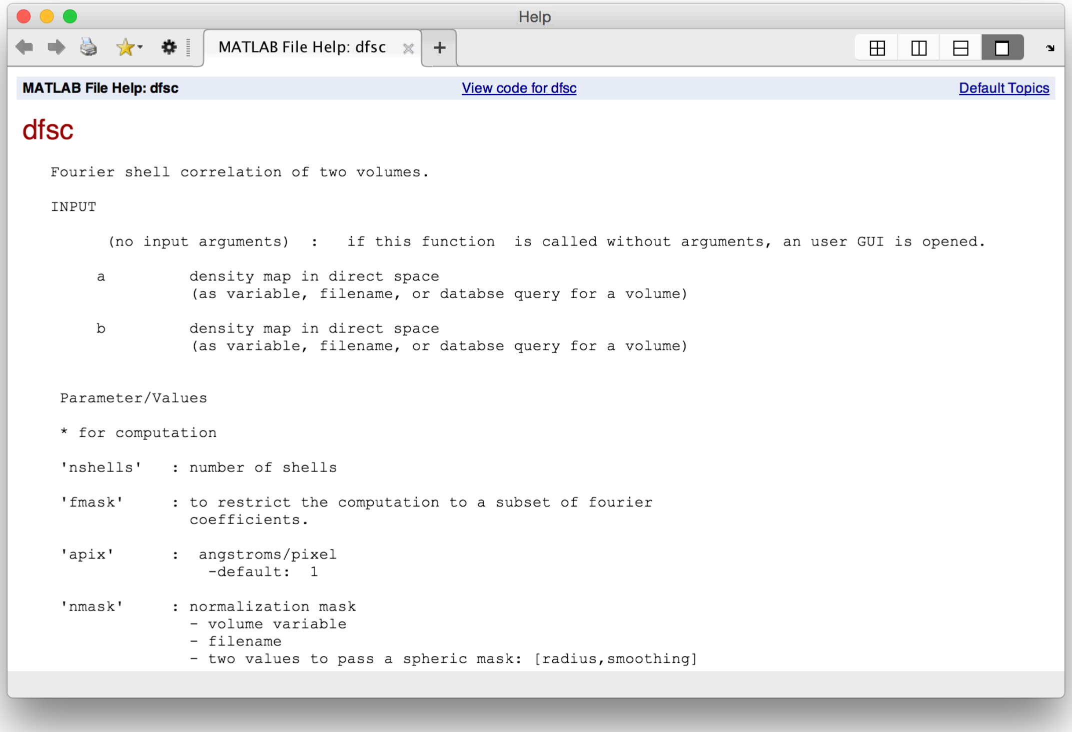
```
----------------------------------------------------------------
0 or 'full'
1 or 'single'           beam along z, tilt around y
2 or 'singlex'          beam along z, tilt around x
3 or 'cone',            beam along z
4 or 'double', 'dual'   beam along z, tilts around y and x
5 or 'custom'
6 or 'tilt_z_beam_y'    beam along y, tilt around z
7 or 'tilt_x_beam_y'    beam along y, tilt around x
```

Besides the tutorials and the Help Center, every Dynamo command provides help about its syntax.

You’ll need this when you are not operating through the GUI.

`>>doc dfsc`

Shows the help in an interactive window.



dfsc

```
Fourier shell correlation of two volumes.

INPUT

      (no input arguments)  :   if this function  is called without arguments, an user GUI is opened.

     a          density map in direct space
                (as variable, filename, or databse query for a volume)

     b          density map in direct space
                (as variable, filename, or databse query for a volume)


 Parameter/Values

 * for computation

 'nshells'   : number of shells

 'fmask'     : to restrict the computation to a subset of fourier
               coefficients.

 'apix'      :  angstroms/pixel
                 -default:  1

 'nmask'     : normalization mask
               - volume variable
               - filename
               - two values to pass a spheric mask: [radius,smoothing]
```

`>>help dfsc`

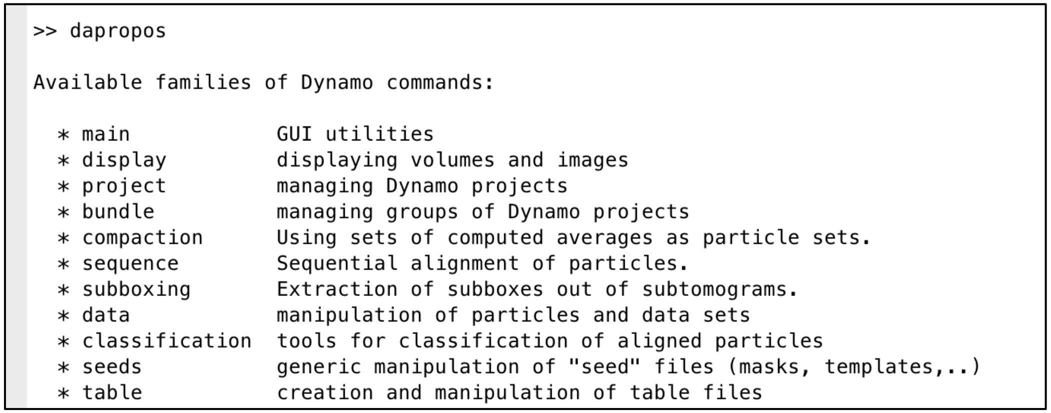
Prints the output directly on the screen.

But… how to know which commands are available in Dynamo?

In general, the different tutorials will introduce you to the most important commands in each functionality area.

To find them on a more systematical way,
a selection of commands is classified through areas of functionality:

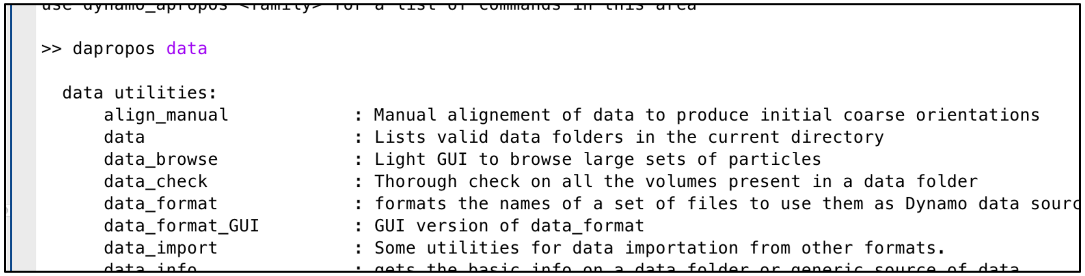
`>>dapropos`
Shows the names of the areas:



```
>> dapropos

Available families of Dynamo commands:

  * main             GUI utilities
  * display          displaying volumes and images
  * project          managing Dynamo projects
  * bundle           managing groups of Dynamo projects
  * compaction       Using sets of computed averages as particle sets.
  * sequence         Sequential alignment of particles.
  * subboxing        Extraction of subboxes out of subtomograms.
  * data             manipulation of particles and data sets
  * classification   tools for classification of aligned particles
  * seeds            generic manipulation of "seed" files (masks, templates,..)
  * table            creation and manipulation of table files
```

You can select then one area to show a selection of commands related to one functionality:

`>>dapropos data`
Shows the main classification related commands:



```
use dynamo_apropos <family> for a list of commands in this area

>> dapropos data

  data utilities:
      align_manual            : Manual alignement of data to produce initial coarse orientations
      data                    : Lists valid data folders in the current directory
      data_browse             : Light GUI to browse large sets of particles
      data_check              : Thorough check on all the volumes present in a data folder
      data_format             : formats the names of a set of files to use them as Dynamo data sourc
      data_format_GUI         : GUI version of data_format
      data_import             : Some utilities for data importation from other formats.
      data_info               : gets the basic info on a data folder or generic source of data
```

Basic formats

There are three basic object types that will appear continuously while performing subtomogram averaging with Dynamo:

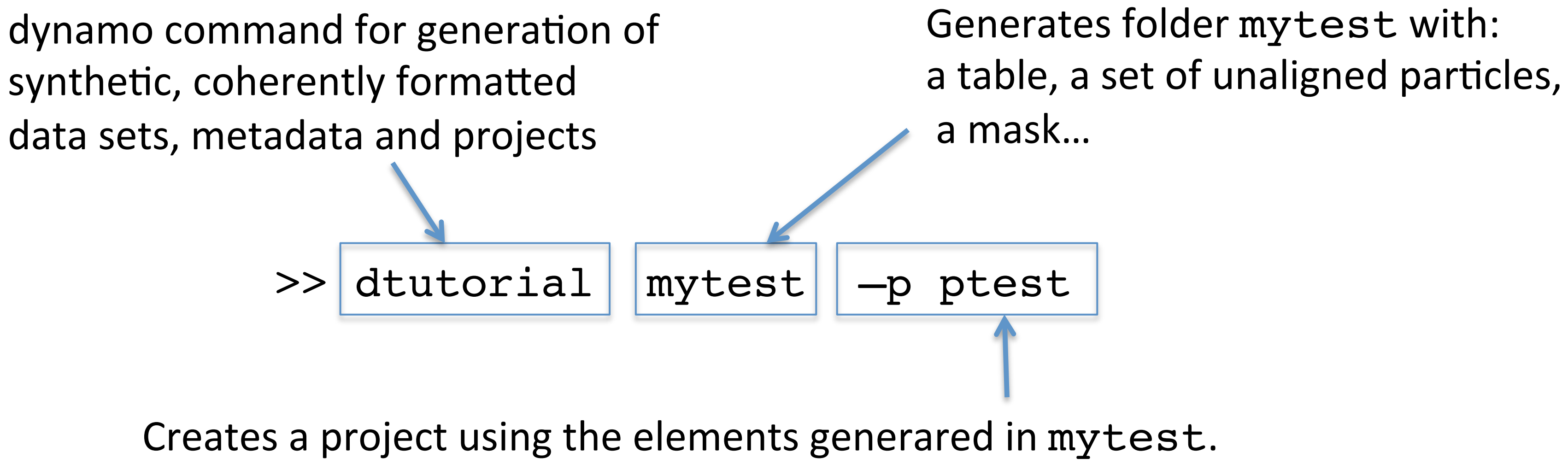
- metadata: *tables*
- data: *data folders*
- computation: *projects*

(we are assuming in this part that 3D particles are already cropped from out of the tomograms, and that we are interested in aligning and averaging a set of subtomograms)

We are going to get familiar with them running an example on a synthetic data set.

We will generate synthetically a folder that contains all elements that would appear during a real work session.

So, go to an empty folder in your file system and generate a tutorial with the command:



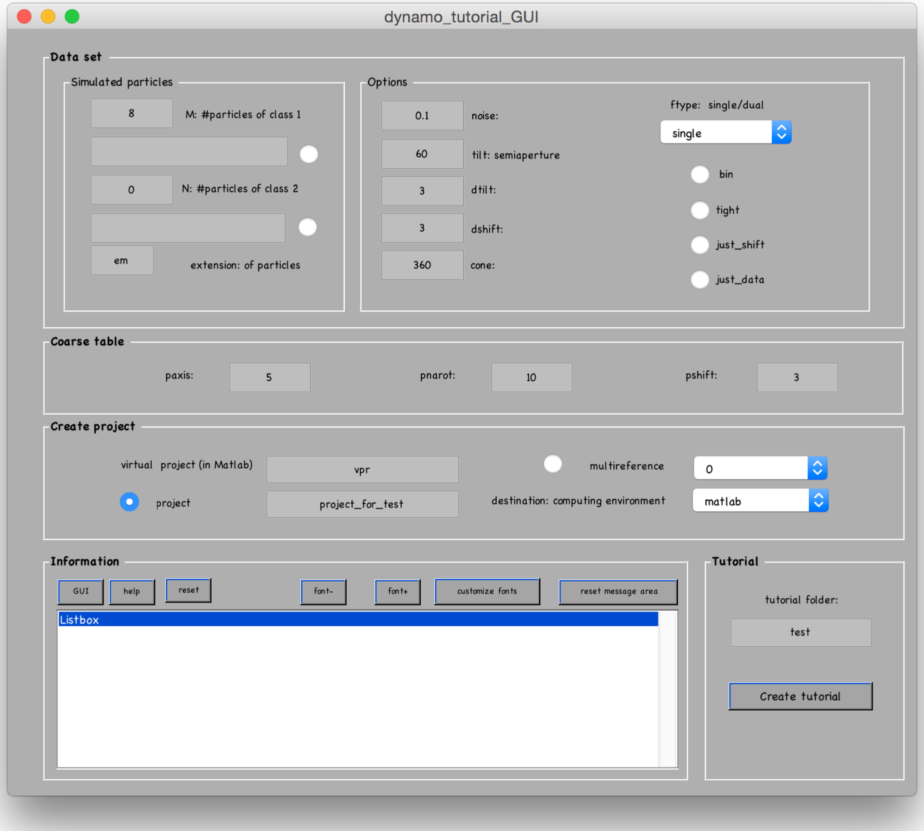
dynamo command for generation of synthetic, coherently formatted data sets, metadata and projects

Generates folder `mytest` with: a table, a set of unaligned particles, a mask...

```
>> dtutorial mytest –p ptest
```

Creates a project using the elements generared in `mytest`.

Side note: notice that `dtutorial` can be invoked without arguments. This generates a GUI that allows entering the main parameters. Command line use normally provides more functionality.



dynamo_tutorial_GUI

Data set

Simulated particles

- M: #particles of class 1: 8
- ○
- N: #particles of class 2: 0
- ○
- extension: of particles: em

Options

- noise: 0.1
- tilt: semiaperture: 60
- dtilt: 3
- dshift: 3
- cone: 360
- ftype: single/dual: single
- ○ bin
- ○ tight
- ○ just_shift
- ○ just_data

Coarse table

- paxis: 5
- pnarot: 10
- pshift: 3

Create project

- virtual project (in Matlab): vpr
- ● project: project_for_test
- ○ multireference: 0
- destination: computing environment: matlab

Information

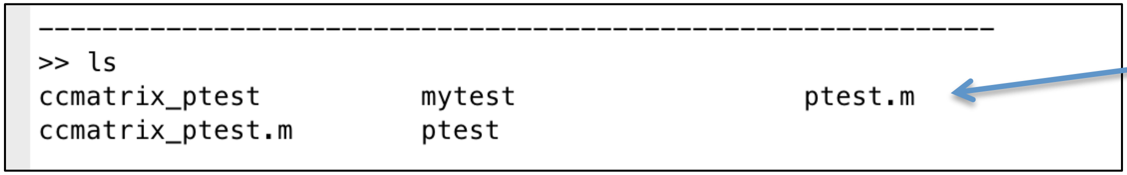
Listbox

Tutorial

- tutorial folder: test

Many of the main commands in Dynamo work this way.

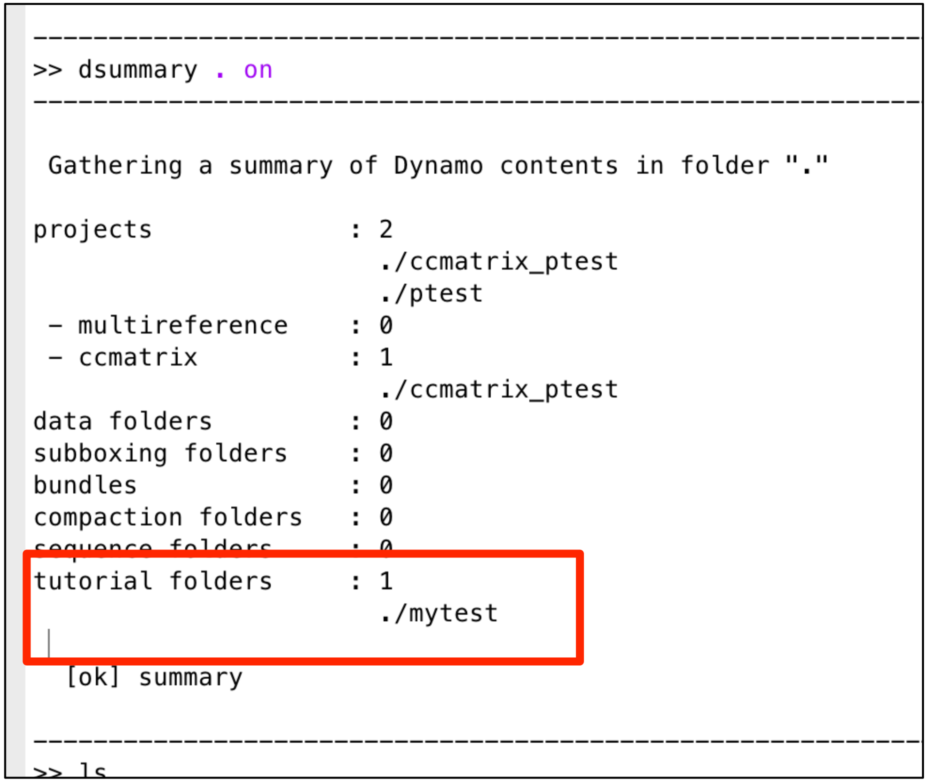
After some moments, *Dynamo* will have create some contents in the folder. We can explore it with the system/Matlab command ls



```
------------------------------------------------------------
>> ls
ccmatrix_ptest        mytest                ptest.m
ccmatrix_ptest.m      ptest
```

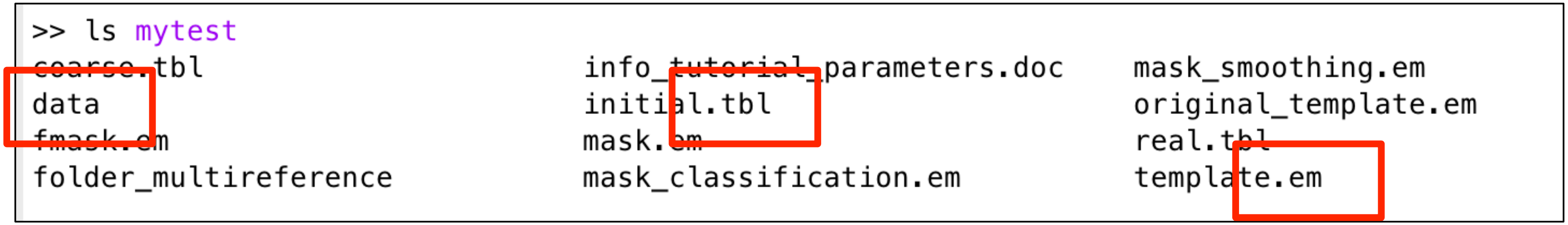
Extension .m:
a Matlab script

Also notice the *Dynamo* command `dsummary which` reports the *Dynamo*-relevant contents found in a folder:



```
------------------------------------------------------------
>> dsummary . on
------------------------------------------------------------

 Gathering a summary of Dynamo contents in folder "."

projects                 : 2
                           ./ccmatrix_ptest
                           ./ptest
 - multireference        : 0
 - ccmatrix              : 1
                           ./ccmatrix_ptest
data folders             : 0
subboxing folders        : 0
bundles                  : 0
compaction folders       : 0
sequence folders         : 0
tutorial folders         : 1
                           ./mytest

  [ok] summary

------------------------------------------------------------
>> ls
```

Let’s see what is inside what *Dynamo* marks as a tutorial folder



```
>> ls mytest
coarse.tbl                        info_tutorial_parameters.doc    mask_smoothing.em
data                              initial.tbl                     original_template.em
fmask.em                          mask.em                         real.tbl
folder_multireference             mask_classification.em          template.em
```

`data` folder is a directory where the subtomograms have been stored as individual files

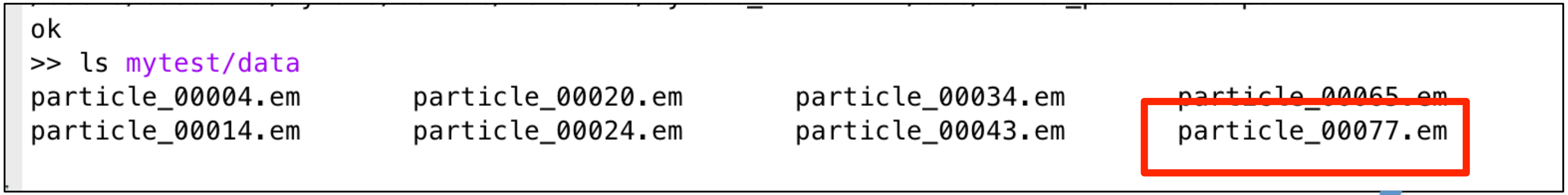
`.tbl` files are tables (metadata objects)

a real.tbl that actually algn the particles, provided for demonstration purposes

`.em` files are 3d volumes

Files suggested to be used as initial reference, mask, fourier mask, etc…

DATA folders



```
ok
>> ls mytest/data
particle_00004.em       particle_00020.em       particle_00034.em       particle_00065.em
particle_00014.em       particle_00024.em       particle_00043.em       particle_00077.em
```

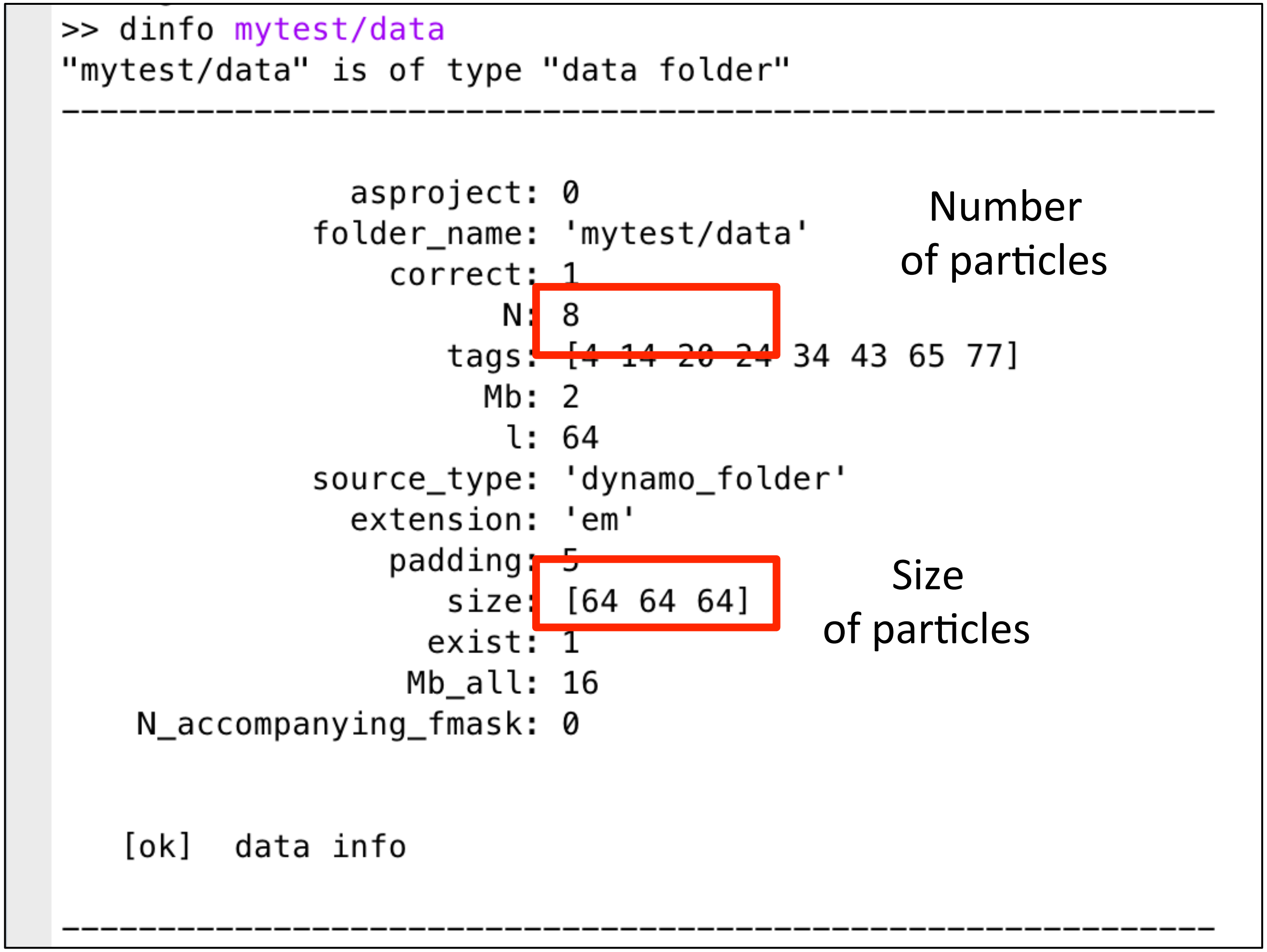
Dynamo needs the particles named after the convention:

`<folder_name>/particle_XXXXX.em.`

*The integer that identifies a particle is called **tag** in the Dynamo documentation.*

- Any number of integers, left padded with zeros.
- Extensions .mrc, .spi, .vol and .em are accepted
- For very large data sets (>100k subtomograms), other formats can be used: `dData`

The formatting data tutorial provides more exhaustive information about ways to ease the generation of this data folders `dinfo` will print a quick check on the contest of a data folder



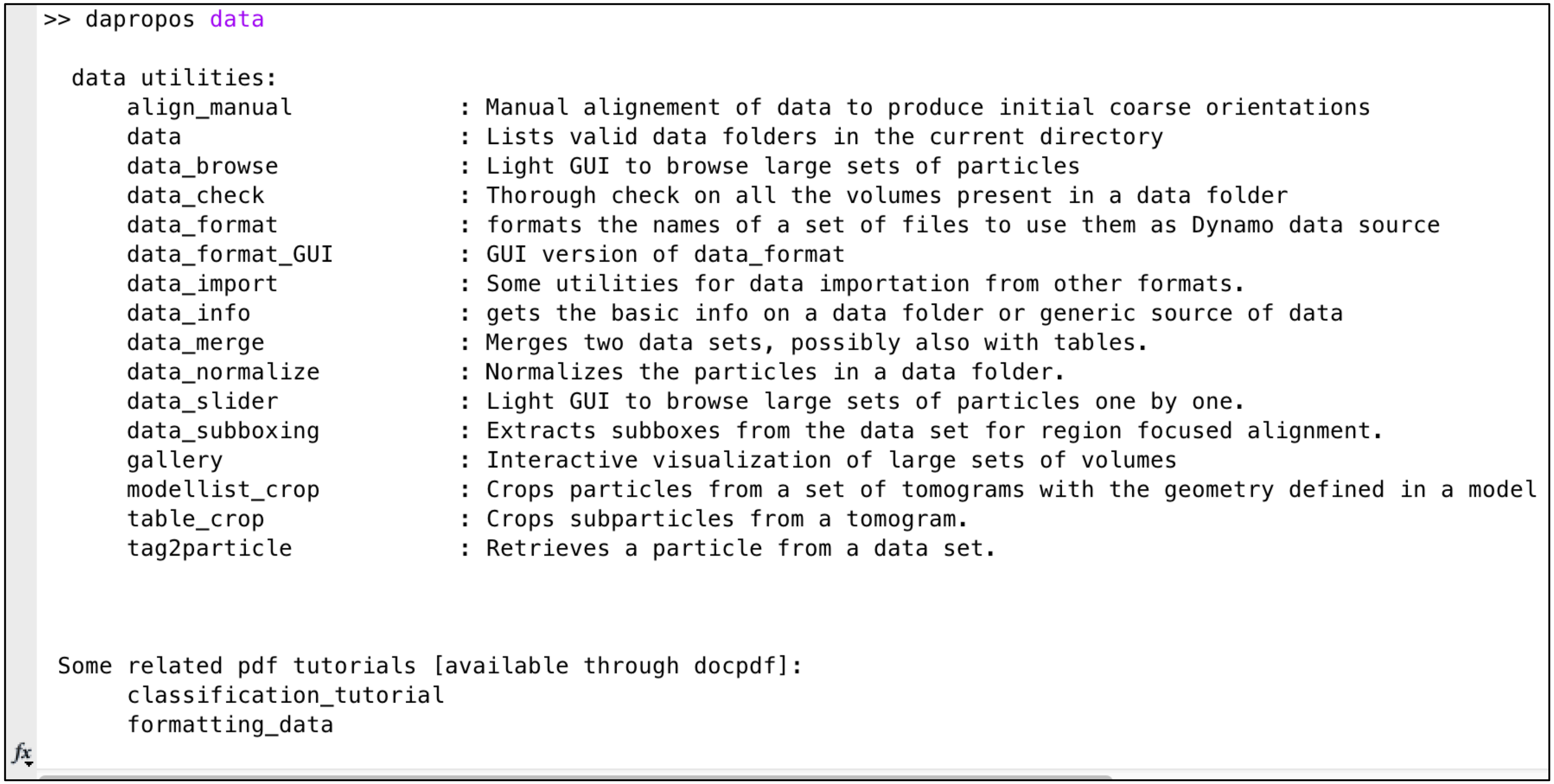
```
>> dinfo mytest/data
"mytest/data" is of type "data folder"
------------------------------------------------------------


                asproject: 0
              folder_name: 'mytest/data'
                  correct: 1
                        N: 8
                     tags: [4 14 20 24 34 43 65 77]
                       Mb: 2
                        l: 64
              source_type: 'dynamo_folder'
                extension: 'em'
                  padding: 5
                     size: [64 64 64]
                    exist: 1
                   Mb_all: 16
     N_accompanying_fmask: 0


   [ok]  data info

------------------------------------------------------------
```

Number
of particles

Size
of particles

`dinfo` is a general command that will work on other types of objects: tables, projects, etc

You can check the different commands related to management of data folders



```
>> dapropos data

  data utilities:
      align_manual            : Manual alignement of data to produce initial coarse orientations
      data                    : Lists valid data folders in the current directory
      data_browse             : Light GUI to browse large sets of particles
      data_check              : Thorough check on all the volumes present in a data folder
      data_format             : formats the names of a set of files to use them as Dynamo data source
      data_format_GUI         : GUI version of data_format
      data_import             : Some utilities for data importation from other formats.
      data_info               : gets the basic info on a data folder or generic source of data
      data_merge              : Merges two data sets, possibly also with tables.
      data_normalize          : Normalizes the particles in a data folder.
      data_slider             : Light GUI to browse large sets of particles one by one.
      data_subboxing          : Extracts subboxes from the data set for region focused alignment.
      gallery                 : Interactive visualization of large sets of volumes
      modellist_crop          : Crops particles from a set of tomograms with the geometry defined in a model
      table_crop              : Crops subparticles from a tomogram.
      tag2particle            : Retrieves a particle from a data set.



 Some related pdf tutorials [available through docpdf]:
      classification_tutorial
      formatting_data
```

`dynamo_data_check` (shortened to `ddcheck`) for instance, scans for the contents of e particle inside the data folder.

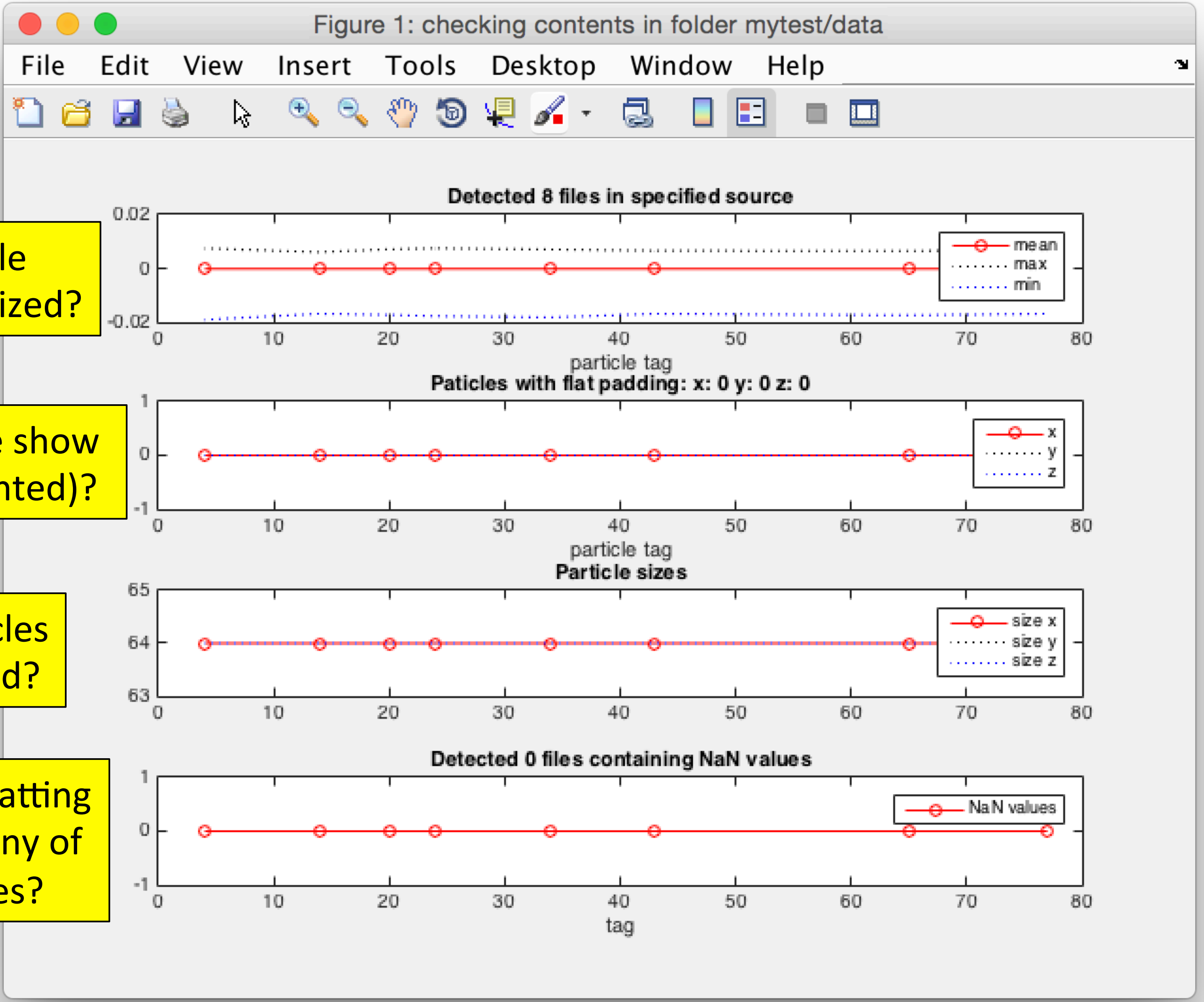


Figure 1: checking contents in folder mytest/data
File Edit View Insert Tools Desktop Window Help
Detected 8 files in specified source
0.02
0
-0.02
0 10 20 30 40 50 60 70 80
particle tag
mean
max
min
Paticles with flat padding: x: 0 y: 0 z: 0
1
0
-1
0 10 20 30 40 50 60 70 80
particle tag
x
y
z
Particle sizes
65
64
63
0 10 20 30 40 50 60 70 80
size x
size y
size z
Detected 0 files containing NaN values
1
0
-1
0 10 20 30 40 50 60 70 80
tag
NaN values

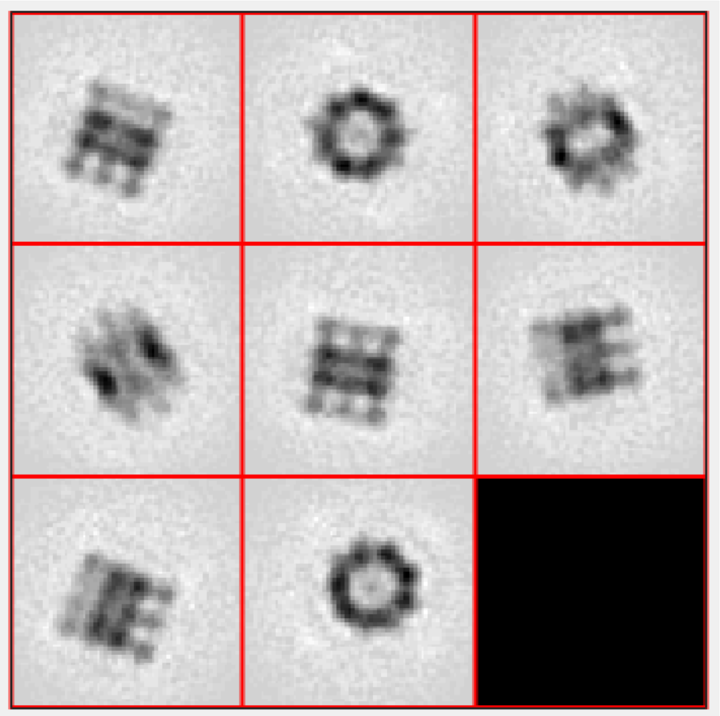
is each particle correctly normalized?

does any particle show flat areas (unwanted)?

Are the particles correctly sized?

is there any formatting problem inside any of the particle files?

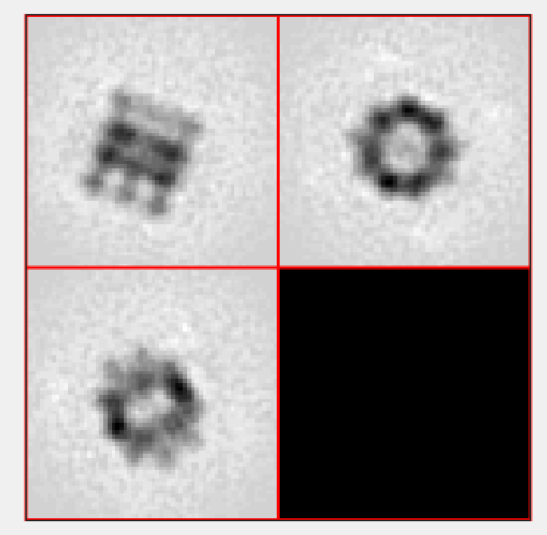
A data folder name is used by several *Dynamo* commands (as `dslices` in this example) to refer collectively to all the 3d particles in a data folder:



```
dslices mytest/data -jz *;
```

“project [j] all slices [*] along the z direction"

In these commands, the flag ‘tags’ is normally Used to refer to a subset of particles



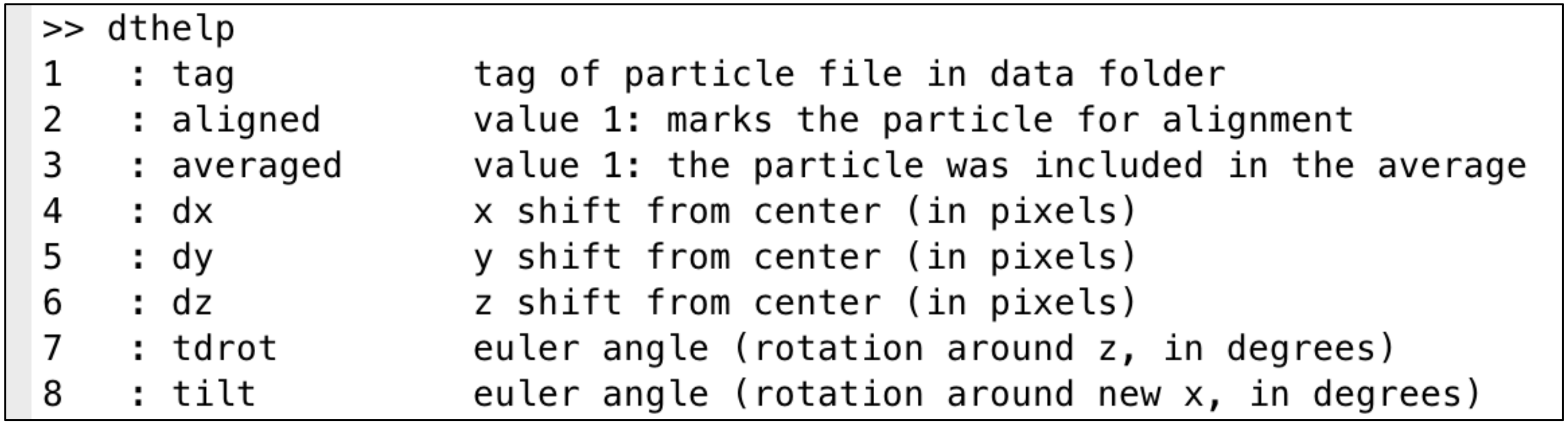
```
dslices mytest/data -jz * -tags [4:20];
```

Table files

Table files have several uses.
In their simple form, they are related to a data folder.
 A table is simply a matrix that contains

- N rows [for N particles]
- P columns [for P described properties]

The convention that maps column numbers to properties is predefined, and can be consulted in the data center (or with the `dthelp` command)



```
>> dthelp
1   : tag            tag of particle file in data folder
2   : aligned        value 1: marks the particle for alignment
3   : averaged       value 1: the particle was included in the average
4   : dx             x shift from center (in pixels)
5   : dy             y shift from center (in pixels)
6   : dz             z shift from center (in pixels)
7   : tdrot          euler angle (rotation around z, in degrees)
8   : tilt           euler angle (rotation around new x, in degrees)
```

Column 1 is the “**tag**” that identifies individual particle numbers.
This is the link between a row in a table and a file in a table folder

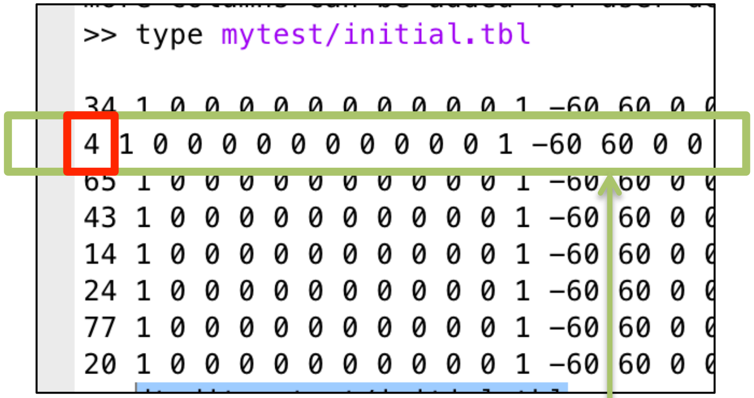
A table is normally stored as a text file,
So it can be inspected with any usual tool

```
type mytest/initial.tbl
```

or

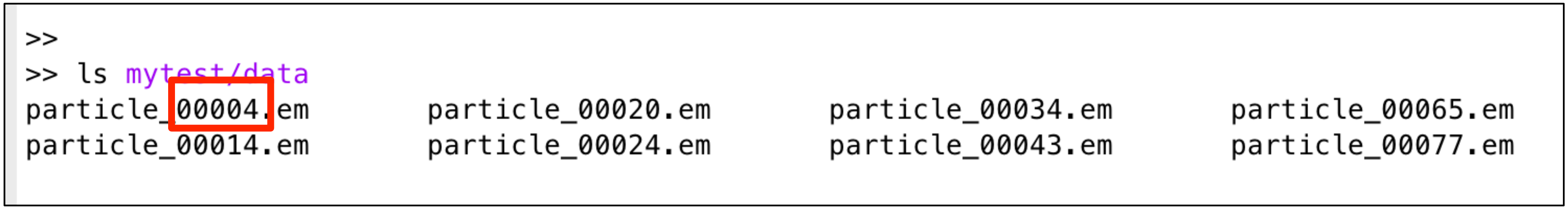
```
!cat mytest/initial.tbl
```

In the standalone version



```
>> type mytest/initial.tbl

34 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
4 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
65 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
43 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
14 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
24 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
77 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
20 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0
```

This row in the table will store the metadata for a particle with tag 4’ in a data folder

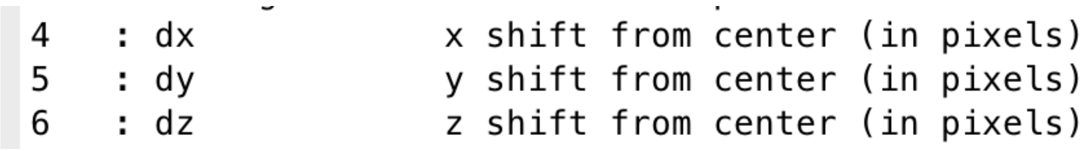


```
>>
>> ls mytest/data
particle_00004.em        particle_00020.em        particle_00034.em        particle_00065.em
particle_00014.em        particle_00024.em        particle_00043.em        particle_00077.em
```

Which information is coded inside a table?

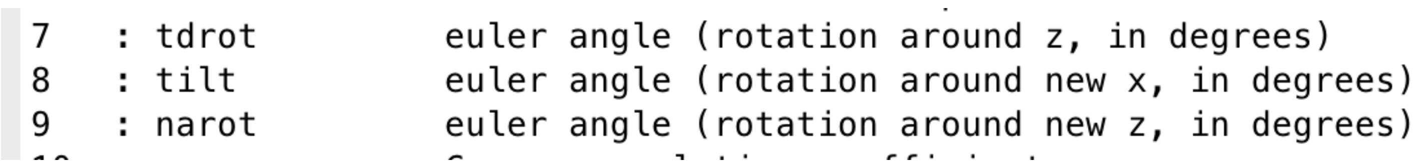
Most important are *alignment parameters:*

Columns 4 to 6: **shifts**



```
4   : dx              x shift from center (in pixels)
5   : dy              y shift from center (in pixels)
6   : dz              z shift from center (in pixels)
```

Columns 7 to 9: **rotation** (Euler angles)



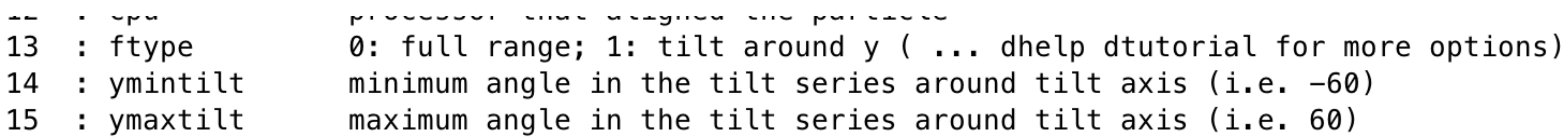
```
7   : tdrot           euler angle (rotation around z, in degrees)
8   : tilt            euler angle (rotation around new x, in degrees)
9   : narot           euler angle (rotation around new z, in degrees)
```

When you start a project for alignment in Dynamo, you need to provide a table with initial guesses for these parameters.
Leave them to zero if you don't have any a priori informatio

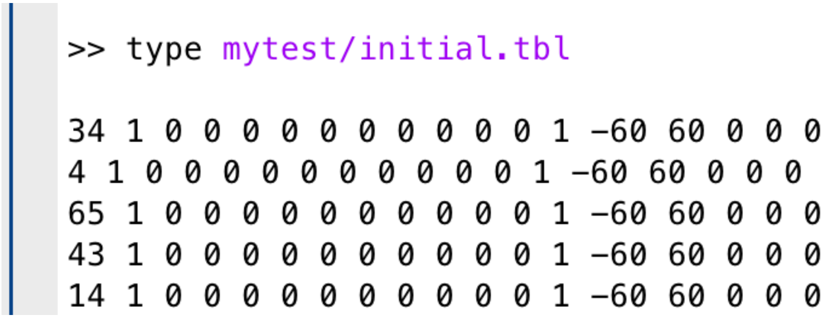
The final result of a *Dynamo* alignment project will be a table that contains the alignment parameters for each particle.

The next most important information that you need to provide is the ”fourier type”

Column 13 defines the “ftype" (fourier type): a description of the geometry.
Type 1 is the most usual one: electron beam comes along z, tilt axis is in y.
For this type, columns 14 and 15 describe the extent of the missing wedge.



```
12  : cpu          processor that aligned the particle
13  : ftype        0: full range; 1: tilt around y ( ... dhelp dtutorial for more options)
14  : ymintilt     minimum angle in the tilt series around tilt axis (i.e. -60)
15  : ymaxtilt     maximum angle in the tilt series around tilt axis (i.e. 60)
```

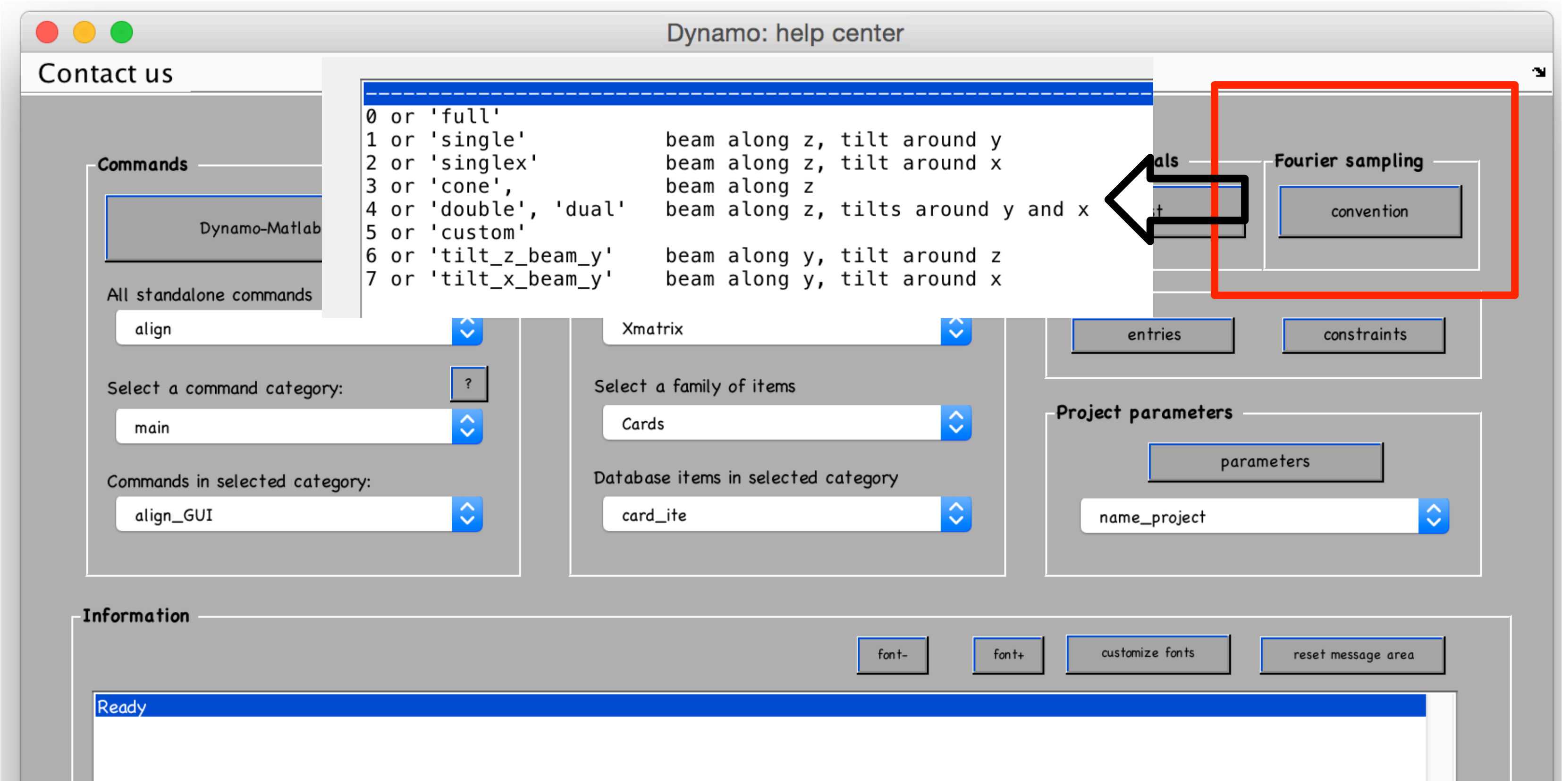


```
>> type mytest/initial.tbl

34 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0 0
4 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0 0
65 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0 0
43 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0 0
14 1 0 0 0 0 0 0 0 0 0 0 1 -60 60 0 0 0
```

Here, each particle is noted to be of ftype 1, with a missing wedge between -60 and 60 deg.

For advanced users:
it is possible to define your own fourier type (ftype =5), as long as you pass a file in the data folder for each particle.

The coding of the fourier type is one of the conventions listed in the dhelp Help Center of Dynamo



```
---------------------------------------------------------------
0 or 'full'
1 or 'single'           beam along z, tilt around y
2 or 'singlex'          beam along z, tilt around x
3 or 'cone',            beam along z
4 or 'double', 'dual'   beam along z, tilts around y and x
5 or 'custom'
6 or 'tilt_z_beam_y'    beam along y, tilt around z
7 or 'tilt_x_beam_y'    beam along y, tilt around x
```

All standalone commands: align

Select a command category: main

Commands in selected category: align_GUI

Xmatrix

Select a family of items: Cards

Database items in selected category: card_ite

name_project

Ready

Other entries are provided by *Dynamo* during the alignment (use `dhelp`)

```
10  : cc                Cross correlation coefficient
```

… related to cropping particles from tomograms…

```
20  : tomo            tomogram number
24   : x                x coordinate in original volume
25   : y                y coordinate in original volume
26   : z                z coordinate in original volume
```

… related to classification tasks….

```
41   : eig1             "eigencoefficient" #1
42   : eig2             "eigencoefficient" #2
```

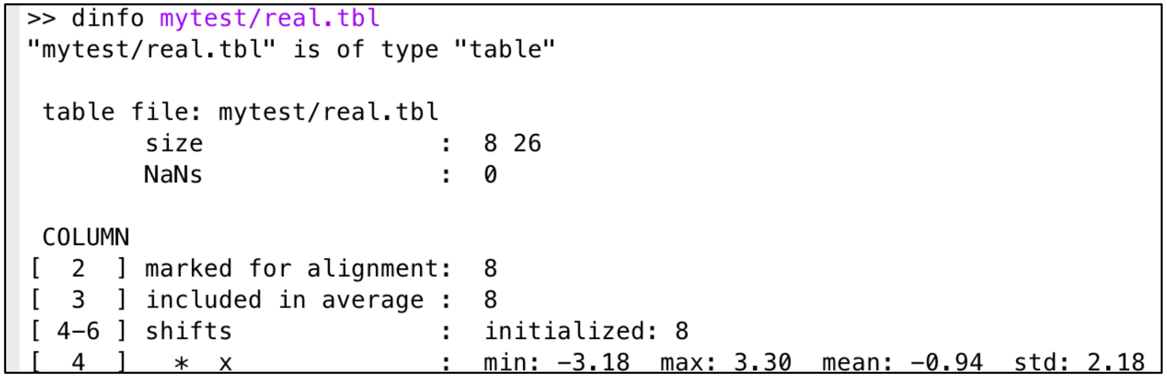
… or foreseen for user annotations….

```
21 : reg              for arbitrary region assignations
23 : annotation        arbitrary annotations
```

Check on dapropos table to see the commands to manage tables.
Probably, the most useful ones will be:

dtgrep selects subsets of particles inside a table using different criteria

dinfo
prints summaries
of the content of a table
for each column



```
>> dinfo mytest/real.tbl
"mytest/real.tbl" is of type "table"

 table file: mytest/real.tbl
        size                 :  8 26
        NaNs                 :  0

 COLUMN
[  2  ] marked for alignment:  8
[  3  ] included in average :  8
[ 4-6 ] shifts               :  initialized: 8
[  4  ]   *  x               :  min: -3.18  max: 3.30  mean: -0.94  std: 2.18
```

dtshow To quickly plot contents inside a table (dtview for a more complex browser)

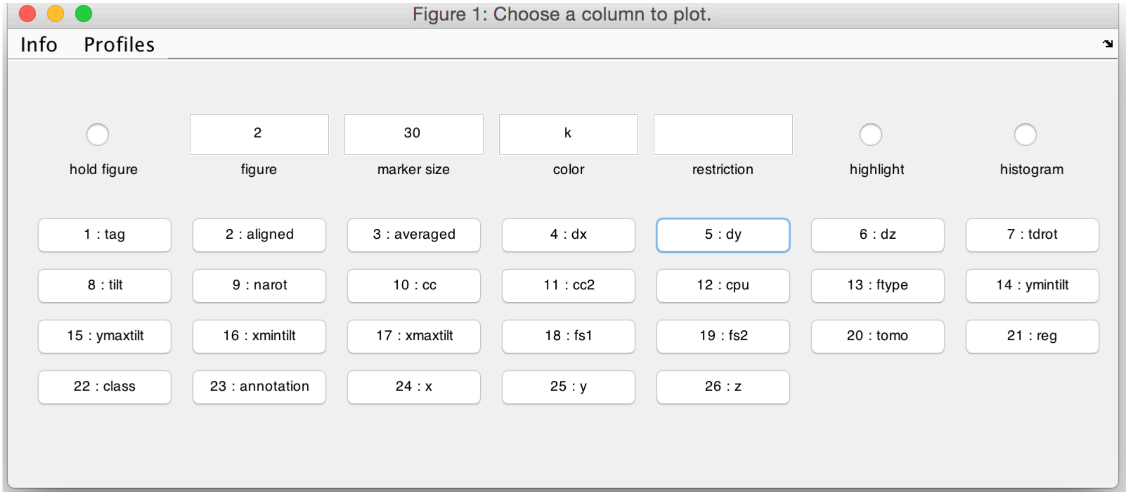


Figure 1: Choose a column to plot.

○ hold figure

figure: 2

marker size: 30

color: k

restriction:

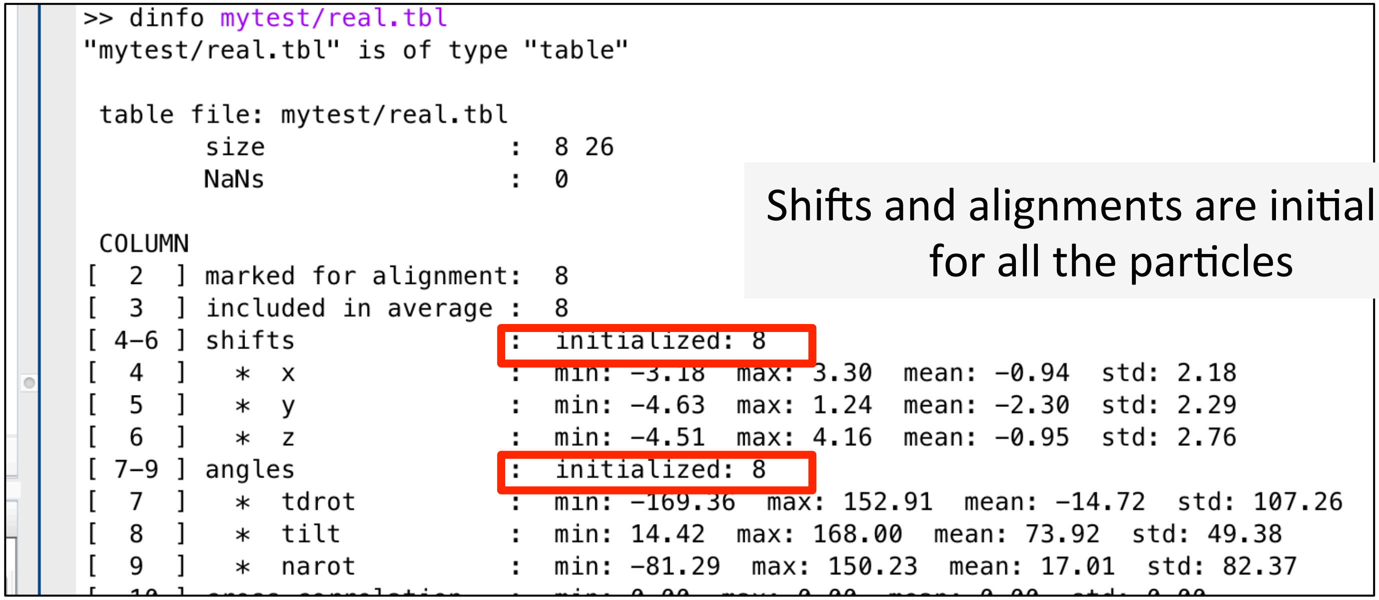
○ highlight

○ histogram

Combining TABLE and DATA

Tables are interesting as they express the alignment of a data set.

The `dtutorial` command provided use with a `real.tbl` table, that contains the actual alignment parameters of the provided synthetic data set.



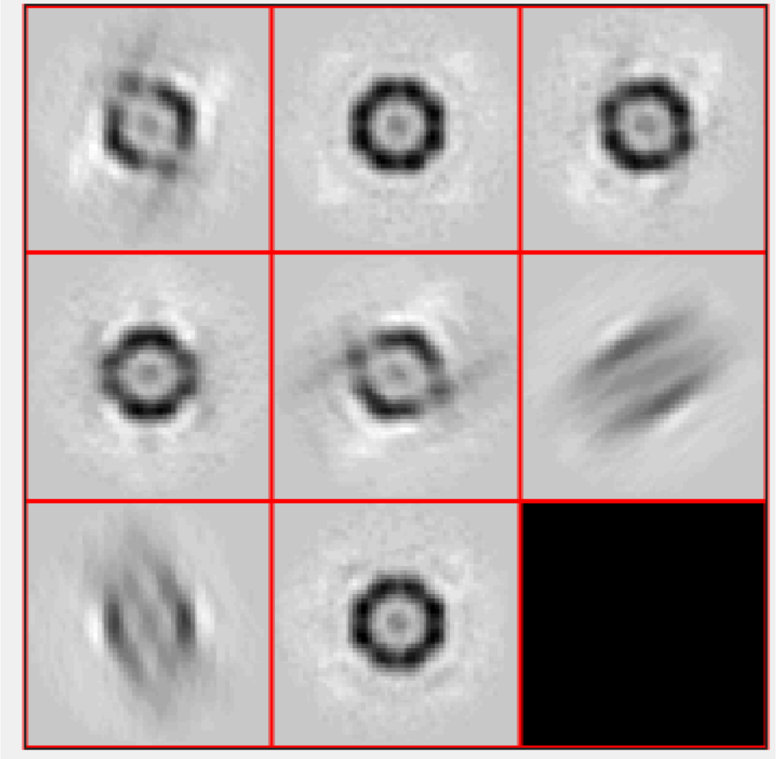
```
>> dinfo mytest/real.tbl
"mytest/real.tbl" is of type "table"

 table file: mytest/real.tbl
        size                 :  8 26
        NaNs                 :  0

 COLUMN
[  2  ] marked for alignment:  8
[  3  ] included in average :  8
[ 4-6 ] shifts               :  initialized: 8
[  4  ]   *  x               :  min: -3.18  max: 3.30   mean: -0.94   std: 2.18
[  5  ]   *  y               :  min: -4.63  max: 1.24   mean: -2.30   std: 2.29
[  6  ]   *  z               :  min: -4.51  max: 4.16   mean: -0.95   std: 2.76
[ 7-9 ] angles               :  initialized: 8
[  7  ]   *  tdrot           :  min: -169.36  max: 152.91   mean: -14.72   std: 107.26
[  8  ]   *  tilt            :  min: 14.42  max: 168.00   mean: 73.92   std: 49.38
[  9  ]   *  narot           :  min: -81.29  max: 150.23   mean: 17.01   std: 82.37
[  10 ] cross correlation   :  min: 0.00  max: 0.00  mean: 0.00  std: 0.00
```

Shifts and alignments are initialized for all the particles

So, let's see how the data set looks when the table is used to align the particles

We use the previously shown command `dslices`, adding:
a table (`-t mytest/real.tbl`) and
the order of use it to align the particles (`-align on`)

```
dslices mytest/data -jz *  -t mytest/real.tbl -align on
```



Now Dynamo shows all [*] the sections of each particle along the [z] direction after alignment

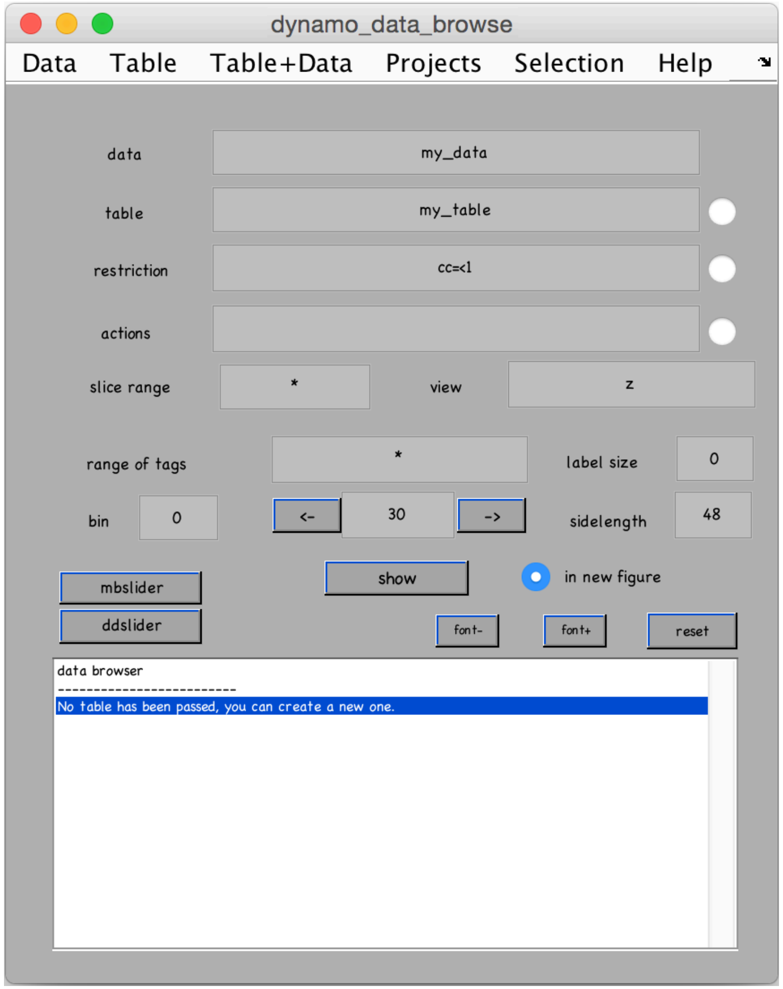
Notice how the missing wedge
Works differently for different particles
depending on their original orientation

The most often way of looking at aligned particles is with the command `ddbrowse`.

`ddbrowse` is a lightweight GUI for exploration of subsets of particles in a data set.

Many Dynamo GUIs will launch instances of this command when you are running preset workflows, but you can also launch it independently.

```
>>  ddbrowse
```



data: my_data
table: my_table ○
restriction: cc=<1 ○
actions: ○
slice range: * view: z
range of tags: * label size: 0
bin: 0 30 sidelength: 48
● in new figure

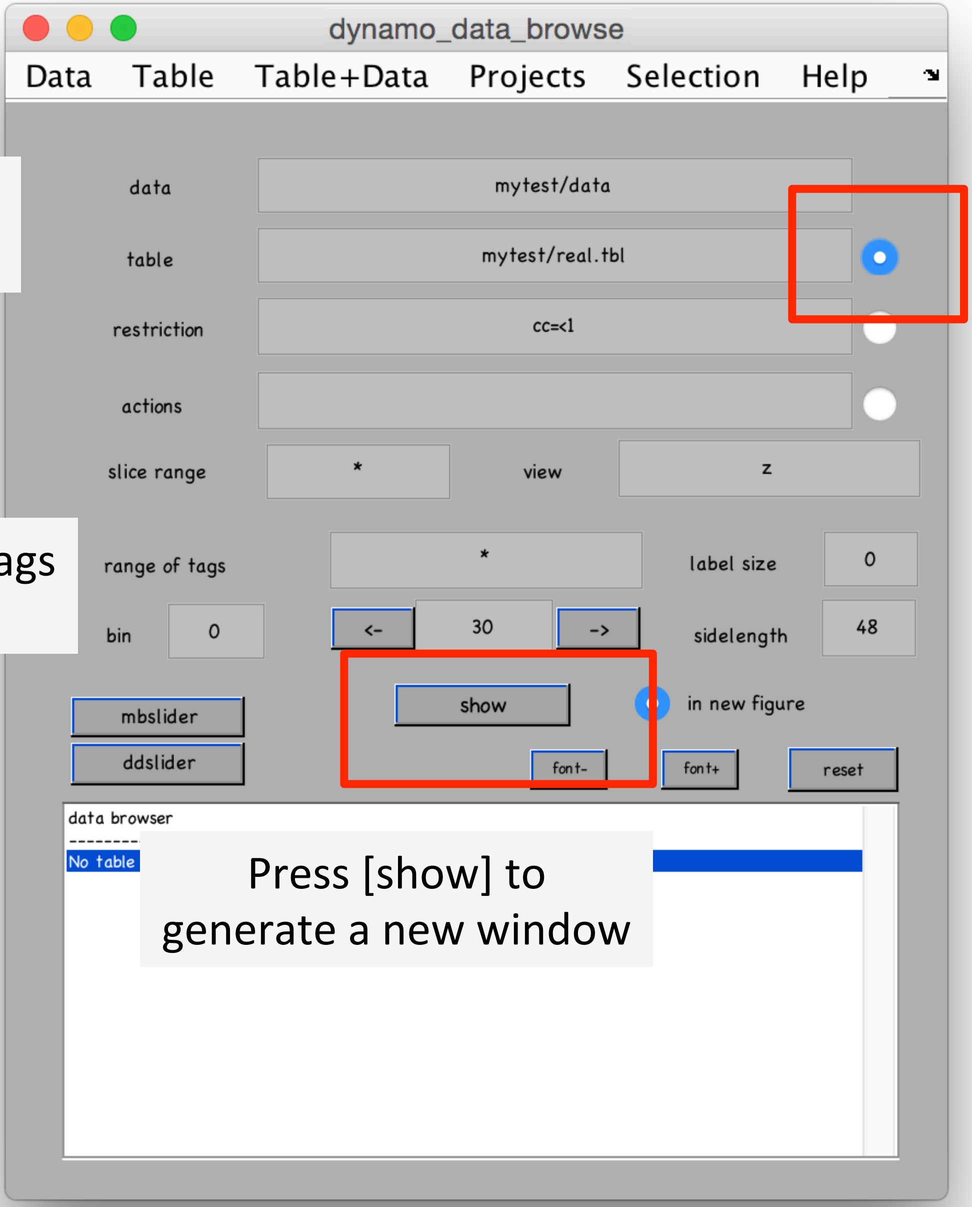
```
data browser
-------------------------
No table has been passed, you can create a new one.
```

or you can launch it

```
>>  ddbrowse -d mytest/data -t mytest/real.tbl
```

Enter table and data

Select a range of tags (* for all)



data: mytest/data

table: mytest/real.tbl ●

restriction: cc=<1 ○

actions: ○

slice range: *

view: z

range of tags: *

label size: 0

bin: 0

30

sidelength: 48

● in new figure

data browser

No table

Press [show] to generate a new window

Switch on the table if you want to see the aligned particles (otherwise they will be shown as contained in the raw files).

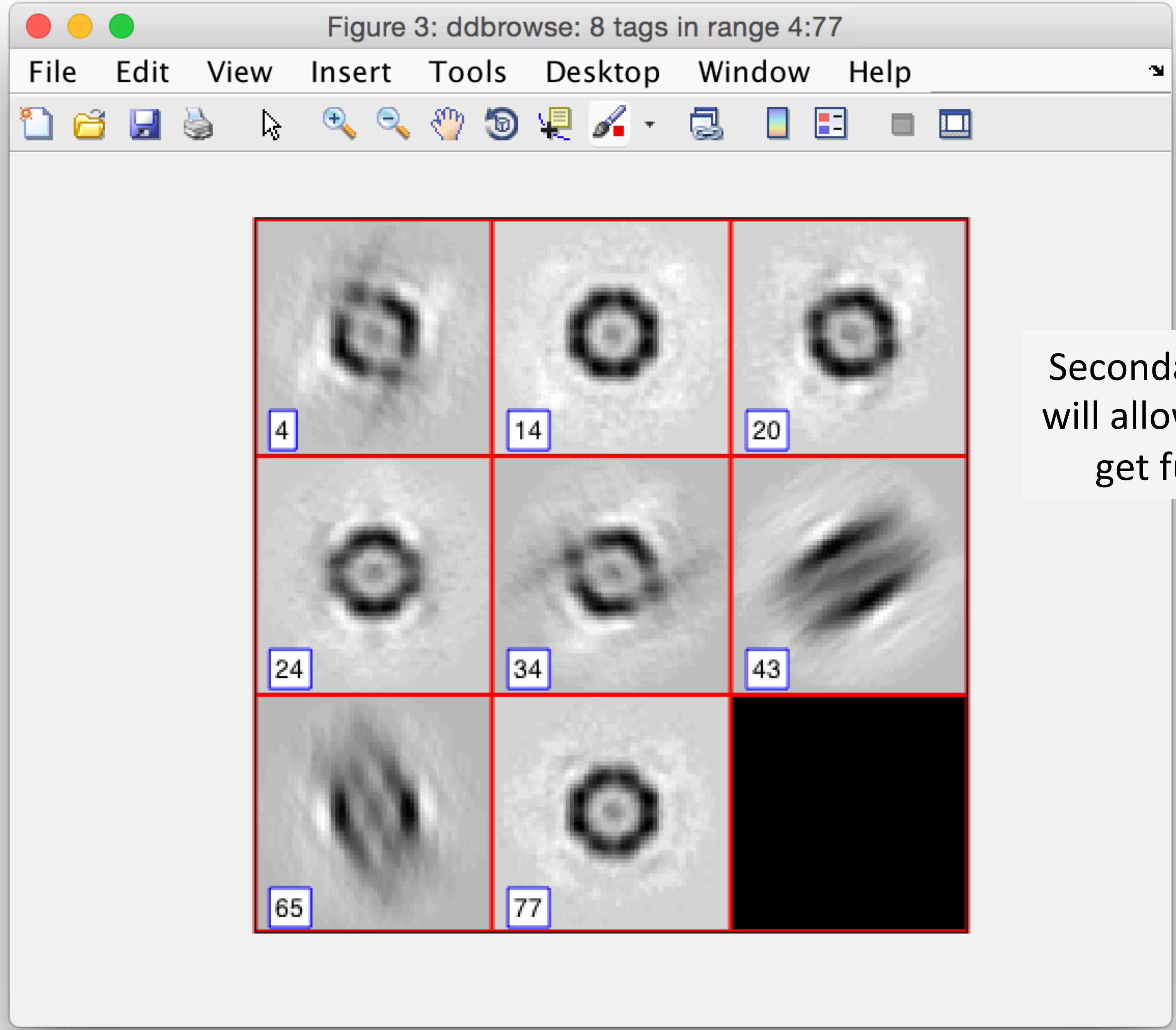


Figure 3: ddbrowse: 8 tags in range 4:77
File Edit View Insert Tools Desktop Window Help
4
14
20
24
34
43
65
77

Secondary click on a particle will allow you to select it and get further information

Note: you can get labels with the tag number with the [label size] entry

You can access the metadata of an individual particle:

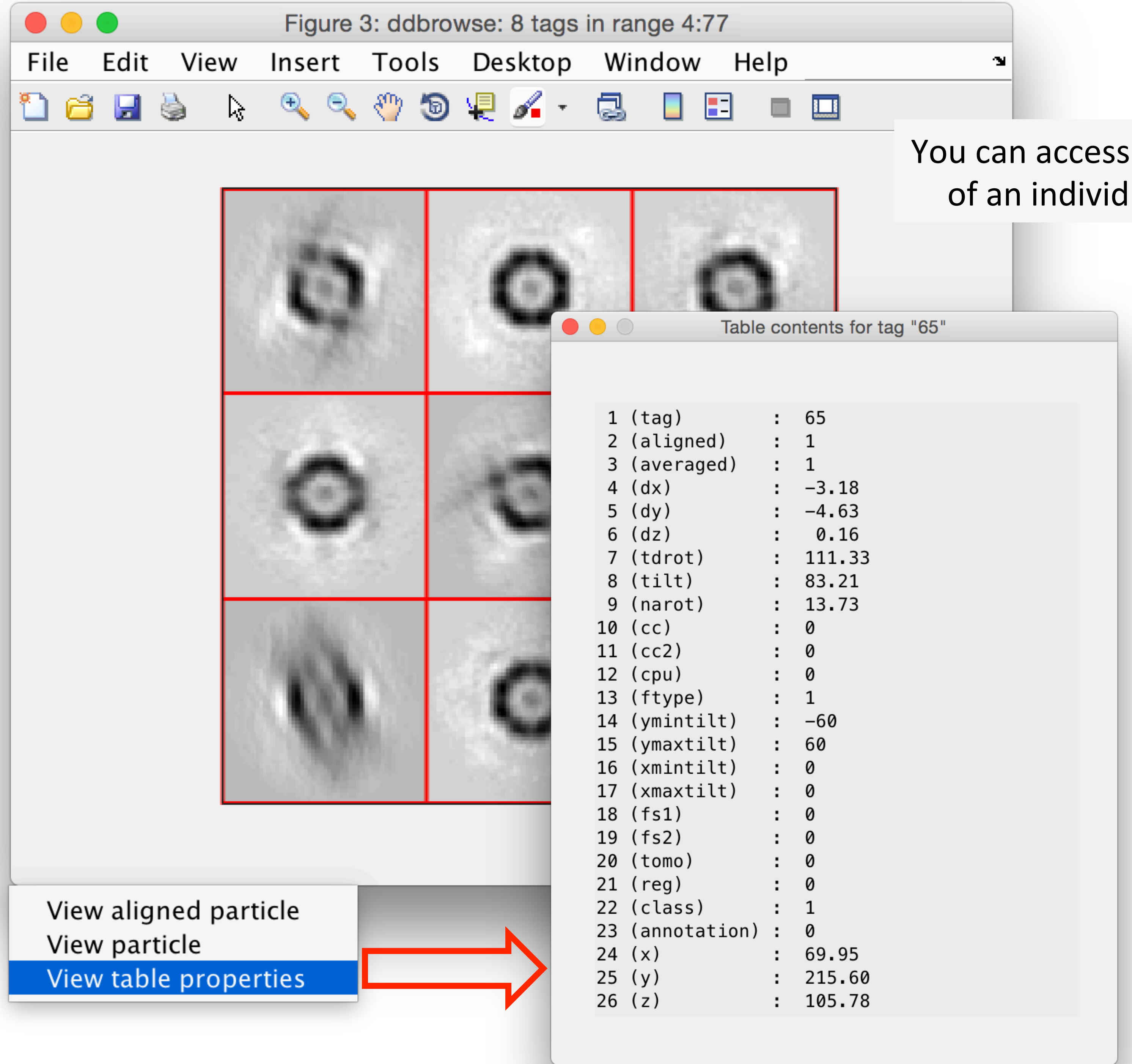
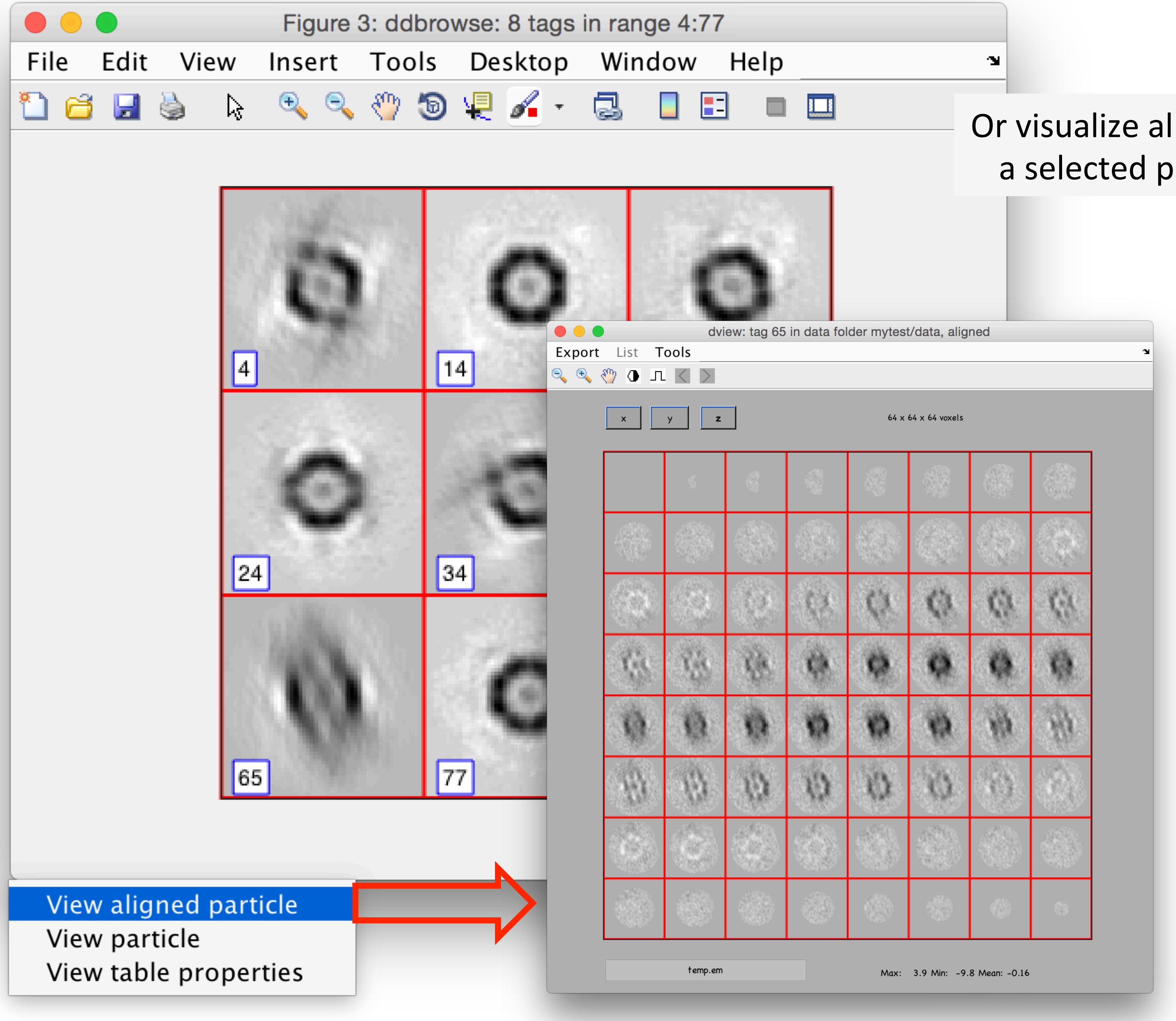


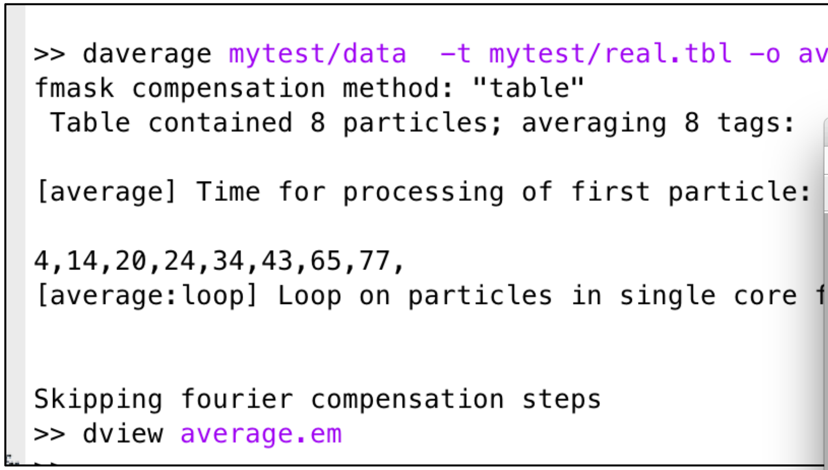
Figure 3: ddbrowse: 8 tags in range 4:77
File Edit View Insert Tools Desktop Window Help
View aligned particle
View particle
View table properties
Table contents for tag "65"

```
 1 (tag)        :  65
 2 (aligned)    :  1
 3 (averaged)   :  1
 4 (dx)         :  -3.18
 5 (dy)         :  -4.63
 6 (dz)         :   0.16
 7 (tdrot)      :  111.33
 8 (tilt)       :  83.21
 9 (narot)      :  13.73
10 (cc)         :  0
11 (cc2)        :  0
12 (cpu)        :  0
13 (ftype)      :  1
14 (ymintilt)   :  -60
15 (ymaxtilt)   :  60
16 (xmintilt)   :  0
17 (xmaxtilt)   :  0
18 (fs1)        :  0
19 (fs2)        :  0
20 (tomo)       :  0
21 (reg)        :  0
22 (class)      :  1
23 (annotation) :  0
24 (x)          :  69.95
25 (y)          :  215.60
26 (z)          :  105.78
```

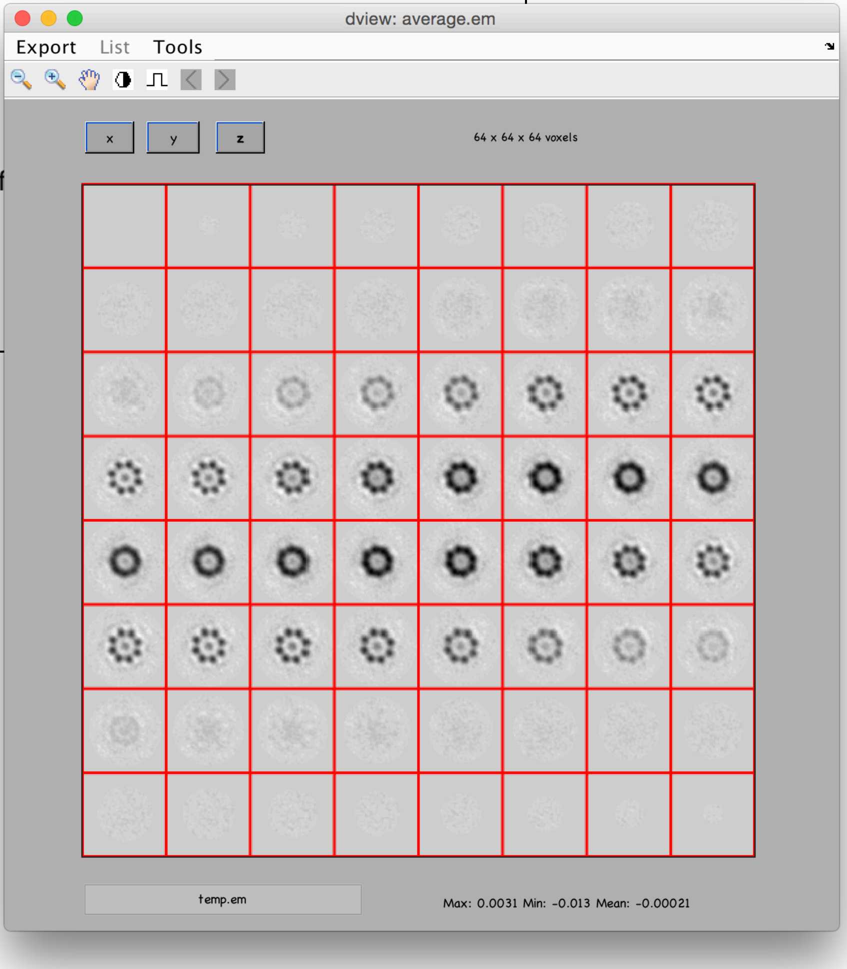
Or visualize all slices of a selected particle:



Obviously the final goal of an alignment is creating an average from the aligned data set. Our last command is `daverage`.



```
>> daverage mytest/data  -t mytest/real.tbl -o average.em
fmask compensation method: "table"
 Table contained 8 particles; averaging 8 tags:

[average] Time for processing of first particle:

4,14,20,24,34,43,65,77,
[average:loop] Loop on particles in single core f


Skipping fourier compensation steps
>> dview average.em
```



Note that this is a handy way to run sanity checks on the coherence of your table and data!
(when providing coarse initial angles or after running a format conversion)

Suggested further material

Deepening into using tables and data folders
PDF tutorial: commandline.pdf
(available through `dynamo_docpdf`)

Creating alignment projects:
PDF tutorial: introduction2AlignmentProjects.pdf