



Multireference alignment
Use of plugins

Multireference and plugins

You have probably noticed the [nref] boxes foreseen in project_manager to setup multireference projects. Also, overall in *Dynamo* you have also seen edit fields allowing to specify a “ref” or “reference” number.

These slides show how to setup a project that includes several references, (or “reference channels”, as called in the *Dynamo* documentation).

There are two basic guidelines

1 – Each multireference channel needs its own seed files for

- * initial table
- * initial template
- * initial fourier mask

This means that the syntax will be slightly different, as you need to pass multiple files.

This is resolved

- a) by passing the name of a folder, and using a convention to name the individual files, or
- b) by passing a .sel file, i.e., a text file that contains the names of the actual files

2- Each reference channel is an independent alignment process.

To create a real multireference experiment, the different channels need to interact with each other. There are many, many ways you might imagine such an interaction, and this is a natural battlefield for plugins.

This tutorial

In this tutorial we show how to use the *Dynamo* tutorial tool to create a correctly formatted multireference project on a provided data set, and how to connect it to a provided plugin..

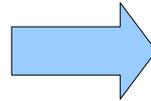
Creating the tutorial data set

open the tutorial interface
by opening first the *Dynamo* tools:

Matlab (all platforms)
>> dynamo

Standalone Linux/MacOS
\$ dynamo &

Standalone Windows
> dynamo.exe



Projects

- project manager: Main tool for project setup and management
- desktop: Projects browser and working desktop
- vpr convergence: Scans convergence history of a stored project
- ccmatrix project manager: Projects for computation of covariance matrices
- ccmatrix analysis: Classification by analysis of covariance matrix

Visualization and data browsing

- gallery: Browser for large sets of 3d particles
- tableview: Visualization of alignment results and settings
- mapview: Visualization of 3d graymaps.

Pocket tools

- data format: Formats sets of subvolumes
- align manual: Manual click for coarse alignment
- volume: Editor for volumetric transforms
- mask: Editor of solid shapes
- average: Averaging a set of data particles
- align: Alignment of two volumes
- FSC: Fourier shell correlation of two volumes
- ctfview: Contrast Transfer Function computation

User support

- help: Parameter syntax and used conventions
- tutorial: Generation of synthetic data sets and projects

for command line tools: type "doc dynamo" in your Matlab shell

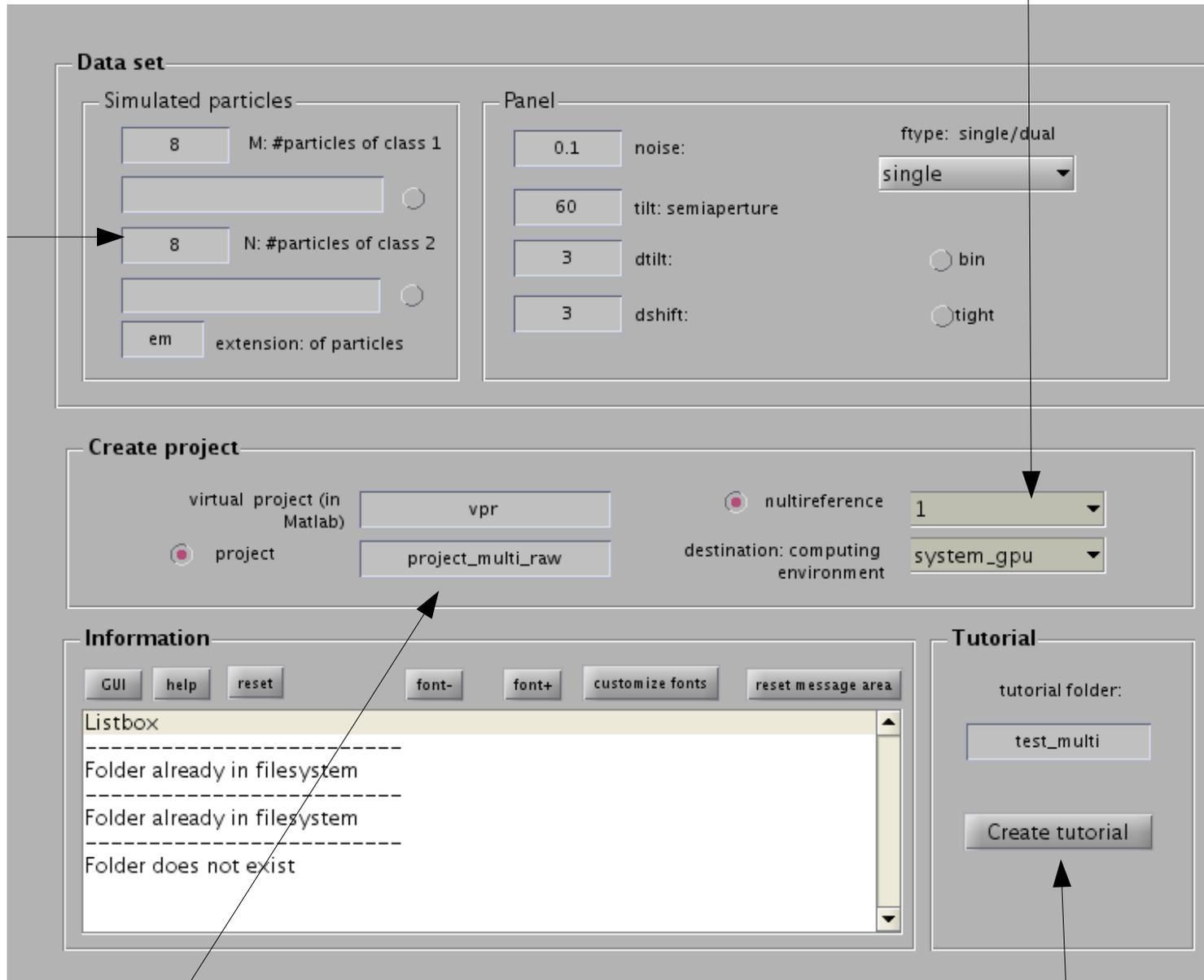
reset font- font+ fonts

initiate a tutorial ←

We introduce some changes on the standard settings:

use the multireference option (of type 1 or 2 indistintly)

use 8 particles of the "second" class



put a recognizable name for this project

... And create the tutorial

We are producing a similar data set to the one we created to get familiar with the classification tools of *Dynamo*

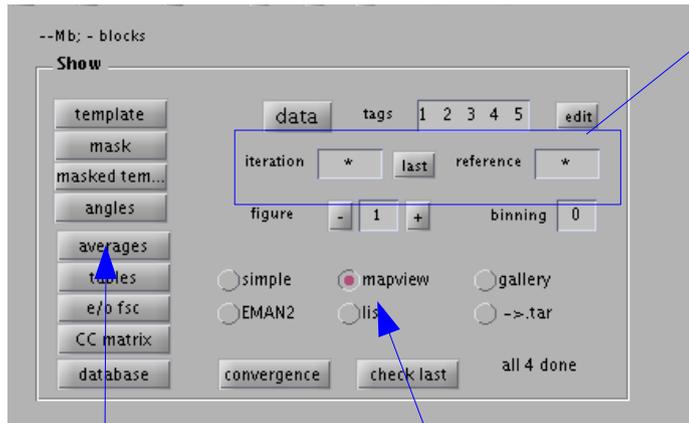
The data set will comprise 8 randomly oriented copies of a thermosome model and another 8 copies of the a slightly shrunken version of this model.

In the classification tutorial we used PCA analysis to separate the two classes.

Now, we will use a “classical” multireference approach implemented in a plugin to classify the particles by aligning them against different templates and letting them recognize the template that yields the best score.

Take a look on the results:

1) select all (*) references and all (iterations)

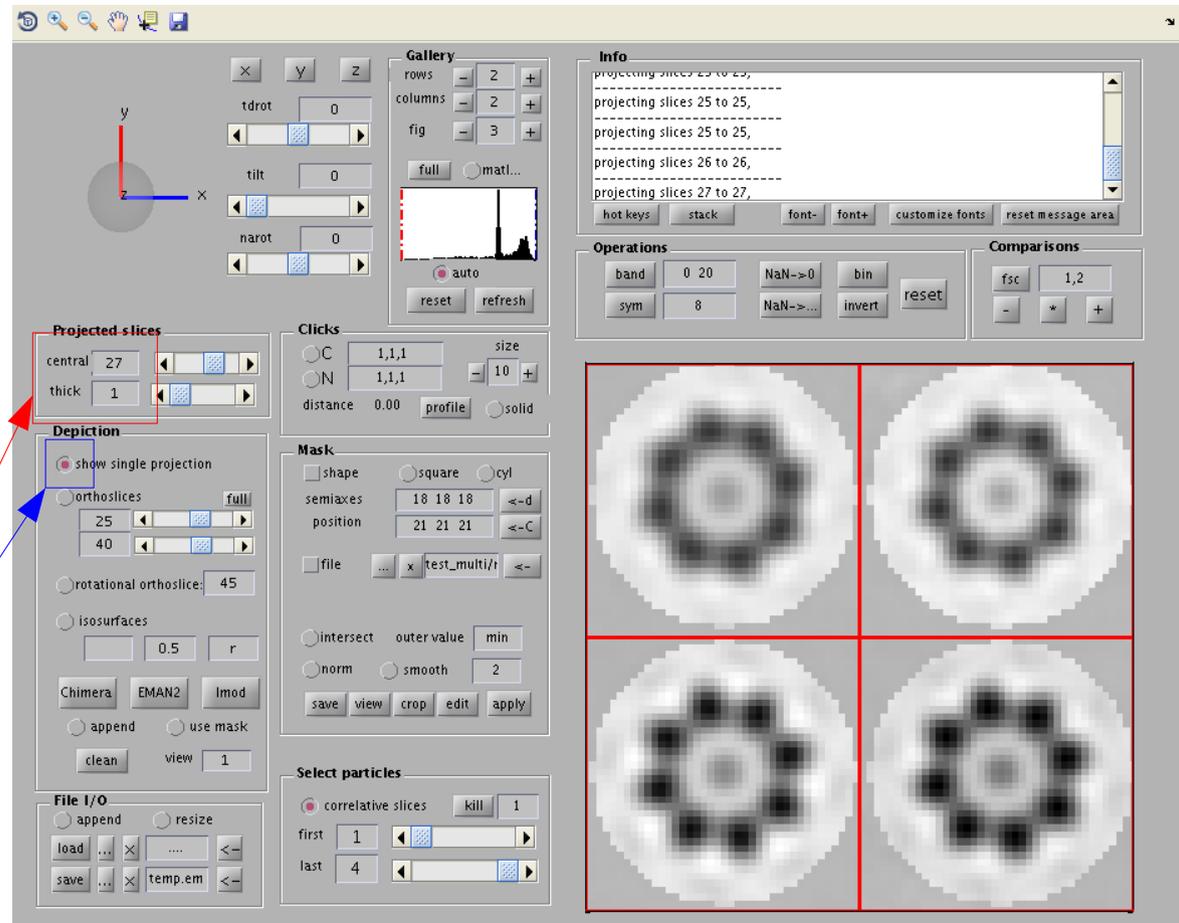


2) select mapview as viewer

3) press on [averages]

inside mapview use these viewing settings

now, each row is the result of one iteration
each column is the result for one reference



The results do not look very impressive: the averages provided by both "references channels" look pretty much the same.

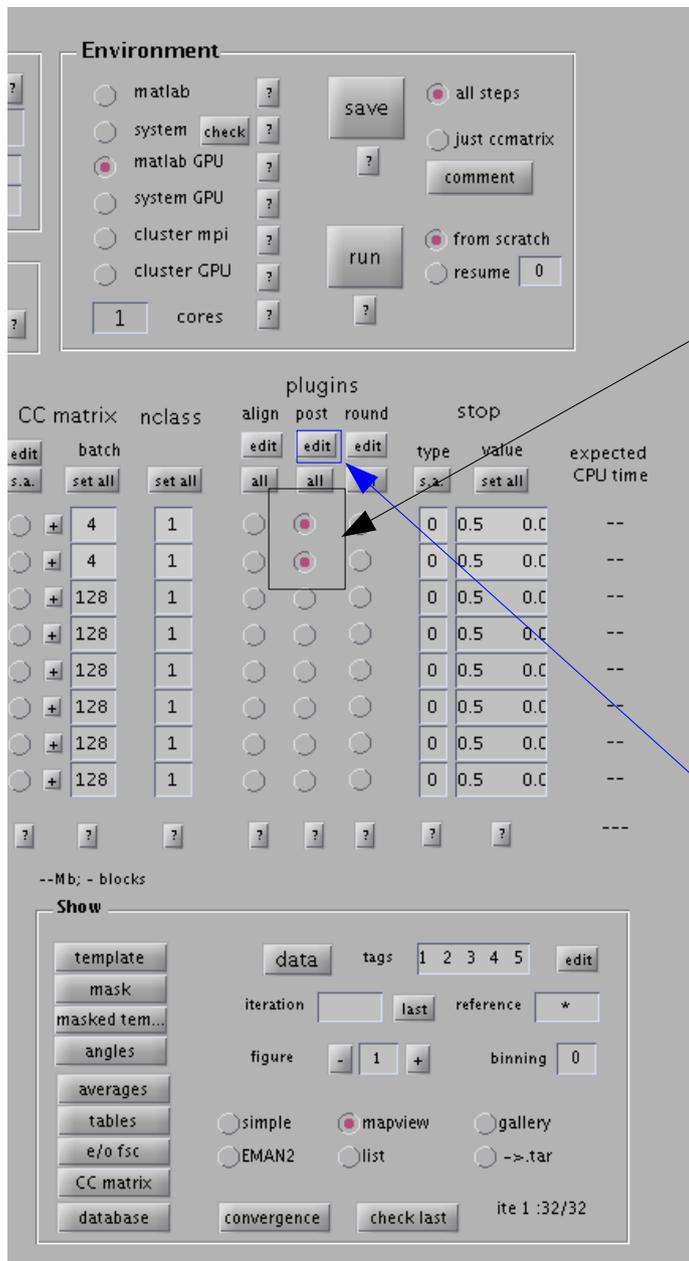
By now we have not used any real multireference approach.

If we just indicate several multireference channels, but no post-processing plugin, *Dynamo* will not try to discriminate the particles by their affinity to the different templates.

Instead, all particles will be compared to all templates, and all the new averages will comprise all particles.

Thus, the two averages will contain all the particles, although with different alignment parameters.

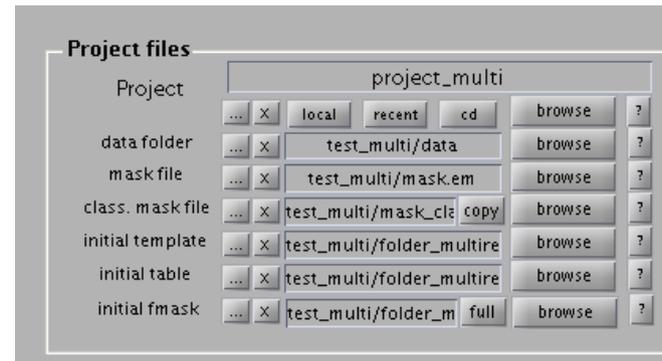
Thus, the next step is to use a plugin that considers the correlation coefficient collected by each particle against each template, and uses this information to discriminate particles.



...now let us repeat it with a real multireference police.

Open the project in the project_manager and check on the radiobuttons that activate the plugins for postprocessing.

Then provide a new name for the project, save it and run it.



while your project runs, we can take a look on the syntax used to link the plugin that we have activated

Press the [edit] button: the plugin linking editor will appear.

This editor just exposes how Dynamo will invoke the plugins during its execution pipeline.

Plugins themselves are defined elsewhere, and are just executables located somewhere in the execution Path of Matlab or the operative system.



... that's it. This particular plugin does not need any parameter, so that you only pass the name of the plugin. The () notation corresponds to Matlab notation: if you are executing a standalone version, just skip these brackets. Now, take a look on the results on the results of the “really” multireference project we've called project_multi

using the same viewing settings as in the no multireference case for the averages...

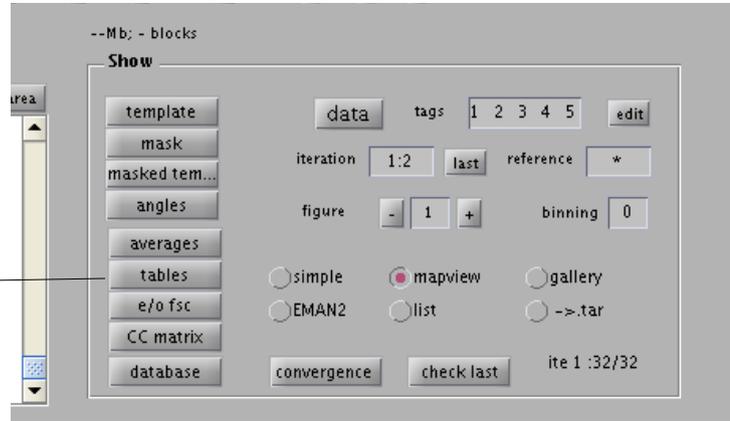
The screenshot displays a software interface with several control panels and a main display area. The top-left panel includes a 3D coordinate system (x, y, z) and sliders for 'tdrot', 'tilt', and 'narot'. The 'Gallery' panel shows 'rows' (2), 'columns' (2), and 'fig' (3), with a small histogram and 'full'/'matl...' radio buttons. The 'Info' panel on the right contains a text area with the message 'no data passed yet' and buttons for 'hot keys', 'stack', 'font-', 'font+', 'customize fonts', and 'reset message area'. Below this are 'Operations' and 'Comparisons' panels with various numerical inputs and buttons like 'band', 'sym', 'NaN->0', 'bin', 'invert', 'reset', 'fsc', and '+'. The bottom-left section contains 'Projected slices' (central: 27, thick: 1), 'Depiction' (show single projection, orthoslices: 1, 40, rotational orthoslice: 45, isosurfaces: 0.5, r), 'Mask' (shape: square, cyl; semiaxes: 18 18 18; position: 21 21 21; file: test_multi/r), and 'File I/O' (append, resize, load, save: temp.em). The bottom-right section is 'Select particles' (correlative slices: 1, first: 1, last: 4). The main display area shows a 2x2 grid of grayscale images, each depicting a circular pattern with a central spot, framed by a red border.

... we see that when we use the plugin the second iteration separates the two populations

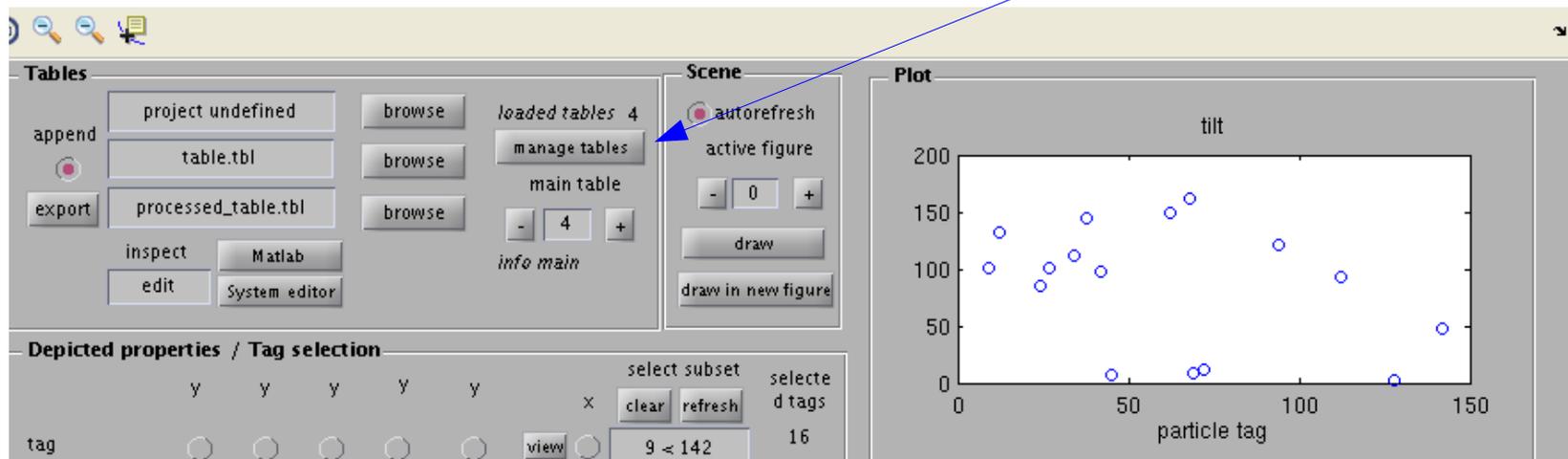
It is also instructive to take a look on the tables:

We will “cheat” using information hidden in the tables: the column 22 marks each particle with 1 or 2, signaling to which class the particle actually belongs (by construction as the tutorial tool generated them). Anyway, we want to see how the different particles behave during the iteration.

open in tableview
our four tables
(two iterations and two
references)



tableview will open containing these four files. You can check the ordering in the table manager



Listbox

font - font + fonts clear

# in memory	table name	from project	tags	first
1	./project_multi/results/ite_0001/averages/refined_table_ref_001_ite_0001.tbl	project_...	16	9
2	./project_multi/results/ite_0001/averages/refined_table_ref_002_ite_0001.tbl	project_...	16	9
3	./project_multi/results/ite_0002/averages/refined_table_ref_001_ite_0002.tbl	project_...	16	9
4	./project_multi/results/ite_0002/averages/refined_table_ref_002_ite_0002.tbl	project_...	16	9

delete table difference table c1 compute mean and std for column: - 10 + --

copy table save table ... x temporal.tbl

Restrict selected table

interval 10 using as criterion column: 22

grep 1 using as criterion column: 22

restrict to tags ... x 10 --

restrict

restrict

cancel

ok, back to GUI

ok, let us take a look on the last table, which corresponds to the second iteration, second reference:

use these settings:

focus on table 4 (i.e. ref 2, ite 2)

The screenshot displays a software interface with several panels:

- Tables:** Contains buttons for 'project undefined', 'table.tbl', 'processed_table.tbl', 'inspect', 'edit', 'Matlab', and 'System editor'. A 'loaded tables 4' section shows 'main table' with a value of 4.
- Scene:** Includes 'autorefresh', 'active figure', and 'draw' buttons.
- Depicted properties / Tag selection:** A table with columns for 'tag', 'x', 'y', 'z', 'point color', 'text label', 'select subset', and 'selected tags'. The 'class' row has 'x' and 'y' selected. A red box highlights the 'class' row.
- Plot options:** Includes 'depict only Selection', 'highlight Selection', 'size', 'only main table', 'depicted tables: 1', 'colorbar', and 'legend'.
- Scatter plot:** A plot titled '1 table(s), 2 class(es)' showing 'tag' on the x-axis (0 to 150) and 'class' on the y-axis (0.45 to 0.95). Red dots are labeled '2' and blue dots are labeled '1'. A green box highlights the 'show classes' and 'classes in column' settings.
- Plot options (bottom):** Includes 'scatterplot' mode selected, 'marker size', and 'classes in column' set to 22.

depict correlation against tag

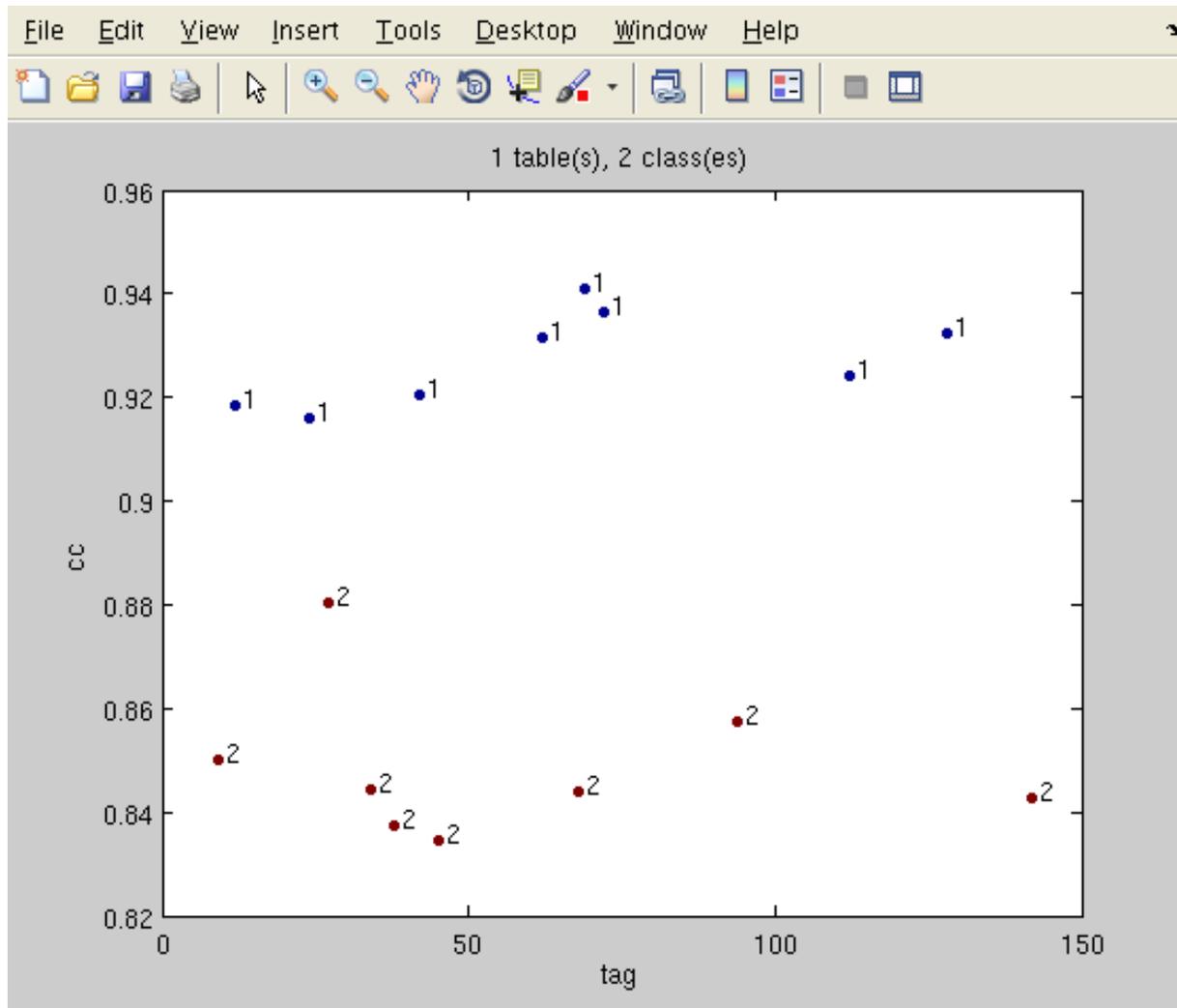
color and label depicted classes

see all classes (1 and 2)

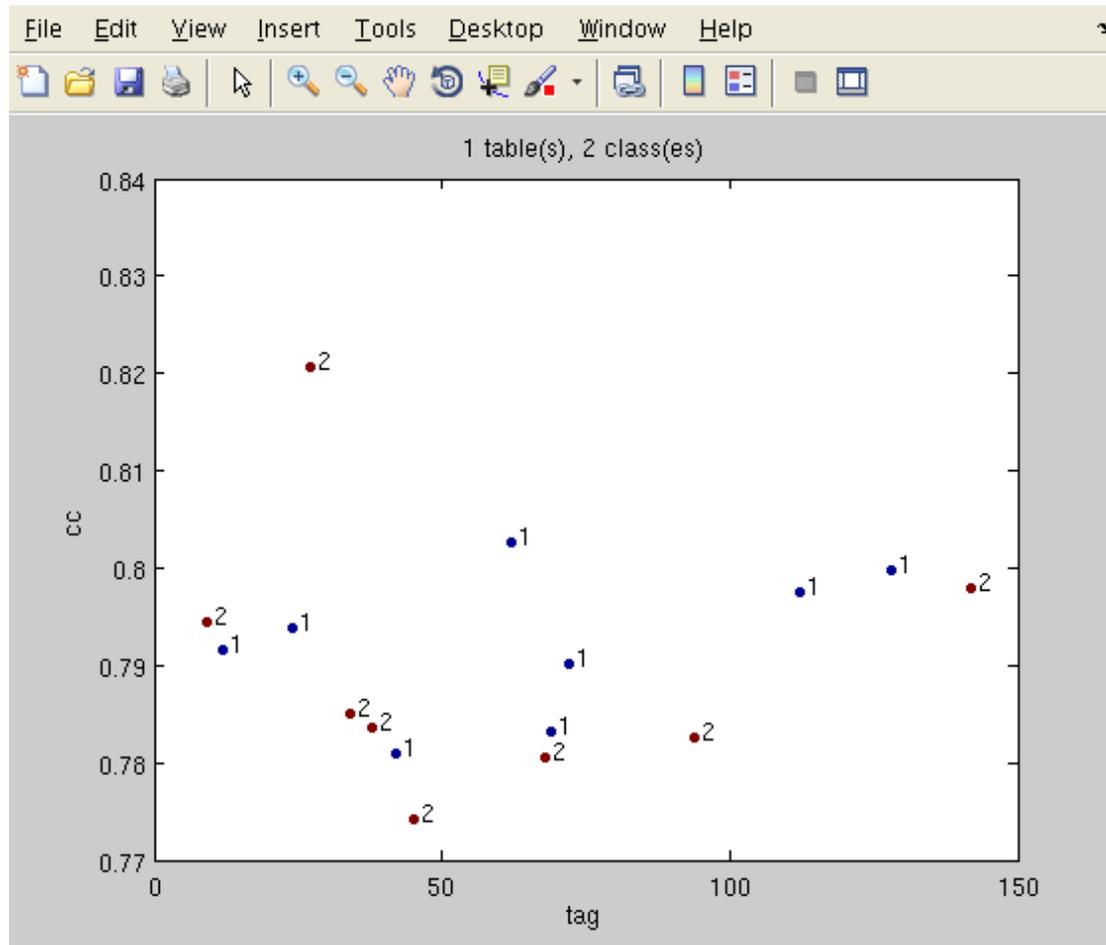
scatterplot modus

The scatter plot shows that particles actually belonging to class 2 (per construction) actually yield systematically a better score when compared to the average produced in channel 2 that particles from class 2

you can move the main table to value “3” and see that class 1 behave analogously, at least for the second iteration:



... but note that this separation is not a given! Look at the first iteration in reference 1:



class 1 and 2 have the same quantitative response to the template in channel one at this stage. As we have seen in the previous slide, in the second iteration the template on this channel has improved and the cross correlation could actually disentangle the two populations