Orange-NMF add-on

Fajwel Fogel (Ecole Polytechnique ParisTech)
Marinka Zitnik (Bioinformatics Laboratory, FRI UL)

Outline

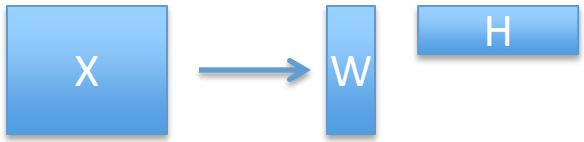
- What is NMF?
- What is Orange?
- NMF widgets in Orange (demonstration):
  - Overview of the workflow and the widgets
  - Example: a synthetic dataset

What is NMF?

What is NMF? Goal

- Perform non-negative matrix factorization with a given rank k on $X \geq 0$

$$X = W \cdot H$$
$$W, H \geq 0$$



- Where
  - X is the initial (non negative) matrix, columns are attributes and rows are observations , or the contrary (size $n \times p$)
  - W is the matrix of basis vectors (weights) (size $n \times k$)
  - H is the matrix of mixture coefficients (size $k \times p$)
- The factorization is generally non-unique
- NMF differs from other matrix factorization methods by incorporating a constraint of non-negativity on the factors W and H, i.e., all elements must be equal or greater than zero
- If $k \ll n, p$ we have summarized the information by discovering "hidden features", since for each i we now have

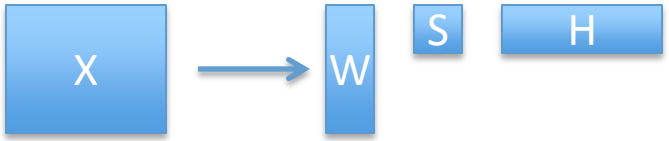
$x_i = \sum_{j=1}^{p} H_{ji} w_i$ (where x_i and w_i are the column vectors of X and W)

What is NMF? Goal

- In order to get an interpretable factorization, we add a scaling (diagonal) matrix S

  For a given rank k we now have:

$$X = W \cdot S \cdot H$$



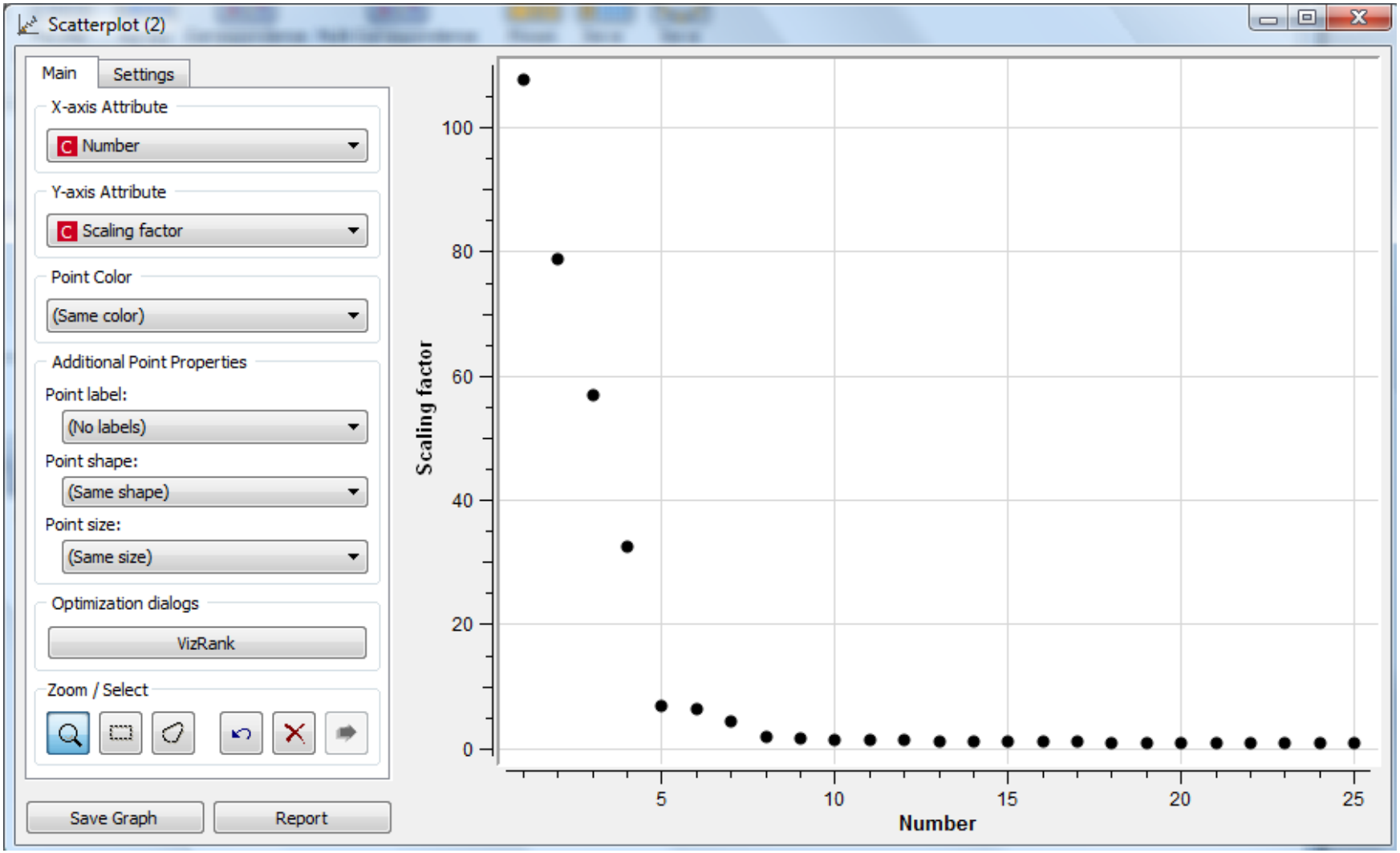
  And $x_i = \sum_{j=1}^{p} H_{ji} S_{ii} w_i$

- Where
  - X is the initial (non negative) matrix, columns are attributes and rows are observations (size $n \times p$)
  - W is the matrix of basis vectors (weights) (size $n \times k$)
  - S is the scaling (diagonal) matrix (size $k \times k$)
  - H is the matrix of mixture coefficients (size $k \times p$)
  - x_i and w_i are column vectors

What is the rank k?
How to choose it?

- The rank k corresponds to the number of underlying “hidden features”
- Choosing k is a complex task
  - Several measures can be computed in order to assess what is the best k (cophenetic correlation coefficient etc.)
  - It seems that k can be best determined by looking at the scree plot of the SVD/PCA decomposition
  - Both ways can be lengthy, depending on the size of the data (minutes to hours)

A typical scree plot



What is NMF?
The math:

$$minimize \|X - W \cdot H\|$$
$$subject\ to\ W, H \geq 0$$

- $\|\cdot\|$ is a divergence function
- Two simple divergence functions studied by Lee and Seung (1999) are
  - the squared error (or Frobenius norm)
  - an extension of the Kullback-Leibler divergence to positive matrices

What is NMF?
In practice:

- There is no closed form to resolve this optimization problem

  -> perform numerical optimization

- Several optimization methods were developed for NMF, including “sparse” methods
- There are available codes in Python, Matlab, JMP, R etc.

Other matrix factorization methods

- Principle component analysis (PCA)
- Singular value decomposition (SVD)
- Independent component analysis (ICA)

- While these other methods are very well diffused, NMF is still less used in many fields, although it has a lot of advantages, mainly for interpretation

Why choose NMF rather than PCA/SVD

- PCA/SVD fails for interpretation:
  - 1st component is a general sum
  - 2nd component is a "contrast"

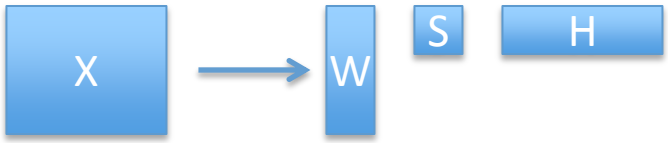
  Interpretation of loadings is typically difficult
- Contention: thanks to the non-negativity constraint NMF finds "parts"
  - NMF commits one vector pair to each part
  - Or finds the generating vector pairs

Summary of NMF

- Perform non-negative matrix factorization with a given rank k on $X \geq 0$

$$X = W \cdot S \cdot H$$
$$W, H \geq 0$$

$$x_i = \sum_{j=1}^{p} H_{ji} S_{ii} w_i$$



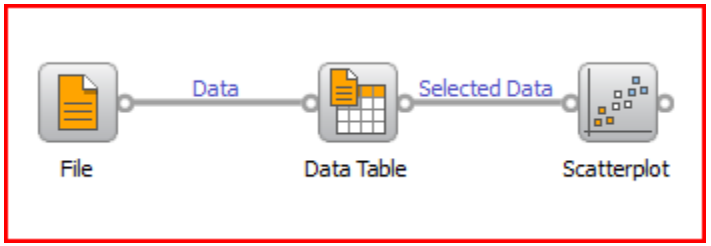
What is Orange?

Orange

- Orange is developed at Bioinformatics Laboratory, Faculty of Computer and Information Science, University of Ljubljana, Slovenia
- It is an open source project
  -> Download is free at
  http://orange.biolab.si/download/
- Orange provides:
  - data visualization and analysis
  - data mining through visual programming or Python scripting
  - components for machine learning
  - add-ons for bioinformatics and text mining
  - features for data analytics

A visual programming environment for data mining/machine learning

- No need to code!
- Connect widgets together to represent your workflow



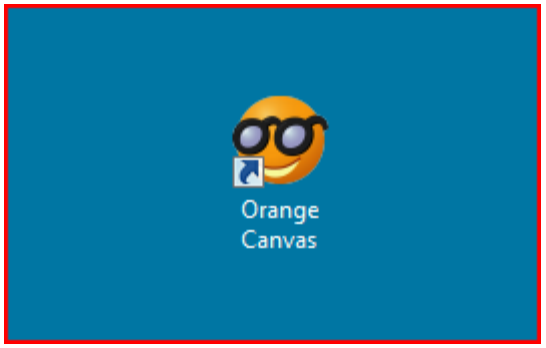
- For advanced users, use Python scripting language as an additional tool

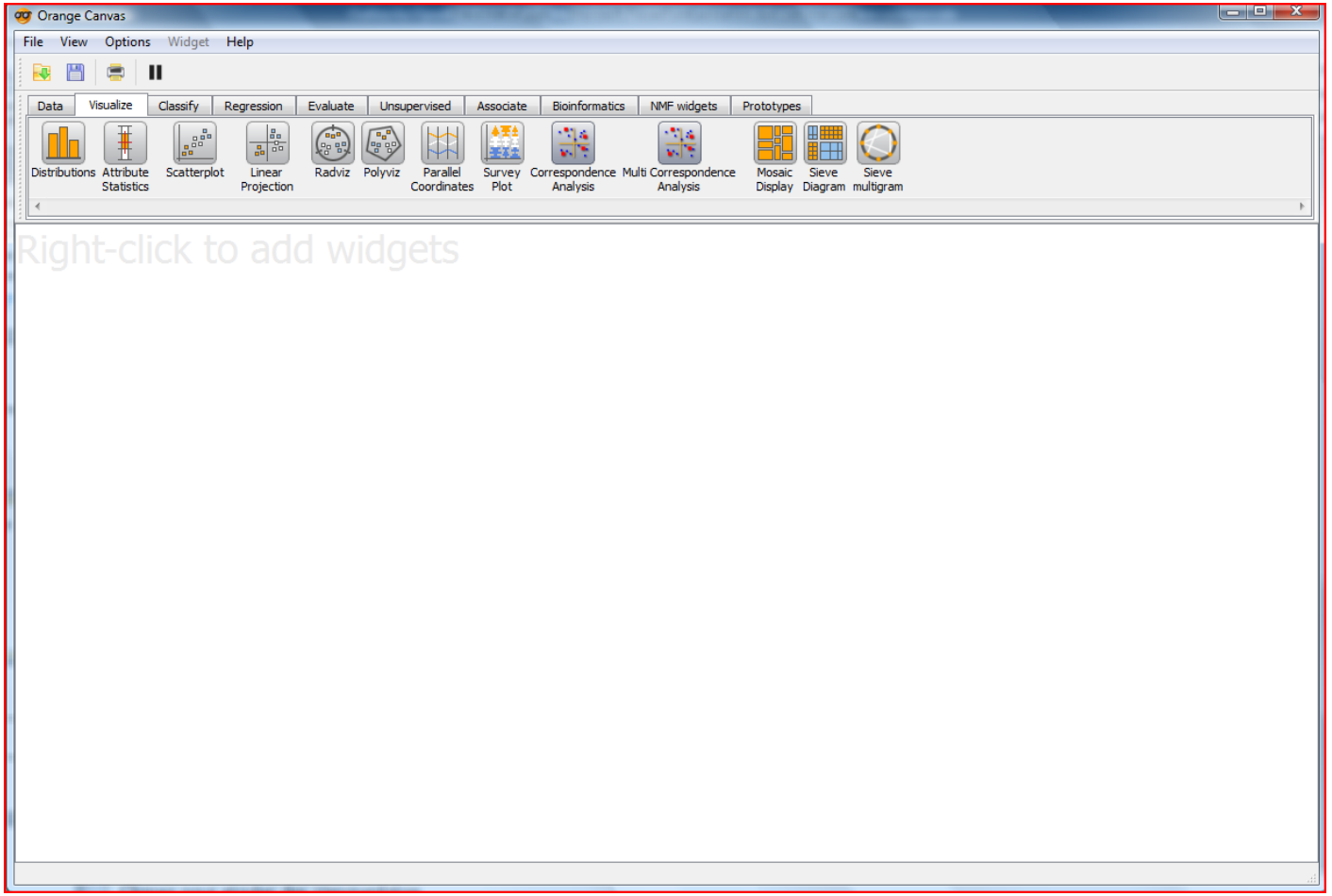
Use of Python and its libraries

- Python is a widely used programming language and is freely available
- Many scientific libraries are available in Python
  - the most popular (and used by Orange) are Numpy and Scipy (enable to use matrix calculus)
- While Orange core is implemented in C++, all the widgets are implemented in Python
- For NMF, we have used the NIMFA library provided by Marinka Zitnik (http://nimfa.biolab.si/)

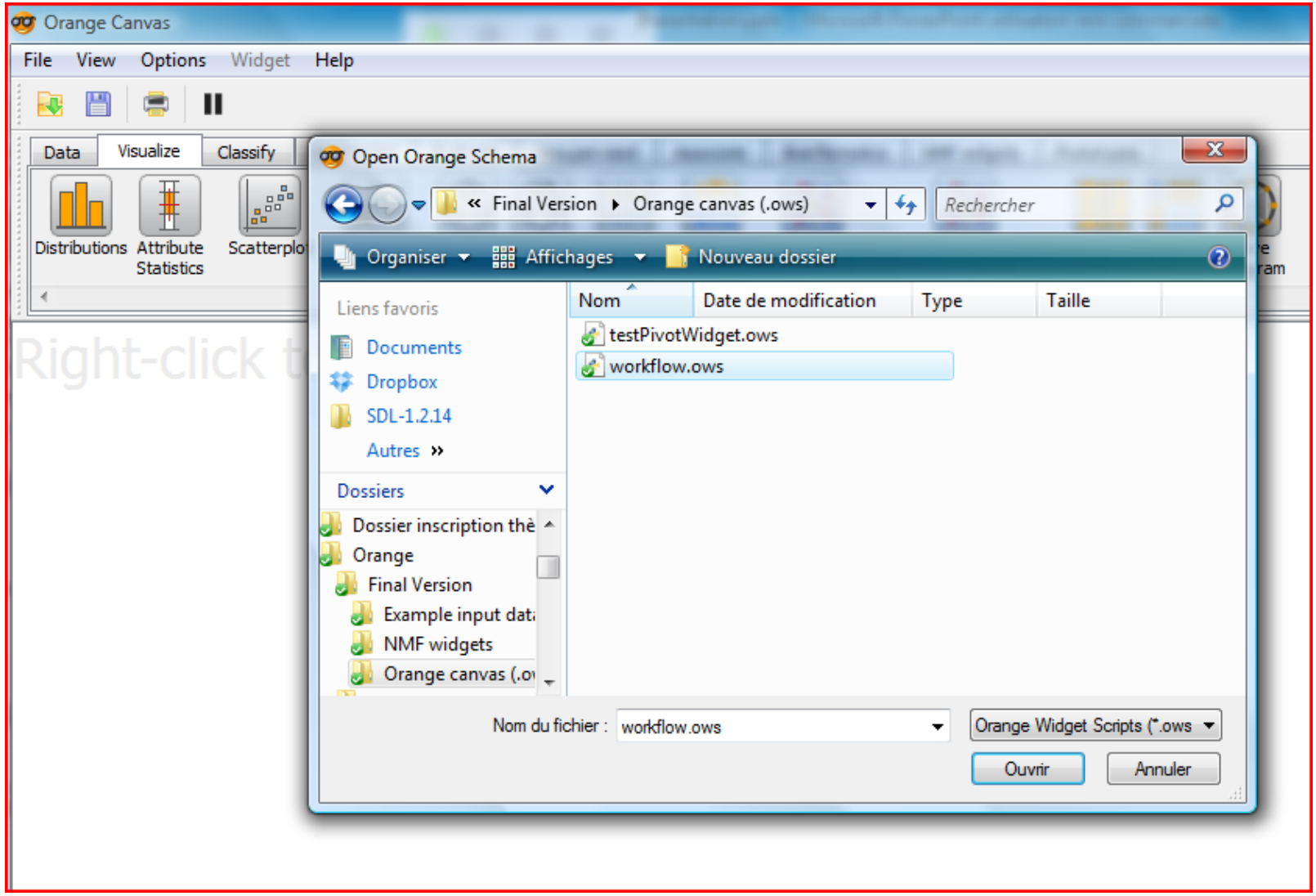
How to use the NMF widgets?

Open Orange

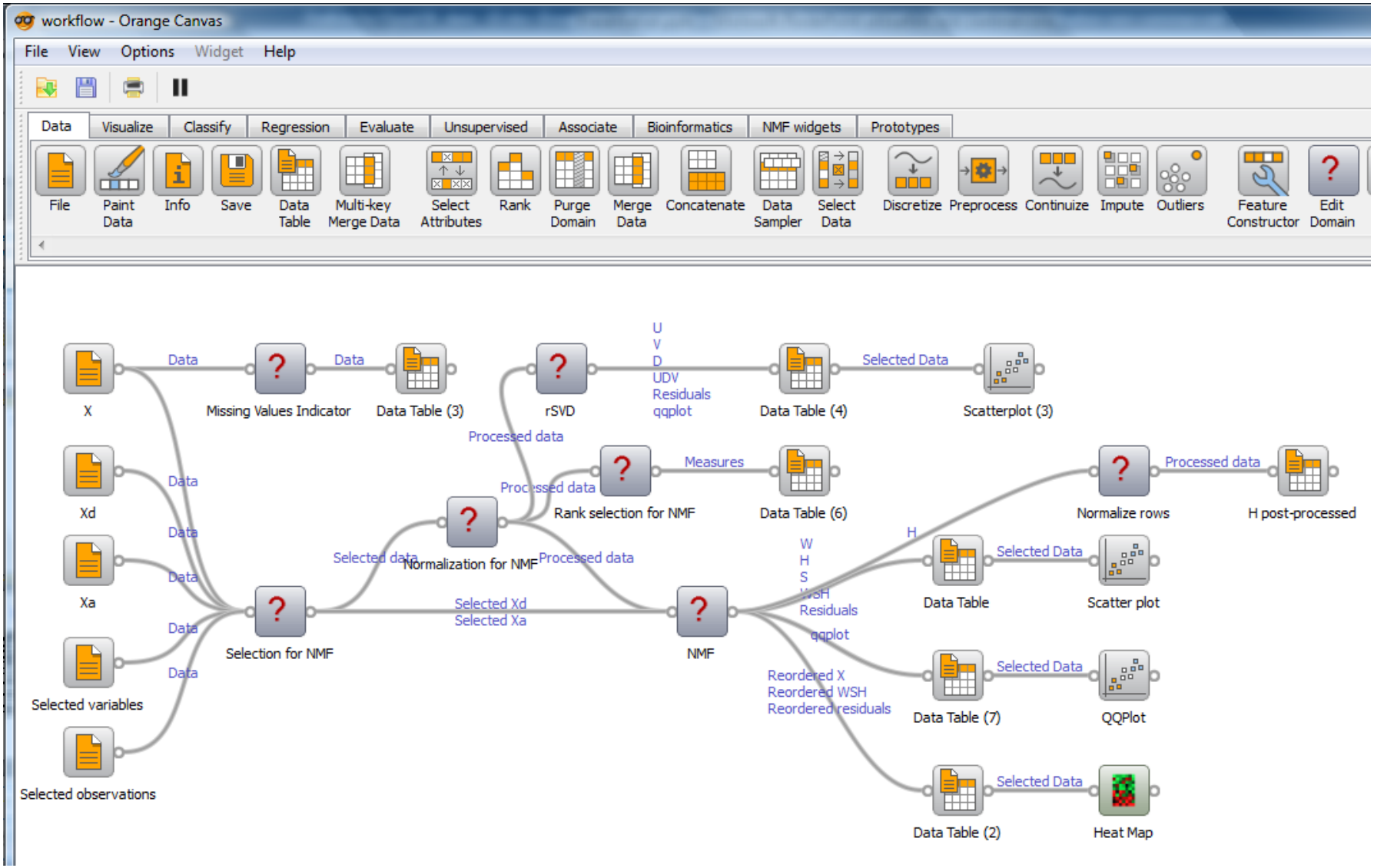

Orange
Canvas


Orange Canvas
File View Options Widget Help
Data Visualize Classify Regression Evaluate Unsupervised Associate Bioinformatics NMF widgets Prototypes
Distributions
Attribute Statistics
Scatterplot
Linear Projection
Radviz
Polyviz
Parallel Coordinates
Survey Plot
Correspondence Analysis
Multi Correspondence Analysis
Mosaic Display
Sieve Diagram
Sieve multigram
Right-click to add widgets

Open the NMF workflow



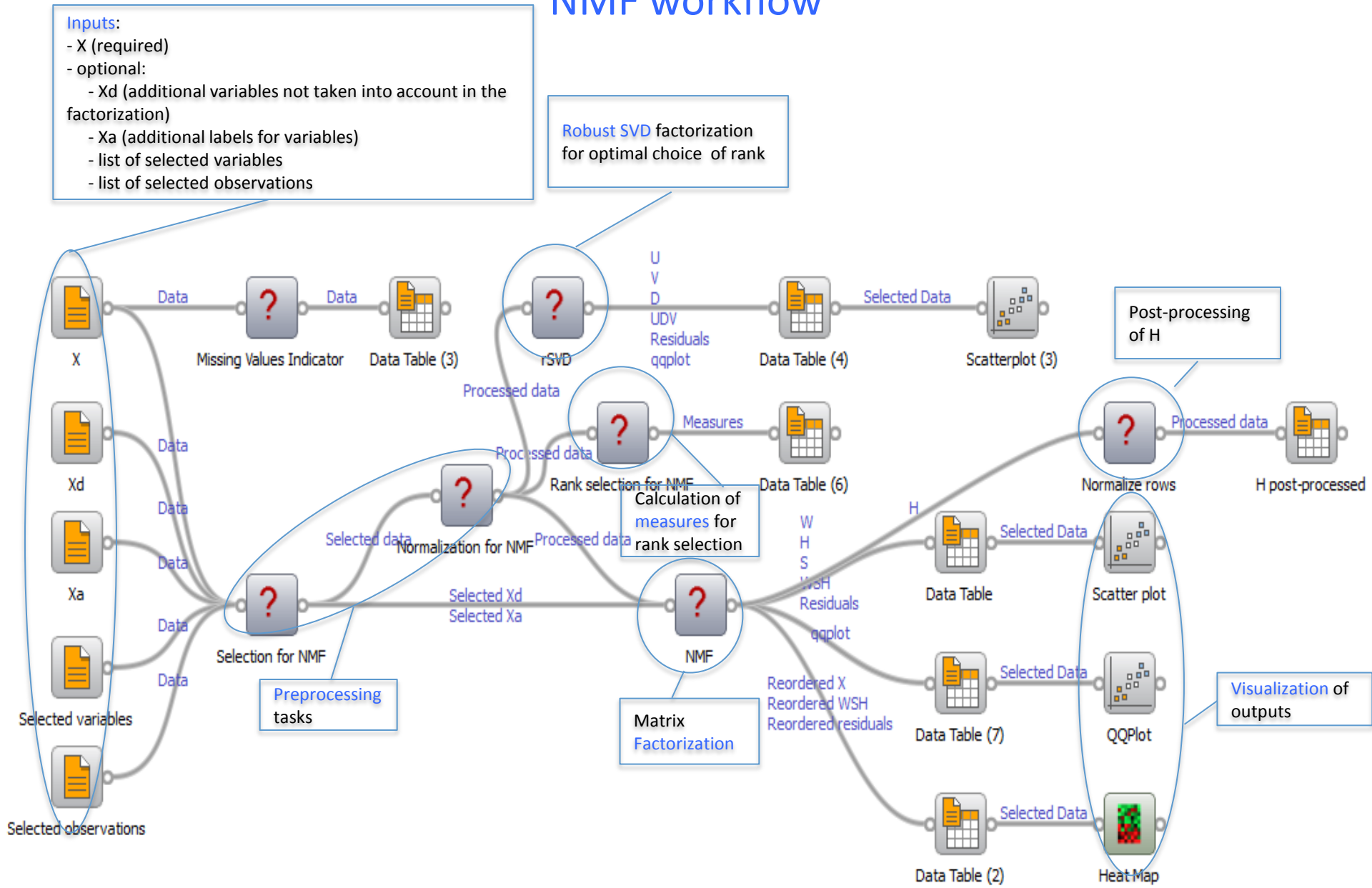
NMF workflow



Workflow

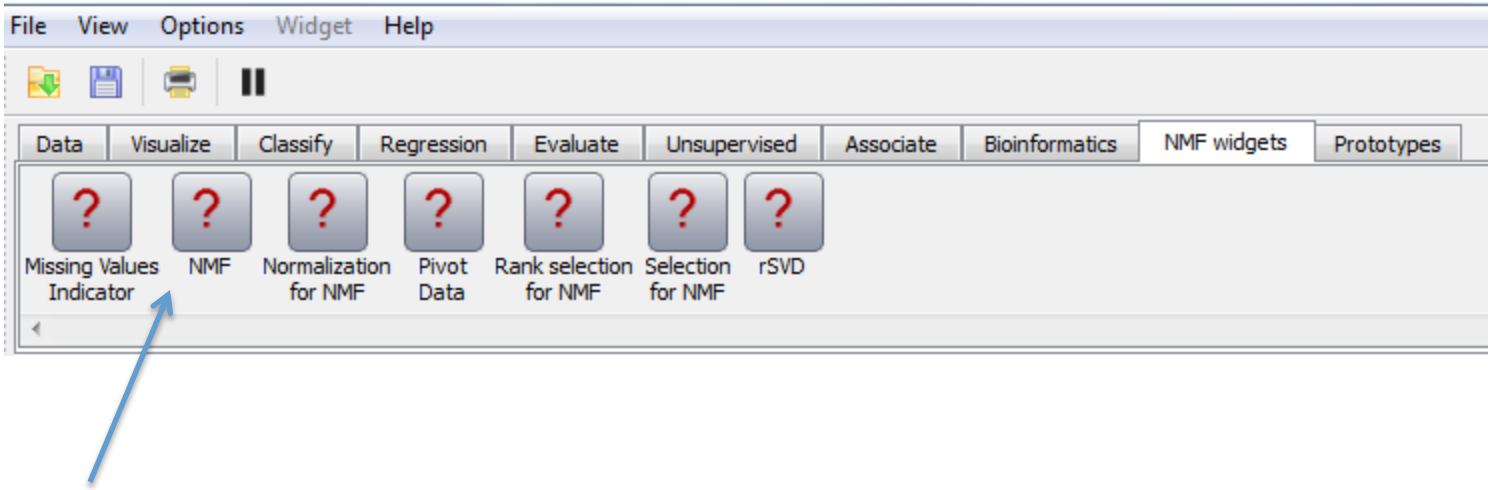
- Possible pre-processing options:
  - select attributes and observations
  - take the log of the data
  - normalize the data (scale, divide by median, subtract minimum of each variable)
- Choose best rank according to:
  - value of different measures:
    - cophenetic correlation coefficient
    - residual sum of squares (RSS)
    - sparseness of output matrices
  - Scree plot of SVD decomposition
- Perform NFM
  - Choose factorization algorithm (NMF, SNMF, LSNMF, BMF) and initial matrices (random, fixed)
- Visualize factorization
  - Score plots of components
  - Heat maps of reordered X, fitted matrix, residuals
  - QQ plot of residuals

NMF workflow



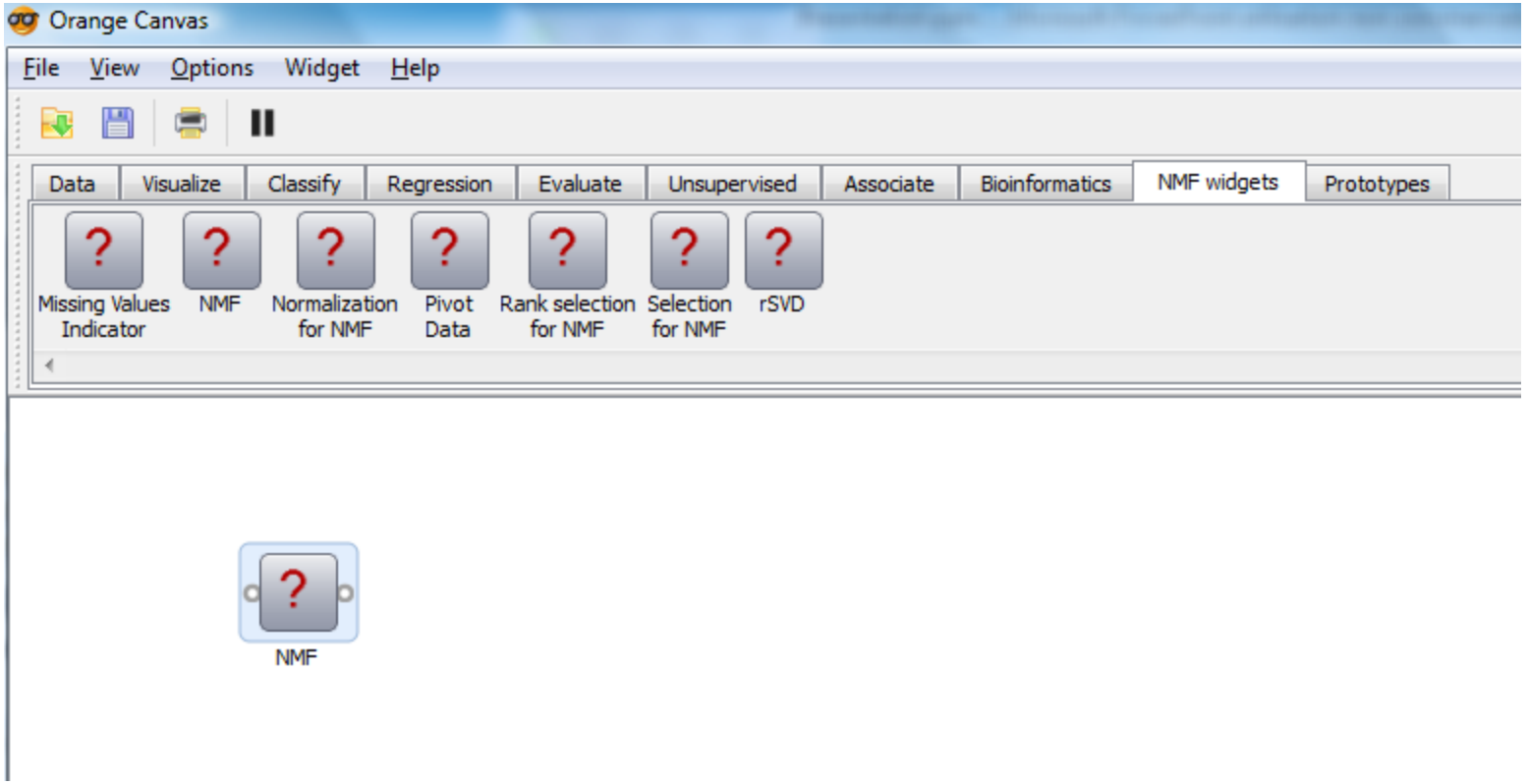
How to build the workflow (1)

Click on the widgets in the tabs



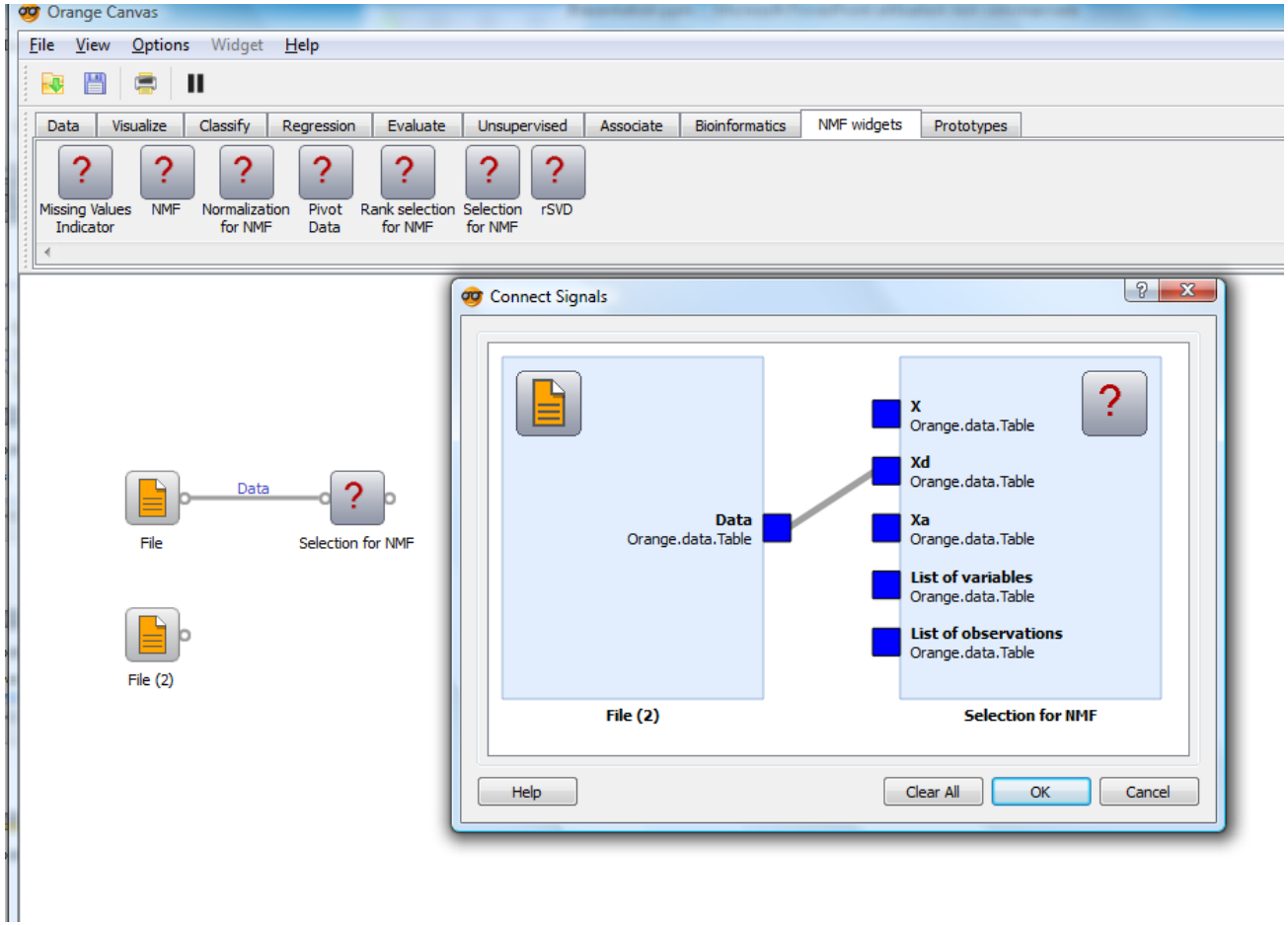
How to build the workflow (2)

Drag and drop them on the canvas



How to build the workflow (3)

Link them together



Format of input files

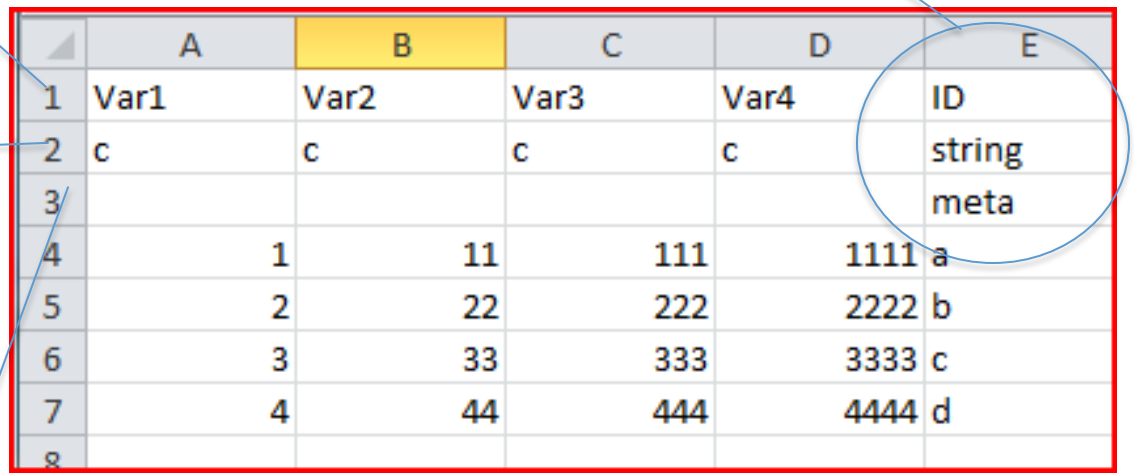
Input files must have a *.tab extension and follow Orange format. You can easily format your files in Excel (save them in *.txt tab-delimited format and then manually change the extension to *.tab)

There must be one column of identifiers named “ID” and declared as “string”, “meta”

The first row contains the names of the variables

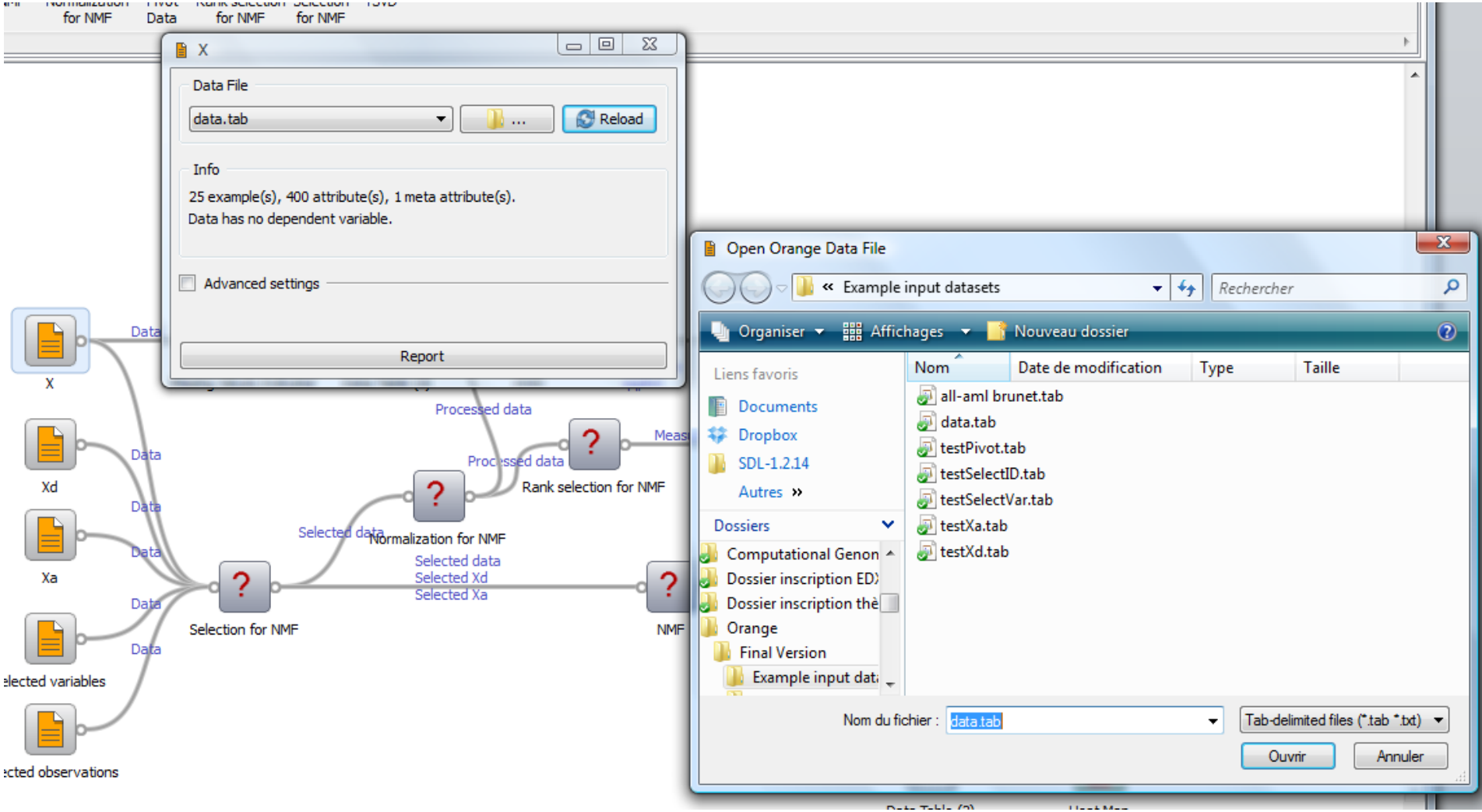
The second row contains the types of the variables:
- “continuous” (or “c”)
- “discrete” (or “d”)
- “string”

- The third row declares the “meta” attributes and “class” variables (no class variables for NMF widgets inputs)
- Variables in Xd/Xa should all be declared as “string” and “meta”
- The list of variables/ observations to select (Selection widget) should just be declared as string (not meta)

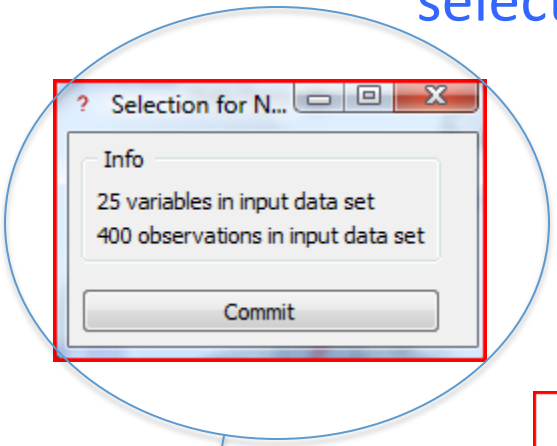


	A	B	C	D	E
1	Var1	Var2	Var3	Var4	ID
2	c	c	c	c	string
3					meta
4	1	11	111	1111	a
5	2	22	222	2222	b
6	3	33	333	3333	c
7	4	44	444	4444	d
8					

Widget file: connect to external data

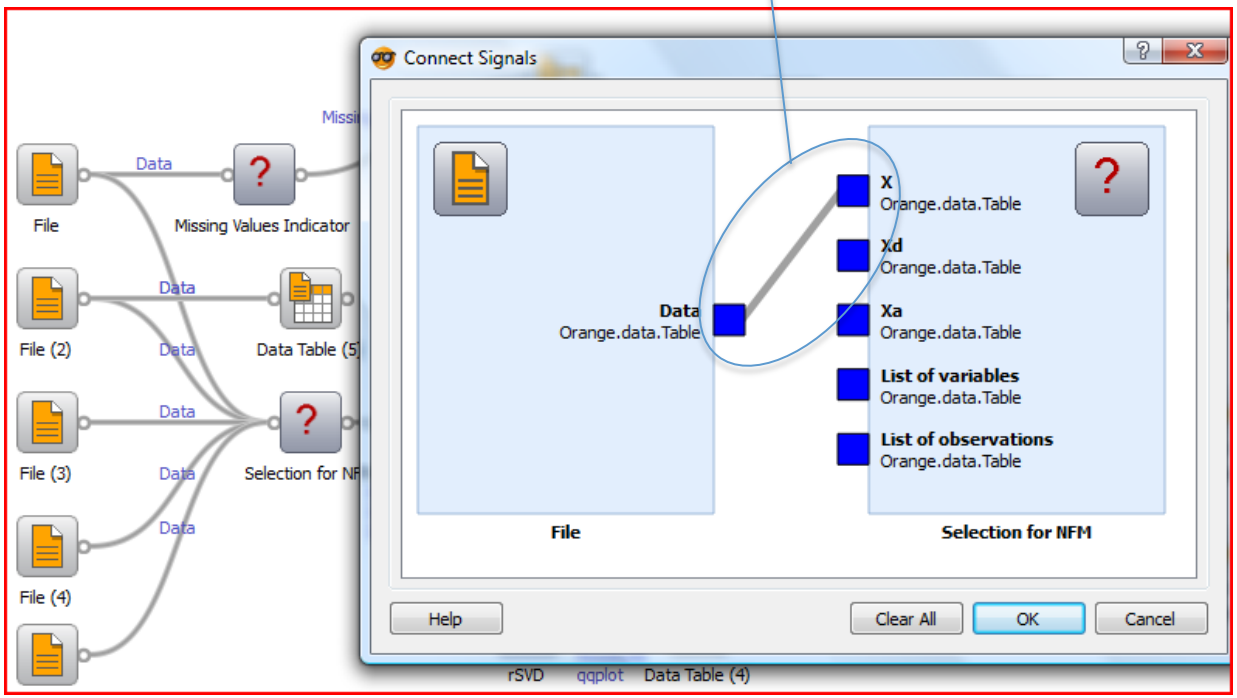


Widget “Selection for NMF”: select variables and observations in X

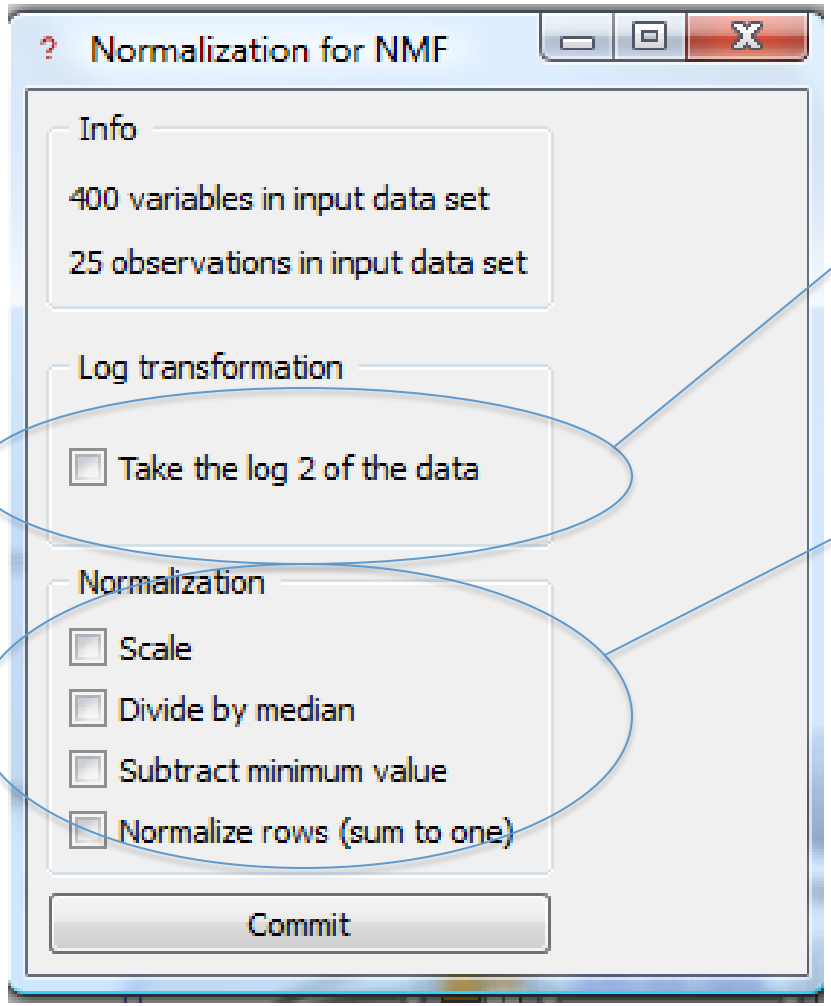


- Click on “Commit” to select variables and observations
- If you have no “List of variables” and/or “List of observations” inputs, then all variables and/or observations are selected by default
- Selection is automatically updated each time you change an input file or reopen your canvas (it means you do not have to click on “commit” again if you change your input files or reopen your canvas)

Connect the widgets “file” with the corresponding inputs



Widget “Normalization for NMF”: pre-process your data before the factorization

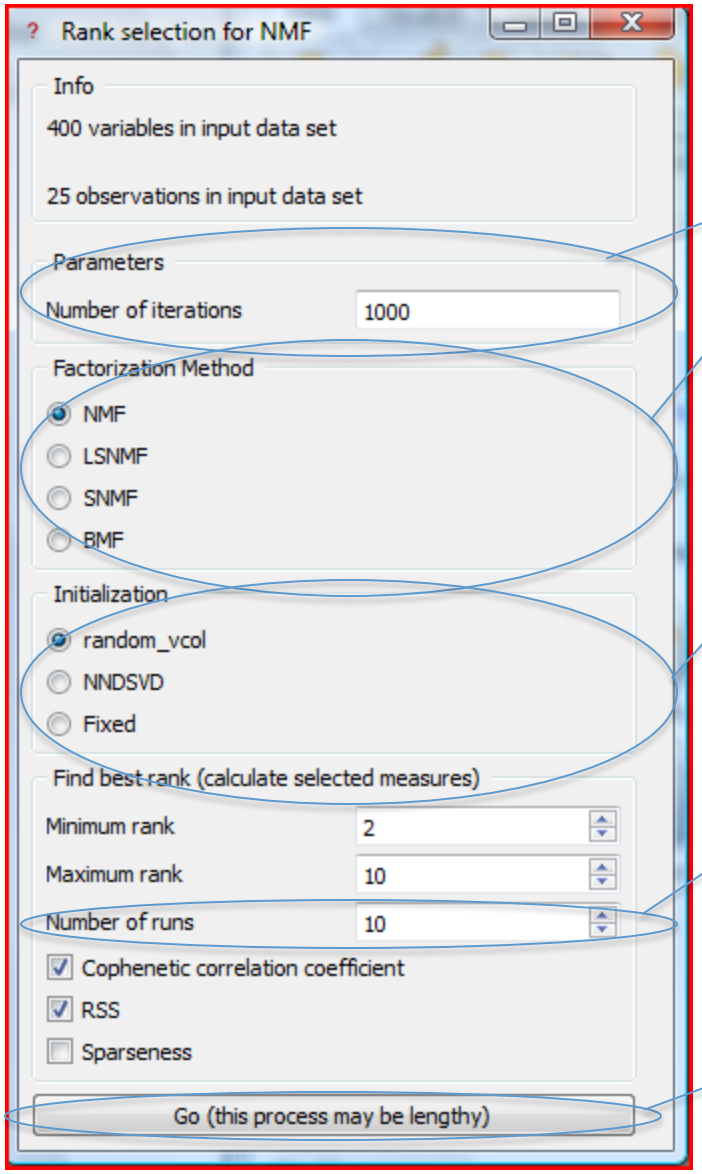


Take the logarithm (base 2) of your data

Normalize your data:
- “Scale”: Divide each variable by its L1 norm (the sum of the absolute value of its observations)
- “Divide by median”: divide each variable by its median
- “Subtract minimum value: subtract to each variable its minimum
  -> get positive data
- “Normalize rows”: sum to one each row

Selection is automatically updated each time you change an input file or reopen your canvas (it means you do not have to click on “commit” again if you change your input files or reopen your canvas)

Widget “Rank Selection for NMF”: get measures on a set of different ranks to find the best one



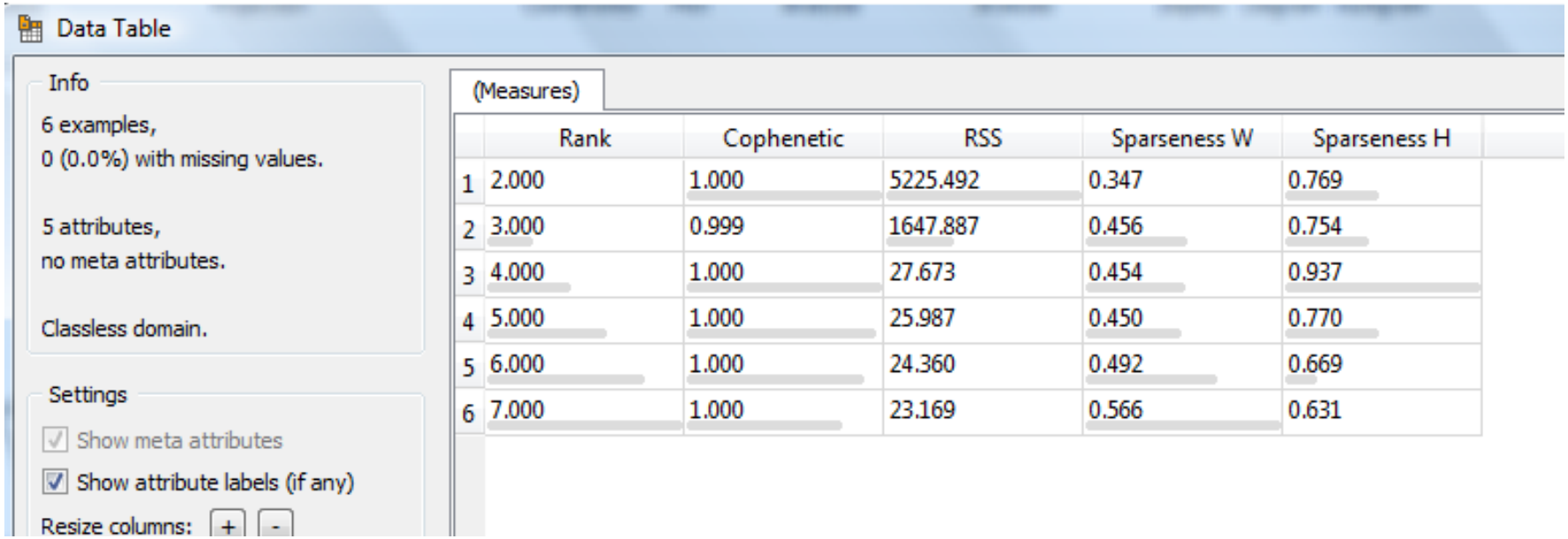
Choose your factorization method.
The number of iterations are updated to their default respective value each time you select a factorization method.

Choose your initialization method for W and H:
- “random_vcol” option is a type of random initialization
- “NNDSVD” option performs SVD to find initial factors
- “Fixed”option requires that you give your own initial factors W and H as inputs

Set the number of runs carefully. By default it is 10 if you calculate the “cophenetic correlation coefficient” and 1 otherwise.
Calculating the “cophenetic correlation coefficient” takes a lot of time since you need to perform “number of runs” factorizations.

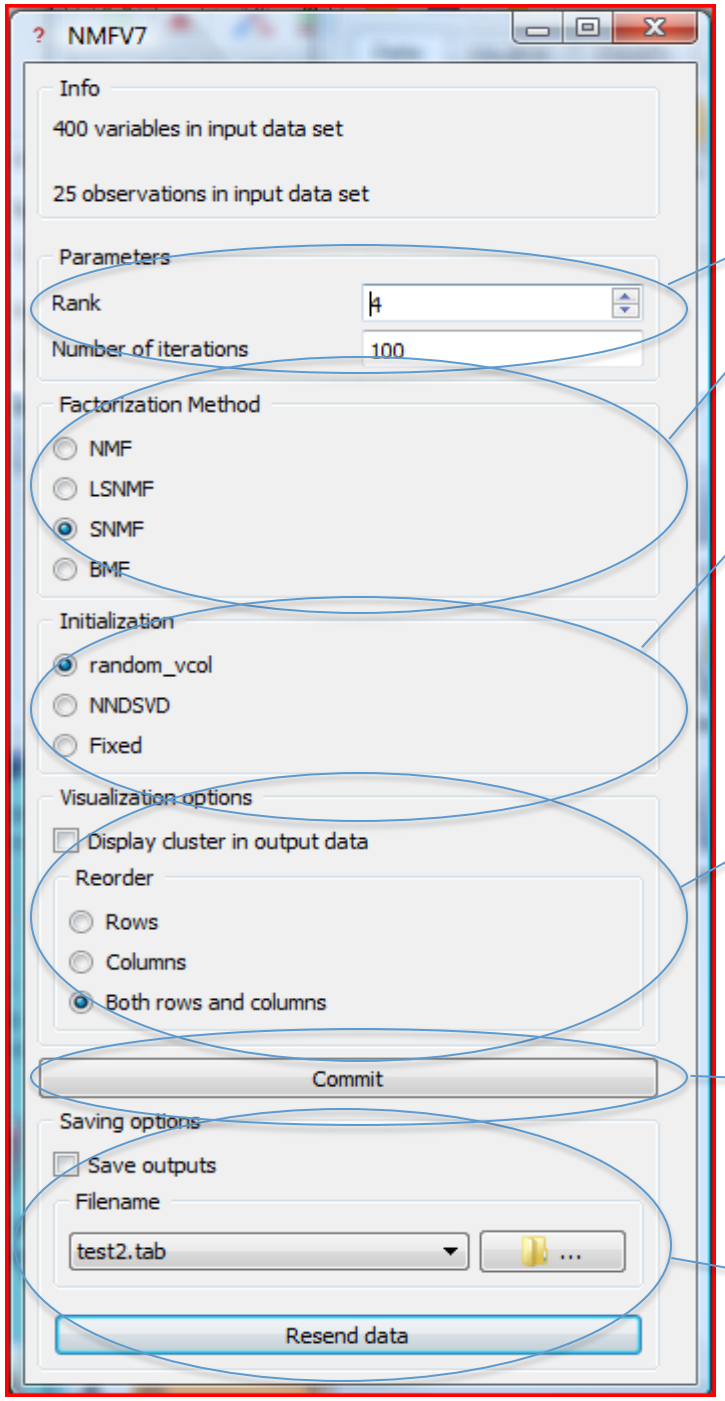
Calculate measures and send them to a “Data Table” widget for visualization

Several measures to choose the best k



	Rank	Cophenetic	RSS	Sparseness W	Sparseness H
1	2.000	1.000	5225.492	0.347	0.769
2	3.000	0.999	1647.887	0.456	0.754
3	4.000	1.000	27.673	0.454	0.937
4	5.000	1.000	25.987	0.450	0.770
5	6.000	1.000	24.360	0.492	0.669
6	7.000	1.000	23.169	0.566	0.631

Widget NMF: perform NMF



Choose your factorization method.
The number of iterations are updated to their default respective value each time you select a factorization method.

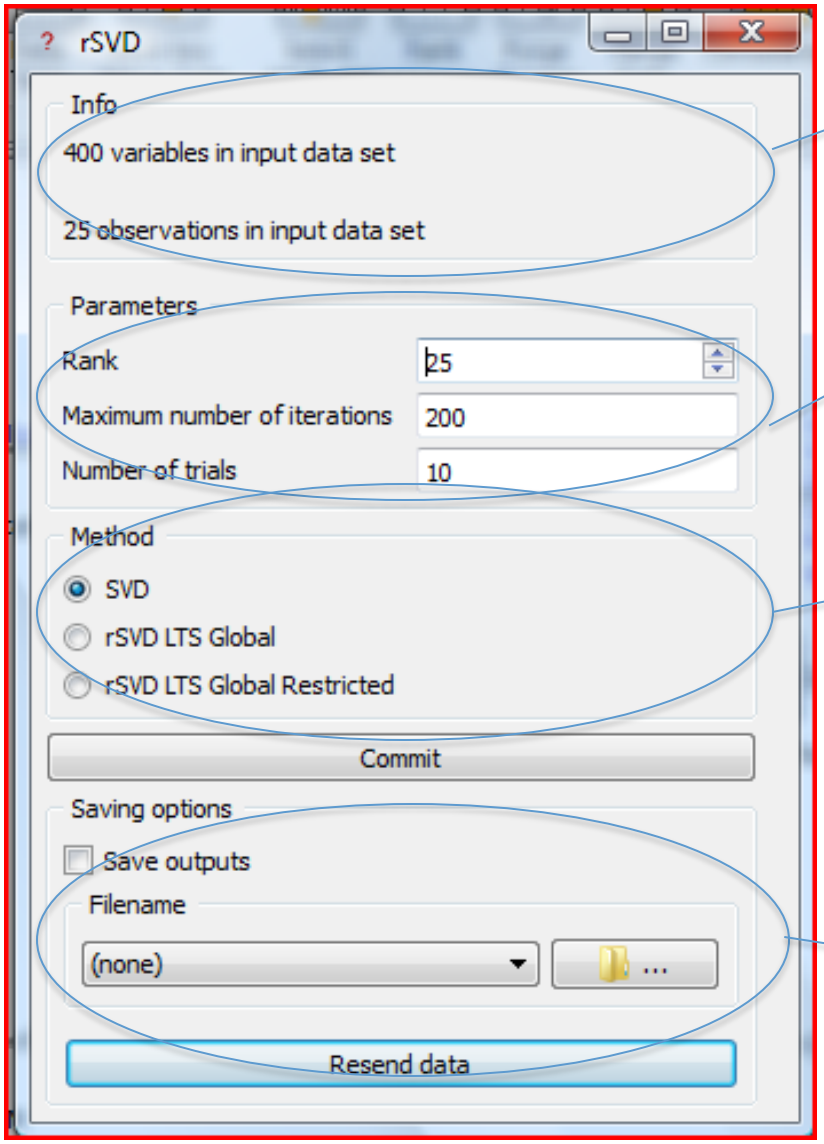
Choose your initialization method for W and H:
- "random_vcol" option is a type of random initialization
- "NNDSVD" option performs SVD to find initial factors
- "Fixed"option requires that you give your own initial factors W and H as inputs

Visualization options:
- The "Display cluster in output data" enables to have a class column in your outputs representing the assigned cluster for each observation (based on the highest weight). The only inconvenient of this option can be that it produces as many heat maps as there are clusters in the" heat map" widget (confer to "Display heat maps").
- Additionally to W, S, H, WSH (fitted matrix) and residuals, X, WSH and residuals are outputted reordered by row and/or column, according respectively to their highest weight (W) and/or mixture coefficient (H).

Perform NMF and send outputs to "Data table" widgets for visualization. The process may be lengthy if your data is big!

Save your outputs: confer to "saving your work"

Widget SVD decomposition



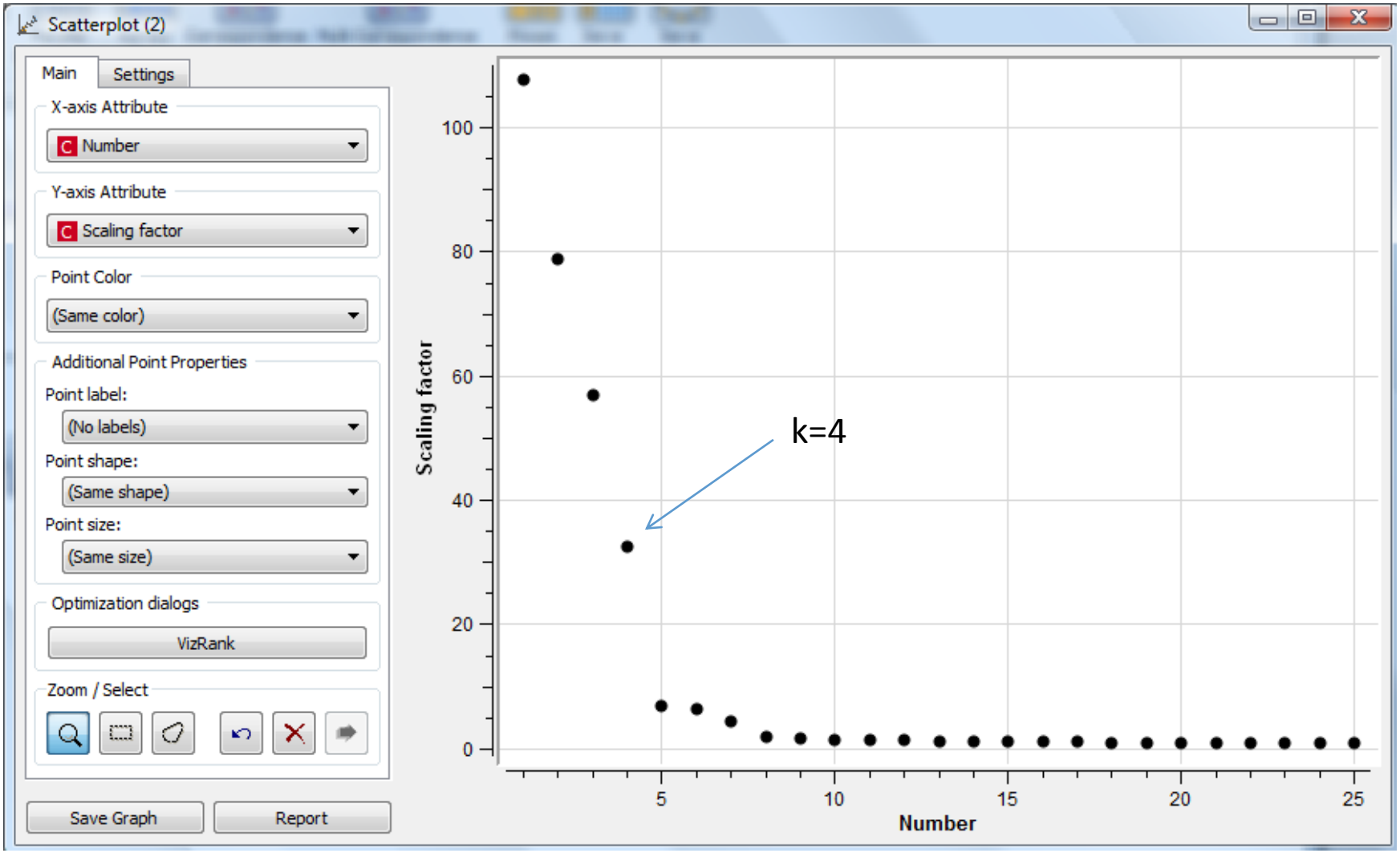
- Contrarily to NMF, inputs are allowed to have missing values.

- Default rank is the minimum of *n* and *p* where (*n*, *p*) are the dimensions of X.
- The maximum number of iterations should normally remain 200.
- The number of trials is only taken into account for rSVD. It can be increased to get a more accurate output or decreased to perform a faster SVD.

- "SVD" is the standard singular value decomposition
- "rSVD LTS Global" and "rSVD LTS Global Restricted" are robust versions of SVD. Note that rSVD takes more time to compute than standard SVD, since many "trials" are performed.

Save your outputs: confer to "saving your work"

Choose the best k by looking at the scree plot of SVD

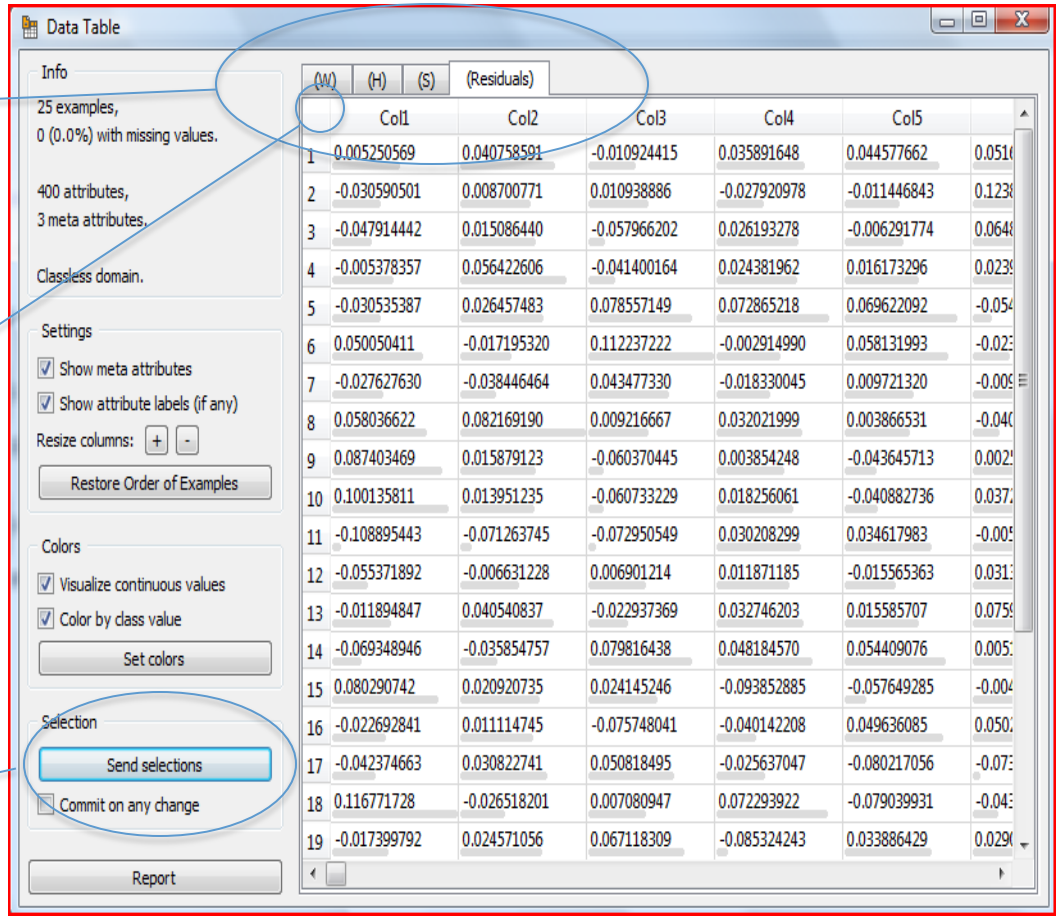


Widget “Data Table”: Display Data Table(s)

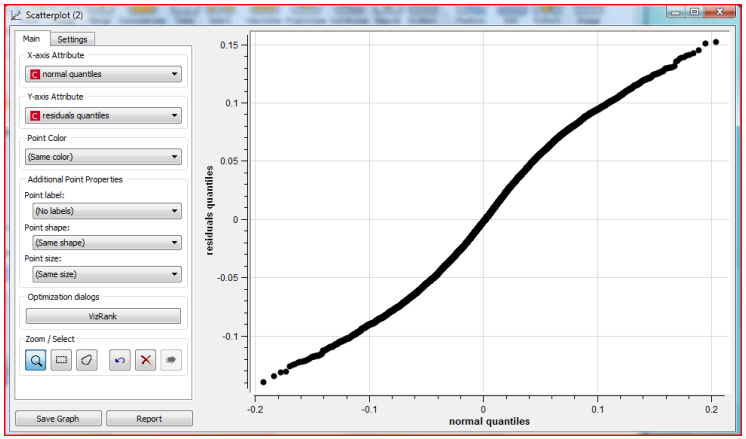
- You can display several tables in one “Data Table” widget

- Click there to select the whole data table

- Click on “Send selections” to send the selected rows to a visualization widget (e.g. “heatmap” widget, “scatter plot” widget etc.)

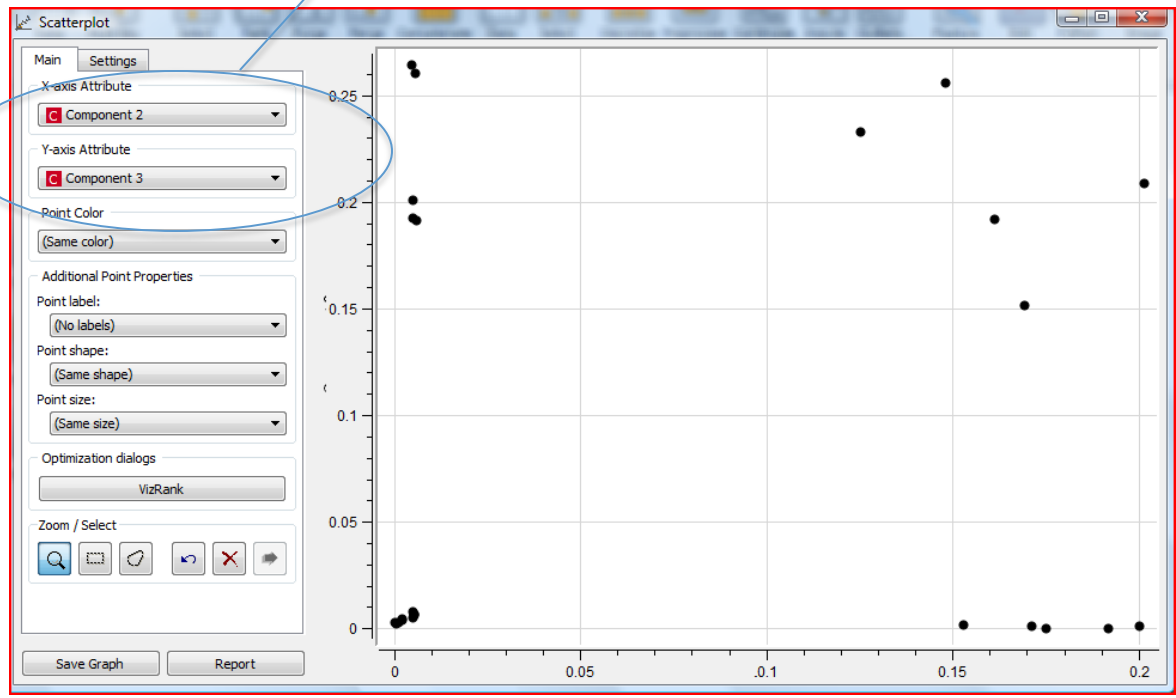


Widget “Scatter plot”: Display Score Plots and QQPlot of Residuals



qqplot of residuals

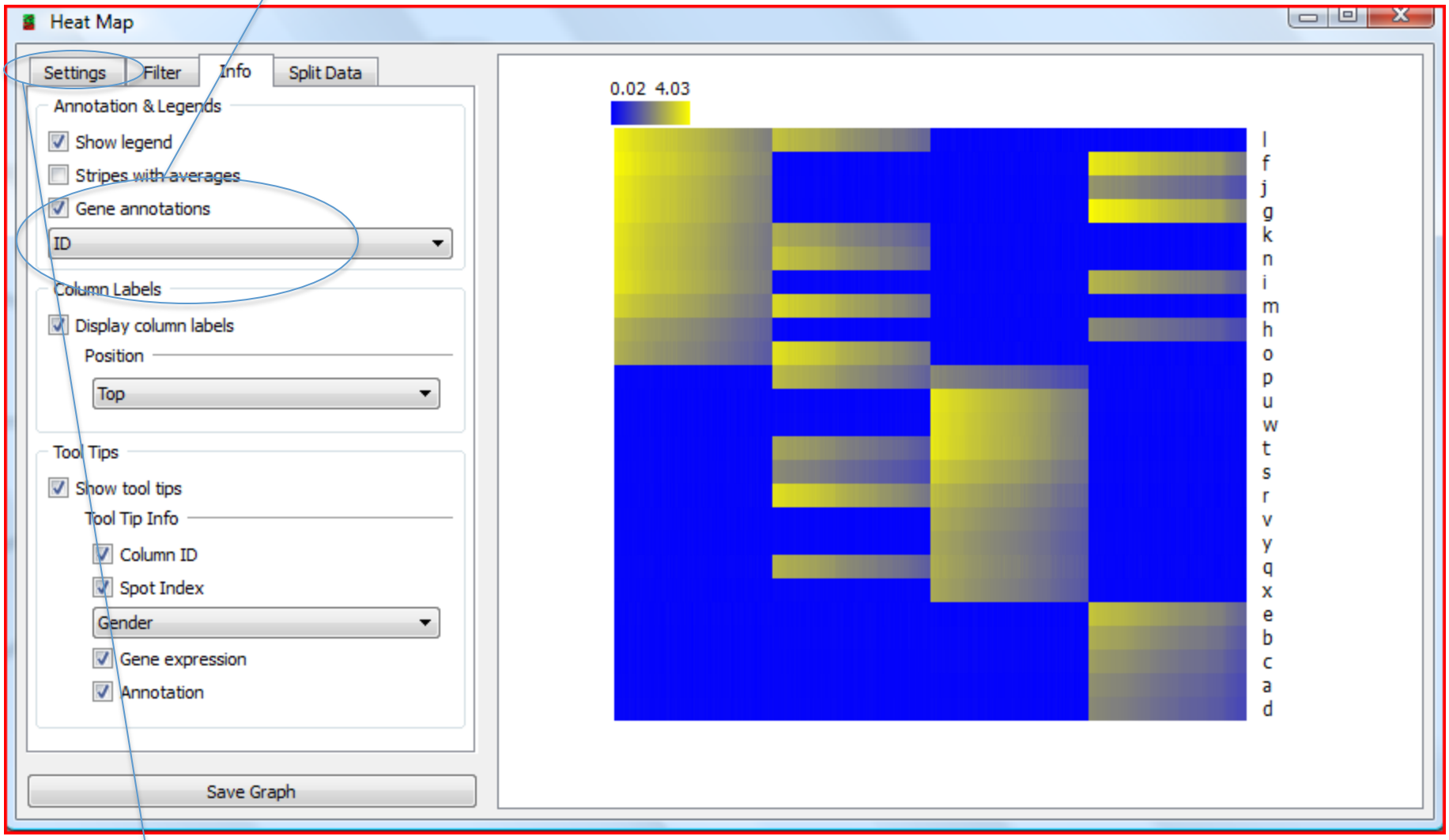
- Select the components you want to plot (here the input data table is W)



Widget “Heat Map” Display Heat Maps:

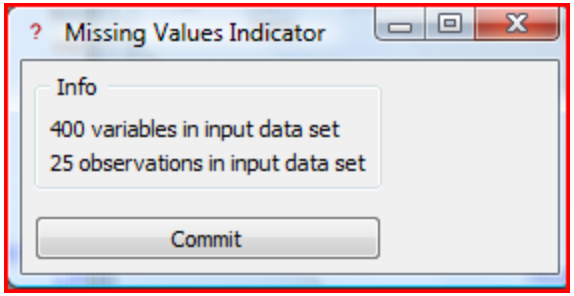
http://www.biolab.si/supp/bi-visprog/Orange%20Genomics.pdf for more information

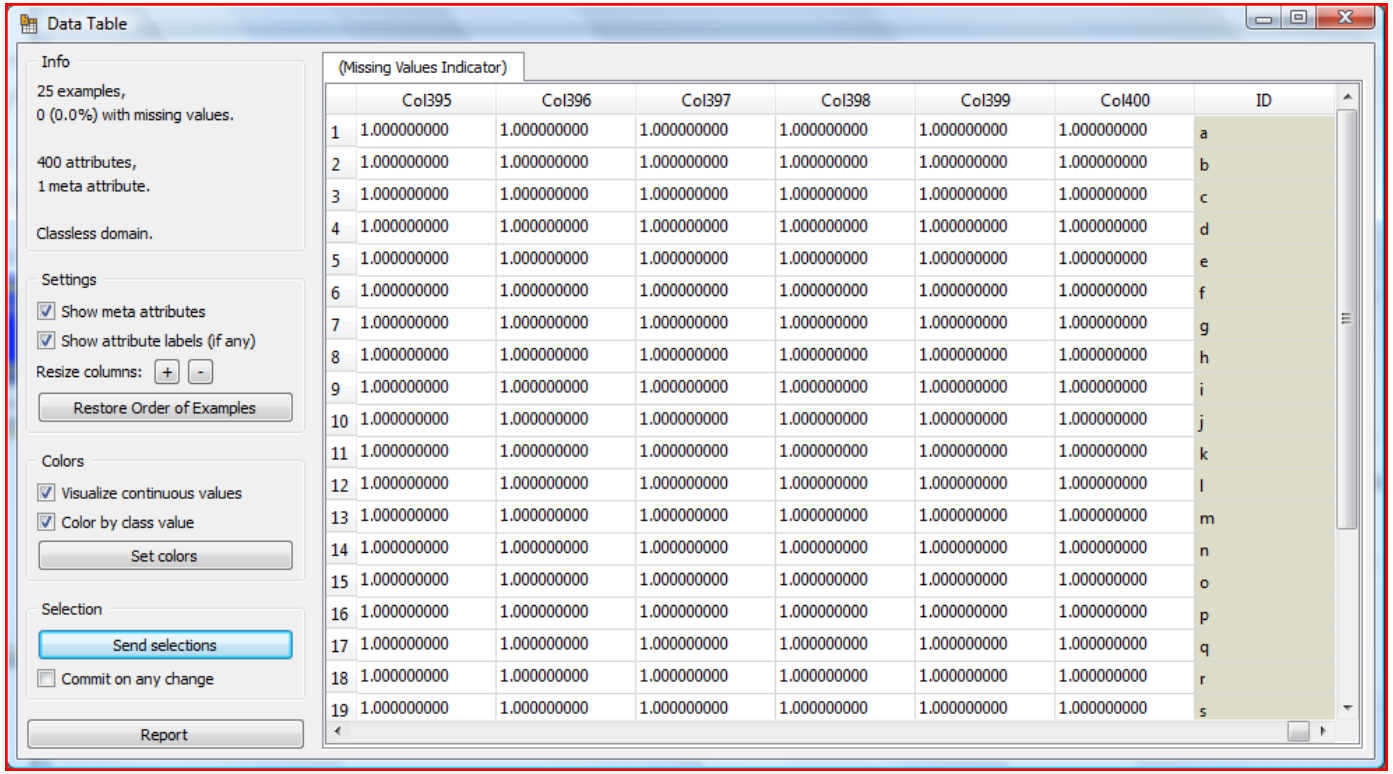
Select the additional variables from Xd you want to display on the heatmap



Change the size of the heatmap in the “Settings” tab

Widget “Missing Values Indicator”: return an indicator table of missing values (0 for missing value, 1 for available)

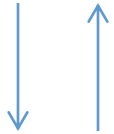




Widget Pivot

testPivot (Data)

	C1	C2	C3	C4	ID
1	1	2	3	4	R1
2	5	6	7	8	R2
3	9	10	11	12	R3

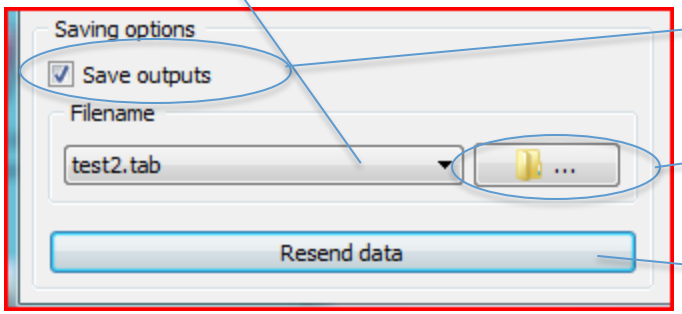


(Pivoted Data)

	Value	ID	Names
1	1.000	R1	C1
2	5.000	R2	C1
3	9.000	R3	C1
4	2.000	R1	C2
5	6.000	R2	C2
6	10.000	R3	C2
7	3.000	R1	C3
8	7.000	R2	C3
9	11.000	R3	C3
10	4.000	R1	C4
11	8.000	R2	C4
12	12.000	R3	C4

Saving your work

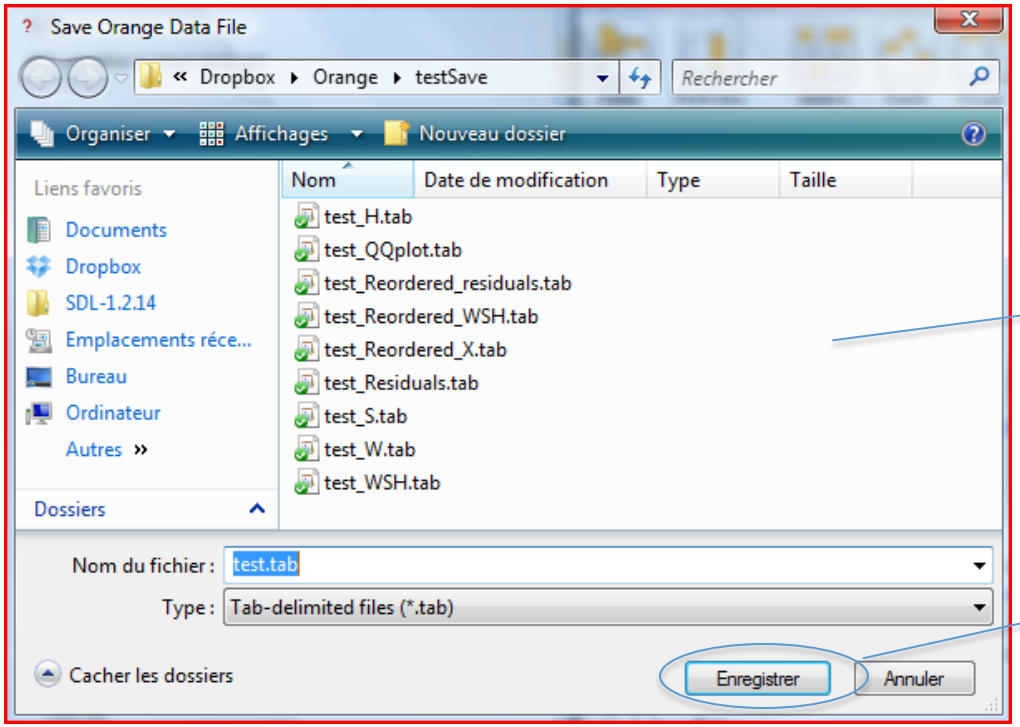
List of recently saved files



Select this option to automatically save your outputs in the directory of the “filename”

Browse button to choose the directory where you want to save your outputs

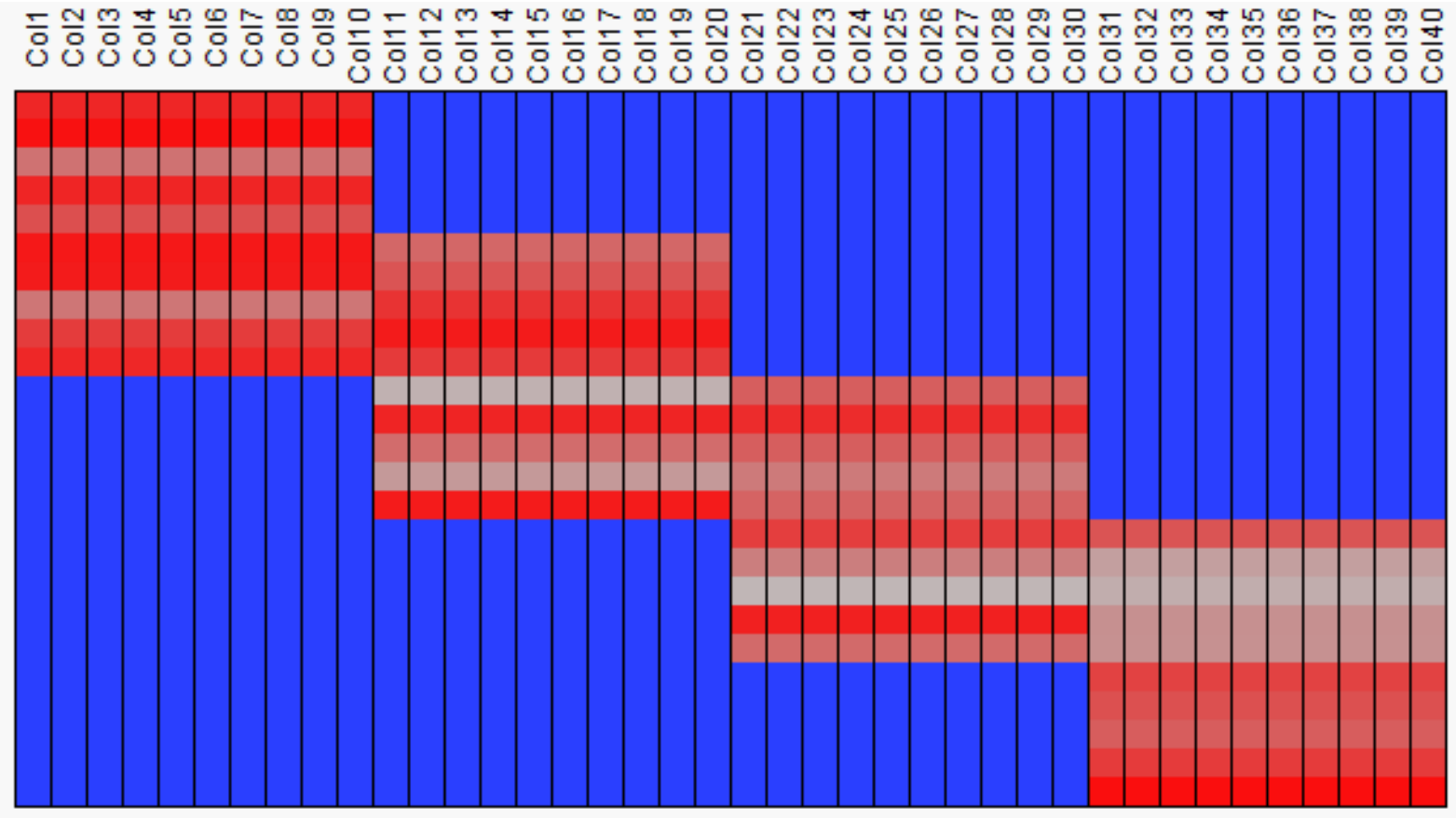
Resend the data saved in the directory of the “filename” to your visualization widgets (data tables etc.)



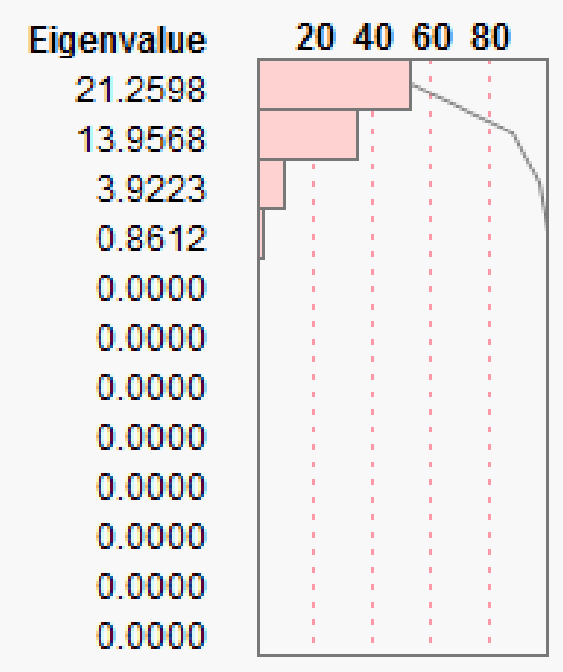
All outputs are saved in the chosen directory, and have their name beginning by the chosen “filename”

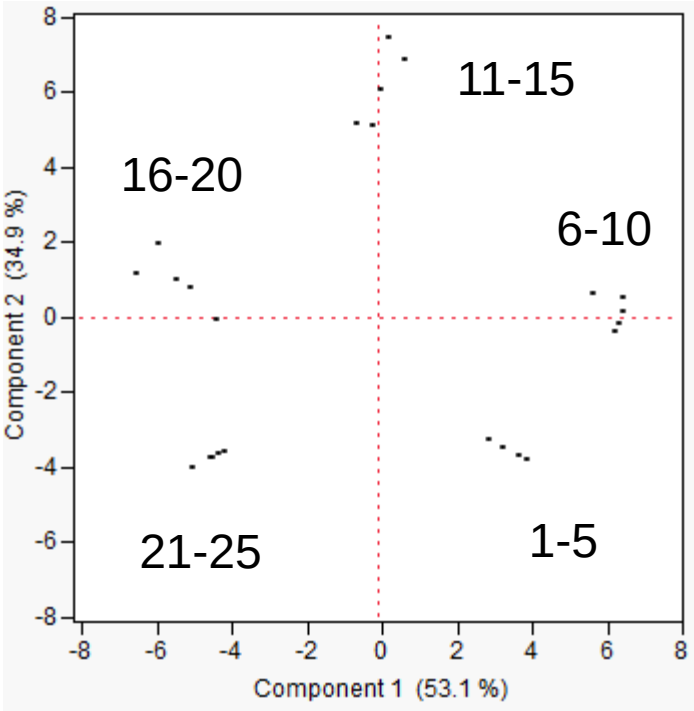
Saving is done automatically if you have checked the option “Save outputs”, however you can still save “manually” by clicking on the “browse” button and then on “save”.

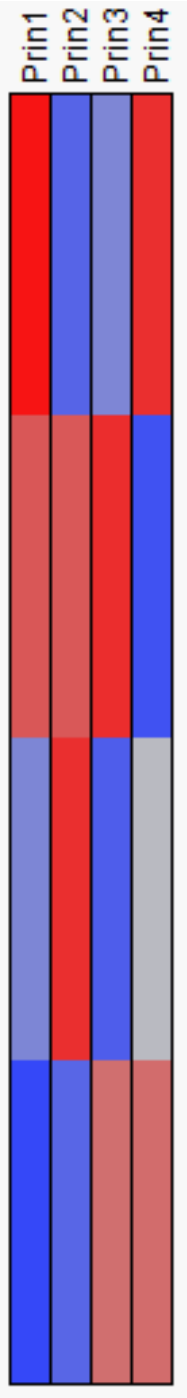
Example: a synthetic dataset



PCA outputs

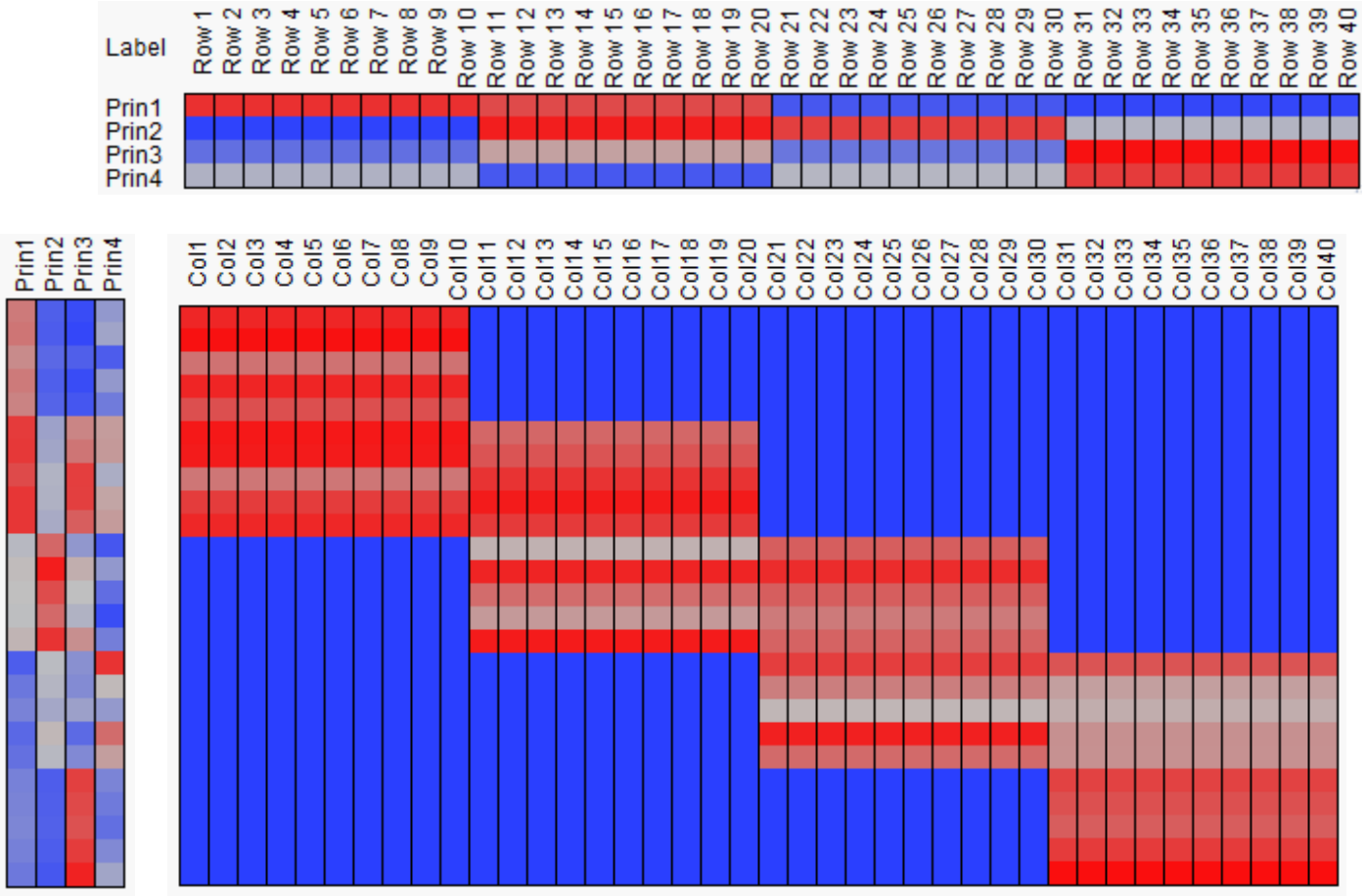




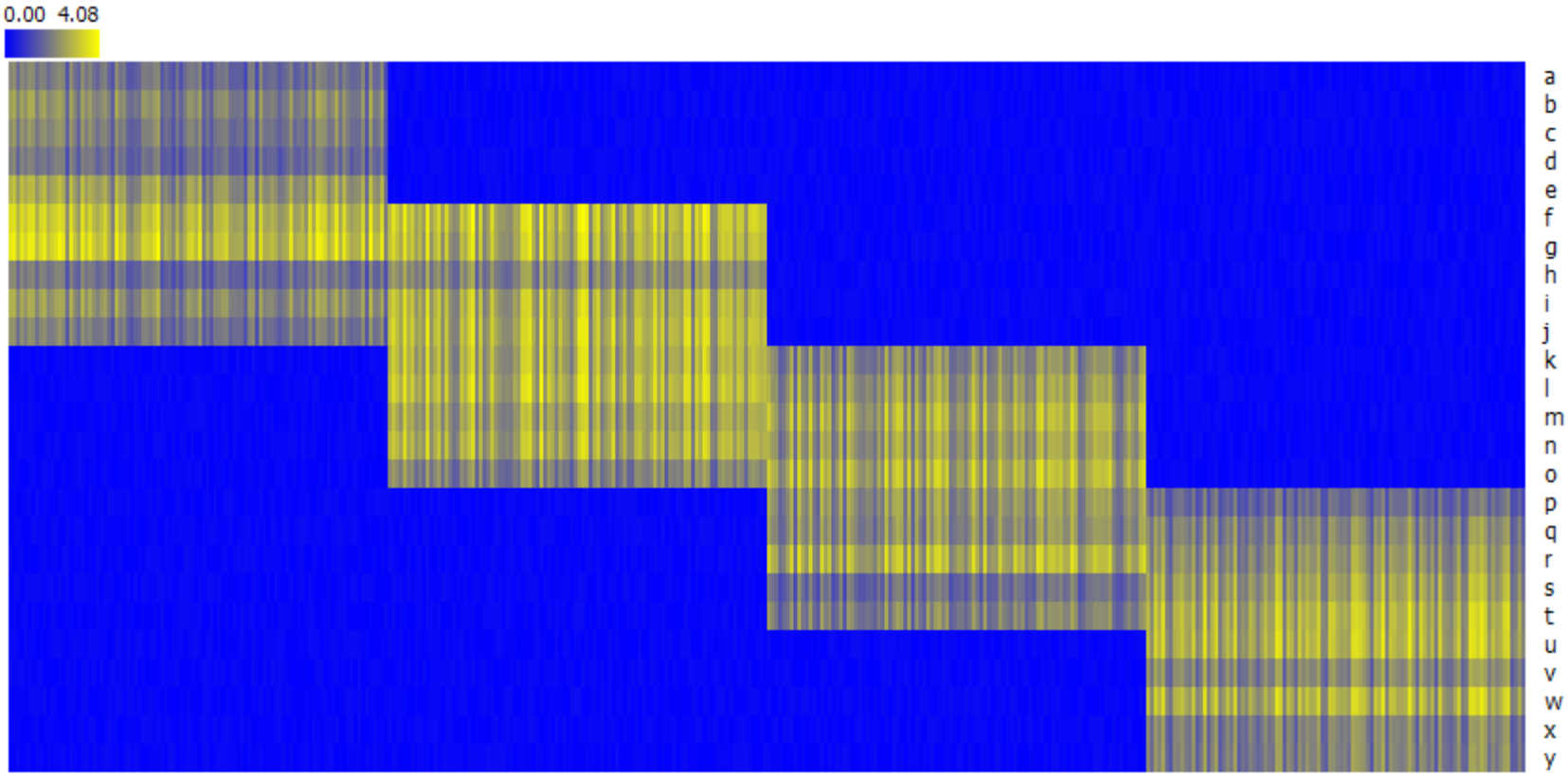


But we know we have 4 orthogonal components!
And we know we have 4 overlapping groups.

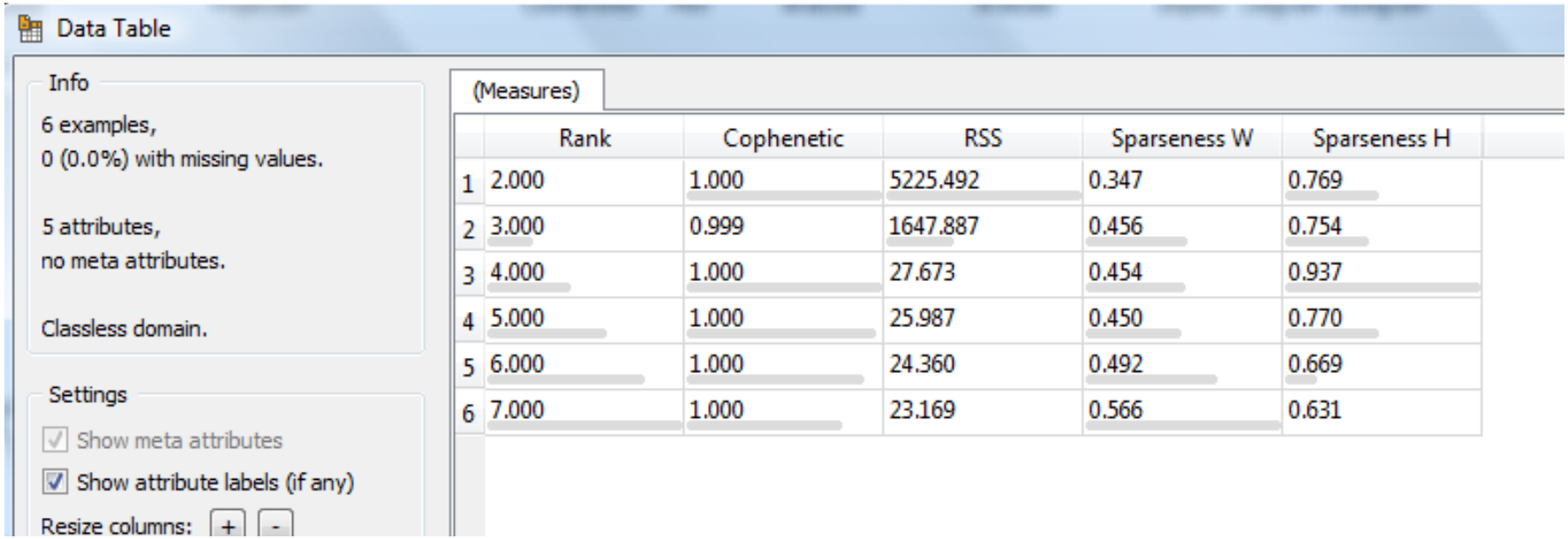
PCA ouptuts


Label
Row 1
Row 2
Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
Row 9
Row 10
Row 11
Row 12
Row 13
Row 14
Row 15
Row 16
Row 17
Row 18
Row 19
Row 20
Row 21
Row 22
Row 23
Row 24
Row 25
Row 26
Row 27
Row 28
Row 29
Row 30
Row 31
Row 32
Row 33
Row 34
Row 35
Row 36
Row 37
Row 38
Row 39
Row 40
Prin1
Prin2
Prin3
Prin4
Prin1
Prin2
Prin3
Prin4
Col1
Col2
Col3
Col4
Col5
Col6
Col7
Col8
Col9
Col10
Col11
Col12
Col13
Col14
Col15
Col16
Col17
Col18
Col19
Col20
Col21
Col22
Col23
Col24
Col25
Col26
Col27
Col28
Col29
Col30
Col31
Col32
Col33
Col34
Col35
Col36
Col37
Col38
Col39
Col40

Synthetic dataset: input matrix

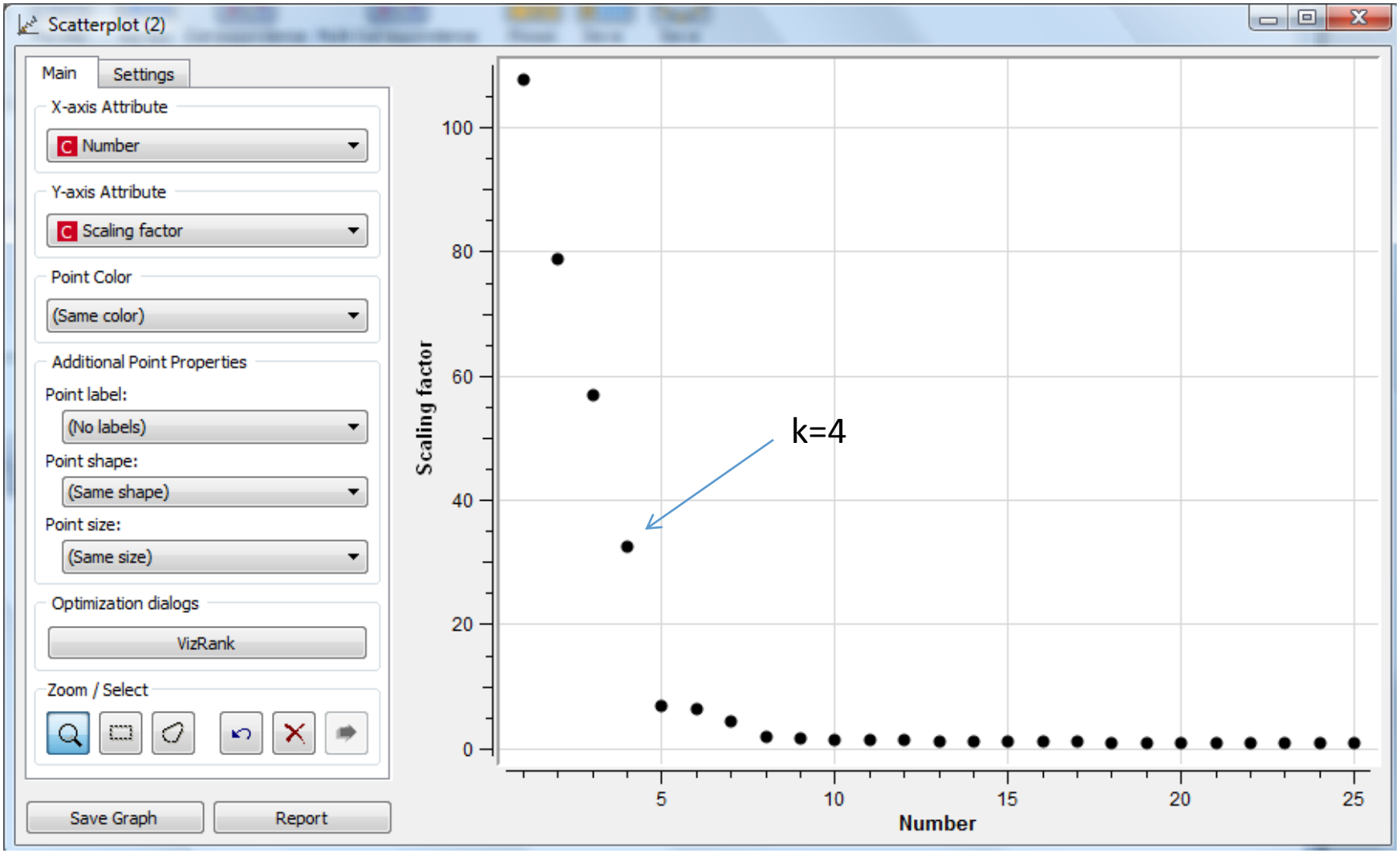


Several measures to choose the best k



	Rank	Cophenetic	RSS	Sparseness W	Sparseness H
1	2.000	1.000	5225.492	0.347	0.769
2	3.000	0.999	1647.887	0.456	0.754
3	4.000	1.000	27.673	0.454	0.937
4	5.000	1.000	25.987	0.450	0.770
5	6.000	1.000	24.360	0.492	0.669
6	7.000	1.000	23.169	0.566	0.631

Choose the best k by looking at the scree plot of SVD



Synthetic dataset: NMF outputs