

# **eResearch 2014**

## **Data Mining Workshop**

Scientific workflow management with ADAMS

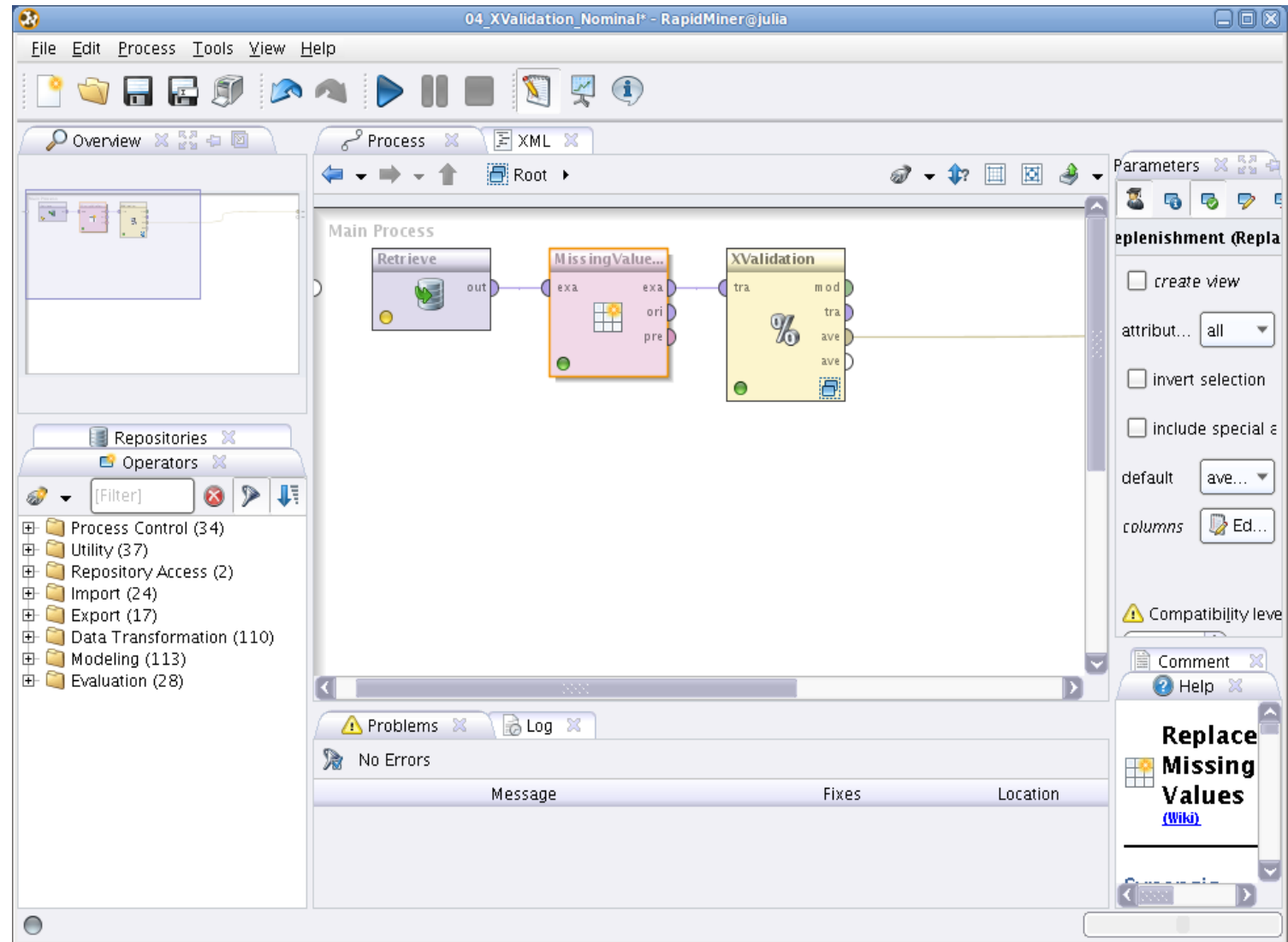
# Outline

---

- Basics
  - Bit different, eh?
  - Features, User interface
- Data Mining
  - Feature generation, evaluation, visualization
  - Generate and use model
- Scripting
  - R, Groovy, Jython

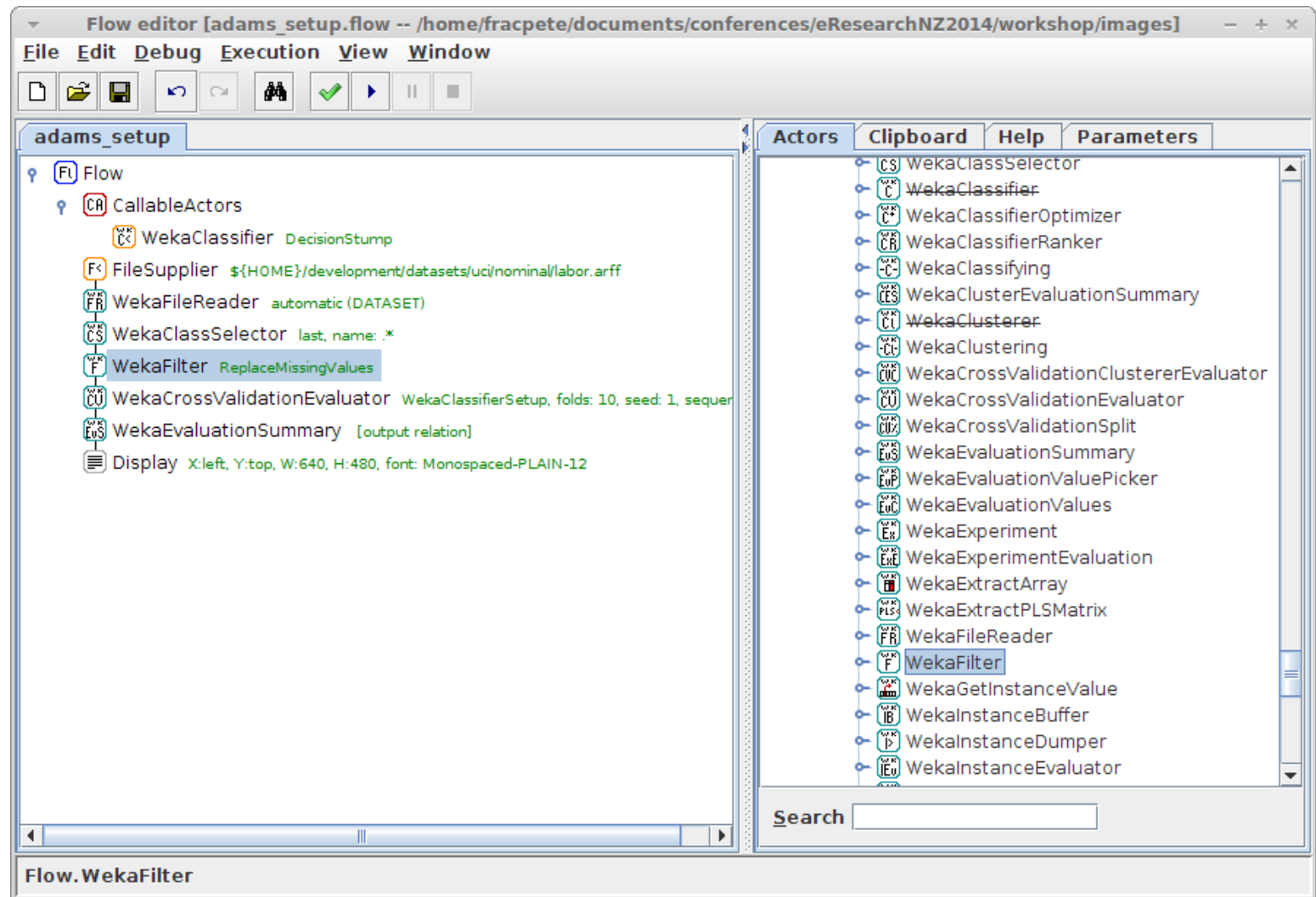
# Basics

- Canvas








# Basics (2)

- Tree



# Basics (3)

---

- How it works
  - No explicit connections
  - Actors snap into place in tree
  - Color coded with name (+ annotation)
    -  CA standalone (no I/O)
    -  F< source (only O)
    -  FR transformer (I/O)
    -  ≡ sink (only I)
    -  FI control actor (data flow)

# Basics (4)

---

- Advantages
  - compact layout
  - scales to 1000s of actors
  - context-aware adding of actors
  - interactive components
  - modular framework (Maven)
  - easy to add acctors: 1 Java class, 1 icon

# Basics (5)

---

- Limitations
  - only 1-to-n connections with tree layout
  - only single input/output
- Countermeasures
  - callable actors: n-to-1
  - containers: multiple outputs
  - variables: change options at runtime
  - internal storage: reuse data in multiple locations

# Features

Feature	Available
Machine learning/data mining	WEKA, WEKA webservice, MOA, MEKA, parameter optimization, experiment generation on-the-fly, setup generators, time series
Data processing	WEKA, R-Project, XML, XSLT, XPath, HTML, JSON
Streaming	MOA, Twitter (record/replay)
Spreadsheets	MS Excel (r/w), ODF (r/w), CSV (r/w), Gnumeric (r/w)
Imaging	ImageJ, JAI, ImageMagick, Gnuplot, OCR (tesseract)
Graphics output	BMP, JPG, PNG, TIF, PDF
Visualization	Scatter and line plots, Images, GIS (OpenStreetMap)
Scripting	Groovy, Jython
Documentation	DocBook, HTML
Web	HTTP, FTP, SFTP, SSH, Email, Webservices
Other	de/-compression (tar, zip, bzip2, gzip, lzma), Java code generation



# User interface

---

- Main interfaces available from  
WEKA, MOA, MEKA, ImageJ
- Visualization  
Preview browser, Time series explorer, Openstreetmap
- Tools  
Flow editor/runner, Text editor/diff, PDF Viewer,  
Spreadsheet file viewer
- Misc preference and configuration panels

# Data Mining

- **Lesson:** Image processing

- Caltech 101 dataset

[http://www.vision.caltech.edu/Image\\_Datasets/Caltech101/](http://www.vision.caltech.edu/Image_Datasets/Caltech101/)

- Pictures of objects belonging to 101 categories
- Examples



- Lesson downloads

<https://adams.cms.waikato.ac.nz/ernz2014.html>

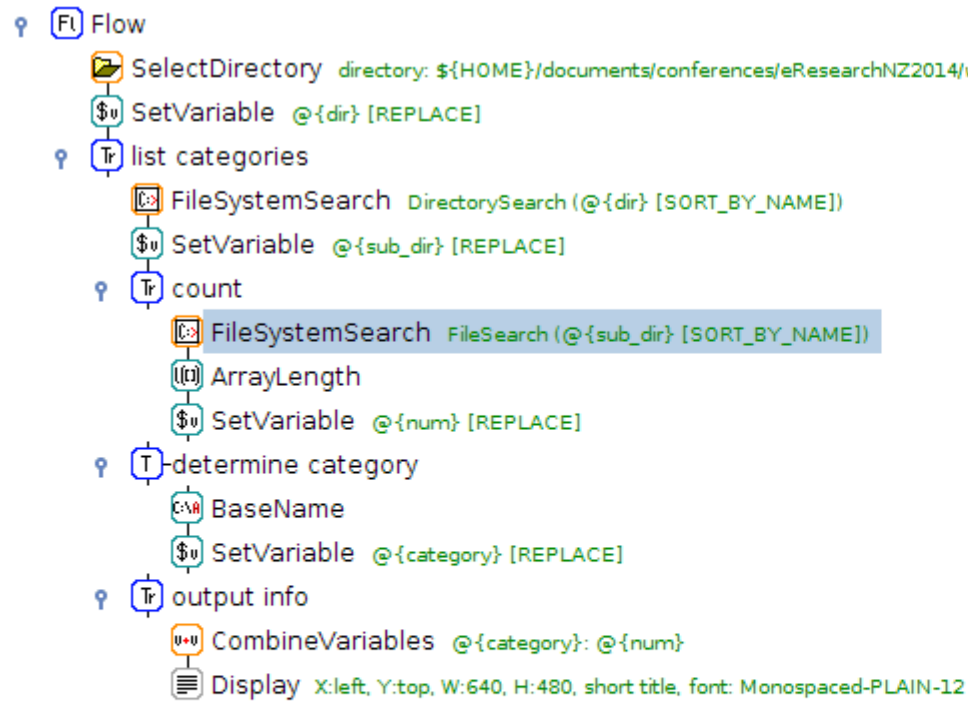
# Data Mining (2)

- Get a feel for the data



# Data Mining (3)

- Analyze categories



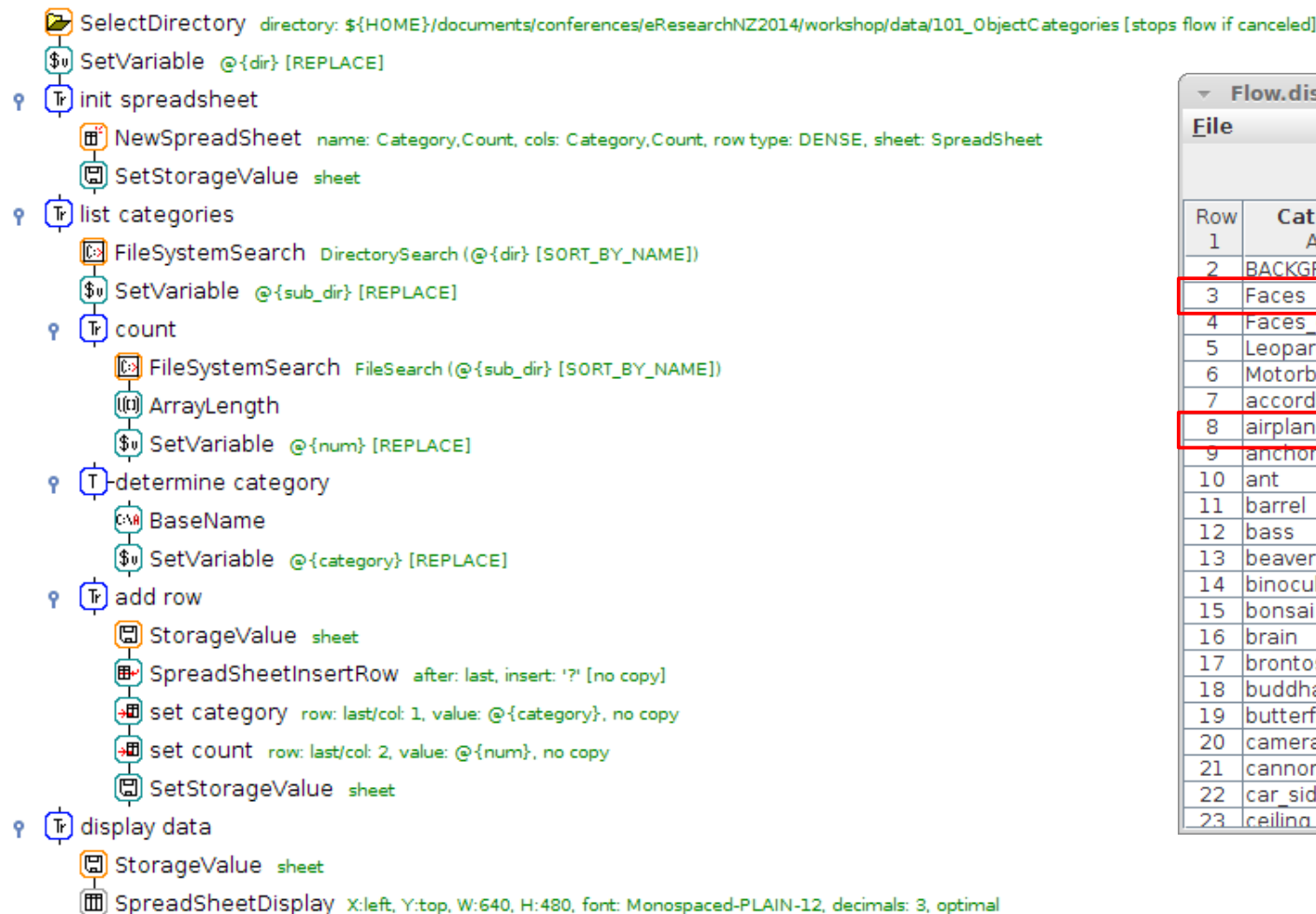
Display

File	Edit	View
BACKGROUND_Google: 467		
Faces: 435		
Faces_easy: 435		
Leopards: 200		
Motorbikes: 798		
accordion: 55		
airplanes: 800		
anchor: 42		
ant: 42		
barrel: 47		
bass: 54		
beaver: 46		
binocular: 33		
bonsai: 128		
brain: 98		
brontosaurus: 43		
buddha: 85		
butterfly: 91		
camera: 50		
cannon: 43		
car_side: 123		
ceiling_fan: 47		
cellphone: 59		
chair: 62		
chandelier: 107		
cougar_body: 47		
cougar_face: 69		
crab: 73		
crayfish: 70		
crocodile: 50		

# Data Mining (4)

- “Fancy” display of categories

Flow



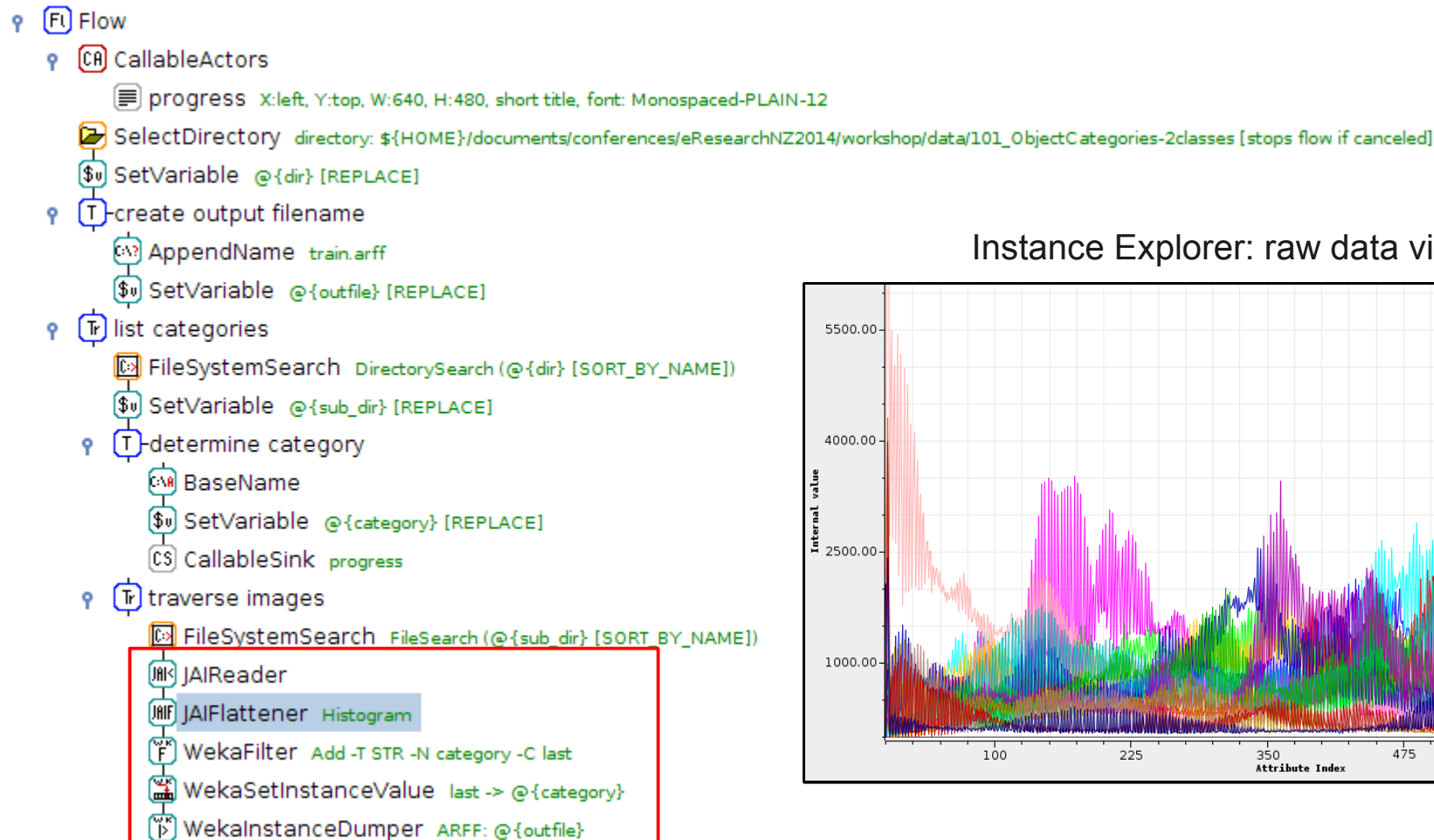
Flow.display data.SpreadSheetDisp

File

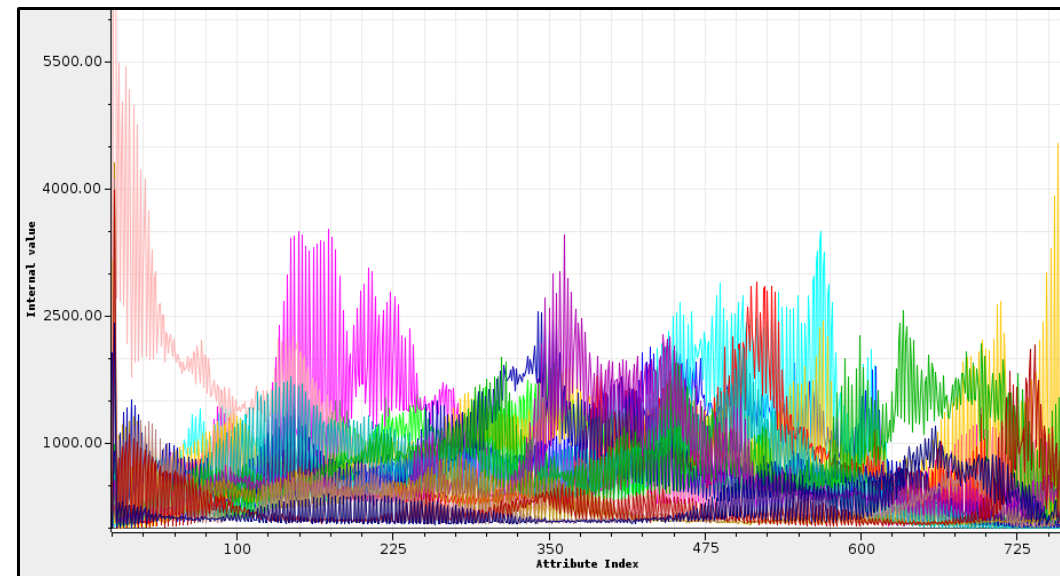
Category [1]

Row	Category	Count
1	A / 1	B / 2
2	BACKGROUND...	467
3	Faces	435
4	Faces_easy	435
5	Leopards	200
6	Motorbikes	798
7	accordion	55
8	airplanes	800
9	anchor	42
10	ant	42
11	barrel	47
12	bass	54
13	beaver	46
14	binocular	33
15	bonsai	128
16	brain	98
17	brontosaurus	43
18	buddha	85
19	butterfly	91
20	camera	50
21	cannon	43
22	car_side	123
23	ceiling fan	47

- Generate features/training data

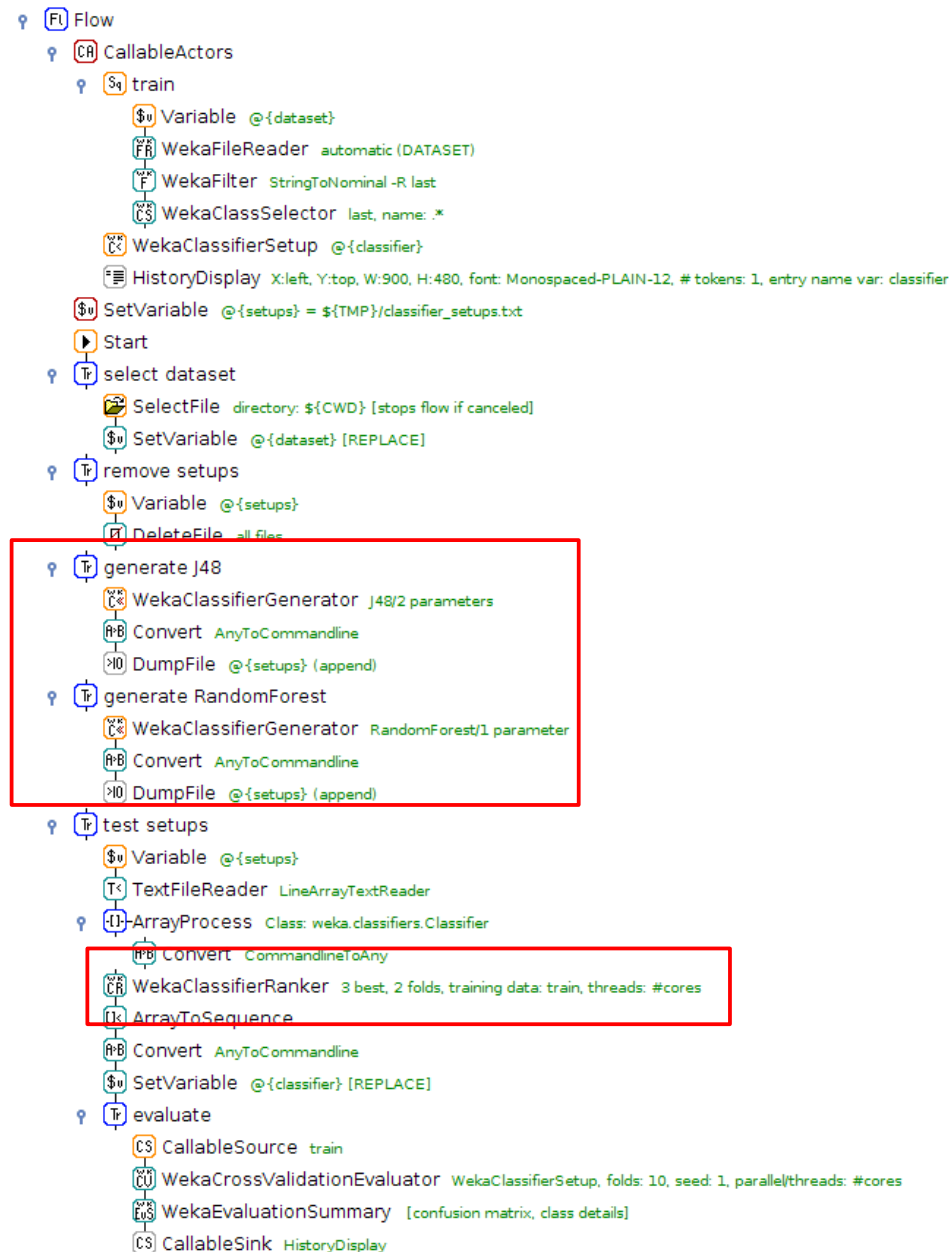


## Instance Explorer: raw data view



# Data Mining (6)

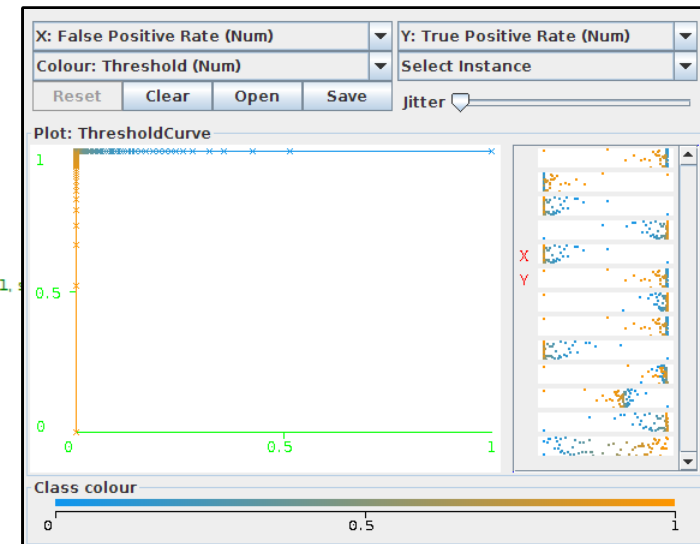
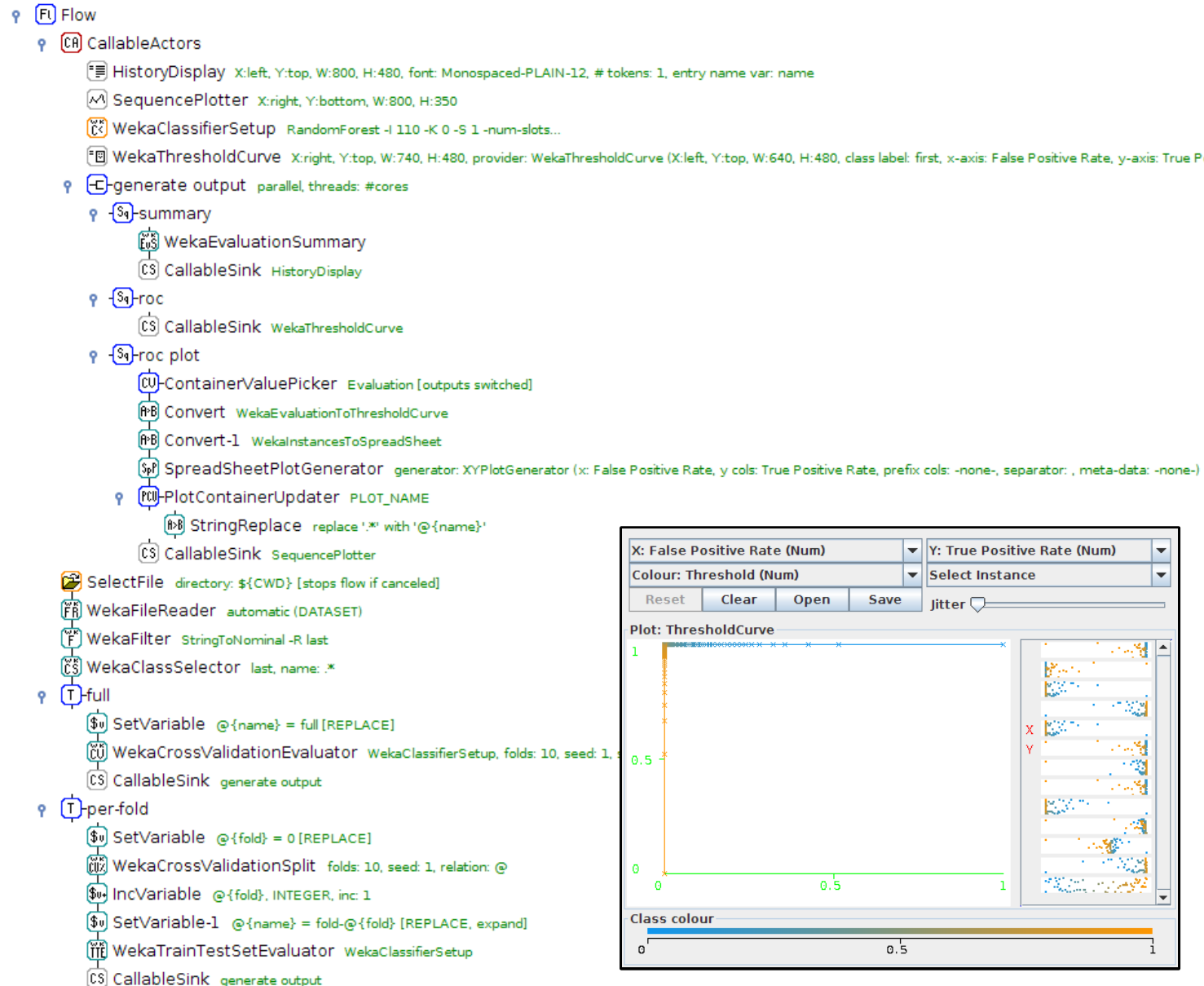
- Test classifier setups





# Data Mining (7)

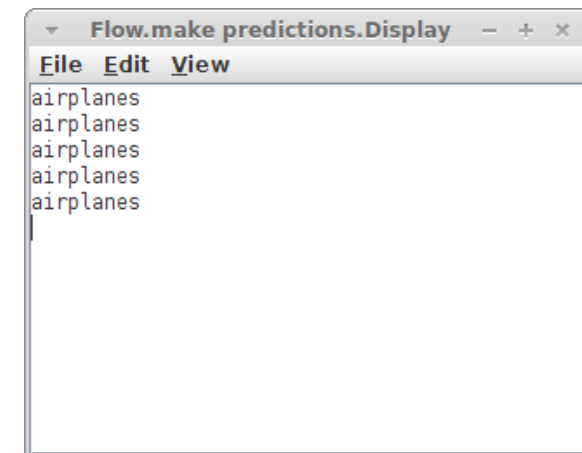
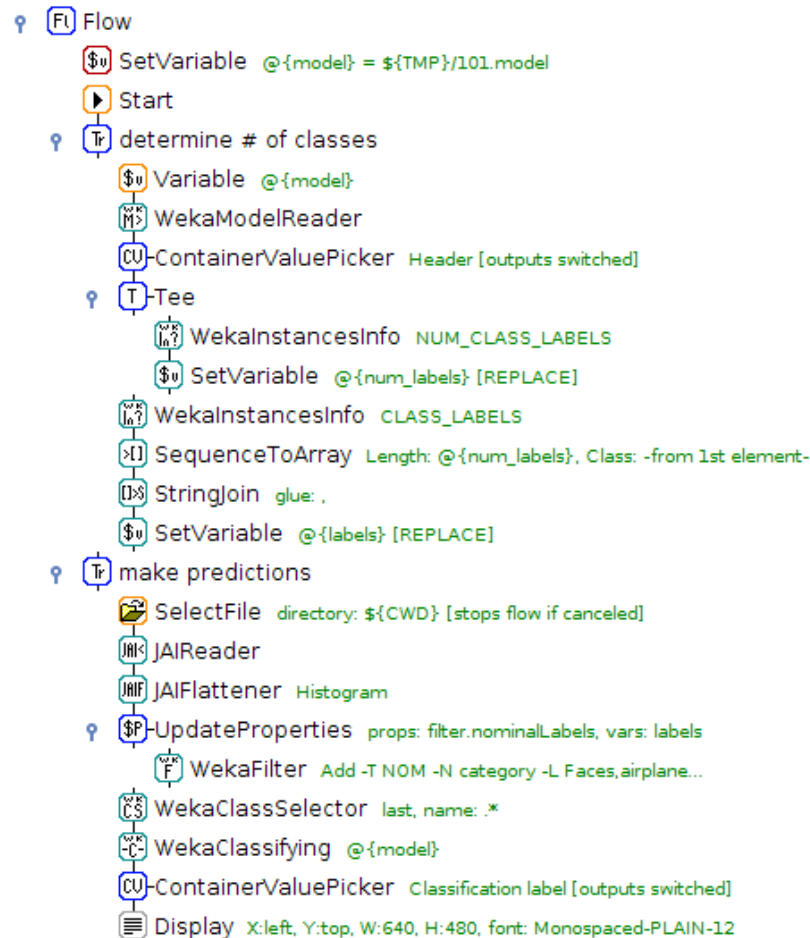
- Evaluate classifiers





# Data Mining (8)

- Build and use model



# Scripting

---

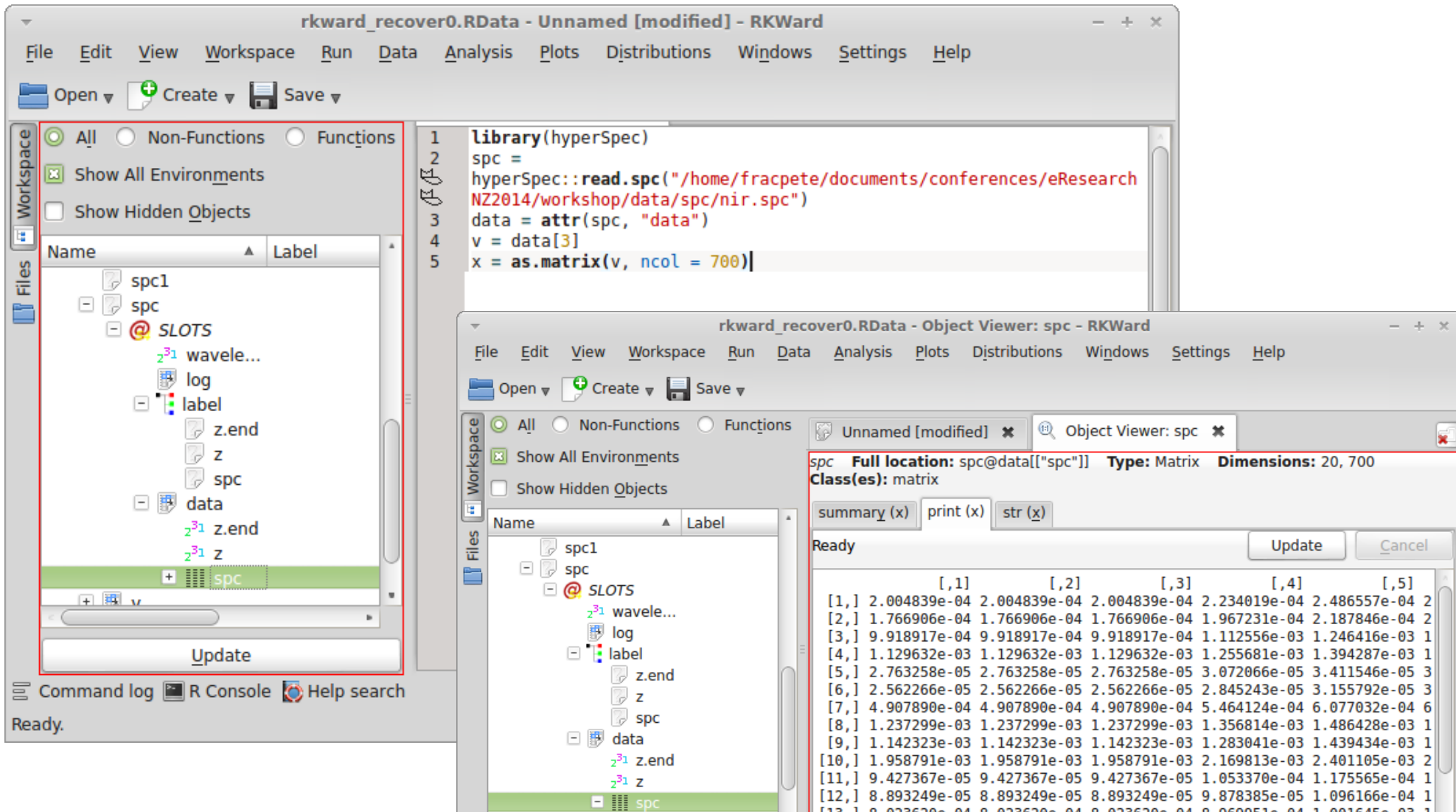
- **Lesson:** Scripting with R, Groovy, Jython
  - Near Infrared (NIR) data in proprietary Bruker SPC format
  - hyperSpec R package reads certain SPC files  
<http://cran.r-project.org/web/packages/hyperSpec/>
  - Install R packages

```
> install.packages("Rserve")  
> install.packages("hyperSpec")
```
  - Configure Jython \$HOME/.jython

```
python.security.respectJavaAccessibility=false
```

# Scripting (2)

- Use rkward to inspect data



The screenshot displays the RKward environment with two windows. The main window, titled 'rkward\_recover0.RData - Unnamed [modified] - RKward', shows a script being executed. The script defines a hyperSpec object 'spc' and extracts its data into a matrix 'x'. The 'Workspace' pane on the left shows the objects created during execution, including 'spc1', 'spc', 'SLOTS', 'wavele...', 'log', 'label', 'z.end', 'z', 'data', and 'x'. The 'Object Viewer: spc' window is open, displaying the details of the 'spc' object, which is a matrix with dimensions 20, 700. The viewer shows the first 13 rows of the matrix data.

```
1 library(hyperSpec)
2 spc =
3 hyperSpec::read.spc("/home/fracpete/documents/conferences/eResearch
4 NZ2014/workshop/data/spc/nir.spc")
5 data = attr(spc, "data")
6 v = data[3]
7 x = as.matrix(v, ncol = 700)
```

**Object Viewer: spc**

spc Full location: spc@data[["spc"]] Type: Matrix Dimensions: 20, 700  
Class(es): matrix

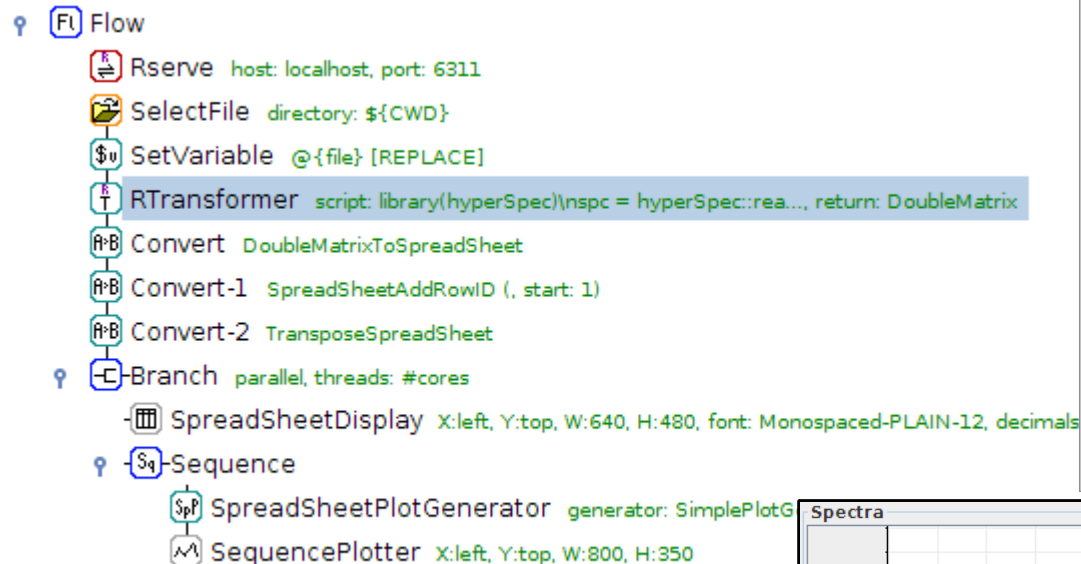
summary (x) print (x) str (x)

Ready [Update] [Cancel]

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	2.004839e-04	2.004839e-04	2.004839e-04	2.234019e-04	2.486557e-04
[2,]	1.766906e-04	1.766906e-04	1.766906e-04	1.967231e-04	2.187846e-04
[3,]	9.918917e-04	9.918917e-04	9.918917e-04	1.112556e-03	1.246416e-03
[4,]	1.129632e-03	1.129632e-03	1.129632e-03	1.255681e-03	1.394287e-03
[5,]	2.763258e-05	2.763258e-05	2.763258e-05	3.072066e-05	3.411546e-05
[6,]	2.562266e-05	2.562266e-05	2.562266e-05	2.845243e-05	3.155792e-05
[7,]	4.907890e-04	4.907890e-04	4.907890e-04	5.464124e-04	6.077032e-04
[8,]	1.237299e-03	1.237299e-03	1.237299e-03	1.356814e-03	1.486428e-03
[9,]	1.142323e-03	1.142323e-03	1.142323e-03	1.283041e-03	1.439434e-03
[10,]	1.958791e-03	1.958791e-03	1.958791e-03	2.169813e-03	2.401105e-03
[11,]	9.427367e-05	9.427367e-05	9.427367e-05	1.053370e-04	1.175565e-04
[12,]	8.893249e-05	8.893249e-05	8.893249e-05	9.878385e-05	1.096166e-04
[13,]	8.823620e-04	8.823620e-04	8.823620e-04	9.860051e-04	1.081645e-03

# Scripting (3)

- Connect to R via Rserve and run script to extract double matrix

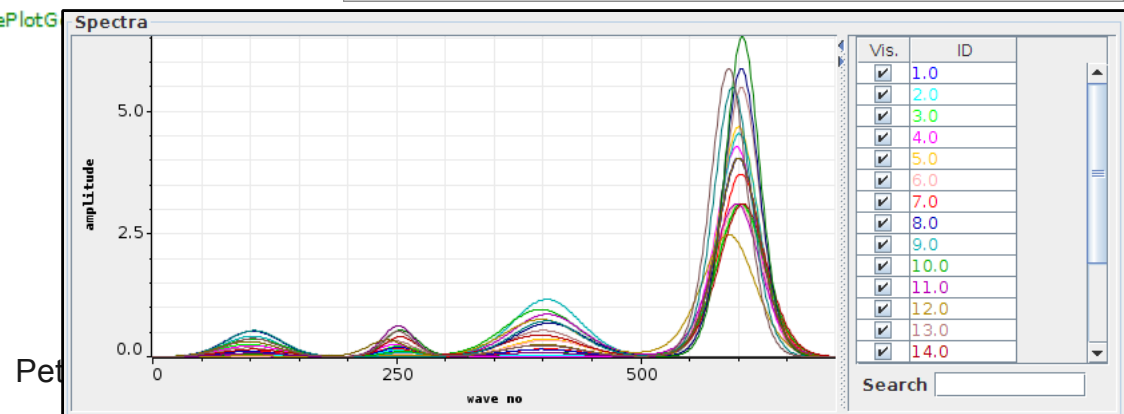


Flow.Branch.SpreadSheetDisplay

File

1.0 [1]

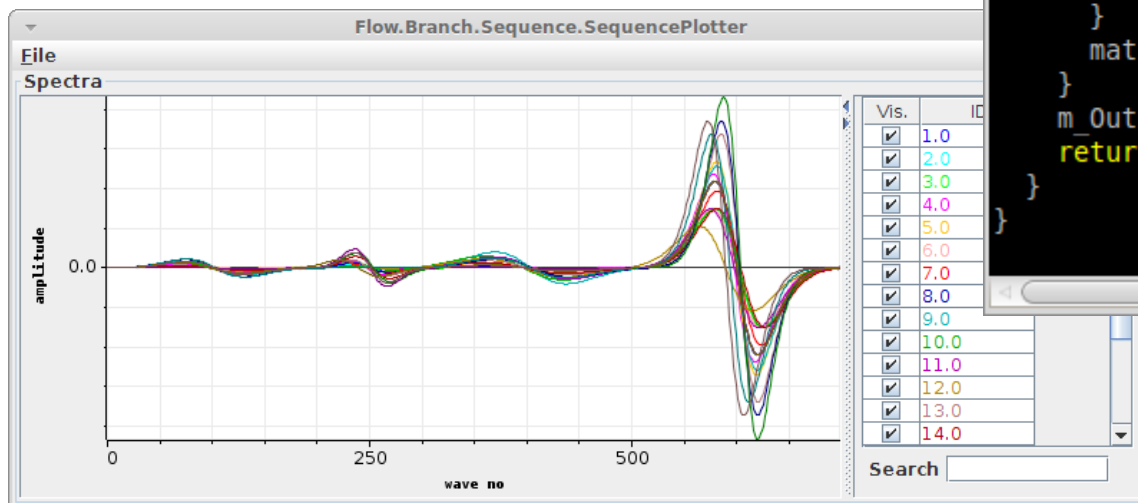
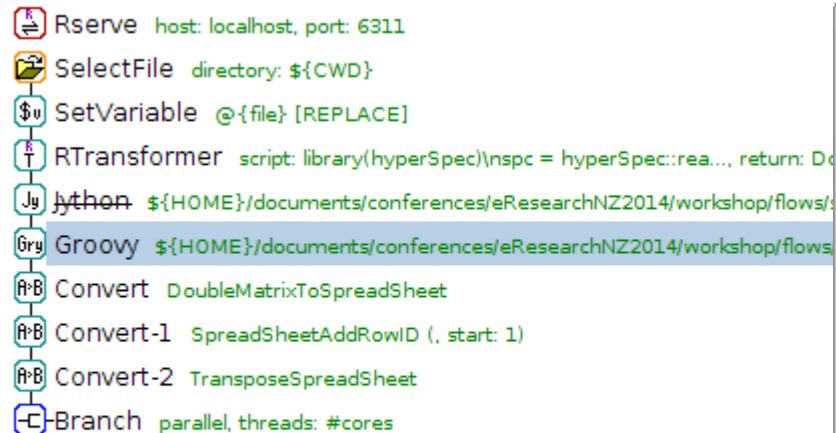
Row	1.0 A / 1	2.0 B / 2	3.0 C / 3	4.0 D / 4	5.0 E / 5	6.0 F / 6	7.0 G / 7	8.0 H / 8	9.0 I / 9
1									
2	0.00020	0.00018	0.00099	0.00112	0.00003	0.00003	0.00049	0.00124	0.00114
3	0.00020	0.00018	0.00099	0.00112	0.00003	0.00003	0.00049	0.00124	0.00114
4	0.00020	0.00018	0.00099	0.00112	0.00003	0.00003	0.00049	0.00124	0.00114
5	0.00022	0.00020	0.00111	0.00126	0.00003	0.00003	0.00055	0.00136	0.00128
6	0.00025	0.00022	0.00125	0.00139	0.00003	0.00003	0.00061	0.00149	0.00144
7	0.00027	0.00024	0.00139	0.00155	0.00004	0.00003	0.00068	0.00163	0.00161
8	0.00031	0.00027	0.00156	0.00171	0.00004	0.00004	0.00075	0.00178	0.00181
9	0.00034	0.00029	0.00174	0.00190	0.00005	0.00004	0.00083	0.00194	0.00202
10	0.00038	0.00033	0.00194	0.00210	0.00005	0.00005	0.00092	0.00212	0.00224
11	0.00042	0.00037	0.00216	0.00232	0.00006	0.00005	0.00102	0.00231	0.00251
12	0.00046	0.00040	0.00239	0.00256	0.00006	0.00006	0.00111	0.00252	0.00280
13	0.00051	0.00045	0.00267	0.00282	0.00006	0.00006	0.00124	0.00274	0.00312
14	0.00055	0.00049	0.00296	0.00310	0.00008	0.00006	0.00137	0.00298	0.00346
15	0.00062	0.00054	0.00328	0.00341	0.00008	0.00008	0.00151	0.00323	0.00385
16	0.00068	0.00059	0.00363	0.00375	0.00009	0.00008	0.00166	0.00351	0.00426
17	0.00075	0.00066	0.00402	0.00411	0.00010	0.00009	0.00182	0.00380	0.00472
18	0.00082	0.00072	0.00443	0.00451	0.00011	0.00010	0.00200	0.00412	0.00523
19	0.00090	0.00079	0.00489	0.00493	0.00012	0.00011	0.00220	0.00445	0.00577
20	0.00099	0.00087	0.00539	0.00540	0.00012	0.00012	0.00240	0.00481	0.00637
21	0.00109	0.00095	0.00593	0.00590	0.00013	0.00012	0.00263	0.00520	0.00703
22	0.00119	0.00103	0.00651	0.00644	0.00016	0.00013	0.00288	0.00561	0.00774
23	0.00130	0.00112	0.00715	0.00702	0.00017	0.00016	0.00314	0.00604	0.00851
24	0.00141	0.00123	0.00784	0.00765	0.00019	0.00017	0.00343	0.00650	0.00935



# Scripting (4)

- Use Groovy to compute 1<sup>st</sup> derivative

Fl Flow



```

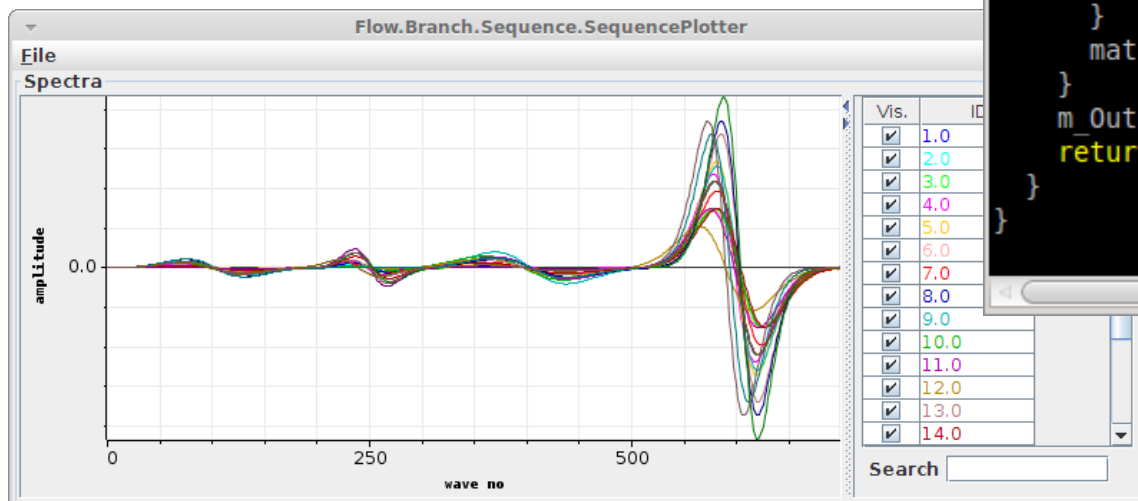
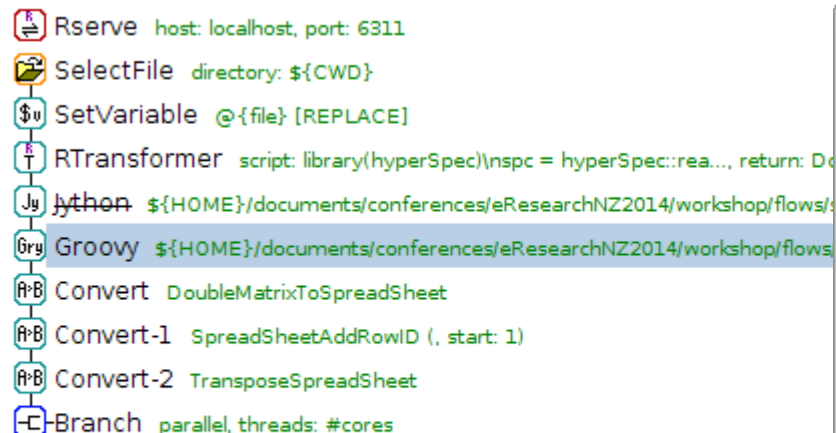
spc2spreadsheet.groovy ...4/workshop/flows) - GVIM2
File Edit Tools Syntax Buffers Window Programming Spell Help

*
* @return null if everything is fine, otherwise
*/
protected String doExecute() {
    Double[][] input = m_InputToken.getPayload()
    Double[][] matrix = new Double[input.length][input[0].length]
    for (int i = 0; i < input.length; i++) {
        Double[] row = new Double[input[i].length - 1]
        for (int n = 1; n < input[i].length; n++) {
            row[n - 1] = input[i][n] - input[i][n - 1]
        }
        matrix[i] = row
    }
    m_OutputToken = new Token(matrix)
    return null
}
47,1 97%
  
```

# Scripting (4)

- Use Groovy to compute 1<sup>st</sup> derivative

Fl Flow



```

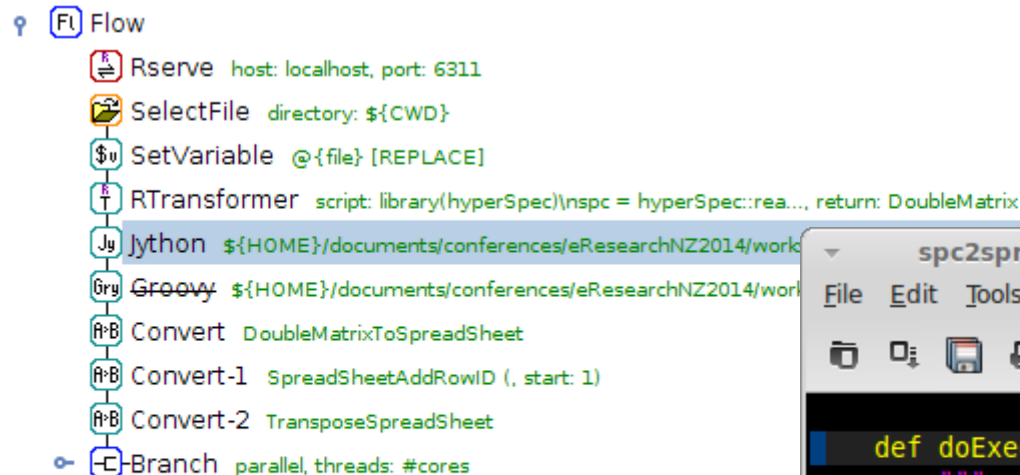
spc2spreadsheet.groovy ...4/workshop/flows) - GVIM2
File Edit Tools Syntax Buffers Window Programming Spell Help

*
* @return null if everything is fine, otherwise
*/
protected String doExecute() {
    Double[][] input = m_InputToken.getPayload()
    Double[][] matrix = new Double[input.length][]
    for (int i = 0; i < input.length; i++) {
        Double[] row = new Double[input[i].length - 1]
        for (int n = 1; n < input[i].length; n++) {
            row[n - 1] = input[i][n] - input[i][n - 1]
        }
        matrix[i] = row
    }
    m_OutputToken = new Token(matrix)
    return null
}
47,1 97%
  
```



# Scripting (5)

- Use Jython to computer 1<sup>st</sup> derivative



```
spc2spreadsheet.py + (~/.do...2014/workshop/flows) - GVIM3
File Edit Tools Syntax Buffers Window Programming Spell Help

def doExecute(self):
    """
    Executes the flow item.
    @return: None if everything is fine, otherwise error message
    """
    inp = self.m_InputToken.getPayload()
    rows = []
    for n in xrange(len(inp)):
        rowin = inp[n]
        rowout = zeros(len(rowin) - 1, java.lang.Double)
        for i in xrange(len(rowin) - 1):
            rowout[i] = rowin[i+1] - rowin[i]
        rows.append(rowout)
    matrix = array(rows, Class.forName('[Ljava.lang.Double;'))
    self.m_OutputToken = Token(matrix)
    return None
```

61,1 98%

## Questions?

