A state-of-the-art machine learning pipeline for the analysis of spatial proteomics data

L. Gatto[1,2,*], LM. Breckels[1,2], T. Naake[1,2], S. Wieczorek[3], T. Burger[3] and KS. Lilley[2]

[1]Computational Proteomics Unit and [2]Cambridge Centre for Proteomics, Department of Biochemistry, University of Cambridge, UK
[3]Université Grenoble-Alpes, CEA (iRSTV/BGE), INSERM (U1038), CNRS (FR3425), 38054 Grenoble, France

*lg390@cam.ac.uk http://cpu.sysbiol.cam.ac.uk

Introduction

Organelle proteomics, or spatial proteomics, is the systematic study of proteins and their assignment to subcellular niches including organelles. `pRoloc` and `pRolocGUI` are `R`/Bioconductor packages that implement all the necessary tools for the sound and reproducible analysis and interactive exploration of spatial proteomics data from any type of experiment such as LOPIT, PCP or PCP-SILAC.

Below, we illustrate a typical `pRoloc` analysis pipeline:

1. Loading data into `R` and adding markers
2. QC: checking resolution in the data and organelle markers
3. Detection of new organelle clusters
4. Classification of unlabelled proteins
5. Results, interpretation and visualisation

1) Data input

We start by reading quantitative data from 10 fractions (at positions 2 to 11 in the data spreadsheet) sampled along a separation gradient from a `csv` file and add *Drosophila* organelle markers. This code creates an `MSnSet` data object (named `spat`) that stores the quantitative data and the metadata, and can subsequently be easily manipulated, plotted and further processed.

```
spat <- readMSnSet2("quant-data.csv", ecols = 2:11)
spat <- addMarkers(spat, "dmel")
```

	Fraction_1	Fraction_2	...	Fraction_m
prot_1	$q_{1,1}$	$q_{1,2}$	...	$q_{1,\,m}$
prot_2	$q_{2,1}$	$q_{2,2}$	...	$q_{2,\,m}$
prot_3	$q_{3,1}$	$q_{3,2}$	...	$q_{3,\,m}$
prot_4	$q_{4,1}$	$q_{4,2}$	...	$q_{4,\,m}$
⋮	⋮	⋮	⋮	⋮
prot_i	$q_{i,1}$	$q_{i,2}$	...	$q_{i,\,m}$
⋮	⋮	⋮	⋮	⋮
prot_n	$q_{n,1}$	$q_{n,2}$	...	$q_{n,\,m}$

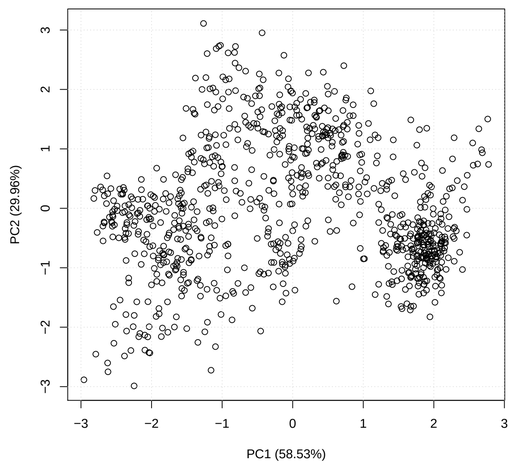
	markers	
...	*unknown*	...
	$organelle_1$	
	unknown	
	$organelle_2$	
⋮	⋮	⋮
	$organelle_k$	
⋮	⋮	⋮
...	*unknown*	

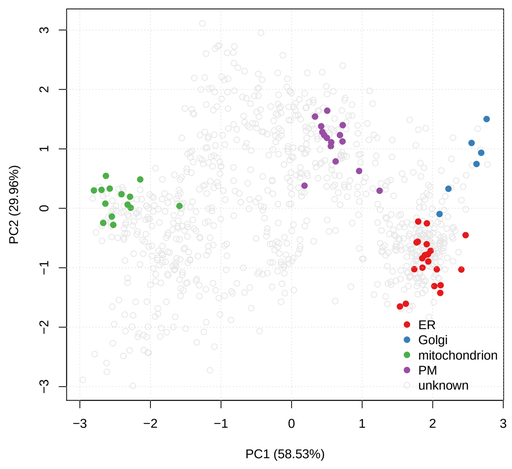
	Fraction_1	Fraction_2	...	Fraction_m
	...	...	...	...
	⋮	⋮	⋮	⋮
	...	...	...	...

2) Quality control

We verify on PCA plots that there is structure in the data (i.e we distinguish well defined clusters, left) and that the markers defined well resolved organelle clusters (right).

```
plot2D(spat)
addLegend(spat)
```





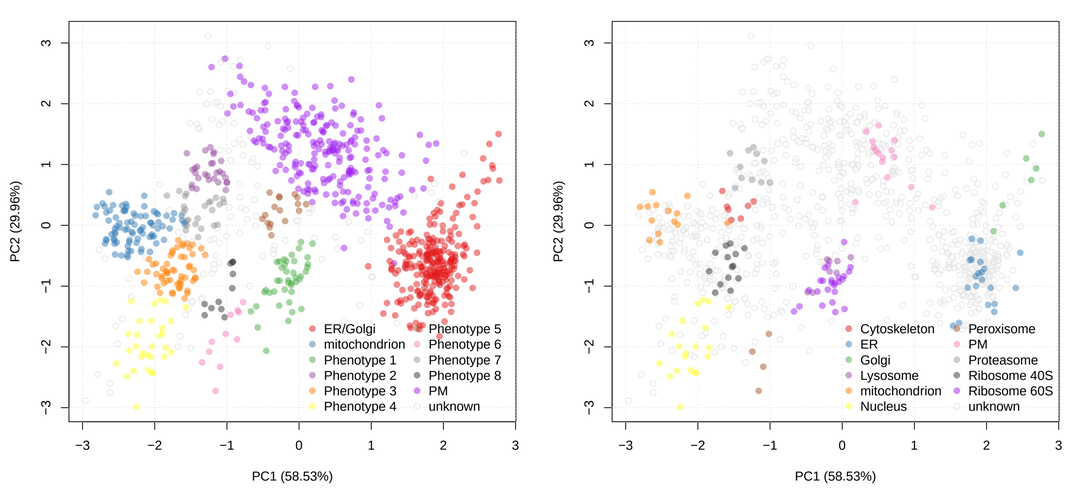
Gatto *et al.* *Mass-spectrometry-based spatial proteomics data analysis using pRoloc and pRolocdata.* Bioinformatics. 2014 May 1;30(9):1322-4.
Gatto *et al. A foundation for reliable spatial proteomics data analysis.* Mol Cell Proteomics. 2014 Aug;13(8):1937-52.

software `http://is.gd/pRoloc`
documentation `http://is.gd/pRoloc_tutorial`
GUI `http://is.gd/pRolocGUI`
data `http://is.gd/pRolocdata`
Videos `http://is.gd/R4ProteomicsVideos`

3) Novelty detection

Our manually curated markers do not cover the entire sub-cellular diversity. We use a semi-supervised machine learning algorithms (Breckels *et al.*, J Proteomics. 2013 Aug 2;88:129-40.) to identify new putative organelle clusters, called *phenotypes* (left), which require validation by the user (right).

```
spat <- phenoDisco(spat)
plot2D(spat, fcol = "pd")
```



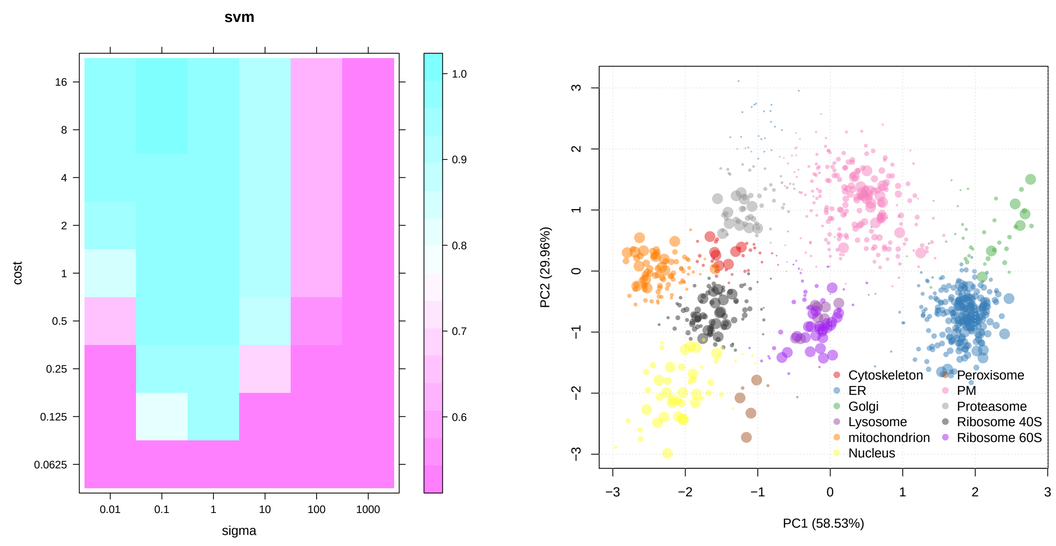
4) Classification

We can now classify unlabelled proteins to any of the augmented classes using a supervised machine (SVM) learning algorithms, for example, a support vector machine classifier. It is essential to tune the classification model parameters (here $\boldsymbol{sigma}$ and $\boldsymbol{cost}$) prior to actual classification (left).

```
params <- svmOptimisation(spat, fcol = "pd.markers")
spat <- svmClassification(spat, params, fcol = "pd.markers")
```

The classification algorithm calculates classification probabilities that reflect the distance of a protein to the decision boundaries defined by the SVM model (right). Eventually, the data can be exported to a spreadsheet file.

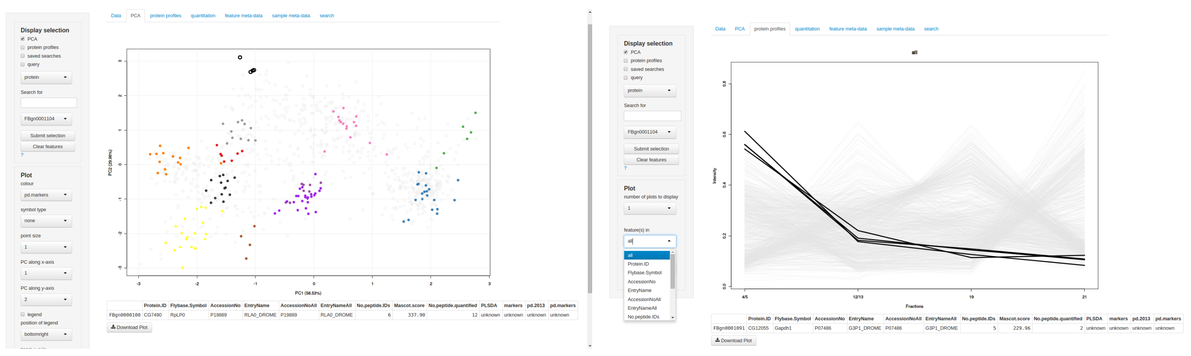
```
ptsze <- exp(fData(spat)$svm.scores) - 1
plot2D(spat, fcol = "svm", cex = ptsze)
write.exprs(spat, file = "spat-results.csv")
```



5) Interpretation

The graphical user interface implemented in the `pRolocGUI` package enables one to interactively explore the data.

```
library("pRolocGUI")
pRolocVis(spat)
```



This work was supported by the **European Union 7th Framework Program PRIME-XS project** and a **BBSRC Tools and Resources Development Fund**.