Assignment of Proteins to Subcellular Locations Using protlocassign

Dirk Moore

2020-04-10

Determining the locations of proteins in the cell is an important but complex problem. The major biochemical approach for this involves centrifugation-based methods to partially separate different organelles and other cellular compartments and then to determine the relative distribtuion of different proteins among the centrifugation fractions. The location of proteins of interest are then inferred by comparing their distribution to those of known locations. This package takes data from centrifugation fractions for large numbers of proteins to determine their relative abundance among the fractions, and by extension to their relative abundance among subcelluar locations.

This package implements a subcellular protein assignment procedure as “constrained proportional assignmet”, or CPA (Jadot et al, 2016). The input data consists of a file with one column, geneName, of gene/protein identifiers, or of peptide identifiers, and multiple columns of the relative abundances of the proteins among the centrifugal fractions. The sum of the abundance levels is constrained to 1 but can be further transfromed to improve the quality of the fit. Another key input to the program is a curated list of reference genes, and their subcellular locations. The method first determines the profiles of abundance levels of genes in a curated reference set. Then, for each gene, it finds either a profile matching a single compartment or a linear combination of compartment profiles that match the profile of that gene. The relative weights of the linear combination in principle reflect the relative abundance of a given protein among different compartments. The CPA method, unlike other previously described methods, can thus account for proteins that have multiple residences, and estimate the relative proportion among these residences. The method can also be applied to classify individual peptides, gaining potential insight into subcellular location of protein isoforms. There is also a tool to compute nearest neighbor distances of proteins, allowing grouping of proteins with similar distributions.

## Tutorial on gene assignment

We illustrate with an example. Tannous et al. (2020) presented abundance levels of proteins among six fractions from a differential centrifugation experiment: N, M, L1, L2, P, and S, and three fractions from a Nycodenz density gradient separation of the differential fraction L1 (Nyc1, Nyc2, and Nyc3). Eight subcellular compartments were considered: nucleus, mitochondria, lysosomes, peroxisomes, Golgi apparatus, plasma membrane, and cytosol. The CPA method assigns each of a large number of genes to one or more of these compartments, based on profiles from a set of reference genes.

## Installation

Start by installing the devtools package from CRAN, by typing

install.packages("devtools")

Then install the protlocassign package from the github repository by typing

devtools::install\_github("mooredf22/protlocassign0p1p1")

This will make the programs and datasets available. Also, the BB library is required; it may be downloaded from CRAN by typing

install.packages("BB")

Once you have installed a package, you do not need to re-install it the next time you start up R.

These are the only packages you need to run the programs that work with data that are presented as mean levels for a protein/gene. In another vignette there is a description of R functions that work with data at the spectral level; additional packages will be needed to process spectral-level data and produce gene-level data.

In this example we will work with the TMT ms2 data from Tannous et al. (2020). To get started, load the package and look at the dimensions of the data set and then the first few rows:

library(protlocassign)  
#> Loading required package: lme4  
#> Loading required package: Matrix  
#> Loading required package: BB  
dim(geneProfileSummaryTMTms2)  
#> [1] 7893 12  
options(digits=3)  
head(geneProfileSummaryTMTms2)  
#> geneName N M L1 L2 P S Nyc.1 Nyc.2  
#> 1 2900026A02RIK 0.1025 0.0851 0.1125 0.0930 0.1051 0.1254 0.1524 0.1484  
#> 2 A1CF 0.1577 0.0805 0.0765 0.1264 0.2554 0.1118 0.0657 0.0634  
#> 3 A930018M24RIK 0.0544 0.0503 0.1129 0.0760 0.0399 0.0507 0.1358 0.3969  
#> 4 AAAS 0.3692 0.0388 0.0619 0.0995 0.2070 0.0336 0.0709 0.0699  
#> 5 AABR07001519.1 0.0579 0.0528 0.0732 0.0687 0.0504 0.3653 0.1381 0.1431  
#> 6 AABR07002683.1 0.0720 0.0749 0.1059 0.1189 0.1090 0.0787 0.2232 0.1334  
#> Nyc.3 Nspectra Nseq  
#> 1 0.0758 4 3  
#> 2 0.0624 25 16  
#> 3 0.0830 1 1  
#> 4 0.0492 3 3  
#> 5 0.0505 1 1  
#> 6 0.0839 2 2

The data set gives relative fraction levels for 7893 genes. The last two columns give the number of spectra and the number of peptides (sequences) for each gene. (The spectra were averaged using a nested random effects model described in Jadot, 2016, to produce the given average for each gene.)

The reference genes, which are from Jadot et al (2016), may be accessed as follows:

dim(refLocProteinsJadot)  
#> [1] 39 2  
refLocProteinsJadot[c(1,2,6,7,12,13,17,18,21,22,27,28,31,32,35,36),]  
#> geneName referenceCompartment  
#> 1 ADH1 Cytosol  
#> 2 DPP3 Cytosol  
#> 6 CANX ER  
#> 7 GANAB ER  
#> 12 B4GALT1 Golgi  
#> 13 MAN1A2 Golgi  
#> 17 ACP2 Lysosome  
#> 18 CTSD Lysosome  
#> 21 Mt-CO2 Mitochondria  
#> 22 CPS1 Mitochondria  
#> 27 HIST1H1D Nucleus  
#> 28 LMNA Nucleus  
#> 31 CAT Peroxisomes  
#> 32 HAO1 Peroxisomes  
#> 35 ATP1A1 Plasma membrane  
#> 36 CD38 Plasma membrane

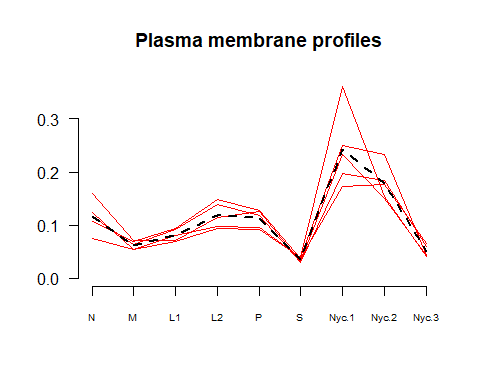
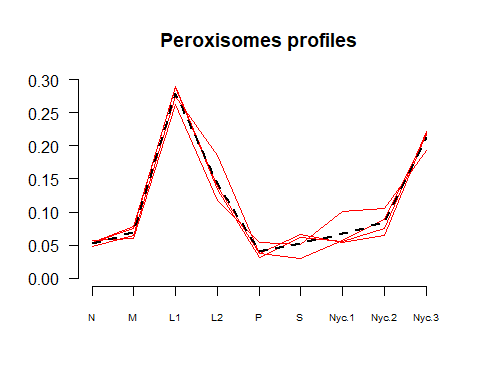
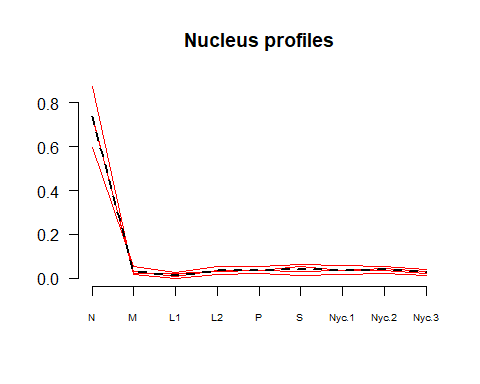
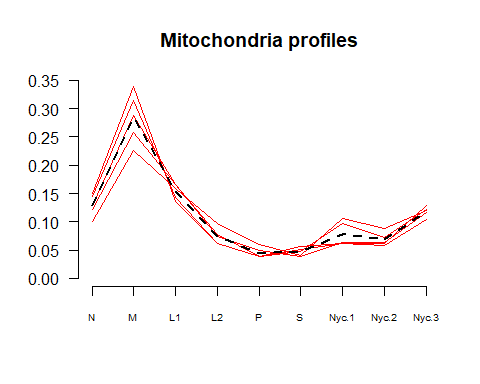
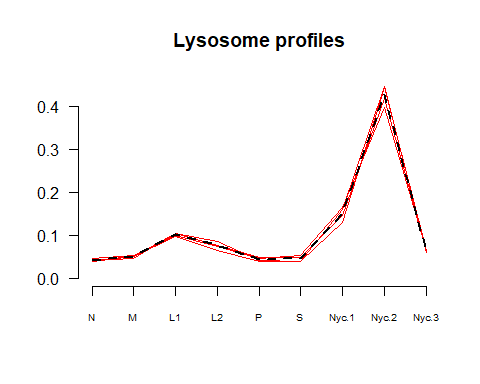
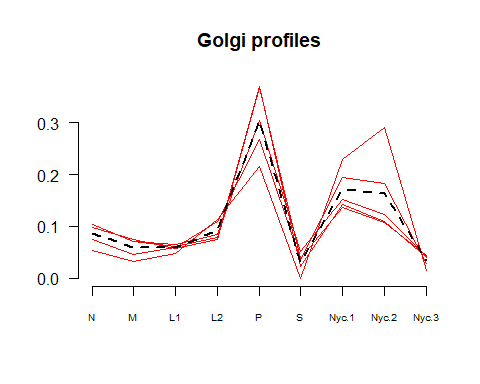
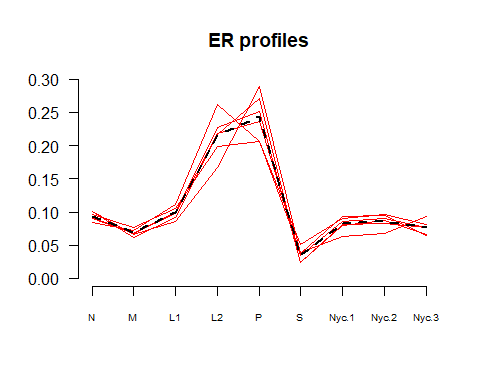
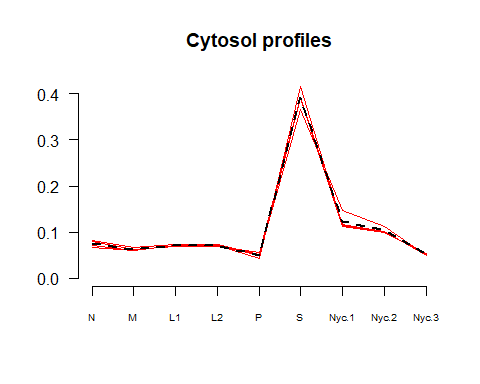
There are 39 genes in this reference set; two for each compartment are shown here.

To obtain profiles for the reference genes, use the function cpaSetup:

matLocRuse <- cpaSetup(geneProfileSummary=geneProfileSummaryTMTms2, refLocProteins=refLocProteinsJadot, n.channels=9)  
matLocRuse  
#> N M L1 L2 P S Nyc.1 Nyc.2  
#> Cytosol 0.0748 0.0626 0.0721 0.0712 0.0506 0.3907 0.1226 0.1035  
#> ER 0.0940 0.0687 0.0986 0.2150 0.2435 0.0350 0.0821 0.0865  
#> Golgi 0.0874 0.0588 0.0586 0.0922 0.3049 0.0297 0.1708 0.1632  
#> Lysosome 0.0426 0.0500 0.1017 0.0755 0.0443 0.0457 0.1505 0.4262  
#> Mitochondria 0.1297 0.2850 0.1542 0.0744 0.0449 0.0470 0.0779 0.0686  
#> Nucleus 0.7354 0.0312 0.0154 0.0348 0.0375 0.0407 0.0378 0.0395  
#> Peroxisomes 0.0529 0.0700 0.2784 0.1437 0.0403 0.0521 0.0670 0.0836  
#> Plasma membrane 0.1165 0.0626 0.0812 0.1189 0.1120 0.0354 0.2430 0.1794  
#> Nyc.3  
#> Cytosol 0.0517  
#> ER 0.0766  
#> Golgi 0.0343  
#> Lysosome 0.0636  
#> Mitochondria 0.1183  
#> Nucleus 0.0276  
#> Peroxisomes 0.2119  
#> Plasma membrane 0.0510

To view them, use referenceProfilePlot:

#par(mfrow=c(4,3))  
referenceProfilePlot(refLocProteins=refLocProteinsJadot, geneProfileSummary=geneProfileSummaryTMTms2,  
 matLocR=matLocRuse, n.channels=9)

 Now we can run the CPA routine; this may take several minutes to complete. The last column is a convergence indicator, with 1 indicating successful convergence. We remove this last column to get a matrix of gene names, and columns indicating the estimated proportional assignments of each gene among the eight subcellular locations.

assignPropsOutTemp <- proLocAll(geneProfileSummary=geneProfileSummaryTMTms2, matLocR=matLocRuse,  
 n.channels=9)  
#> [1] "500 genes"  
#> [1] "1000 genes"  
#> [1] "1500 genes"  
#> [1] "2000 genes"  
#> [1] "2500 genes"  
#> [1] "3000 genes"  
#> [1] "3500 genes"  
#> [1] "4000 genes"  
#> [1] "4500 genes"  
#> [1] "5000 genes"  
#> [1] "5500 genes"  
#> [1] "6000 genes"  
#> [1] "6500 genes"  
#> [1] "7000 genes"  
#> [1] "7500 genes"  
ncol.a <- ncol(assignPropsOutTemp) # last column is a convergence indicator, which should be 1  
assignPropsUse <- data.frame(assignPropsOutTemp[,-ncol.a] ) # drop this last column  
head(assignPropsUse)  
#> geneName Cytosol ER Golgi Lysosome Mitochondria Nucleus  
#> 1 2900026A02RIK 0.24331 0.000 0.1583 0.0705 0.10715 0.0203  
#> 2 A1CF 0.15226 0.559 0.1906 0.0000 0.00000 0.0986  
#> 3 A930018M24RIK 0.00374 0.000 0.0000 0.9047 0.00000 0.0141  
#> 4 AAAS 0.00000 0.321 0.2587 0.0000 0.00000 0.4205  
#> 5 AABR07001519.1 0.89996 0.000 0.0000 0.1000 0.00000 0.0000  
#> 6 AABR07002683.1 0.11047 0.000 0.0559 0.0000 0.00556 0.0000  
#> Peroxisomes Plasma.membrane Nspectra Npeptides  
#> 1 0.1312 0.269 4 3  
#> 2 0.0000 0.000 25 16  
#> 3 0.0775 0.000 1 1  
#> 4 0.0000 0.000 3 3  
#> 5 0.0000 0.000 1 1  
#> 6 0.1529 0.675 2 2

We can look at the profile of, for example, the gene “AADAC” by first finding the gene number in the dataset:

protIndex("AADAC")  
#> Gene index number Gene name  
#> 1 93 AADAC  
#protPlotfun(protPlot=93, assignProbsOut=assignProbsOut)

This function also accepts partial matching of the first few letters of a gene. For example, we can find the indices of the genes starting with “AAD”"

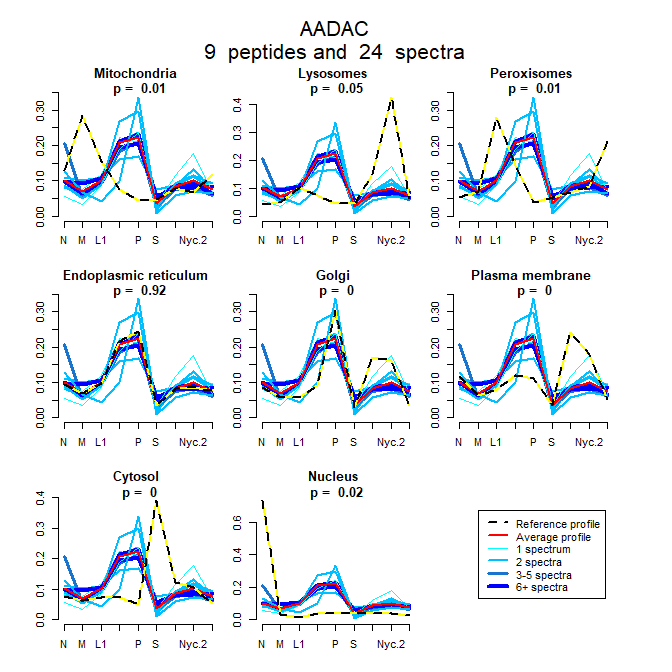
protIndex("AAD")  
#> Gene index number Gene name  
#> 1 93 AADAC  
#> 2 94 AADAT

Next, load the original data, with spectra and peptides, and select the columns for gene name, peptide name, and the nine fractions ranging from N to Nyc:

finalListUse <- subset(tmtMS2orig, select={c(gene:peptide, N:Nyc.3)})

Now we plot the results for gene number AADAC, which has index number 93:

protPlotfun(protPlot=93, geneProfileSummary=geneProfileSummaryTMTms2, finalList=finalListUse, n.fractions=9,  
 Nspectra=T, matLocR=matLocRuse, assignPropsMat=assignPropsUse)



The horizontal axis represents the nine fractions, which are N, M, L1, L2, P, S, Nyc.1, Nyc.2, and Nyc.3. In each of the eight plots, the red line is the average profile of the peptides, which are represented by the blue lines; thicker blue lines indicate larger numbers of spectra for a particular peptide. The dashed yellow-black lines show the expected profile for a protein entirely resident in the respective subcellular location. In this set of plots, we see that the CPA procedure assigns a 92% residence proportion to endoplasmic reticulum. Visually, we see that the observed red profile closely matches the expected yellow-black profile for endoplasmic reticulum.

## Nearest genes in a data set

We may find the genes with profiles nearest to a given gene using the function “nearestGenes”. Distance is computed as the Euclidean distance between profiles. To use the function, we first create a distance matrix for the genes in a list of mean profiles, such as “geneProfileSummaryTMTms2”. In this dataset, the profiles are in columns 2 through 10.

distUse <- dist(geneProfileSummaryTMTms2[,2:10], method="euclidean")

Then select the gene names:

genesUse <- geneProfileSummaryTMTms2[,1]

Finally, provide a gene name. Here, for the gene “AADAC”, we find the 10 nearest genes.

protIndex("AAD")  
#> Gene index number Gene name  
#> 1 93 AADAC  
#> 2 94 AADAT  
nearestGenes("AADAC", n.nearest=10, distGenes=distUse, geneNames=genesUse)  
#> geneName euclidean distance  
#> 93 AADAC 0.0000  
#> 2921 H6PD 0.0193  
#> 740 ATP13A1 0.0211  
#> 1282 CISD2 0.0215  
#> 5317 PRKCSH 0.0222  
#> 7504 UGGT1 0.0222  
#> 2242 ERMP1 0.0239  
#> 4095 MMGT1 0.0240  
#> 2189 ENTPD5 0.0244  
#> 2035 EDEM1 0.0247

## References

Christoforou A, Mulvey, CM, Breckels LM, Geladaki A, Hurrell T, Hayward PC, Naake T, Gatto L, Viner R, Arias AM, and Lilley KS (2016), “A draft map of the mouse pluripotent stem cell spatial proteome” Nature Communications 7, 8992. DOI: 10.1038/ncomms9992

Jadot M, Boonen M, Thirion J, Wang N, Xing J, Zhao C, Tannous A, Qian M, Zheng H, Everett JK, Moore DF, Sleat DE, Lobel P (2016) Accounting for protein subcellular localization: a compartmental map of the rat liver proteome. Molecular and Cellular Proteomics 16, 194-212. <doi:10.1074/mcp.M116.064527> PMCID: PMC5294208

Tannous A, Boonen M, Zheng H, Zhao C, Germain C, Moore D, Sleat D, Jadot M, Lobel P. Comparative Analysis of Quantitative Mass Spectrometric Methods for Subcellular Proteomics. Journal of Proteome Research 2020, in press