

plasma: Theoretical Considerations

Kevin R. Coombes

Contents

Background	1
Approach	2
Theory	2
References	3

Background

Recent years have seen the development of numerous algorithms and computational packages for the analysis of multi-omics data sets. At this point, one can find several review articles summarizing progress in the field (Subramanian *et al.*, 2020; Graw *et al.*, 2021; Heo *et al.*, 2021; Picard *et al.*, 2021; Reel *et al.*, 2021; Vlachavas *et al.*, 2021; Adossa *et al.*, 2021). As with other applications of machine learning, the kinds of problems addressed by these algorithms are divided into two categories: unsupervised (e.g., clustering or class discovery) or supervised (including class comparison and class prediction) (Simon and Dobbin, 2003). Advances in the area of unsupervised learning have been broader and deeper than advances on supervised learning.

One of the most effective unsupervised methods is Multi-Omic Factor Analysis (MOFA) (Argelaguet *et al.*, 2018, 2020). A key property of MOFA is that it does not require all omics assays to have been performed on all samples under study. In particular, it can effectively discover class structure across omics data sets even when data for many patients have only been acquired on a subset of the omics technologies. As of this writing, we do not know of any supervised multi-omics method that can effectively learn to predict outcomes when samples have only been assayed on a subset of the omics data sets.

MOFA starts with a standard method – Latent Factor Analysis – that is known to work well on a single omics data set. It then fits a coherent model that identifies latent factors that are common to, and able to explain the data well in, all the omics data sets under study. Our investigation (unpublished) of the factors found by MOFA suggests that, at least in some cases, it is approximately equivalent to a two-step process:

1. Use principal components analysis to identify initial latent factors in each individual omics data set.
2. For each pair of omics data sets, use overlapping samples to train and extend models of each factor to the union of assayed samples.

That re-interpretation of MOFA suggests that an analogous procedure might work for supervised analyses as well. In this article, we describe a two-step algorithm, which we call “*plasma*”, to find models that can predict time-to-event outcomes on samples from multi-omics data sets even in the presence of incomplete data. We use partial least squares (PLS) for both steps, using Cox regression (Bertrand and Maumy-Bertrand, 2021) to learn the single omics models and linear regression (Mishra and Liland, 2022) to learn how to extend models from one omics data set to another. To illustrate the method, we use a subset of the esophageal cancer (ESCA) data set from The Cancer Genome Atlas (TCGA).

Approach

The `plasma` algorithm is based on Partial Least Squares (PLS), which has been shown to be an effective method for finding components that can predict clinically interesting outcomes (Bastien *et al.*, 2015). The workflow of the `plasma` algorithm is illustrated in **Figure 1** in the case of three omics data sets. First, for each of the omics data sets, we apply the PLS Cox regression algorithm (`plsRcox` Version 1.7.6 (Bertrand and Maumy-Bertrand, 2021)) to the time-to-event outcome data to learn three separate predictive models (indicated in red, green, and blue, respectively). Each of these models may be incomplete, since they are not defined for patients who have not been assayed (shown in white) using that particular omics technology. Second, for each pair of omics data sets, we apply the PLS linear regression algorithm (`pls` Version 2.8.1 (Mishra and Liland, 2022)) to learn how to predict the coefficients of the Cox regression components from one data set using features from the other data set. This step extends (shown in pastel red, green, and blue, resp.) each of the original models, in different ways, from the intersection of samples assayed on both data sets to their union. Third, we average all of the different extended models (ignoring missing data) to get a single coherent model of component coefficients across all omics data sets. Assuming that this process has been applied to learn the model from a training data set, we can evaluate the final Cox regression model on both the training set and a test set of patient samples.

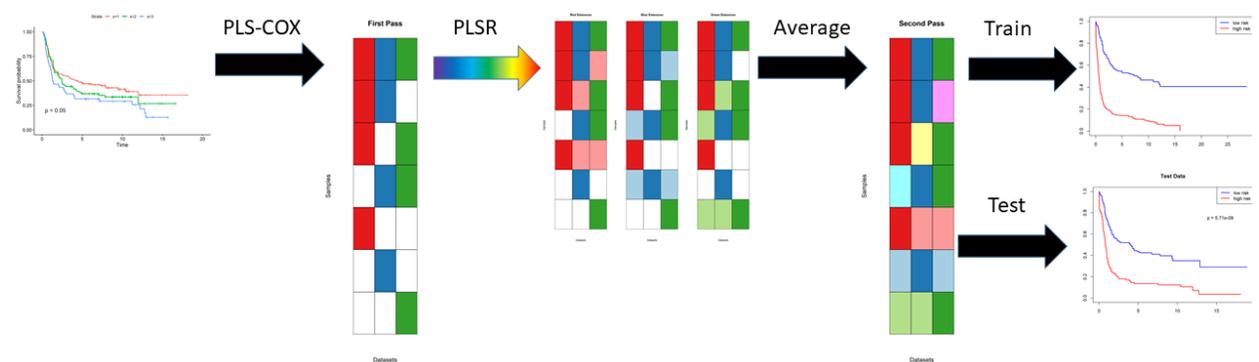


Figure 1: Workflow schematic for `plasma` algorithm with three omics data sets. See main text for an explanation.

All computations were performed in R version 4.2.2 (2022-10-31 ucrt) of the R Statistical Software Environment (R Core Team, 2022). Cox proportional hazards models for survival analysis were fit using version 3.4.0 of the `survival` R package. We used additional exploratory graphical tools from version 1.3.1 of the `beanplot` R package (Kampstra, 2008) and version 1.5.1 of the `Polychrome` R package (Coombes *et al.*, 2019). Gene enrichment (pathway or annotation) analysis was performed using ToppGene (<https://toppgene.cchmc.org/>) (Chen *et al.*, 2009).

Theory

The given data for a PLS model consists of a matrix, Y , of patient by outcomes and a matrix X of patients by (potential predictive) features from an omics data set. The underlying model indicates that we want to find approximate decompositions

$$X = TP^T + E, \quad Y = UQ^T + F,$$

where E and F are error terms, P and Q are orthogonal loading matrices, and T (the X-scores) and U (the Y-scores) are the projections of the given data. In the multi-omic case, we assume that we have $I = 1, \dots, N$ different omics data sets containing potential predictors.

The first pass in the plasma algorithm is to solve this problem separately for each data set, obtaining solutions of the form

$$X_I = T_I P_I^T + E_I, \quad Y = U_I Q_I^T + F_I,$$

To keep track of dimensions, we let p be the number of patients, m the number of outcomes, f_I the number of features in the I^{th} data set, and ℓ_I the number of components desired from the I^{th} data set. So,

- Y is $p \times m$;
- X_I is $n \times f_I$;
- T_I and U_I are $n \times \ell_I$;
- P_I is $f_I \times \ell_I$; and
- Q_I is $m \times \ell_I$.

Now suppose we concatenate all of the omics data matrices (i.e., do the mathematical equivalent of `cbind` in R) as

$$X = [X_1 | X_2 | \dots | X_N].$$

Now X is a $n \times f$ matrix, where $f = \sum_I f_i$. Of course, we still have the same outcome matrix, Y . We can decompose X pretty easily: We have

$$X = [T^1 P_1^T | \dots | T_N P_N^T] = [T_1 | \dots | T_N] * \text{diag}(P_1, \dots, P_N)^T = T P^T,$$

where T is $n \times \ell$ and the block diagonal matrix P is $f \times \ell$ for $\ell = \sum_I \ell_I$.

Next, we have to factor Y as products of an $n \times \ell$ matrix U and the transpose of an $m \times \ell$ matrix Q . If we take

$$U = [U_1 | \dots | U_N], \quad \text{and} \quad Q = [Q_1 | \dots | Q_N],$$

then they have the correct dimensions. But

$$U Q^T = \sum_I U_I Q_I = N Y,$$

so we have to divide by N somewhere.

Now the second pass in the plasma algorithm is to fit the X-scores, T_I , from each data set using each of the other data sets X_J . This comes down to solving equations like

$$X_J = W_{IJ} S_{IJ}^T + \text{error}, \quad T_I = V_{IJ} R_{IJ}^T + \text{error}.$$

This ought to correspond to computing the off-diagonal blocks in the decomposition above defining P .

References

- Adossa, N. *et al.* (2021) Computational strategies for single-cell multi-omics integration. *Comput Struct Biotechnol J*, **19**, 2588–2596.
- Argelaguet, R. *et al.* (2020) MOFA+: A statistical framework for comprehensive integration of multi-modal single-cell data. *Genome Biol*, **21**, 111.
- Argelaguet, R. *et al.* (2018) Multi-omics factor analysis—a framework for unsupervised integration of multi-omics data sets. *Mol Syst Biol*, **14**, e8124.
- Bastien, P. *et al.* (2015) Deviance residuals-based sparse PLS and sparse kernel PLS regression for censored data. *Bioinformatics*, **31**, 397–404.
- Bertrand, F. and Maumy-Bertrand, M. (2021) Fitting and cross-validating cox models to censored big data with missing values using extensions of partial least squares regression models. *Front Big Data*, **4**, 684794.
- Chen, J. *et al.* (2009) ToppGene suite for gene list enrichment analysis and candidate gene prioritization. *Nucleic Acids Res*, **37**, W305–11.
- Coombes, K.R. *et al.* (2019) Polychrome: Creating and assessing qualitative palettes with many colors. *Journal of Statistical Software*, **90**, 1–23.

- Graw,S. *et al.* (2021) Multi-omics data integration considerations and study design for biological systems and disease. *Mol Omics*, **17**, 170–185.
- Heo,Y.J. *et al.* (2021) Integrative multi-omics approaches in cancer research: From biological networks to clinical subtypes. *Mol Cells*, **44**, 433–443.
- Kampstra,P. (2008) Beanplot: A boxplot alternative for visual comparison of distributions. *Journal of Statistical Software*, **28**, 1–9.
- Mishra,P. and Liland,K.H. (2022) Swiss knife partial least squares (SKPLS): One tool for modelling single block, multiblock, multiway, multiway multiblock including multi-responses and meta information under the ROSA framework. *Anal Chim Acta*, **1206**, 339786.
- Picard,M. *et al.* (2021) Integration strategies of multi-omics data for machine learning analysis. *Comput Struct Biotechnol J*, **19**, 3735–3746.
- R Core Team (2022) R: A language and environment for statistical computing R Foundation for Statistical Computing, Vienna, Austria.
- Reel,P.S. *et al.* (2021) Using machine learning approaches for multi-omics data analysis: A review. *Biotechnol Adv*, **49**, 107739.
- Simon,R.M. and Dobbin,K. (2003) Experimental design of DNA microarray experiments. *Biotechniques*, **Suppl**, 16–21.
- Subramanian,I. *et al.* (2020) Multi-omics data integration, interpretation, and its application. *Bioinform Biol Insights*, **14**, 1177932219899051.
- Vlachavas,E.I. *et al.* (2021) A detailed catalogue of multi-omics methodologies for identification of putative biomarkers and causal molecular networks in translational cancer research. *Int J Mol Sci*, **22**.