Tensor decompositions of ABBA-BABA scores

April 16, 2019

Contents

1 Note on viewing this Readme

This Readme may not render correctly on github. I recommend viewing the PDF version, or viewing it in a text editor.

2 Model

This repository implements a method, that takes Z-scores from qpDstat, represents them as a 4-dimensional tensor $\mathcal{Z}$, and then does a tensor rank decomposition to try and decompose $\mathcal{Z}$ into a few admixture components.

In particular, let $Z_{ijkl} = \frac{BABA_{ijkl} - ABBA_{ijkl}}{\sigma_{ijkl}}$ be the Z-score for populations i, j, k, l. Then the Z-scores form a 4-tensor indexed by i, j, k, l.

We use a slightly modified version of this tensor, which we denote by $\mathcal{Z}$. First, note that every quartet has 3 unrooted topologies, and so 3 tests for

$BABA - ABBA \neq 0$. We only use the Z-score of the best supported topology, i.e. permutations with BBAA > ABBA,BABA, and set entries for the other topologies to zero. Furthermore, note that within a topology, the order of the populations determines the sign of Z; for example, $Z_{ijkl} = -Z_{jikl}$. We restrict ourselves to only the positive Z-scores, i.e. those permutations with BABA > ABBA. Thus, the entries of our tensor $\mathcal{Z}$ are defined as follows:

$$(\mathcal{Z})_{ijkl} = \begin{cases} Z_{ijkl} & \text{if } BBAA_i \geq BABA_i \geq ABBA_i \\ 0 & \text{else.} \end{cases}$$

Note that $\mathcal{Z}$ is symmetric under certain permutations of the axes. For example, (i, j, k, l) and (j, i, l, k) have the same BABA-ABBA scores, i.e. transposing the first 2 axes and the last 2 axes leaves the tensor unchanged. Let $\mathcal{S}_\sigma$ be the operator that permutes the axes of $\mathcal{Z}$; then $\mathcal{Z}$ is invariant under the tensor transposes $\mathcal{S}_{(12)(34)}$, $\mathcal{S}_{(13)(24)}$, and $\mathcal{S}_{(14)(23)}$ (cf. permutation cycle notation).

To decompose $\mathcal{Z}$, we fit the model

$$\begin{aligned} \mathcal{Z} \approx (\mathcal{I} + \mathcal{S}_{(13)(24)} + \mathcal{S}_{(12)(34)} + \mathcal{S}_{(14)(23)}) \\ \left(\sum_{k=1}^{K} \mathbf{a}^{(k)} \otimes \mathbf{b}^{(k)} \otimes \mathbf{c}^{(k)} \otimes \mathbf{d}^{(k)} \right) \end{aligned} \tag{1}$$

where $\mathbf{a}^{(k)}$ is a non-negative vector, whose non-negative entries we interpret as a cluster of populations all having the same "position" within admixture event k (and similarly for $\mathbf{b}^{(k)}, \mathbf{c}^{(k)}, \mathbf{d}^{(k)}$.)

3 Optimization

We minimize the following loss function:

$$\|\mathcal{Z} - \hat{\mathcal{Z}}\|_2^2 + \lambda \sum_{k=1}^{K} \left(\|\mathbf{a}^{(k)}\|_1 + \|\mathbf{b}^{(k)}\|_1 + \|\mathbf{c}^{(k)}\|_1 + \|\mathbf{d}^{(k)}\|_1 \right) \tag{2}$$

where $\hat{\mathcal{Z}}$ is the fitted tensor according to (1), $\|\cdot\|_2$ is the Frobenius norm, and $\|\cdot\|_1$ is the L_1 norm. λ is a tuning parameter used to encourage sparsity.

To fit the model, we initialize $\mathbf{a}^{(k)}, \mathbf{b}^{(k)}, \mathbf{c}^{(k)}, \mathbf{d}^{(k)}$ to random values, then minimize the loss function (2) with gradient descent.

To explore how the tuning parameter λ affects the result, we fit a whole path of λ values. We start by fitting (2) with $\lambda := \lambda_{\max}$. Then we decrease the tuning parameter by ϵ, i.e. $\lambda := \lambda - \epsilon$, and refit (2) starting from the current position. This yields a continuous path of decompositions along a grid of λ values, which can be visualized by dragging a slider in an R shiny widget.

4 Code

Here is a rough guide to the code.

4.1 `example/`

This directory contains a python script for running BABA, and an R script to visualize the results, either in a shiny widget or as a PNG.

WARNING: These scripts were copied from another project and slightly edited to be standalone scripts. I have not rerun/tested them.

4.2 `baba/`

This directory is a Python package that implements the tensor decomposition. It has the following modules.

4.2.1 `quartet_stats.py`

This module contains the `quartet_stats` class for representing output from qpDstat. The `from_qpDstat()` method reads in output from qpDstat. The 4-tensor of Z-scores is stored in the `z_score` attribute. The 4-tensor of BABA counts are stored in the `baba` attribute. Tensors for the ABBA and BBAA counts are obtained by transposing the axes of the BABA tensor.

4.2.2 `quartet_decomposition.py`

This module represents and fits the tensor decomposition model. The `quartet_decomposition` class keeps a representation of the current decomposition in the `components` and `weights` attributes. The `fit_decomposition()` and `optimize()` method wrap around `scipy` and `autograd` to fit the model.

4.2.3 `symmetries.py`

Utility functions for dealing with tensor symmetries.