# nnTensor: An R package for non-negative matrix/tensor decomposition

Koki Tsuyuzaki [*,1,2] **and Itoshi Nikaido** [1,3,4]

1 Laboratory for Bioinformatics Research, RIKEN Center for Biosystems Dynamics Research, Japan **2** Japan Science and Technology Agency, PRESTO, Japan **3** Science Innovation (Bioinformatics), Degree Programs in Systems and Information Engineering, Graduate School of Science and Technology), University of Tsukuba, Japan **4** Department of Functional Genome Informatics, Division of Biological Data Science, Medical Research Institute, Tokyo Medical and Dental University, Japan

Summary

Non-negative matrix factorization (NMF) is a widely used algorithm to decompose non-negative matrix data into factor matrices. Due to the interpretability of its non-negativity and the convenience of using decomposition results as clustering, there are many applications of NMF in image processing, audio processing, and bioinformatics (Cichocki et al., 2009).

NMF has been applied to matrix data but there is a growing demand to apply NMF to more heterogeneous non-negative data such as multiple matrices and tensors (high-dimensional arrays), which are higher-order data structures than matrices (Cichocki et al., 2009). To meet these requirements, we originally developed nnTensor, which is an R/CRAN package to perform some non-negative matrix/tensor decomposition algorithms (https://cran.r-project.org/web/packages/nnTensor/index.html).

Statement of need

In many cases, the latest tensor decomposition algorithms are implemented in MATLAB in the original papers, which means that those are not freely applicable to users' data immediately. To fill this gap, we originally implemented some non-negative matrix/tensor decomposition algorithms in R language, which is one of the popular open-source programming languages.

nnTensor provides the matrix/decomposition decomposition functions as follows:

- `NMF`: Non-negative matrix factorization for non-negative matrix (Cichocki et al., 2009)
- `NMTF`: Non-negative matrix tri-factorization for non-negative matrix (Čopar et al., 2019)
- `siNMF`: Simultaneous non-negative matrix factorization for non-negative matrix (Badea, 2009)
- `jNMF`: Joint non-negative matrices factorization for non-negative matrix (Yang & Michailidis, 2016)
- `NTF`: Non-negative tensor factorization for non-negative tensor (Cichocki et al., 2009)
- `NTD`: Non-negative Tucker decomposition for non-negative tensor (Cichocki et al., 2009)

All the objective functions are the divergence between the original matrix/tensor and the reconstructed matrix/tensor from the decomposition. Users can easily try some divergences including Frobenius norm, Kullback–Leibler (KL) divergence, and Itakura-Saito (IS) divergence by specifying `algorithm` argument. Some utility functions are available such as `recTensor` to reconstruct data matrix/tensor, `plotTensor2D`/`plotTensor3D` to visualize the structure of data

---
[*]first author

matter/tensor, and toyModel to provide some toy datasets so that users can immediately check the operation of these functions and learn how to use them. Besides, mask matrices/tensors can be specified in the matrix/tensor decomposition functions above to support rank estimation with cross-validation.

Example

The non-negative tensor decomposition and plots in Figure Figure 1 can be easily reproduced on any machine where R is pre-installed by using the following commands in R:

```r
# Install package required (one per computer)
install.packages("nnTensor")

# Load required package (once per R instance)
library("nnTensor")

# Load Toy data
data <- toyModel("Tucker")

# Perform NTD
set.seed(1234)
out.ntd <- NTD(data, rank=c(4,5,6), algorithm="KL", num.iter=30)

# Reconstruction of data tensor
rec.data <- recTensor(S=out.ntd$S, A=out.ntd$A)

# Visualization
plotTensor3D(data)
plotTensor3D(rec.data)
plot(out.ntd$RecError, type="b", xlab="Iteration", ylab="RecError")
```
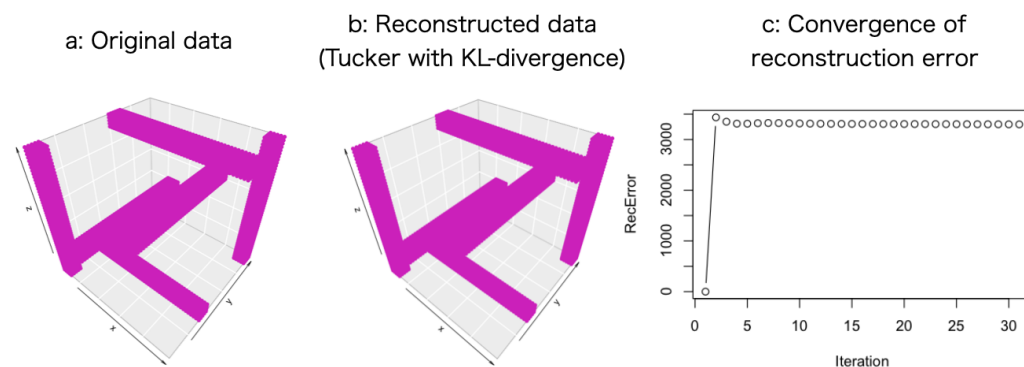

Figure 1: Toydata decomposed by NTD with KL-divergence

Related work

There are some versatile tools to perform tensor decomposition such as Tensorly (Python, Kossaifi et al. (2019)), TensorToolbox (MATLAB, Kolda & Bader (2009)), and TensorLab (MATLAB, Vervliet et al. (2016)). In the R language, rTensor (Li et al., 2018) is a widely used package to perform tensor decomposition but none of the non-negative type algorithms is implemented and nnTensor is the first package to meet the demand.

References

Badea, L. (2009). Extracting Gene Expression Profiles Common to Colon and Pancreatic Adenocarcinoma using Simultaneous nonnegative matrix factorization. *Pacific Symposium on Biocomputing 2008*. https://doi.org/10.1142/9789812776136_0027

Cichocki, A., Zdunek, F., Phan, A. H., & Amari, S. (2009). Wiley. https://onlinelibrary.wiley.com/doi/book/10.1002/9780470747278

Čopar, A., Zupan, B., & Zitnik, M. (2019). Fast Optimization of Non-Negative Matrix Tri-Factorization. *PLOS ONE*. https://doi.org/10.1371/journal.pone.0217994

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review*. https://doi.org/10.1137/07070111X

Kossaifi, J., Panagakis, Y., Anandkumar, A., & Pantic, M. (2019). TensorLy: Tensor Learning in Python. *Journal of Machine Learning Research*. https://jmlr.org/papers/v20/18-277.html

Li, J., Bien, J., & Wells, M. T. (2018). rTensor: An R Package for Multidimensional Array (Tensor) Unfolding, Multiplication, and Decomposition. *Journal of Statistical Software*. https://doi.org/10.18637/jss.v087.i10

Vervliet, N., Debals, O., & De Lathauwer, L. (2016). Tensorlab 3.0 - Numerical optimization strategies for large-scale constrained and coupled matrix/tensor factorization. *2016 50th Asilomar Conference on Signals, Systems and Computers*. https://doi.org/10.1109/acssc.2016.7869679

Yang, Z., & Michailidis, G. (2016). A non-negative matrix factorization method for detecting modules in heterogeneous omics multi-modal data. *Bioinformatics*. https://doi.org/10.1093/bioinformatics/btv544