Bioinformatics, 2023, **39(7)**, btad420
https://doi.org/10.1093/bioinformatics/btad420
Advance access publication 8 July 2023
Applications Note

OXFORD

Gene expression

LSMMD-MA: scaling multimodal data integration for single-cell genomics data analysis

Laetitia Meng-Papaxanthos [1,*], Ran Zhang [2,3], Gang Li [2,3], Marco Cuturi [4,5,†], William Stafford Noble [2,6], Jean-Philippe Vert [4,7,*]

[1]Google Research, Brain Team, Google, Brandschenkestrasse 110, Zurich 8002, Switzerland
[2]Department of Genome Sciences, University of Washington, 3720 15th Ave NE, Seattle, WA 98195, United States
[3]eScience Institute, University of Washington, 3910 15th Ave NE, Seattle, WA 98195, United States
[4]Google Research, Brain Team, Google, 8 Rue de Londres, Paris 75009, France
[5]Apple ML Research, Apple, 7 Av. d'Iéna, Paris 75116, France
[6]Paul G. Allen School of Computer Science and Engineering, University of Washington, 185 E Stevens Way NE, Seattle, WA 98195, United States
[7]Owkin, Inc., 14/16 Bd Poissonnière, Paris 75009, France

*Corresponding authors. Google Research, Brain Team, Google, Zurich 8002, Switzerland. E-mail: lpapaxanthos@google.com (L.M.-P.); Google Research, Brain Team, Google, Paris 75009, France. E-mail: jean-philippe.vert@owkin.com (J.-P.V.)

[†]Work done while at Google.

Associate Editor: Anthony Mathelier

Abstract

Motivation: Modality matching in single-cell omics data analysis—i.e. matching cells across datasets collected using different types of genomic assays—has become an important problem, because unifying perspectives across different technologies holds the promise of yielding biological and clinical discoveries. However, single-cell dataset sizes can now reach hundreds of thousands to millions of cells, which remain out of reach for most multimodal computational methods.

Results: We propose `LSMMD-MA`, a large-scale Python implementation of the `MMD-MA` method for multimodal data integration. In `LSMMD-MA`, we reformulate the `MMD-MA` optimization problem using linear algebra and solve it with KeOps, a CUDA framework for symbolic matrix computation in Python. We show that `LSMMD-MA` scales to a million cells in each modality, two orders of magnitude greater than existing implementations.

Availability and implementation: `LSMMD-MA` is freely available at https://github.com/google-research/large_scale_mmdma and archived at https://doi.org/10.5281/zenodo.8076311.

1 Introduction

Modality matching in single-cell genomics data analysis can enhance our understanding of the relationships between cellular modalities and help us resolve cell states. In this problem, single-cell measurements collected using two or more different types of assays are projected into a shared space or are otherwise matched across modalities, with the goal of achieving insights into the joint multimodal dataset. Most existing multimodal models rely on learning cell representations in each modality in a joint low-dimensional space (Welch *et al.* 2019, Cao *et al.* 2020, Jin *et al.* 2020, Stark *et al.* 2020, Gayoso *et al.* 2021, Hao *et al.* 2021, Raimundo *et al.* 2021, Cao and Gao 2022). `MMD-MA` (Liu *et al.* 2019, Singh *et al.* 2020) is one such method that has shown promising results on datasets containing several thousand cells in each modality. However, thanks to new single-cell technologies, the size of single-cell datasets has increased significantly in the past 2 years, now reaching several hundreds of thousands to millions of cells (Papatheodorou *et al.* 2020, Hao *et al.* 2021, Rozenblatt-Rosen *et al.* 2021). These datasets cannot be analysed by current implementations of `MMD-MA` due to memory issues.

More precisely, `MMD-MA` (Liu *et al.* 2019) is a multimodal approach that maps each cell in each modality to a shared, low-dimensional representation space. The linear mappings from the input spaces to the representation space are learned by minimizing an objective function composed of several terms: (i) a "matching" term based on the squared maximum mean discrepancy (MMD) with a Gaussian radial basis function (RBF) kernel to ensure that the different modalities overlap in the representation space, (ii) two "noncollapsing" penalties to prevent trivial solutions, and (iii) two "distortion" penalties to ensure that as much information from the input data as possible is captured in the shared representation. More details about the method are provided Supplementary Appendix Section A. However, current implementations of `MMD-MA` (Liu *et al.* 2019, Singh *et al.* 2020) scale quadratically as a function of the number of cells in memory and runtime, which is prohibitive for datasets with more than a few thousand samples (see Supplementary Table SA4).

To increase the scalibility of `MMD-MA`, we introduce `LSMMD-MA`, a reformulation and PyTorch implementation of

Received: March 24, 2022. **Revised:** June 25, 2023. **Editorial Decision:** June 26, 2023
© The Author(s) 2023. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

MMD-MA that overcomes the memory explosion issue. To achieve this, we (i) reformulate MMD-MA's optimization problem in the primal, which is beneficial when the number of cells is larger than the number of informative features and (ii) implement the MMD matching term with the CUDA-based KeOps library for symbolic matrices (Charlier *et al.* 2021), tailored to handle matrices that do not fit in RAM or GPU memory. The resulting algorithm scales only linearly in memory with the number of cells and can handle up to a million cells in each modality.

2 Materials and methods

2.1 Reformulating MMD-MA in the primal

Let $X \in \mathbb{R}^{n_x \times p_x}$ (respectively, $Y \in \mathbb{R}^{n_y \times p_y}$) be the data matrix of the first (respectively, second) modality, where n_x (respectively, n_y) is the number of cells in the first (respectively, second) modality and p_x (respectively, p_y) is the number of features in the first (respectively, second) modality. The goal of MMD-MA is to learn two mappings from the input spaces $\mathbb{R}^{n_x}$ and $\mathbb{R}^{n_y}$ to a shared representation space $\mathbb{R}^d$. We focus specifically on linear mappings, as in the original publications (Liu *et al.* 2019, Singh *et al.* 2020), where mappings are parameterized with dual variables $\alpha_x \in \mathbb{R}^{n_x \times d}$ and $\alpha_y \in \mathbb{R}^{n_y \times d}$ such that the embedding of the first (respectively, second) modality is $XX^\top \alpha_x$ (respectively, $YY^\top \alpha_y$). Instead, we equivalently parameterize the mappings by primal variables $W_x \in \mathbb{R}^{p_x \times d}$ and $W_y \in \mathbb{R}^{p_y \times d}$, such that the embedding of the first (respectively, second) modality is XW_x (respectively, YW_y). We can then rewrite the MMD-MA optimization problem in the primal:

$$\min_{W_x, W_y} L_{\text{primal}}(W_x, W_y) = \min_{W_x, W_y} [MMD(XW_x, YW_y)^2 + \lambda_1(\text{pen}(W_x) + \text{pen}(W_y)) + \lambda_2(\text{dis}(X, W_x) + \text{dis}(Y, W_y))], \quad (1)$$

where λ_1 and λ_2 are hyperparameters. Under the assumption that $n \gg p \gg d$ for each modality, efficiently implementing the primal loss (Equation 1) scales better than implementing the dual loss, as shown in Supplementary Table A2. The primal loss does not require the computation and storage of the linear kernel matrices $XX^\top$ and $YY^\top$, which are $O(n^2)$ in time and memory, and the penalty and distortion terms are not $O(n^2)$ in runtime anymore. However, computing the MMD term remains $O(n^2)$ in runtime and memory if we implement it naively. A description of MMD-MA in the dual and a comparison between the formulations of MMD-MA and LSMMD-MA are available in Supplementary Appendix Sections A and B.

2.2 Using KeOps

To overcome the $O(n^2)$ memory burden of computing the MMD term, we implement it using the CUDA-based Map-Reduce scheme of KeOps (Charlier *et al.* 2021). This allows us to compute the MMD term without instantiating the $n \times n$ Gaussian RBF kernel in memory, using symbolic matrix computation with $O(n)$ memory complexity, as detailed in Supplementary Appendix Section D. KeOps therefore optimizes (Equation 1) with a linear memory complexity and also improves runtime by a significant multiplicative factor when the number of samples is >1000.

2.3 Implementation

We make four algorithms available, including LS-MMDMA and three variants: primal formulation without KeOps, dual formulation with KeOps, and dual formulation without KeOps (an efficient implementation of the original algorithm). The code is implemented in PyTorch (Paszke *et al.* 2019) and can run on CPU or GPU. The package is open source with an Apache license, available at github.com/google-research/large_scale_mmdma. It is referenced on PyPI and can be installed with the command: `pip install lsmmdma`. Details about I/O, command line instructions and tutorials are given in the `Readme.md` and in the `examples` folder.

3 Results and conclusion

We tested the scalability of the implementation of LSMMD-MA (primal formulation with KeOps) against three comparison partners (primal formulation without KeOps, dual formulations with KeOps and dual formulation without KeOps) and against the two original implementations (Liu *et al.* 2019, Singh *et al.* 2020) which focus on the dual formulation in TensorFlow (Abadi *et al.* 2016) and PyTorch (Paszke *et al.* 2019), respectively. Additionally, Singh *et al.* (2020) proposes to use the linear time approximation of MMD for large numbers of samples (> 5000) (see Lemma 14 in Gretton *et al.* 2012).

We ran all algorithms on simulated datasets of different sizes where the latent space is shaped as a branch (see lsmmdma/data/data_pipeline.py in GitHub and Supplementary Appendix Section G for more details). All algorithms were run for 500 epochs and a low-dimensional representation of dimension $d = 10$. A V100 GPU (16GB) was used for the experiments. We observe that LSMMD-MA, using the primal formulation and KeOps, scales to one million cells in each modality, whereas the original implementations runs out of memory for >14 000 cells (Fig. 1). In the same figure, we can also notice that our dual implementations are faster than the original implementations irrespective of the number of samples. We also show that LSMMD-MA obtains good accuracies, as measured by the Fraction Of Samples Closer than The True Match (FOSCTTM) (Liu *et al.* 2019), on a selection of the simulated datasets (see Supplementary Appendix Section E). MMD-MA and LSMMD-MA have the same optimal objective values as LSMMD-MA is a reformulation of MMD-MA. Furthermore, we show that both algorithms obtain similar FOSCTTM performance on 12 synthetic datasets with varying numbers of features and samples (see Supplementary Appendix Section B).

As a proof of principle, we also ran LSMMD-MA on a real-world CITE-seq dataset containing 90 261 human bone marrow mononuclear cells, with 13 953 gene IDs for the gene expression modality and 134 proteins for the protein marker modality (Luecken *et al.* 2021). We obtain an FOSCTTM of 0.22 after 100 000 epochs (10.3 h) with LSMMD-MA, which would have been infeasible with previous versions of MMD-MA. More details about the preprocessing of the dataset and the hyperparameters are available in Supplementary Appendix Section G. We additionally compared LSMMD-MA with baselines such as Procrustes superimposition (with and without aligned data) (Gower 1975), LIGER (PyLiger) (Liu *et al.* 2020, Lu and Welch 2022) and Harmonic alignment (Stanley *et al.* 2020) and show that LSMMD-MA is competitive (see Supplementary Appendix Section H for detail).

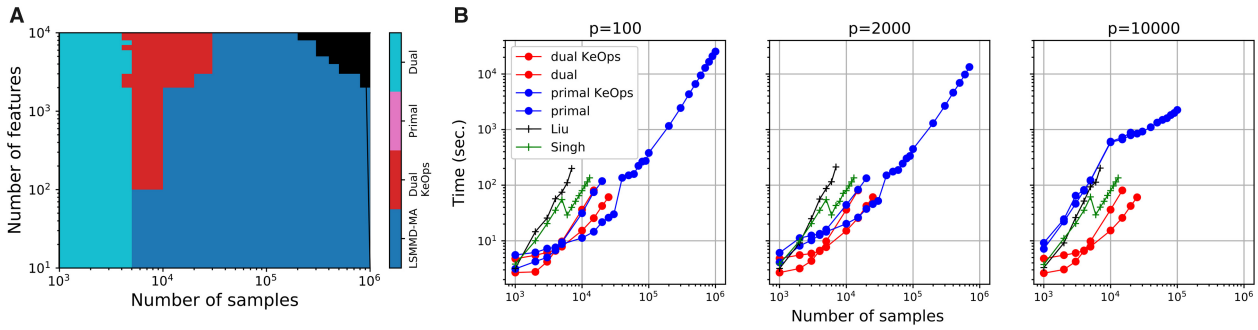

Figure 1. (A) Fastest MMD-MA variant as a function of the number of samples and the number of features. The black region in the top right corner means that all variants ran out of memory. (B) Runtime as a function of number of cells for different implementations of `MMD-MA`, when the dimension p of the input data varies. The black and green dotted lines with cross markers correspond to the original implementations of `MMD-MA` as written by Liu *et al.* (2019) (black) and Singh *et al.* (2020) (green). The runtime for different values of p, from 100 to 10 000, is shown in Supplementary Appendix Fig. A1.

These results suggest that an optimized implementation, exploiting the primal formulation and taking advantage of the KeOps library, are key to building a multimodal model that scales to the size of current single-cell datasets.

Acknowledgements

The authors thank the anonymous reviewers for their valuable suggestions.

Supplementary data

Supplementary data are available at *Bioinformatics* online.

Conflict of interest

None declared.

Funding

This work was supported by the NIH award UM1 HG011531.

Data availability

The simulations used in this article are available in GitHub at https://github.com/google-research/large_scale_mmdma and the CITE-seq dataset is available from NCBI GEO under accession GSE194122.

References

Abadi M, Agarwal A, Barham P *et al.* TensorFlow: large-scale machine learning on heterogeneous systems. [Computer software]. *arXiv preprint arXiv:1603.04467*, 2016. https://www.tensorflow.org.

Cao K, Bai X, Hong Y *et al.* Unsupervised topological alignment for single-cell multi-omics integration. *Bioinformatics* 2020;**36**:i48–56. https://doi.org/10.1093/bioinformatics/btaa443.

Cao Z-J, Gao G. Multi-omics integration and regulatory inference for unpaired single-cell data with a graph-linked unified embedding framework. *Nat Biotechnol* 2022;**40**:1458–66.

Charlier B, Feydy J, Glaunès JA *et al.* Kernel operations on the GPU, with autodiff, without memory overflows. *J Mach Learn Res* 2021;**22**:1–6.

Gayoso A, Steier Z, Lopez R *et al.* Joint probabilistic modeling of single-cell multi-omic data with totalVI. *Nat Methods* 2021;**18**:272–82. https://doi.org/10.1038/s41592-020-01050-x.

Gower JC. Generalized procrustes analysis. *Psychometrika* 1975;**40**:33–51.

Gretton A, Borgwardt KM, Rasch MJ *et al.* A kernel two-sample test. *J Mach Learn Res* 2012;**13**:723–73.

Hao Y, Hao S, Andersen-Nissen E *et al.* Integrated analysis of multimodal single-cell data. *Cell* 2021;**184**:3573–87.e29.

Jin S, Zhang L, Nie Q. scAI: an unsupervised approach for the integrative analysis of parallel single-cell transcriptomic and epigenomic profiles. *Genome Biol* 2020;**21**:1–19. https://doi.org/10.1186/s13059-020-1932-8.

Liu J, Huang Y, Singh R *et al.* Jointly embedding multiple single-cell omics measurements. In: *19th International Workshop on Algorithms in Bioinformatics (WABI 2019)*, volume 143 of *Leibniz International Proceedings in Informatics (LIPIcs)* Niagara Falls, NY, USA, Vol. **10**, p.1–10, 2019. https://doi.org/10.4230/LIPIcs.WABI.2019.10.

Liu J, Gao C, Sodicoff J *et al.* Jointly defining cell types from multiple single-cell datasets using liger. *Nat Protoc* 2020;**15**:3632–62.

Lu L, Welch JD. Pyliger: scalable single-cell multi-omic data integration in python. *Bioinformatics* 2022;**38**:2946–8.

Luecken MD, Burkhardt DB, Cannoodt R *et al.* A sandbox for prediction and integration of DNA, RNA, and proteins in single cells. In: J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, Vol. **1**. Curran, 2021.

Papatheodorou I, Moreno P, Manning J *et al.* Expression atlas update: from tissues to single cells. *Nucleic Acids Res* 2020;**48**:D77–83.

Paszke A, Gross S, Massa F *et al.* Pytorch: an imperative style, high-performance deep learning library. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems, Vancouver, Canada* Vol. **32**. Curran Associates, Inc., 2019, 8024–35.

Raimundo F, Meng-Papaxanthos L, Vallot C *et al.* Machine learning for single-cell genomics data analysis. *Curr Opin Syst Biol* 2021;**26**:64–71.

Rozenblatt-Rosen O, Shin JW, Rood JE *et al.*; Human Cell Atlas Standards and Technology Working Group. Building a high-quality human cell atlas. *Nat Biotechnol* 2021;**39**:149–53.

Singh R, Demetci P, Bonora G *et al.* Unsupervised manifold alignment for single-cell multi-omics data. In: *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, p.1–10, 2020. https://doi.org/10.1101/2020.06.13.149195.

Stanley JS III, Gigante S, Wolf G *et al.* Harmonic alignment. In: *Proceedings of the 2020 SIAM International Conference on Data Mining*, p.316–24. SIAM, 2020.

Stark SG, Ficek J, Locatello F *et al.*; 12 Tumor Profiler Consortium. SCIM: universal single-cell matching with unpaired feature sets. *Bioinformatics* 2020;**36**:i919–27. https://doi.org/10.1093/bioinformatics/btaa843.

Welch JD, Kozareva V, Ferreira A *et al.* Single-cell multi-omic integration compares and contrasts features of brain cell identity. *Cell* 2019;**177**:1873–87.e17. https://doi.org/10.1016/j.cell.2019.05.006.