The Journal of Open Source Software

Jointly-HiC: A toolkit for integrative analysis of Hi-C genome architecture data

Thomas M. Reimonn [1], **Vedat Yilmaz**[1], **Guoyun Chen**[1], **and Nezar Abdennur** [1¶]

1 Department of Genomics and Computational Biology, UMass Chan Medical School, Worcester 01655, MA, USA ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Jointly-HiC is a Python-based toolkit designed for integrative embedding and analysis of Hi-C chromatin contact datasets from multiple biosamples. Genome architecture profiling methods such as Hi-C generate contact frequency data on the organization of DNA within the nucleus (**?**). Traditional analysis methods rely on rank-1 matrix factorization of a suitably pre-processed chromatin contact frequency map to produce the "A" and "B" compartment scores, and higher rank components reveal subtleties in "sub-compartment" nuclear organization (**?** al._2014_Spracklin_Abdennur_Imakaev_Chowdhury_Pradhan_Mirny_Dekker_2023). However, when integrating results from multiple experiments, these decompositions lie in separate vector spaces and are not directly comparable. To address this shortcoming, we developed Jointly-HiC to perform joint decomposition using incremental principal component analysis (PCA), non-negative matrix factorization (NMF), and singular value decomposition (SVD) algorithms. Jointly-HiC enables simultaneous embedding of Hi-C datasets into a shared vector space with fixed memory and linear compute scaling, allowing for direct comparative analyses of genome architecture across biological conditions and tissues. Further, Jointly-HiC supports building a JointDb with signals from gene expression, chromatin accessibility, histone modification and other profiling modalities for integrative analysis and comparative Hi-C studies.

Statement of Need

The organization of DNA in the nucleus forms a fundamental structure-function relationship and is linked to gene regulation, cellular identity, and developmental processes (**?** Berkum_Williams_Imakaev_Ragoczy_Telling_Amit_Lajoie_Sabo_Dorschner_et al._2009). Understanding the patterns of genome organization and how they differ across tissues, during organ development, and between health and disease is an important and open question in the field of structural genomics. Chromatin conformation capture (3C) methods such as Hi-C profile the 3D organization of the genome by cross-ligating DNA that is in close physical proximity (Belton et al., 2012). Ligation pairs are purified, sequenced and aligned to the genome, generating a two-dimensional genome-wide map of chromatin contact frequencies (Lafontaine et al., 2021). This map reveals genomic loci that strongly interact in 3D space or separate into non-interacting domains.

The traditional analysis framework for Hi-C data consists of suitability pre-processing the chromatin contact frequency matrix and applying eigenvector decomposition (Imakaev et al., 2012). The first eigenvector becomes the A/B compartment score and its sign defines the active and inactive genomic compartments, and higher order components can be clustered to identify genomic subcompartments. These subcompartments correlate with markers of

genetic activity: histone modifications, gene expression, regulatory element activity readouts, GC content and other epigenetic features (**?** al._2015). However, this framework involves separately processing input experiments, resulting in embeddings that are in distinct latent vector spaces. Making direct comparisons across samples embedded in distinct vector spaces presents a challenge because the meaning and direction of latent vectors differs across samples. Although normalization methods can harmonize embeddings to some extent, these methods inherently introduce biases and lose fine-scale biological information, compromising the accuracy and interpretability of comparative analyses. This fundamental limitation hinders integrative studies, particularly when scaling to large-scale analyses involving many samples such as developmental trajectories or tissue atlases.

To address this critical gap, we developed Jointly-HiC to uniformly pre-process and simultaneously embed multiple Hi-C datasets within a shared vector space. This framework avoids biases introduced by post-hoc harmonization and ensures embeddings from different datasets are directly comparable. Joint embedding generates higher order principle components that have the same biological interpretation across samples and are directly compatible. It leverages scikit-learn's incremental PCA, NMF and SVD algorithms and online data loading and unloading to use a fixed amount of memory for training and scale linearly with the number of input samples. Further, it includes a post-processing pipeline of genomic loci clustering and UMAP visualization and data storage in a JointDb. Other epigenetic signal tracks from gene expression, chromatin accessibility and histone modification profiling experiments can also be loaded into the JointDb to facilitate visualization and downstream analysis.

Software Description

Jointly-HiC is implemented as Python package and developed using widely adopted scientific libraries including: numpy, pandas, cooler, cooltools, bioframe, and hdf5 (**?** der Walt_Gommers_Virtanen_Cournapeau_Wieser_Taylor_Berg_Smith_et al._2020_Open2C_Abdennur_Abraham_Fudenberg_Flyamer_Galitsyna_Goloborodko_Imakaev_Oksuz_Venev_et al._Open2C_Abdennur_Fudenberg_Flyamer_Galitsyna_Goloborodko_Imakaev_Venev_2024). These libraries collectively provide robust, efficient, and scalable functionality that is used to build an analysis workflow for integrative genome architecture analysis.

The software workflow begins with data preprocessing. Genomic bins that fail to balance in any input sample are combined into a shared exclusion list of bad bins, which are filtered from input matrices. Intra-chromosomal contacts are masked by filling with random data from cross-chromosomal contacts. Outlier bins are clipped, and these matrices are balanced so that the sum of each row or column is one. The pre-processed results are saved to disk for future use.

The core embedding module of Jointly-HiC applies incremental dimensionality reduction algorithms provided by scikit-learn and supports PCA, NMF, or SVD (**?** al._2011). These algorithms operate in a minibatch fashion, and Jointly-HiC loads one sample into memory at a time to incrementally fit the embedding model then subsequently frees the memory before loading the next dataset. This incremental processing requires linear compute time to the number of samples and a fixed amount of memory. Following model training, Jointly-HiC performs a second pass over the datasets to compute final embeddings. These results are stored in both CSV and Parquet formats.

Post-processing steps begin with rescaling of embeddings using the Frobenius norm for each sample. Embeddings are clustered and visualizations are generated using UMAP Armstrong_Martino_Rahman_Gonzalez_Vázquez-Baeza_Mishne_Knight_2021. To support efficiently storing, querying and analyzing embeddings we developed JointDb, an HDF5-based database designed to integrate Hi-C embeddings with additional genomic signals. Signal tracks from gene expression (RNA-seq), chromatin accessibility (ATAC-seq), and histone modifications (ChIP-seq) are loaded via pybbi from bigwig files and aggregated to match the resolution of Hi-C embeddings.

JointDb serves as a powerful resource for integrative analyses, allowing researchers to interpret chromatin architecture changes within broader genomic contexts.

Availability, Examples, Testing and Development

Jointly-HiC is freely available on GitHub under a GPLv3 license and can be installed as a package from PyPI or as container image from the Github container repository. It includes an automated regression test suite and release pipeline. Examples analyzing public datasets are available as notebooks in the repository. Jointly-HiC embraces an open-source development model and encourages community feedback or contributions through issues and pull requests.

Citations

Citations to entries in paper.bib should be in rMarkdown format.

If you want to cite a software repository URL (e.g. something on GitHub without a preferred citation) then you can do it with the example BibTeX entry below for (**?**).

For a quick reference, the following citation commands can be used: - `@author:2001` -> "Author et al. (2001)" - `[@author:2001]` -> "(Author et al., 2001)" - `[@author1:2001; @author2:2001]` -> "(Author1 et al., 2001; Author2 et al., 2002)"

Figures

Figures can be included like this: Caption for example figure. and referenced from text using section .

Figure sizes can be customized by adding an optional second parameter: Caption for example figure.

Acknowledgements

We acknowledge contributions from Brigitta Sipocz, Syrtis Major, and Semyeong Oh, and support from Kathryn Johnston during the genesis of this project.

References

Belton, J.-M., McCord, R. P., Gibcus, J. H., Naumova, N., Zhan, Y., & Dekker, J. (2012). Hi-c: A comprehensive technique to capture the conformation of genomes. *Methods (San Diego, Calif.)*, *58*(3), 268–276. https://doi.org/10.1016/j.ymeth.2012.05.001

Imakaev, M., Fudenberg, G., McCord, R. P., Naumova, N., Goloborodko, A., Lajoie, B. R., Dekker, J., & Mirny, L. A. (2012). Iterative correction of hi-c data reveals hallmarks of chromosome organization. *Nature Methods*, *9*(10), 999–1003. https://doi.org/10.1038/nmeth.2148

Lafontaine, D. L., Yang, L., Dekker, J., & Gibcus, J. H. (2021). Hi-c 3.0: Improved protocol for genome-wide chromosome conformation capture. *Current Protocols*, *1*(7), e198. https://doi.org/10.1002/cpz1.198