The Journal of Open Source Software

MatrixLM: A Julia package to obtain closed-form least squares estimates for matrix linear models.

Gregory Farage [1*], **Jane W Liang** [2*], **Chenhao Zhao** [1], **Harsh Vardhan Dubey** [1], **and Śaunak Sen** [1]

1 Division of Biostatistics, Department of Preventive Medicine, University of Tennessee Health Science Center, United States **2** Division of Research, Kaiser Permanente Northern California, United States * These authors contributed equally.

DOI: 10.21105/joss.10530

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @palday
- @cecileane

Submitted: 05 February 2026
Published: 24 September 2026

Summary

`MatrixLM` is an open-source Julia (Bezanson et al., 2017) package for fitting matrix linear models, which extend classical linear regression to a bilinear framework for matrix-valued responses. It is designed for analyzing high-throughput assays in which both rows and columns of the data matrix have associated covariates, such as in metabolomics, proteomics, or chemical genetic screens.

In a matrix linear model, the entries of a response matrix are modeled as a joint function of sample-level covariates (e.g., treatment group, demographic factors) and feature-level covariates (e.g., molecular or anatomical annotations, biological groups, pathways). `MatrixLM` implements efficient estimation and inference for this class of models using fast matrix operations whenever possible allowing users to fit large numbers of models while retaining an explicit linear model interpretation. The inputs include a response matrix and two design matrices encoding the row and column covariates, and the main outputs include estimated coefficients, standard errors, and test statistics for user-specified contrasts.

Compared with workflows built from many separate univariate models, `MatrixLM` provides a unified interface for specifying, fitting, and summarizing matrix linear models. This reduces code duplication, improves reproducibility, and makes it easier to express hypotheses that naturally involve both sample- and feature-level information (for example, testing for differential effects across feature groups or experimental conditions). By providing an efficient implementation in Julia, `MatrixLM` enables researchers to perform interpretable analyses of large structured matrix-valued data.

Statement of need

High-throughput studies in biology and medicine often produce matrix-shaped data where each row corresponds to a sample (e.g., a patient, mutant strain, or experimental unit) and each column represents a molecular measurement (e.g., metabolite, gene, or phenotype). In many applications, both the samples and the measured features have associated metadata that should be incorporated into the analysis. However, existing tools either ignore these annotations or handle them in a fragmented, two-step fashion.

Standard approaches often involve fitting a separate model to each feature (e.g., using t-tests or linear models) and then performing a second-stage enrichment or grouping analysis. This approach is limited in two key ways: (1) it does not handle overlapping or quantitative annotations well, and (2) it fails to exploit shared structure among features or among samples. Dimension-reduction and machine learning methods address some of these issues, but often

sacrifice interpretability and do not provide familiar statistical outputs like effect sizes or confidence intervals.

MatrixLM addresses this gap by implementing matrix linear models (MLMs), a class of bilinear models that allow researchers to directly model associations between sample-level and feature-level characteristics. MLMs naturally accommodate both categorical and continuous annotations, support hypothesis testing, and enable users to assess the effect of covariates while adjusting for confounding structure. Compared to standard univariate workflows, MLMs offer better interpretability and power, especially when annotations overlap or when feature relationships are complex, as demonstrated in both chemical genetic screens (Liang et al., 2019) and metabolomics applications (Farage et al., 2025).

Despite their utility, matrix linear models have not been widely available in reusable, general-purpose software. MatrixLM provides a fast, open-source Julia implementation with a user-friendly formula interface, making it easier for applied researchers to fit, interpret, and extend these models in large-scale studies.

State of the field

The best-known software ecosystem for high-throughput biological data analysis is Bioconductor (Huber et al., 2015). We are not aware of any package that provides the ability for large-scale bilinear models in this ecosystem. The LIMMA family of packages (Ritchie et al., 2015), fits separate linear models for each feature and borrows information across features using empirical Bayes methods. While they use sample or individual-level information, they do not use feature annotations in their modeling, which is central to MLMs. In the plant breeding literature, bilinear models are often used for assessing gene-environment interactions. The best known packages are metan (Olivoto & Lúcio, 2020) and Bilinear (Santantonio, 2022), but neither is suited for high-throughput phenotypes, although they provide other more specialized features such as missing data imputation using the EM algorithm. In principle, general-purpose packages such as GLM.jl (Bates & others, 2024) and MixedModels.jl (Alday & Bates, 2026) could be used to fit bilinear models using the Kronecker product formulation of MLMs. In practice, this is impractical because the Kronecker product approach has memory requirements beyond standard workstation hardware.

MatrixLM addresses these gaps by providing a dedicated Julia implementation for high-throughput biological data that avoids explicit Kronecker-product formulations. It also offers a clear interface for specifying row and column covariates, efficient estimation, and interpretable statistical summaries.

Software design

`MatrixLM` is designed around two main goals: preserving interpretability and making matrix linear models practical and fast for use on high-throughput data.

In addition to leveraging the speed of the Julia programming language, a central design choice in `MatrixLM` is to exploit the algebraic structure of matrix linear models to compute least-squares estimates directly, rather than relying on a generic regression engine built on an explicit Kronecker-product design matrix. This reduces memory load and improves computational speed. The implementation leverages Julia's `LinearAlgebra.jl` and its integration with OpenBLAS. This allows MatrixLM to offload heavy matrix multiplications to highly optimized routines.

In addition to estimation, the package provides standard errors, t-statistics, and p-values for the estimated coefficients. For high-throughput data where standard assumptions such as normality may not hold, or for small sample sizes, `MatrixLM` also supports permutation testing, offering a flexible alternative to traditional parametric tests.

The package additionally emphasizes usability through a model formula specification for constructing the row and column design matrices. This lowers the barrier for researchers who are already familiar with standard regression workflow syntax, while still allowing flexible encodings of both categorical and continuous annotations.

Together, these design choices favor clarity, extensibility, and domain-oriented model specification.

Mathematical framework

Matrix linear models extend ordinary linear regression to situations where the outcome is a whole matrix rather than a single response vector. We arrange the data as follows (see schematic representation in Figure 1):

- Y is an $n \times m$ matrix of high-throughput measurements (rows = samples, columns = features).
- X is an $n \times p$ matrix of sample-level covariates (e.g., treatment group, sex, clinical variables).
- Z is an $m \times q$ matrix of feature-level covariates (e.g., metabolite class, pathway, or other annotations).
- B is a $p \times q$ matrix of regression coefficients linking the sample and feature covariates.

The matrix linear model assumes

$$Y = XBZ^T + E,$$

where E is an $n \times m$ matrix of residuals. In element-wise form, each entry y_{ij} is written as

$$y_{ij} = \sum_{k=1}^{p} \sum_{\ell=1}^{q} x_{ik}\, z_{j\ell}\, b_{k\ell} + e_{ij}.$$

As the equation above shows, the elements of B may be interpreted as interactions between the columns of X and the columns of Z. The residuals are assumed to have zero mean, with a covariance Σ across the features and independent across samples,

$$V(\text{vec}(E)) = \Sigma \otimes I.$$

The current implementation of MatrixLM requires complete response and predictor matrices. Missing data should therefore be addressed before model fitting using an appropriate method selected according to the assumed missingness mechanism and the structure of the data.

We treat X and Z as known and estimate B by least squares, choosing $\hat{B}$ to minimize the Frobenius norm of the residuals,

$$\hat{B} = \arg\min_{B} \|Y - XBZ^T\|_F^2.$$

This optimization problem has a closed-form solution:

$$\hat{B} = (X^T X)^{-1} X^T Y Z\, (Z^T Z)^{-1},$$

when $X^T X$ and $Z^T Z$ are invertible. From $\hat{B}$, the software can construct fitted values $\hat{Y} = X\hat{B}Z^T$ and standard errors and test statistics for entries of $\hat{B}$ or for user-specified linear contrasts, directly analogous to classical linear models. The covariance matrix Σ is estimated from the empirical covariance of the residuals. Since the number of features can be comparable or greater than the number of samples, the covariance estimate can be singular. The package provides the user the option of using a shrinkage estimator for the covariance matrix which is non-singular.

When there is only a single feature (one column of Y) and no feature-level design matrix Z, this framework reduces to ordinary linear regression with design matrix X. The `MatrixLM` package therefore generalizes familiar linear modeling ideas to matrix-valued outcomes with structured annotations on both rows and columns.

Figures

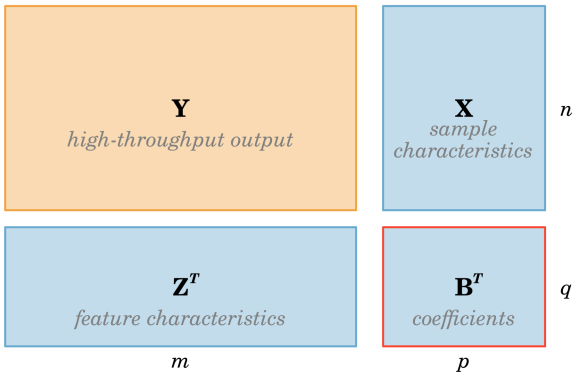

Figure 1: A visualization of the response ($\mathbf{Y} : n \times m$), sample covariates ($\mathbf{X} : n \times p$), feature covariates ($\mathbf{Z} : m \times q$), and coefficients ($\mathbf{B} : p \times q$) matrices for a matrix linear model. The dimensions in the model correspond to n samples/individuals, m features/measurements, p sample covariates, and q feature covariates; the matrix $\mathbf{B}$ is to be estimated.

Research impact statement

`MatrixLM` provides a practical implementation of matrix linear models for encoding relationships and groupings high-throughput matrix-shaped data with annotations on both samples and features. By combining a flexible formula interface with fast, closed-form least-squares estimation, the package makes it straightforward for applied researchers to encode biological or experimental structure directly into their models and apply matrix linear models in reproducible workflows.

Conclusion and future directions

In ongoing work, we are extending this framework to penalized matrix linear models for high-dimensional settings. In particular, we are developing a companion Julia package, `MatrixLMnet`, which implements elastic-net and related penalties on the coefficient matrix to support variable selection and regularization in matrix linear models.

AI usage disclosure

Grammarly was used during the drafting of this manuscript to assist with linguistic polishing. The authors used GitHub Copilot to assist in coding testing functions. All Copilot-suggested code was reviewed, tested, and validated by the human authors to ensure correctness. The authors take full responsibility for the content of this manuscript.

Acknowledgements

This work started when JWL was a summer intern at UCSF, and continued when she was a scientific programmer at the University of Tennessee Health Science Center (UTHSC). We thank both UCSF and UTHSC for funding, and a supportive environment. This work was partly supported by National Institutes of Health grants GM-070683 (SS), GM-078338 (SS), GM-123489 (SS, GF), ES-022841 (SS), and DA-044223 (SS).

References

Alday, P. M., & Bates, D. (2026). *MixedModels.jl*. Zenodo. https://doi.org/10.5281/ZENODO.18930339

Bates, D., & others. (2024). *GLM.jl: Linear and generalized linear models in Julia*. https://github.com/JuliaStats/GLM.jl.

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Farage, G., Zhao, C., Choi, H. Y., Garrett, T. J., Elam, M. B., Kechris, K., & Sen, Ś. (2025). Matrix linear models for connecting metabolite composition to individual characteristics. *Metabolites*, *15*(2). https://doi.org/10.3390/metabo15020140

Huber, W., Carey, V. J., Gentleman, R., Anders, S., Carlson, M., Carvalho, B. S., Bravo, H. C., Davis, S., Gatto, L., Girke, T., & others. (2015). Orchestrating high-throughput genomic analysis with bioconductor. *Nature Methods*, *12*(2), 115–121. https://doi.org/10.1038/nmeth.3252

Liang, J. W., Nichols, R. J., & Sen, Ś. (2019). Matrix Linear Models for High-Throughput Chemical Genetic Screens. *Genetics*, *4*(212), 1063–1073. https://doi.org/10.1534/genetics.119.302299

Olivoto, T., & Lúcio, A. D. (2020). Metan: An R package for multi-environment trial analysis. *Methods in Ecology and Evolution*, *11*(6), 783–789. https://doi.org/10.1111/2041-210X.13384

Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., & Smyth, G. K. (2015). Limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Research*, *43*(7), e47–e47. https://doi.org/10.1093/nar/gkv007

Santantonio, N. (2022). *Nsantantonio/Bilinear*. https://github.com/nsantantonio/Bilinear