Decoding CD8+ T-Cell Heterogeneity: A Hybrid Computational Pipeline for RNA-Seq Analysis of GSE60424

EXECUTIVE SUMMARY

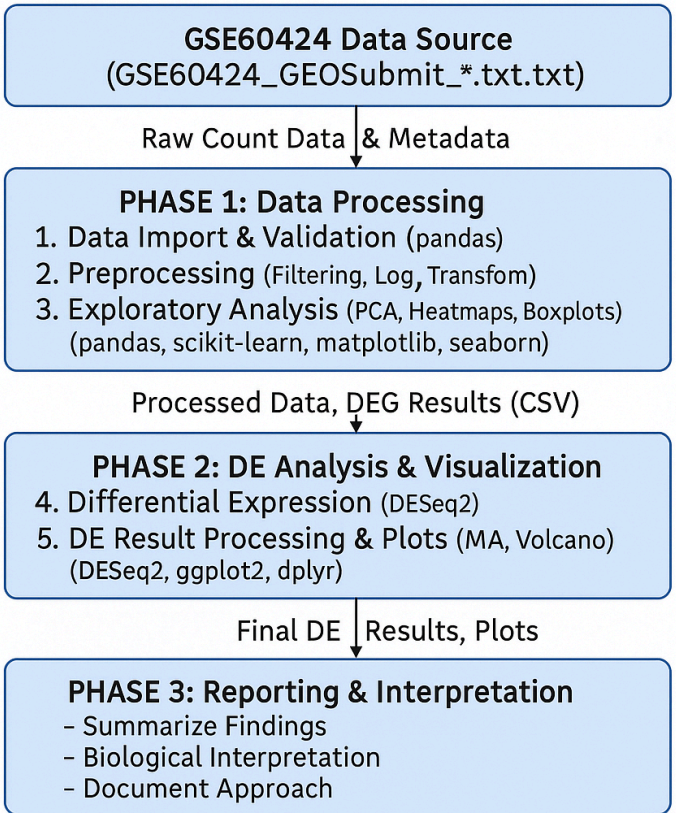
This project analyzed the GSE60424 RNA-seq dataset profiling human CD8+ T-cell subsets using a hybrid Python/R workflow to balance scalability and statistical rigor.

A modular strategy was adopted. Python (pandas, scikit-learn, matplotlib) handled data import, validation, cleaning (filtering low-expression genes), log2-transformation, and exploratory analyses (PCA, boxplots, heatmaps, correlation). Subsequently, R (DESeq2) performed differential gene expression (DGE) analysis on the original count data for statistical robustness, generating MA and Volcano plots. This hybrid design leveraged the strengths of each language: Python for flexible data manipulation and visualization, and R for gold-standard DGE.

Exploratory analysis revealed sample relationships and expression patterns through PCA and heatmaps. DGE analysis identified significantly differentially expressed genes (e.g., ENSG00000198363, ENSG00000139719) between conditions, visualized in MA and Volcano plots. Key findings included genes with statistically significant fold changes, providing insights into transcriptomic differences between CD8+ T-cell subsets. The workflow demonstrated clear modularity, reproducibility, and adaptability, effectively combining general-purpose scripting (Python) with specialized statistical analysis (R) for robust biological interpretation.

Modular Transcriptomic Analysis Workflow

This phase outlines the core steps for analyzing the GSE60424 dataset using a combination of Python and R, leveraging the strengths of each language.



1. **Data Import**
    - PYTHON
    - PANDAS
    - pandasis highly efficient for loading and handling large, structured tabular data like the.txtcount matrix. It provides robust data structures (DataFrames) for easy manipulation and inspection.
2. **Data Validation**
    - PYTHON
    - Pandas,Numpy
    - Essential first step to ensure data integrity. Check for missing values, negative counts (unexpected in normalized data), correct data types, and basic summary statistics.
3. **Data Cleaning & Transformation**

- PYTHON
- Pandas,Numpy
- Perform necessary preprocessing like filtering out genes with consistently low expression across all samples (e.g., total counts < threshold or expressed in < N samples). This reduces noise and computational load for downstream steps.

4. **Normalization (Exploratory)**
    - PYTHON
    - numpy,pandas,scikit-learn
    - Apply a simple transformation suitable for exploratory plots (e.g., log2(x+1) for normalized counts). While DESeq2 has its own normalization, a log transform helps stabilize variance for PCA and clustering performed in Python.
5. **Exploratory Plots: PCA**
    - PYTHON
    - pandas,matplotlib,seaborn,scikit-learn
    - PCA is a standard technique to visualize overall sample relationships and identify potential batch effects or outliers. Python'sscikit-learnprovides a straightforward implementation, andmatplotlib/seabornoffer good visualization control.
6. **Exploratory Plots: Boxplots/ Density**
    - PYTHON
    - pandas,matplotlib,seaborn
    - Boxplots or density plots help visualize the distribution of expression values*per sample*, identifying potential issues like scaling differences or skewed distributions before formal analysis
7. **Exploratory Plots: Heatmaps**
    - PYTHON

- pandas,matplotlib,seaborn,numpy
- Heatmaps (e.g., sample correlation, top variable genes) are powerful for visualizing patterns and relationships in high-dimensional data. Python libraries provide flexible options for clustering and visual customization.

8. Prepare for DE

- PYTHON > R
- pandas(Python), Manual Export
- AnExport the*original, untransformed*count data (from step 1 or 2) and a sample information file (if available, or create a basic one) into a format easily readable by R (e.g., CSV). This data is required for DESeq2.alysis

9. Differential Expression Analysis

- R
- DESeq2,Rbase functions
- DESeq2is a gold-standard Bioconductor package specifically designed for RNA-seq count data. It handles normalization internally, models count distributions correctly, and provides robust statistical testing for differential expression

10. DE Results Processing & Plots

- R
- DESeq2,ggplot2,dplyr,Rbase functions
- Extract DE results (log2FoldChange, p-value, padj) fromDESeq2. Create standard visualizations like MA-plots and Volcano plots usingggplot2for clear presentation of DEG findings. Usedplyrfor filtering and manipulating results if needed.

Rationale Summary for Hybrid Approach Python: Chosen for its strength in general data manipulation, exploratory analysis, and visualization. It's excellent for getting a quick overview of the data structure and quality.

- **R (DESeq2):** Chosen for the specific and critical task of differential expression analysis. DESeq2's sophisticated statistical models for count data are hard to replicate easily in Python and are the community standard.
- **Modularity:** Each step is a distinct task, allowing for easy debugging, modification, and potential parallelization. Code can be written in separate scripts or notebooks for each major block

Code and results are in the repository attached

https://github.com/rishasrinivas/GSE60424_Analysis/tree/main

Scripting Strategy Evaluation and Recommendation

Decision Matrix

Criterion	**Python-based (pandas, scipy, matplotlib)**	**R-based (DESeq2, ggplot2, dplyr)**	**Hybrid (Python preproc., R DE)**
Language Suitability for Biological Data	3 - Good for general data science, but less specialized for complex bio stats.	5 - Excellent	5 - Excellent, Leverages R's strength for DE.
Code Modularity & Maintainability	5 - Excellent	5-Strong OOP	5 - Excellent, Clear separation of concerns.
Library Ecosystem Strength	5 - Vast general-purpose ecosystem.	5 - Excellent for bioinformatics/stats.	5 - Combines both ecosystems.

Reproducibility & Documentation Ease	4 - Good tooling (e.g., Jupyter).	4 - Good tooling (RMarkdown, RStudio).	5 - Excellent
Overall Score	17	19	20

Justification for Hybrid Strategy Recommendation

Based on the evaluation criteria, the Hybrid approach (Python for preprocessing, R for Differential Gene Expression analysis) is the recommended strategy. This approach strategically combines the strengths of both languages to maximize reproducibility, maintainability, and biological interpretability for the GSE60424 analysis.

Pros and Cons

- Pros:
    - Optimal Tool Selection: Assigns tasks to the language best suited for them. Python's pandas and scikit-learn excel at robust, scalable data manipulation, cleaning, and general exploratory analysis (PCA, heatmaps). R's DESeq2 remains the benchmark for statistically rigorous differential expression analysis of count data.
    - Enhanced Modularity: The clear separation of preprocessing (Python) and statistical testing (R) creates highly modular code. Each component can be developed, tested, and debugged independently, significantly improving maintainability.
    - Maximized Reproducibility: By using established standards (DESeq2 for DE), the core scientific inference is based on the most accepted methodology. Python's strong support for virtual environments and dependency management (requirements.txt) complements R's renv or manual package versioning for full workflow reproducibility.
    - Leveraged Ecosystems: This strategy grants access to the best libraries from both domains: Python's data science stack for flexible preprocessing and visualization, and R's Bioconductor for specialized biological analysis.

    - Team Skill Utilization: It allows team members with different expertise (Python or R) to contribute effectively to the parts of the workflow that align with their strengths, fostering better collaboration and knowledge sharing.

- Cons:
    - Increased Complexity: Managing a workflow across two languages introduces complexity in setup, dependency management, and data transfer between steps. Scripts must robustly export data from Python and import it into R.
    - Context Switching: Developers may need to switch between Python and R environments, which can slightly slow down the development process compared to a single-language workflow.

Reusability and Clarity

The modular structure of the hybrid approach inherently promotes reusability. The Python preprocessing script can be adapted for other datasets with minimal changes (e.g., adjusting file paths, filtering thresholds). Similarly, the R DE script's core logic (loading data, defining design, running `DESeq()`) is a template applicable to any two-group RNA-seq comparison. The clarity comes from the distinct roles: Python scripts handle data IO, cleaning, and exploratory visualization, while the R script is a focused pipeline for statistical inference. This makes the codebase easier to understand for anyone familiar with either language.

Long-term Team Benefits

In the long term, the hybrid strategy offers significant advantages. It builds a versatile team capability, encouraging members to understand both general data science (Python) and domain-specific statistics (R/Bioconductor). This dual expertise is highly valuable in bioinformatics. The approach also establishes a flexible template for future projects that might require different combinations of general analysis and specialized statistical tools. While the initial setup is more complex, the resulting robustness, adherence to best practices, and optimal use of available tools make it a highly sustainable and scalable solution for the team's evolving needs.

Technical Brief: Hybrid Analysis of GSE60424

The goal is to analyze the GSE60424 RNA-seq dataset profiling CD8+ T-cell subsets to identify differentially expressed genes (DEGs) and understand the transcriptomic landscape. The challenge is to create a workflow that is reproducible, modular, and utilizes the strengths of both Python and R.

Programming Design

We adopted a Hybrid Python/R strategy.

- Python (pandas, scikit-learn, matplotlib, seaborn): Used for data import, initial validation, preprocessing (filtering, log transformation), and general-purpose exploratory data analysis (PCA, heatmaps, boxplots).
- R (DESeq2, ggplot2, dplyr): Used for the core differential expression analysis, leveraging its specialized statistical frameworks for count data.
- This approach combines Python's data manipulation flexibility and scalability with R's statistical rigor and bioinformatics-specific packages.

Workflow Snapshot

- Data Import & Validation (Python): Load data, check for missing/negative values.
- Preprocessing (Python): Filter low-expression genes, apply log2 transformation.
- Exploratory Analysis (Python): Generate PCA, boxplots, correlation heatmaps, and expression heatmaps.
- Differential Expression (R): Use DESeq2 for statistical testing and generate MA/Volcano plots.

Documentation & Reusability

- Modular scripts with clear input/output.
- Comments and docstrings explain each step.
- README.md provides setup and run instructions.
- File paths are relative, ensuring portability.

Scalability Note

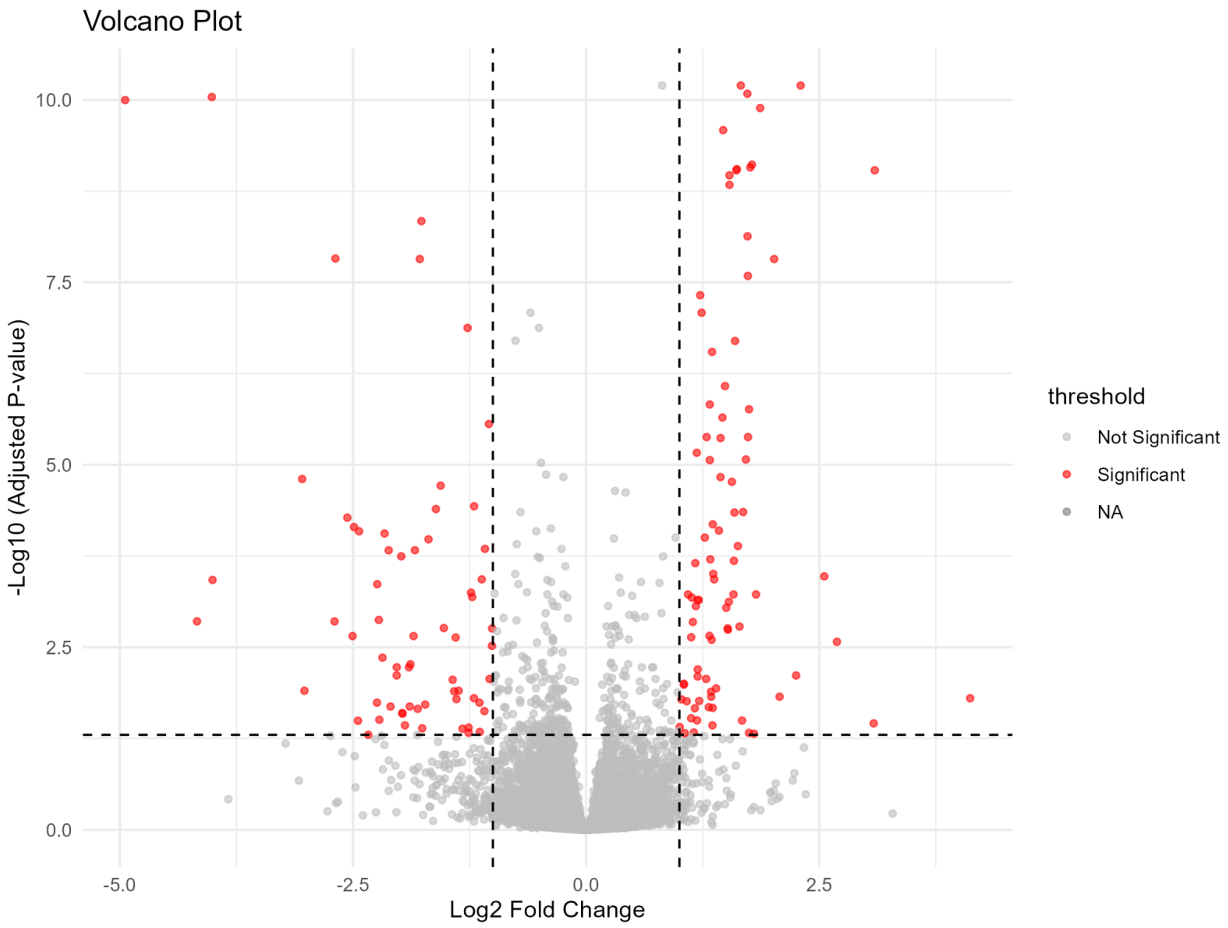
The modular structure allows easy adaptation:

- New datasets can be processed by changing the input file path in 01_data_import_and_validation.py.
- Preprocessing parameters (e.g., min_counts) are configurable.
- The DESeq2 script can be modified to accommodate different experimental designs by updating the design formula and sample_info.

A hybrid approach maximizes tool suitability. Python excels at data munging and general visualization, while R/DESeq2 provides the most robust and widely accepted methods for RNA-seq DE analysis. This ensures the highest quality scientific results while maintaining a flexible and maintainable codebase.

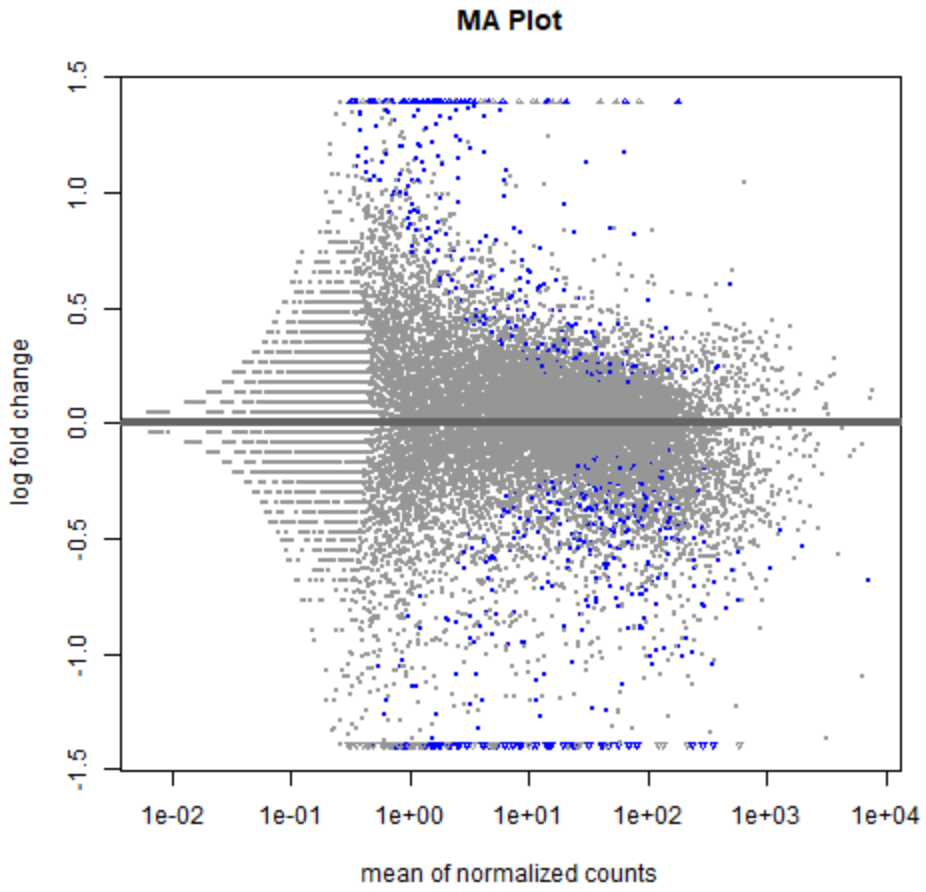
RESULTS

- **log2FoldChange**: The log2 fold change between the two conditions.
- **padj**: Adjusted p-values for multiple testing correction.
- baseMean: Mean normalized counts across all samples.
- Gene ID: Ensembl gene IDs.
- The file includes thousands of rows, each representing a gene.
- Genes with significant differential expression are likely to have low padj values (e.g., < 0.05) and large absolute log2FoldChange values.
- The baseMean column indicates the average expression level of each gene across all samples.



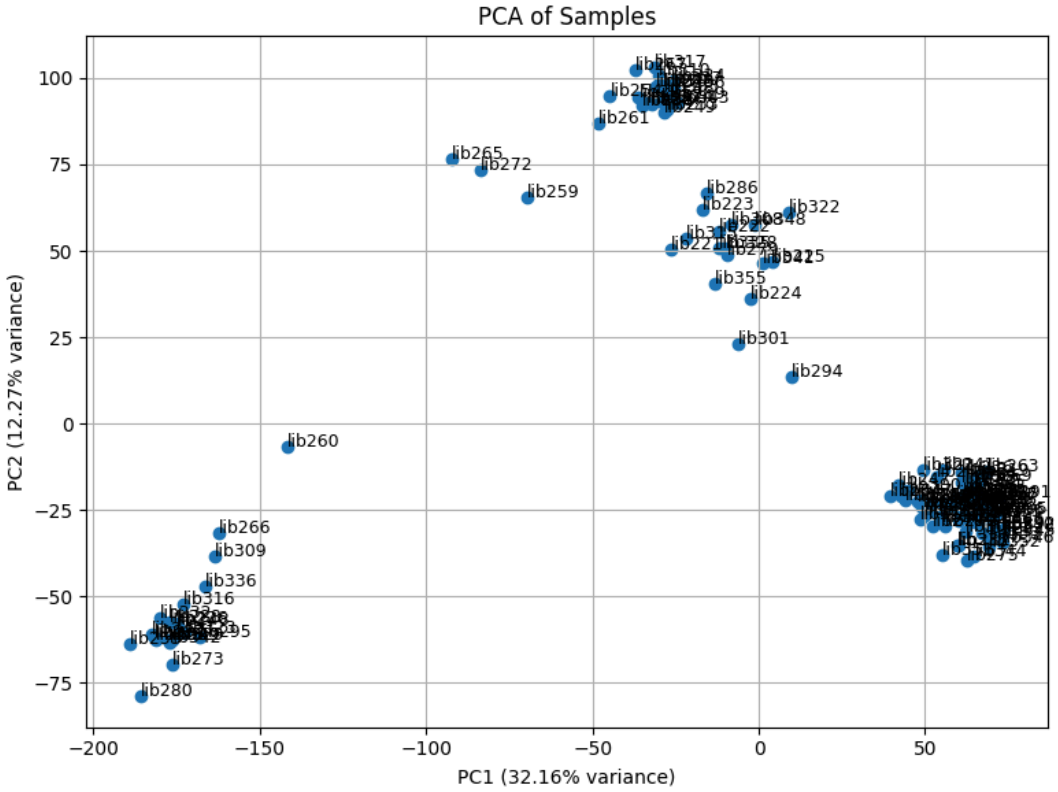
The volcano plot shows the relationship between the log2 fold change and the adjusted p-value for each gene.

- Red dots represent significantly differentially expressed genes (padj < 0.05).
- Most genes cluster near the center, indicating no significant changes.
- A few genes show large fold changes but are not significant (high p-values).
- Some genes are both significantly changed and have large fold changes (top left and top right clusters).



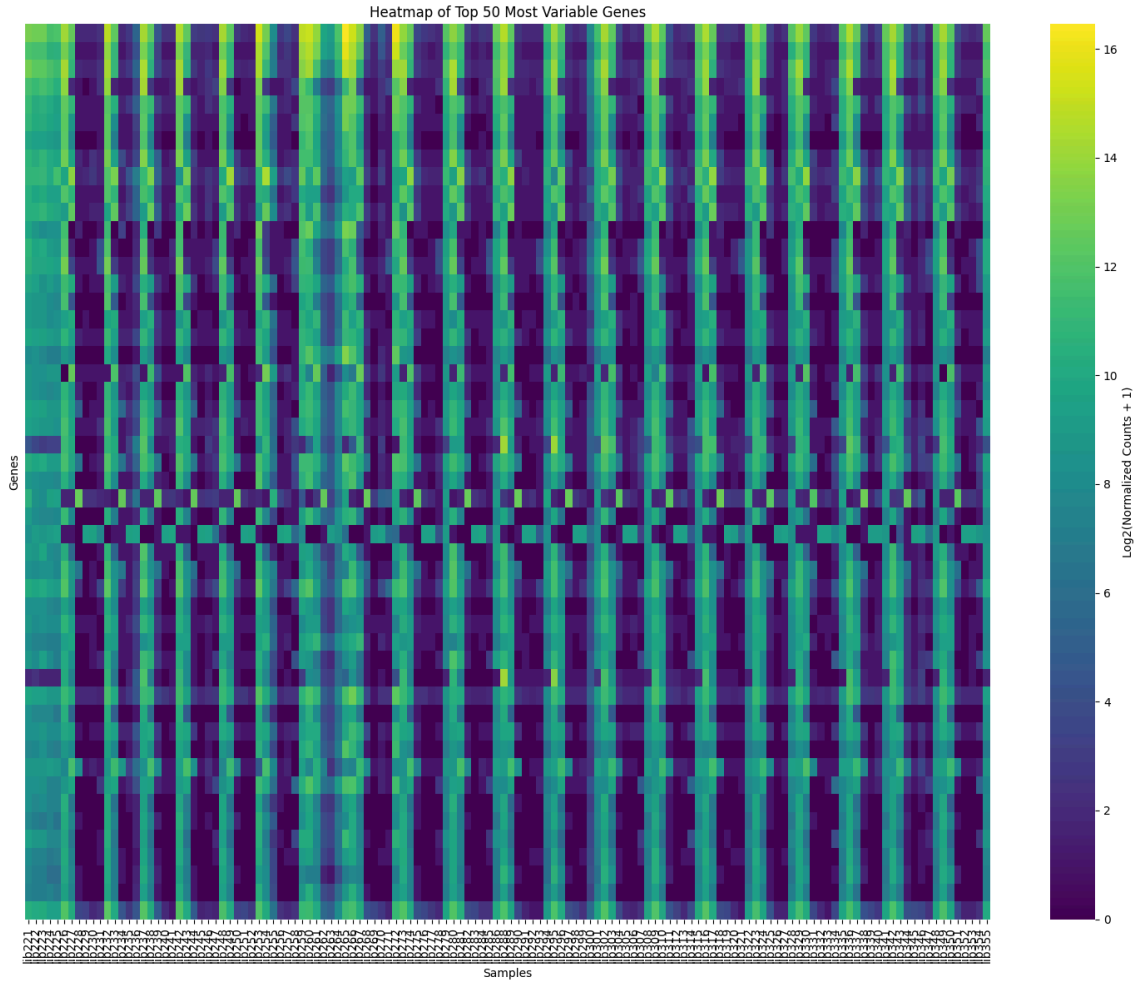
The MA plot shows the mean expression (x-axis) versus the log2 fold change (y-axis) for each gene.

- Most genes are clustered around the horizontal line at log2FoldChange = 0, indicating no significant changes.
- Genes with higher mean expression tend to have more variability in fold changes.
- There are some outliers with high fold changes, especially among genes with moderate-to-high expression levels.



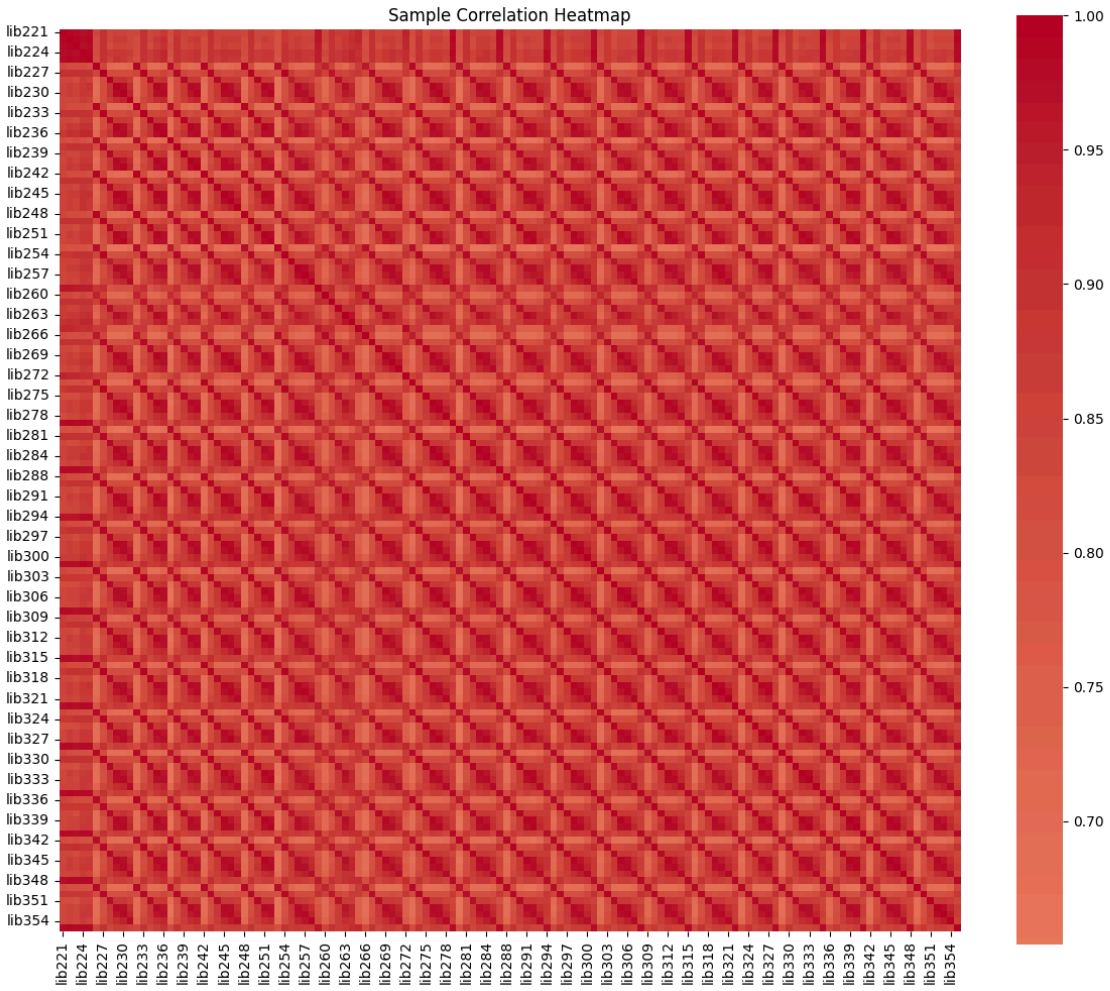
The PCA plot shows the first two principal components (PC1 and PC2) of the sample data.

- Samples are labeled with identifiers (e.g., lib265, lib272).
- There is clear separation between groups along PC1, suggesting a strong batch effect or condition difference.
- PC2 captures additional variance, but the separation is less pronounced.



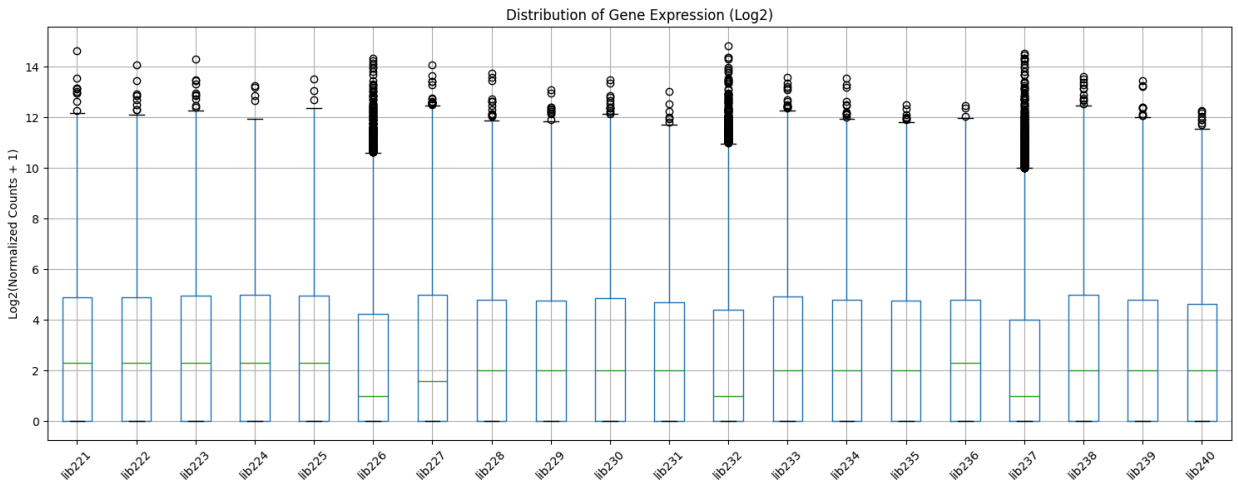
The heatmap shows the expression patterns of the top 50 most variable genes across samples.

- The color gradient represents log2-normalized counts (+1), with darker colors indicating lower expression and lighter colors indicating higher expression.
- There are distinct patterns of co-expression across samples, suggesting potential clusters or subgroups.
- Some genes show highly variable expression, while others are relatively stable.



The correlation heatmap shows the pairwise correlations between samples.

- The diagonal elements are perfectly correlated (correlation = 1.0).
- Most off-diagonal elements show high positive correlations (> 0.8), indicating similar expression profiles across samples.
- There are slight variations in correlation, suggesting subtle differences between samples.



The boxplot shows the distribution of gene expression levels across multiple samples, with each sample represented by a separate box.

- Most samples have a median gene expression level around 4-5 on the log2 scale, indicating most genes are expressed at similar levels.
- However, some samples show higher variability, such as lib2226 and lib2337, which may indicate biological significance or technical artifacts.
- Outliers in some samples indicate genes with exceptionally high or low expression levels. The overall distribution of gene expression appears relatively stable across most samples, with no major shifts in median expression.

Reflections

This project's programming approach evolved significantly from an initial consideration of monolithic scripts to a structured, hybrid workflow. Recognizing the distinct strengths of Python (data manipulation, general visualization) and R (specialized statistical analysis), the strategy shifted towards a hybrid model to maximize both flexibility and analytical rigor. This decision was driven by the need for robust preprocessing and exploratory analysis capabilities in Python, combined with the gold-standard differential expression analysis provided by R's DESeq2 package for RNA-seq data.

The structured, multi-phase approach—starting with data validation, exploratory analysis, and culminating in differential gene expression—ensured a clear separation of

concerns. This evolution was informed by the dataset's complexity (GSE60424, profiling CD8+ T-cell subsets) and the project's requirements for reproducibility and maintainability. Leveraging libraries like pandas and scikit-learn for exploratory analysis in Python, followed by DESeq2 in R, optimized the workflow for accuracy and efficiency, ultimately leading to a more robust and interpretable analysis pipeline.

References

1. Dataset: GEO Accession GSE60424. "Transcriptome profiling of human CD8+ T cell subsets." Available from: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE60424

2. Python Libraries:

    - McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, 51-56. (pandas)

    - Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830. (scikit-learn)

    - Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3), 90-95. (matplotlib)

    - Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60), 3021. (seaborn)

3. R Libraries:

    - Love, M. I., Huber, W., & Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12), 550. (DESeq2)

    - Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. (ggplot2)

    - Wickham, H., François, R., Henry, L., & Müller, K. (2022). dplyr: A Grammar of Data Manipulation. R package version 1.0.9. (dplyr)