Gene Regulatory Network Reconstruction Pipeline

Biological Context

Genome-wide association studies (GWAS) have identified numerous risk loci for various cancers that often influence gene expression through regulatory mechanisms. Understanding these regulatory circuits can reveal insights into cancer susceptibility and progression, aiding the identification of therapeutic targets and biomarkers. This pipeline focuses on constructing gene regulatory networks linking **master regulators** (transcription factors, TFs) to their putative target genes (regulons) using ARACNe, which infers networks based on mutual information between TFs and downstream targets. Networks are adjusted for somatic copy number alterations to reduce false positives. Additionally, Variant Set Enrichment (VSE) assesses whether network genes overlap disease-associated loci. This framework can be applied to cancer datasets from RNA-seq, microarrays, TCGA, and integrated with GWAS information to prioritize candidate regulators for experimental validation.

Repository Structure

```
Gene-Regulatory-Network/
├── docs/                          # Project overview & main instructions
│   └── README.pdf
├── scripts/
│   └── network_reconstruction.txt  # Main R-based network inference with verbose comments
├── data/
│   ├── expression_matrix_01.txt   # Gene expression data (TPM or counts)
│   ├── TF_list.txt               # List of transcription factors to include
│   ├── gene_sets.gmt             # Gene sets for enrichment testing
│   └── gwas_loci.txt             # GWAS loci for VSE
├── hpc/                           # HPC-specific scripts & instructions
│   ├── setup.sh                   # Environment setup for HPC jobs
│   ├── aracne_inference.sh        # Basic ARACNe job submission script
│   ├── post_processing.sh         # Post-inference aggregation and processing
│   └── README.pdf                 # HPC main instructions
└── results/                       # Generated outputs (networks, enrichment, logs)
```

Main Pipeline: network_reconstruction.txt

This script (located under `scripts/network_reconstruction.txt`) performs the following steps:

1. **Load Dependencies & Data**
   - Load expression matrix (`data/expression_matrix_01.txt`) and TF list (`data/TF_list.txt`).
   - Optionally load copy number alteration (CNA) matrix to adjust MI calculations.
2. **Preprocessing**
   - Convert TPM to log2 scale (or apply variance-stabilizing transformation) and filter genes with low expression.
   - Filter TFs according to `TF_list.txt` to define master regulator candidates.
3. **ARACNe Network Inference**
   - For each TF, calculate mutual information (MI) with all potential targets.
   - Apply data processing inequality (DPI) to remove indirect interactions.
   - Adjust MI values for CNAs if provided (to reduce biases from copy number).

    - Output TF–target pairs with MI above threshold (e.g., MI > 0.05).

4. **Network Aggregation**
    - Combine TF–target pairs into a global edge list (`results/ARACNe_edges.txt`).
    - Generate adjacency matrix (binary or weighted) for downstream analyses.
5. **Variant Set Enrichment (VSE)**
    - Load `data/gwas_loci.txt` and `data/gene_sets.gmt`.
    - Perform enrichment testing (e.g., Fisher's exact test) to assess overlap between network genes and GWAS loci.
    - Output enrichment statistics (`results/VSE_results.tsv`).
6. **Master Regulator Prioritization**
    - Rank TFs by number of targets, MI strength, and enrichment overlap.
    - Generate a report (`results/MR_ranking.tsv`) listing top candidate regulators.
7. **Session Logging & Reproducibility**
    - Write `sessionInfo()` and timing logs to `results/network_reconstruction_log.txt`.

Note: Do **not** modify the detailed steps or inline comments in this script; they document each analytical step extensively.

Quick Start (Local Execution)

1. **Clone the repo**
```
git clone https://github.com/<YOUR-USERNAME>/Gene-Regulatory-Network.git
cd Gene-Regulatory-Network
```
2. **Ensure R (≥ 4.0) and required R packages** are installed:
```
install.packages(c("data.table","dplyr","minet","IRanges","clusterProfiler","optparse"))
# If using Bioconductor:
if (!requireNamespace("BiocManager", quietly=TRUE)) install.packages("BiocManager")
BiocManager::install(c("aracne.network","GeneSetDb"))
```
3. **Review and adjust** data file paths at the top of `scripts/network_reconstruction.txt` (e.g., expression, TF list, CNA matrix).
4. **Run network reconstruction**:
```
Rscript scripts/network_reconstruction.txt \
  --expr data/expression_matrix_01.txt \
  --tf data/TF_list.txt \
  --cna data/CNA_matrix.txt \
  --out results/network
```
    - Replace `--cna` argument if no CNA adjustment is needed (omit or set `--cna NULL`).
5. **Inspect outputs** in `results/`:
    - `ARACNe_edges.txt` (TF–target edge list)
    - `VSE_results.tsv` (enrichment statistics)
    - `MR_ranking.tsv` (ranked master regulators)
    - `network_reconstruction_log.txt` (session info & logs)

HPC Scaling

For large datasets, use the HPC scripts under `hpc/`. See `hpc/README.pdf` for brief instructions on job submission, resource allocation, and batch processing.

Data Files

- **`data/expression_matrix_01.txt`**: Tab-delimited gene expression (e.g., TPM or counts).
- **`data/TF_list.txt`**: List of TF gene symbols/IDs to include as master regulators.
- **`data/gene_sets.gmt`**: Gene sets (GMT format) for enrichment (e.g., pathways, GO terms).
- **`data/gwas_loci.txt`**: GWAS loci mapped to gene IDs for VSE.

Ensure the expression matrix and metadata match identifiers in TF_list and GWAS loci for consistency.

Contact & License

- **Author**: Sally Yepes
- **Email**: sallyepes233@gmail.com

- **Last Updated**: June 2025
- **License**: MIT

Find detailed comments in `network_reconstruction.txt` for reproducibility and clarity of each analytical step.