OmniGS-R (v1.0)

Genomic Selection Pipeline Using R Packages

OmniGS-R is a powerful, flexible, and user-friendly Java-based pipeline designed for performing Genomic Selection (GS) analysis. It seamlessly integrates a wide array of popular R packages for statistical modeling, providing a unified platform for both cross-validation and prediction tasks in plant and animal breeding programs.

The pipeline supports multiple genomic marker types (SNPs, Haplotypes, Principal Components), and a suite of GS modeling algorithms, making it a comprehensive tool for breeders and researchers.

Table of Contents

Introduction

Genomic Selection accelerates genetic improvement by predicting the breeding values of individuals based on their genomic markers. OmniGS-R automates the complex workflow of GS, which includes data preprocessing, quality control, imputation, model training, and validation. By leveraging the robust statistical capabilities of R within a managed Java pipeline, OmniGS-R ensures reproducibility, scalability, and ease of use for both small-scale studies and large breeding populations.

Key Features

- **Flexible Analysis Modes:** Supports both **cross-validation** (for model evaluation) and independent **across-population prediction** (using a training set to predict a test set).
- **Multiple Marker Views:**
    - **SNP:** Direct use of Single Nucleotide Polymorphisms.
    - **HAP:** Conversion of SNPs into haplotype blocks using RTM-GWAS SNP-LD for potentially capturing epistatic effects.
    - **PCA:** Use of Principal Components as markers to reduce dimensionality and address multicollinearity.
- **Comprehensive Data Preprocessing:** Includes sample alignment, genotype harmonization, and missing data imputation.
- **Diverse GS Modeling Methods:** Integrates several state-of-the-art models via R packages:
    - **Bayesian Approaches:** BL (Bayesian LASSO), BRR (Bayesian Ridge Regression), BayesA, BayesB, BayesC via `BGLR`.
    - **Machine Learning:** Random Forest for Regression (`RFR`) and Classification (`RFC`), Support Vector Regression (`SVR`) and Classification (`SVC`).
    - **Linear Models:** Ridge-Regression BLUP (`RR-BLUP`) via `rrBLUP`.
    - **Kernel Methods:** Genomic BLUP (`GBLUP`) and Reproducing Kernel Hilbert Spaces (`RKHS`).

System Requirements & Installation

Prerequisites

1. **Java Runtime Environment (JRE):** Version 17 or higher must be installed. You can check by running `java -version` in your terminal.
2. **R:** Version 3.5 or higher must be installed and accessible from the command line. Check with `R --version`.
3. **Rscript:** This executable (included with R) must be in your system's PATH.

Installing R Libraries

Before running OmniGS-R, you must install the required R packages. Start an R session and run the following commands:

r

```
# Install required packages from CRAN

install.packages(c("rrBLUP", "BGLR", "randomForest", "e1071", "ade4", "sommer", "ggplot2", "G2P"))

If G2P is not available on CRAN, please download from GitHub and follow installation instruction:

https://github.com/cma2015/G2P
```

Installing `rtm-gwas-snpldb` tool

The `rtm-gwas-snpldb` tool for haplotype block identification can be downloaded from:

https://github.com/njau-sri/rtm-gwas

Obtaining OmniGS-R

Download the latest release JAR file (e.g., `gspipeline.jar`) from the Releases page of this repository.

Quick Start

1. **Prepare your data:** Have your VCF marker files and phenotypic data files ready.
2. **Create a configuration file:** Copy the sample below and modify the paths to match your system and data.
3. **Run the pipeline:**

bash

```
java -jar gspipeline.jar /path/to/your/config.txt
```

Configuration File

The pipeline is controlled by a single configuration file using an INI-style format.

Sample Configuration

ini

```
# This is a configuration file for OmniGS-R pipeline.

[Tools]
# installation folder (absolute path) of the GSPipeline
pipeline_home = /home/user/OmniGS-R

# haplotype block identification tool (included with OmniGS-R)
rtm_gwas_snpldb_path = /home/user/OmniGS-R/rtm_gwas/rtm-gwas-snpldb

# R path
RScriptPath = /usr/bin/Rscript

[General]
# variance explained for selection of number of principal components
pca_variance_explained = 0.95

# result output folder
result_folder = sample_results_CV

# Number of threads for parallel computation
threads = 7

# number of replicates in CROSS-VALIDATION mode
Replicates = 2

[GS_Mode]
# Mode: CROSS-VALIDATION | PREDICTION
mode = CROSS-VALIDATION

[Feature_view]
# Three marker types: raw SNPs (SNP), haplotypes (HAP) and principal components
(PCA)
```

```
marker_type = PCA

[Data]
# (training) marker file (for cross_validation or Prediction)
marker_file=/path/to/training_markers.vcf

# test marker file (required for PREDICTION mode, optional for CROSS-VALIDATION)
test_marker_file=/path/to/test_markers.vcf

# training phenotypic data file for both modes
training_pheno_file=/path/to/training_pheno.txt

# test phenotypic data file (optional, for PREDICTION mode only)
test_pheno_file=/path/to/test_pheno.txt

[Models]
# Choose GS modeling methods: True | False
# Parametric/linear models
RR-BLUP = True
GBLUP = True
BRR = True
BL = True
BayesA = True
BayesB = True
BayesC = True

# Non-parametric machine learning methods
RFR = True
SVR = True
RKHS = True

# Classifiers
RFC = True
SVC = True
```

```
[Hyperparameters]
# Model parameters for Bayesian methods
nIter = 12000
burnIn = 2000
```

Parameter Details

Section	Parameter	Description	Values
Tools	`pipeline_home`	Absolute path to OmniGS-R installation directory	File path
	`rtm_gwas_snpldb_path`	Path to haplotype block identification tool	File path
	`RScriptPath`	Path to RScript executable	File path
General	`pca_variance_explained`	Variance cutoff for PCA component selection	0.0-1.0 (e.g., 0.95)
	`result_folder`	Output directory for results	Directory path
	`threads`	Number of CPU threads for parallel processing	Integer
	`Replicates`	Number of CV replicates	Integer
GS_Mode	`mode`	Analysis mode	`CROSS-VALIDATION` or `PREDICTION`
Feature_view	`marker_type`	Type of markers to use	`SNP`, `HAP`, or `PCA`
Data	`marker_file`	Training population VCF file	File path
	`test_marker_file`	Test population VCF file (Prediction mode)	File path
	`training_pheno_file`	Training phenotype data	File path
	`test_pheno_file`	Test phenotype data (optional)	File path
Models	Various	Enable/disable specific GS models	`True` or `False`

Section	Parameter	Description	Values
Hyperparameters	`nIter`	MCMC iterations for Bayesian models	Integer (e.g., 12000)
	`burnIn`	MCMC burn-in period	Integer (e.g., 2000)

Input Files

Genotypic Data (Markers)

- **Format:** VCF (Variant Call Format) - can be compressed (.vcf.gz) or uncompressed
- **Requirements:**
    - For **Cross-Validation:** One VCF file for the training population
    - For **Prediction:** Two VCF files (training and test)

Phenotypic Data

- **Format:** Tab-delimited text file **with a header row**
- **Structure:**
    - First column: Individual/Sample IDs
    - Subsequent columns: Phenotypic values for different traits

Example `training_pheno.txt`**:**

text

```
SampleID        Yield   Height  Weight
sample_1        5.6     112     45
sample_2        4.8     105     42
sample_3        NA      108     44
```

Missing values should be coded as `NA`*. The pipeline will handle them automatically.*

Usage

1. **Prepare your configuration file** following the template above
2. **Run the pipeline:**

bash

```
java -jar gspipeline.jar /path/to/your/config.txt
```

3. **Monitor progress:** The pipeline will display progress in the console and write detailed logs to the output directory

For large datasets, you may need to increase memory allocation:

bash

```
java -Xmx8g -jar gspipeline.jar config.txt
```

Output

The pipeline generates a well-organized directory structure:

text

```
result_folder/
├── gs_<timestamp>.log              # Detailed log file
├── all_CV_results.txt              # Detailed CV results (CV mode)
├── CV_summary_statistics.csv       # Summary statistics (CV mode)
├── prediction_detailed_results.txt # Model results (Prediction mode)
│
├── trait_predictions/              # Predicted values for test set
│   └── <Trait>_<Model>_prediction_data.txt
├── plots/                          # Diagnostic plots
│   ├── MDS_plot.png               # Population structure
│   └── ...                        # Other visualizations
├── intermediate_data/              # Processed intermediate files
└── pheno_data/                     # Preprocessed phenotypic data
```

Troubleshooting

- **"RScript not found":** Verify the `RScriptPath` in your configuration file is correct
- **Missing R packages:** Check the log file for package errors and install missing packages in R
- **Memory errors:** Use `-Xmx` parameter to increase Java heap space (e.g., `-Xmx8g` for 8GB)
- **VCF file errors:** Ensure your VCF files are properly formatted and indexed

Citation

If you use OmniGS-R in your research, please cite:

> **OmniGS-R: A Comprehensive Genomic Selection Pipeline Using R Packages.** [Your Name/Institution]. Version 1.0. [URL to GitHub repository].

License

This project is licensed under the MIT License - see the LICENSE file for details.