R-COMPLETE Documentation

Vishvesh Karthik

Contents

ToC

- About
- Installation
- Requirements
- Run Examples
- Documentation
  - Parameters
  - User Data
  - COMPLETE.format.ids
  - QuickBLAST

    * QuickBLAST Options
  - Flow
    * EXTRACT_DATA()
    * FIND_TRANSCRIPT_ORTHOLOGS()

(TODO : Add Omit list which negates from the total gene list of all organisms)

R - COMPLETE

Pipeline for extracting localization elements/motifs using a comparitive approach. Library can be installed and tested on R(>=4.1). For a list of genes, the pipeline downloads full transcript sequences for all organisms (or selected organisms) from ENSEMBL or NCBI, Formats the headers and stores them according to the clusters (optionally taken from OrthoDB or clustered otherwise). Sequences without cluster information are clustered based on sequence identity/sequence coverage of Reciprocal Bidirectional BLAST Hits (RBH) of CDS regions. Next step would be to stitch the transcript regions into full length transcripts and align them. Pipeline is format agnostic(to my knowledge - TMK), packaged with a multi-threaded BLAST framework, to BLAST direclty from R (QuickBLAST - available a precompiled library) coupled with Arrow IPC, can load and convert BLAST formats (internally - to GRanges) and sports functions for performing Reciprocal Bidirectional Hits. The multithreaded BLAST framework can handle many-many BLAST Hits across organisms. The package is interfaced with bash, R transforms the data & calls the shots and bash handles files & BLASTing. Check requirements and installation instuctions before proceeding.

Ironically this repo is incomplete but the functionality in it works. Under Construction Indefinitely. Documentation can be found within the package, Play around with the functions for the rest

Installation (on R - Linux or RStudio Docker with WSL in Windows) :

```
sudo apt-get update && sudo apt-get install curl bzip2 parallel liblmdb-dev ncbi-blast+ samtools bedtool
```

From `bash` :

Install AGAT :

```
git clone https://github.com/NBISweden/AGAT.git # Clone AGAT
cd AGAT                                         # move into AGAT folder
perl Makefile.PL                                # Check all the dependencies*
make                                            # Compile
make test                                       # Test
sudo make install                                   # Install
```

```
BiocManager::install(c("Rhtslib", "devtools", "BiocManager", "Biostrings", "biomaRt", "S4Vectors", "IRan
remotes::install_git("https://github.com/vizkidd/R-COMPLETE.git")
```

From `R` :

REQUIRES :

- Linux with BASH ($SHELL must be set or /bin/bash must exist) (export SHELL="/bin/bash")
- **Config Files**
- Lot of space in `genomes_path`, `fasta_path` and `annos_path` path locations (in parameters file)
- GNU parallel (in $PATH - BASH functions)
- GffRead
- Samtools (in $PATH - BASH functions)
- Bedtools (in $PATH - BASH functions)
- ncbi-blast+ (Compile from .src.tar.gz with ./configure && make all_r && sudo make install) (or sudo alien -i ncbi-blast-X.XX.X+-2.src.rpm) (or download binaries) (Docs & Compilation)
  - Check if you have the binaries for *blastdb_path* and *makeblastdb*
- Zlib - (Compile from sources) (or) (sudo apt install libz-dev or yum install zlib-devel)
- LZMA SDK - (sudo apt-get install liblzma-dev or yum install xz-devel)
- BZLIB - (sudo apt-get install libbz2-dev libclang-dev or yum install bzip2-devel.x86_64)

OrthoDB : (Optional)

- OrthoDB (ODB) Flat Files (>= v10.1) (Pipeline is tested with ODB v12.2)
  - odb12v2_species.tab.gz - Ortho DB organism ids based on NCBI taxonomy ids (mostly species level)
  - odb12v2_genes.tab.gz -Ortho DB genes with some info
  - odb12v2_OG2genes.tab.gz - OGs to genes correspondence **(OR)**
  - odb12v2_OGgenes_fixed.tab.gz - Merged & Transformed ODB file (Done within pipeline - Only once)
  - odb12v2_OGgenes_fixed_user.tab.gz - Merged & Transformed ODB file BASED on user gene list (Done within pipeline - For different gene sets)

NOTE : Set `orthodb_path_prefix` in parameters file if you use OrthoDB files

Tools - (Paths in parameters file)

- MACSE (Path to the .jar)
- MAFFT (Compile from sources with extensions because *mafft-qinsi* is required)
- TRANSAT (Download preferred tarball and check INSTALL file)
- RNADECODER (Compiled program is in the bin/ of the repo) (Give path to the folder containing the binary)
- FastTree

Files (Config)

- **Parameters**
- User Data (Optional)
- Reference Organisms (`COMPLETE_env$org.meta` has the list of organisms available)

Run Example :

To run the example, from the context of your current working directory, + Download OrthoDB(ODB) files (optional) and Tools + Check config files + Provide paths and options in the parameters file + **NOTE : Default parameters file (parameters.txt) is at `fs::path_package("COMPLETE","pkg_data","parameters.txt")`**

```
params_list <- COMPLETE::load_params(fs::path_package("COMPLETE","pkg_data","parameters.txt"))
gene_list = fs::path_package("COMPLETE","pkg_data","genelist.txt")
user_data = fs::path_package("COMPLETE","pkg_data", "user_data.txt")
COMPLETE::EXTRACT_DATA(params_list = params_list, gene_list = gene_list, user_data = user_data, keep_da
COMPLETE::FIND_TRANSCRIPT_ORTHOLOGS(params_list = params_list, gene_list = gene_list, blast_program = "
```

NOTE : First run will take some time due to conversion of ODB file structure (if OrthoDB is used)

Documentation :

`{R} ?COMPLETE_PIPELINE_DESIGN (in R docs)`

PARAMETERS

The pipeline takes a single **parameter** file. This design was chosen, + To expose as many options as possible to the end-user + The pipeline uses BASH to BLAST and handle files (significantly faster than R) and the parameter file is shared between R and BASH.
+ Parameters tagged as output in comment column are outputs from COMPLETE

```
* Delimited by '=='
* Inputs and Ouputs are specified in the comments
* The file is of the format [param_id==value==comment] where param_id and value columns are CASE-SENSIT
* A default/example file is in fs::path_package("COMPLETE","pkg_data","parameters.txt")
```

USER DATA : (Optional)

```
* Columns Org,Version, genome, gtf
* Can accept empty or '-' in genome and/or gtf column. If empty or '-', the genome/gtf is looked up in
* A default/example file is in fs::path_package("COMPLETE","pkg_data","user_data.txt")
```

COMPLETE.format.ids

- Order of FASTA ID labels are stored in `COMPLETE_env$FORMAT_ID_INDEX`
- Sequences are labelled with the following long ID format of R-COMPLETE (specific to this pipeline and referred to as COMPLETE.format.ids) (seqID_delimiter & transcripID_delimiter set in parameters, `::` & `||` respectively in this context)
- COMPLETE.format.ids are indexed(internally) with `COMPLETE::index_BLAST_tables()` for compatibility

```
>$transcript_id $transcripID_delimiter $transcript_region ($strand) $seqID_delimiter $org_name/$DB/$org_
>SOME_TRANSCRIPT||cds(+)::SOMEORG/DB/VERSION::RANDOMGENE::ORTHOLOG_CLUSTERS
>ENSDART00000193157||cds(+)::danio_rerio/ensembl/115::sulf1::18335at7898,51668at7742,360590at33208
```

BLAST Functions

- QuickBLAST

QuickBLAST Options Same as BLAST but OUTPUT Format is not available. List of available options can be checked with `QuickBLAST::GetAvailableBLASTOptions()` (Empty elements from the list are removed and BLAST defaults are set on the c++ side). Enums used by QuickBLAST in C++ are not exposed in R and only integers are used, check `COMPLETE::GetQuickBLASTEnums()`.

FLOW

1) **EXTRACT_DATA():** Extracts the transcript regions for Protein Coding Transcripts (`provided in parameters, pipeline requires cds,5utr,3utr`) from BIOMART(ensembl),genbank(ncbi) and/or User provided genomes & GTFs. This functions uses biomaRt/biomartr for extracting data from BIOMART and BASH function *`extract_transcript_regions()`* for user provided data. Extraction priority/flow : `User Data > biomaRt > biomartr`

   - ODB Files are merged and transformed with BASH function *`merge_OG2genes_OrthoDB()`*
   - Orthologous genes are found for genes which are not present in the organism with BASH function *`check_OrthoDB()`*
   - Flank lengths are calculated from GTF data for missing UTRs (with variance correction, check *`?calculate_gtf_stats`*)
   - FASTA Nucleotide Sequences for given TRANSCRIPT_REGIONS are fetched from BIOMART/Genome

2) FIND_TRANSCRIPT_ORTHOLOGS() - Finds transcript-level orthologs based on minimum coverage and/or maximum sequence identity (check `?extract_transcript_orthologs`). Has a grouping mode(`group.mode`) and run mode(`run.mode`), to group transcript orthologs at the level of organisms, genes or Ortholog Clusters, sequences are grouped into any level of `COMPLETE_env$FORMAT_ID_INDEX` (Default - `COMPLETE_env$FORMAT_ID_INDEX$CLUSTERS`) and select transcript orthologs. Gene level grouping has more tight orthology and fewer transcript orthologs. Ortholog Cluster level grouping is a level higher than Genes (Because an Ortholog Cluster can have more than one gene) and have highest number of transcript orthologs with a lot of dissimilarity. Different run modes determine how transcript-level orthologs are selected by their HSP coverages after grouping. After grouping, non-overlapping BLAST hits which maximize coverage for each transcript are chosen with WISARD. Transcripts which do not have bi-directional hits are discarded with `COMPLETE::RBH()`. Finally, HSP coverage is calculated with `COMPLETE::calculate_HSP_coverage()` and transcripts are processed according to `run.mode` option which can be one of,

   - "coverage_distance" - Hits are filtered based on distance between bi-directional minimum HSP coverages (coverage_distance <= min_coverage_filter). This option selects more BLAST hits and should be used when the coverage values are very low (and the BLAST Hits/sequences are distant). `coverage_distance = 1 - (2 * aligned_length) / (query_length + subject_length)`. `(coverage_distance >= min_coverage_filter)`
   - "coverage_filter" - Filters Hits based on minimum coverage of HSPs from either direction. Use this option when the coverage values are high (and the BLAST Hits/sequences are closely related). `(cov_q >= min_coverage_filter && cov_s >= min_coverage_filter)`
   - "both" - Uses both "coverage_distance" and "coverage_filter" and is very strict. (Default) . `(coverage_distance >= min_coverage_filter && cov_q >= min_coverage_filter && cov_s >= min_coverage_filter)`
   - "no_filter" - Only calculates HSP coverages and does not filter any Hits .

   NOTE : ONLY USE THIS FUNCTION WHEN RUNNING THE PIPELINE OF R-COMPLETE. Use other helper function to work with custom BLAST files not generated by this R package.