Dirichlet Study Software Infrastructure

This documents explains the software components of the Dirichlet Study on 2025-07-17.

src/DirichletStudy.cpp

- DirichletStudy.cpp is a C++ source file that integrates with R using the Rcpp package.

- It includes the Rcpp header and a custom header rcpp_interface.hpp, which defines the DirichletStudyInterface class.

- The file uses the Rcpp module system to expose the DirichletStudyInterface class and its methods to R.

- The exposed methods are:

  - setCompositionData

  - setSimplexData

  - runAnalysis

  - getResults

- This allows R users to create and manipulate DirichletStudyInterface objects and call these methods from R scripts.

- The file acts as a bridge between R and C++ for Dirichlet study analysis.

src/RcppExports.cpp

- RcppExports.cpp is auto-generated by Rcpp and should not be edited by hand.

- It provides the necessary C++ glue code to register the Rcpp module (named ds) with R.

- It declares and exports the function _rcpp_module_boot_ds, which initializes the Rcpp module for the DirichletStudy package.

- The CallEntries array registers the C++ functions that can be called from R, in this case only _rcpp_module_boot_ds.

- The R_init_DirichletStudy function is called when the R package is loaded, registering the routines and setting up dynamic symbol usage.

- This file enables R to call C++ code defined in the DirichletStudy package via the Rcpp interface.

inst/include/dirichlet_composition.hpp

- The dirichlet_composition.hpp header defines a C++ template class DirichletComposition for modeling Dirichlet compositions.

- The class inherits from atl::FunctionMinimizer<T>, suggesting it is used for optimization tasks.

- It defines an enum DirichletCompositionType with options: DEFAULT, LINEAR, and SATURATED, to specify the type of Dirichlet composition.

- The class holds a shared pointer to a FunctorBase<T> object, which likely represents the function to be minimized.

- It provides methods to set the functor, initialize the object, and evaluate the function (though Initialize and Evaluate are currently empty).

- The class is designed to be flexible and extensible for different types of Dirichlet composition models.

- The file includes guards to prevent multiple inclusion and includes dependencies for optimization and random number generation.

inst/include/dirichlet.cpp

- The C++ file dirichlet.cpp implements a main program to run various Dirichlet model analyses on a CSV data file.

- Parses command-line arguments to optionally enable writing values and/or derivatives for each analysis.

- Instantiates five different Dirichlet model classes: Dirichlet_Default, Dirichlet_Thorson, Dirichlet_Fisch, Dirichlet_Linear, and Dirichlet_Saturated, each templated on double and initialized with the same data file.

- For each model:

  - Sets flags to control output (write_values, write_derivatives) based on user input.

  - Calls Initialize(), disables parameter set building, runs Analyze(), and calls Finalize().

- The file path for the data is hardcoded and should be updated for different environments.

- Designed for batch analysis and comparison of multiple Dirichlet model implementations.

inst/include/dirichlet_fa.hpp

- The header dirichlet_fa.hpp declares and implements several C++ template classes for functional analysis of Dirichlet distributions.

- Defines a struct simplex_data to hold simplex (compositional) data, including probabilities and a score.

- Provides an overloaded operator<< for printing vectors.

- Implements Dirichlet_Study_Base, a base class for Dirichlet model analysis:

  - Handles reading and parsing simplex data from CSV files.

  - Provides methods to normalize data and build input values for analysis.

- Implements five derived classes for different Dirichlet models:

  - Dirichlet_Default

  - Dirichlet_Thorson

  - Dirichlet_Fisch

  - Dirichlet_Linear

  - Dirichlet_Saturated

- Each derived class sets up model-specific parameters, normalization, and evaluation logic.

- Uses a common interface for initialization and evaluation, enabling batch analysis and comparison of different Dirichlet models.

- Designed for extensibility and integration with functional analysis and optimization frameworks.

inst/include/dirichlet.hpp

- The header dirichlet.hpp declares mathematical functions and types for working with Dirichlet and Dirichlet-multinomial distributions in C++.

- Implements a custom lgamma_lanczos function for computing the log-gamma function using the Lanczos approximation.

- Defines an enum class DirichletType to distinguish between different parameterizations: THORSON, FISCHER, LINEAR, SATURATED, and DEFAULT.

- Provides template functions to compute the log-likelihood of the Dirichlet-multinomial distribution under various parameterizations:

  - log_dirichlet_multinom (default)

  - log_dirichlet_multinom_thorson

  - log_dirichlet_multinom_fisch

  - log_dirichlet_multinom_linear

  - log_dirichlet_multinom_saturated

- Implements a generic ddirichlet_multinom function that dispatches to the appropriate log-likelihood calculation based on DirichletType.

- Supports both log and non-log (exponentiated) probability outputs.

- Uses standard C++ libraries for vector operations, numeric accumulation, and math functions.

- Designed for extensibility and integration into statistical modeling or analysis codebases.

inst/include/functional_analysis.hpp

- The header functional_analysis.hpp declares a C++ template class FunctionalAnalysis for analyzing mathematical functions, especially in the context of optimization and sensitivity analysis.

- Provides data structures to store parameters, parameter sets, input values, function values, derivatives, covariance, and correlation matrices.

- Implements methods for:

  - Initializing and clearing analysis data.

- Registering parameters and setting up parameter sets for analysis.

- Running the analysis (Analyze), which evaluates the function over parameter sets, computes derivatives, and tracks min/max values.

- Calculating and writing out function values and derivatives to files.

- Computing statistical summaries: mean, standard deviation, covariance, and correlation.

- Building parameter sets via combinatorial logic.

- Progress reporting during analysis.

- Requires derived classes to implement Initialize() and Evaluate() methods for specific function analysis.

- Integrates with an automatic differentiation library (ATL) for derivative and Hessian calculations.

- Designed for extensibility and reuse in computational experiments involving function evaluation and sensitivity analysis.

inst/include/rcpp_interface.hpp

- The header rcpp_interface.hpp declares the DirichletStudyInterface C++ class to provide an interface between R and C++ using Rcpp.

- Allows R users to set composition data and simplex data via setCompositionData and setSimplexData methods, accepting Rcpp::NumericMatrix objects.

- Provides a runAnalysis method as a placeholder for running the Dirichlet analysis (currently returns true without computation).

- Provides a getResults method as a placeholder to return analysis results to R (currently returns a simple status list).

- Stores the input data as private Rcpp::NumericMatrix members.

- Designed to be exposed to R via Rcpp modules, enabling integration of C++ Dirichlet analysis code with R workflows.

- Includes header guards to prevent multiple inclusion.

inst/include/functors/dirichlet_default.hpp

- The header dirichlet_default.hpp declares a C++ template class DirichletDefault for use as a functor in Dirichlet model analysis.

- Inherits from FunctorBase<Type>, indicating it is intended to be used as a callable object for function evaluation.

- Provides empty (placeholder) implementations for the Initialize and Evaluate methods, which are meant to be overridden or filled in with model-specific logic.

- Uses include guards to prevent multiple inclusion.

- Designed as a base or template for implementing the default Dirichlet functor logic in a modular analysis framework.

inst/include/functors/dirichlet_linear.hpp

- The header dirichlet_linear.hpp declares a C++ template class DirichletLinear for use as a functor in Dirichlet model analysis.

- Inherits from FunctorBase<Type>, making it suitable for use in function evaluation and optimization frameworks.

- Provides empty (placeholder) implementations for the Initialize and Evaluate methods, intended to be filled with logic specific to the linear Dirichlet model.

- Includes a header guard to prevent multiple inclusion.

- Designed as a modular component for implementing linear Dirichlet model functionality within a larger analysis framework.

inst/include/functors/dirichlet_saturated.hpp

- The header dirichlet_saturated.hpp declares a C++ template class DirichletSaturated for use as a functor in Dirichlet model analysis.

- Inherits from FunctorBase<Type>, making it compatible with function evaluation and optimization frameworks.

- Provides empty (placeholder) implementations for the Initialize and Evaluate methods, to be filled with logic specific to the saturated Dirichlet model.

- Includes a header guard to prevent multiple inclusion.

- Designed as a modular component for implementing saturated Dirichlet model functionality within a larger analysis framework.

inst/include/functor/functor_base.hpp

- The header functor_base.hpp declares a C++ template abstract base class FunctorBase for use in function evaluation frameworks.

- Requires derived classes to implement two pure virtual methods: Initialize() and Evaluate().

- Provides a virtual destructor for safe polymorphic use.

- Defines an operator() that calls Evaluate(), allowing objects to be used as callable functors.

- Includes a header guard to prevent multiple inclusion.

- Designed to be inherited by specific functor classes (e.g., DirichletDefault, DirichletLinear, DirichletSaturated) for modular and extensible function evaluation.