PersonAlytics Software Design Document

Stephen Tueller

2018-08-22

**SDD Revisions**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Description | Revision | Editor |
| 20180820 | Document Creation | 0 | Stephen Tueller |

# Introduction

## Purpose

This software design document (SDD) describes the architecture and system design of the PersonAlytics R package. The PersonAlytics package provides analytic tools for single-case, small N, and Idiographic Clinical Trials (ICT) using mixed effect models building the [nlme R package] (<https://cran.r-project.org/web/packages/nlme/index.html>) and the [gamlss R package](https://www.gamlss.com/).

## Scope

This document descrbes the implementation details of the PersonAlytics R package which automates sereval steps of the mixed effects model fitting process for single-case, small N, and ICT data. PersonAlytics does not make decision reccomendations based on its output, and the user assumes all responsibility for data quality and interpretation of results.

This documents assumes that the reader

* Has [R](https://cran.r-project.org/) installed on their system (R operates on Windows, Linux, and Apple systems, and the PersonAlytics source that will be disrtributed to users under the RTI License is platform independent).
* Has installed the required dependencies (see the “PersonAlytics Design Diagram” below).
* Has PersonAlytics installed on their system. In R, type install.packages("path/to/file/PersonAlytics\_0.0.1.tar.gz", repos = NULL, type = "source") into the console replacing path/to/file/ with the directory in which PersonAlytics\_0.0.1.tar.gz is saved. Note that R uses forward slashes even when used in a Windows environment.
* Has R running on their system and PersonAlytics loaded by typing library(PersonAlytics) into the console. This will provide the reader with access to the R help file system, which is accessed by preceding a function name with a question mark. For example, typing ?PersonAlytics into the R console will open the documentation overview.
* It is suggested that the reader use [Rstudio](https://www.rstudio.com/), which has a dedicated pane for help files. Example code within the help pane can be run by highlighting the text and pressing Ctrl+Enter.

## Aim

The design description defined in this document serves the following purposes:

* To describe the required dependencies in R.
* To describe the required data structure being read into R and passed to PersonAlytics functions.
* To describe the algorithms to be implemented.
* To assist in the production of test cases, including
  + verification of results against direct calls to their underlying dependencies
  + validation of results against those obtained from **SAS**
* To be used to verify compliance with requirements.
* To aid in maintenance activities.

## Intended Audience

The intendend audience for this document is

* The RTI development and testing team
* Auditors and reviewers

## People

## Clients

PersonAlytics is being produced by RTI to for project clients who will be users of the software, but who are not directly driving the production of PersonAylitics'. Funding to developPersonAlytics` is being pursued, and those clients will be listed here as the project matures.

## Team

PersonAlytics was conceptualized by Ty Ridenour ([tridenour@rti.org](mailto:tridenour@rti.org)), developed by Stephen Tueller ([stueller@rti.org](mailto:stueller@rti.org)), and tested by Derek Ramirez ([dramirez@rti.org](mailto:dramirez@rti.org)) and Corina Owens ([cowens@rti.org](mailto:cowens@rti.org)).

## External Reviewers

BlueDoor

## Definitions and Acronyms

In what follows, we distinguish between two types of data. “Data” refers to a rectangular dataframe that the user imports into R. “inputs” refers to options that the user passes to the functions they interact with directly, namely PersonAlytic and PersonAlyticHTP.

|  |  |
| --- | --- |
| Acronym | Definition |
| ICT | Idiographic Clinical Trial |
| OOO | Object Oriented Programing |
| FDR | False Discovery Rate |
| AR | Autoregressive |
| MA | Moving Average |
| ARMA | Autoregressive Moving Average |
| lme | Linear Mixed Effects |
| p | the AR parameter |
| q | the MA parameter |

# System Overview

The system overview is given in details of the help file for the PersonAyltics package, type ?PersonAyltics in R after installing the required dependencies, installing PersonAylytics, and loading the package using library(PersonAylytics).

**more background here by Ty**

# System Architecture

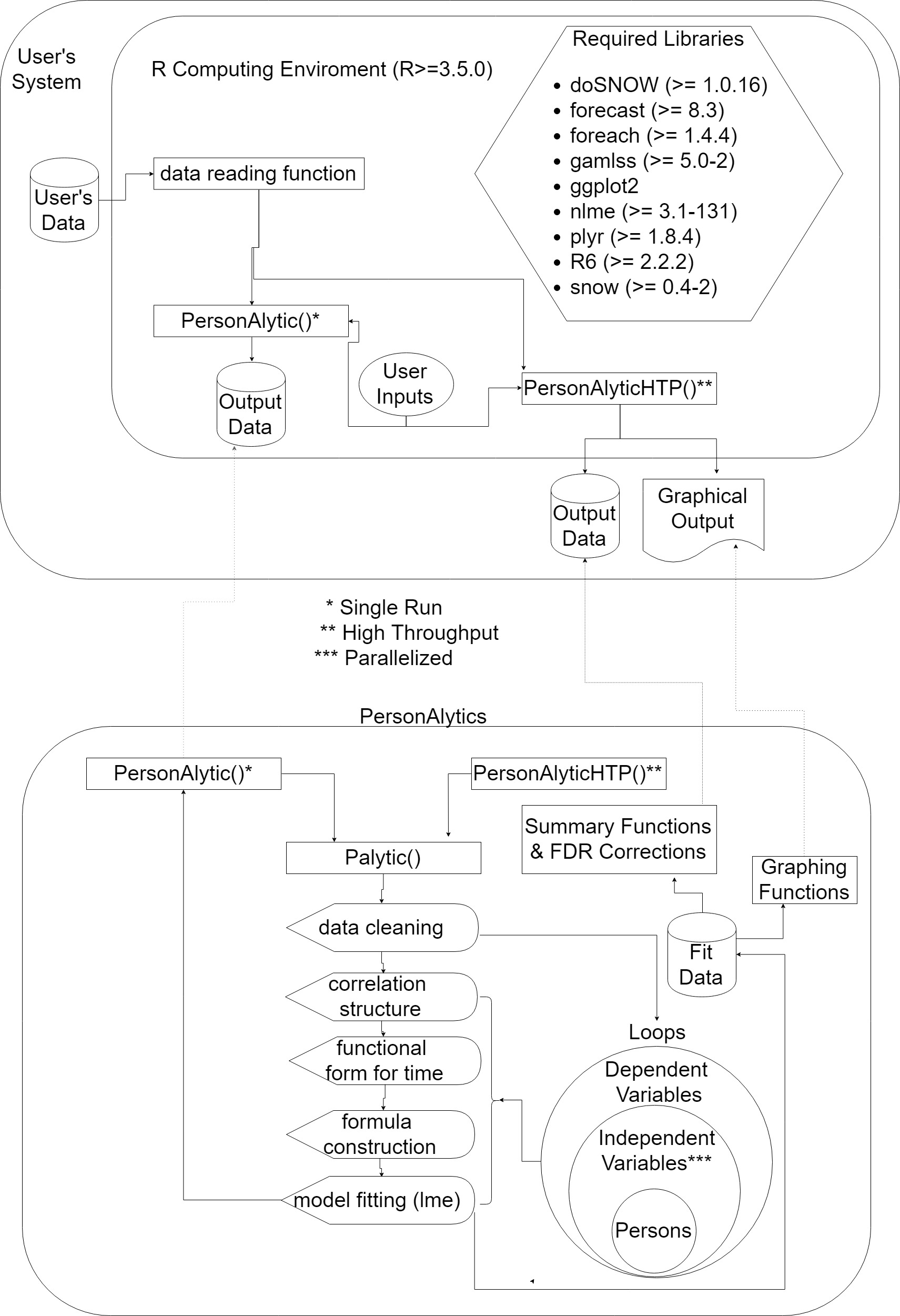
## Architectural Design

The PersonAlytics Design Diagram shows how user data, R, and PersonAlytics interact on the user’s system. In the diagram,

* Cylinders represent data
* Rectangles represent functions implemented in a functional programming framework
* Cicles desginate critical loops within a function
* Left-arrowed shapes indicate methods for Palytic() class objects implemented in the R6 OOP framework

Overview of the architecture:

* PersonAlytics functions are utilized within an R session.
* The user’s data may exist outside of R, and if so, must be read into the current R session. This can be done using any R function appropriate to the reading of a user’s data (e.g., the native read.csv() function or specialized functions in the [haven](https://cran.r-project.org/web/packages/haven/index.html) package). The end requirement is that data are in a format with
  + as many rows per participants as there are repeated measures per participant
  + time is ordered correctly within each participant
* Once the data are in R, the user will load PersonAlytics using library(PersonAlytics). This will load all the required depedencies (see the hexagon) assuming they are installed. If they are not, an R (not PersonAlytics) error will appear stating which depedencies are missing. This can be rectified by using, for example install.packages("gamlss"). This process should be repeated until all depednencies are installed.
* The user will then call PersonAlytic() or PersonAlyticHTP() with their desired inputs. Inputs are described in detail in these function’s help files accessed using ?PersonAlytic and ?PersonAlyticHTP in the R console.
* Inputs used to create a Palytic class object, which checks data, checks inputs, creates additional required formulae, and fits the requested model. For the PersonAlyticHTP() function, this process is looped over multiple depedent variables, independent variables, and individual participants as requested.
* PersonAlytic() returns output to the R session for the user to view or save.
* PersonAlyticHTP() saves output in the form of text, csv, and pdf files in a directory specified by the user.



PersonAlytics Design Diagram

### PersonAlytic()

**Purpose:** Primary user interface for running one model with one outcome for one subset of the data.

**Inputs:** See *Usage* and *Arugments* in ?PersonAlytic in R.

**Outputs:** See *Value* in ?PersonAlytic in R.

**Called by:** The user in an R session where PersonAlytics and its required dependencies are loaded.

**Algorithm:** -

* Create Palytic object
* If requested, automatically detect the (residual) correlation structure
* If requested, automatically detect the polynomial relationship between time and the dependent variable
* Fit the model implied by the user inputs using using nlme

### Palytic Class Generator

**Purpose:** Store and validate data and user inputs

**Inputs:** See *Usage* and *Arugments* in ?Palytic in R.

**Outputs:** Outputs are saved back to the Palytic object which are extracted by Palytic methods or the PersonAlytic(), PersonAlyticHTP(), or htp.foreach() functions.

**Methods:** See *Methods* in ?Palytic in R. These include -

* lme for fitting linear mixed effects models to the requested subset of data:
  + Try fitting the model with default settings, if that fails
  + Try fitting the model with using the optim optimizer, if that fails
  + Try fitting the model without the random slopes or residual correlation structure, if that fails
  + Return that the “Model did not converge”
* gamlss for fitting generalized additive models for location, scale and shape to the requested subset of data:
  + Try fitting the model with default settings, if that fails
  + Try fitting the model with additional cycles, if that fails
  + Try fitting the model without the random slopes, if that fails
  + Return that the “Model did not converge”
* getAR\_order and GroupAR\_order for automatically detecting the ARMA() order for all participants or for the full sample, respectively:
  + Check whether time is equally spaced, if yes
    - Use the auto.arima function of the [forecast](https://cran.r-project.org/web/packages/forecast/index.html) package to detect the best fitting ARMA model for autocerrelation.
  + If time is not equally spaced, fit lme models up to the maximum specified p and q for ARMA(p,q) and compare models using either LRT or IC as requested.
  + Set the correlation or corStructs values of the Palytic object.
* getTime\_Power and GroupTime\_power for automatically detecting the polynomial for the relationship between time and the dependent variable for all participants or for the full sample, respectively:
  + Fit lme models up the largest power of time specified by the user.
  + Set the time\_power or time\_powers values of the Palytic object.

**Called by:** The PersonAlytic(), PersonAlyticHTP(), or htp.foreach() functions.

**Active bindings:** Any time a Palytic object is updated (e.g., when iterating over depedent variables), data and inputs are revalidated and formulae are updated.

**Algorithm:** -

* Clean and validate data.
* Create formulae implied by the user inputs to PersonAlytic() or PersonAlyticHTP().
* Check whether time is monotonically increasing.
* Create the Palatyic class object.

### PersonAlyticHTP()

**Purpose:** Primary user interface for running models iterating across multiple outcomes (dependent variables), target independent variables, or individuals.

**Inputs:** See *Fields* in ?PersonAlyticHTP in R.

**Outputs:** See *Value* in ?PersonAlyticHTP in R.

**Called by:** The user in an R session where PersonAlytics and its required dependencies are loaded.

**Algorithm:** -

* Pre-check data and inputs
* Create iterator lists for dependent variables, independent variables, and individuals
* Send inputs, data, and iterator lists to htp.foreach()
* Post-estimation output checks
* Write output data to a csv file
* If requested, apply FDR corrections using psuite()

### htp.foreach()

**Purpose:** Function for running models iterating across multiple outcomes (dependent variables), target independent variables, or individuals.

**Inputs:** See *Fields* in ?PersonAlyticHTP in R.

**Outputs:** See *Value* in ?PersonAlyticHTP in R.

**Called by:** PersonAlyticsHTP.

**Algorithm:** -

* Start outer loop for dependent variables
* Set up NULL Palytic object
* Call getAR\_order() or GroupAR\_order() (see Palytic methods above).
* Call getTime\_power() or GroupTime\_power() (see Palytic methods above).
* Initialize parallelization.
* Start parallelized middle loop for target independent variables.
* Copy the NULL Palytic object and add the target independent variable.
* Start the inner loop for participant
  + Determine the subset of data corresponding to the current participant.
  + Extract identifying inputs for labeling outputs.
  + Participant level data validations.
  + Fit the model to the data (see the lme and gamlss methods for Palytic objects).
  + Accumulate model outputs.
* Restructure outputs across the three loops.
* Return outputs as a rectangular data frame to PersonAlyticsHTP().

## Design Rationale

Combining functional programming and OOP was intentional. Many intendend users of PersonAlytics will be R novices and most basic advice on using R is design on showing users how to interact with functions. Hence the user interfacing functions PersonAlytic() and PersonAlyticHTP() are design to follow this expectation. Although all single analyses can be run using PersonAlyticHTP(), it invokes computational overhead for parralellizing R which is avoided using PersonAlytic().

The underlying Palytic class was designed in the R6 OOP framework which is ideal for combining user inputs, data and input validation, active bindings, and methods in an extensible framework.

# Data Design

## Data Description

Users import rectangular research data into R and provide their inputs to PersonAlytic or PersonAlyticHTP.

## Output Data

The PersonAlytic function returns an lme or gamlss object to the R console. The PersonAlyticHTP function saves a csv file where each row represents a unique combination of dependent variable, target indepent variable, and individuals. The file has the following columns:

* dv: The name of the dependent variable.
* ivs: The names of the non-target independent variables (i.e., those that do not change across rows).
* target\_iv: the name of the target independent variable (i.e., the independent variable that can change across rows).
* fixed: the fixed effects model formula used by lme.
* random: the random effects model formula used by lme.
* correlation: the residual correlation structure used by lme.
* formula: the fixed, random, and correlation formulae used by gamlss.
* directory: the working director for the R session in which the cvs file is saved.
* date: the date and time in yyyy-mm-dd h:mm:ss format
* *ids*: the numeric values of the user input ids.
* ids: the character value of the user variable names assigned to ids.
* ivsl: redundant with target\_iv, depricate.
* time\_power: the value of the time\_power variable.
* Nobs: the number of observations in the analysis.
* dvVar: the variance of the dependent variable.
* ivVars: flags for non-zero variance of the independent variables.
* converge: convergence status.
* [additional columns with parameters estimates, standard errors, degrees of freedom, t-values, and p-values; column labels depend on the analysis and model].

# Model Verification

# Model Validation