Shiny Interface to the RobStatTM Library

Gregory Brownson

August 13, 2018

Abstract

This vignette explains the use of the Shiny User Interface to the RobStatTM R package that was implemented during the 2018 Google Summer of Code (GSoC). GSoC funding was awarded to UW AMath MS-CFRM student Gregory Brownson as project lead, with Doug Martin and Matias Salibian-Barrera as mentors for the project.

1 Introduction

Overall Goal

Our overall goal in developing a Shiny user interface (UI) to the **RobStatTM** package (ShinyRobStatTM) companion to the **Robust Statistics: Theory and Methods** book (Maronna et al. 2018 second edition). In particular, the Shiny UI broadens the applications of the package by allowing users to work with robust statistical methods without first having to learn to use the R command line interface. The Shiny point-and-click interface substantially lowers the learning curve for computing robust statistics and we believe will thereby result in an increased frequency in use of these methods.

Interface with RobStatTM

The shiny application provides an interface with **RobStatTM** without needing to set many R function arguments or understanding the complexities of the underlying functions. The simplicity of the UI will allow students and practitioners who are new to robust methods to easily use them and compare robust results with classical methods. Within the Shiny UI, the user will be able to load a data set either from an existing R package or a local .csv file and then run the data through different methods of analysis. Currently, the UI provides an interface to robust location and scale, robust linear regression, robust covariance estimation, and robust principal component analysis. More methods will be added in the future as the **RobStatTM** package development continues.

Package Dependencies

The Shiny UI depends on the following packages:

- **DT**
- **fit.models**
- **ggplot2**
- **grid**
- **gridExtra**
- **PerformanceAnalytics**
- **robust**
- **robustbase**
- **shiny**
- **shinyjs**
- **xts**

The Shiny application will detect any missing packages and install them automatically, removing the responsibility of package management from the user. Now that the Shiny UI background has been established, the subsequent sections will explain how to use the UI.

2 Loading Data

Data may be loaded into the Shiny UI from an existing R package or upload a csv file to the application by selecting either "R Package" or "Upload" as shown in Figure 1. Once a user selects a package from the list under **Library Name**, the **Select Dataset** input is populated by existing names of datasets in that package. After selecting a dataset, the user should click on the **Load Data** button to load the data into the application for further analysis.

Figure 1: Interface to load a data set from an existing R package

Figure 2 shows the options for uploading a csv file to the application. The user must specify the file format such as the delimiter and string quotation syntax. If the data is a time series, then the user must check the corresponding box as shown in the example. The first column of data must contain the date or time indices to be read as a time series. In the example, we use the data set **hfunds.ts.csv** which contains returns for five hedge funds and upload it as a time series. The window in the main (right) panel displays the data in a table like format and we may view the returns for each date. There are five columns corresponding to the different types of hedge funds: emerging markets (**EM**), Private Equity (**PE**), U.S. High Yield (**USHY**), Alternative Investments (**AI**), and Bond (**BND**)**.** For datasets with a large number of variables, we suggest viewing the application in **Full Screen**.

Figure 2: Interface to upload a csv file from a user's computer to the application

3 Robust Location-Scale

This section allows the comparison of classical estimates for mean and standard deviation and robust estimates for location and scale for a single variable.

Computing Location and Scale

Estimating the location and scale of a variable from a given data set is simple in the Shiny UI. Simply select a **Variable** and choose a **Method** from the options of *Classical*, *Robust*, or *Both*. The classical and robust methods are compared when *Both* is chosen. Furthermore, if either *Robust* or *Both* are selected, then a set of options for the robust method appear in the bottom of the side panel on the left where the user may select the **Score Function** and **Asymptotic Efficiency**. To compute the estimates and display the summary, click on the **Results** button. Robust estimates are computed using the **locScaleM()** function in **RobStatTM**. Figure 3 shows an example comparing the estimates of location and scale for the Private Equity fund from the **hfunds.ts** data where we used the bi-square score function and an asymptotic efficiency of 90%.

Figure 3: Classical and robust location and scale comparison for Private Equity fund in **hfunds.ts**

Note that the results are displayed in the main panel on the right. This is the layout for each component in the Shiny UI. For location and scale, the summary displays estimates for location, the standard error of the location estimate, and the estimate for scale. In the future, we plan to allow the user to compute just the scale estimate.

4 Robust Linear Regression

This section allows a simple computation and comparison of classical and robust linear regression methods.

Computing Linear Models

There are four methods to choose from in the UI: least-squares (LS), M, MM, and distance constrained maximum-likelihood (DCML). Users have the option to compare models by checking the **Add Second Method** box and selecting inputs for a second model. In Figure 4, the classical least-squares and robust MM methods are being compared. Dependent and independent variables must be selected for the model and the regression formula is automatically populated in the text box. When comparing regressions, the formulas do not need to be identical. To fit the models and view the summary, click the **Results** button at the bottom of the options side-panel.

Figure 4: Least-squares and robust MM regression comparison using **mineral** data from **RobStatTM**

The standard errors, t-statistics, and p-vlues for the robust coefficients for robust fits are computed using a robust covariance matrix for the independent variables, as an important step to ensure that those quantities are robust themselves. Furthermore, the *proportion of variance explained by the model*, or R^2, for robust fits is a robust version of

classical least-squares R^2. In the example, we fit the formula *zinc ~ copper* for the LS and MM estimators while using the modified optimal value for psi with asymptotic efficiency of 99%. The equivalent command-line code is

```
# Fit least-squares linear regression
fit.ls <- lm(zinc ~ copper, data = mineral)

# Fit robust linear regression
fit.mm <- lmrobdetMM(zinc ~ copper, data = mineral,
                     control = lmrobdet.control(family = "modified.optimal",
                                                eff    = 0.96))

# Send fits to fit.models
fm <- fit.models(LS = fit.ls, MM = fit.mm)

# Comparison of results
print(summary(fm))
```

Plots

Once fitting a model(s), results may be visualized in the **Plots** tab. In the side-panel, the user may choose from the following selection: *Residuals v. Fit*, *Response v. Fit*, *Residuals Normal QQ Plot*, *Std. Residuals v. Robust Distances*, *Estimated Residual Density*, *Std. Residuals v. Index (Time)*, and *Scatter with Overlaid Fits*. Note that the *Scatter with Overlaid Fits* plot is only available in the case of two univariate regressions. The following are descriptions and examples of some plots:

Normal QQ-Plots of Residuals

Figure 5: LS and Robust MM Normal QQ-Plots of Residuals: **mineral**

Probability Density Estimates of Residuals

Figure 6: Probability Density Estimates of LS and Robust MM Residuals: **mineral**

Standardized Residuals versus Robust Distances

Figure 7: LS and Robust MM Plots of Standardized Residuals versus Mahalanobis Distances: **mineral**

Standardized Residuals versus Index (Time)

Figure 8: Indexed LS and robust MM Standardized Residuals: **mineral**

Scatterplots with Overlaid Fits

Figure 9: Scatterplot of data with overlaid LS and robust MM fits: **mineral**

5 Robust Covariance

This section allows a simple computation and comparison of classical and robust covariance/correlation estimation.

Computing Robust Covariance

Estimating robust and/or classical covariance/correlation is easy with the Shiny UI since the UI provides a simple interface with the **RobStatTM** functions **covClassic()** and **covRob()**. First off, since the methods used in **RobStatTM** only allow for numeric data, the options for **Variables** are limited to those with numeric values only. The user may choose to compute classical estimates of covariance/correlation, robust estimates of covariance/correlation, or both using the **Method** option. If *Robust*, or *Both* is chosen, then an option to select the **Robust Covariance Estimator** is available. Users may select *MM*, *Rocke*, or *Auto* as the estimator. If *Auto* is selected, then a Rocke-estimator is used if the number of variables is greater than or equal to 10, and an MM-estimator is used otherwise. Lastly, may select whether to compute the covariance matrix or correlation matrix under the **Type** input. Figure 10 displays the comparison of classical and robust covariance estimates for the **hfunds.ts** data.

Figure 10: Classical and robust covariance estimates for funds in **hfunds.ts** data

In the example above, we may see a comparison of the classical and robust estimates for covariance/correlation terms, location, scale, and eigenvalues. A command-line equivalent to the example in Figure 10 is shown below

```
# Use fit.models to estimate centers and covariance matrices
cov.fm <- fit.models(classic = covClassic(hfunds.ts),
                     robust  = covRob(hfunds.ts, type = "auto"))

# Comparison of results
print(summary(cov.fm))
```

Plots

Eigenvalues

Figure 11: Scree plot: **hfunds.ts**

Mahalanobis Distances

Figure 12: Mahalanobis Distances: **hfunds.ts**

Ellipses Matrix

Figure 13: Ellipses Matrices and correlation estimates: **hfunds.ts**

Distances Chi-Squared QQ Plot

Figure 14: Classical and robust Mahalanobis Distances Chi-Squared QQ-Plot: **hfunds.ts**

Distance-Distance scatterplot

Figure 15: Scatterplot of classical and robust Mahalanobis Distances: **hfunds.ts**

6 Robust PCA

This section allows a simple computation and comparison of classical and robust principal component analysis.

Computing Robust Principal Components

The options for computing principal components are similar to the options for computing covariance. Classical principal components are computed using **prcomp()** and robust principal components are computed through an eigen-decomposition of the robust covariance/correlation matrix. Users must select at least two **Variables** for analysis. If the option "Both" is selected for **Method**, then the classical and robust principal components will be computed and compared. The selection for **Type** determines whether the robust performance components are computed from the robust covariance or robust correlation matrix. Since robust principal components are computed from the robust covariance/correlation matrix, the **Robust PCA Estimator** serves the same purpose as in the **Robust Covariance** section. Figure 16 contains an example comparing classical and robust principal components of the **hfunds.ts** data.

Figure 16: Classical and robust estimates for principal components in **hfunds.ts** data

The command-line equivalent to the example from Figure 16 is:

```
# Use fit.models to estimate principal components
pca.fm <- fit.models(classic = prcomp(hfunds.ts), robust = princompRob(hfunds.ts))

# Comparison of results
print(summary(pca.fm))
```

Plots

Plots for principal component analysis are not available in the current version, but will be added in the future.

7 What's Next?

Improvements to the User Interface

The Shiny UI will continue to evolve as we recieve more feedback on the interface and consider new ideas for it. Future versions will generally contain small aesthetic adjustments and not a major overhaul of the current interface.

Additional Components

Future versions will also contain additional robust methods. Currently, we plan to add interfaces to robust logistic regression and robust time series analysis as seen in Chapters 7 and 8 of the book.