Package `ConfidenceQuant`: a vignette

10 June 2018

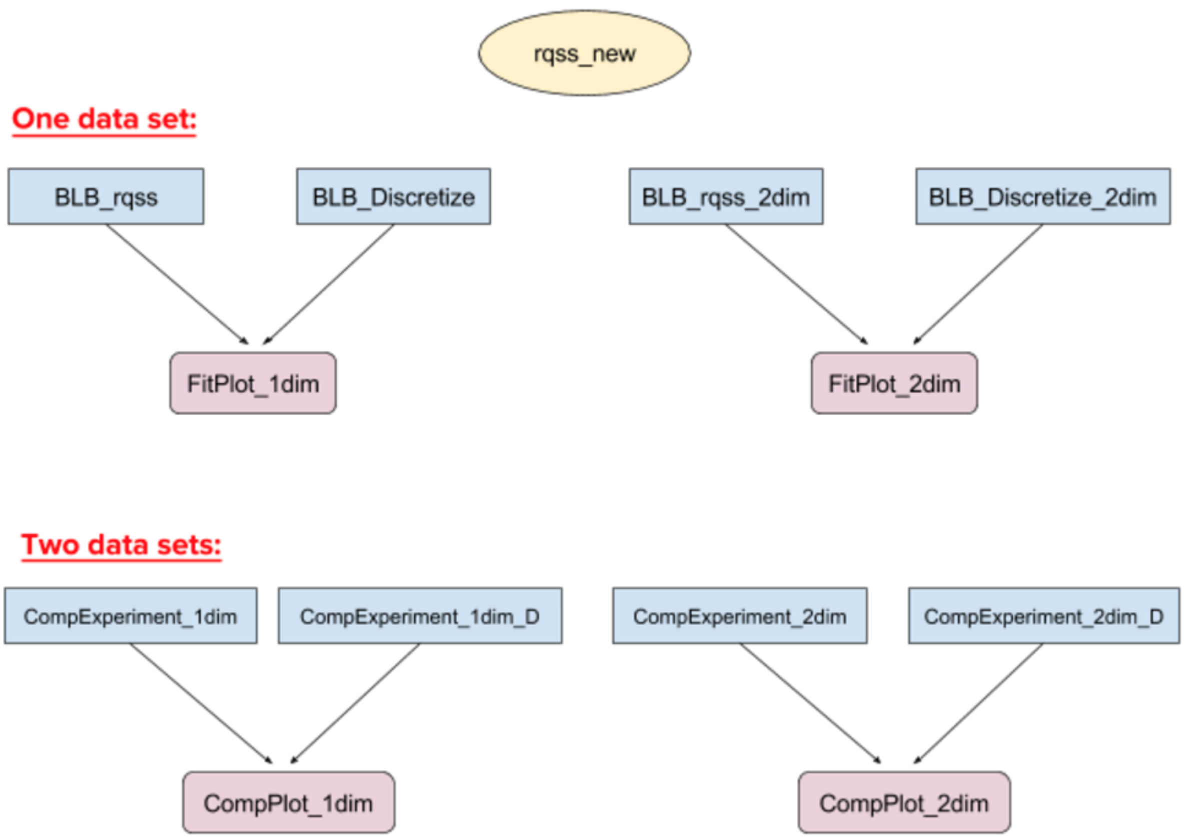


Figure 1: Package Framework

0. Assign number of cores for computing

```
cores<-15
```

We used `doParallel` and `foreach` to parallelize the computation.

When using the package, please make the last column of the data frame input as response.

1. Read in the data set

For demonstration purposes, we use a data set which contains two covariates and one response for 50,000 observations.

Also, `cell` is a cell indicator. This data set contains **two** cells.

Use `help(Vignette_data)` to know more about the data set.

```
library(ConfidenceQuant)
data(Vignette_data)

library(dplyr) #To return rows with matching conditions
control<-filter(Vignette_data,cell==1)[,-1]
treatment<-filter(Vignette_data,cell==2)[,-1]
```

Now we have two data sets that have 3 variables, with the last column being the response.

2. One Dependent Variable

We use 2-dim data frame as input.

```
control_1dim<-control[,c(1,3)]
treatment_1dim<-treatment[,c(1,3)]
```

2.1 One Cell (Without Comparison)

A. Get the confidence bands

We can use `BLB_rqss` or `BLB_Discretize` to get the confidence bands.

△ The optimum smoothing parameter λ can be selected **automatically**.

```
result<-BLB_rqss(cores = cores, data = control_1dim, alpha = 0.05, tau = 0.5, Search = TRUE)
result_D<-BLB_Discretize(cores = cores, data = control_1dim, alpha = 0.05, tau = 0.5, Search = TRUE)
```

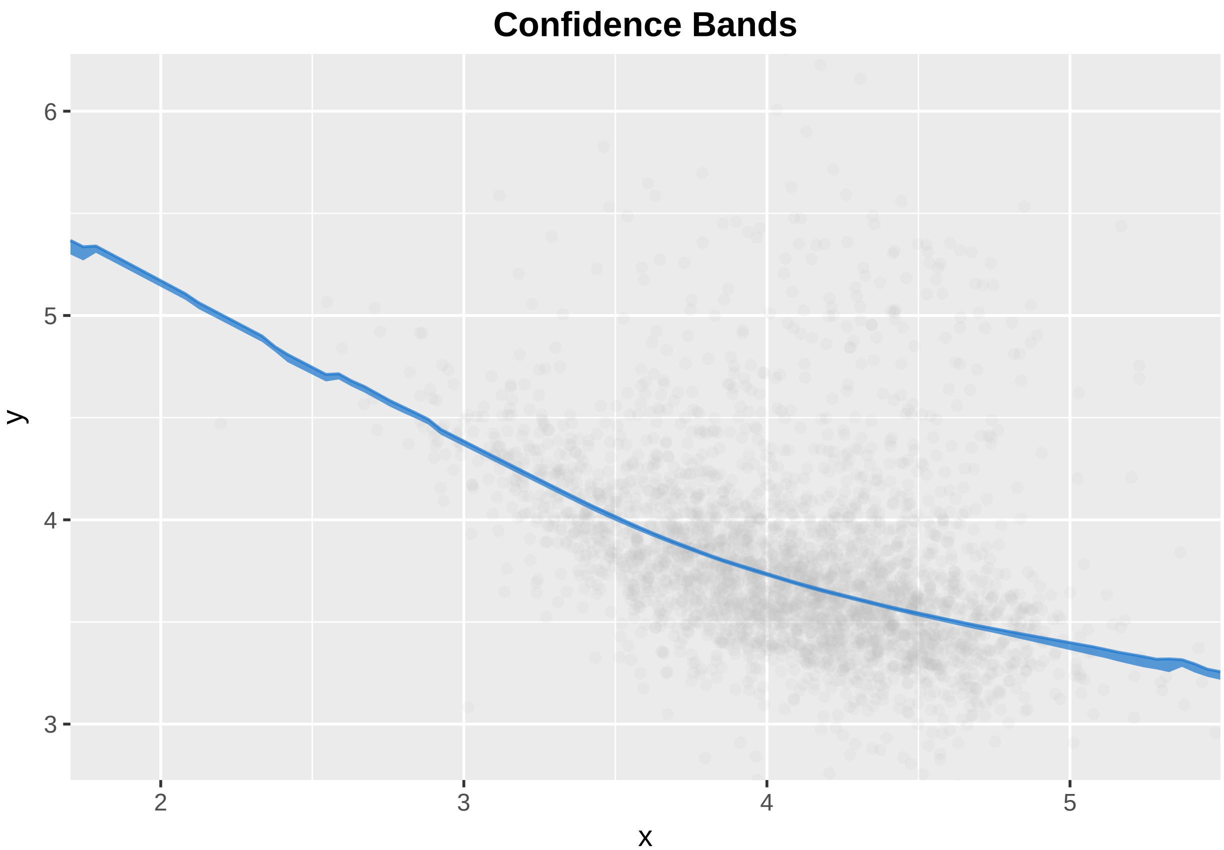
△ Or the user can **specify** some λ values to choose from.

```
result<-BLB_rqss(cores = cores, data = control_1dim, alpha = 0.05,
                 tau = 0.5, lambda = c(10,120,1000))
result_D<-BLB_Discretize(cores = cores, data = control_1dim, alpha = 0.05,
                         tau = 0.5, lambda = c(10,120,1000))
```

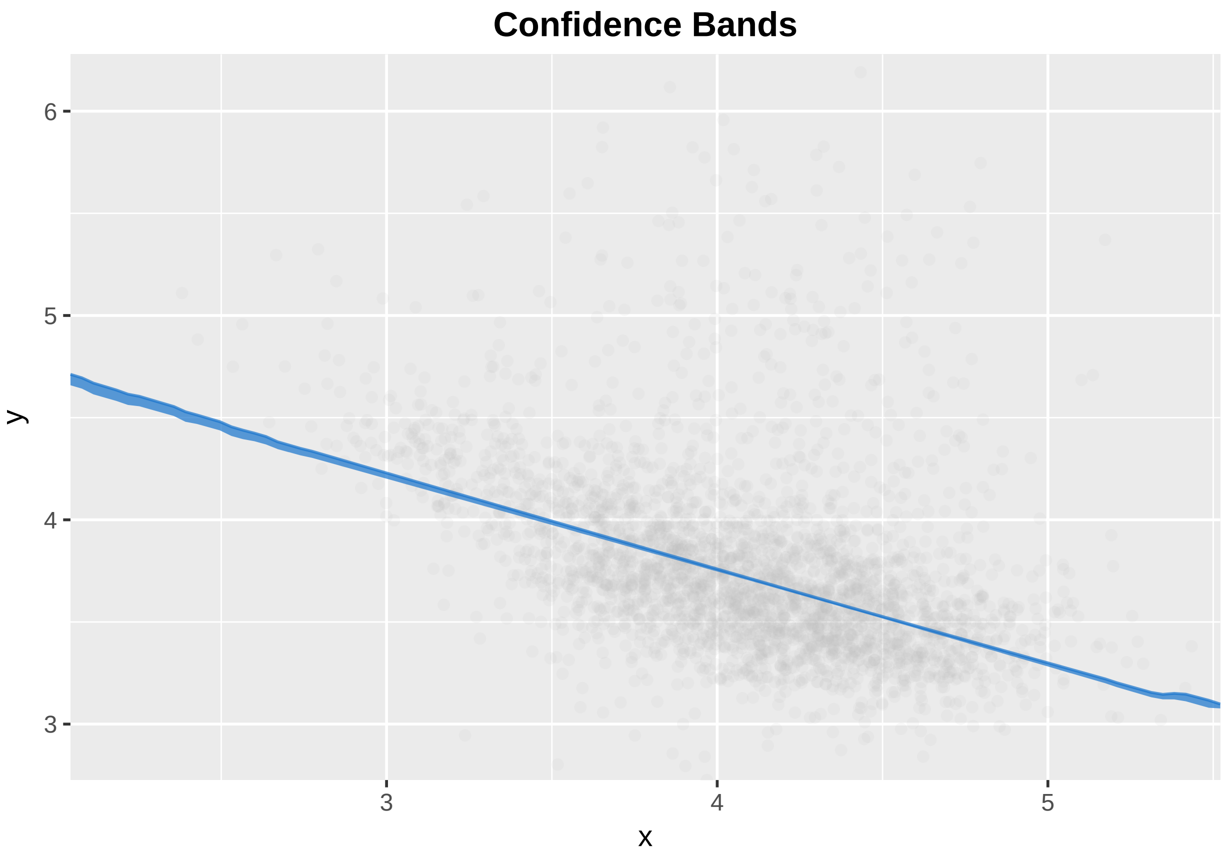
B. Plot the results

We use `FitPlot_1dim` to visualize the results.

```
plot<-FitPlot_1dim(data=control_1dim, result=result, xlab='x',
                   ylab='y')
plot_D<-FitPlot_1dim(data=control_1dim, result=result_D, xlab='x',
                     ylab='y')
plot
```




```
plot_D
```

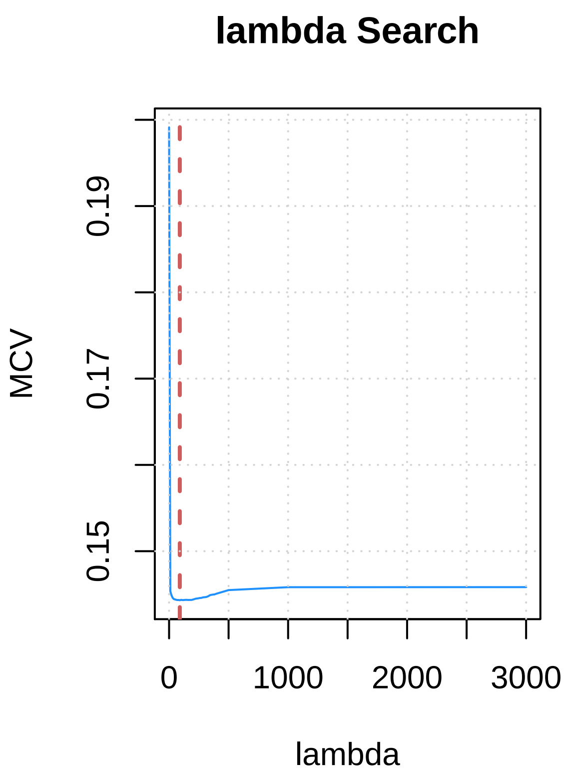


C. (Optional) How λ is selected

If the user is interested in how the optimum λ is selected, they can plot the MCV values from cross-validation in the following way:

```
#For BLB
plot(result$Lambda, result$Fid, type='l',
     xlab='lambda',ylab='MCV',main='BLB_rqss',col='dodgerblue')
abline(v=result$lambda, lty=2, col="indianred", lwd=2)
grid()

#For Discretized
plot(result_D$Lambda, result_D$Fid, type='l',
     xlab='lambda',ylab='MCV',main='BLB_Discretize',col='dodgerblue')
abline(v=result_D$lambda, lty=2, col="indianred", lwd=2)
grid()
```



2.2 Two Cells (With Comparison)

A. Get the confidence bands

We can use `CompExperiment_1dim` or `CompExperiment_1dim_D` to get the confidence bands for **two** data sets.

△ The optimum smoothing parameter λ can be selected **automatically**.

```
all<-CompExperiment_1dim(cores = cores, treatment = treatment_1dim, control = control_1dim,
                         alpha = 0.05, tau = 0.5, Search = TRUE)
all_D<-CompExperiment_1dim_D(cores = cores, treatment = treatment_1dim, control = control_1dim,
                             alpha = 0.05, tau = 0.5, Search = TRUE)
```

△ Or the user can **specify** some λ values to choose from.

```
all<-CompExperiment_1dim(cores = cores, treatment = treatment_1dim, control = control_1dim,
                         alpha = 0.05, tau = 0.5, lambda = c(10,120,1000))
all_D<-CompExperiment_1dim_D(cores = cores, treatment = treatment_1dim, control = control_1dim,
                             alpha = 0.05, tau = 0.5, lambda = c(10,120,1000))
```

B. Plot the results

We use `CompPlot_1dim` to visualize the results.

```
plot<-CompPlot_1dim(treatment = treatment_1dim, control=control_1dim, all = all,
                    xlab='x',ylab='y')
plot$h
plot$g
```

```
plot_D<-CompPlot_1dim(treatment = treatment_1dim, control=control_1dim, all = all_D,
                      xlab='x',ylab='y')
plot_D$h
plot_D$g
```

C. (Optional) How λ is selected

If the user is interested in how the optimum λ is selected, they can plot the MCV values from cross-validation in the following way:

```
#For BLB
plot(all$result1$Lambda, all$result1$Fid, type='l', xlab='lambda', ylab='MCV',
     main='lambda Search', col="dodgerblue")
abline(v=all$result1$lambda, lty=2, col="indianred", lwd=2)
grid()

#For Discretized
plot(all_D$result1$Lambda, all_D$result1$Fid, type='l', xlab='lambda', ylab='MCV',
     main='lambda Search', col="dodgerblue")
abline(v=all_D$result1$lambda, lty=2, col="indianred", lwd=2)
grid()
```

3. Two Dependent Variables

We use 3-dim data frame as input.

```
str(control)
str(treatment)
```

2.1 One Cell (Without Comparison)

A. Get the confidence bands

We can use `BLB_rqss_2dim` or `BLB_Discretize_2dim` to get the confidence bands.

△ The optimum smoothing parameter λ can be selected **automatically**.

```
result<-BLB_rqss_2dim(cores = cores, data = control, alpha = 0.05, tau = 0.5, Search = TRUE)
result_D<-BLB_Discretize_2dim(cores = cores, data = control, alpha = 0.05, tau = 0.5, Search = TRUE)
```

△ Or the user can **specify** some λ values to choose from.

```
result<-BLB_rqss_2dim(cores = cores, data = control, alpha = 0.05, tau = 0.5, c(10,120,1000))
result_D<-BLB_Discretize_2dim(cores = cores, data = control, alpha = 0.05, tau = 0.5, c(10,120,1000))
```

B. Plot the results

We use `FitPlot_2dim` to visualize the results.

```
plot<-FitPlot_2dim(data=control, result=result, xlab='x',
                   ylab='y', zlab='z')
plot_D<-FitPlot_2dim(data=control, result=result_D, xlab='x',
                     ylab='y', zlab='z')
plot
plot_D
```

C. (Optional) How λ is selected

If the user is interested in how the optimum λ is selected, they can plot the MCV values from cross-validation in the following way:

```
#For BLB
plot(result$Lambda, result$Fid, type='l', xlab='lambda', ylab='MCV',
     main='lambda Search', col="dodgerblue")
abline(v=result$lambda, lty=2, col="indianred", lwd=2)
grid()

#For Discretized
plot(result_D$Lambda, result_D$Fid, type='l',xlab='lambda',ylab='MCV',
     main='lambda Search',col='dodgerblue')
abline(v=result_D$lambda, lty=2, col="indianred", lwd=2)
grid()
```

2.2 Two Cells (With Comparison)

A. Get the confidence bands

We can use `CompExperiment_2dim` or `CompExperiment_2dim_D` to get the confidence bands for **two** data sets.

△ The optimum smoothing parameter λ can be selected **automatically**.

```
all<-CompExperiment_2dim(cores = cores, treatment = treatment, control = control,
                         alpha = 0.05, tau = 0.5, Search = TRUE)
all_D<-CompExperiment_2dim_D(cores = cores, treatment = treatment, control = control,
                             alpha = 0.05, tau = 0.5, Search = TRUE)
```

△ Or the user can **specify** some λ values to choose from.

```
all<-CompExperiment_2dim(cores = cores, treatment = treatment, control = control,
                         alpha = 0.05, tau = 0.5, lambda = c(10,120,1000))
all_D<-CompExperiment_2dim_D(cores = cores, treatment = treatment, control = control,
                             alpha = 0.05, tau = 0.5, lambda = c(10,120,1000))
```

B. Plot the results

We use `CompPlot_2dim` to visualize the results.

```
plot<-CompPlot_2dim(treatment = treatment, control=control, all = all,
                    xlab='x', ylab='y', zlab='z')
plot$trace
plot$g
```

```r
plot_D<-CompPlot_2dim(treatment = treatment, control=control, all = all_D,
                      xlab='x', ylab='y', zlab='z')
plot_D$trace
plot_D$g
```

C. (Optional) How λ is selected

If the user is interested in how the optimum λ is selected, they can plot the MCV values from cross-validation in the following way:

```r
#For BLB
plot(all$result1$Lambda, all$result1$Fid, type='l', xlab='lambda', ylab='MCV',
     main='lambda Search', col="dodgerblue")
abline(v=all$result1$lambda, lty=2, col="indianred", lwd=2)
grid()

#For Discretized
plot(all_D$result1$Lambda, all_D$result1$Fid, type='l', xlab='lambda', ylab='MCV',
     main='lambda Search', col="dodgerblue")
abline(v=all_D$result1$lambda, lty=2, col="indianred", lwd=2)
grid()
```