

# Package ‘bigKRLS’

June 14, 2017

**Type** Package

**Title** Optimized Kernel Regularized Least Squares

**Version** 1.5.3

**Date** 2017-06-13

**Description** Functions for Kernel-Regularized Least Squares optimized for speed and memory usage are provided along with visualization tools.  
For working papers, sample code, and recent presentations visit <<https://sites.google.com/site/petemohanty/software/>>.

**License** GPL (>= 2)

**Imports** Rcpp (>= 0.12.4), biganalytics, bigalgebra, shiny, parallel, ggplot2

**LinkingTo** Rcpp, RcppArmadillo, bigmemory, BH

**Depends** R (>= 3.3.0), bigmemory

**RoxygenNote** 6.0.1

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Pete Mohanty [aut, cre],  
Robert Shaffer [aut]

**Maintainer** Pete Mohanty <[pete.mohanty@gmail.com](mailto:pete.mohanty@gmail.com)>

**Repository** CRAN

**Date/Publication** 2017-06-14 19:44:29 UTC

## R topics documented:

bigKRLS	2
load.bigKRLS	4
predict.bigKRLS	5
save.bigKRLS	5
shiny.bigKRLS	6
summary.bigKRLS	6

bigKRLS

*bigKRLS***Description**

Runtime and Memory Optimized Kernel Regularized Least Squares Pete Mohanty (Stanford University) and Robert Shaffer (University of Texas at Austin)

bigKRLS

**Usage**

```
bigKRLS(y = NULL, X = NULL, sigma = NULL, derivative = TRUE,
        which.derivatives = NULL, vcov.est = TRUE, lambda = NULL, L = NULL,
        U = NULL, tol = NULL, noisy = TRUE, model_subfolder_name = NULL,
        overwrite.existing = F, Ncores = NULL)
```

**Arguments**

y	A vector of observations on the dependent variable; missing values not allowed. May be base R matrix or library(bigmemory) big.matrix.
X	A matrix of observations of the independent variables; factors, missing values, and constant vectors not allowed. May be base R matrix or library(bigmemory) big.matrix.
sigma	Bandwidth parameter, shorthand for sigma squared. Default: <code>sigma &lt;- ncol(X)</code> . Since x variables are standardized, facilitates interpretation of the Gaussian kernel, $\exp(-\text{dist}(X)^2/\text{sigma})$ a.k.a the similarity score. Of course, if dist between observation i and j is 0, there similarity is 1 since $\exp(0) = 1$ . Suppose i and j differ by one standard deviation on each dimension. Then the similarity is $\exp(-\text{ncol}(X)/\text{sigma}) = \exp(-1) = 0.368$ .
derivative	Logical: Estimate derivatives (as opposed to just coefficients)? Recommended for interpretability.
which.derivatives	Optional. For which columns of X should marginal effects be estimated ("variables of interest"). If <code>derivative=TRUE</code> and <code>which.derivative=NULL</code> , all will marginal effects estimated (default settings). Example: <code>out = bigKRLS(..., which.derivatives = c(1, 3, 5))</code>
vcov.est	Logical: Estimate variance covariance matrix? Required to obtain derivatives and standard errors on predictions (default = TRUE).
lambda	Regularization parameter. Default: estimated based (in part) on the eigenvalues of the kernel via Golden Search with convergence parameter "tolerance." Must be positive, real number.
L	Lower bound of Golden Search for lambda.
U	Upper bound of Golden Search for lambda.

<code>tol</code>	tolerance parameter for Golden Search for lambda. Default: $N / 1000$ .
<code>noisy</code>	Logical: Display progress to console (intermediate output, time stamps, etc.)? (Recommended particularly for SSH users, who should also use X11 forwarding to see Rcpp progress display.)
<code>model_subfolder_name</code>	If not null, will save estimates to this subfolder of your current working directory. Alternatively, use <code>save.bigKRLS()</code> on the outputted object.
<code>overwrite.existing</code>	Logical: overwrite contents in folder 'model_subfolder_name'? If FALSE, appends lowest possible number to model_subfolder_name name (e.g., <code>../myresults3/</code> ).
<code>Ncores</code>	Number of processor cores to use. Default = $\min(\text{ncol}(X), N - 2)$ (whichever is smaller). More than $N - 2$ NOT recommended. Uses <code>library(parallel)</code> unless <code>Ncores = 1</code> .

## Details

Kernel Regularized Least Squares (KRLS) is a kernel-based, complexity-penalized method developed by Hainmueller and Hazlett (2014) to minimize parametric assumptions while maintaining interpretive clarity. Here, we introduce bigKRLS, an updated version of the original KRLS R package with algorithmic and implementation improvements designed to optimize speed and memory usage. These improvements allow users to straightforwardly fit KRLS models to medium and large datasets ( $N > \sim 2500$ ).

### Major Updates

1. C++ integration. We re-implement most major computations in the model in C++ via Rcpp and RcppArmadillo. These changes produce up to a 50% runtime decrease compared to the original R implementation even on a single core.
2. Leaner algorithm. Because of the Tikhonov regularization and parameter tuning strategies used in KRLS, the method of estimation is inherently memory-heavy  $O(N^2)$ , making memory savings important even in small- and medium-sized applications. We develop and implement a new local derivatives algorithm, which reduces peak memory usage by approximately an order of magnitude, and cut the number of computations needed to find regularization parameter in half.
3. Improved memory management. Most data objects in R perform poorly in memory-intensive applications. We use a series of packages in the bigmemory environment to ease this constraint, allowing our implementation to handle larger datasets more smoothly.
4. Parallel Processing. Parallel processing with `parallel` makes the algorithm much faster for the marginal effects.
5. Interactive data visualization. We've designed an R Shiny app that allows users bigKRLS users to easily share results with collaborators or more general audiences. Simply call `shiny.bigKRLS()` on the outputted regression object.

**Requirements.** bigKRLS is under active development, and currently requires R version 3.3.0 or later. Windows users should use RTools 3.3 or later. RStudio users must have a current version as well.

For details on syntax, load the library and then open our vignette `vignette("bigKRLS_basics")`. Because of the quadratic memory requirement, users working on a typical laptop (8-16 gigabytes of

RAM) may wish to start at  $N = 2,500$  or  $5,000$ , particularly if the number of  $x$  variables is large. When you have a sense of how bigKRLS runs on your system, you may wish to only estimate a subset of the marginal effects at  $N = 10-15,000$  by setting `bigKRLS(... which.derivatives = c(1, 3, 5))` for the marginal effects of the first, third, and fifth  $x$  variable.

Mohanty, Pete and Robert B. Shaffer. 2016. "Messy Data, Robust Inference? Navigating Obstacles to Inference with bigKRLS" Project Presented to the International Methods Colloquium and useR! 2016. Visit <https://sites.google.com/site/petemohanty> for most recent version.

Hainmueller, Jens and Chad Hazlett. 2014. "Kernel Regularized Least Squares: Reducing Misspecification Bias with a Flexible and Interpretable Machine Learning Approach." *Political Analysis*. 22:143-68. <https://web.stanford.edu/~jhain/Paper/PA2014a.pdf> (Accessed May 20th, 2016).

Recent papers, presentations, and other code available at [github.com/rdr1990/code/](https://github.com/rdr1990/code/)

License Code released under GPL ( $\geq 2$ ).

## Value

bigKRLS Object containing slope and uncertainty estimates; `summary()` and `predict()` defined for class bigKRLS, as is `shiny.bigKRLS()`.

## Author(s)

**Maintainer:** Pete Mohanty <pete.mohanty@gmail.com>

Authors:

- Robert Shaffer <rbshaffer@utexas.edu>

## Examples

```
N <- 500 # proceed with caution above N = 5,000 for system with 8 gigs made available to R
P <- 4
X <- matrix(rnorm(N*P), ncol=P)
X <- cbind(X, sample(0:1, replace = TRUE, size = nrow(X)))
b <- runif(ncol(X))
y <- X %*% b + rnorm(nrow(X))
out <- bigKRLS(y, X, Ncores=2)
```

---

load.bigKRLS

load.bigKRLS

---

## Description

Reconstructs bigKRLS output object as list.

## Usage

```
load.bigKRLS(path, newname = NULL, pos = 1)
```

**Arguments**

path	Path to folder where bigKRLS object was saved.
newname	If NULL (default), bigKRLS object will appear as 'bigKRLS_out'
pos	position. Default == 1 (global environment).

---

predict.bigKRLS	<i>predict.bigKRLS</i>
-----------------	------------------------

---

**Description**

predict.bigKRLS

**Usage**

```
## S3 method for class 'bigKRLS'
predict(object, newdata, se.fit = FALSE, ...)
```

**Arguments**

object	bigKRLS output
newdata	new data. ncol(X) == ncol(newdata) but nrow(X) need not be the same as nrow(newdata).
se.fit	get standard errors on predictions?
...	ignore

---

save.bigKRLS	<i>save.bigKRLS</i>
--------------	---------------------

---

**Description**

save function, recommended when bigKRLS output contains big matrices (once  $N > 2,500$  the kernel is stored this way). Base R data will be stored in a list in an .rdata file, big matrices will be stored in .txt files. Call load.bigKRLS() to retrieve.

**Usage**

```
save.bigKRLS(object, model_subfolder_name, overwrite.existing = F)
```

**Arguments**

object	bigKRLS output
model_subfolder_name	A name of a folder where the file(s) will be written.
overwrite.existing	Logical – write over folders with the same name? Default == FALSE.

shiny.bigKRLS

*shiny.bigKRLS***Description**

shiny.bigKRLS

**Usage**

```
shiny.bigKRLS(out, export = F, main.label = "bigKRLS estimates",
  plot.label = NULL, xlabs = NULL, font_size = 20, hline = 0,
  shiny.palette = c("#E41A1C", "#377EB8", "#4DAF4A", "#984EA3", "#FF7F00",
    "#FFFF33", "#A65628", "#F781BF", "#999999"))
```

**Arguments**

out	bigKRLS output. Does not require any N * N matrices.
export	Logical – instead of running Shiny, prepare the key estimates as a separate file? (The N * Ns are too large for most Shiny servers but the key estimates are only N * P).
main.label	Main label (upper left of app)
plot.label	Optional character
xlabs	Optional character vector for the x variables.
font_size	Font size. Default == 20. calls "ggplot2::theme_minimal(base_size = font_size)"
hline	horizontal line. Default == 0 (x axis)
shiny.palette	color scheme for main app. 9 colors.

summary.bigKRLS

*summary.bigKRLS***Description**

Summary function for bigKRLS output. Call `knitr::kable(summary(my_output)[[1]])` or `knitr::kable(summary(my_output)[[2]])` to format with RMarkdown.

**Usage**

```
## S3 method for class 'bigKRLS'
summary(object, probs = c(0.05, 0.25, 0.5, 0.75, 0.95),
  digits = 4, labs = NULL, ...)
```

**Arguments**

<code>object</code>	bigKRLS output. If you saved with <code>save.bigKRLS()</code> , only the <code>.rdata</code> file is needed for this function.
<code>probs</code>	For quantiles.
<code>digits</code>	Number of significant digits.
<code>labs</code>	Optional vector of x labels.
<code>...</code>	ignore

# Index

`bigKRLS`, [2](#)  
`bigKRLS-package (bigKRLS)`, [2](#)  
`load.bigKRLS`, [4](#)  
`predict.bigKRLS`, [5](#)  
`save.bigKRLS`, [5](#)  
`shiny.bigKRLS`, [6](#)  
`summary.bigKRLS`, [6](#)