Final

dan crowley

4/28/2020

```r
library(ape)
library(phylofactor)
```

```
## Loading required package: magrittr

## Loading required package: data.table

## Loading required package: Matrix
```

```r
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------------

## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ---------------------------------------------------------------------------------------------
## x dplyr::between()   masks data.table::between()
## x tidyr::expand()    masks Matrix::expand()
## x tidyr::extract()   masks magrittr::extract()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x tidyr::pack()      masks Matrix::pack()
## x purrr::set_names() masks magrittr::set_names()
## x purrr::transpose() masks data.table::transpose()
## x tidyr::unpack()    masks Matrix::unpack()
```

```r
sim = (seq(0.000001,100,10)) #must be the same as line 11
eps = (seq(0.000001,100,10))
error <- matrix(0, length(sim)*length(eps), 14)
index = 0
```

```r
set.seed(Sys.time())
num= 50 #rpois(1,60)
tree <-rtree(num)
tree$tip.label <- as.character(1:num)
drop_tips = num/20 # determines ratio of test to training data

#determine max groups
max_groups <- 2*num-2

#choose a clade of the good length, we want to make sure its not monophyletic
```

```
clade1 = 0
clade2 = 0

while((length(clade1) < num / 3) | (length(clade2) < num / 3))
{
  #grab clades
  samp = sample(max_groups, 1)
  clade1 = getPhyloGroups(tree)[samp][1][[1]][[1]]
  clade2 = getPhyloGroups(tree)[samp][1][[1]][[2]]
}

#randomly generate data
BodySize <-rlnorm(num, sdlog = 1.6)
BodySize[clade1] <-rlnorm(length(clade1), sdlog = 1.6)*5

#create a data matrix, with the body size, tip labels, and a basis function of all 1s
BodySize = as.data.frame(cbind(BodySize, tree$tip.label, basis = 1, intercept = 1))
BodySize <- BodySize %>%
  dplyr::mutate(Species=  as.character(V2)) %>%
  dplyr::select(-V2)

#now wegenerate the missing data.
#these are species we haven't observed data for
#in this simulation around 1/3 of the data are missing

#num_miss = 1 + rpois(1,nrow(train_BodySize)/2.5)
num_miss = nrow(BodySize)/3
#num_miss = rpois(1,10)
miss_tip = sample(tree$tip.label, num_miss)

#create missing body size variable
BodySize$BodySize_miss = BodySize$BodySize
BodySize$basis_miss = BodySize$basis
BodySize$intercept_miss = BodySize$intercept

#create a new variable, and label the boddy size and basis functions NA for these tips
BodySize[(BodySize$Species %in% miss_tip),]$BodySize_miss = NA
BodySize[(BodySize$Species %in% miss_tip),]$basis_miss = NA
BodySize[(BodySize$Species %in% miss_tip),]$intercept_miss = NA

BodySize$basis = as.numeric(BodySize$basis)
BodySize$BodySize = as.numeric(as.character(BodySize$BodySize))
BodySize$intercept = as.numeric(BodySize$intercept)
```

#grab a sample of N tips for the testing tree

```
test_tree <- ape::drop.tip(tree,tree$tip.label[!(tree$tip.label %in% sample(tree$tip.label, drop_tips))]
train_tree <- ape::drop.tip(tree,tree$tip.label[(tree$tip.label %in% test_tree$tip.label)])
```

#should be equal to 0

```
sum(test_tree$tip.label %in% train_tree$tip.label) == 0
```

```
## [1] TRUE
```

```r
sum(train_tree$tip.label %in% test_tree$tip.label) == 0
```

```
## [1] TRUE
```

```r
num - (length(train_tree$tip.label) + length(test_tree$tip.label)) == 0
```

```
## [1] TRUE
```

#now, split the dataset into the training and testing datasets

```r
train_BodySize = BodySize %>%
  dplyr::filter(Species %in% train_tree$tip.label)

test_BodySize = BodySize %>%
  dplyr::filter(Species %in% test_tree$tip.label)
```

#should both be true:

```r
sum(train_BodySize$Species %in% test_BodySize$Species) == 0
```

```
## [1] TRUE
```

```r
sum(test_BodySize$Species %in% train_BodySize$Species) == 0
```

```
## [1] TRUE
```

```r
train_BodySize$basis_miss = as.numeric(train_BodySize$basis_miss)
train_BodySize$BodySize_miss = as.numeric(as.character(train_BodySize$BodySize_miss))
train_BodySize$intercept_miss = as.numeric(train_BodySize$intercept_miss)
```

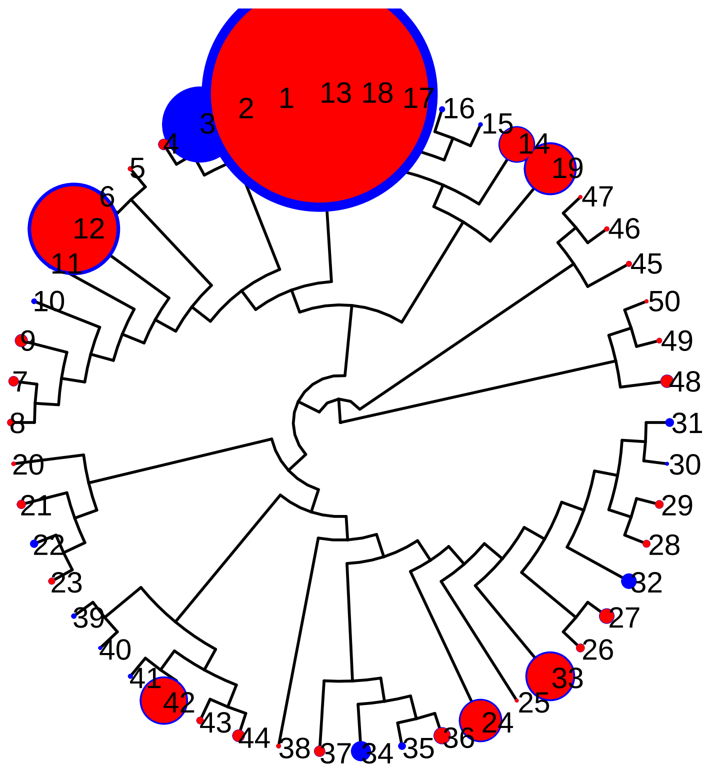
```r
test_BodySize <- test_BodySize %>%
  dplyr::mutate(intercept = as.numeric(as.character(intercept))) %>%
  dplyr::mutate(BodySize = as.numeric(as.character(BodySize))) %>%
  dplyr::mutate(basis = as.numeric(as.character(basis)))

train_BodySize$basis_miss = as.numeric(as.character(train_BodySize$basis_miss))
train_BodySize$intercept_miss = as.numeric(as.character(train_BodySize$intercept_miss))
train_BodySize$train_BodySize_miss = as.numeric(as.character(train_BodySize$BodySize_miss))
```

```r
ggtree::ggtree(tree, branch.length = 'none', layout = 'circular') +
  ggtree::geom_tippoint(size=.25*as.numeric(BodySize$BodySize),col='blue')  +
  ggtree::geom_tippoint(size=.23*as.numeric(as.character(BodySize$BodySize_miss)),col='red') +
  ggtree::geom_tiplab()
```

```
## Registered S3 method overwritten by 'treeio':
##   method     from
##   root.phylo ape
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

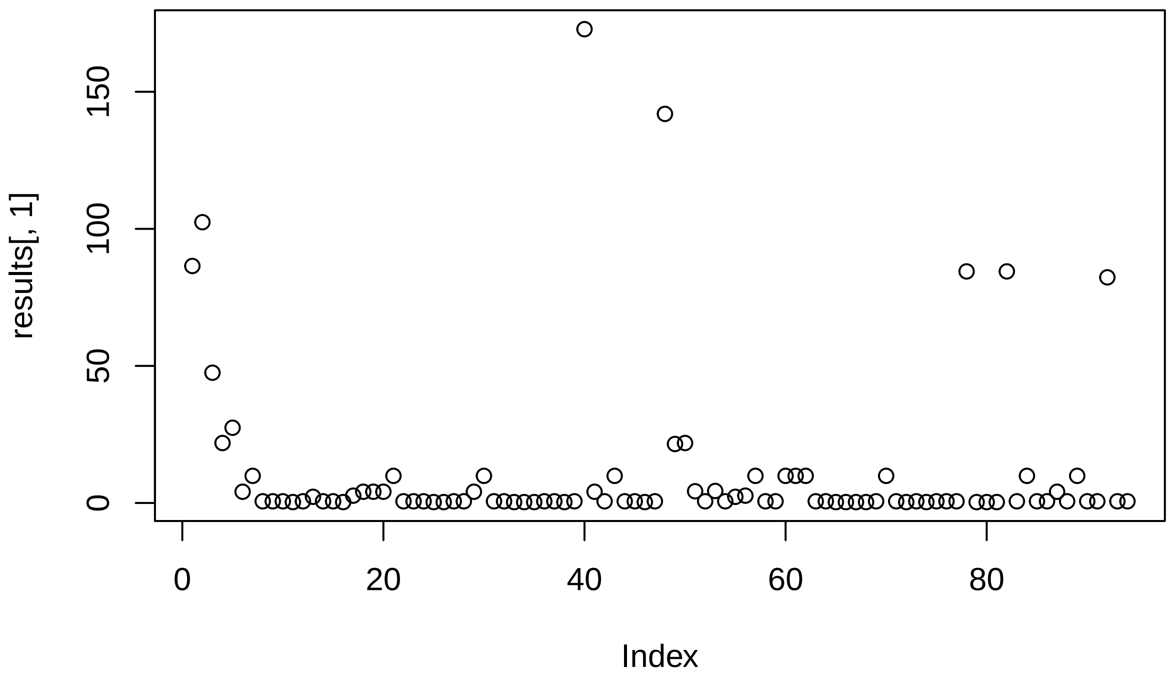
```
## Warning: Removed 16 rows containing missing values (geom_point_g_gtree).
```



```
for( x in sim)
{
  print(paste("X =", x))
  epsilon = x
  j=1
  for(j in eps)
  {

    #switch things up, keep the same tree, but grab different training and tesing everytime

    print(paste("j =", j))
    index = index + 1
    delta = 1 #delta is a control parameter for the bayesian model. as it gets bigger the estimates shr
    delta = delta*(j)
    #delta = delta*j
    #print(num)
    source('6th_attempt.R')

    #create the predictions based on mu_map
    #create a new variable, and label the boddy size and basis functions NA for these tips

    #we need to test the max estimate, and the mle estimate
    #when there are no missing data, they find the same split point
    #however, when the tips are missing they tend to get different
    #we should probably test them both on the actual split point..

    #first, identify the two groups pulled out by GPF in the training dataset

    #grp1 = train_tree$tip.label[gpf_results$groups[1][[1]][[1]]]
    #grp2 = train_tree$tip.label[gpf_results$groups[1][[1]][[2]]]
    min = which.min(results[,1])
    min_2 = which.min(results[,7])

  # results[i,1] <-  sse_map
  # results[i,2] <-  beta_ridge[2]
  # results[i,3] <-  beta_ridge[1]
  # results[i,4] <-  N1
  # results[i,5] <-  N2
  # results[i,6] <-  theta_2
  # results[i,7] <-  SSE_train

   error[index,1]  <-  results[min,1] #sse
   error[index,2]  <-  results[min,2]
   error[index,3]  <-  results[min,3]
   error[index,4]  <-  results[min,4]
   error[index,5]  <-  results[min,5]
   error[index,6]  <-  results[min,6]
   error[index,7]  <-  delta
   error[index,8]  <-  epsilon
   error[index,9]  <-  sse_gpf
   error[index,10] <-  results[min,7]
   error[index,11] <-  results[min_2,1]
   error[index,12] <-  results[min_2,7]
   error[index,13] <-  min
   error[index,14] <-  min_2

   print(error[index,])
  }
}
```

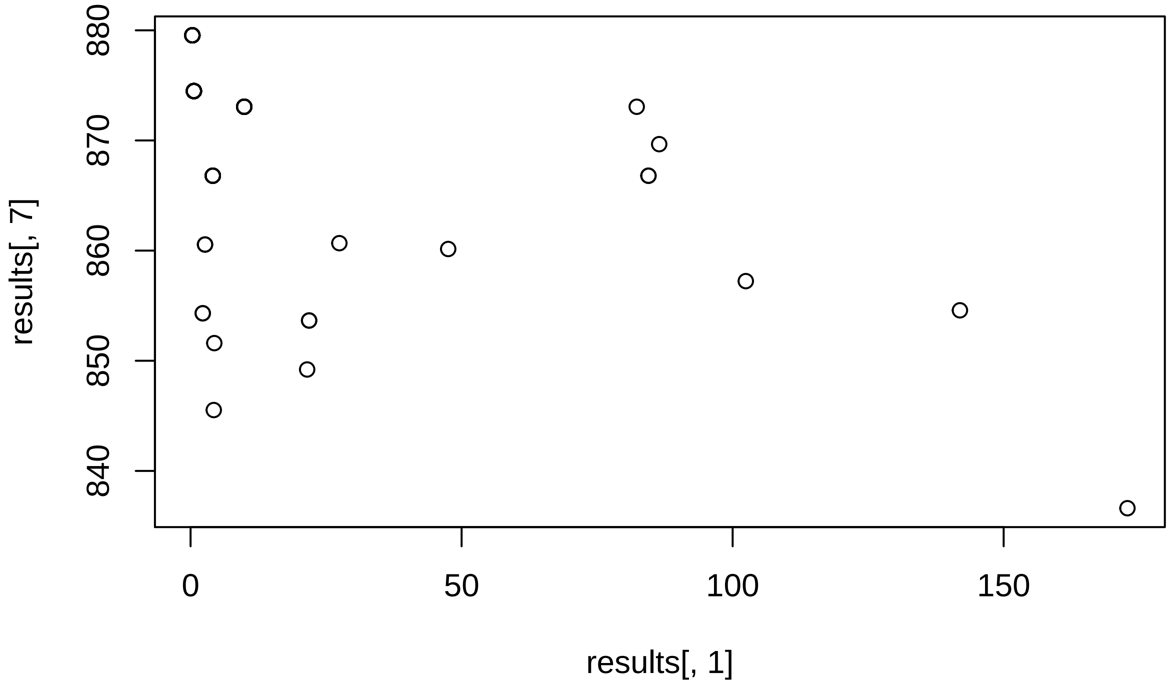
```
## [1] "X = 1e-06"
## [1] "j = 1e-06"
```

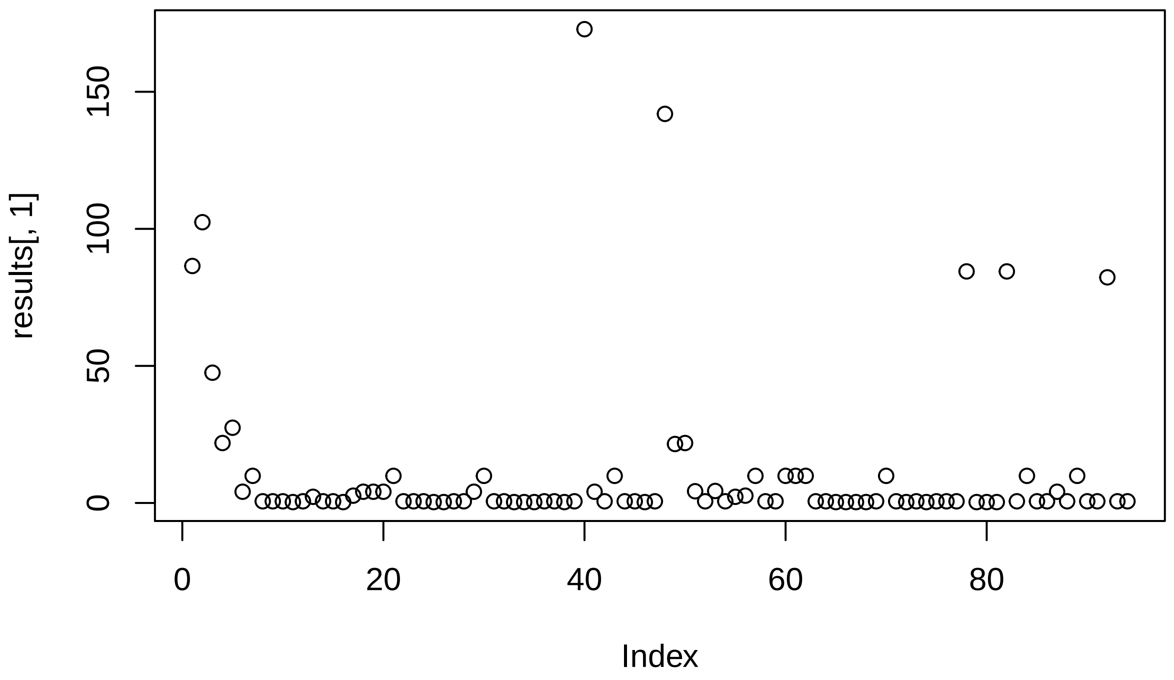
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```



```
##  1 factor completed in 0.00539 minutes.    Estimated time of completion: 2020-04-28 10:37:18    [1]
##  [7]   0.0000010   0.0000010  78.6860377 879.5454611 172.8354081 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 10.000001"
```

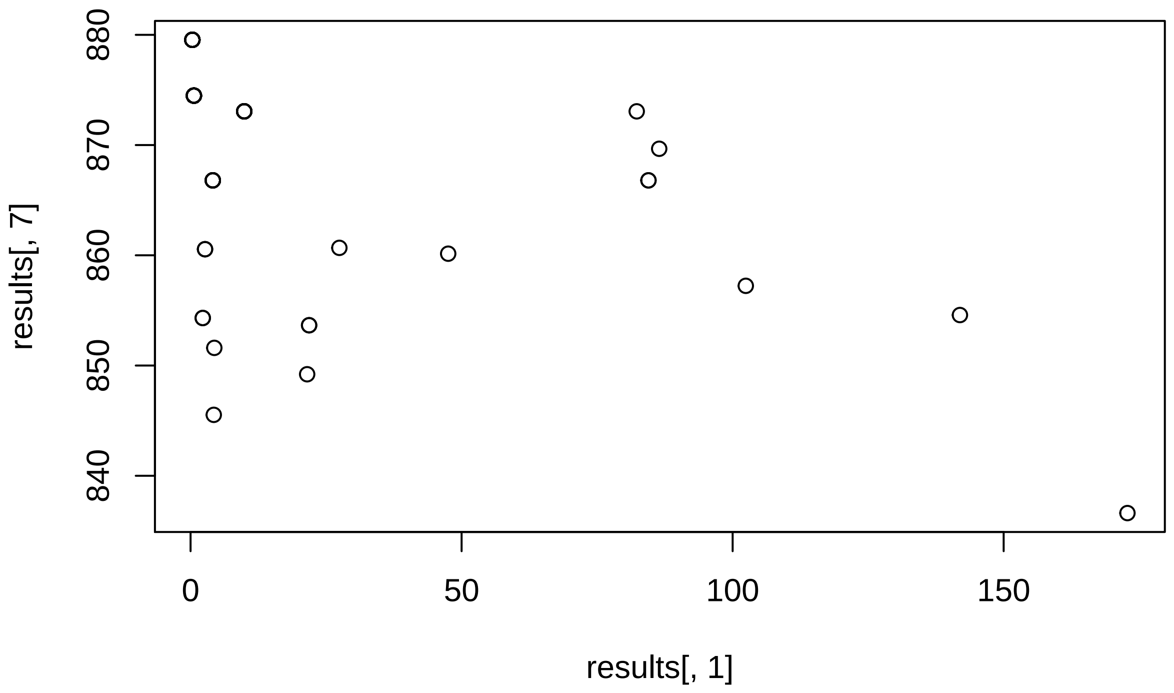
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

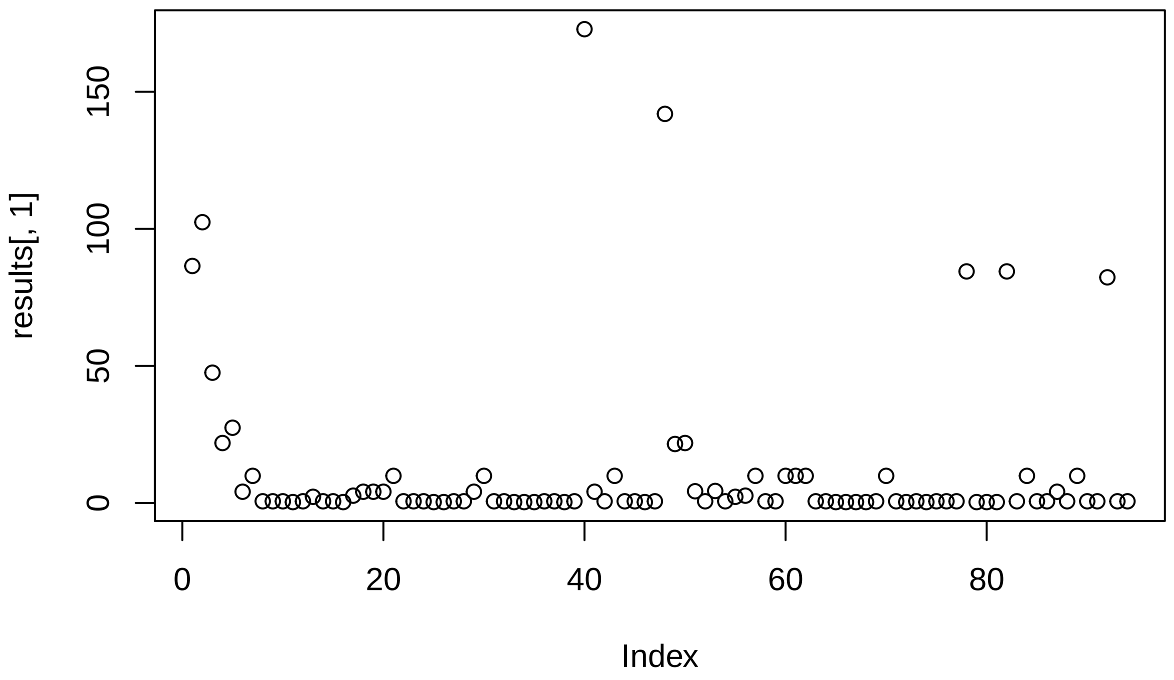




```
##  1 factor completed in 0.00549 minutes.    Estimated time of completion: 2020-04-28 10:37:20    [1]
##  [7]  10.0000010   0.0000010  78.6860377 879.5454611 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 20.000001"
```

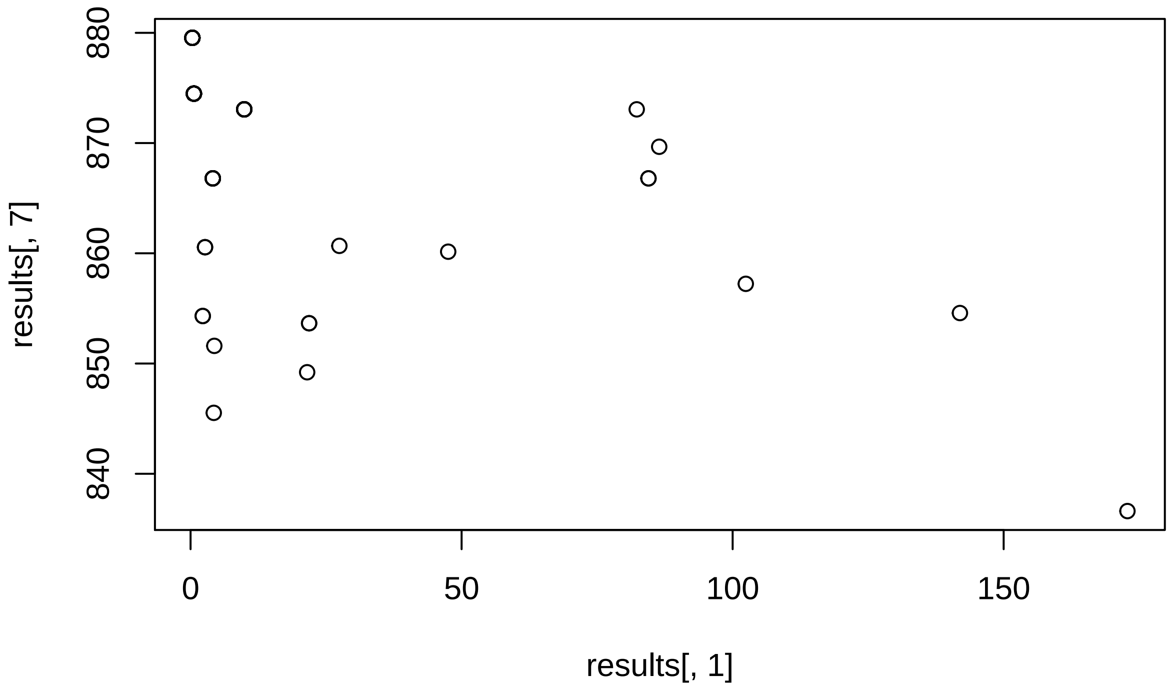
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

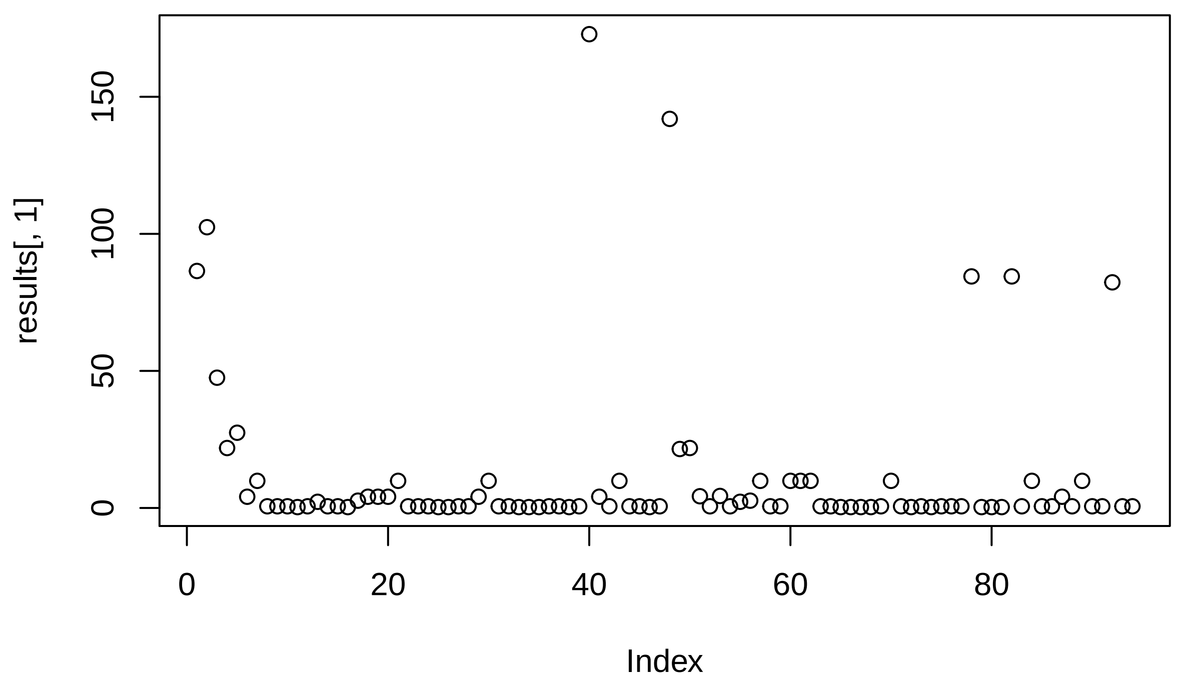




```
##  1 factor completed in 0.00492 minutes.    Estimated time of completion: 2020-04-28 10:37:22    [1]
##  [7]  20.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 30.000001"
```

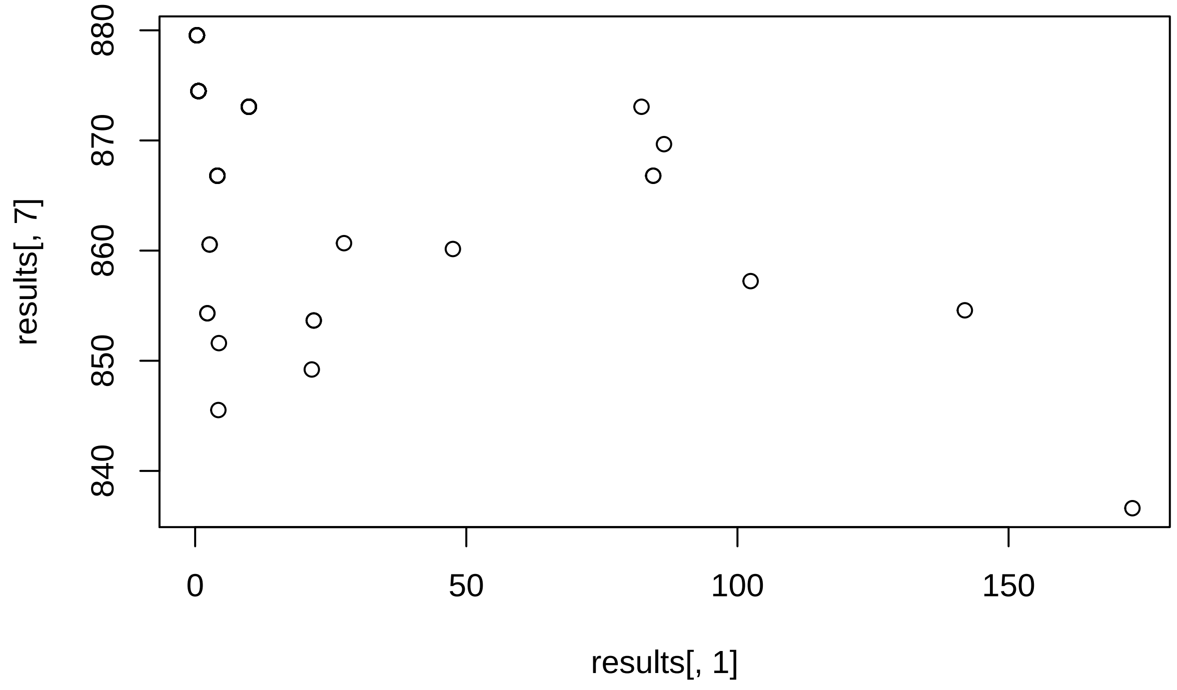
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

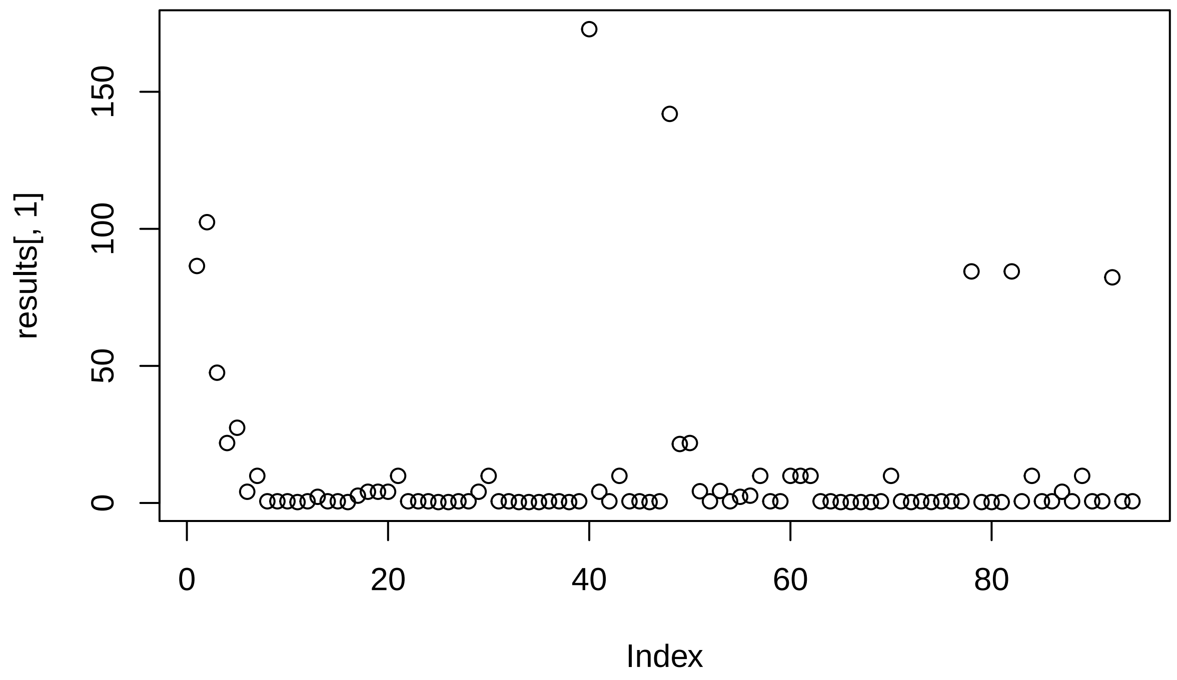




```
##  1 factor completed in 0.00394 minutes.    Estimated time of completion: 2020-04-28 10:37:24    [1]
##  [7]  30.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

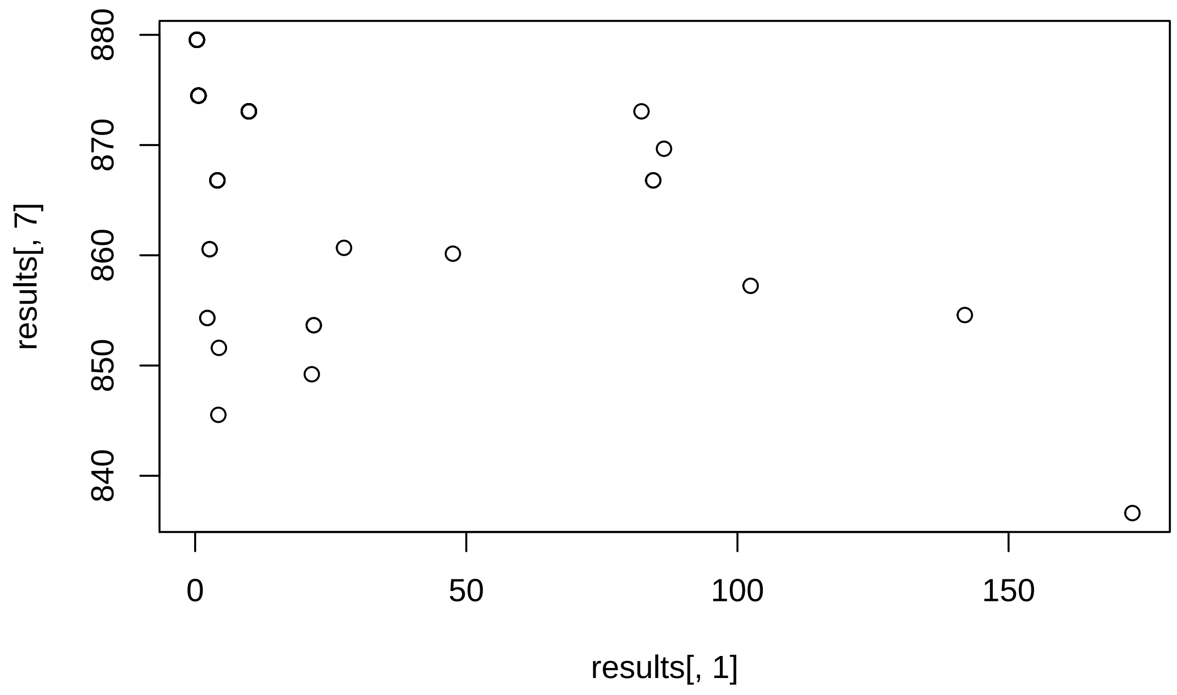
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

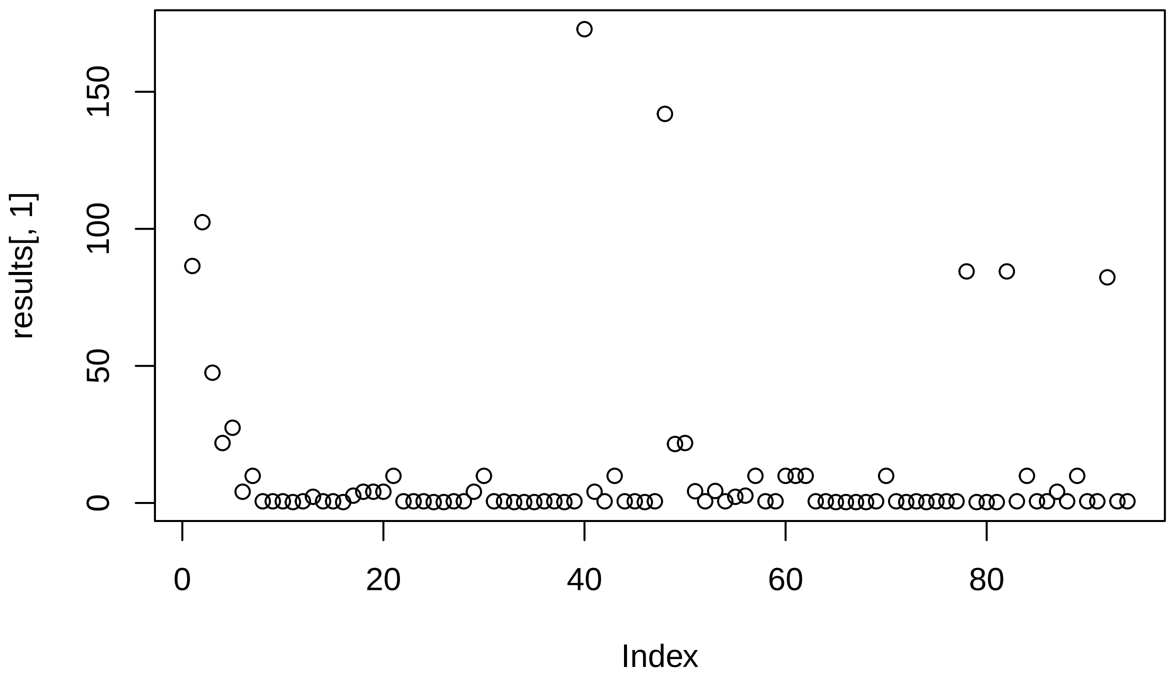




```
##  1 factor completed in 0.00397 minutes.    Estimated time of completion: 2020-04-28 10:37:26    [1]
##  [7]  40.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

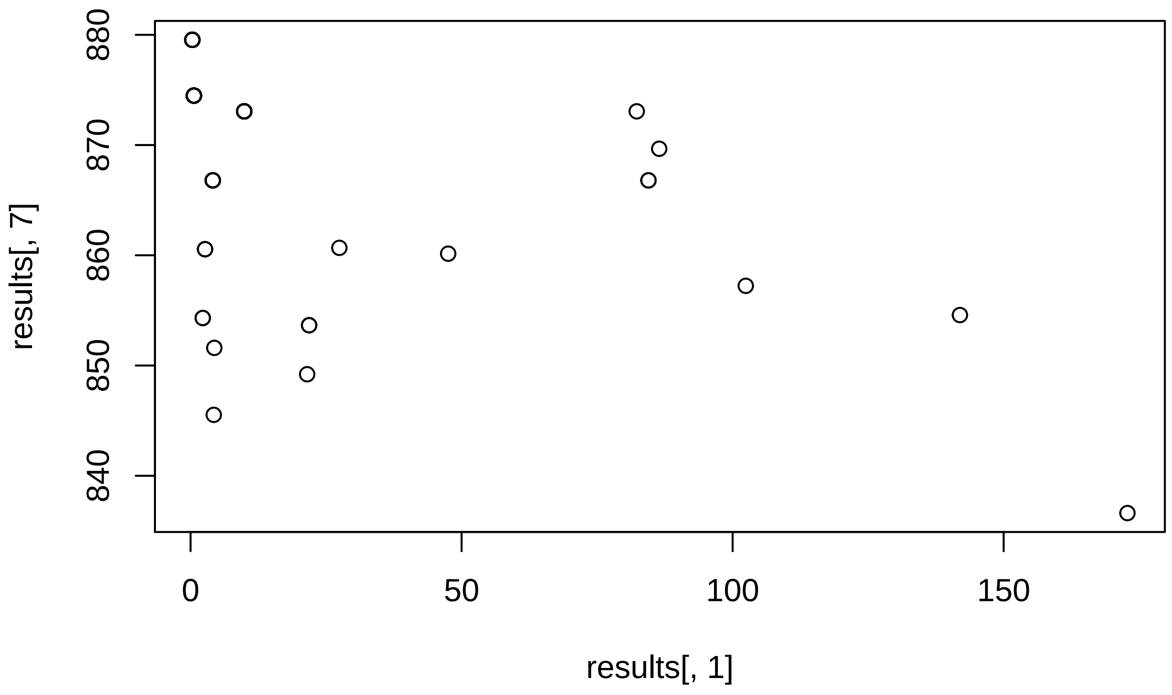
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

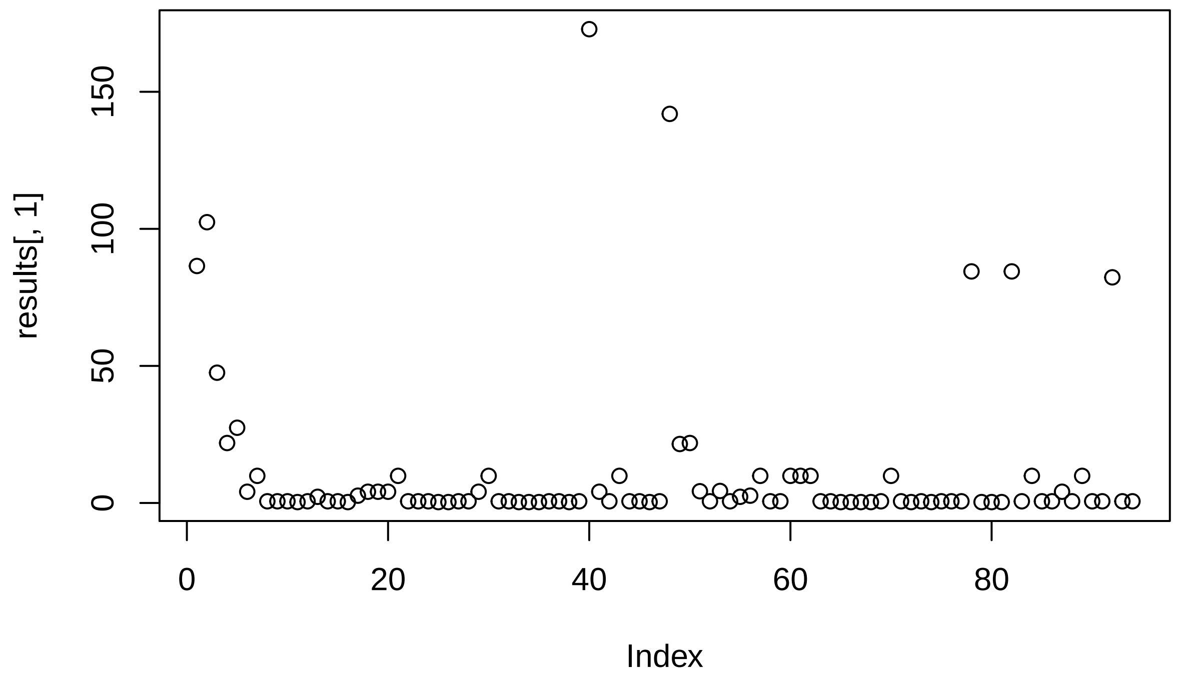




```
##  1 factor completed in 0.00431 minutes.    Estimated time of completion: 2020-04-28 10:37:28    [1]
##  [7]  50.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

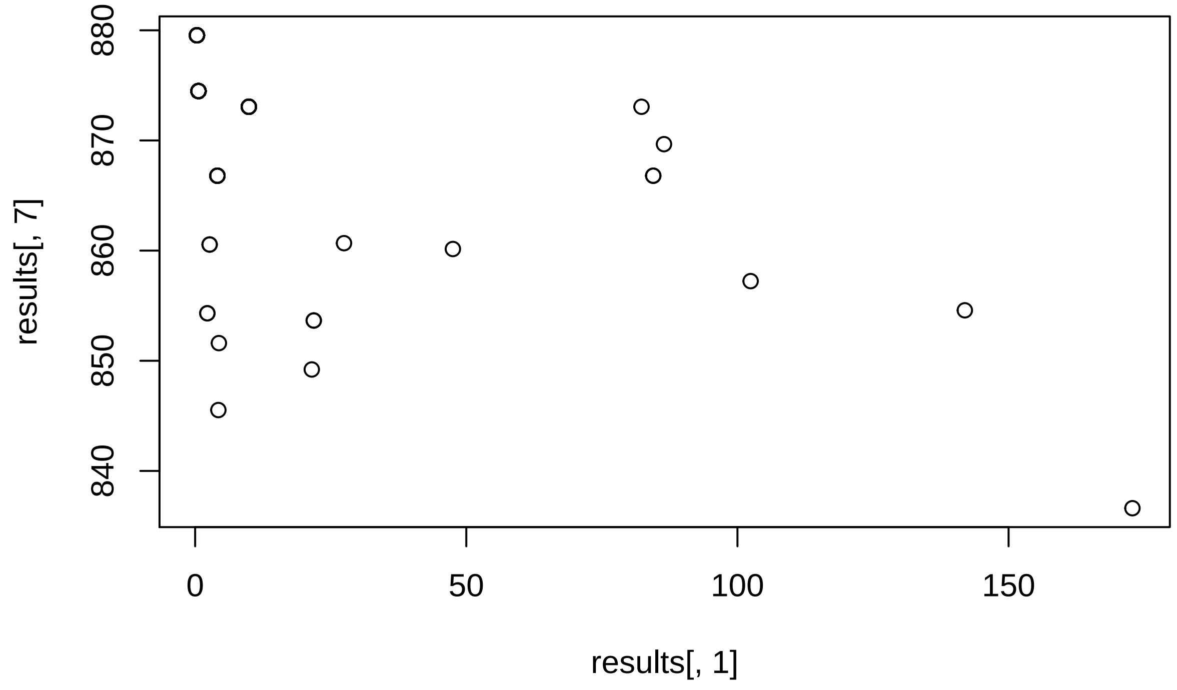
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

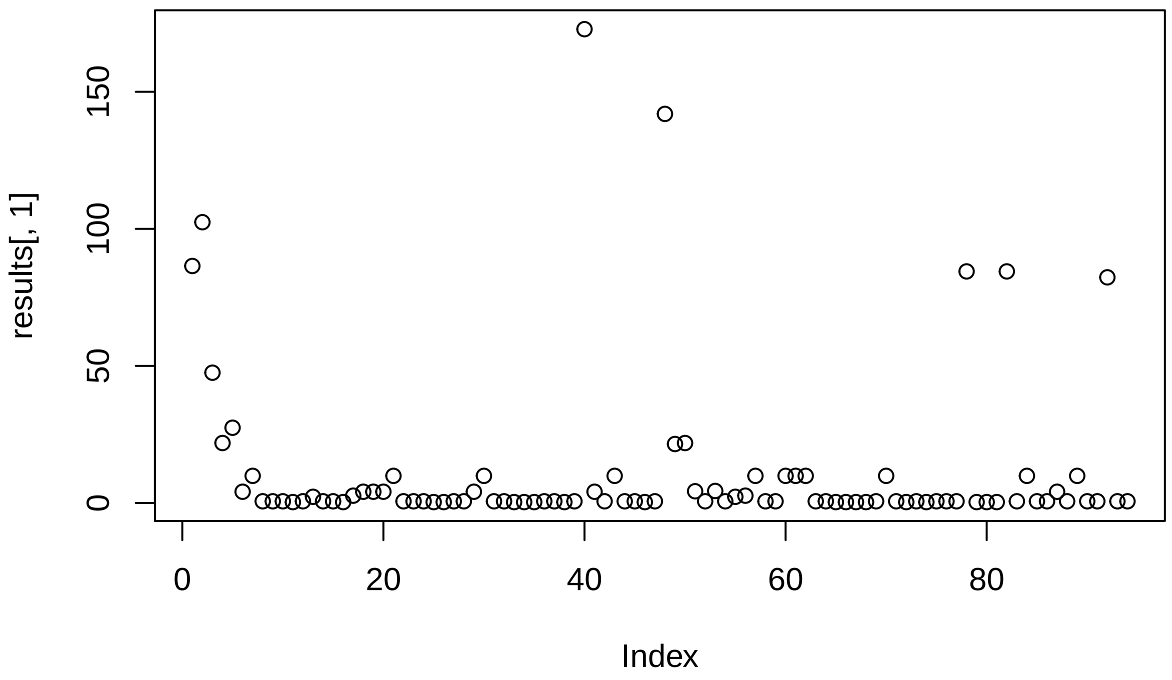




```
##  1 factor completed in 0.00484 minutes.    Estimated time of completion: 2020-04-28 10:37:31    [1]
##  [7]  60.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

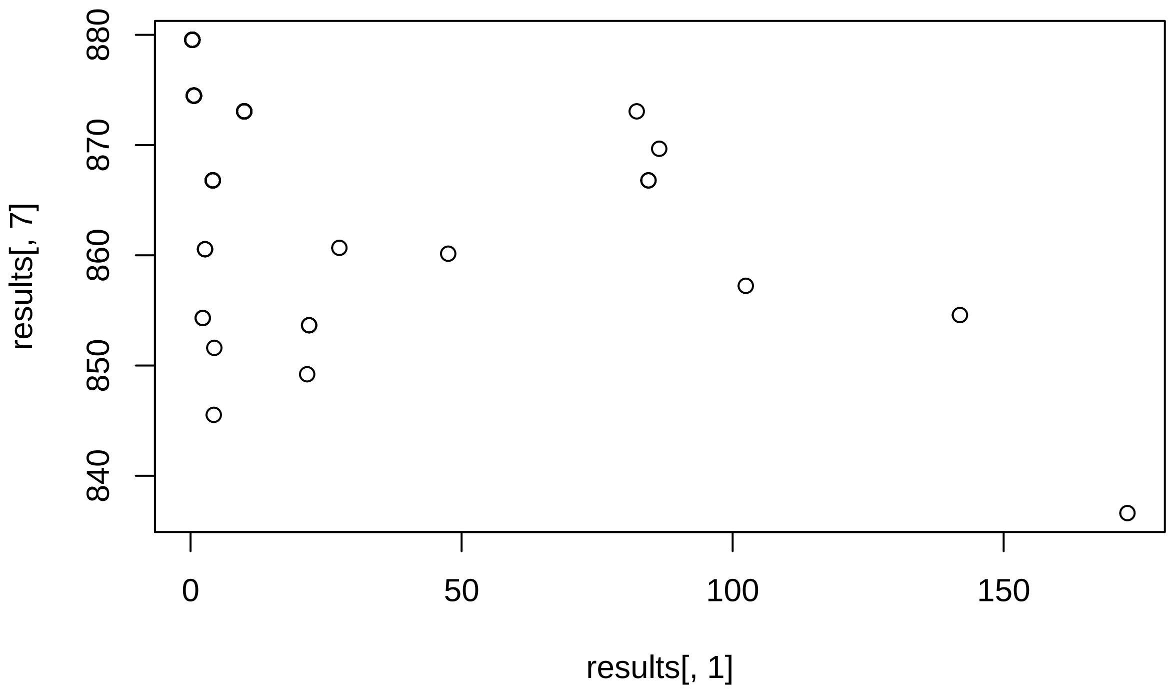
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

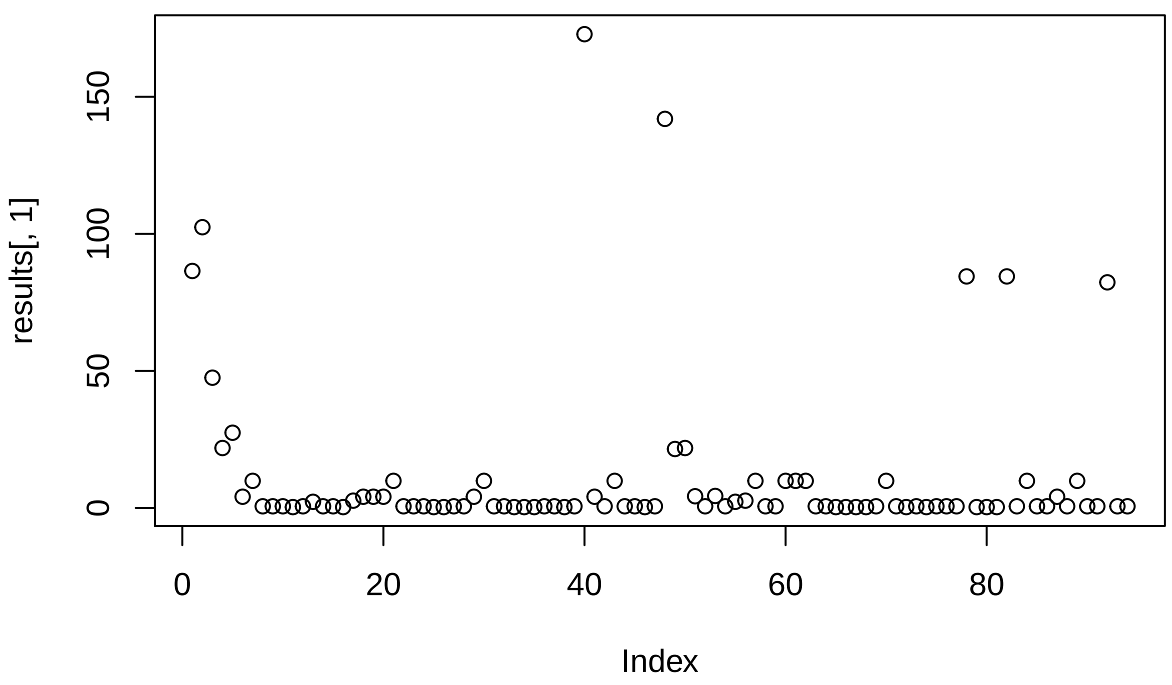




```
##  1 factor completed in 0.0104 minutes.    Estimated time of completion: 2020-04-28 10:37:34    [1]
##  [7]  70.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

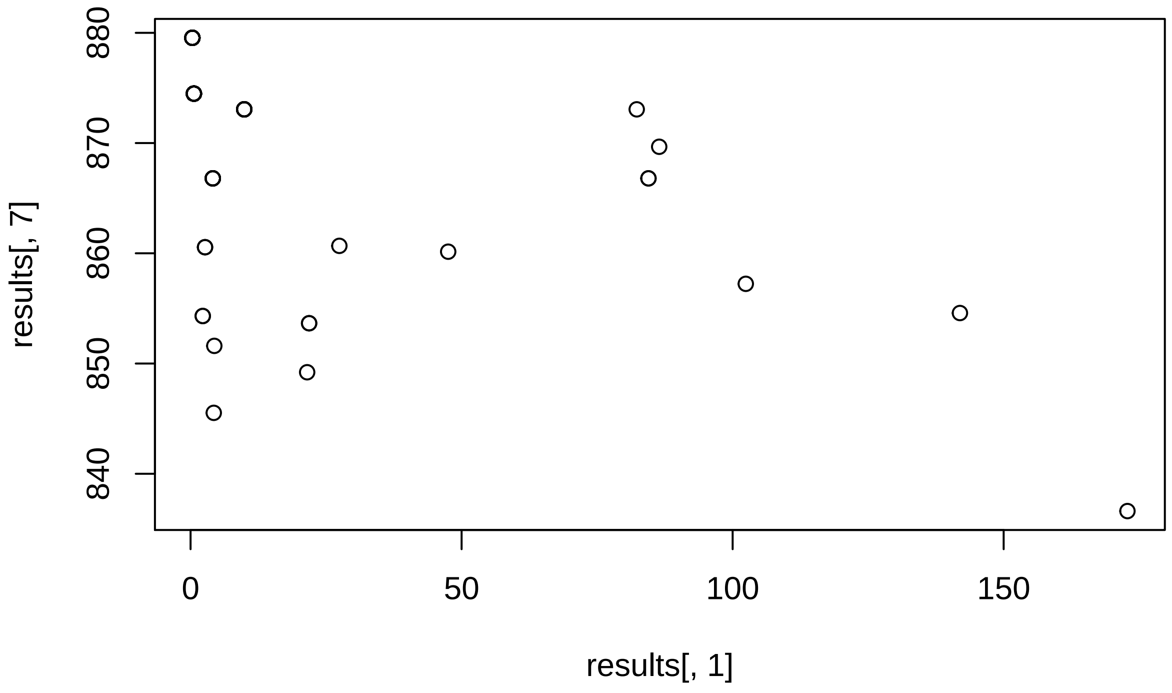
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

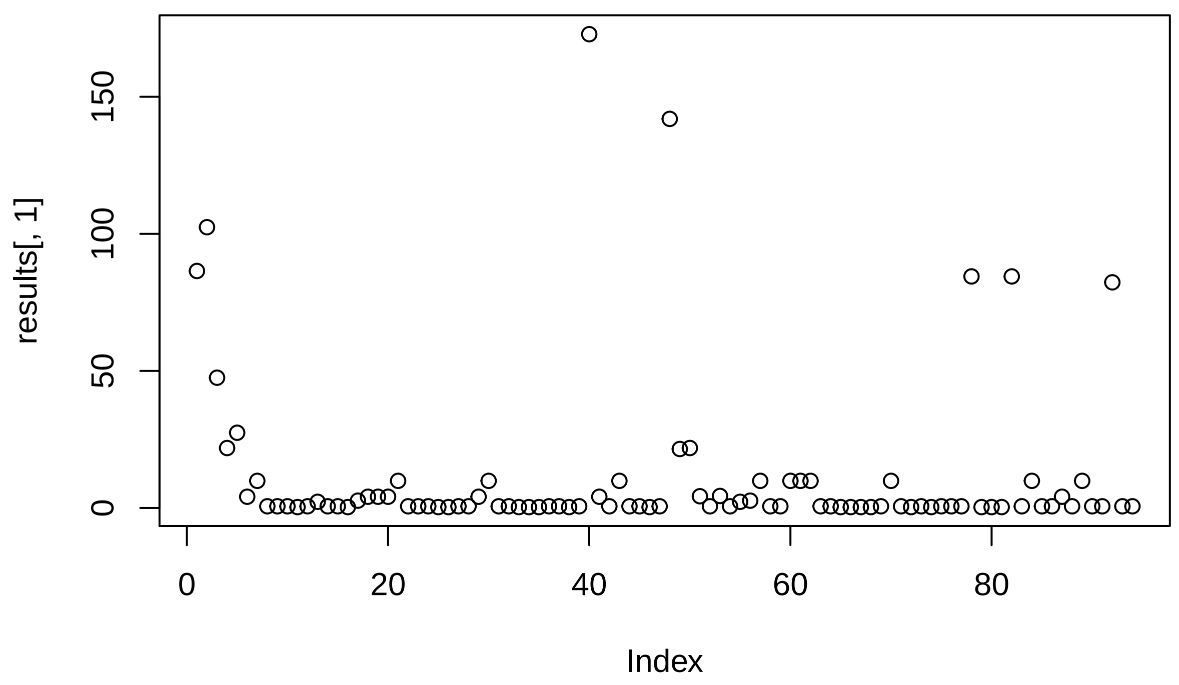




```
##  1 factor completed in 0.00565 minutes.    Estimated time of completion: 2020-04-28 10:37:37    [1]
##  [7]  80.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

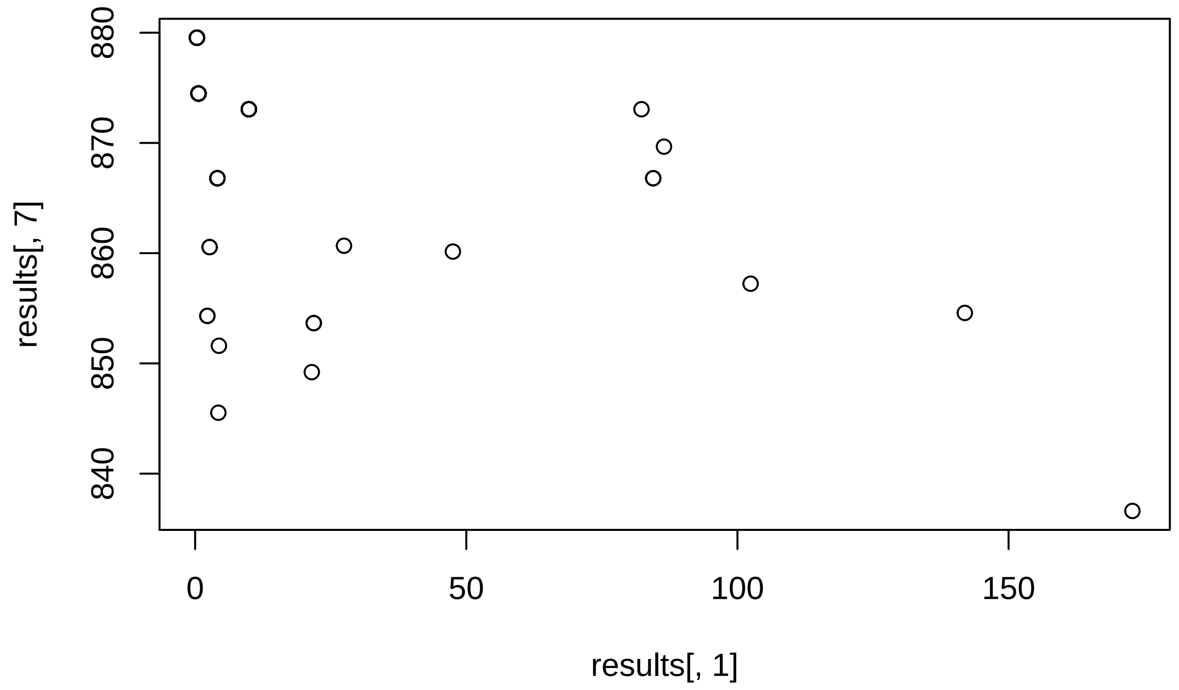
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

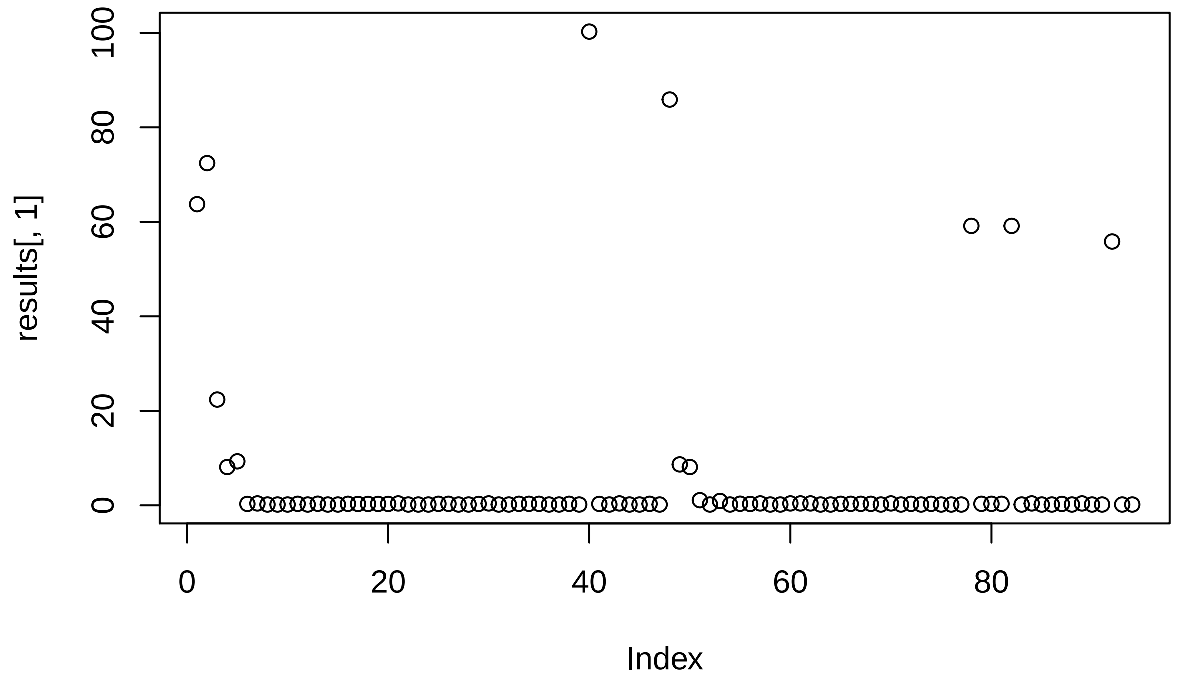




```
##  1 factor completed in 0.0052 minutes.    Estimated time of completion: 2020-04-28 10:37:40    [1]
##  [7]  90.0000010   0.0000010  78.6860377 879.5454611 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 10.000001"
## [1] "j = 1e-06"
```

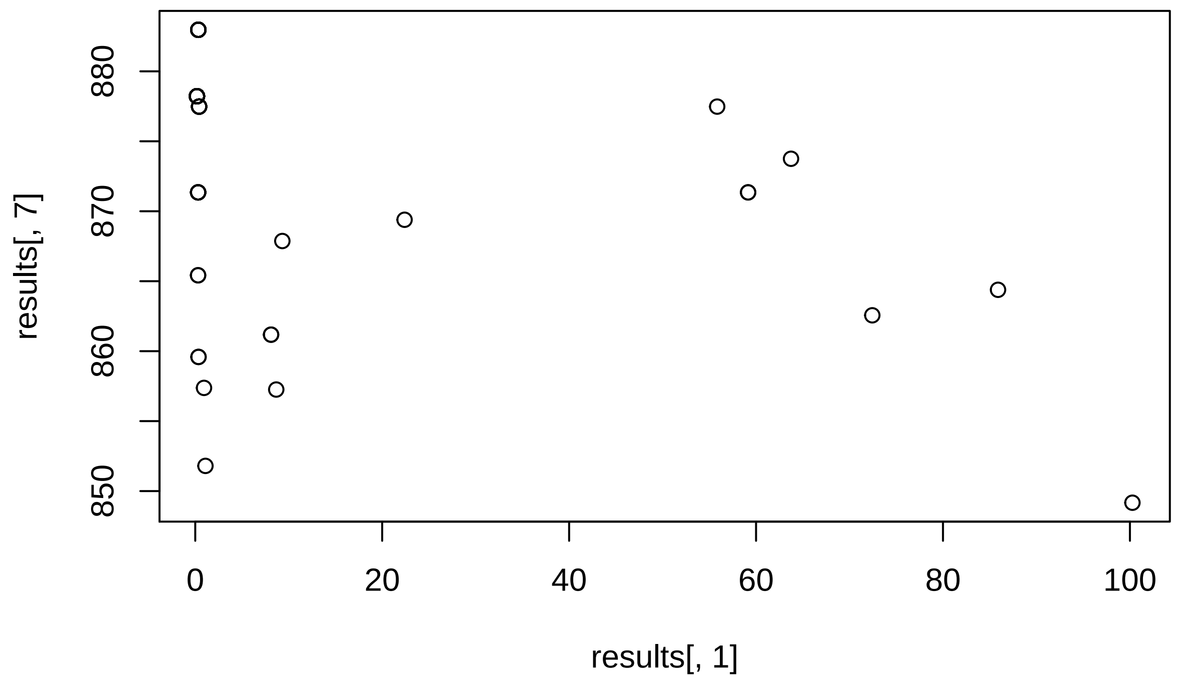
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

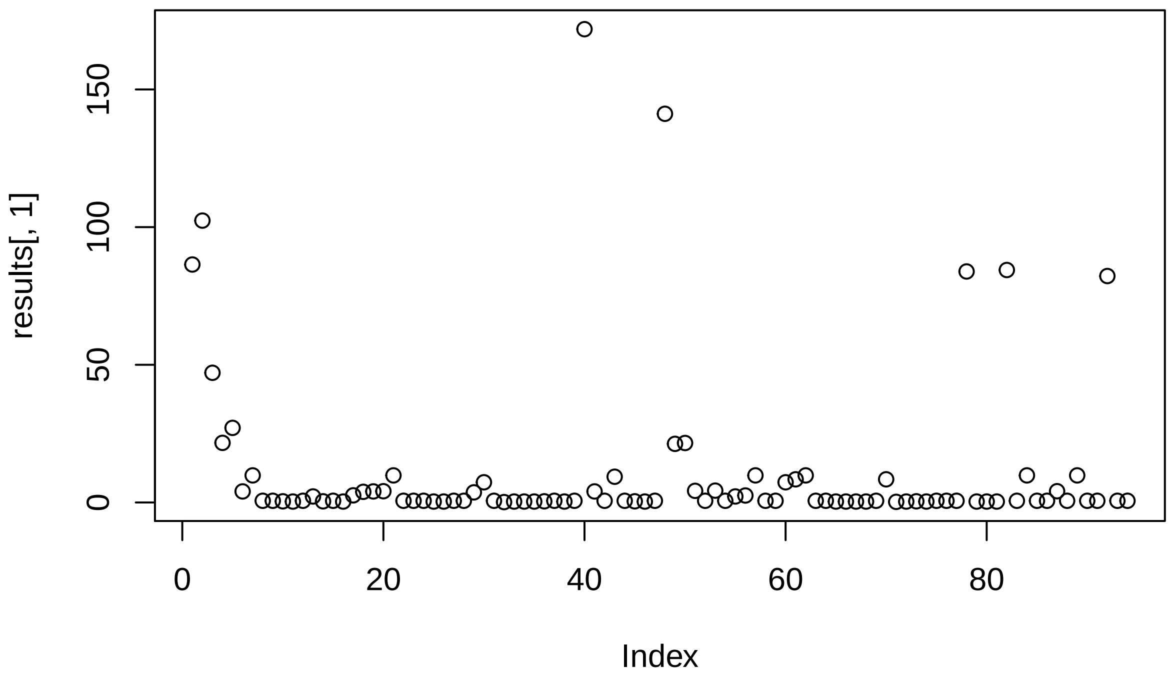




```
##  1 factor completed in 0.00486 minutes.    Estimated time of completion: 2020-04-28 10:37:42    [1]
##  [7]   0.0000010  10.0000010  78.6860377 878.2215943 100.2684175 849.1695938
## [13]   8.0000000  40.0000000
## [1] "j = 10.000001"
```

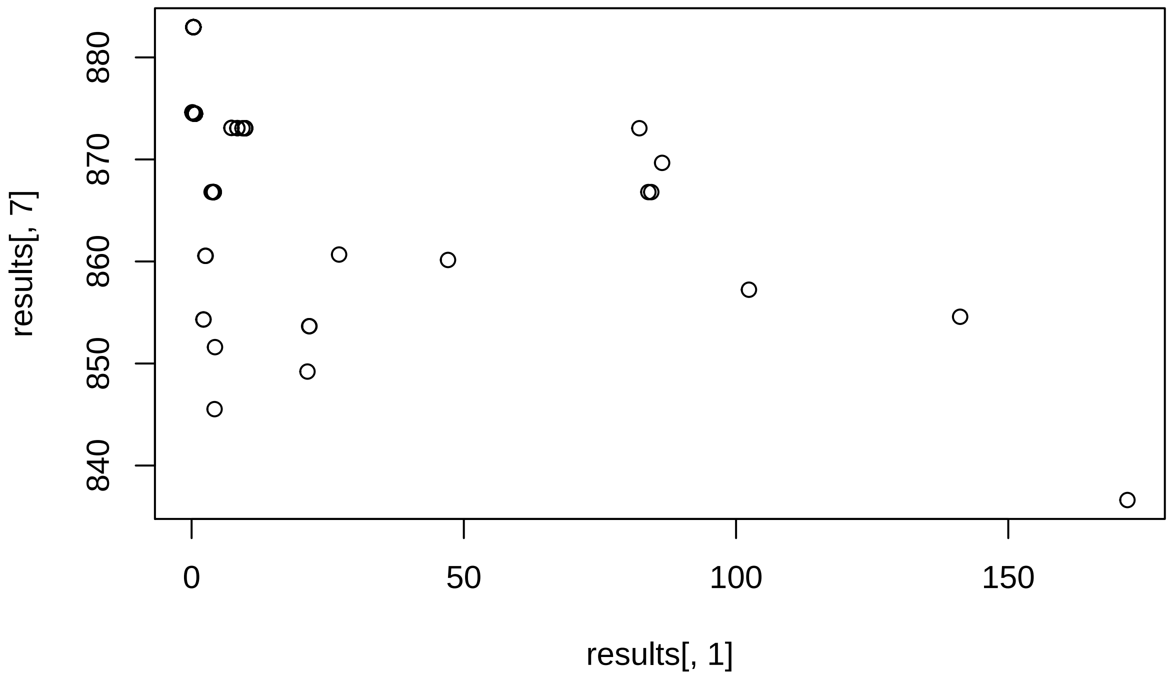
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

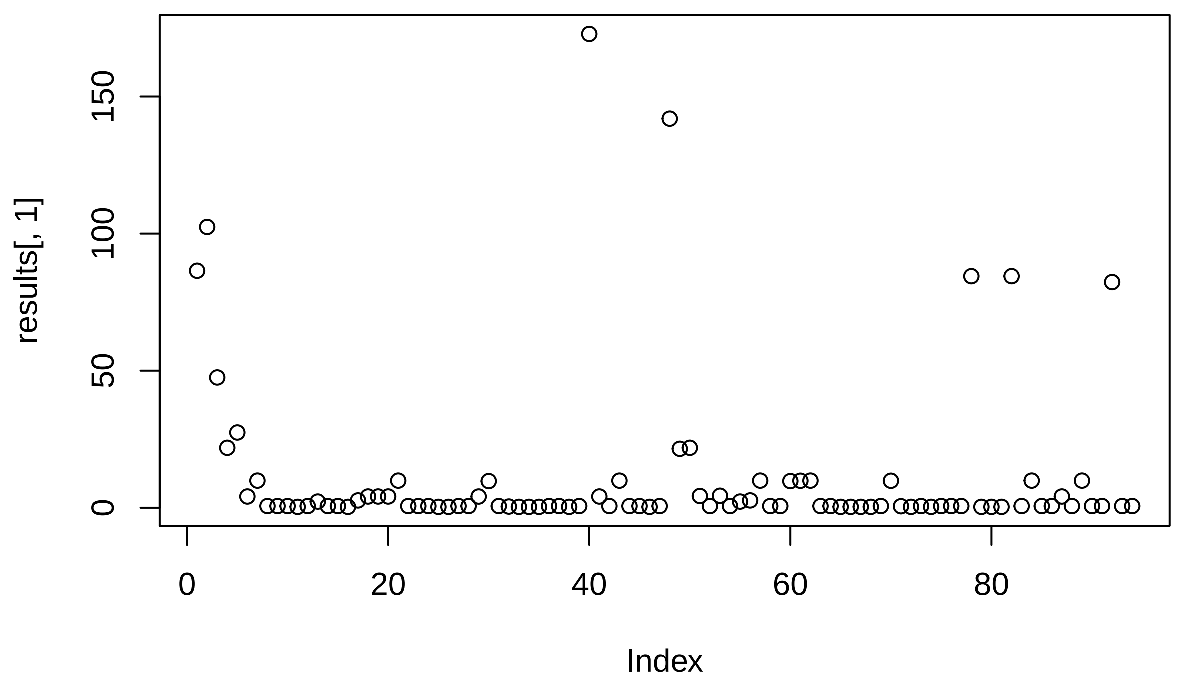




```
##  1 factor completed in 0.00531 minutes.    Estimated time of completion: 2020-04-28 10:37:45    [1]
##  [7]  10.0000010  10.0000010  78.6860377 874.6194925 171.8892003 836.6179951
## [13]  32.0000000  40.0000000
## [1] "j = 20.000001"
```

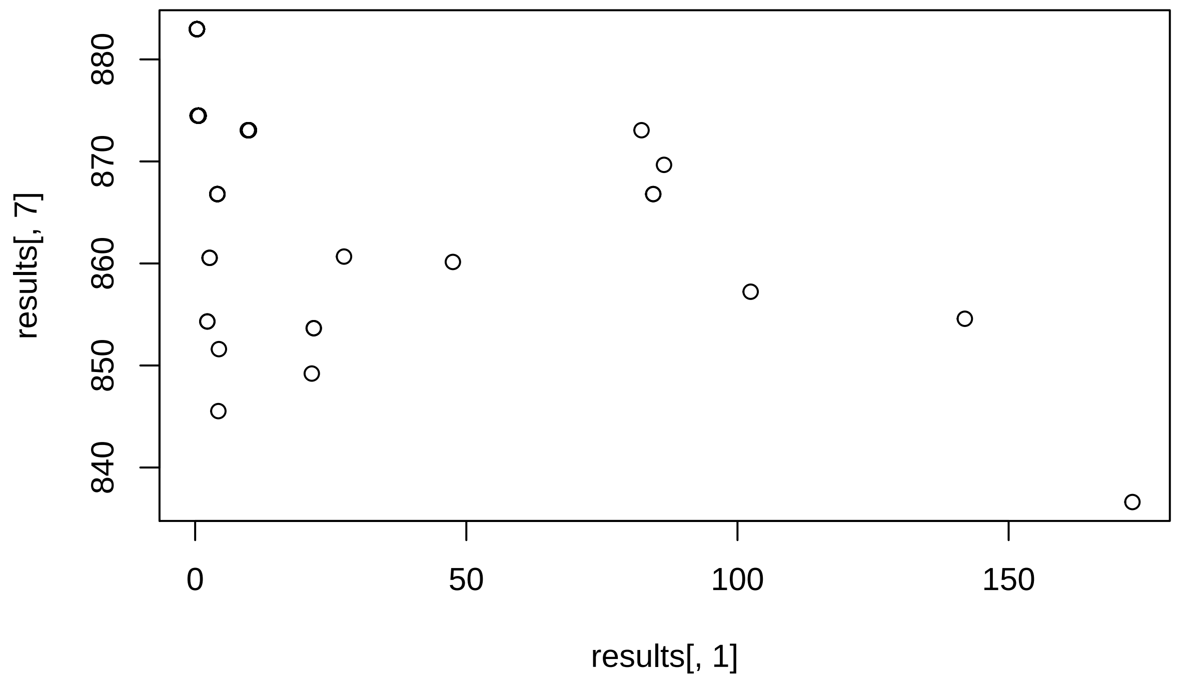
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

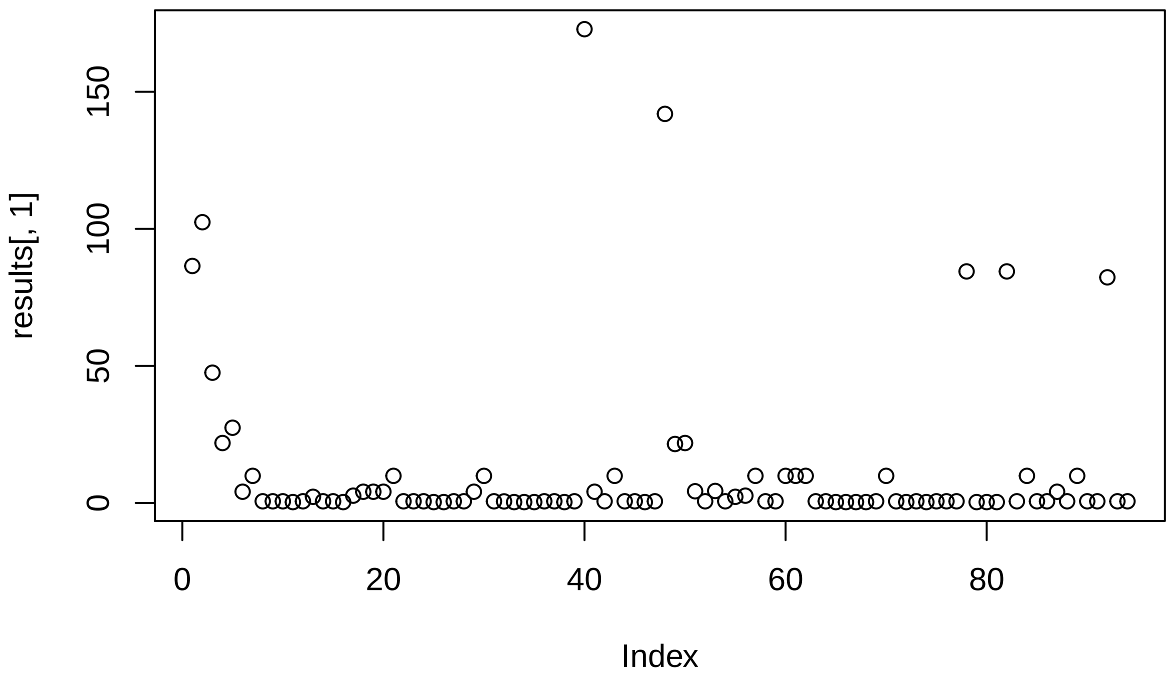




```
##  1 factor completed in 0.00398 minutes.    Estimated time of completion: 2020-04-28 10:37:47    [1]
##  [7]  20.0000010  10.0000010  78.6860377 882.9661846 172.8269683 836.6163349
## [13]  11.0000000  40.0000000
## [1] "j = 30.000001"
```

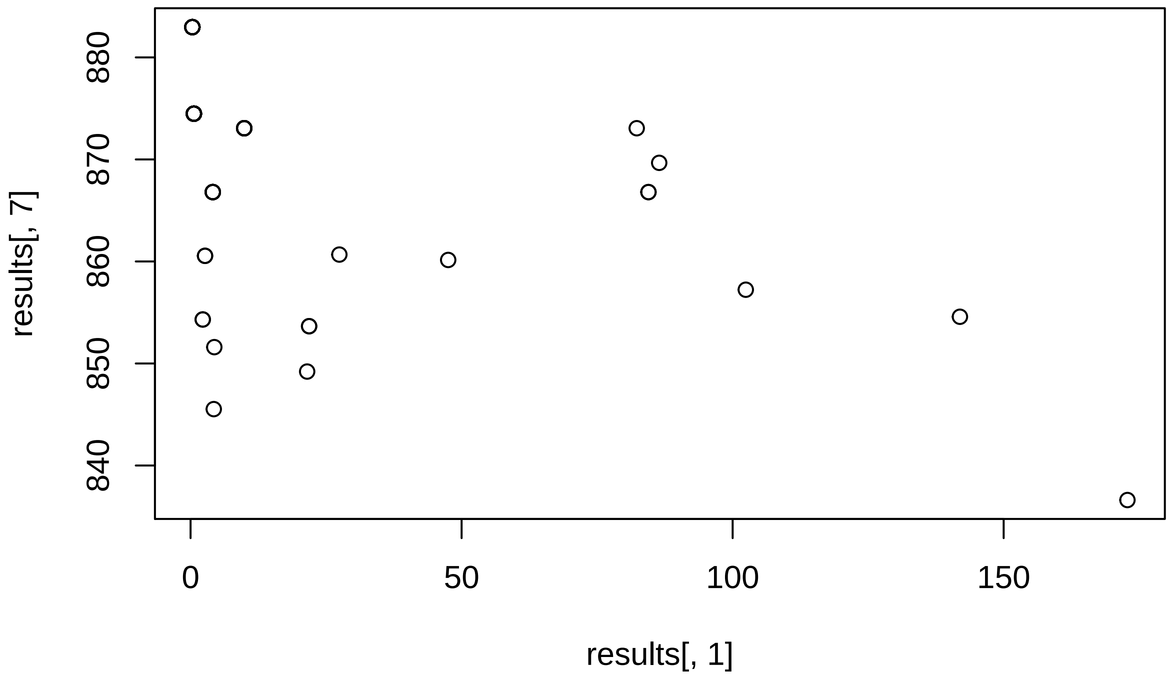
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

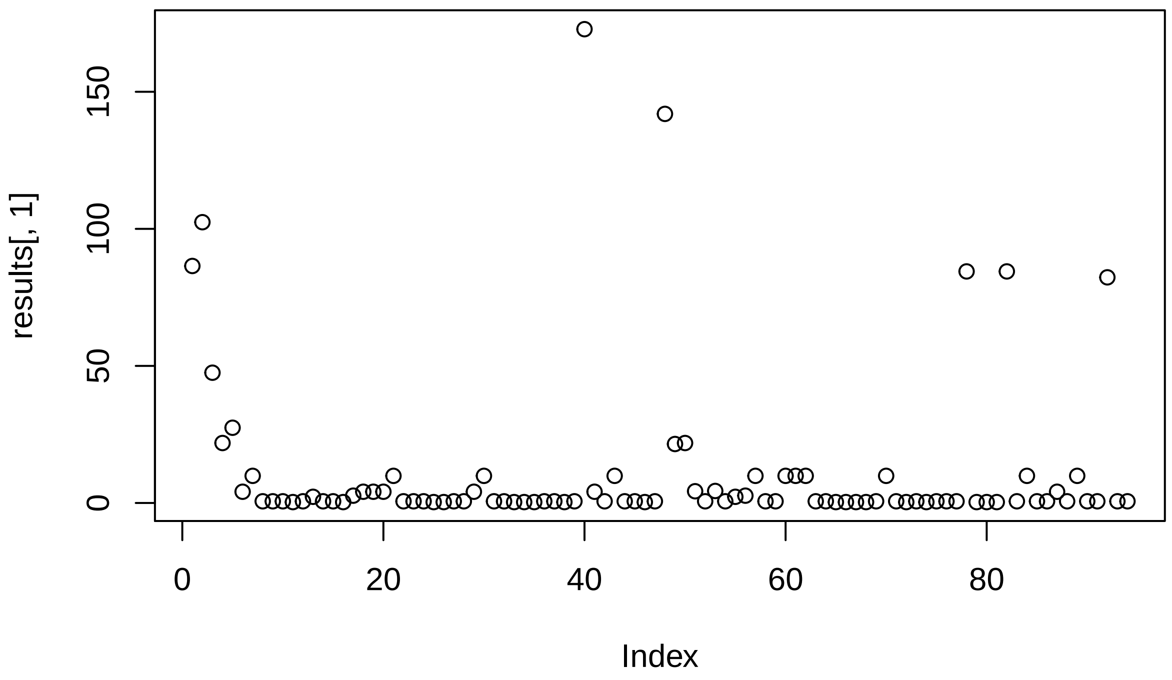




```
##  1 factor completed in 0.00422 minutes.    Estimated time of completion: 2020-04-28 10:37:50    [1]
##  [7]  30.0000010  10.0000010  78.6860377 882.9661846 172.8353436 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

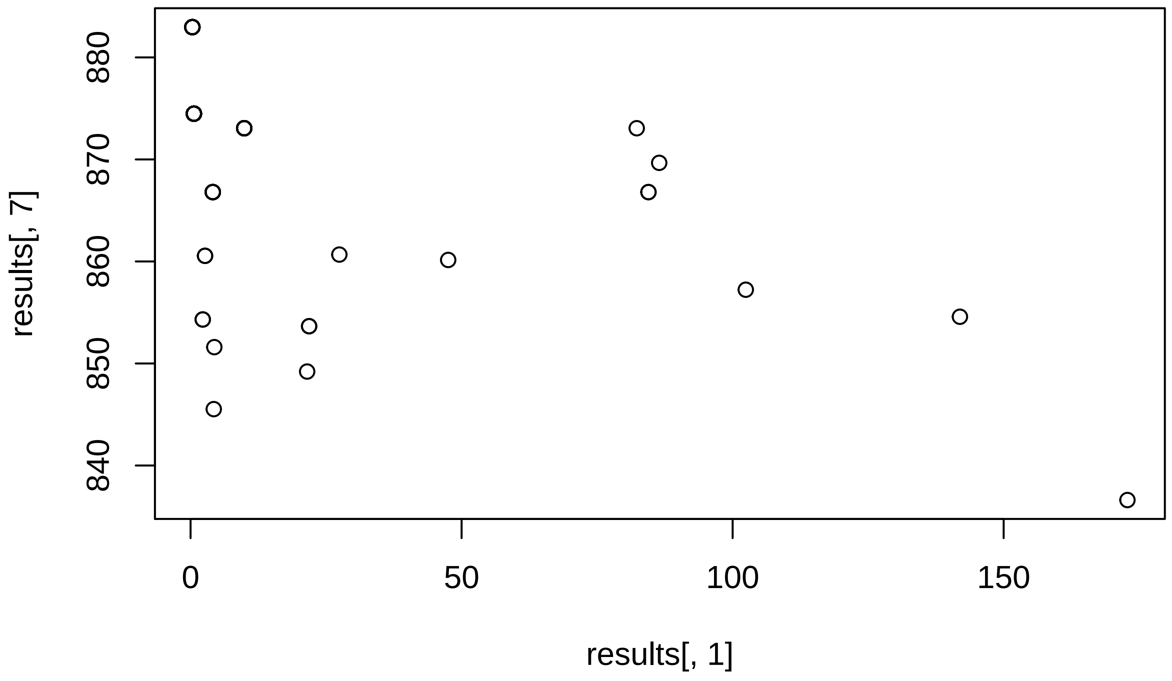
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

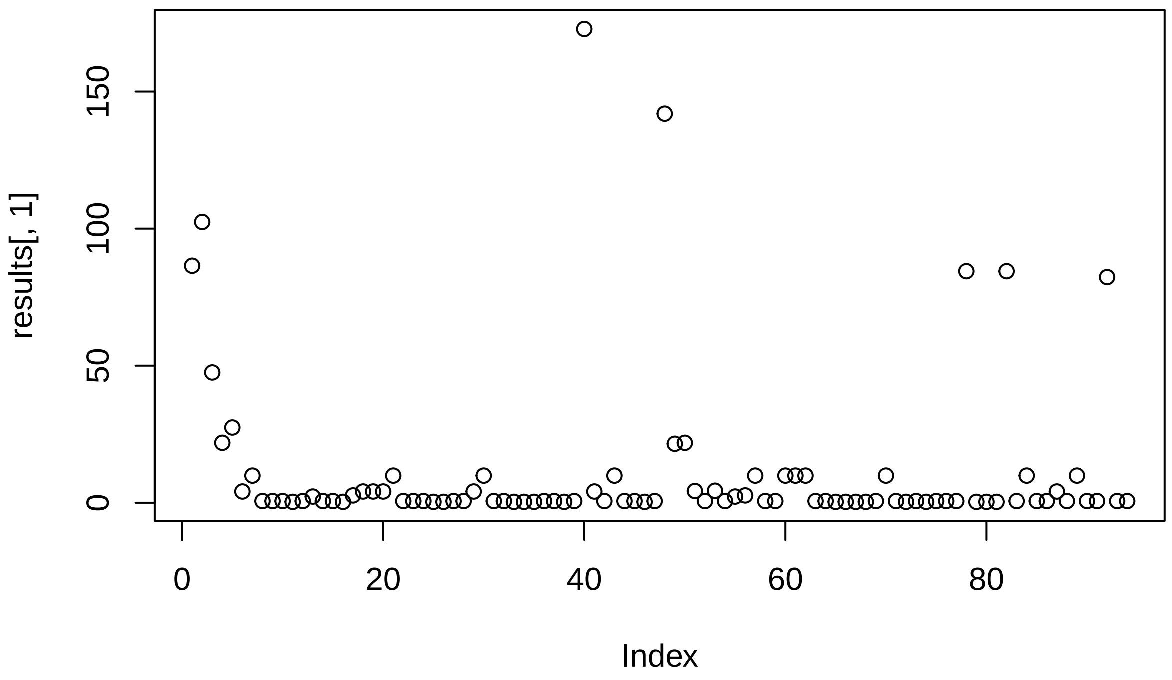




```
##  1 factor completed in 0.00422 minutes.    Estimated time of completion: 2020-04-28 10:37:52    [1]
##  [7]  40.0000010  10.0000010  78.6860377 882.9661846 172.8354181 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

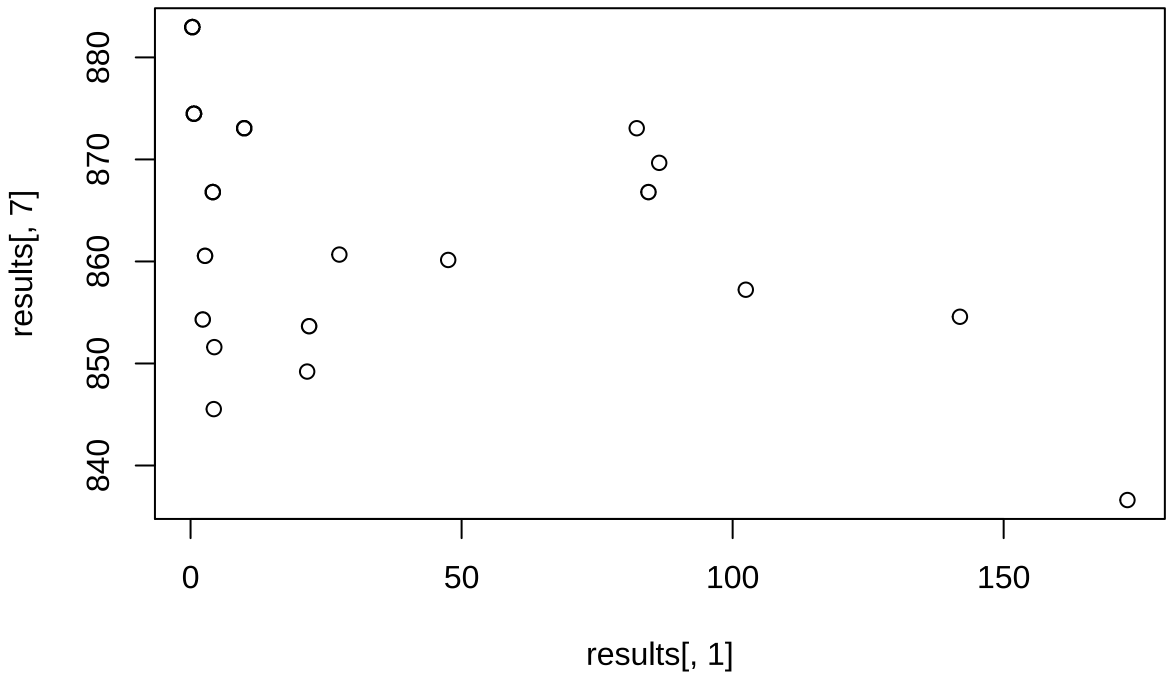
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

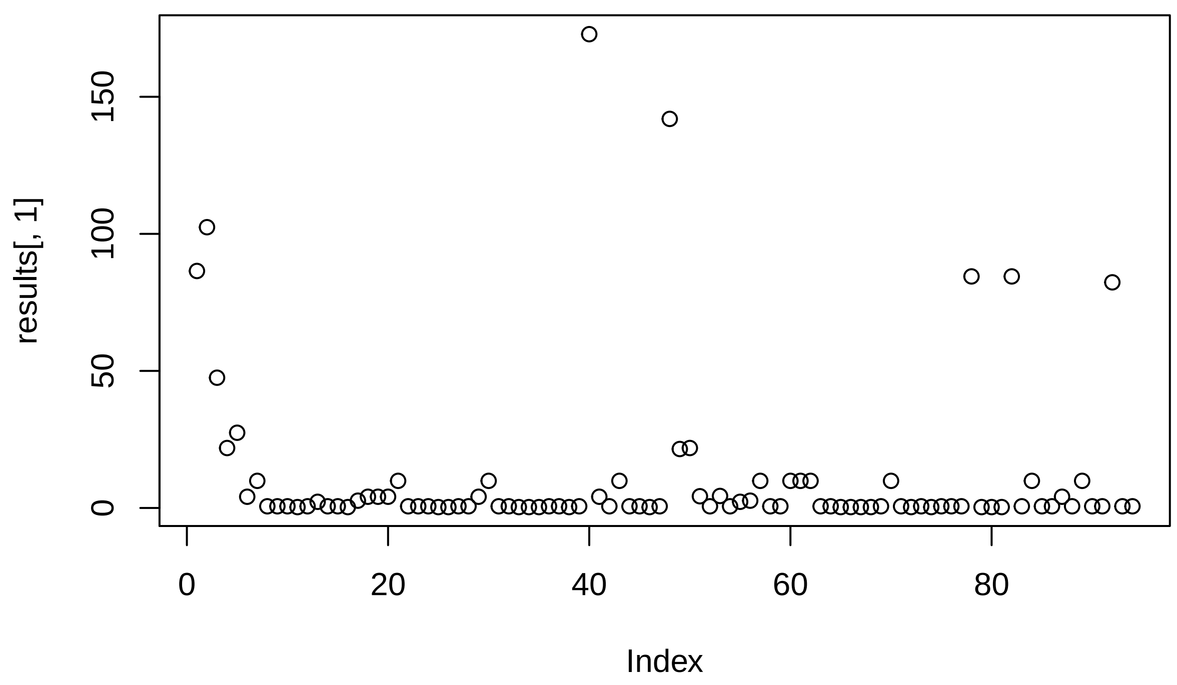




```
##  1 factor completed in 0.00432 minutes.    Estimated time of completion: 2020-04-28 10:37:54    [1]
##  [7]  50.0000010  10.0000010  78.6860377 882.9661846 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

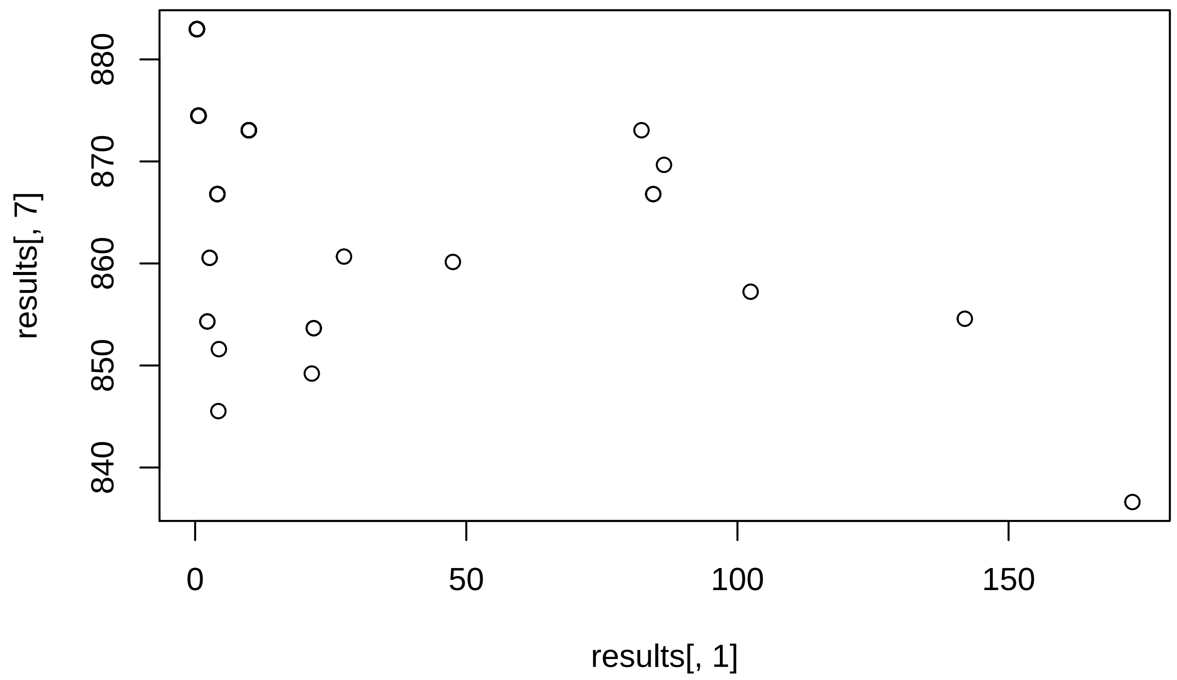
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

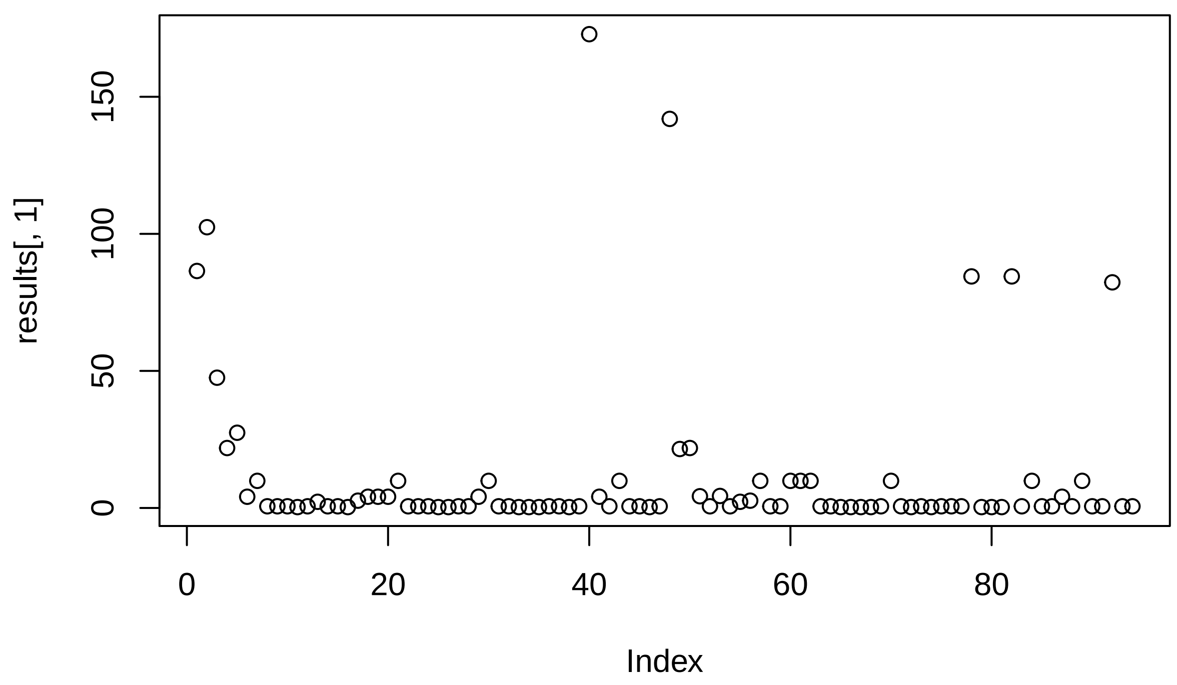




```
##  1 factor completed in 0.00408 minutes.    Estimated time of completion: 2020-04-28 10:37:56    [1]
##  [7]  60.0000010  10.0000010  78.6860377 882.9661846 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

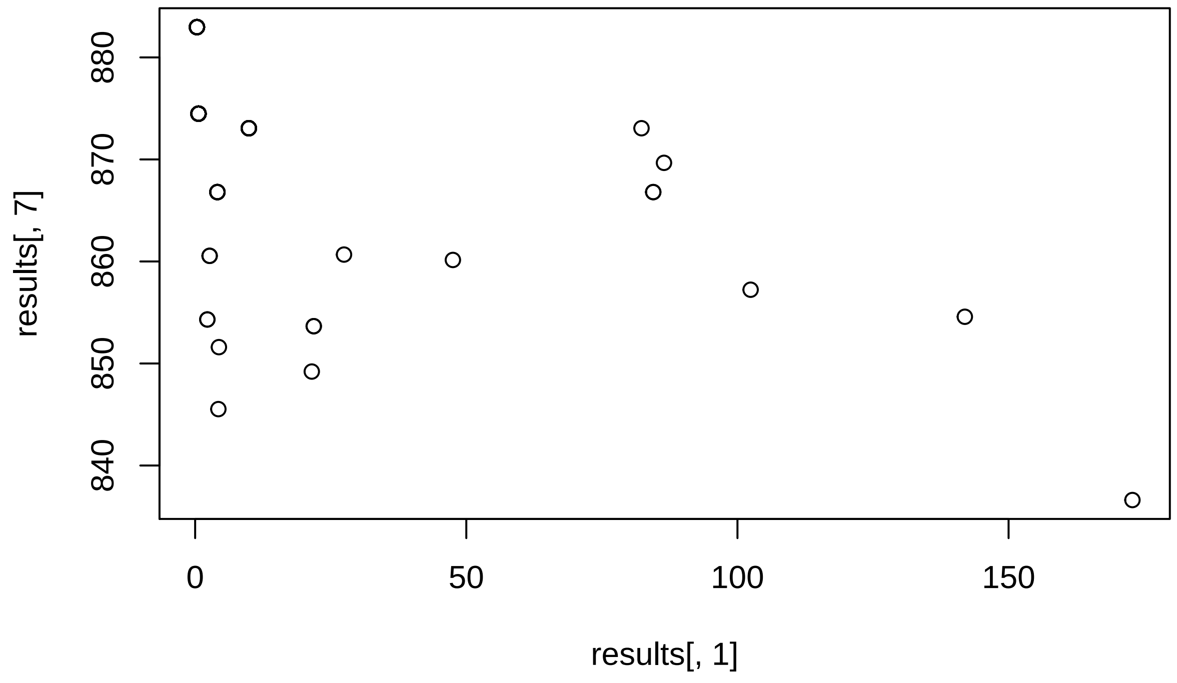
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

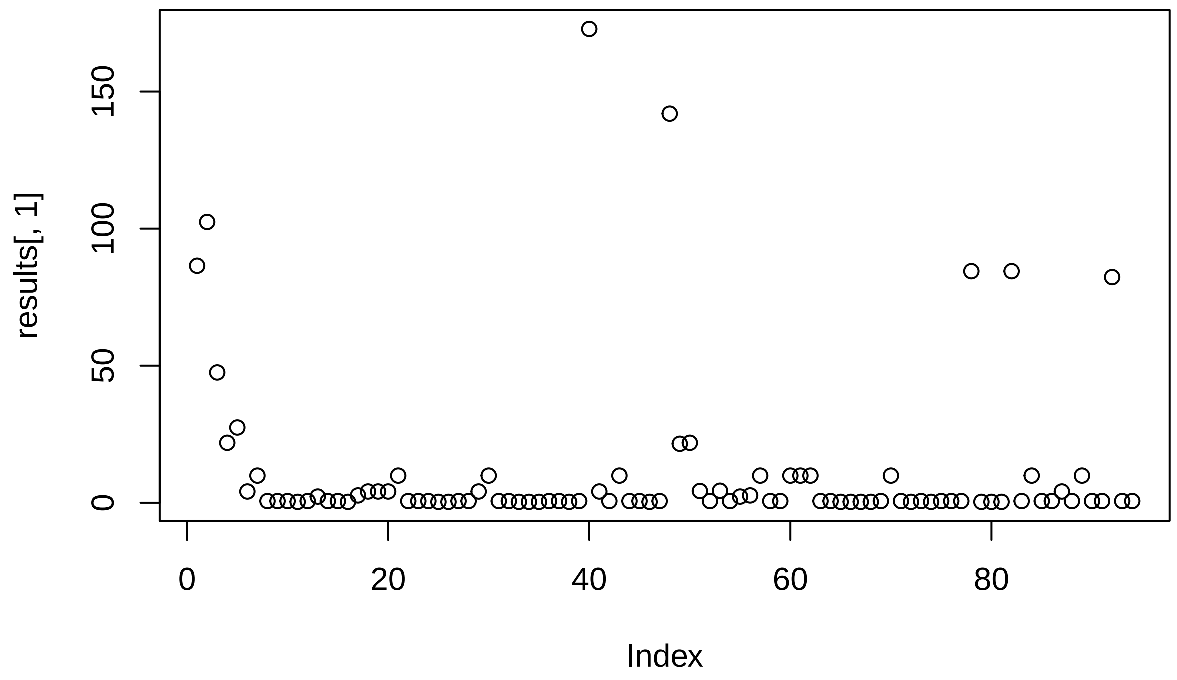




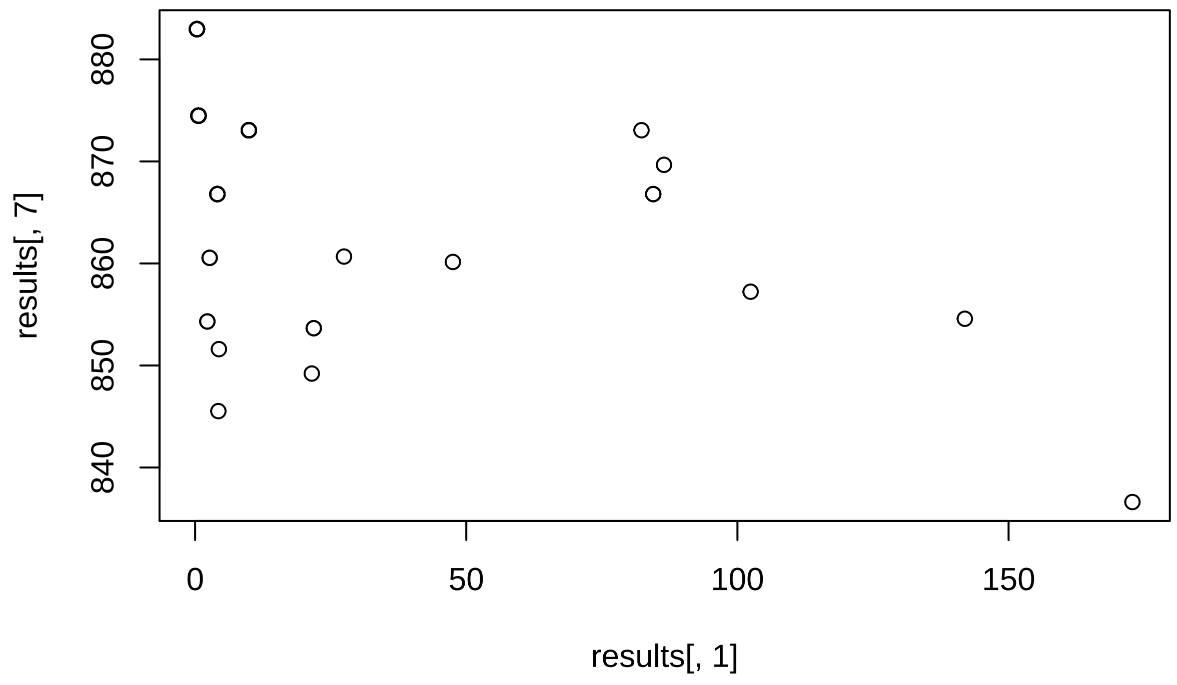
```
##  1 factor completed in 0.004 minutes.    Estimated time of completion: 2020-04-28 10:37:58    [1]
##  [7]  70.0000010  10.0000010  78.6860377 882.9661846 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

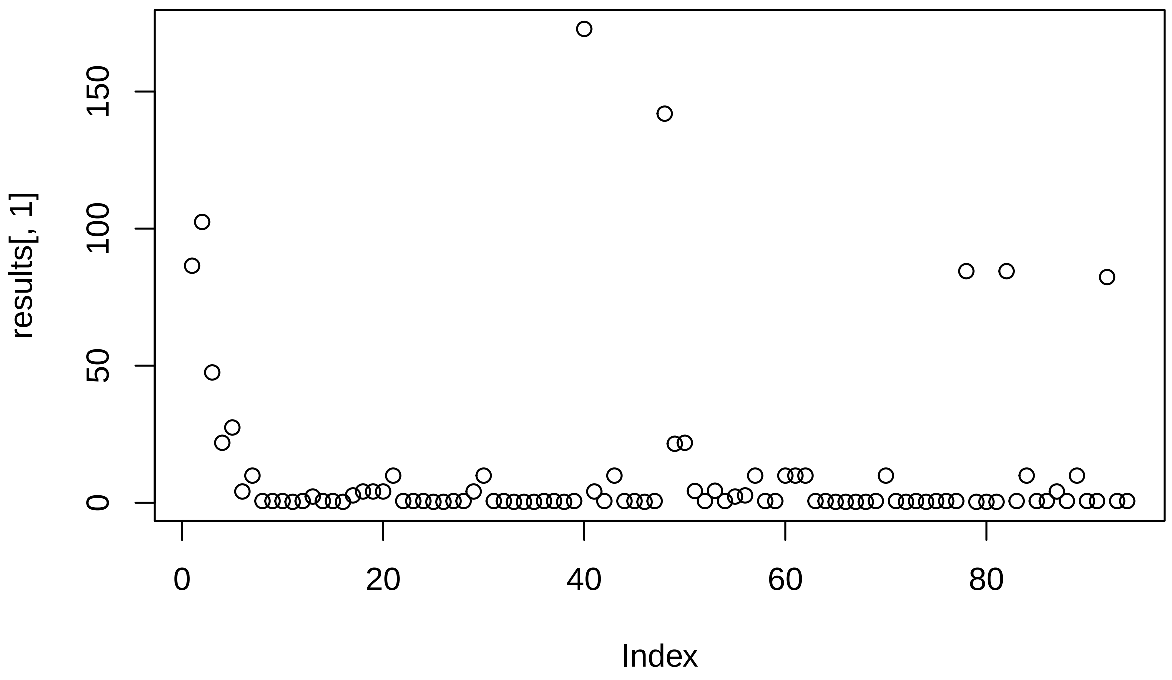
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





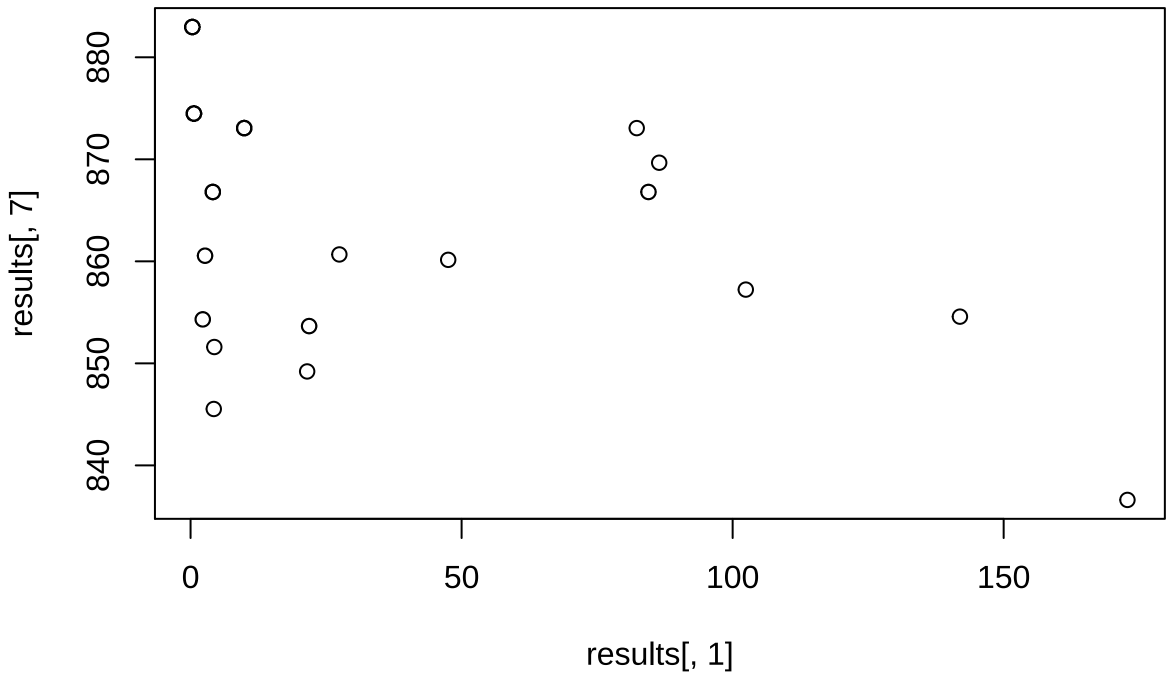
```
##  1 factor completed in 0.0114 minutes.    Estimated time of completion: 2020-04-28 10:38:01    [1]
##  [7]  80.0000010  10.0000010  78.6860377 882.9661846 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

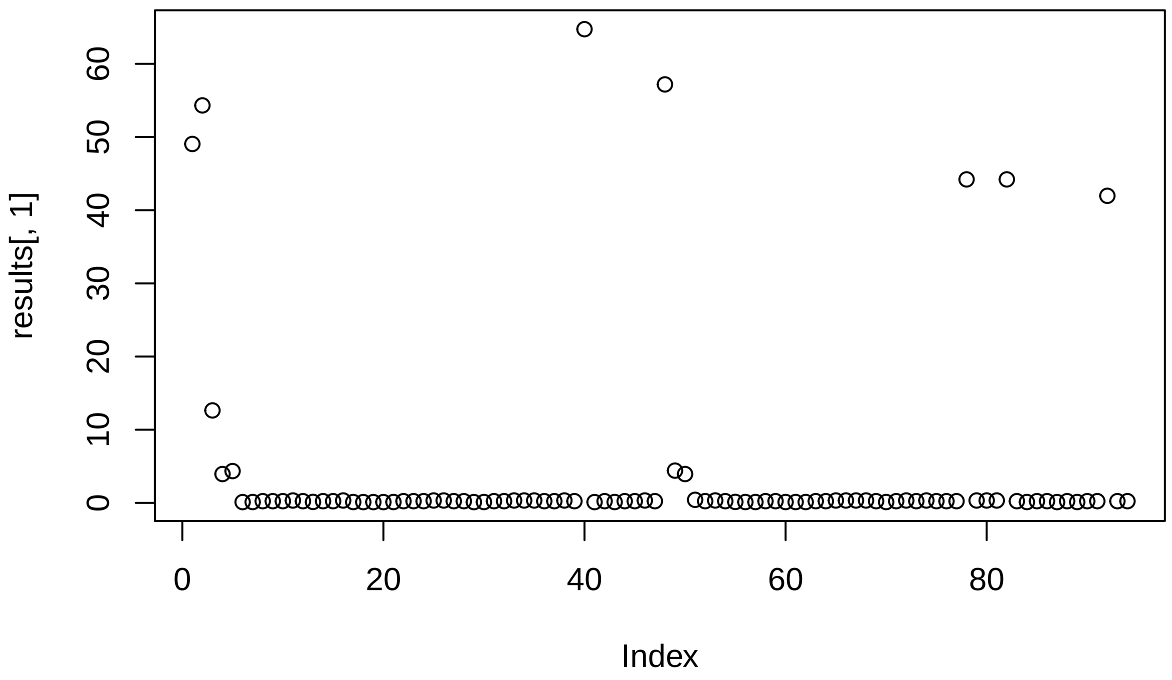




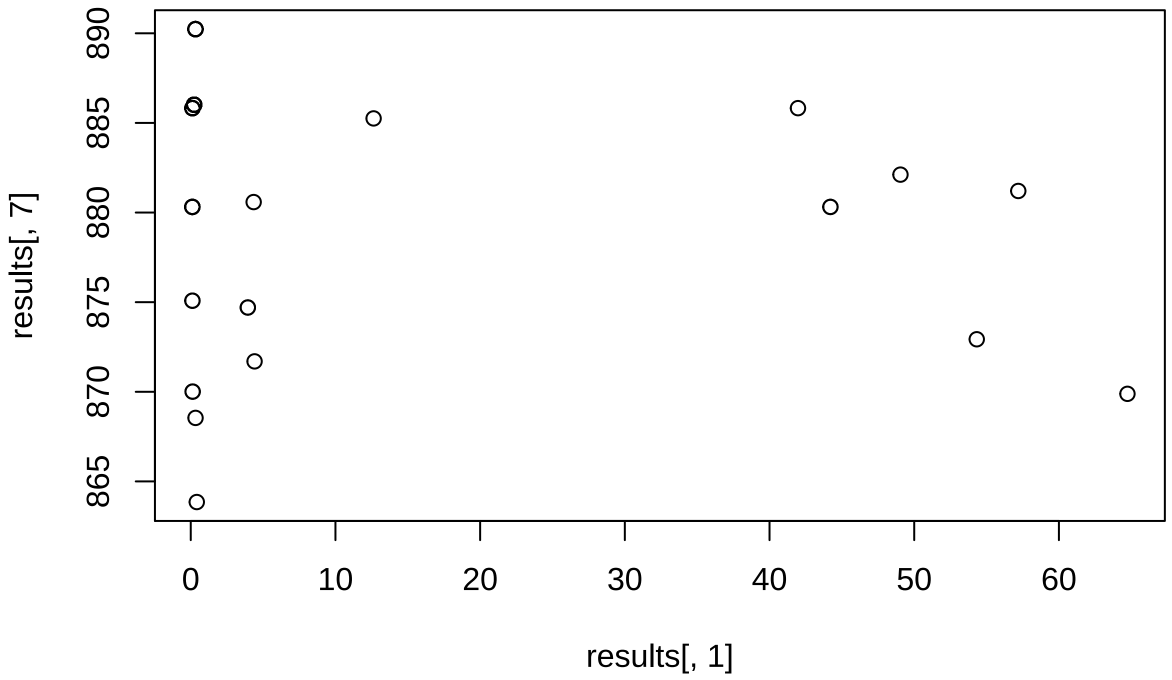
```
##  1 factor completed in 0.0213 minutes.    Estimated time of completion: 2020-04-28 10:38:06    [1]
##  [7]  90.0000010  10.0000010  78.6860377 882.9661846 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 20.000001"
## [1] "j = 1e-06"
```

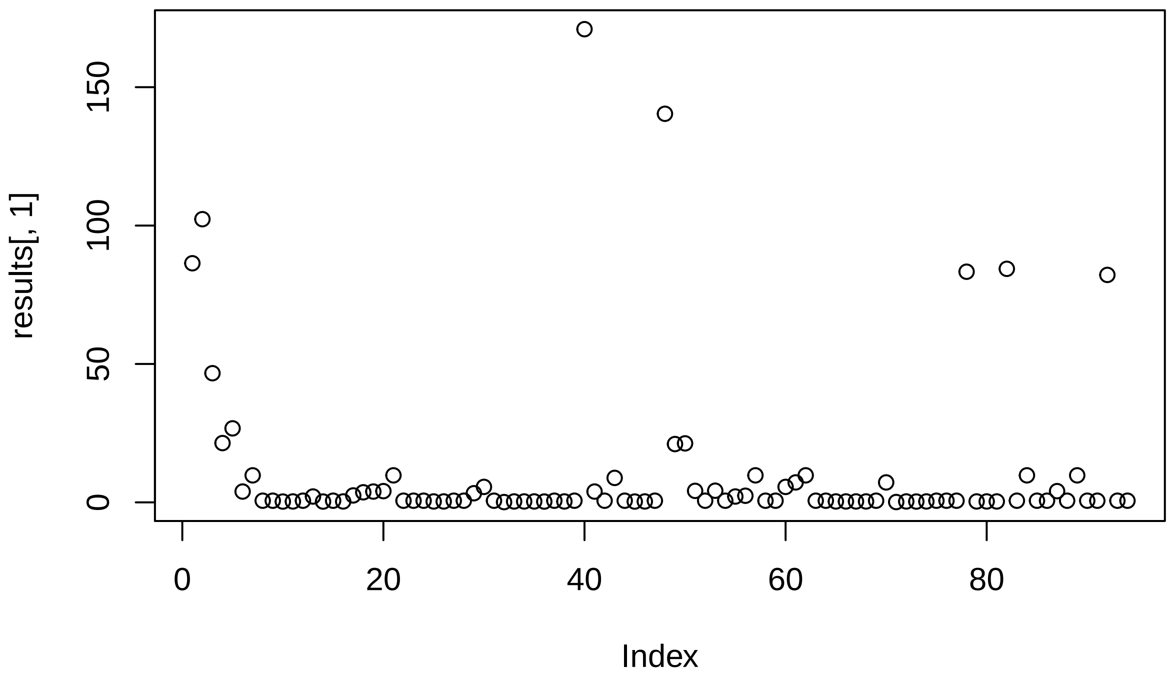
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





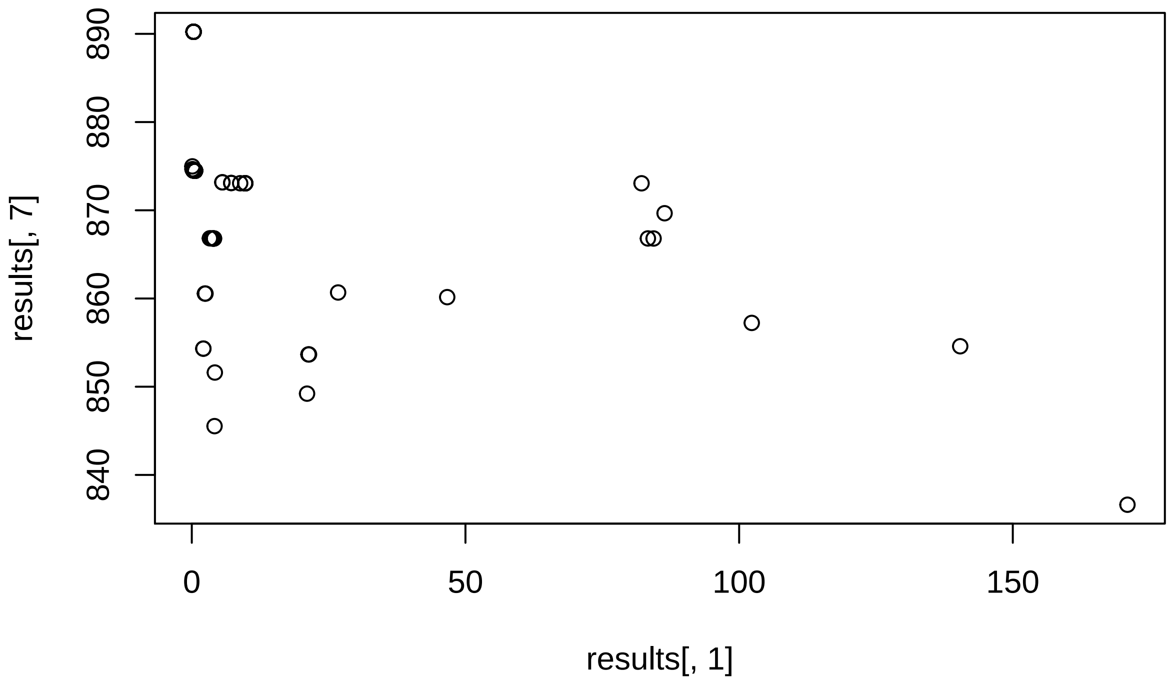
```
##  1 factor completed in 0.00854 minutes.    Estimated time of completion: 2020-04-28 10:38:10    [1]
##  [7]   0.0000010  20.0000010  78.6860377 880.3131050   0.4176145 863.8497453
## [13]  29.0000000  51.0000000
## [1] "j = 10.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

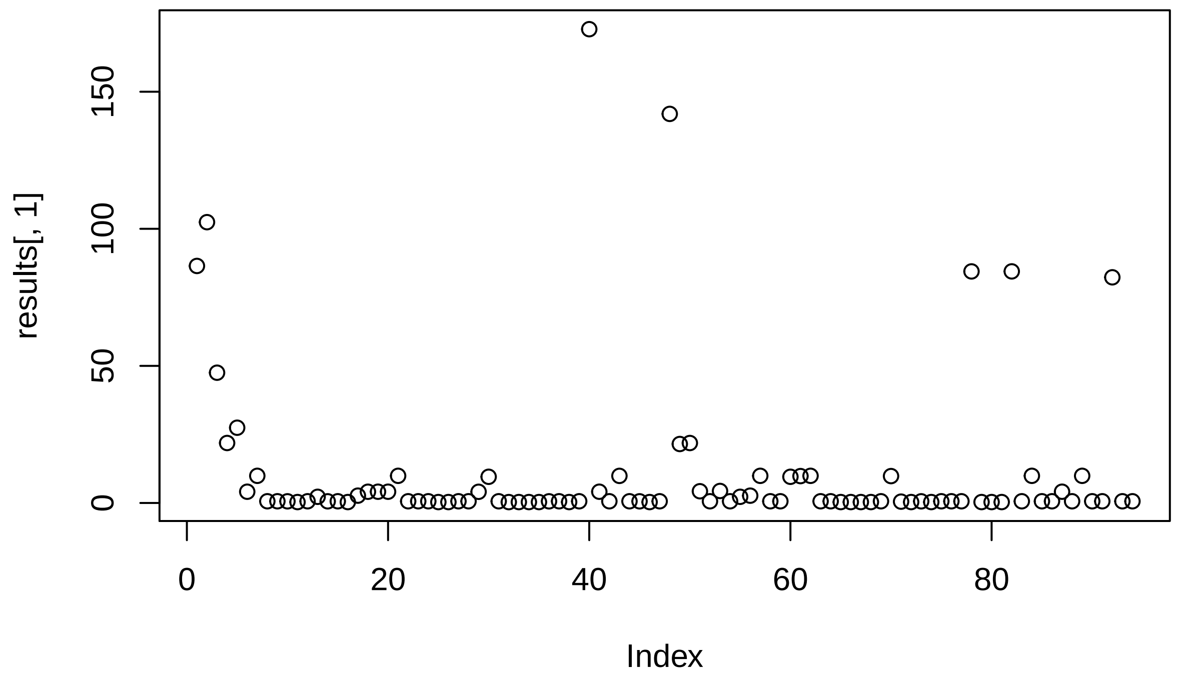




```
##  1 factor completed in 0.00504 minutes.    Estimated time of completion: 2020-04-28 10:38:13    [1]
##  [7]  10.0000010  20.0000010  78.6860377 874.9774719 170.9505218 836.6229413
## [13]  32.0000000  40.0000000
## [1] "j = 20.000001"
```

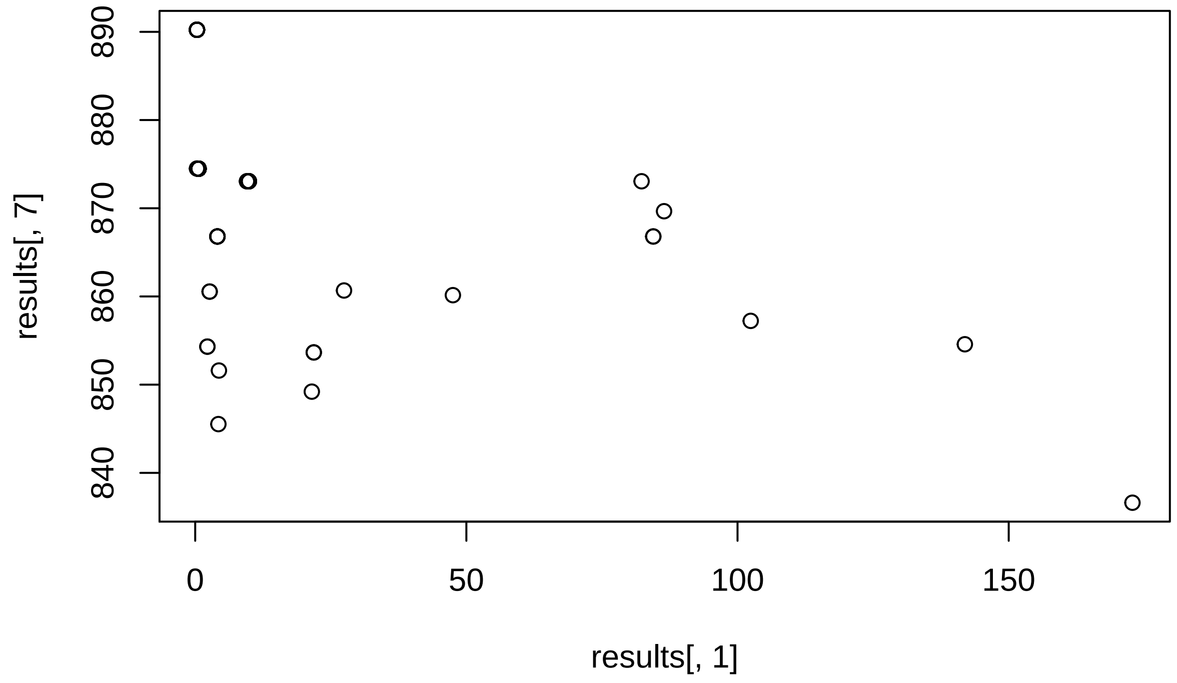
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

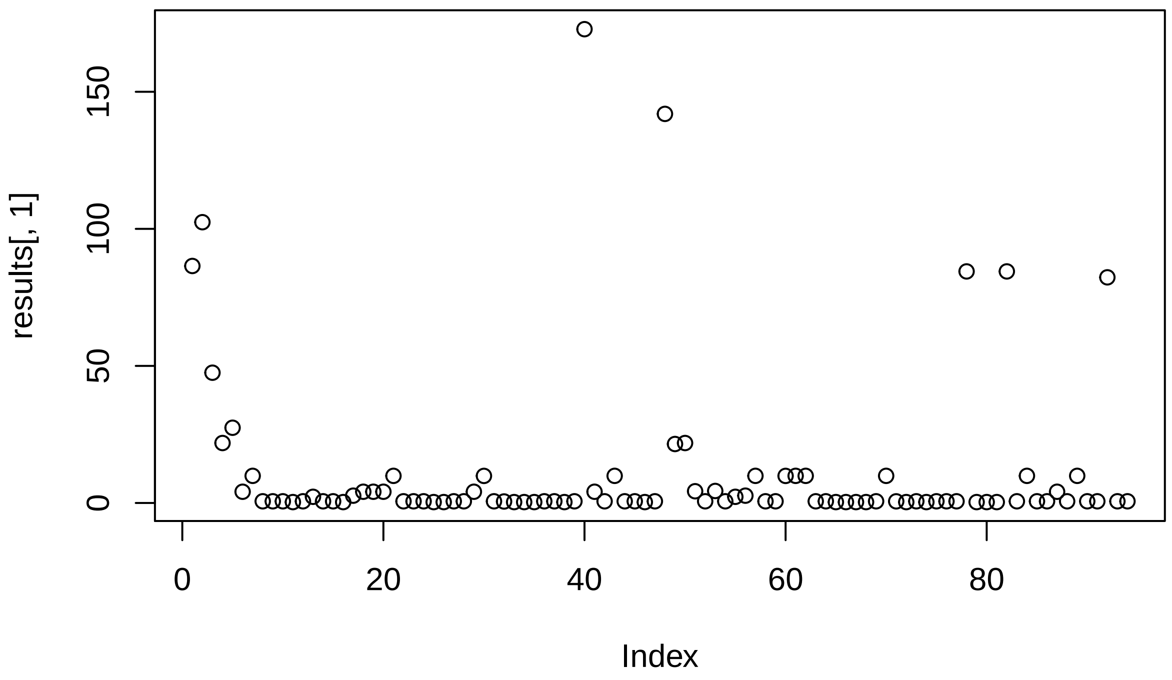




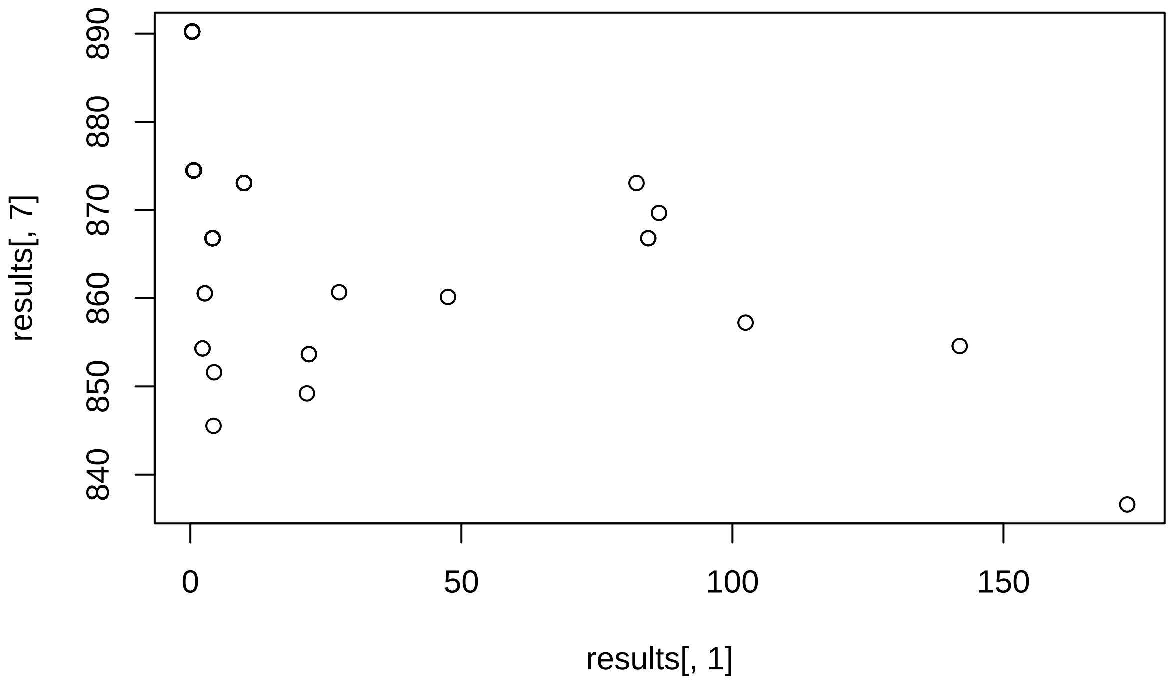
```
##  1 factor completed in 0.00536 minutes.    Estimated time of completion: 2020-04-28 10:38:16    [1]
##  [7]  20.0000010  20.0000010  78.6860377 874.5006172 172.8185185 836.6163353
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

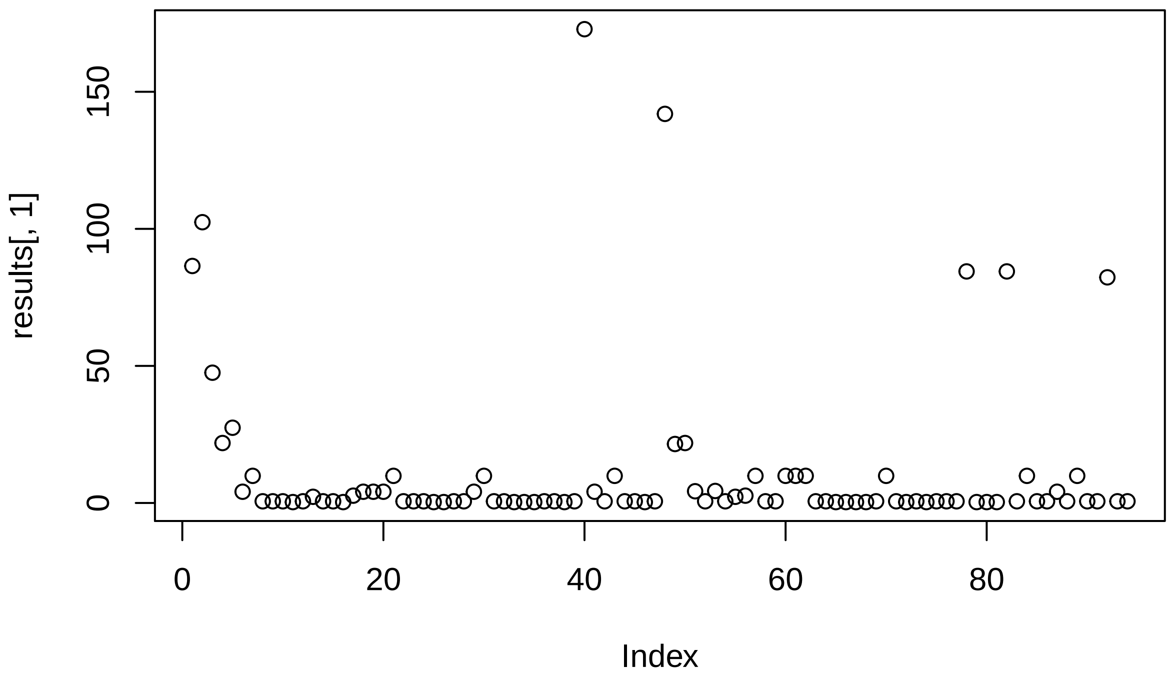
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





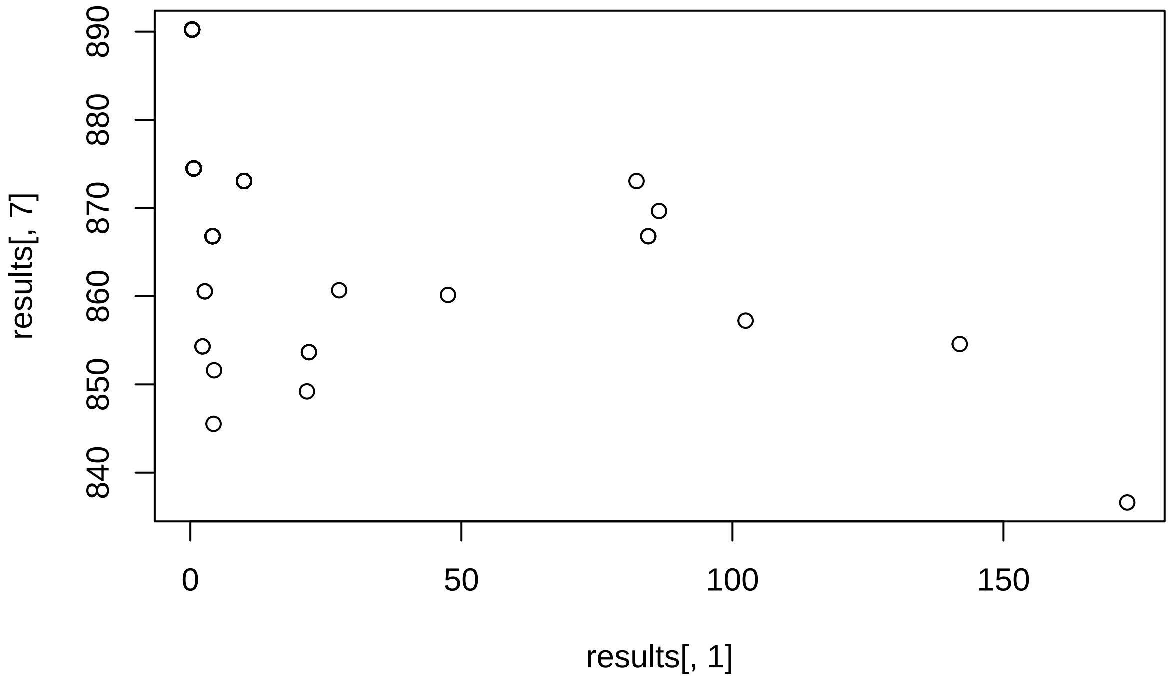
```
##  1 factor completed in 0.00687 minutes.    Estimated time of completion: 2020-04-28 10:38:19    [1]
##  [7]  30.0000010  20.0000010  78.6860377 890.2312897 172.8352684 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

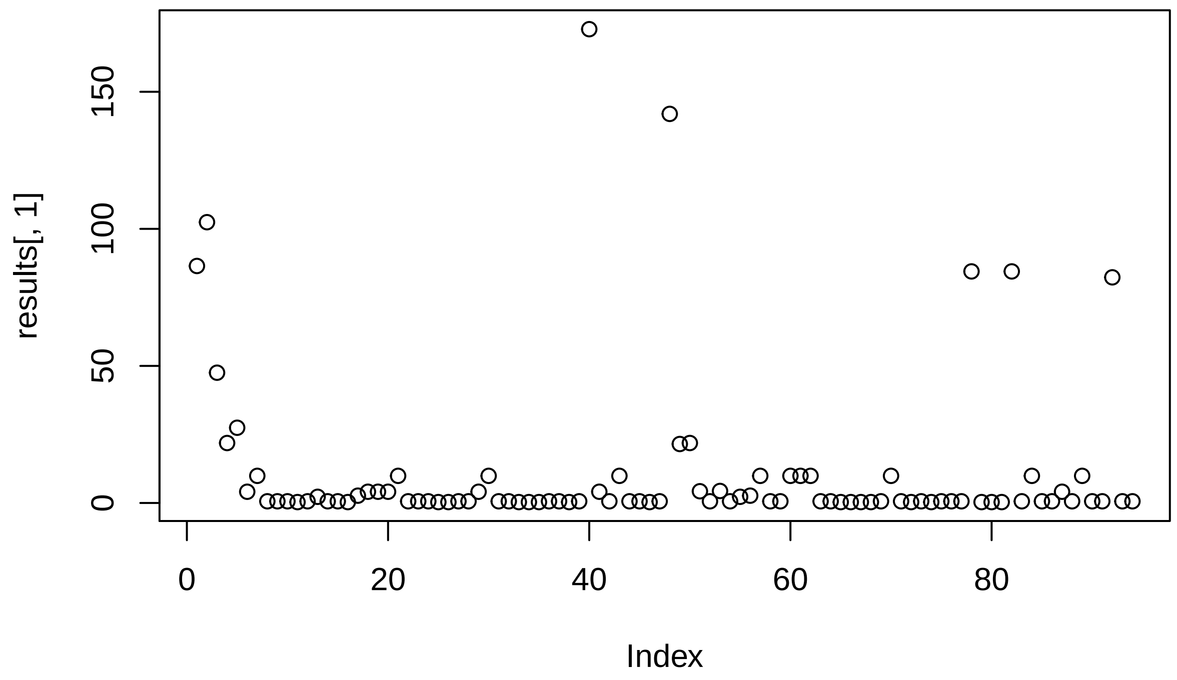




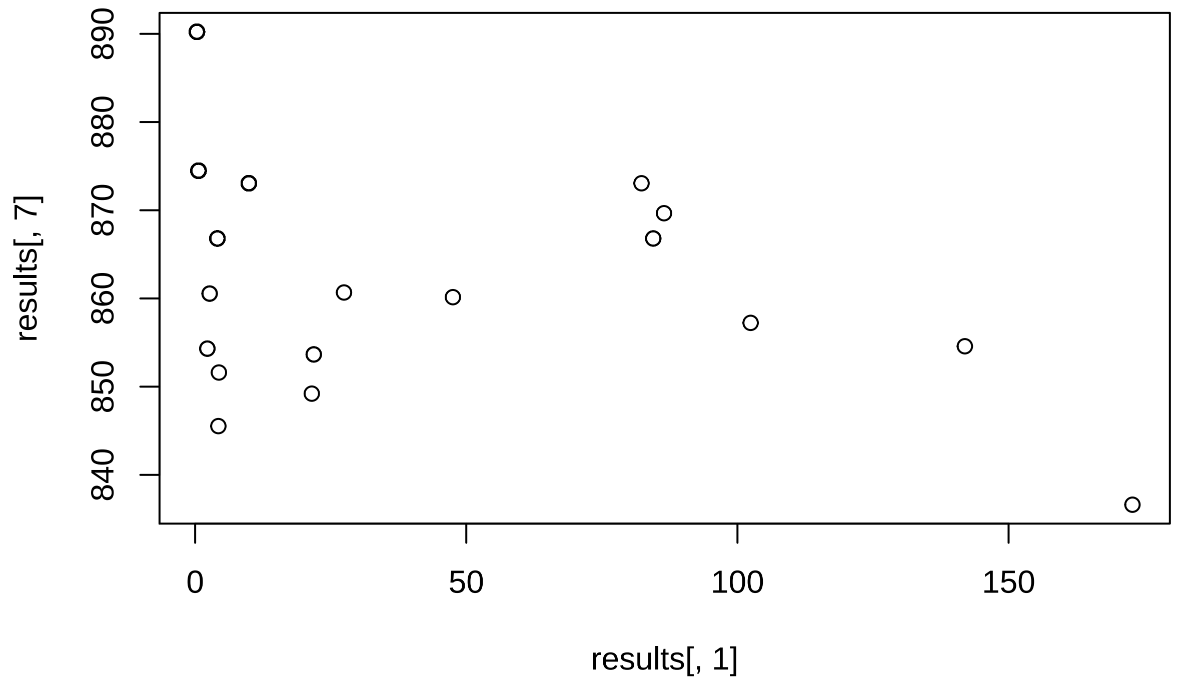
```
##  1 factor completed in 0.00641 minutes.    Estimated time of completion: 2020-04-28 10:38:22    [1]
##  [7]  40.0000010  20.0000010  78.6860377 890.2312897 172.8354174 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

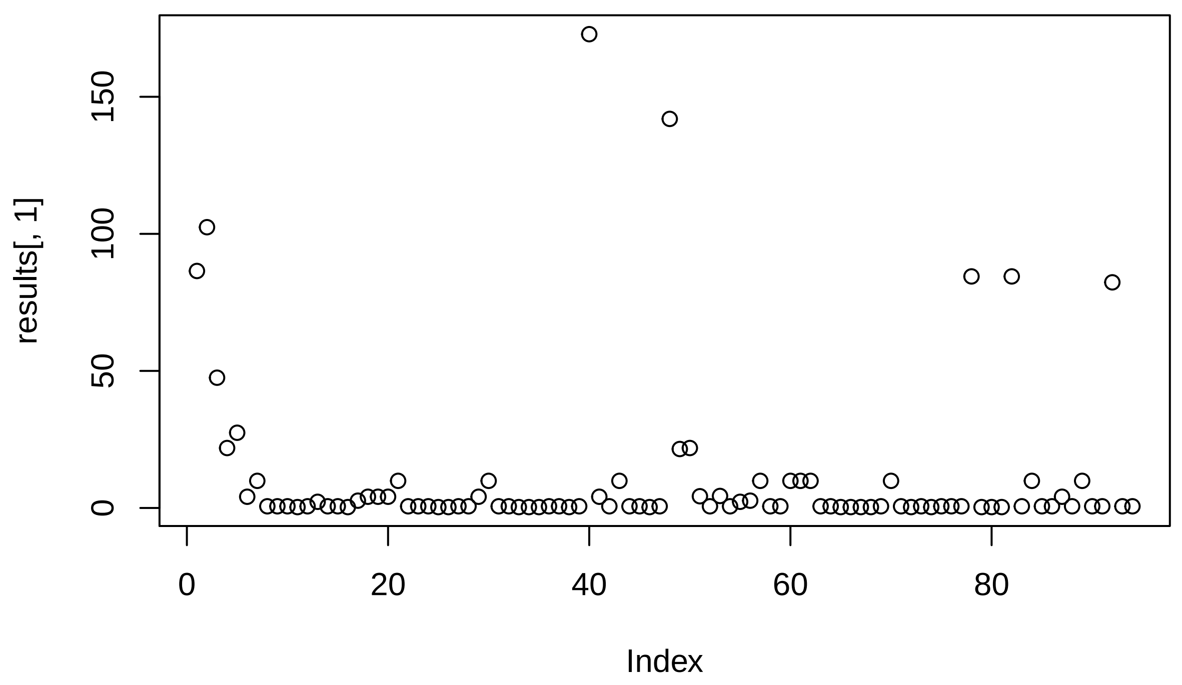
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





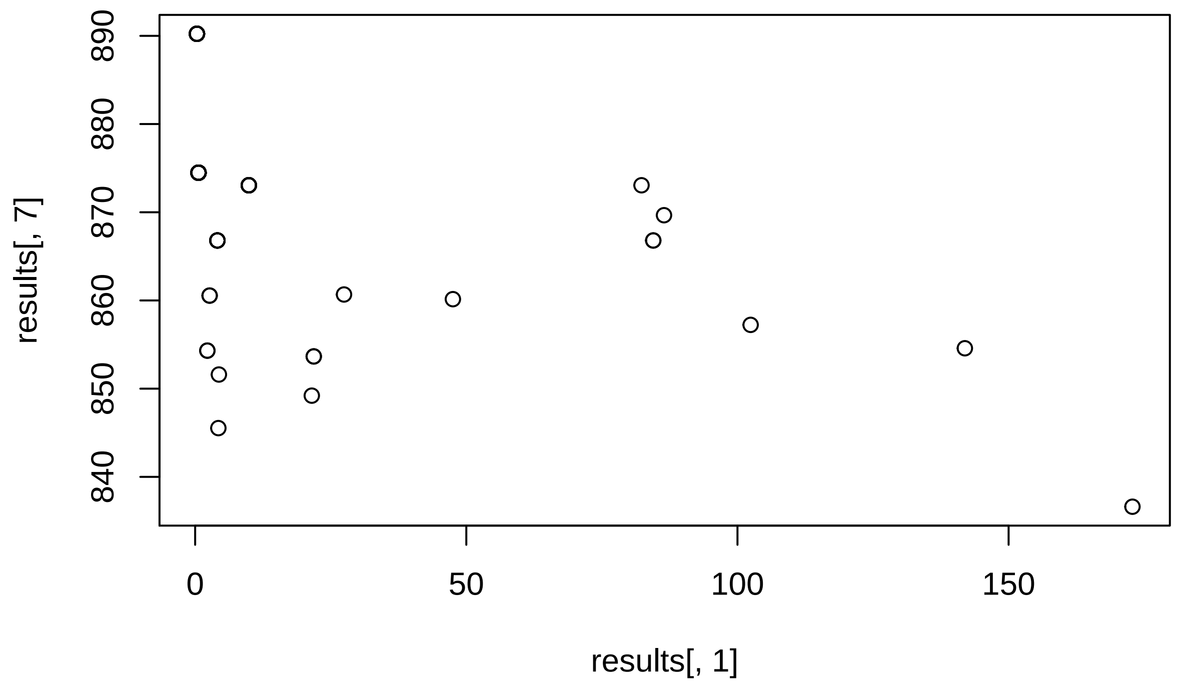
```
##  1 factor completed in 0.00476 minutes.    Estimated time of completion: 2020-04-28 10:38:24    [1]
##  [7]  50.0000010  20.0000010  78.6860377 890.2312897 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

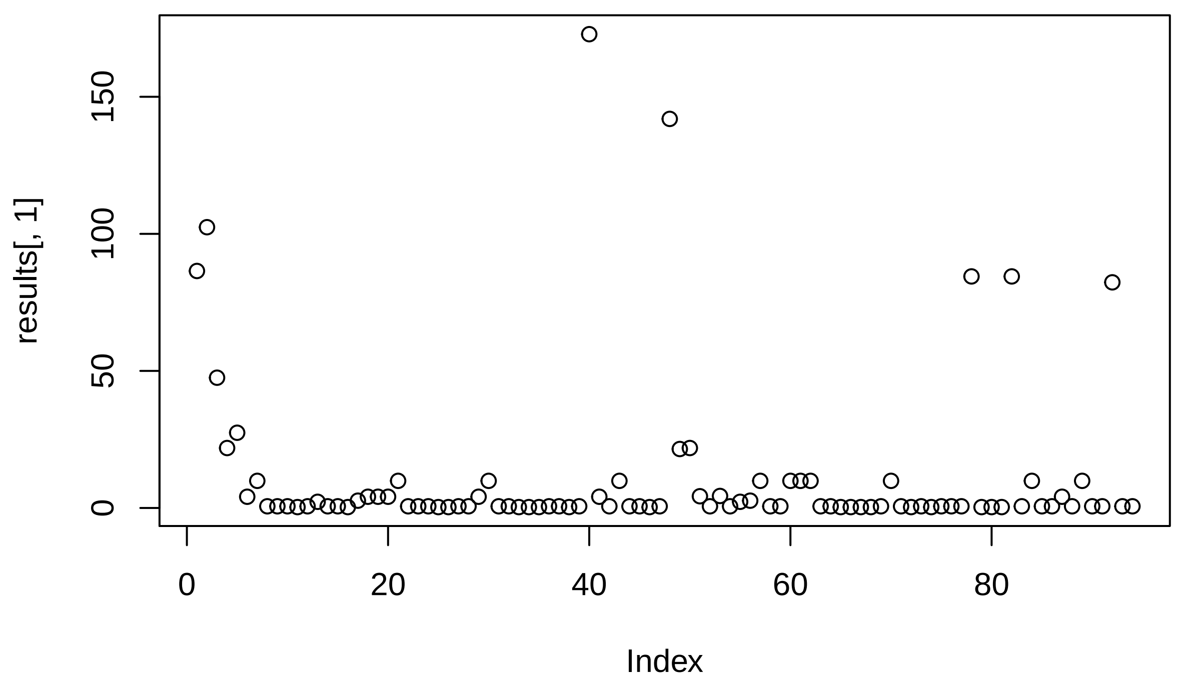




```
##  1 factor completed in 0.00477 minutes.    Estimated time of completion: 2020-04-28 10:38:27    [1]
##  [7]  60.0000010  20.0000010  78.6860377 890.2312897 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

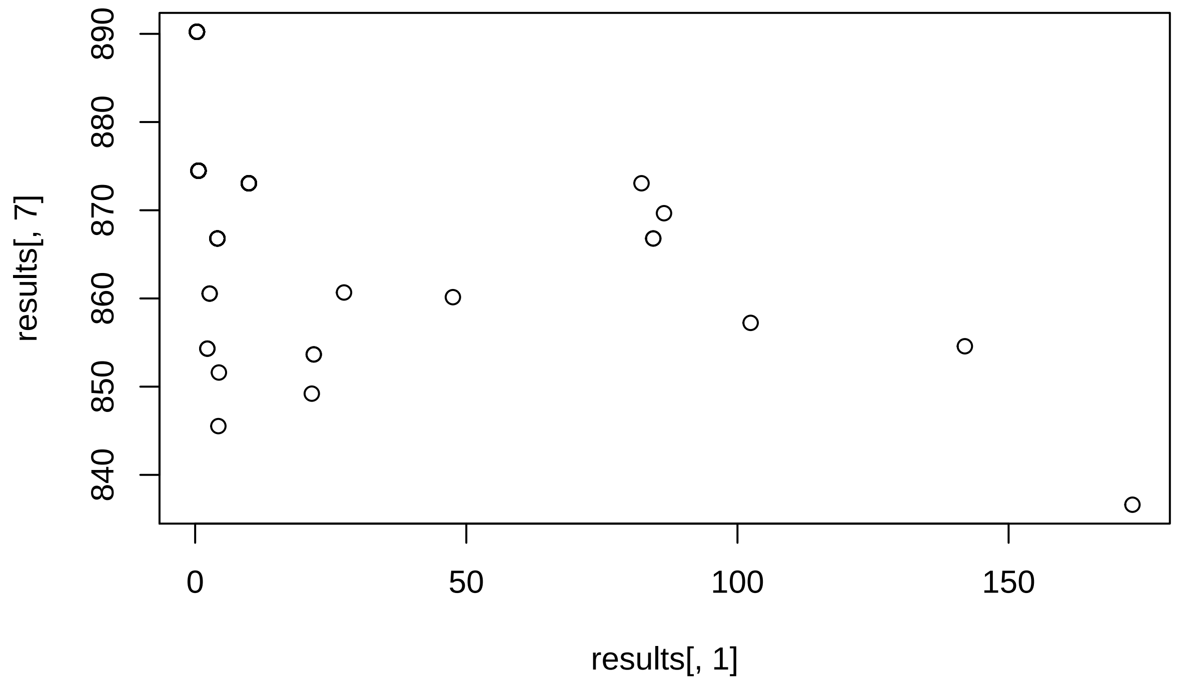
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

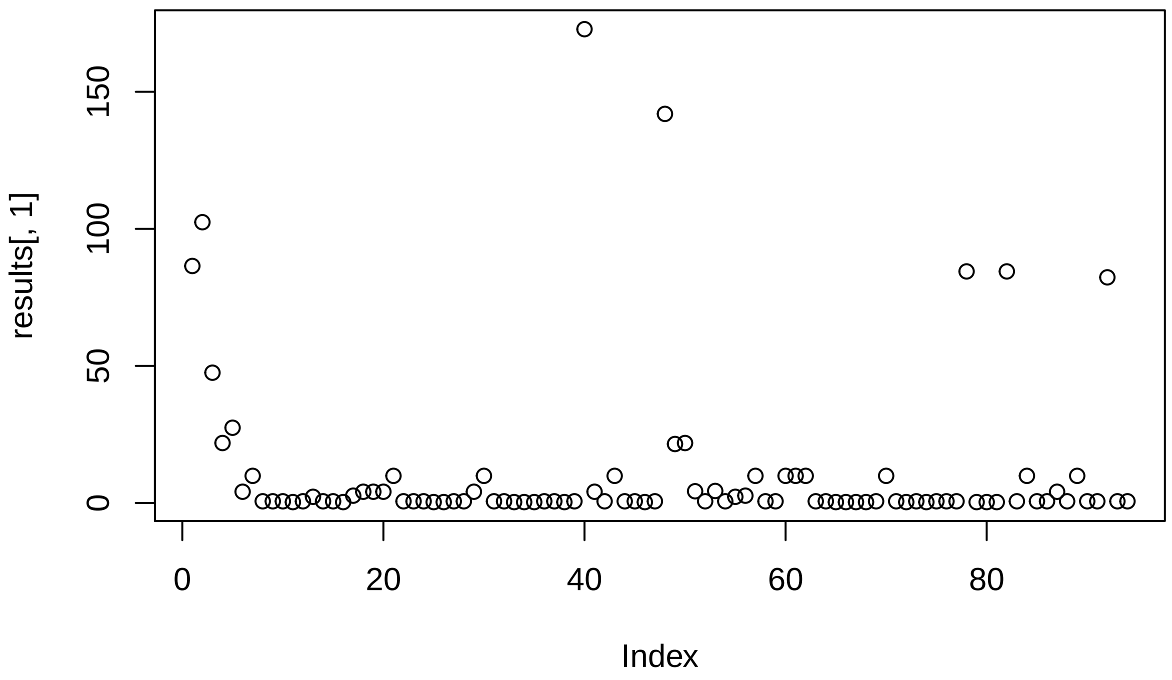




```
##  1 factor completed in 0.00493 minutes.    Estimated time of completion: 2020-04-28 10:38:30    [1]
##  [7]  70.0000010  20.0000010  78.6860377 890.2312897 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

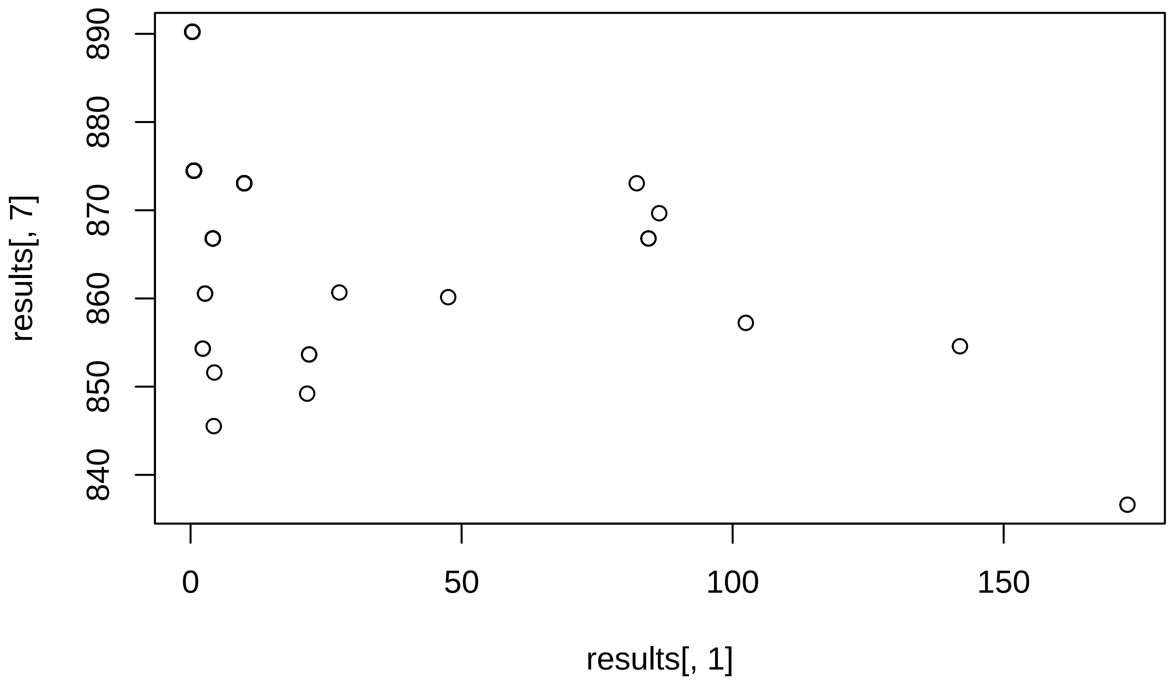
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

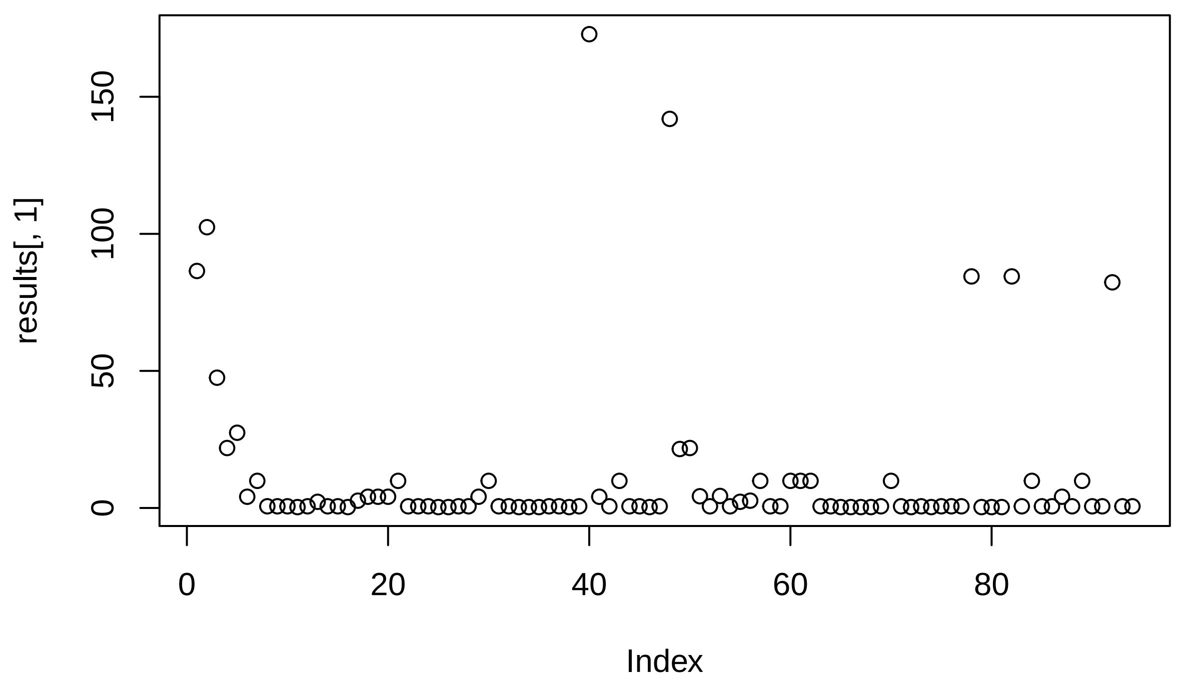




```
##  1 factor completed in 0.00442 minutes.    Estimated time of completion: 2020-04-28 10:38:33    [1]
##  [7]  80.0000010  20.0000010  78.6860377 890.2312897 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

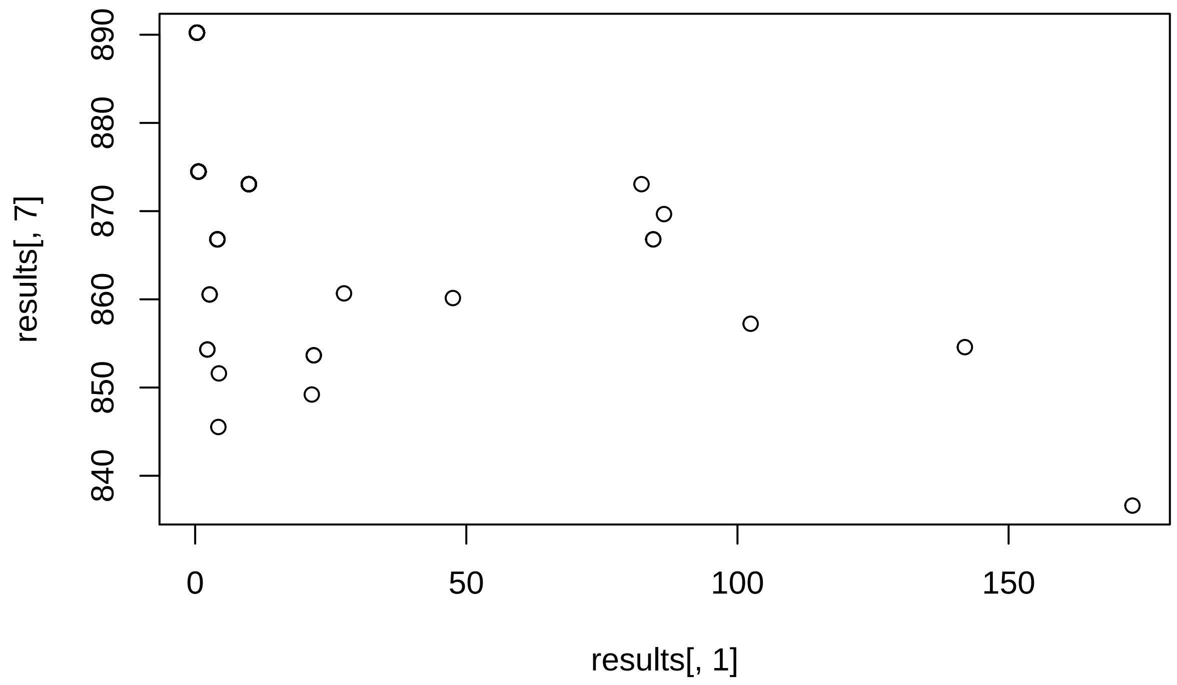
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

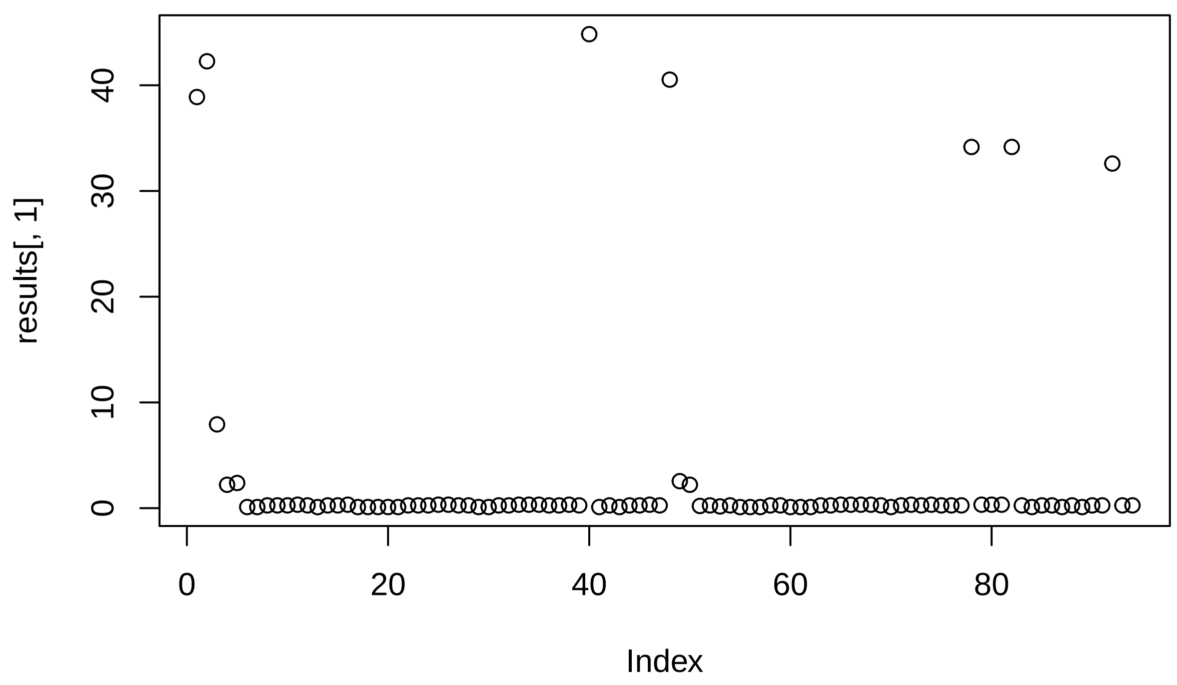




```
## 1 factor completed in 0.00475 minutes.    Estimated time of completion: 2020-04-28 10:38:35    [1]
## [7]  90.0000010  20.0000010  78.6860377 890.2312897 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 30.000001"
## [1] "j = 1e-06"
```

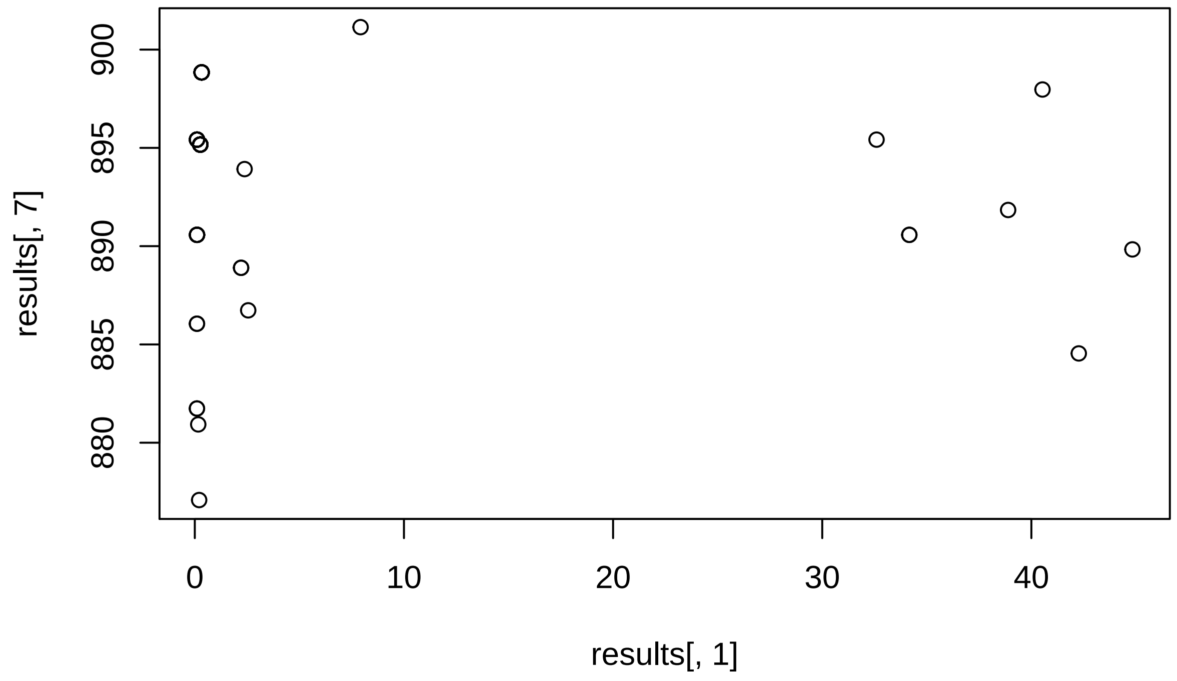
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

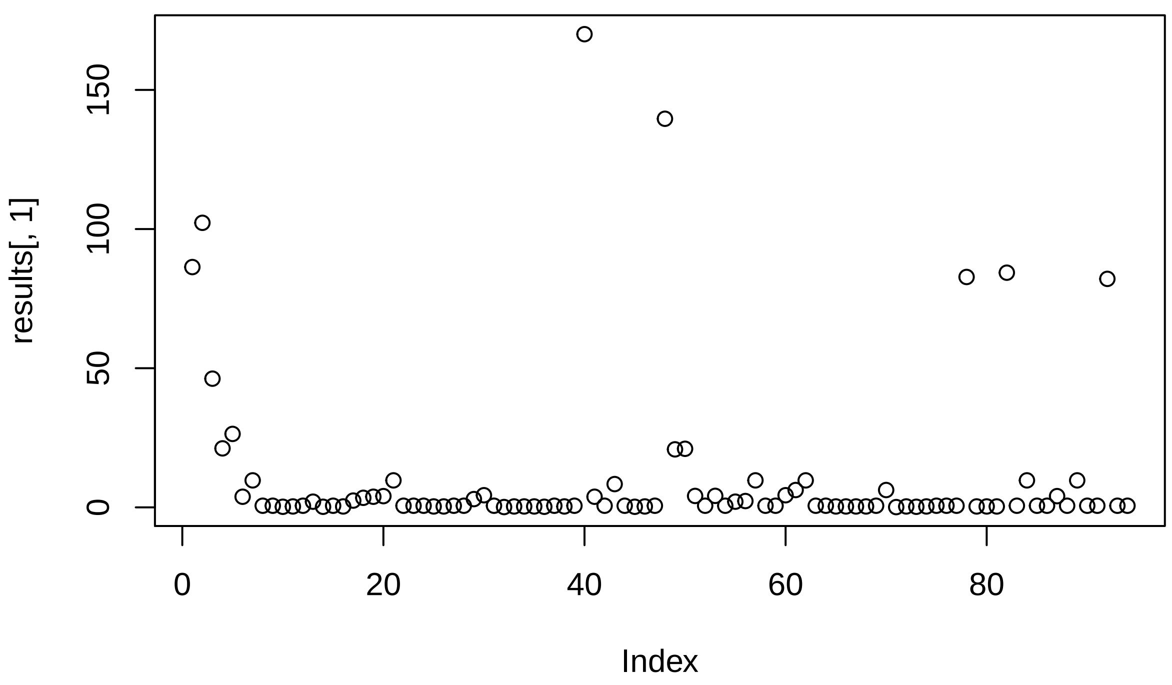




```
##  1 factor completed in 0.00468 minutes.    Estimated time of completion: 2020-04-28 10:38:37    [1]
##  [7]   0.0000010  30.0000010  78.6860377 881.7398047   0.2080415 877.0832675
## [13]  13.0000000  51.0000000
## [1] "j = 10.000001"
```

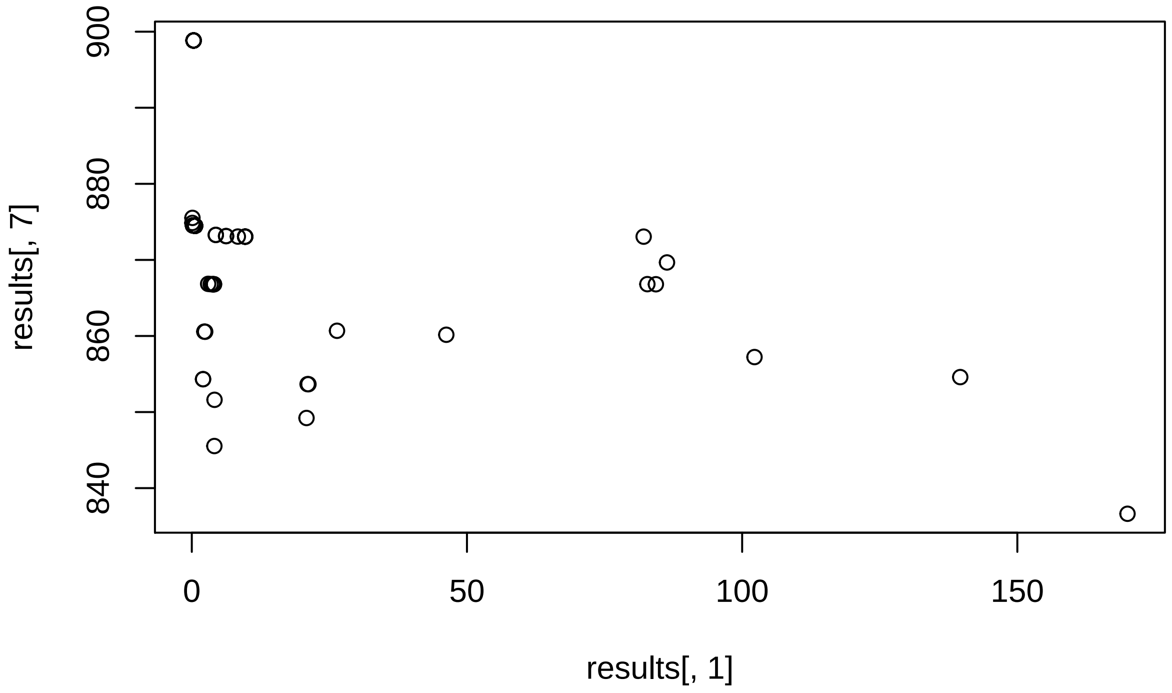
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

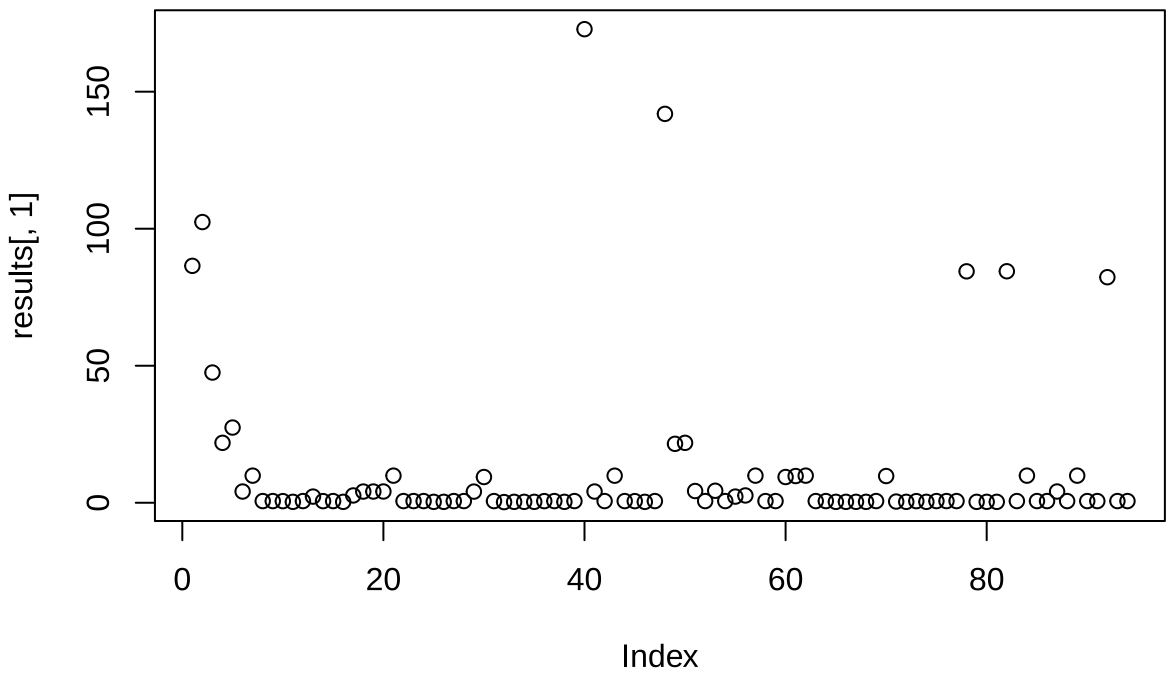




```
##  1 factor completed in 0.00446 minutes.    Estimated time of completion: 2020-04-28 10:38:40    [1]
##  [7]  10.0000010  30.0000010  78.6860377 874.8598149 170.0193040 836.6311219
## [13]  71.0000000  40.0000000
## [1] "j = 20.000001"
```

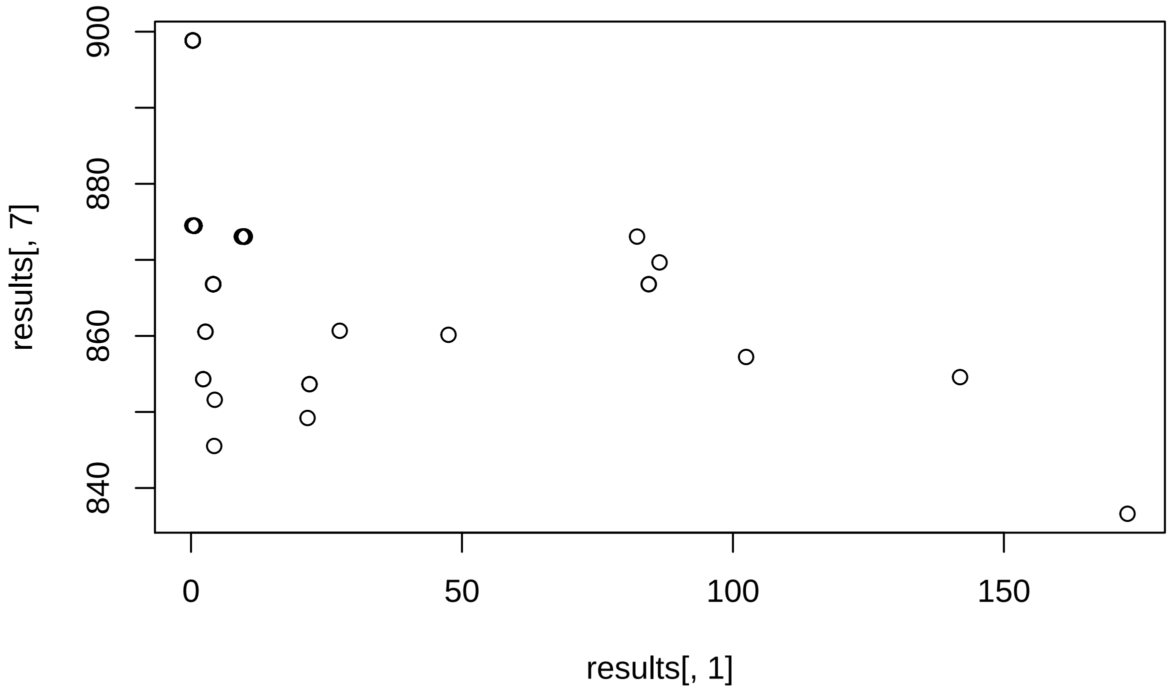
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

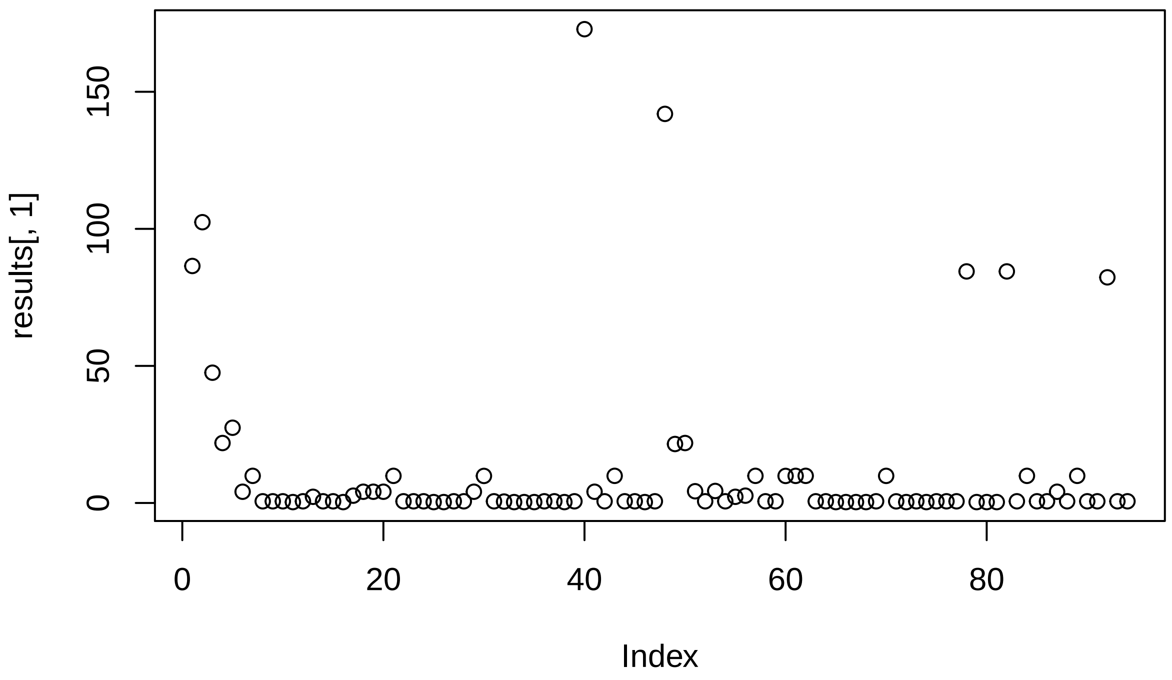




```
##  1 factor completed in 0.00448 minutes.    Estimated time of completion: 2020-04-28 10:38:42    [1]
##  [7]  20.0000010  30.0000010  78.6860377 874.5192313 172.8100692 836.6163359
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

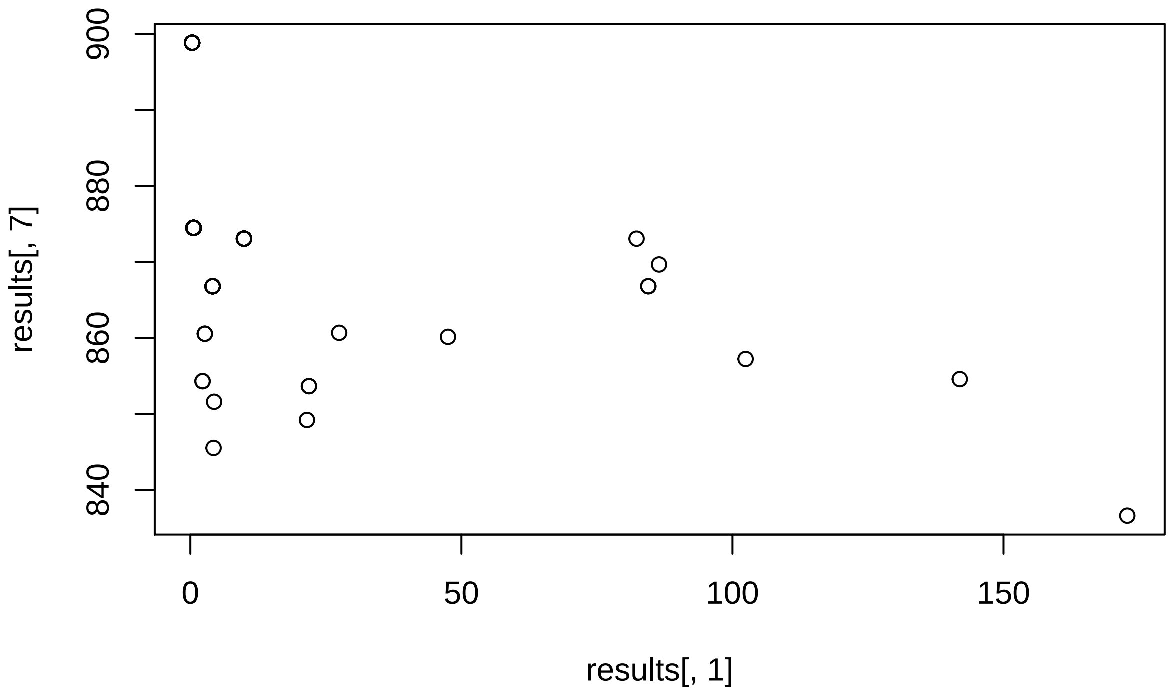
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

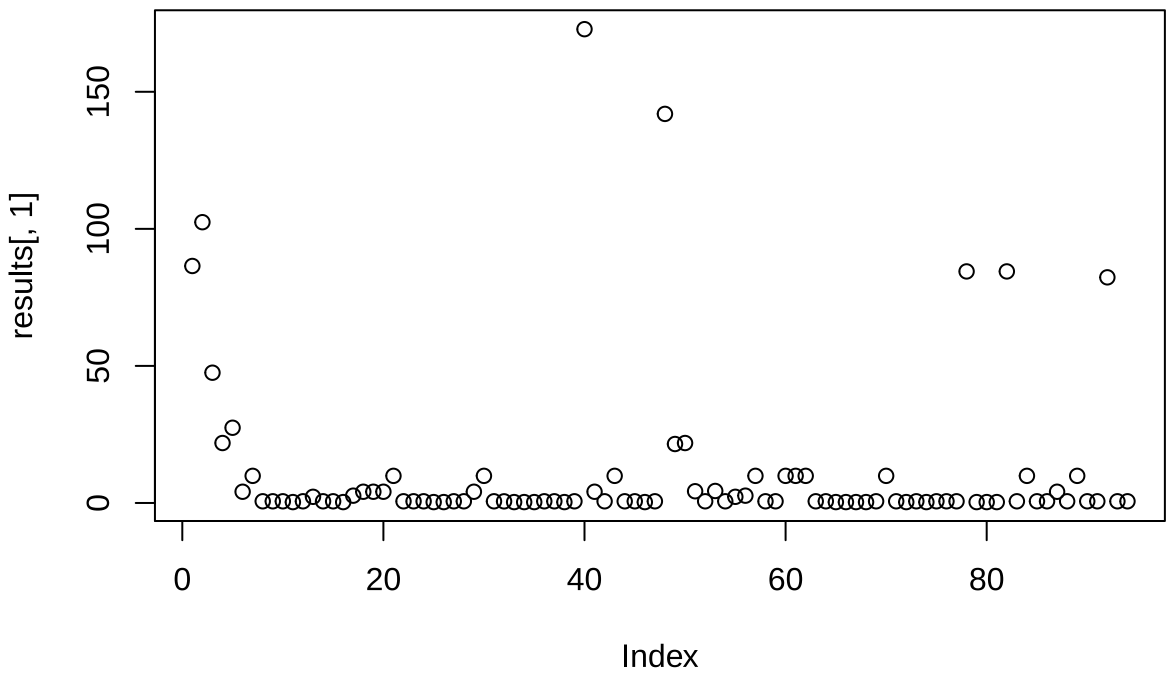




```
##  1 factor completed in 0.00599 minutes.    Estimated time of completion: 2020-04-28 10:38:45    [1]
##  [7]  30.0000010  30.0000010  78.6860377 898.8404771 172.8351932 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

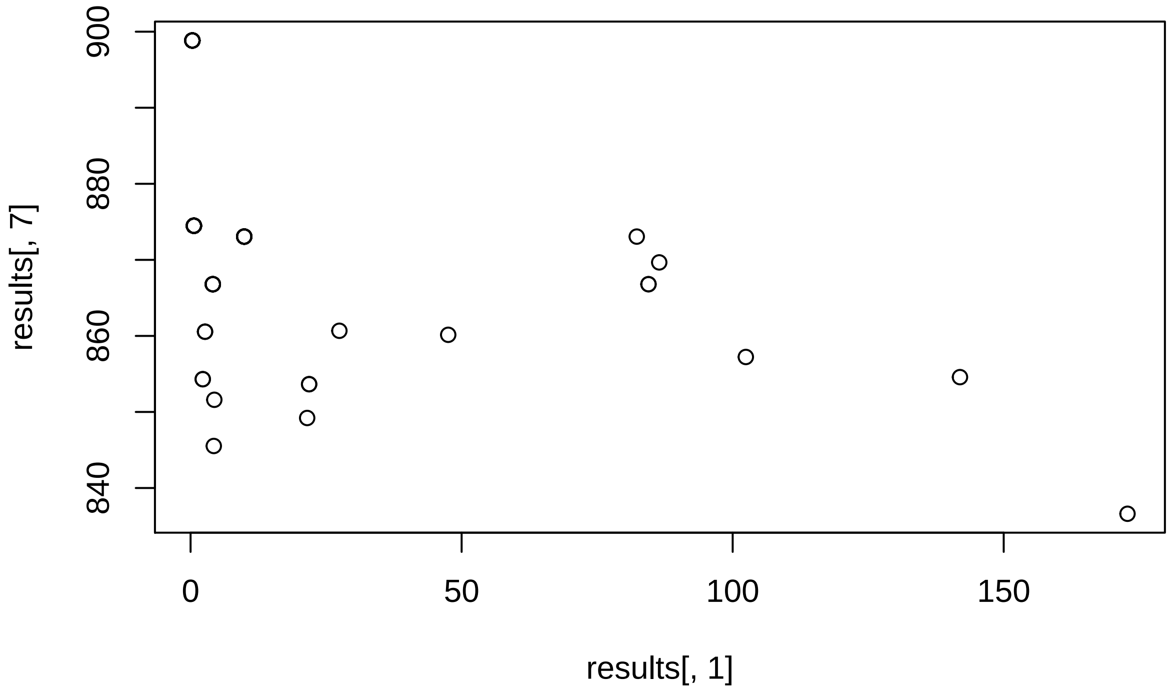
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

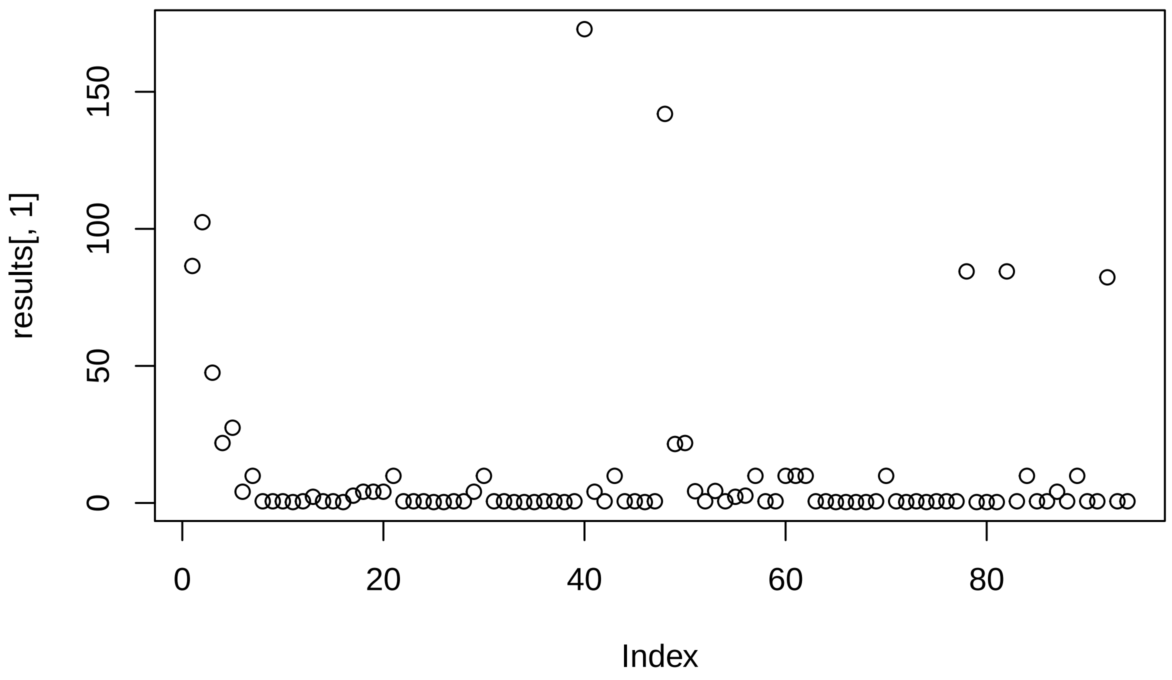




```
##  1 factor completed in 0.00451 minutes.    Estimated time of completion: 2020-04-28 10:38:47    [1]
##  [7]  40.0000010  30.0000010  78.6860377 898.8404771 172.8354168 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

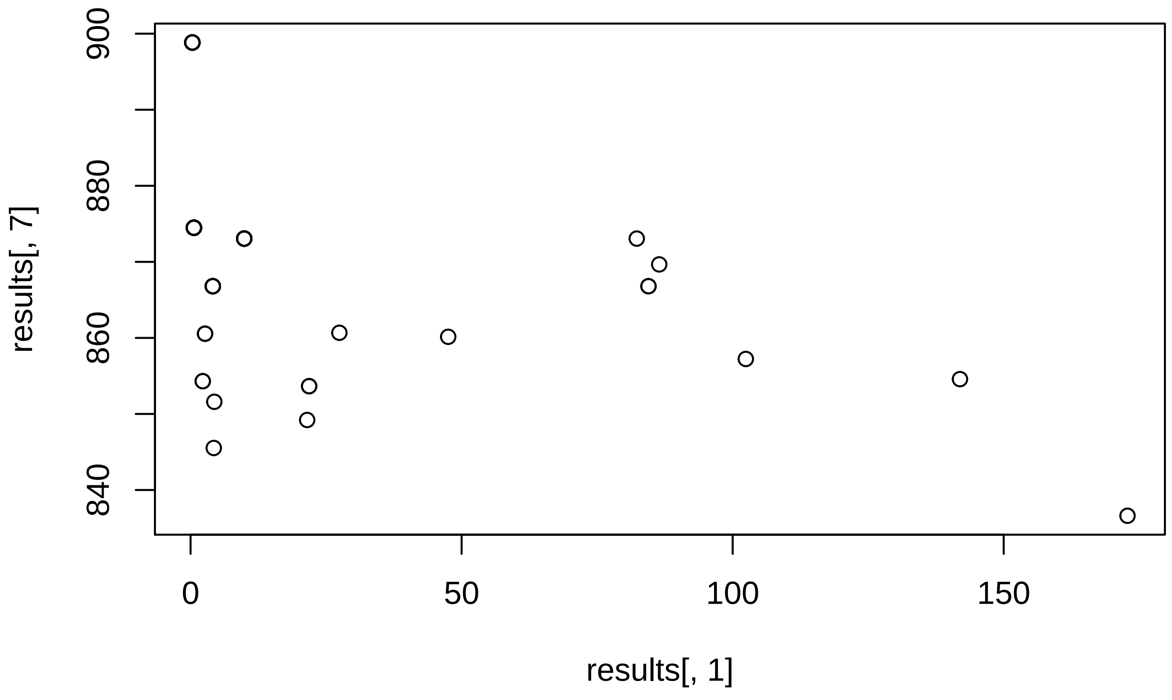
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

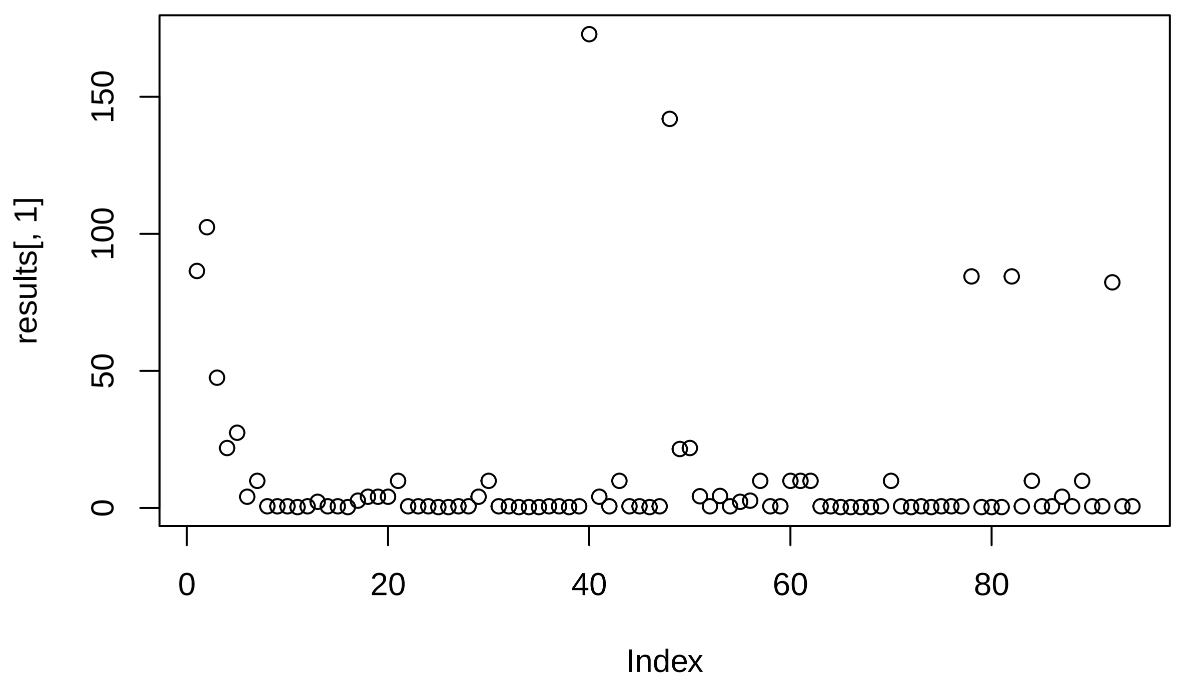




```
##  1 factor completed in 0.00512 minutes.    Estimated time of completion: 2020-04-28 10:38:49    [1]
##  [7]  50.0000010  30.0000010  78.6860377 898.8404771 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

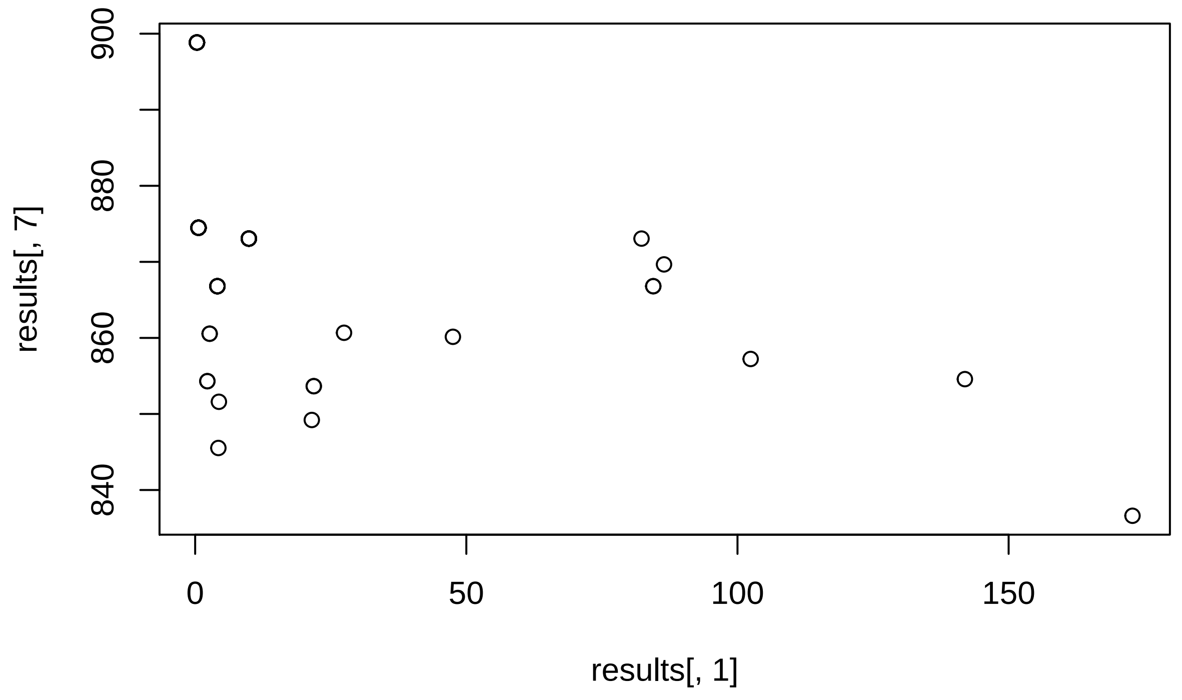
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

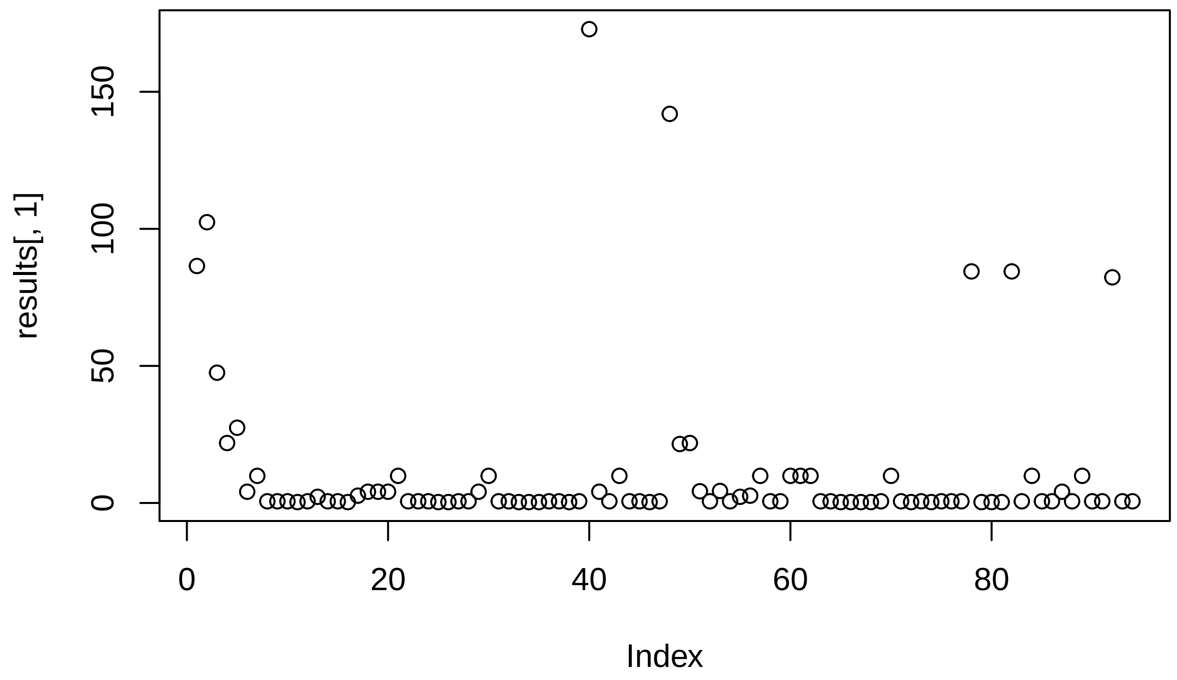




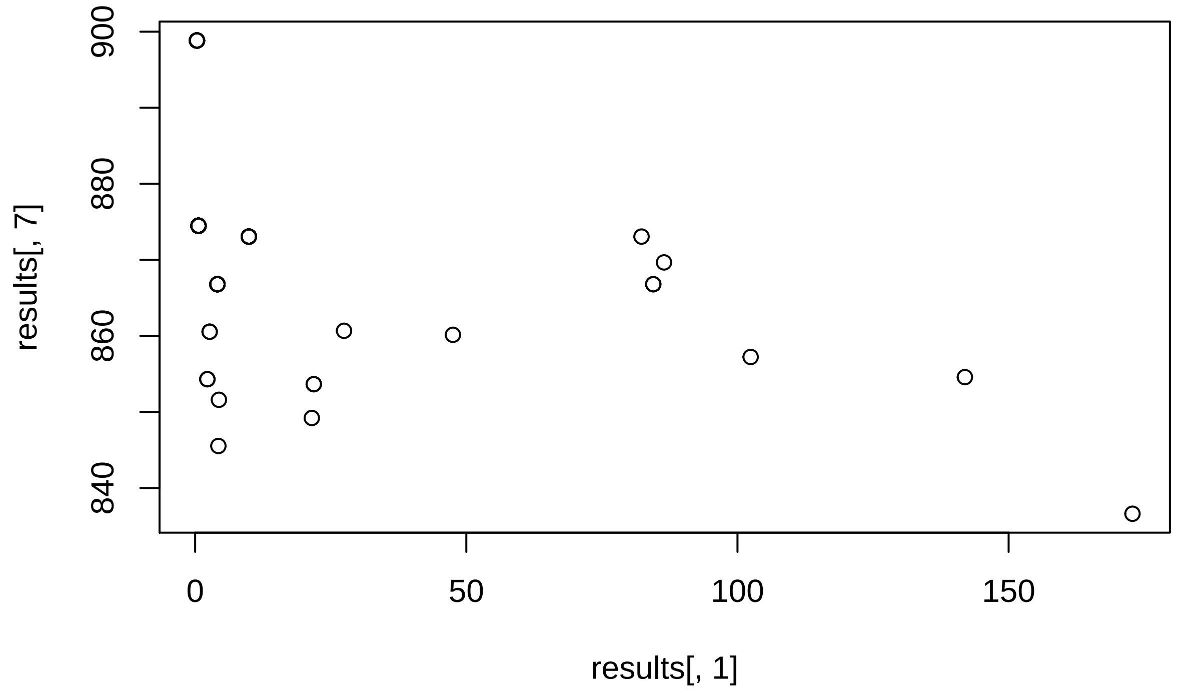
```
##  1 factor completed in 0.00611 minutes.    Estimated time of completion: 2020-04-28 10:38:53    [1]
##  [7]  60.0000010  30.0000010  78.6860377 898.8404771 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

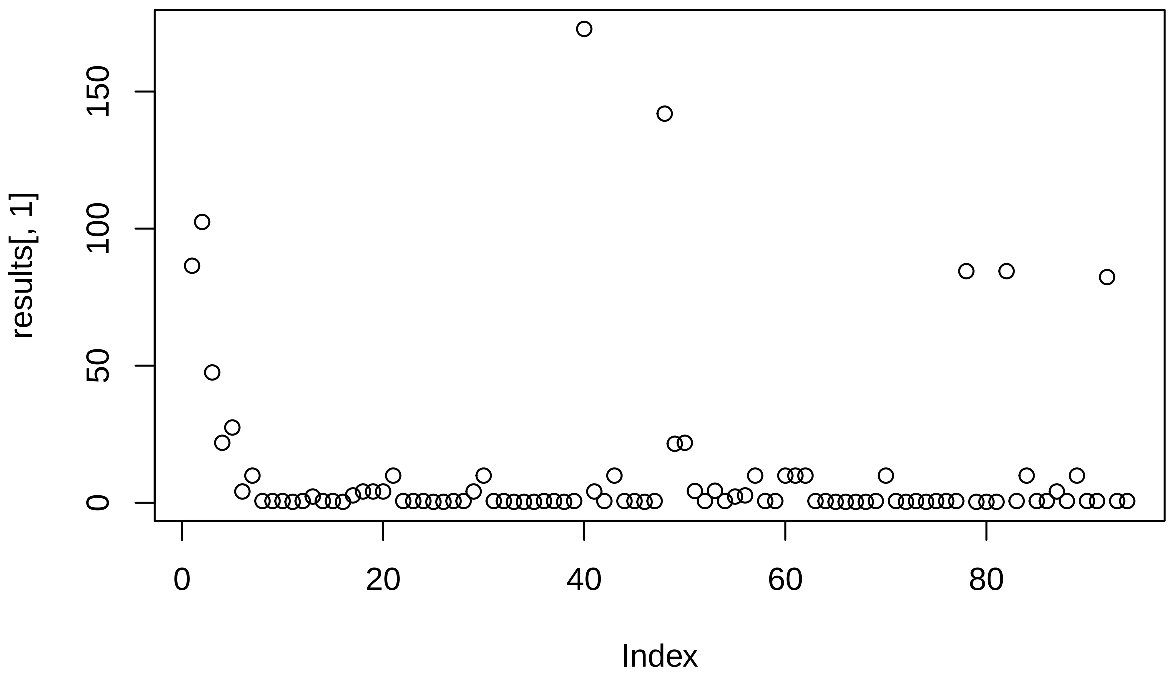
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





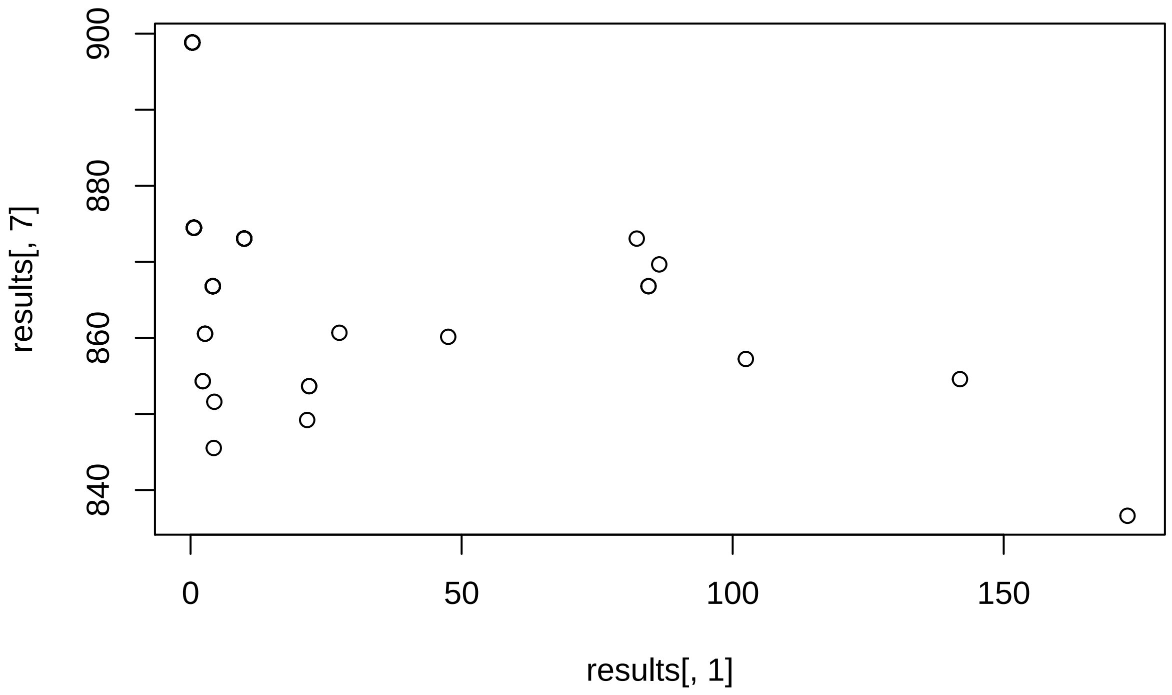
```
##  1 factor completed in 0.00779 minutes.    Estimated time of completion: 2020-04-28 10:38:57    [1]
##  [7]  70.0000010  30.0000010  78.6860377 898.8404771 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

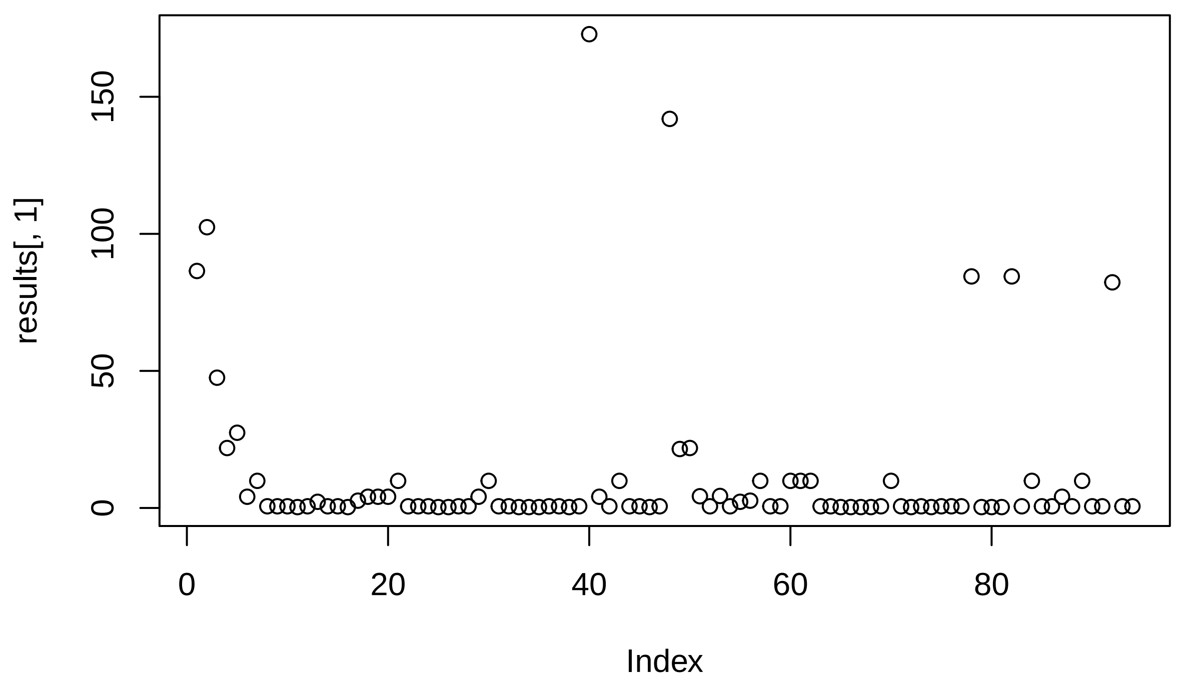




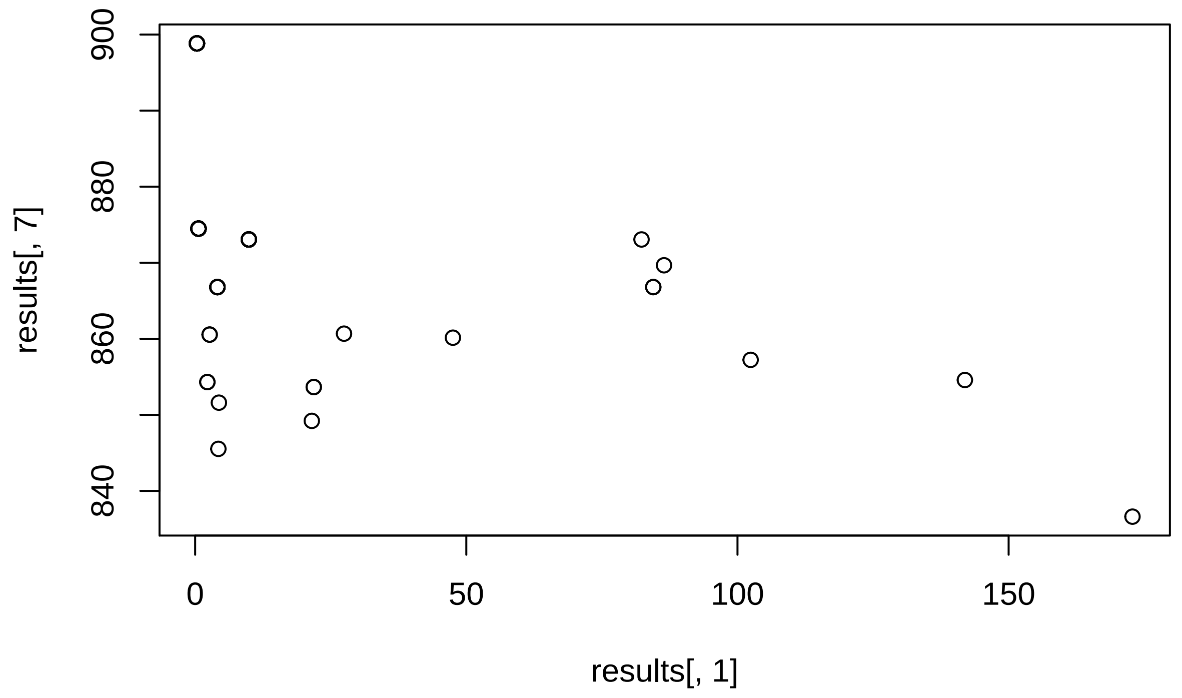
```
##  1 factor completed in 0.0065 minutes.    Estimated time of completion: 2020-04-28 10:39:00    [1]
##  [7]  80.0000010  30.0000010  78.6860377 898.8404771 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

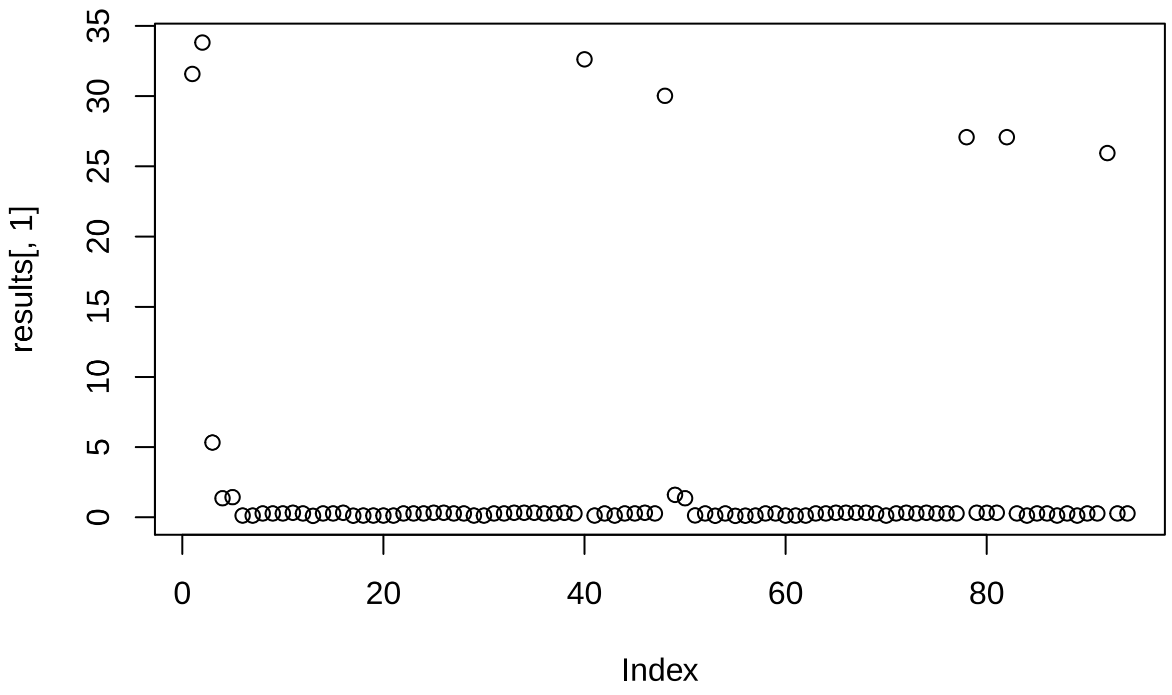
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





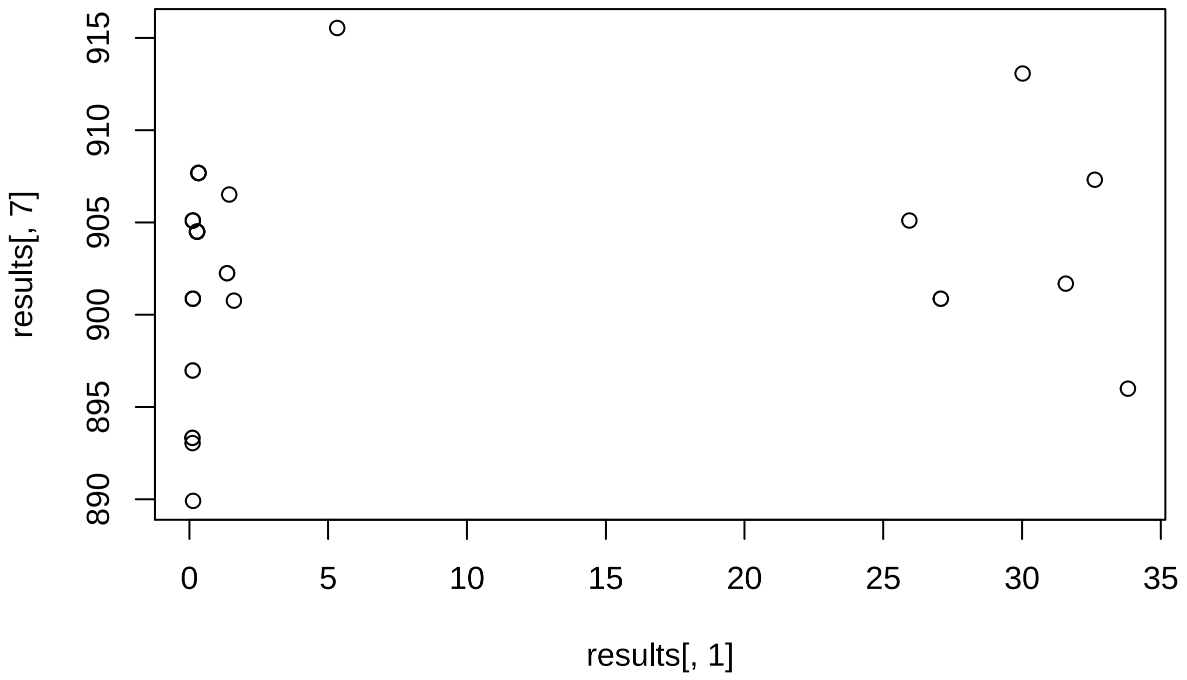
```
##  1 factor completed in 0.00853 minutes.    Estimated time of completion: 2020-04-28 10:39:03    [1]
##  [7]  90.0000010  30.0000010  78.6860377 898.8404771 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 40.000001"
## [1] "j = 1e-06"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

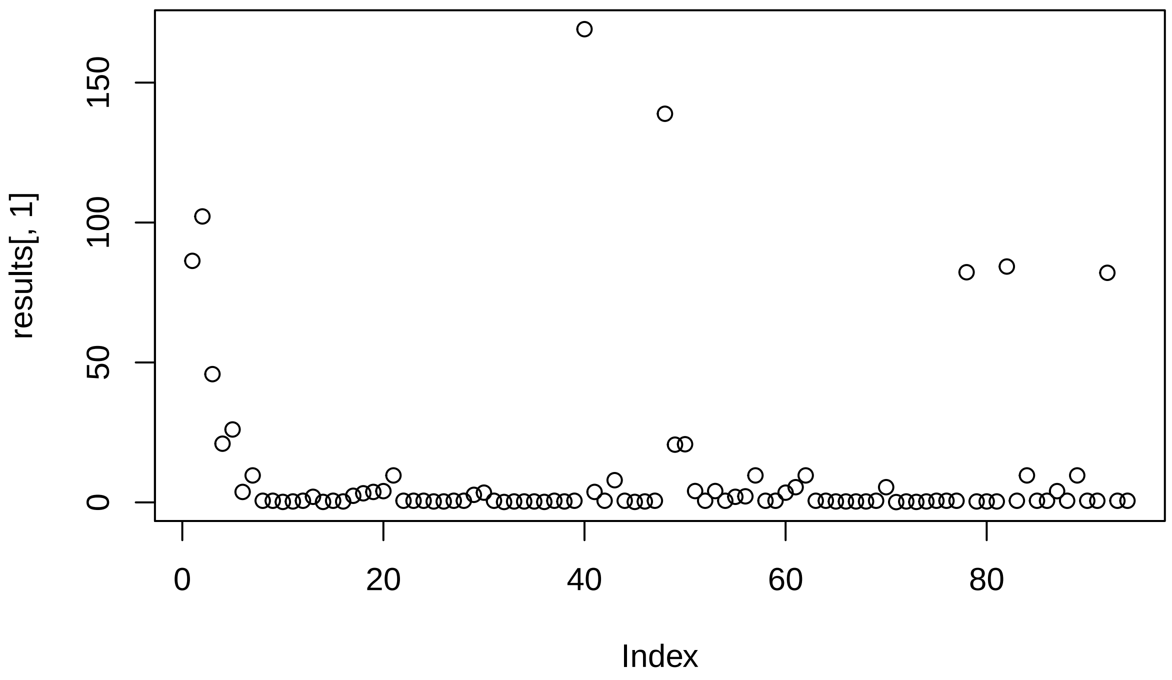




```
##  1 factor completed in 0.00661 minutes.    Estimated time of completion: 2020-04-28 10:39:05    [1]
##  [7]   0.0000010  40.0000010  78.6860377 893.3273907   0.1330806 889.9126791
## [13]  13.0000000  51.0000000
## [1] "j = 10.000001"
```

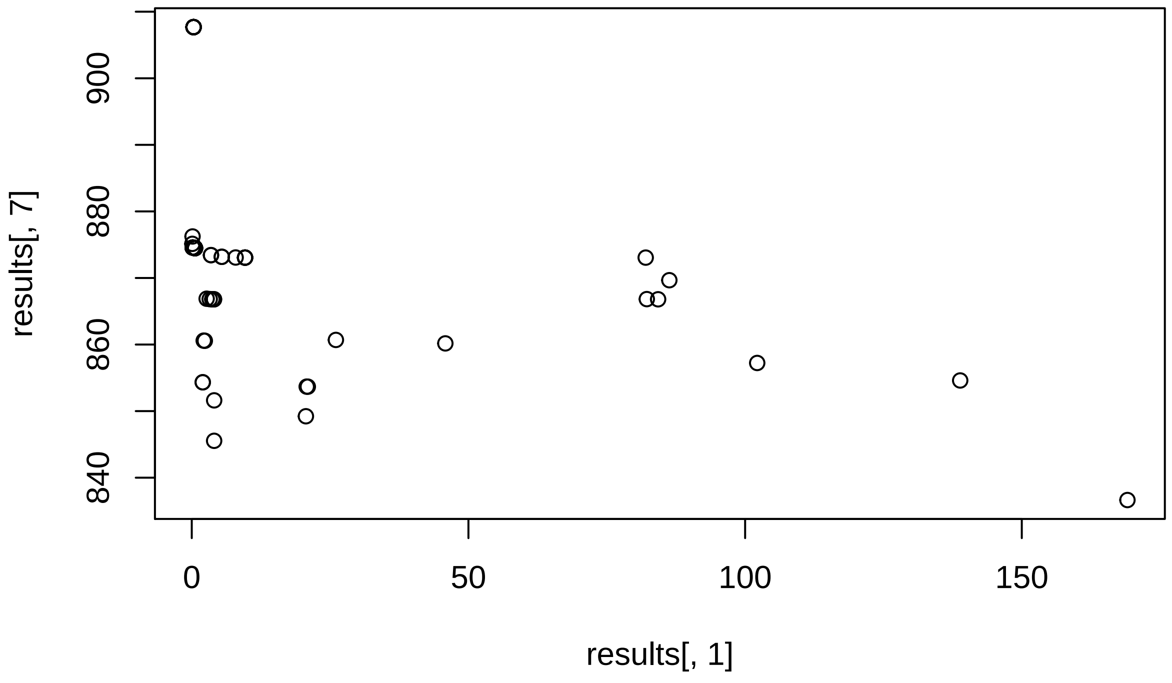
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

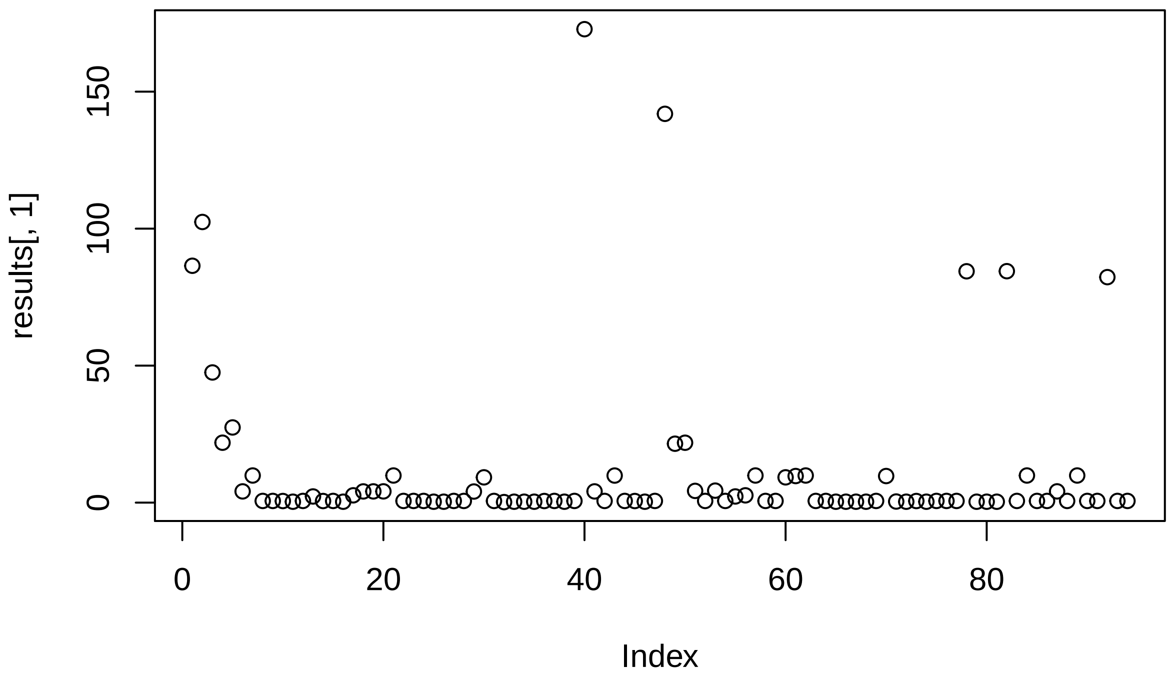




```
##  1 factor completed in 0.00593 minutes.    Estimated time of completion: 2020-04-28 10:39:09    [1]
##  [7]  10.0000010  40.0000010  78.6860377 875.1244138 169.0954685 836.6424860
## [13]  71.0000000  40.0000000
## [1] "j = 20.000001"
```

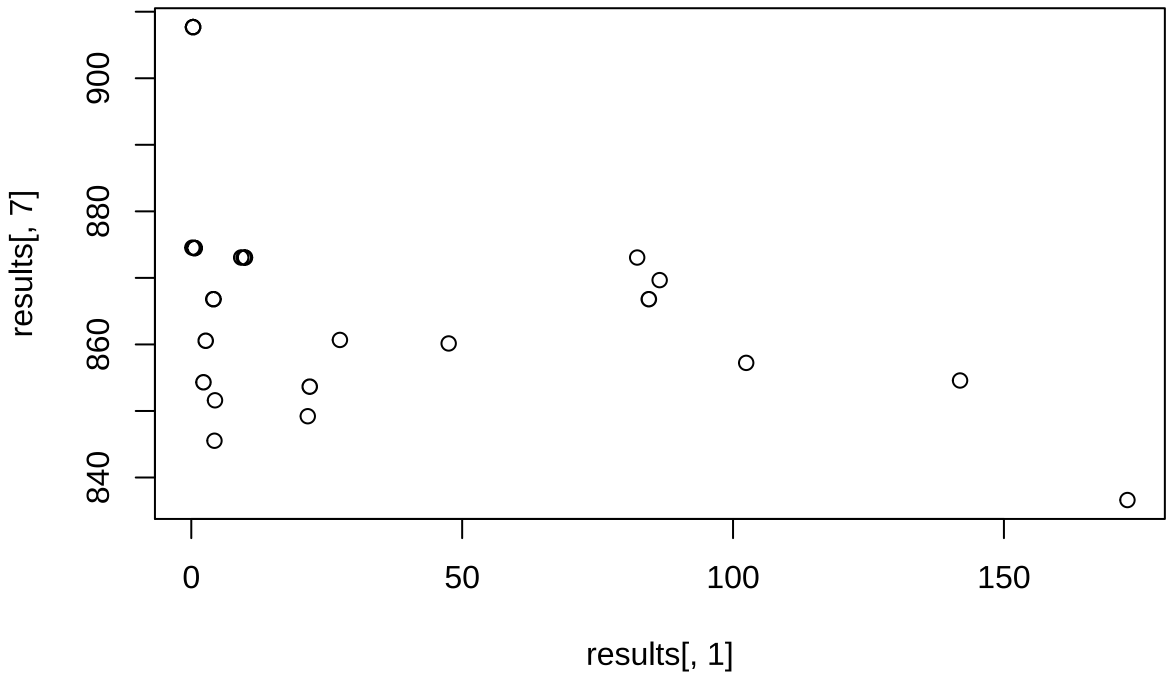
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

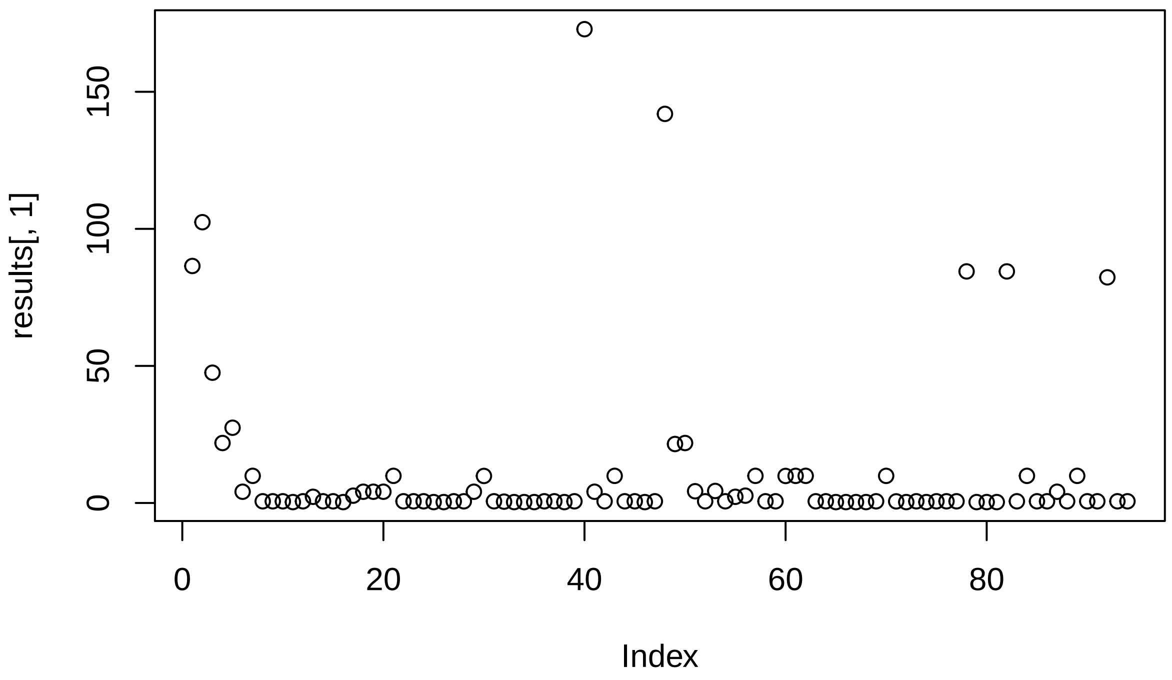




```
##  1 factor completed in 0.00468 minutes.    Estimated time of completion: 2020-04-28 10:39:12    [1]
##  [7]  20.0000010  40.0000010  78.6860377 874.5443353 172.8016206 836.6163369
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

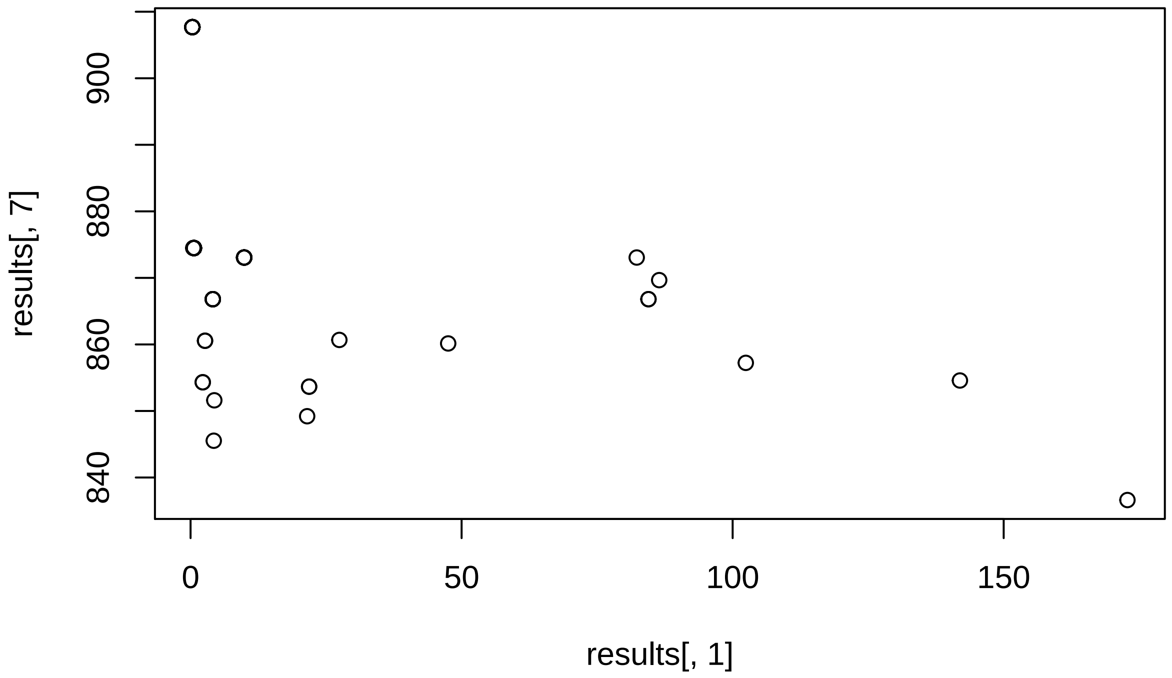
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

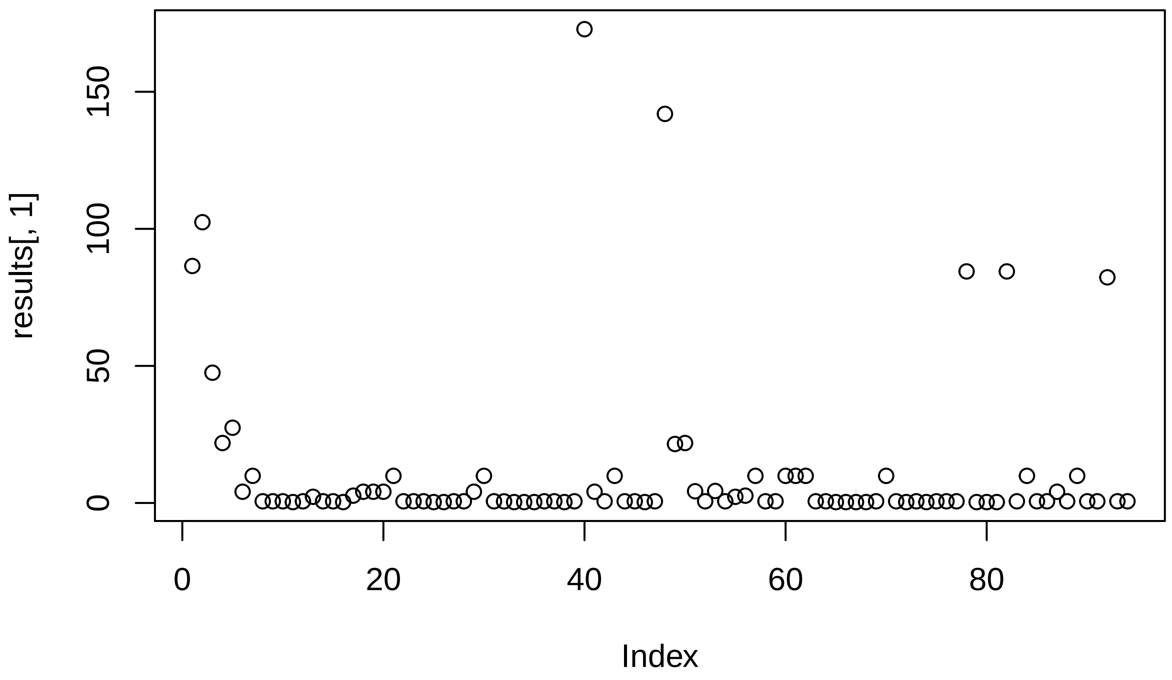




```
##  1 factor completed in 0.00415 minutes.    Estimated time of completion: 2020-04-28 10:39:14    [1]
##  [7]  30.0000010  40.0000010  78.6860377 907.6808775 172.8351181 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

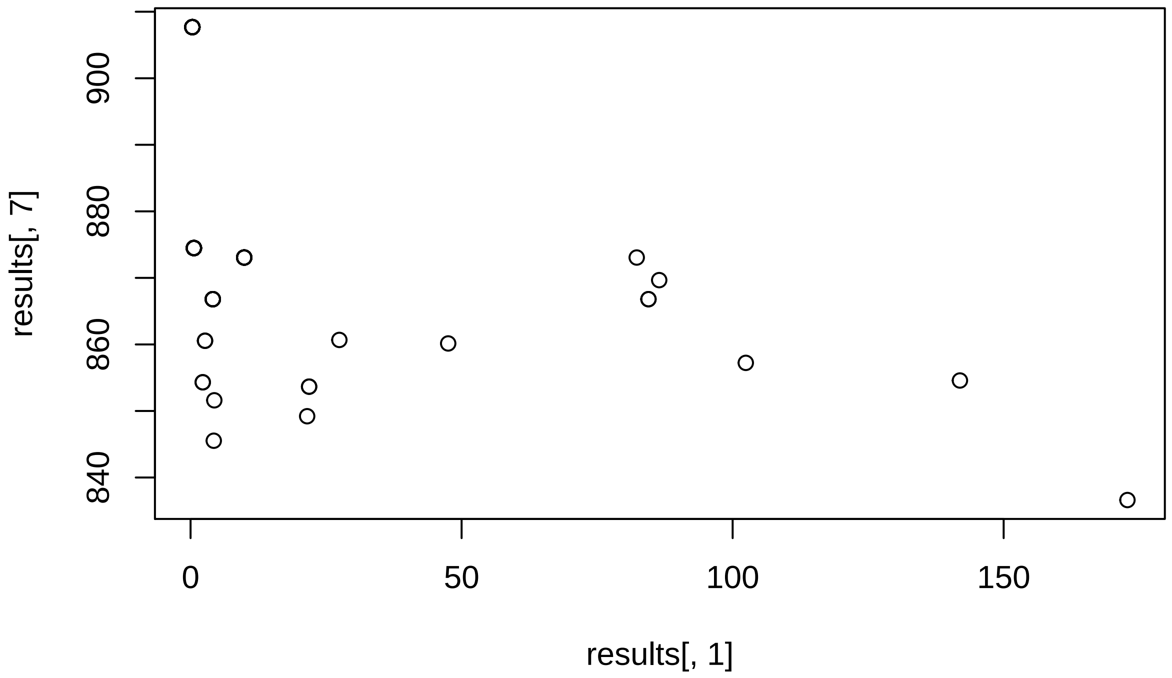
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

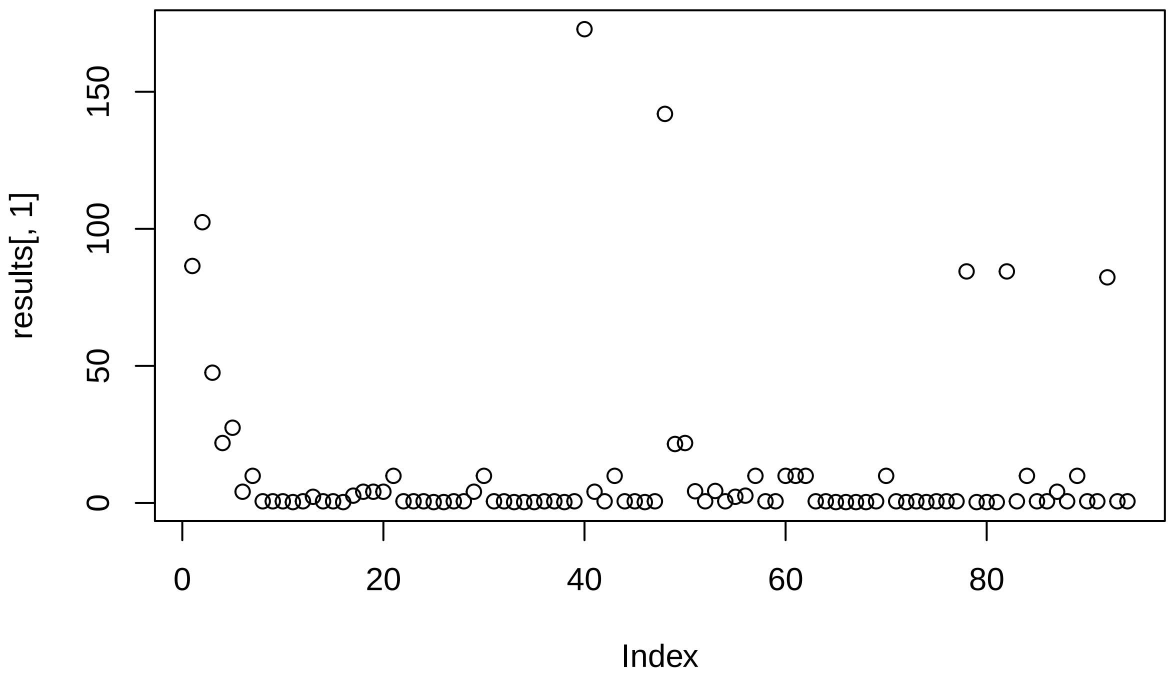




```
##  1 factor completed in 0.00451 minutes.    Estimated time of completion: 2020-04-28 10:39:17    [1]
##  [7]  40.0000010  40.0000010  78.6860377 907.6808775 172.8354161 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

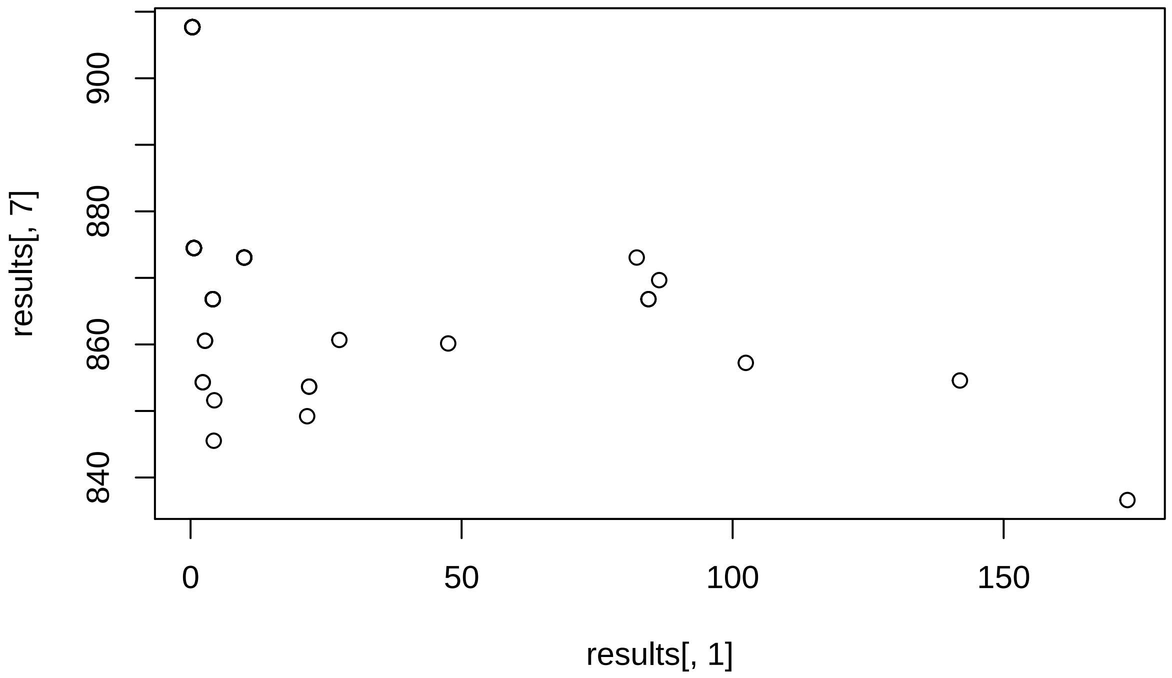
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

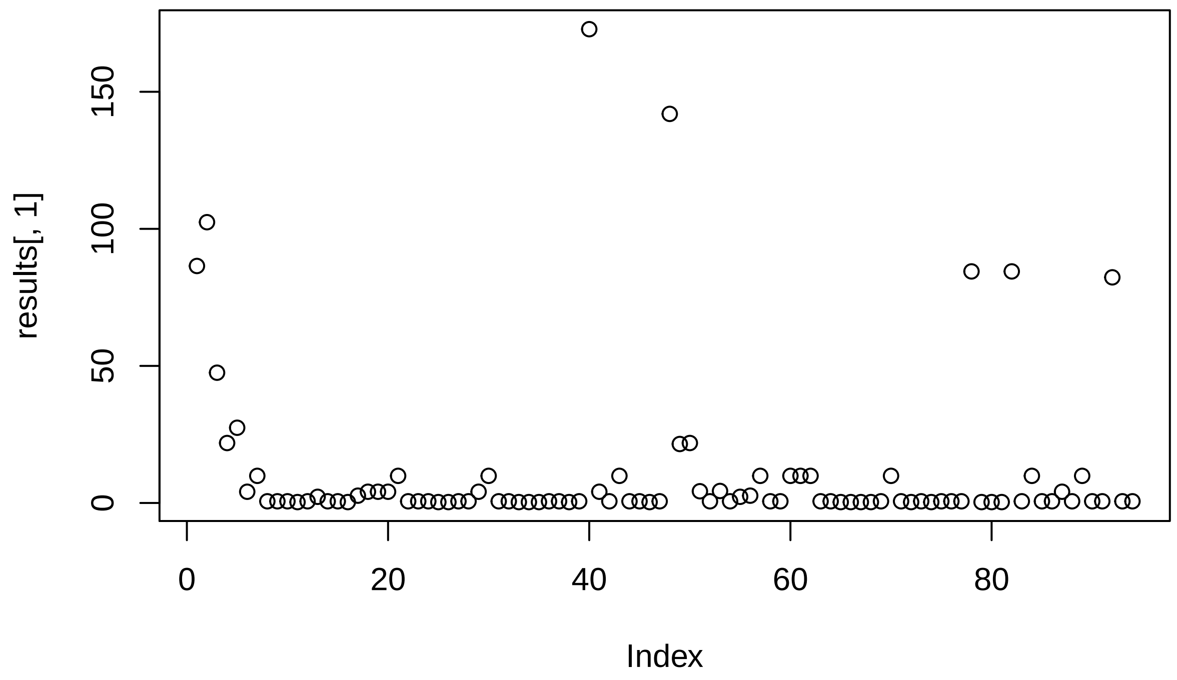




```
##  1 factor completed in 0.00414 minutes.    Estimated time of completion: 2020-04-28 10:39:19    [1]
##  [7]  50.0000010  40.0000010  78.6860377 907.6808775 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

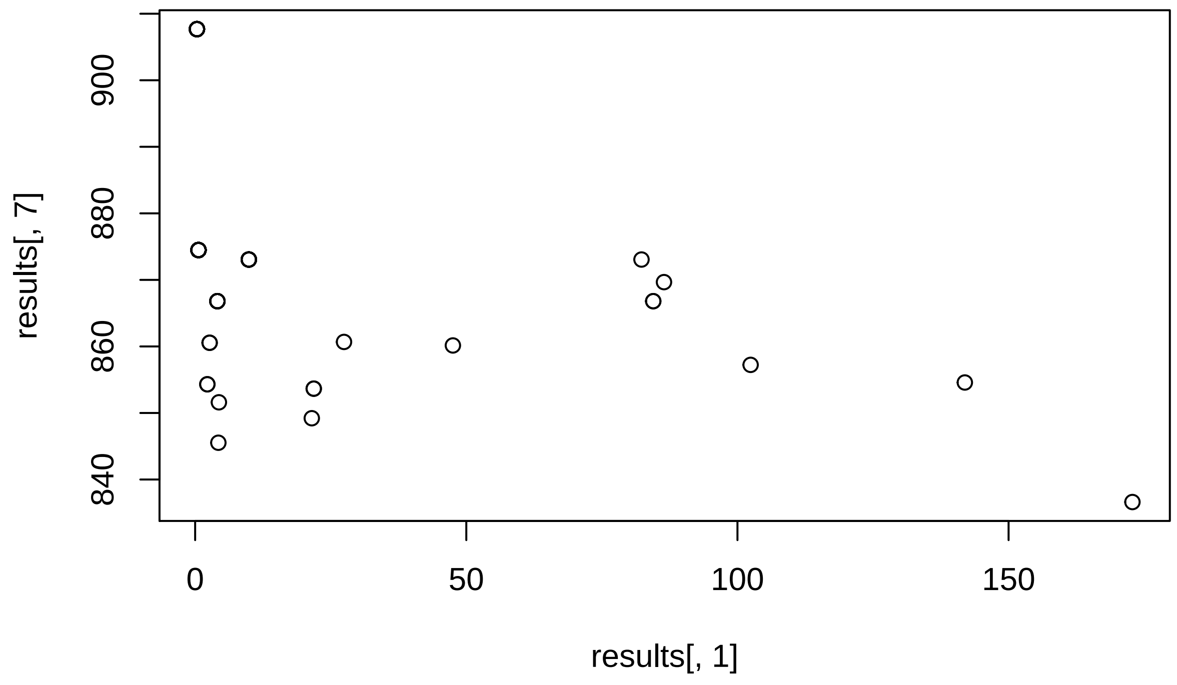
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

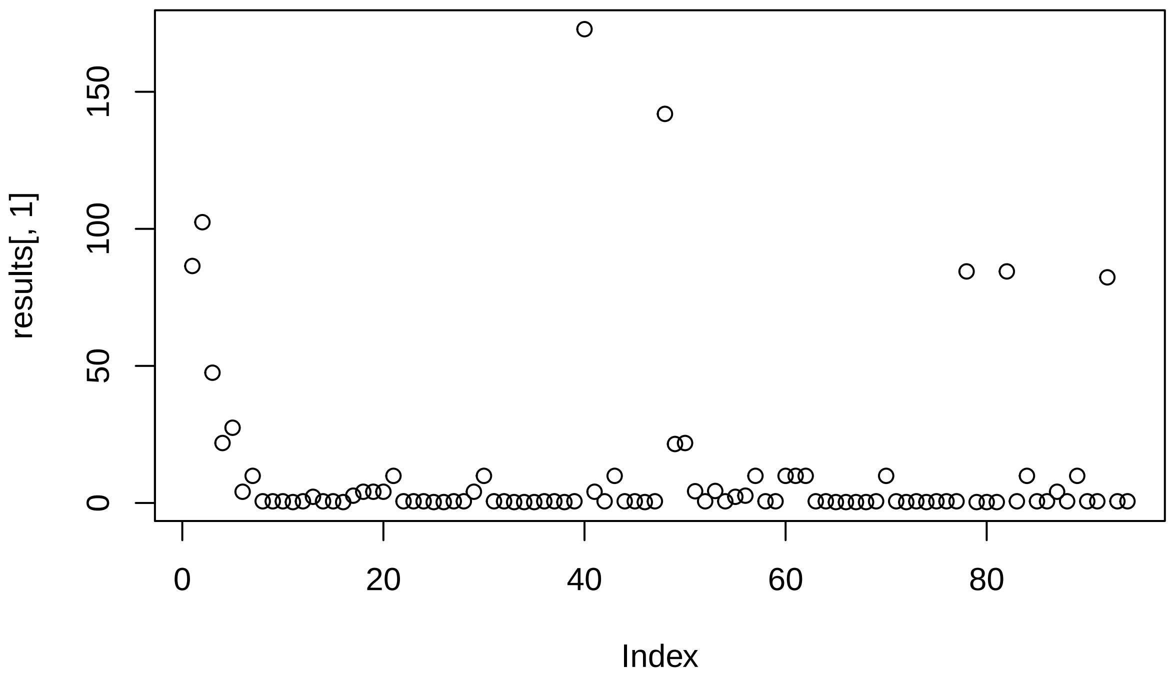




```
##  1 factor completed in 0.00422 minutes.    Estimated time of completion: 2020-04-28 10:39:21    [1]
##  [7]  60.0000010  40.0000010  78.6860377 907.6808775 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

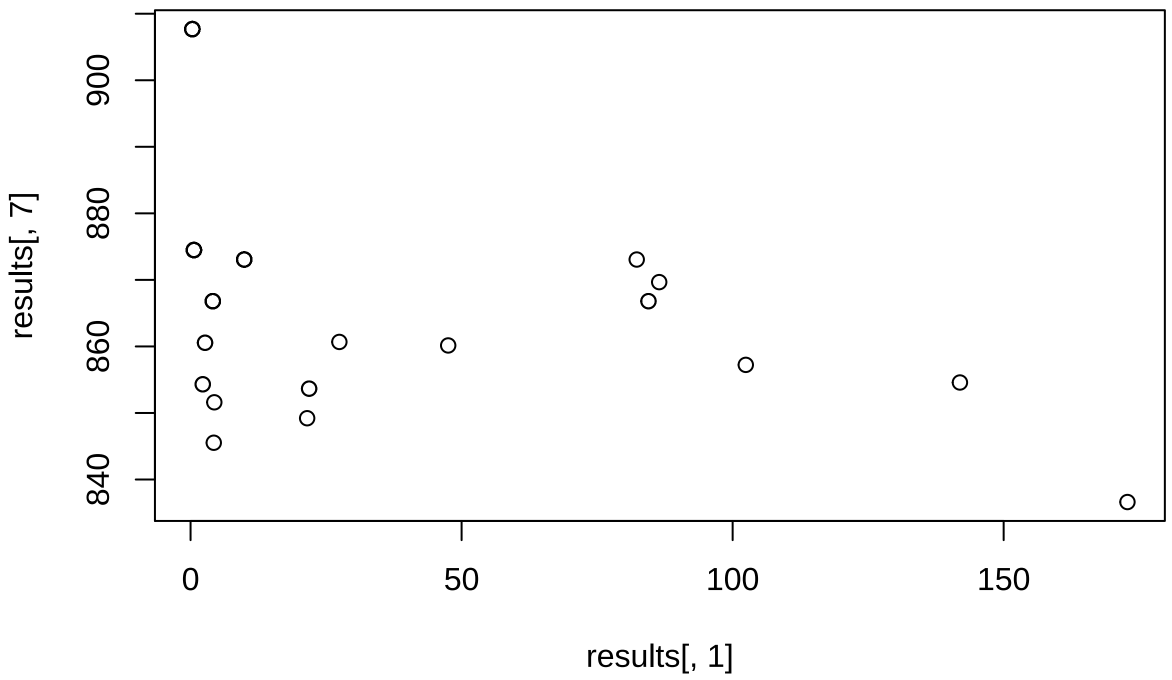
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

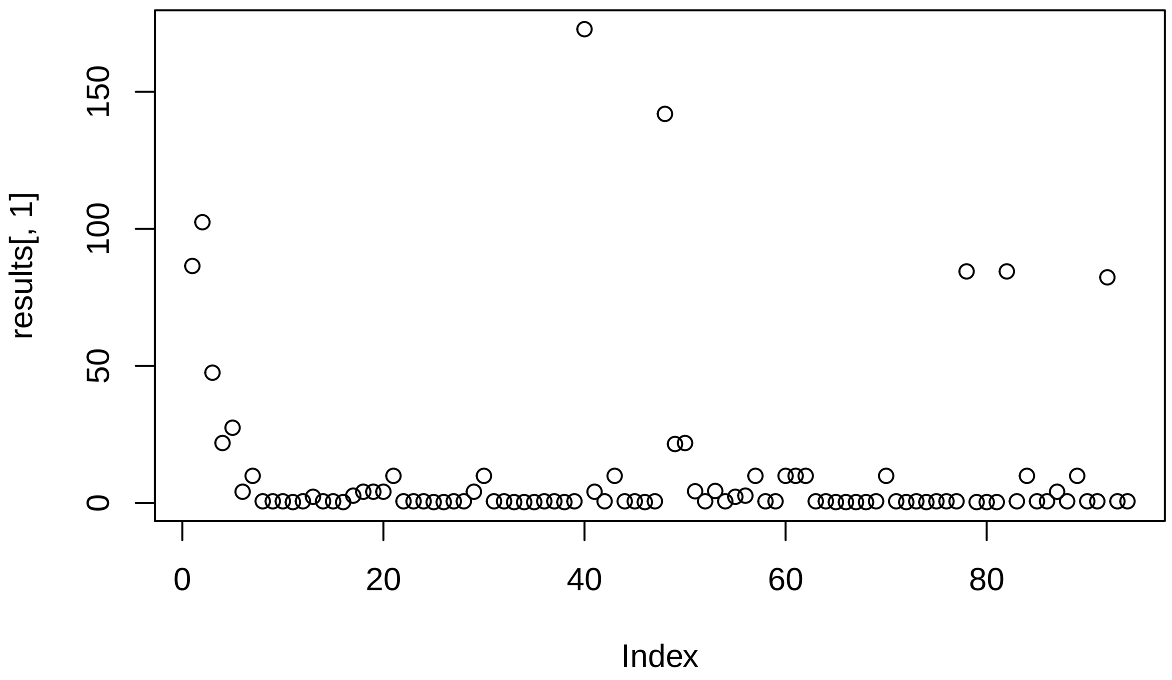




```
##  1 factor completed in 0.00414 minutes.    Estimated time of completion: 2020-04-28 10:39:23    [1]
##  [7]  70.0000010  40.0000010  78.6860377 907.6808775 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

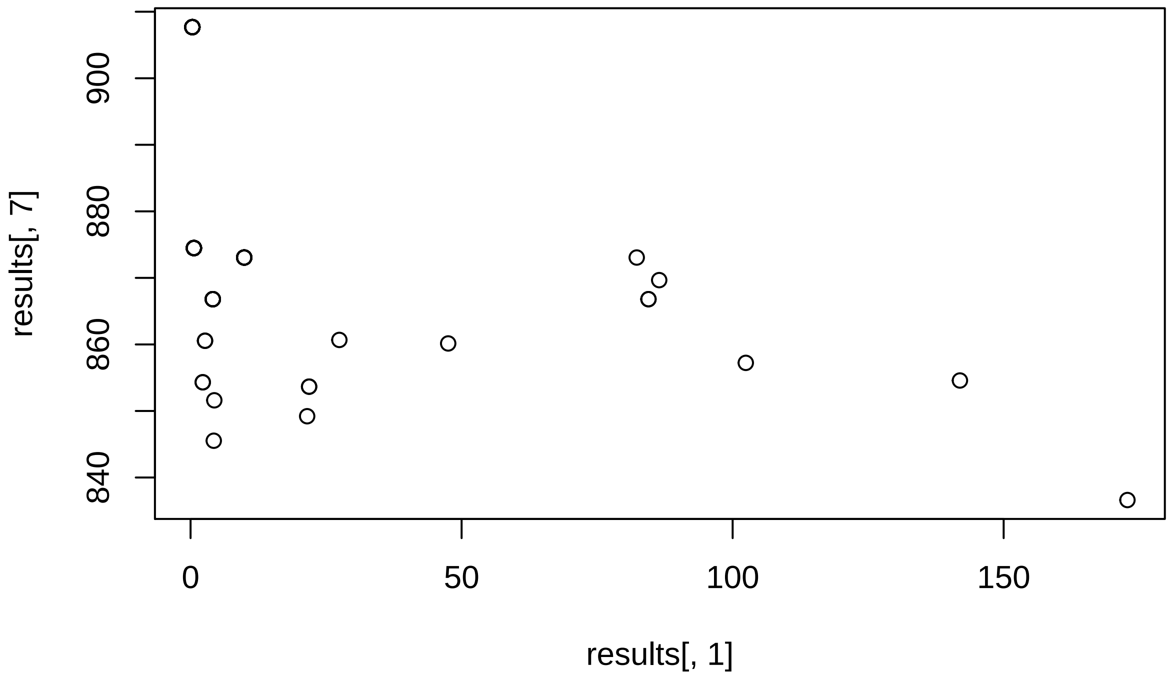
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

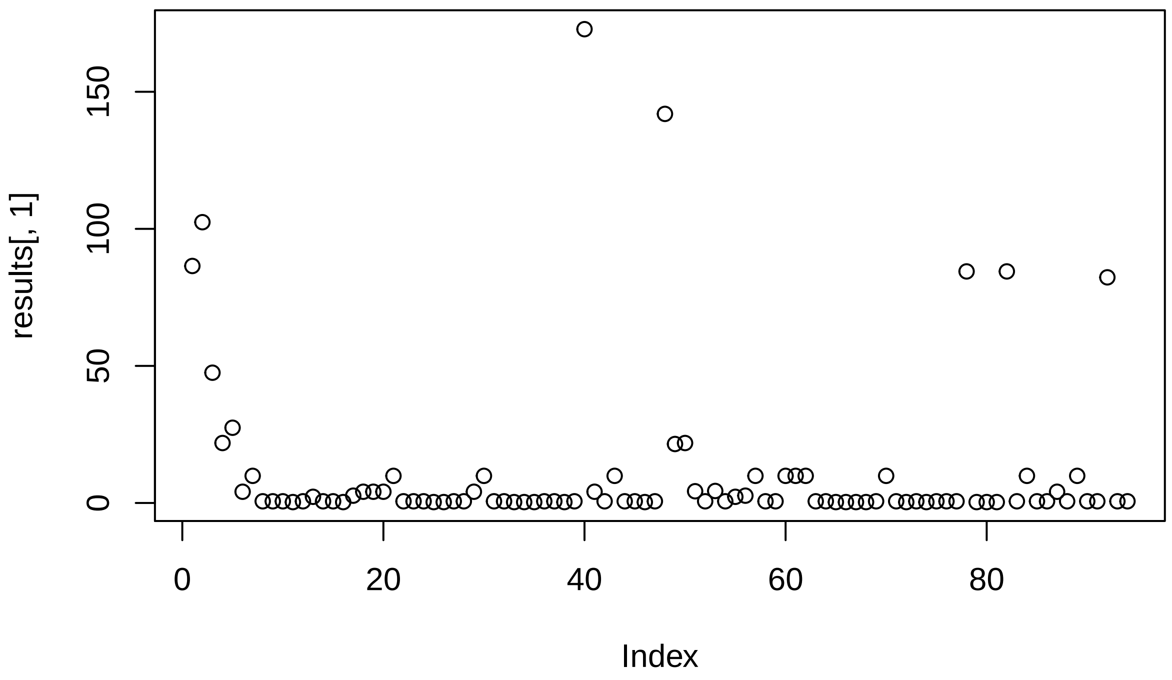




```
##  1 factor completed in 0.00411 minutes.    Estimated time of completion: 2020-04-28 10:39:25    [1]
##  [7]  80.0000010  40.0000010  78.6860377 907.6808775 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

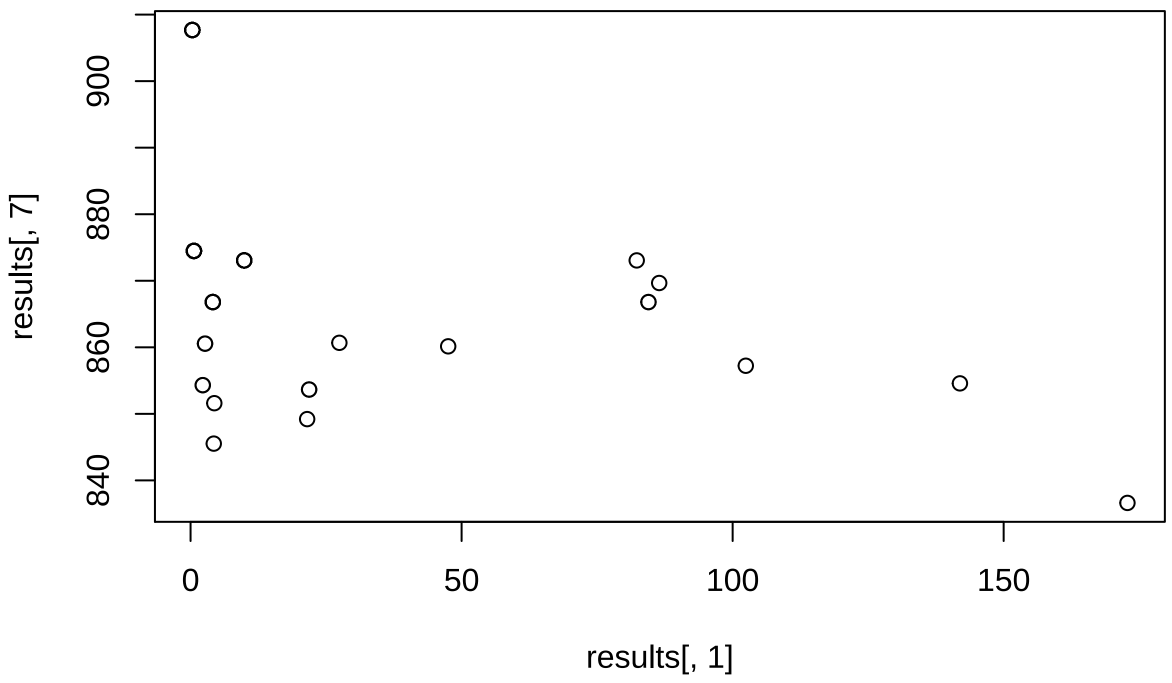
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

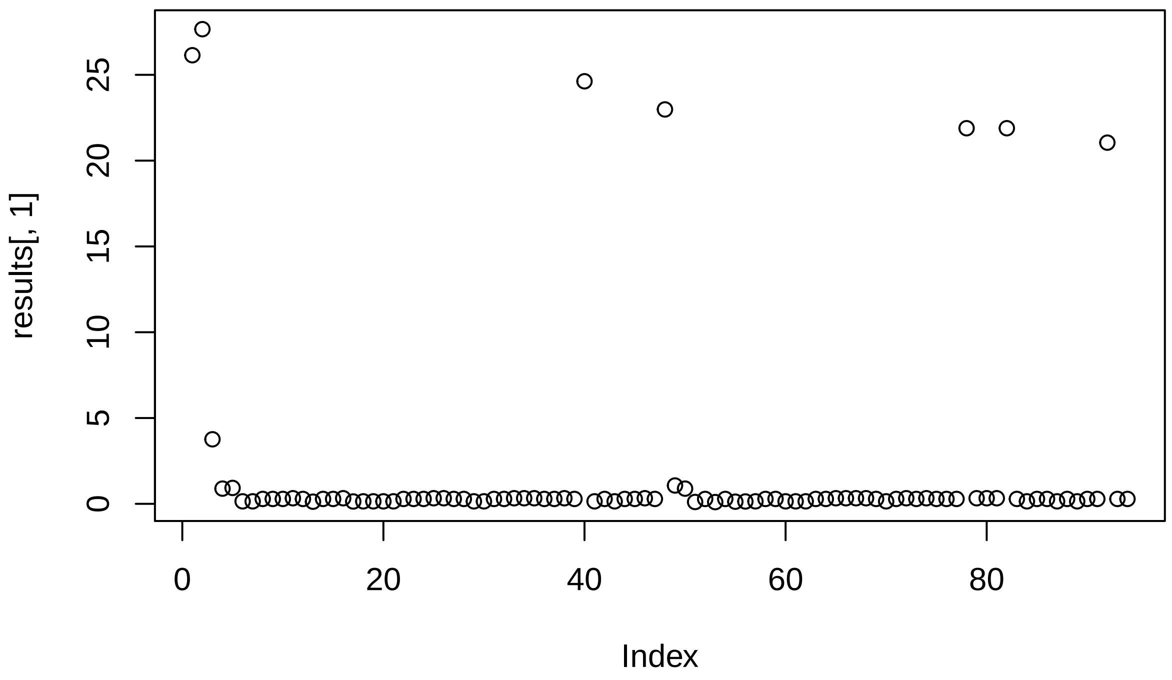




```
##  1 factor completed in 0.00417 minutes.    Estimated time of completion: 2020-04-28 10:39:27    [1]
##  [7]  90.0000010  40.0000010  78.6860377 907.6808775 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 50.000001"
## [1] "j = 1e-06"
```

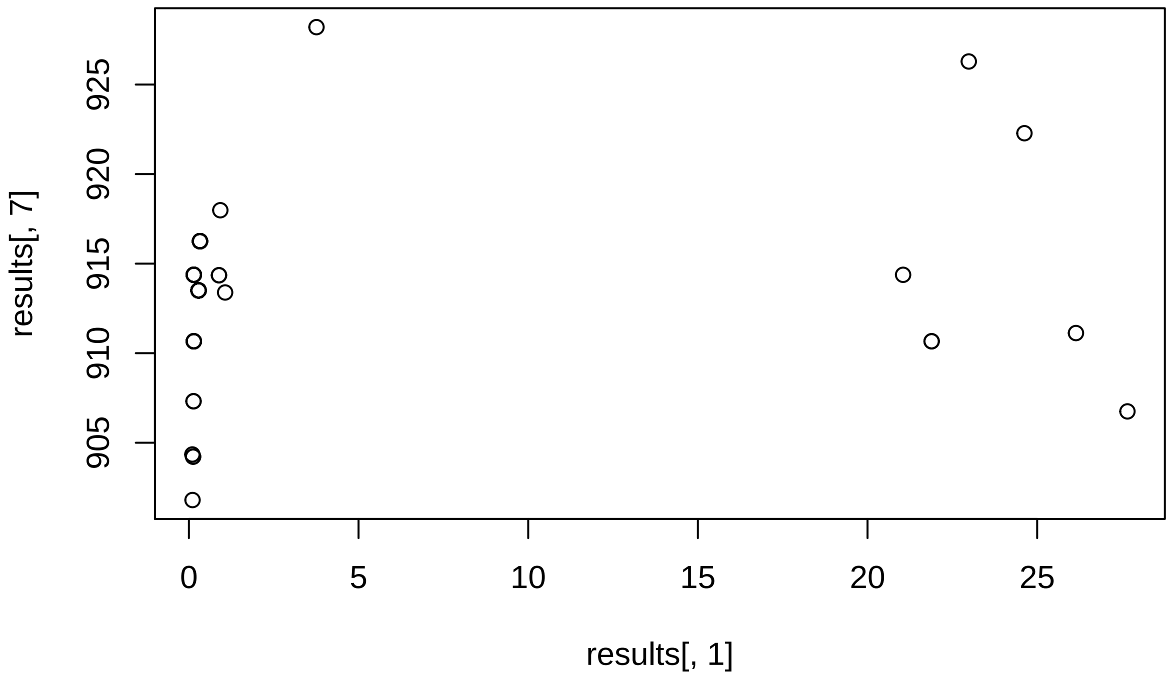
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

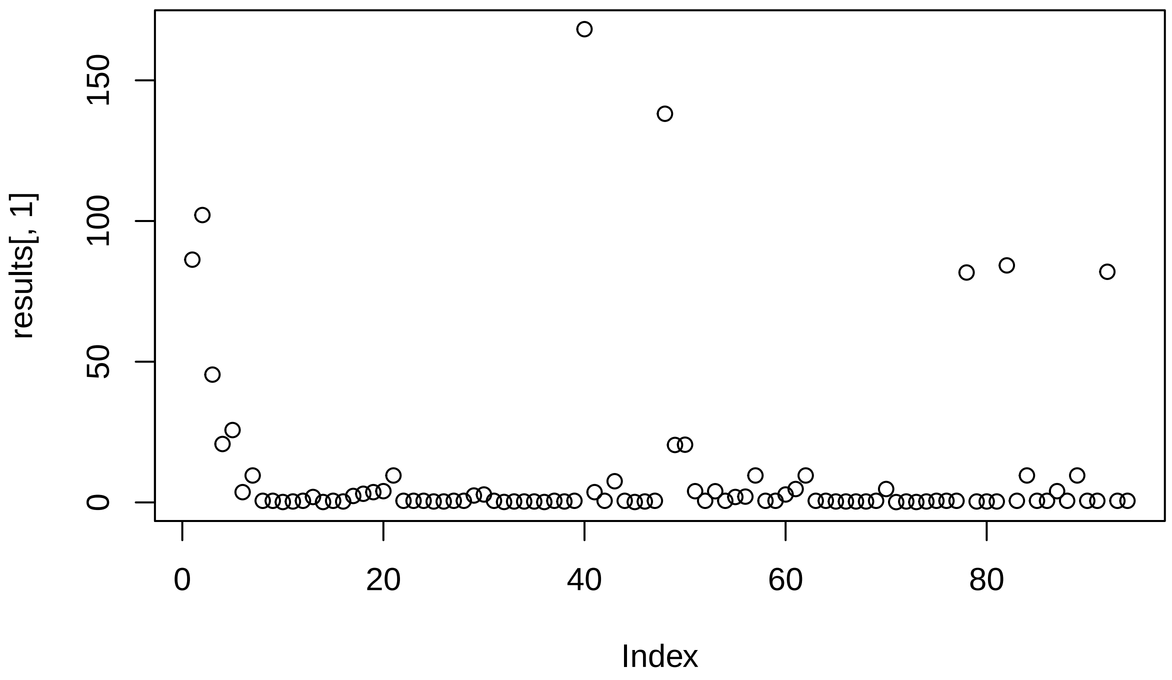




```
##  1 factor completed in 0.00399 minutes.    Estimated time of completion: 2020-04-28 10:39:29    [1]
##  [7]   0.0000010  50.0000010  78.6860377 904.3458835   0.1070913 901.8010704
## [13]  53.0000000  51.0000000
## [1] "j = 10.000001"
```

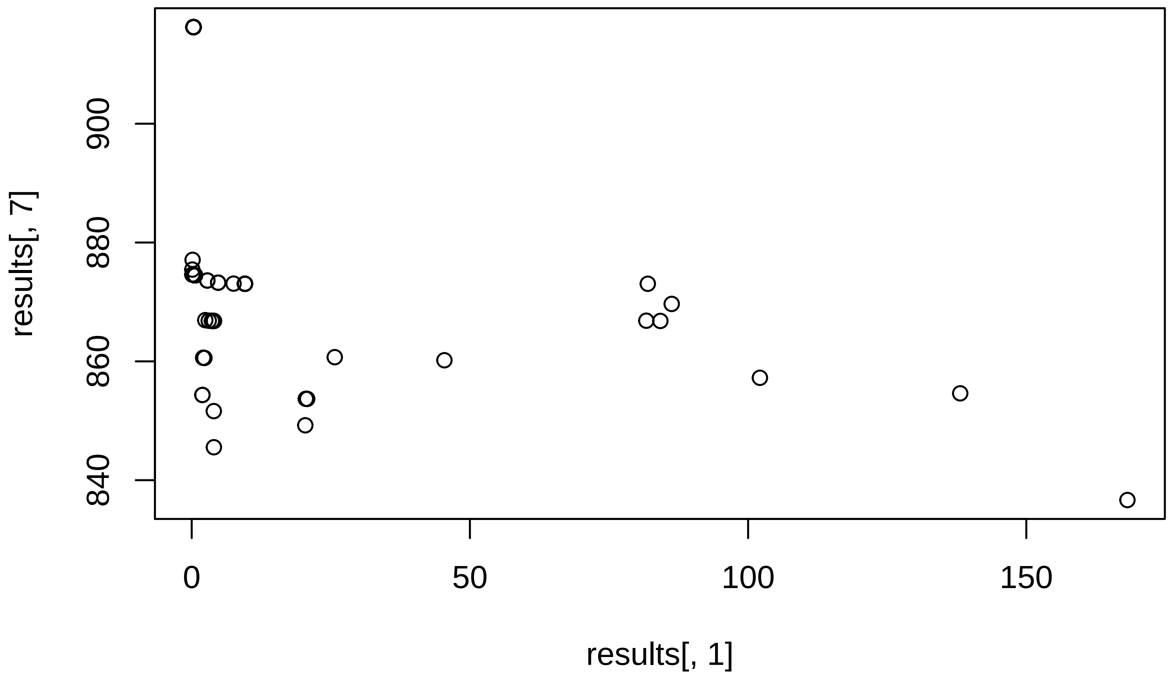
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

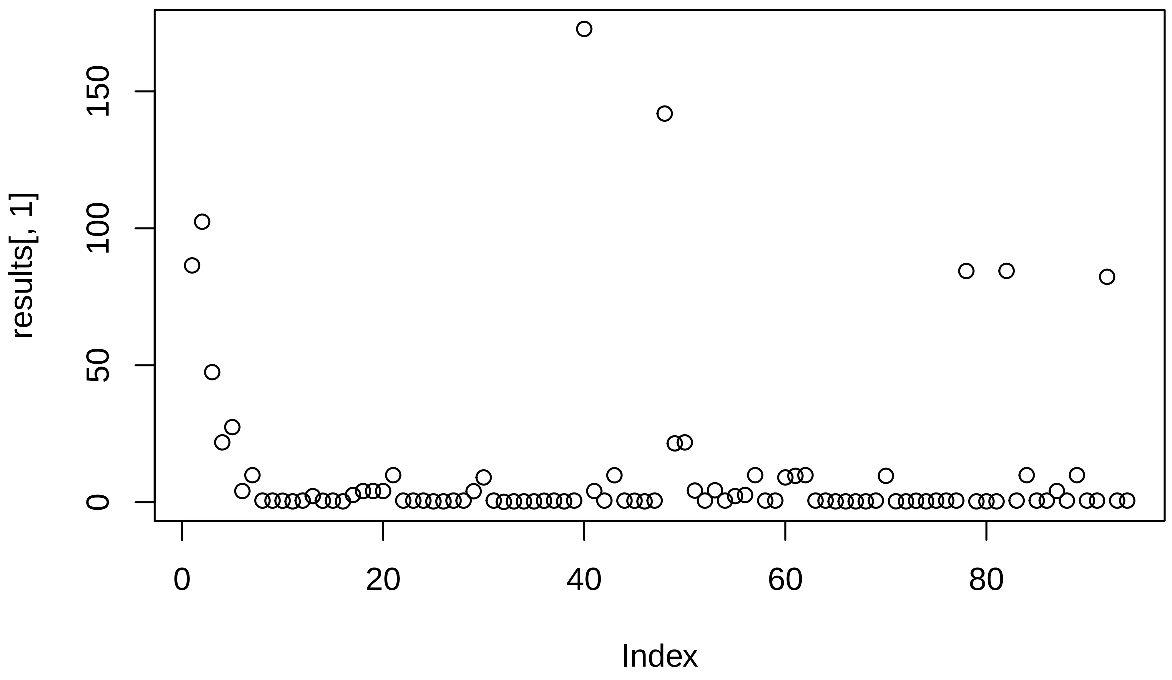




```
##  1 factor completed in 0.0042 minutes.    Estimated time of completion: 2020-04-28 10:39:31    [1]
##  [7]  10.0000010  50.0000010  78.6860377 875.4488905 168.1789379 836.6569837
## [13]  71.0000000  40.0000000
## [1] "j = 20.000001"
```

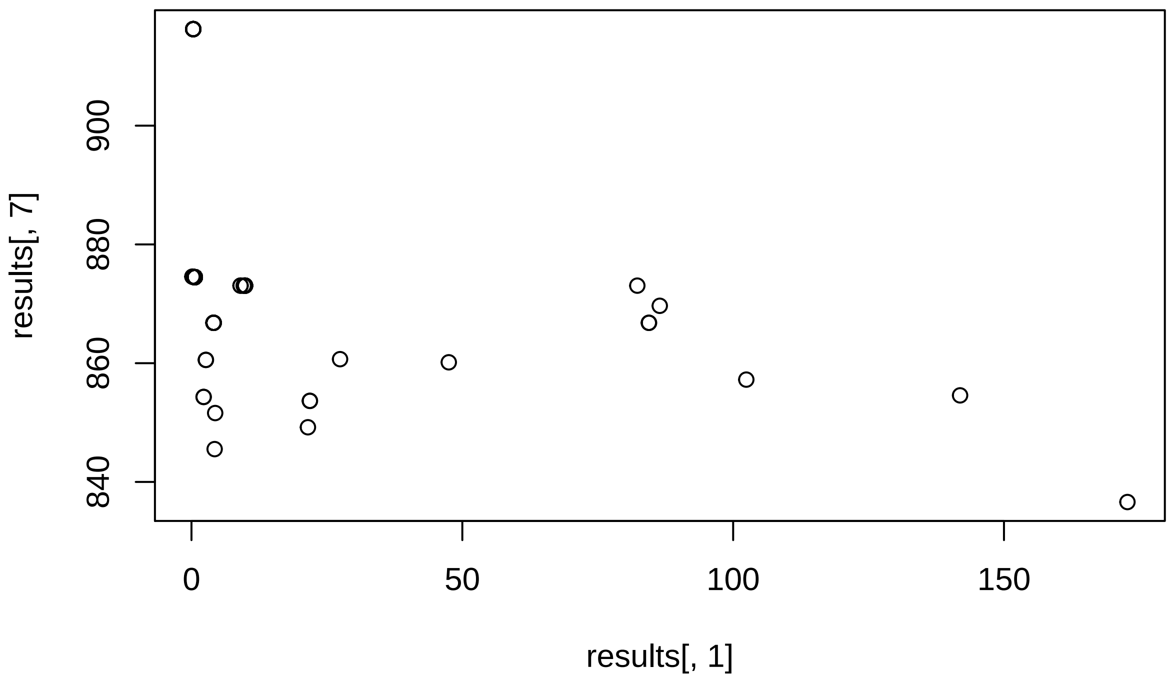
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

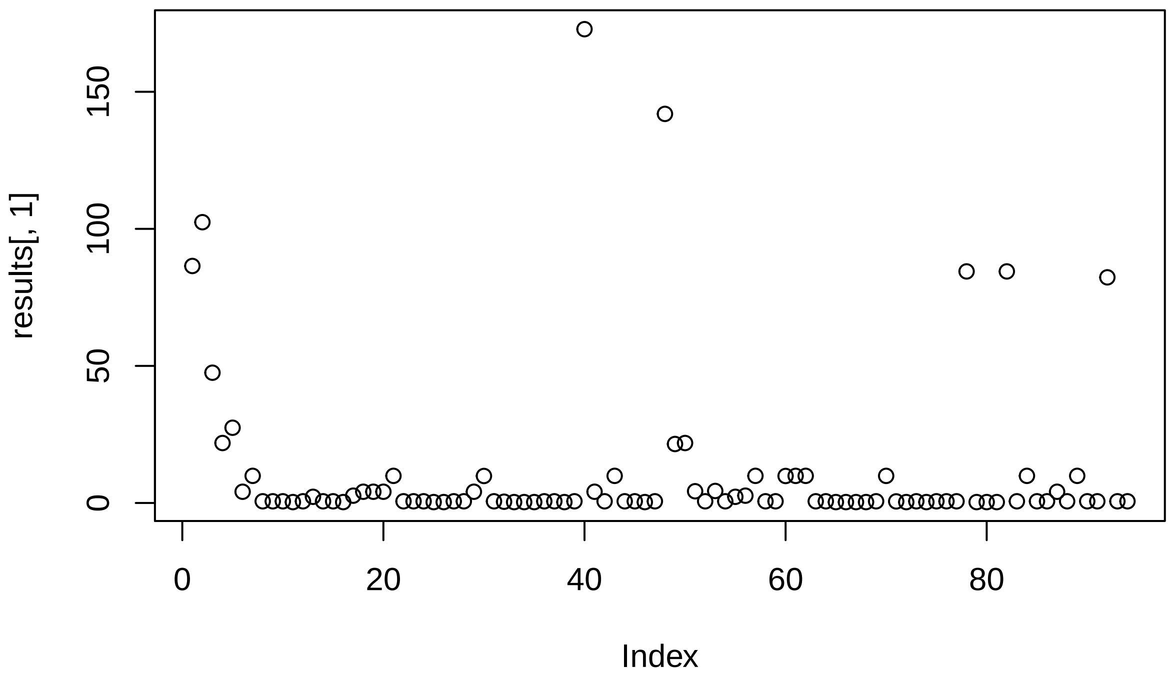




```
##  1 factor completed in 0.00455 minutes.    Estimated time of completion: 2020-04-28 10:39:34    [1]
##  [7]  20.0000010  50.0000010  78.6860377 874.5756471 172.7931725 836.6163381
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

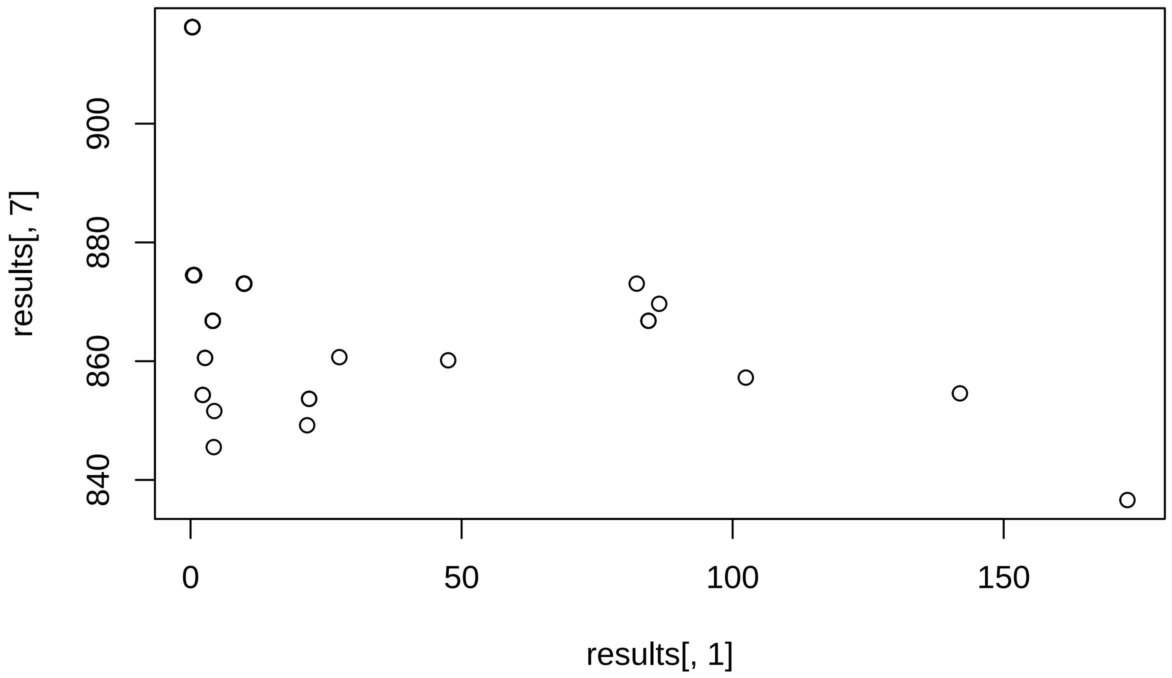
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

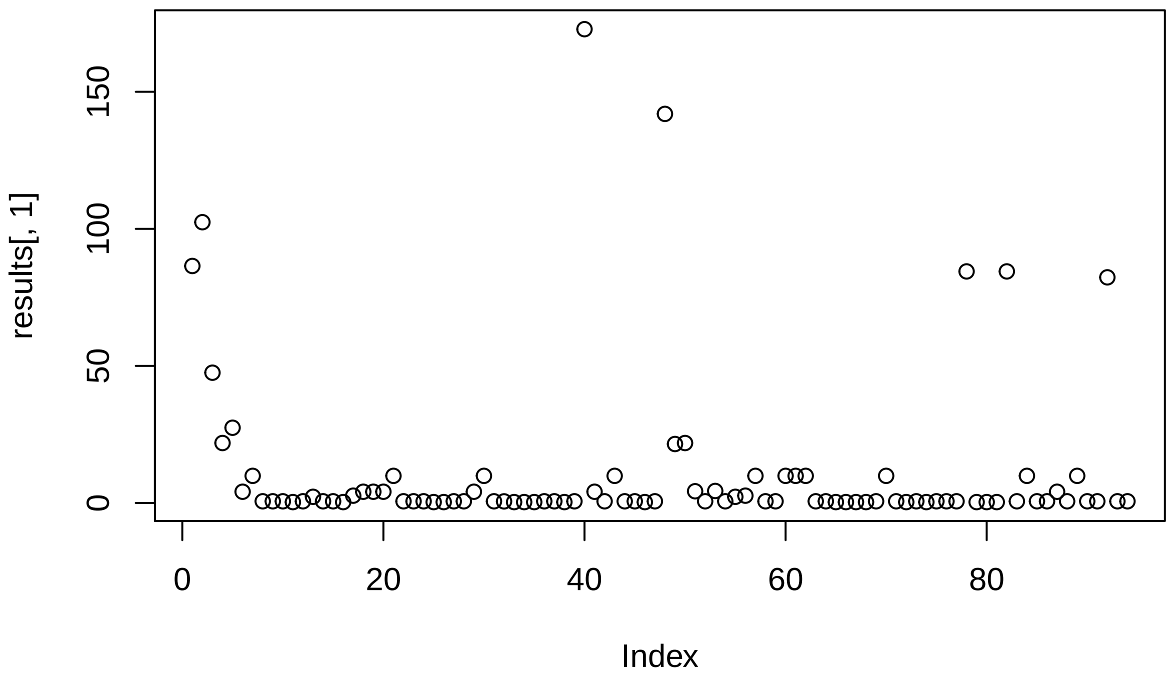




```
##  1 factor completed in 0.00409 minutes.    Estimated time of completion: 2020-04-28 10:39:36    [1]
##  [7]  30.0000010  50.0000010  78.6860377 916.2541709 172.8350429 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

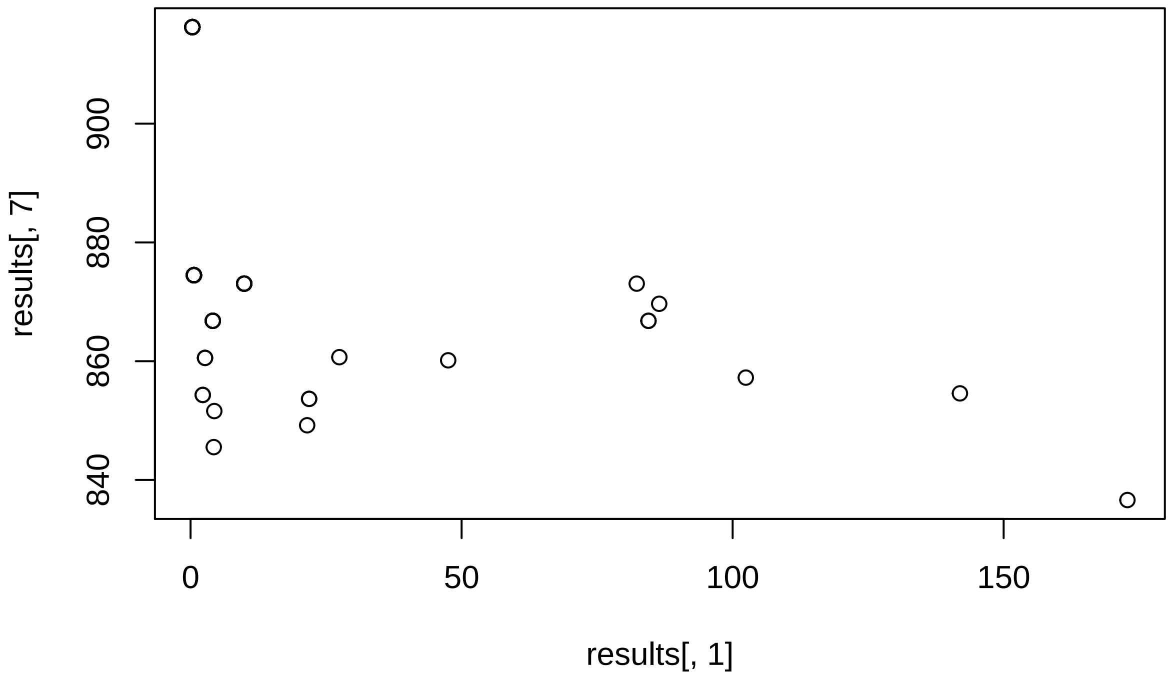
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

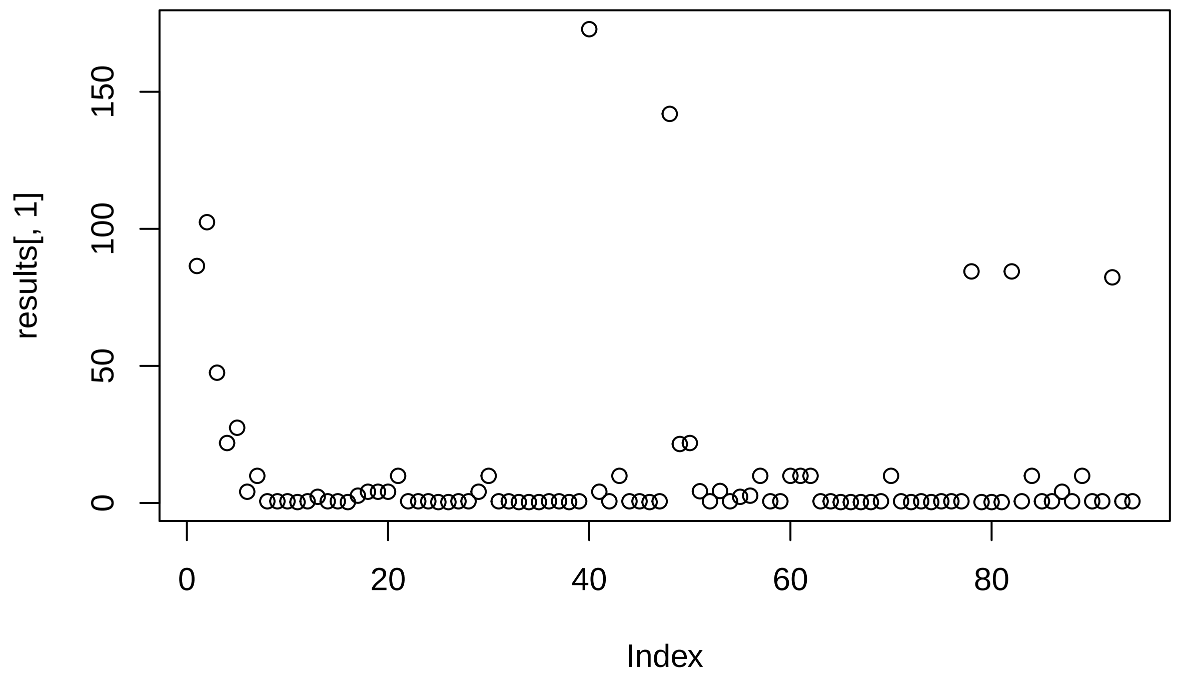




```
##  1 factor completed in 0.00492 minutes.    Estimated time of completion: 2020-04-28 10:39:39    [1]
##  [7]  40.0000010  50.0000010  78.6860377 916.2541709 172.8354154 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

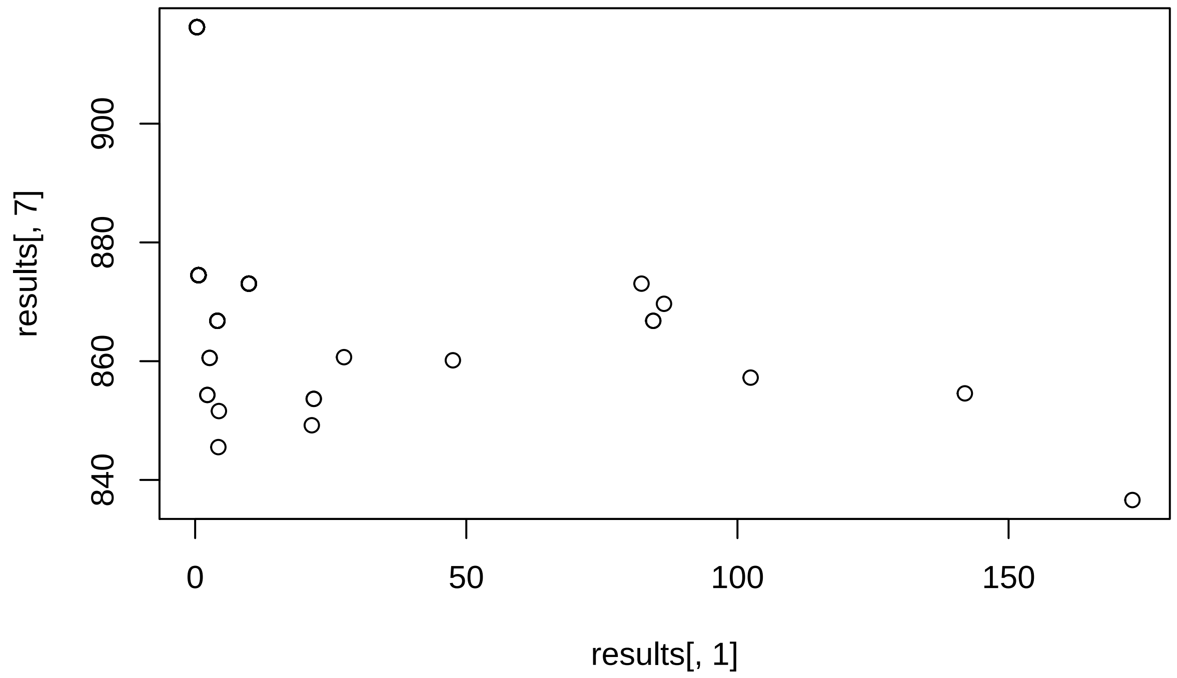
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

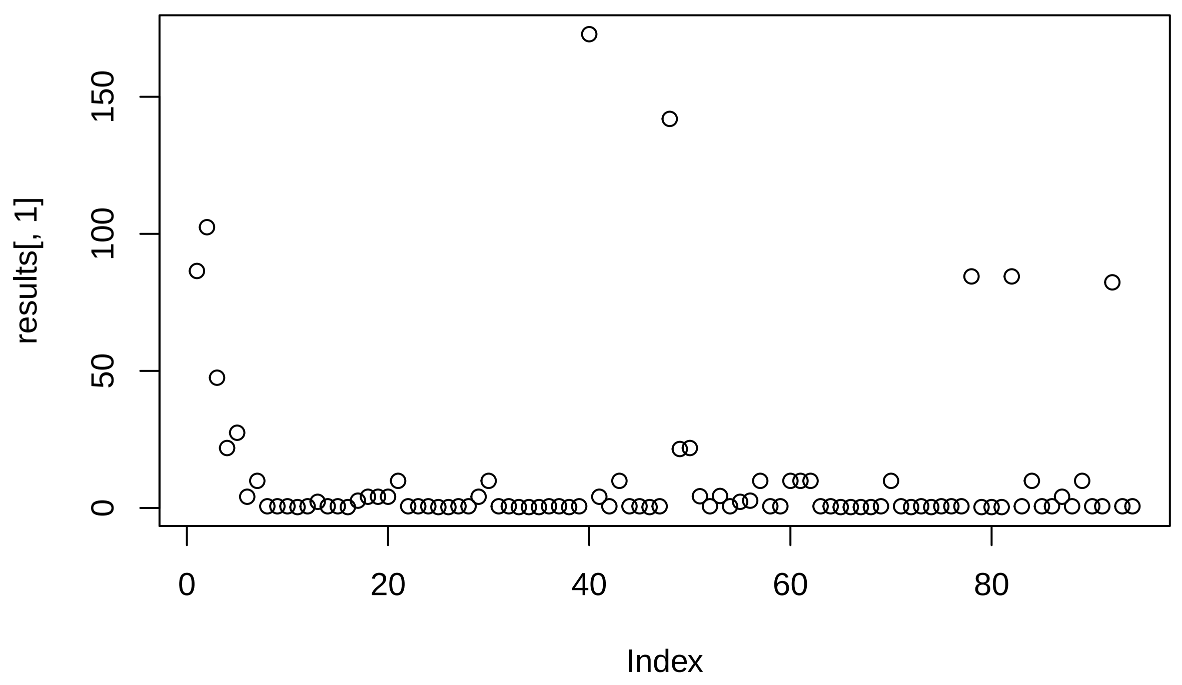




```
##  1 factor completed in 0.00726 minutes.    Estimated time of completion: 2020-04-28 10:39:42    [1]
##  [7]  50.0000010  50.0000010  78.6860377 916.2541709 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

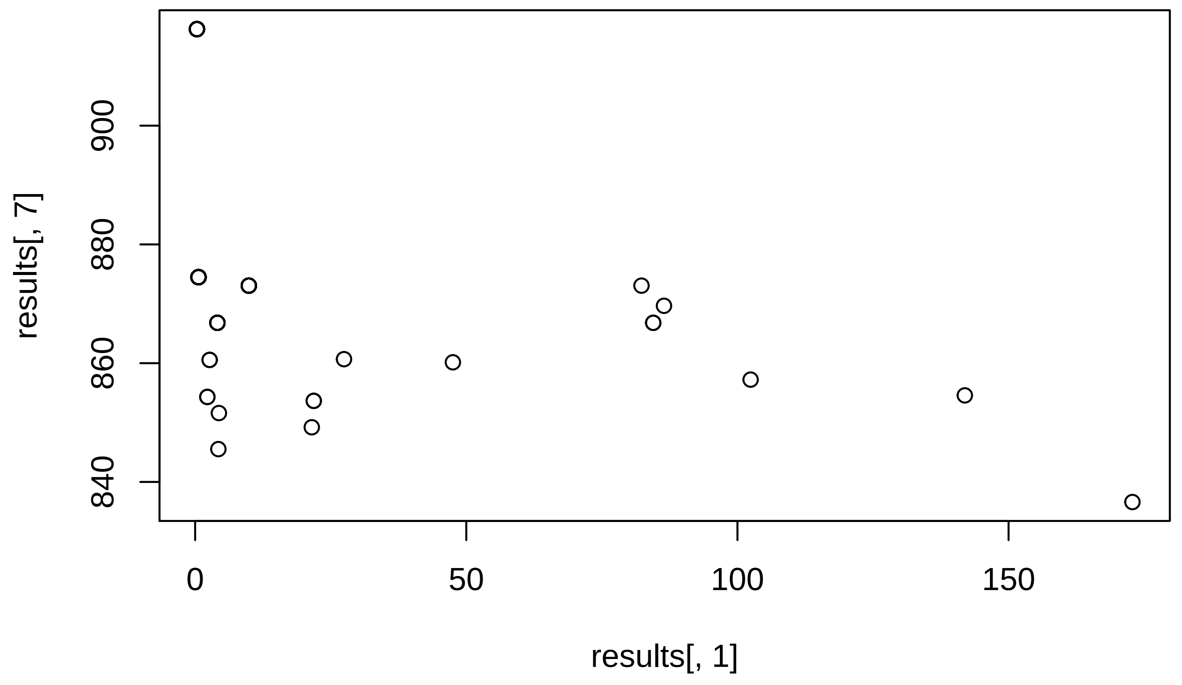
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

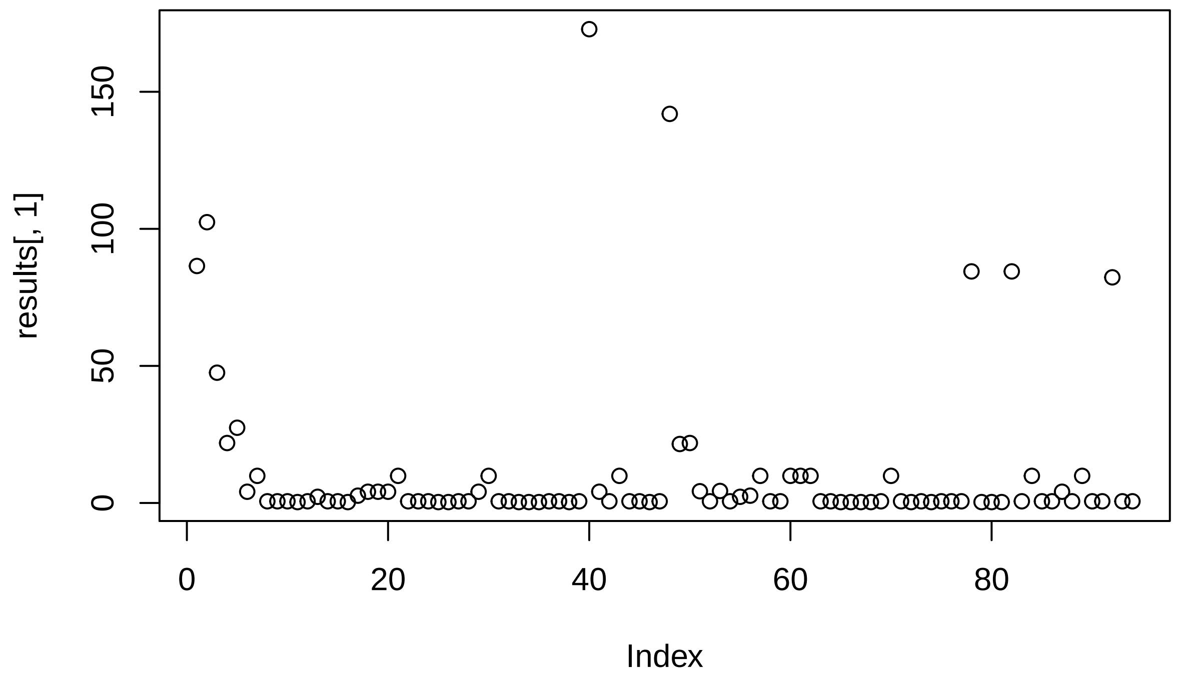




```
##  1 factor completed in 0.00536 minutes.    Estimated time of completion: 2020-04-28 10:39:45    [1]
##  [7]  60.0000010  50.0000010  78.6860377 916.2541709 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

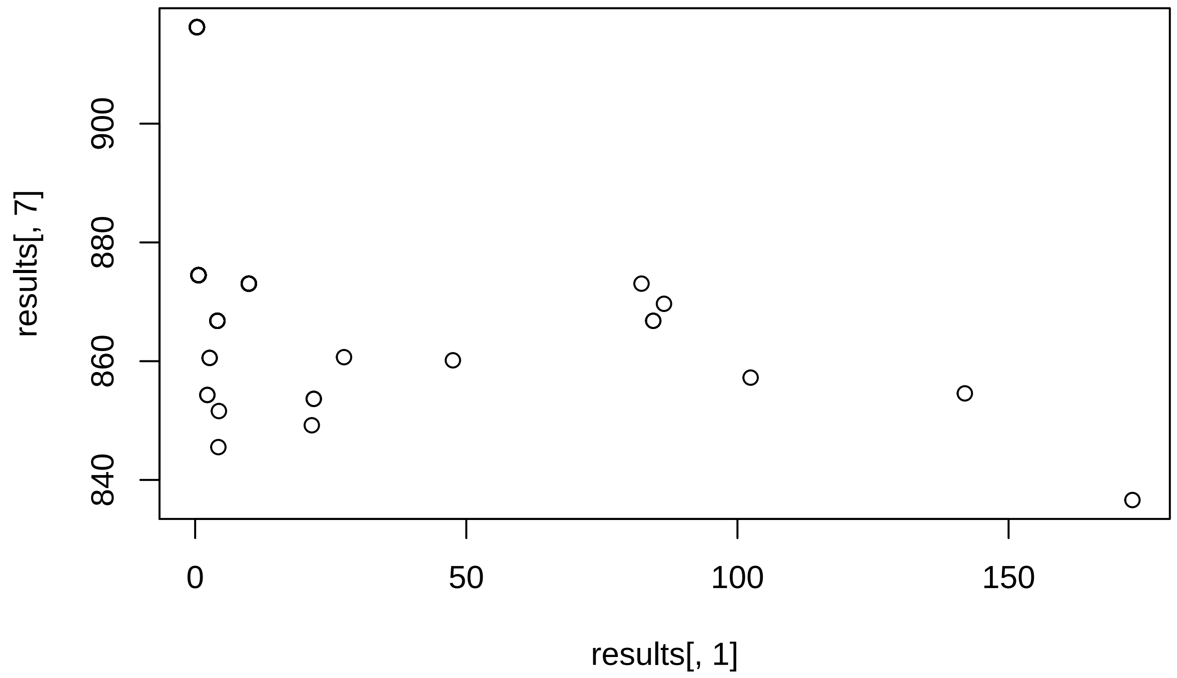
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

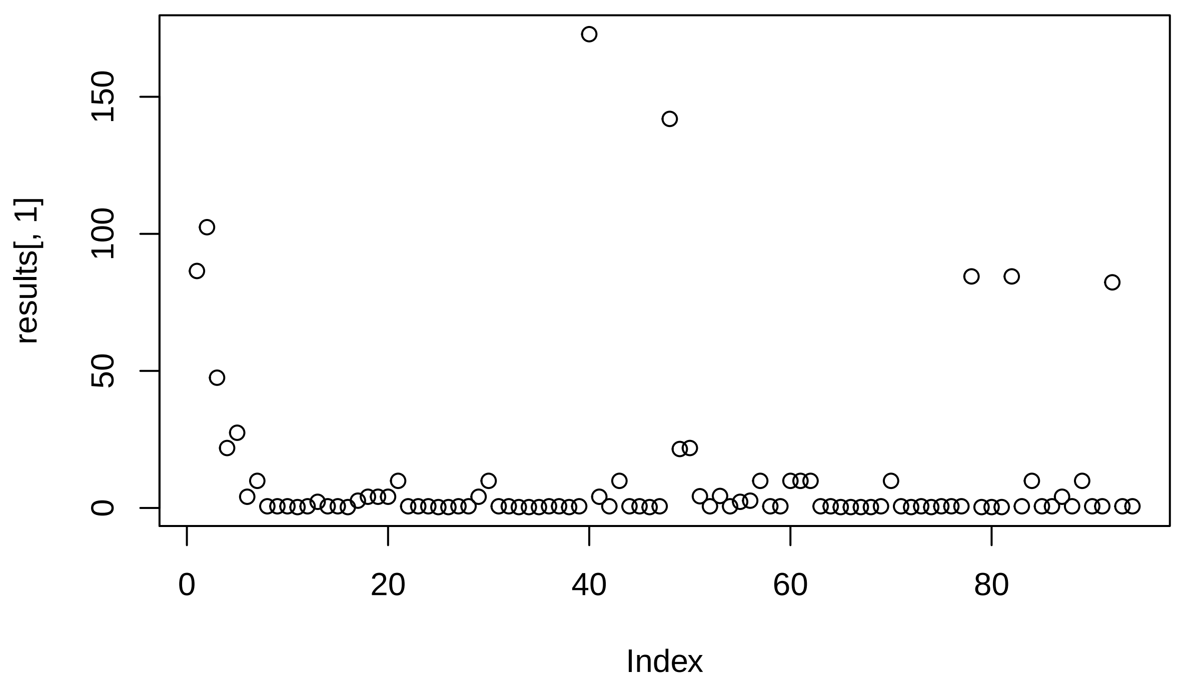




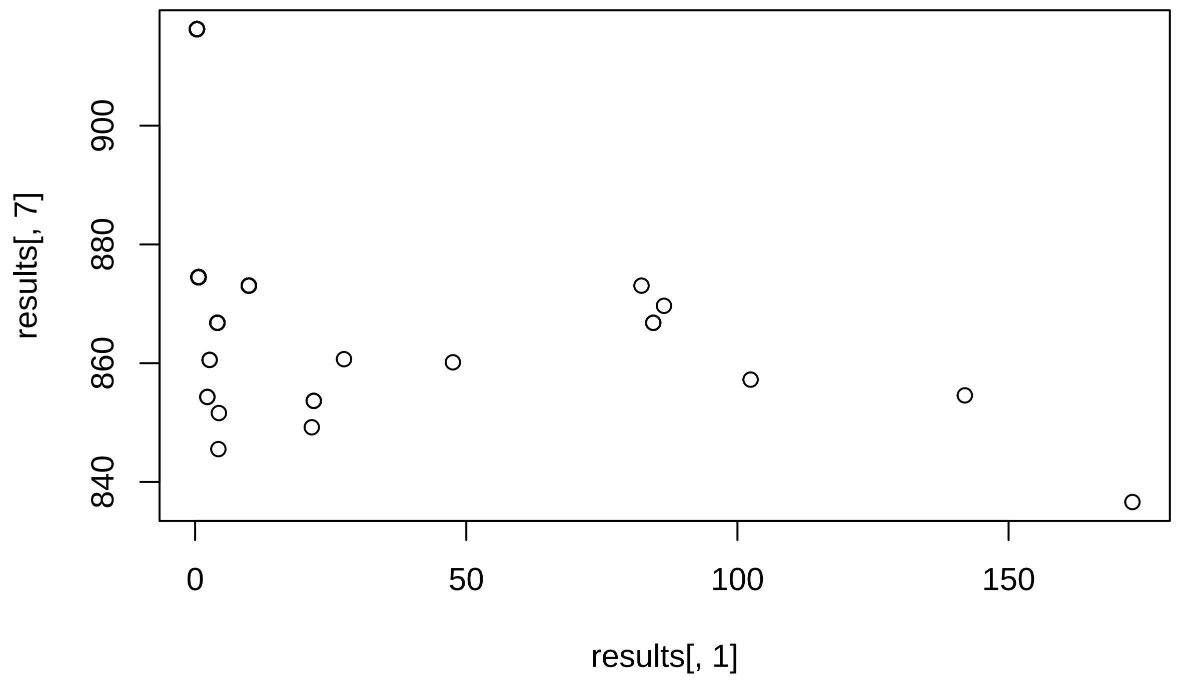
```
##  1 factor completed in 0.00994 minutes.    Estimated time of completion: 2020-04-28 10:39:48    [1]
##  [7]  70.0000010  50.0000010  78.6860377 916.2541709 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

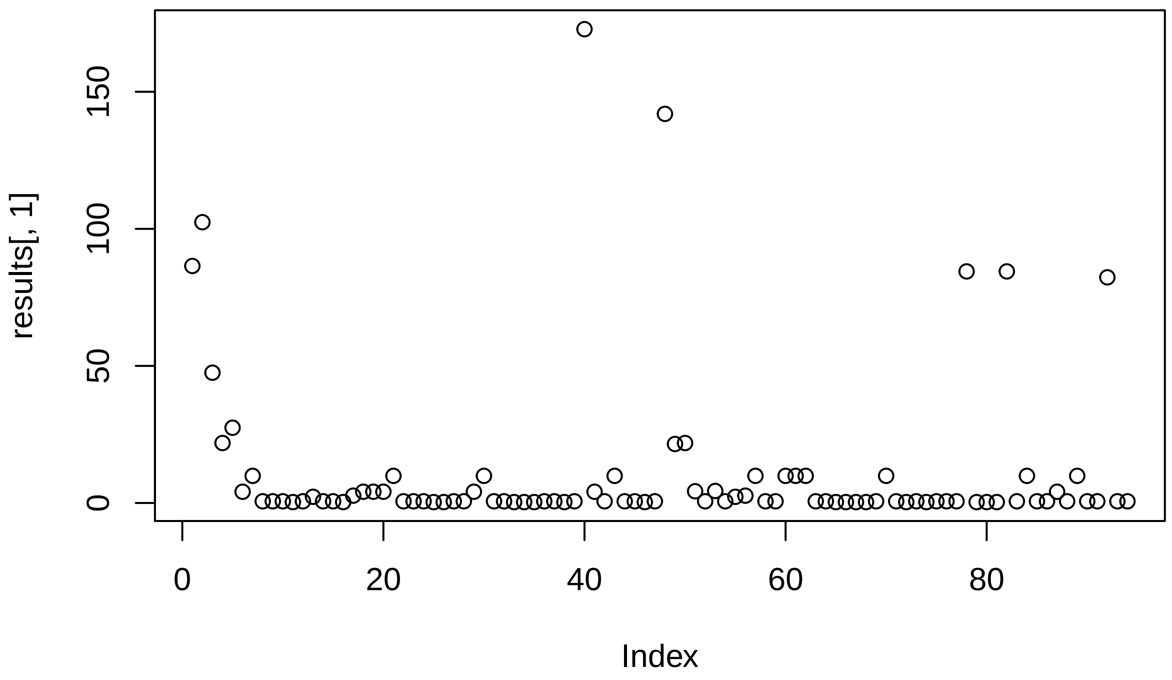
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





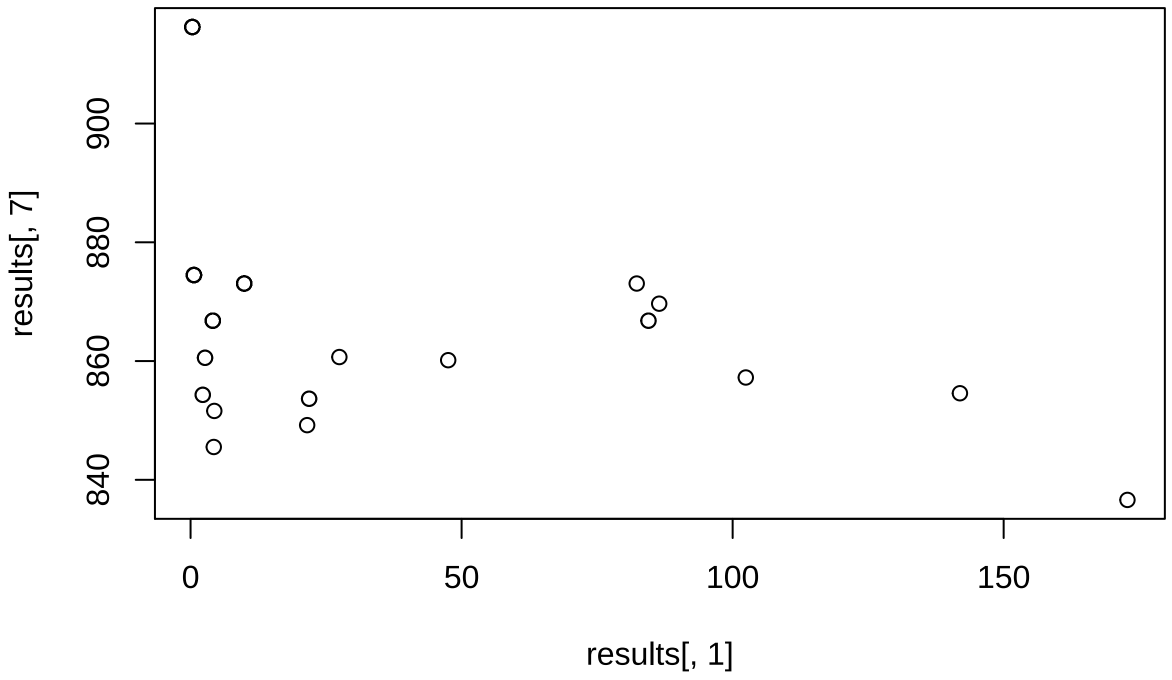
```
##  1 factor completed in 0.00609 minutes.    Estimated time of completion: 2020-04-28 10:39:52    [1]
##  [7]  80.0000010  50.0000010  78.6860377 916.2541709 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

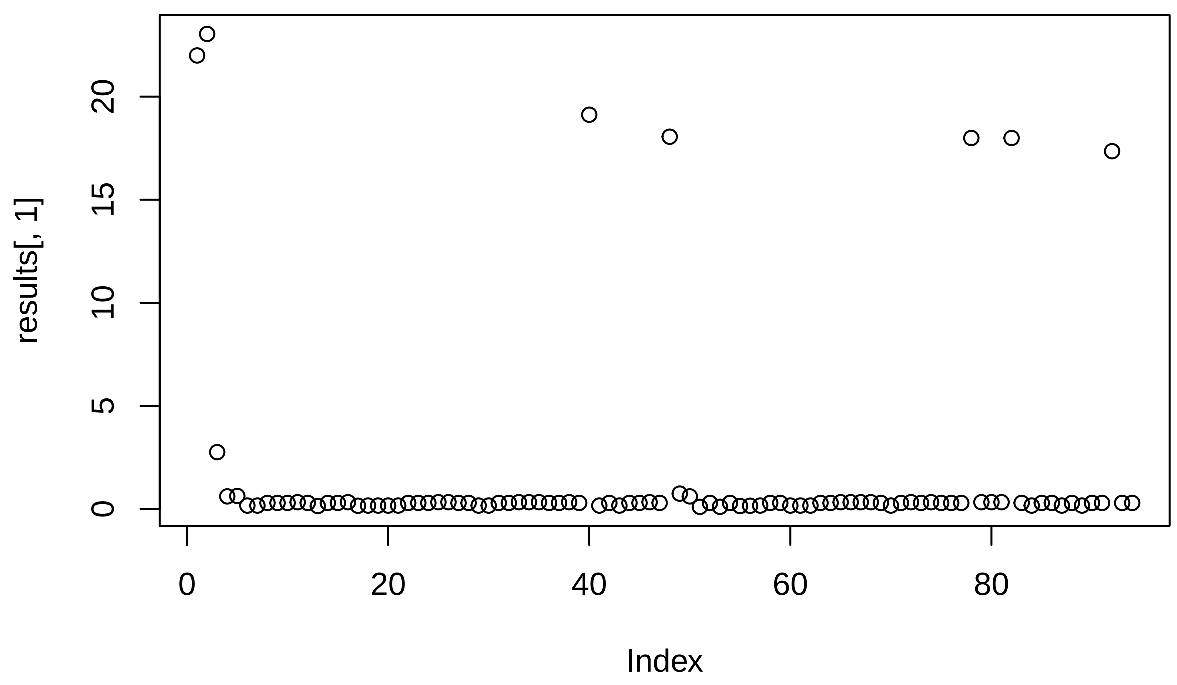




```
##  1 factor completed in 0.00427 minutes.    Estimated time of completion: 2020-04-28 10:39:55    [1]
##  [7]  90.0000010  50.0000010  78.6860377 916.2541709 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 60.000001"
## [1] "j = 1e-06"
```

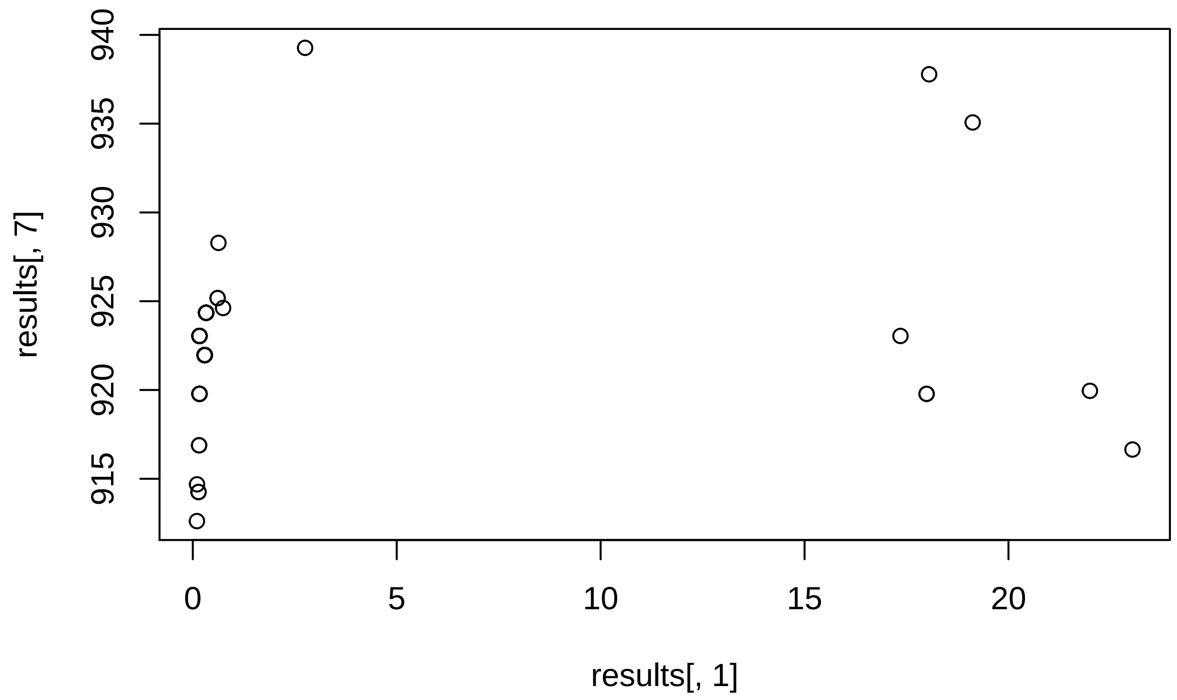
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

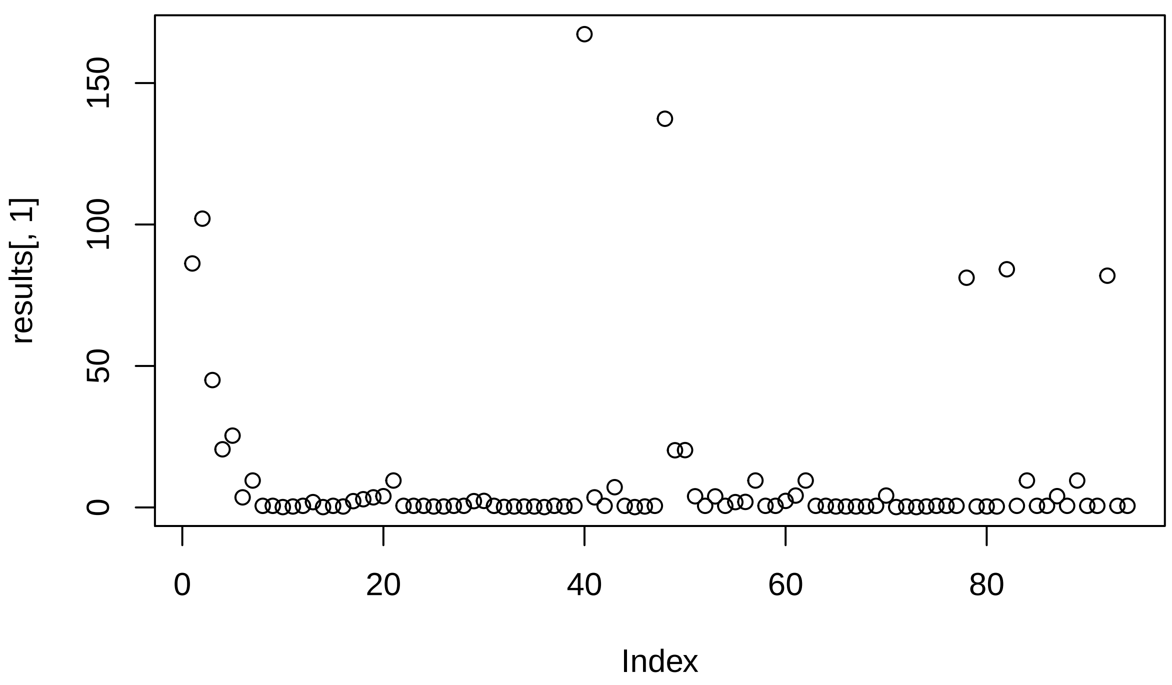




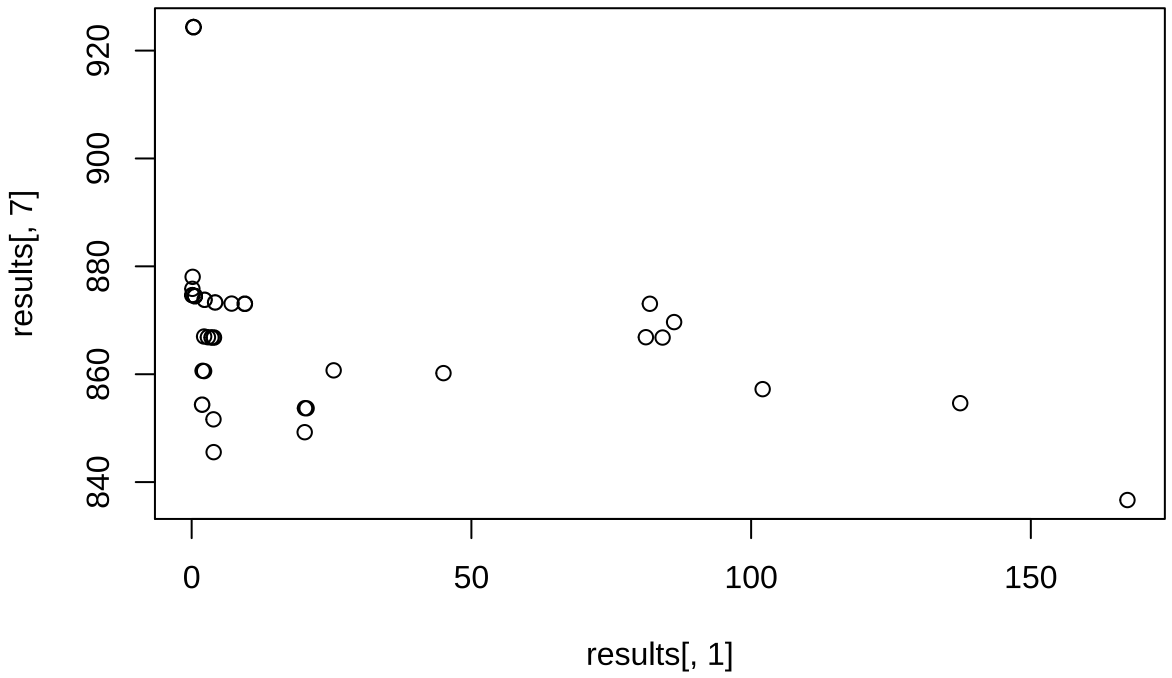
```
##  1 factor completed in 0.00432 minutes.    Estimated time of completion: 2020-04-28 10:39:57    [1]
##  [7]   0.0000010  60.0000010  78.6860377 912.6168040   0.1012633 912.6168040
## [13]  51.0000000  51.0000000
## [1] "j = 10.000001"
```

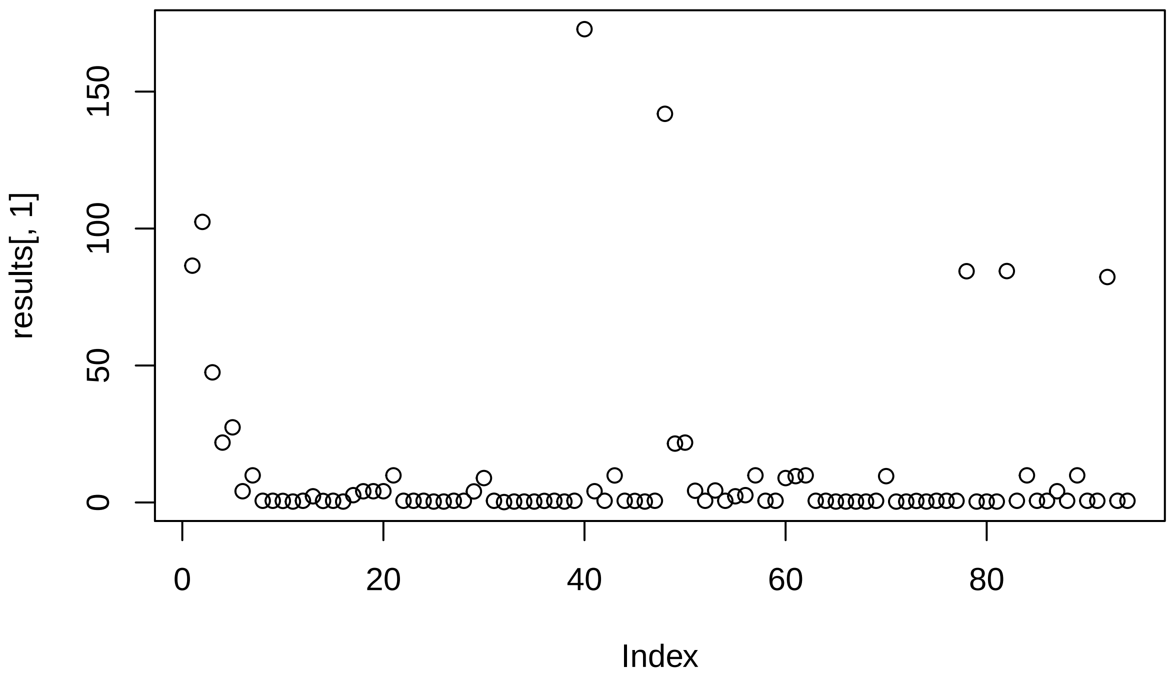
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





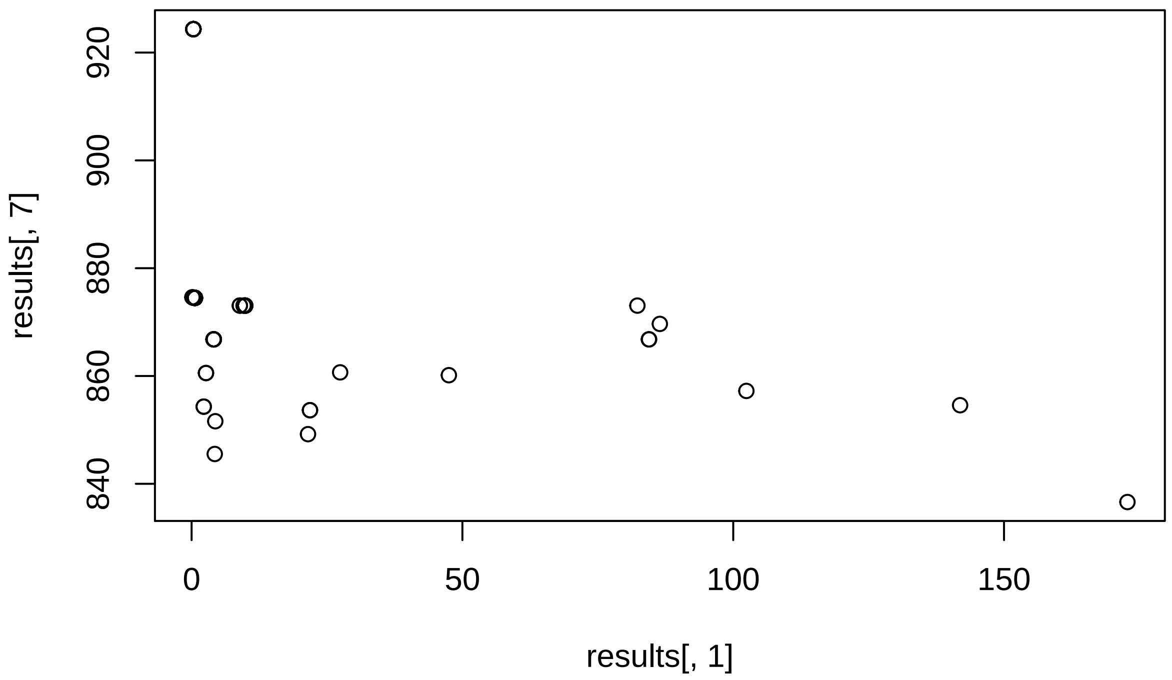
```
##  1 factor completed in 0.00471 minutes.    Estimated time of completion: 2020-04-28 10:39:59    [1]
##  [7]  10.0000010  60.0000010  78.6860377 874.6573526 167.2696361 836.6745659
## [13]  10.0000000  40.0000000
## [1] "j = 20.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

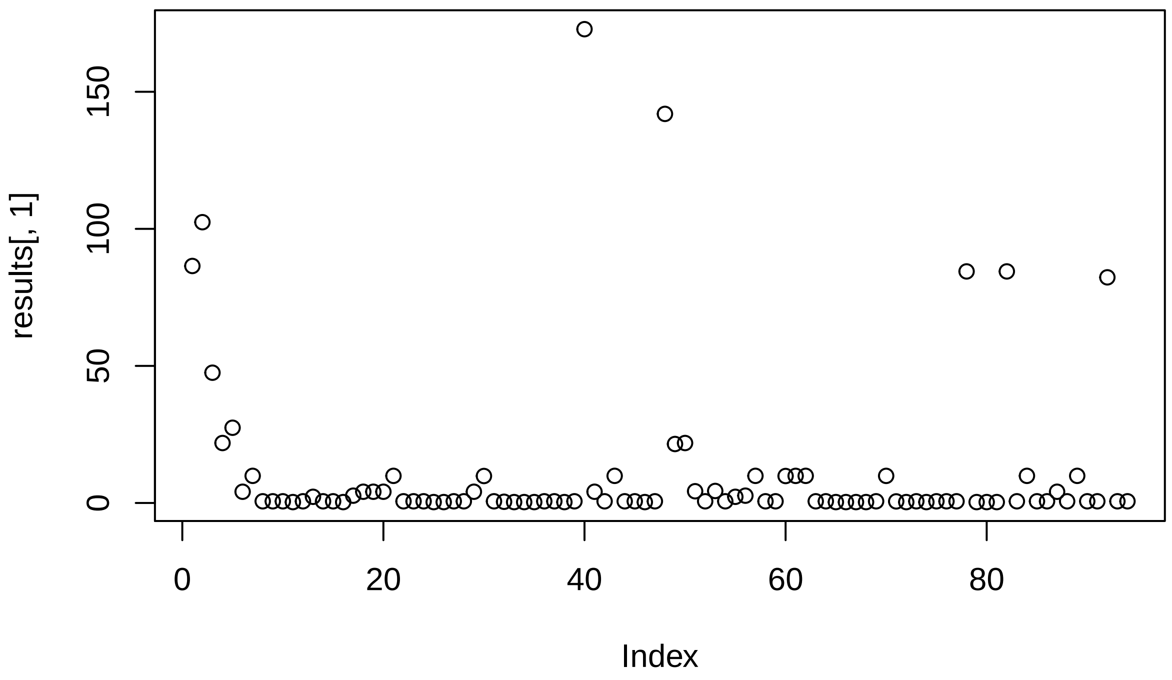




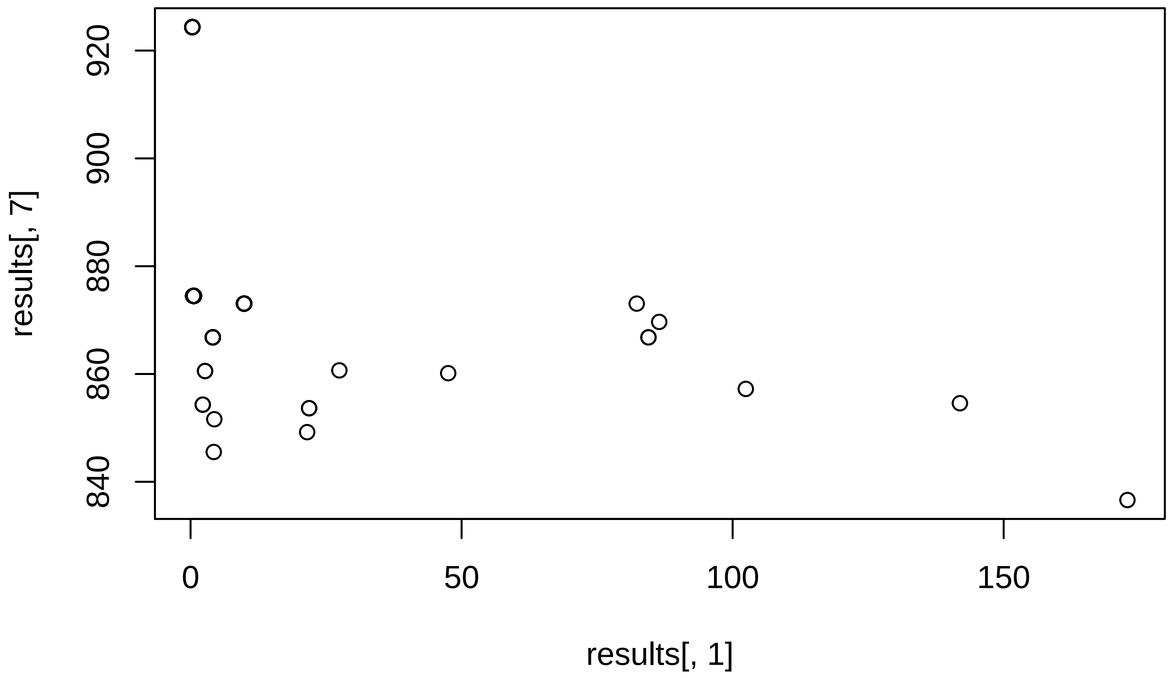
```
##  1 factor completed in 0.00476 minutes.    Estimated time of completion: 2020-04-28 10:40:01    [1]
##  [7]  20.0000010  60.0000010  78.6860377 874.6129529 172.7847251 836.6163395
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

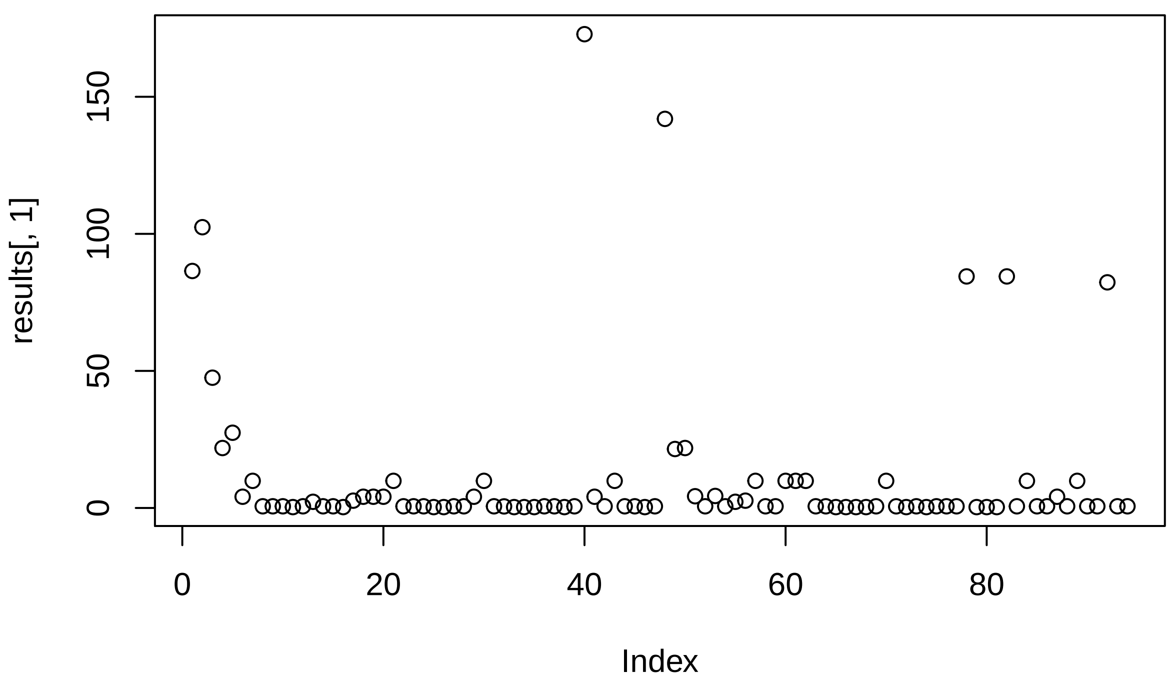
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





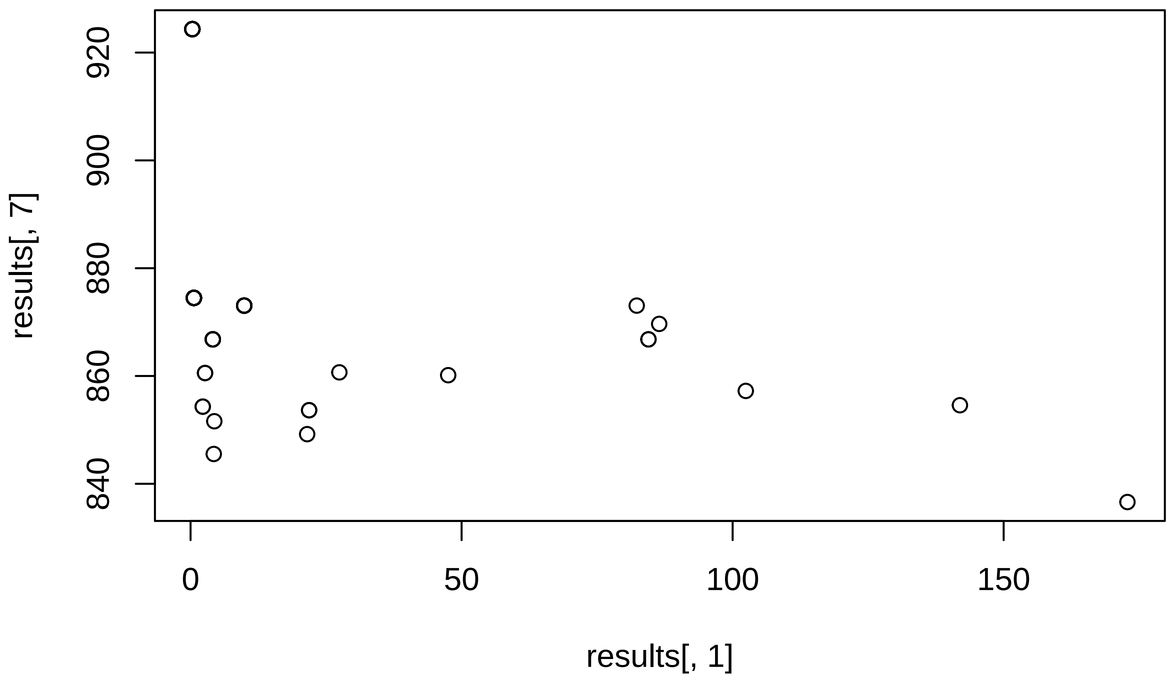
```
##  1 factor completed in 0.00422 minutes.    Estimated time of completion: 2020-04-28 10:40:03    [1]
##  [7]  30.0000010  60.0000010  78.6860377 924.3483996 172.8349677 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

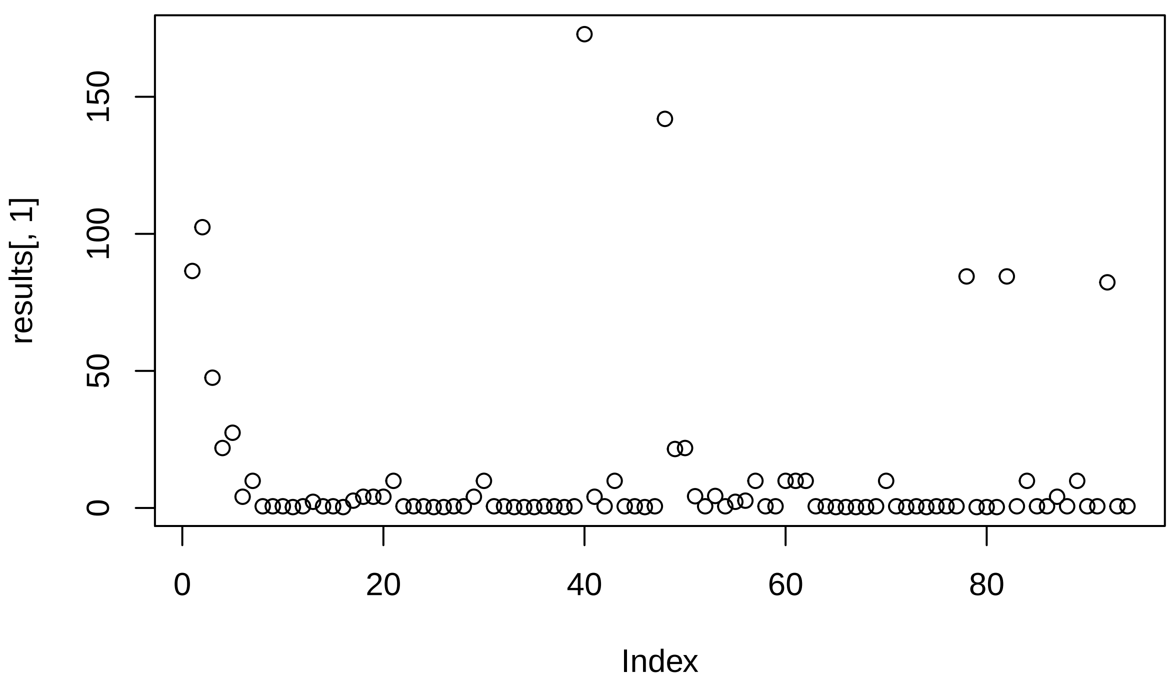




```
##  1 factor completed in 0.00415 minutes.    Estimated time of completion: 2020-04-28 10:40:05    [1]
##  [7]  40.0000010  60.0000010  78.6860377 924.3483996 172.8354148 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

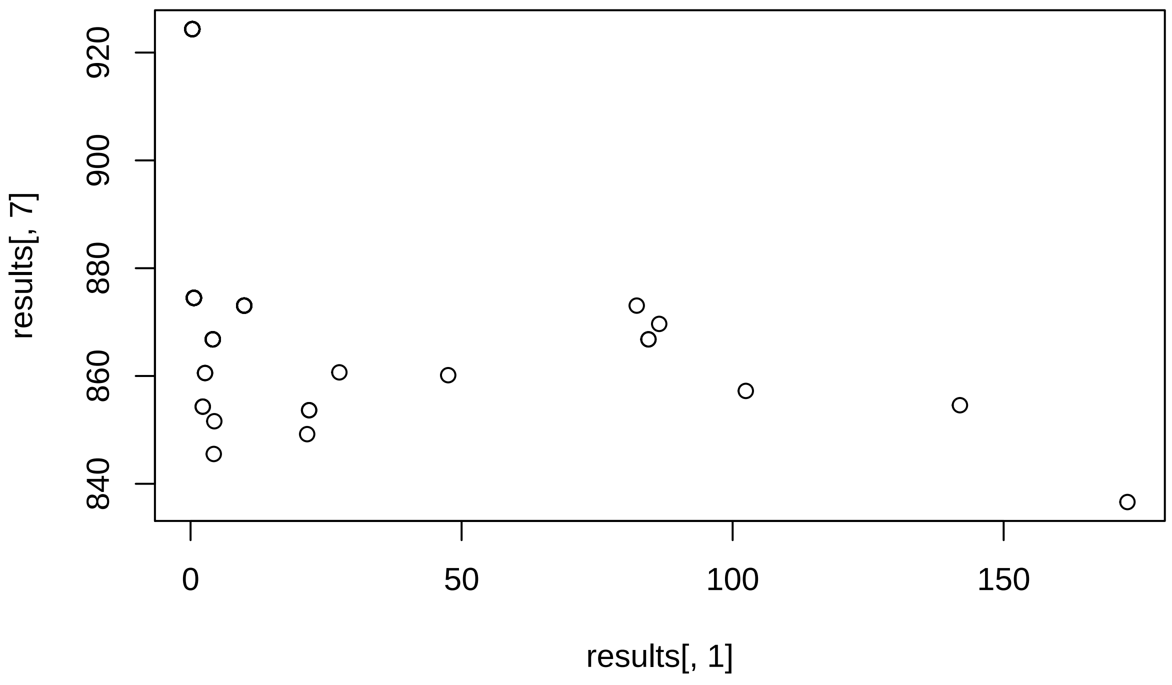
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

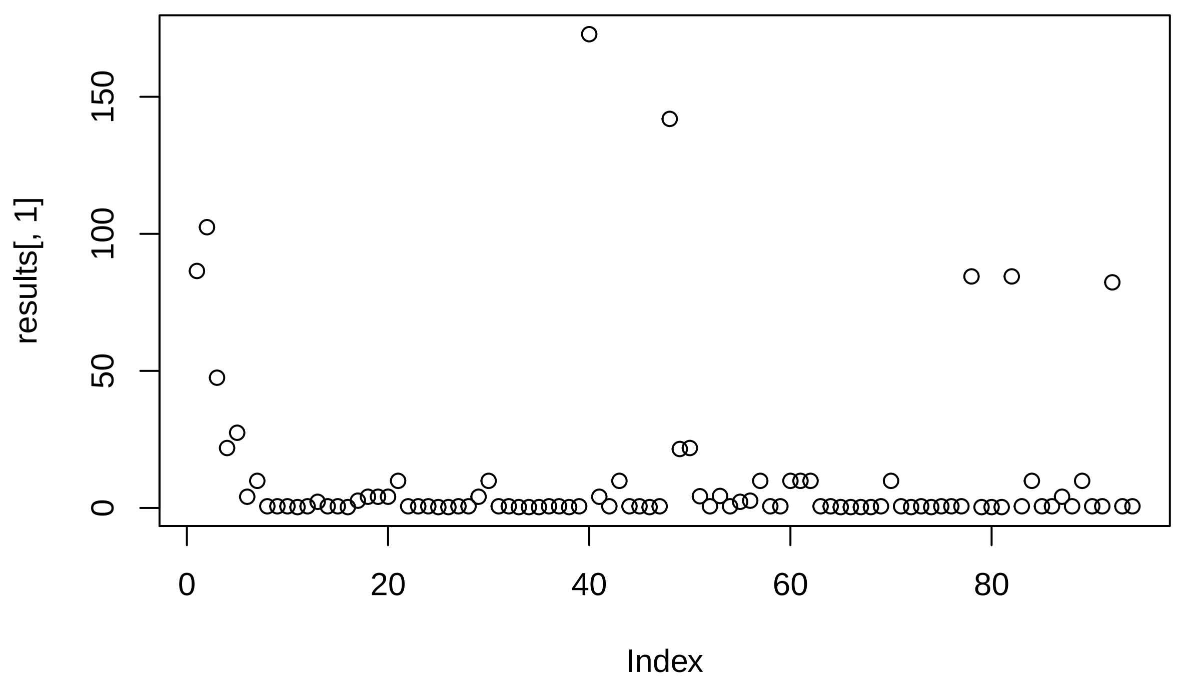




```
##  1 factor completed in 0.00759 minutes.    Estimated time of completion: 2020-04-28 10:40:09    [1]
##  [7]  50.0000010  60.0000010  78.6860377 924.3483996 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

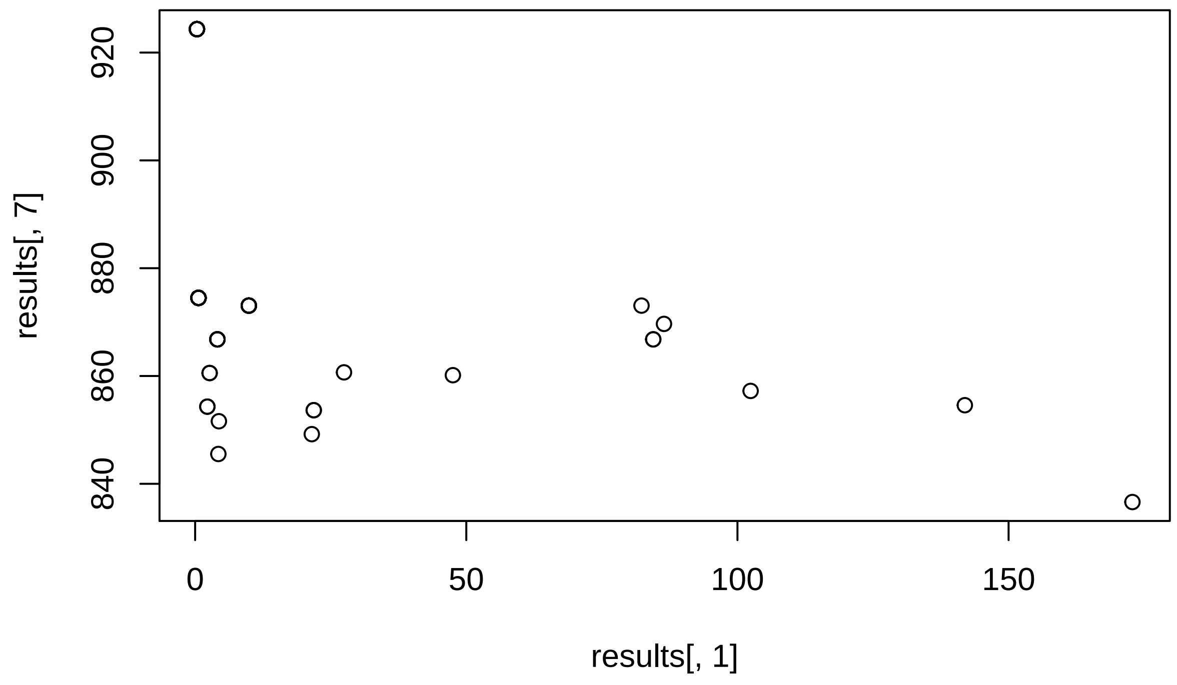
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

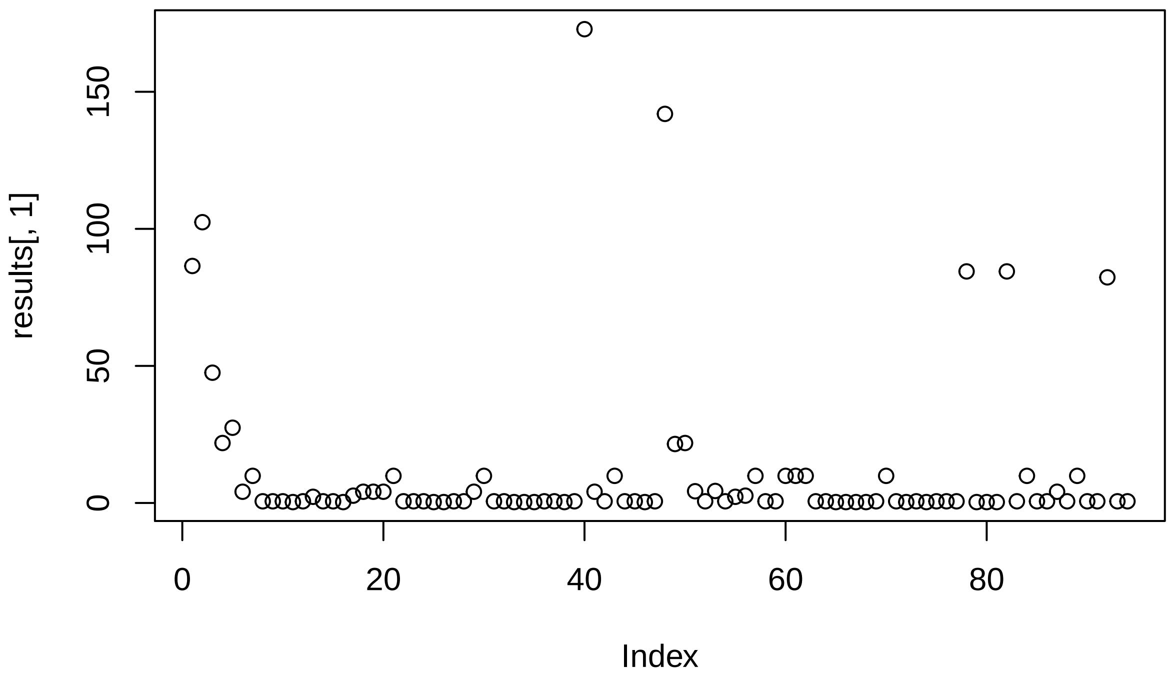




```
##  1 factor completed in 0.00532 minutes.    Estimated time of completion: 2020-04-28 10:40:11    [1]
##  [7]  60.0000010  60.0000010  78.6860377 924.3483996 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

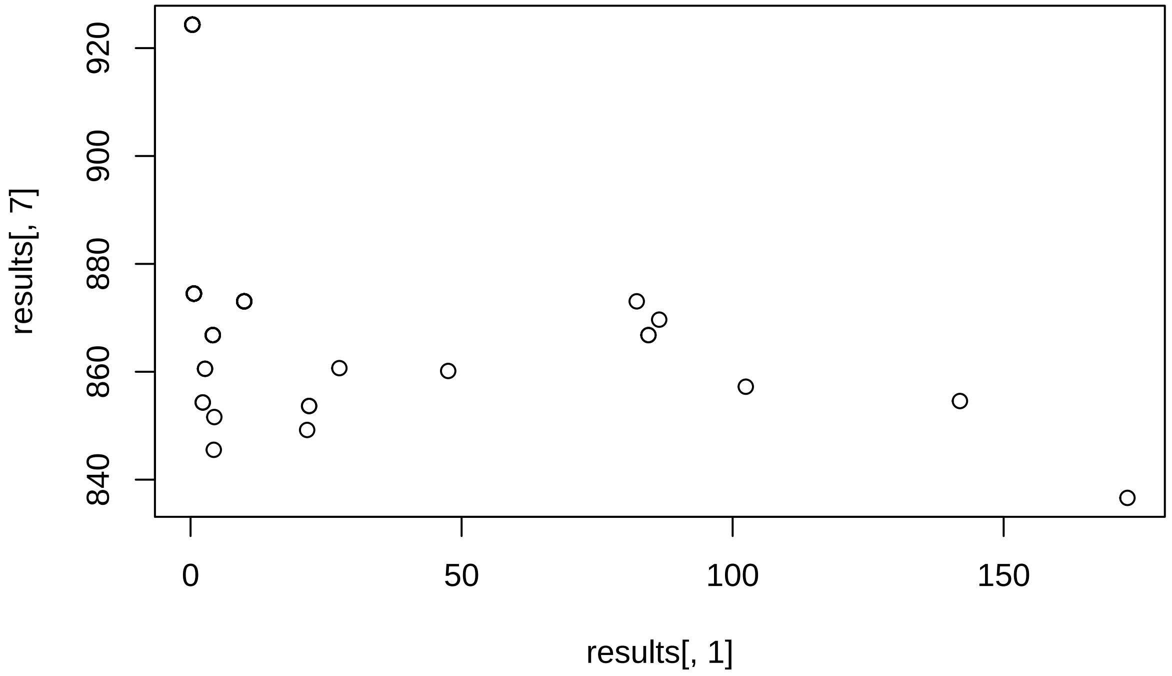
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

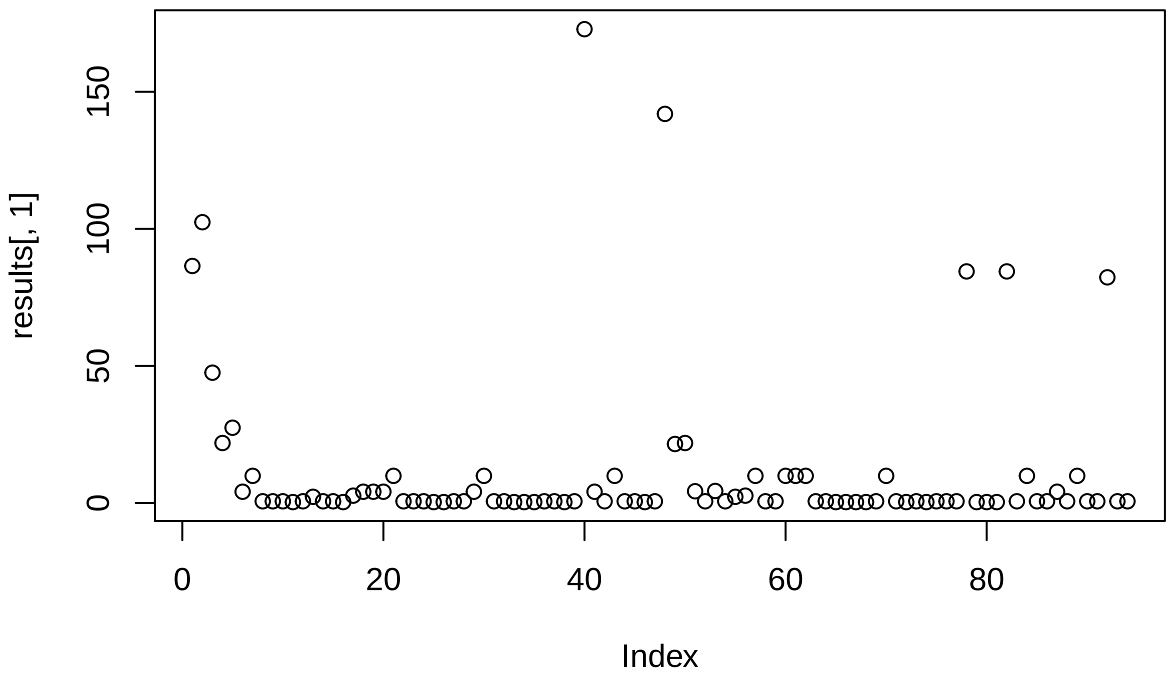




```
##  1 factor completed in 0.00434 minutes.    Estimated time of completion: 2020-04-28 10:40:13    [1]
##  [7]  70.0000010  60.0000010  78.6860377 924.3483996 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

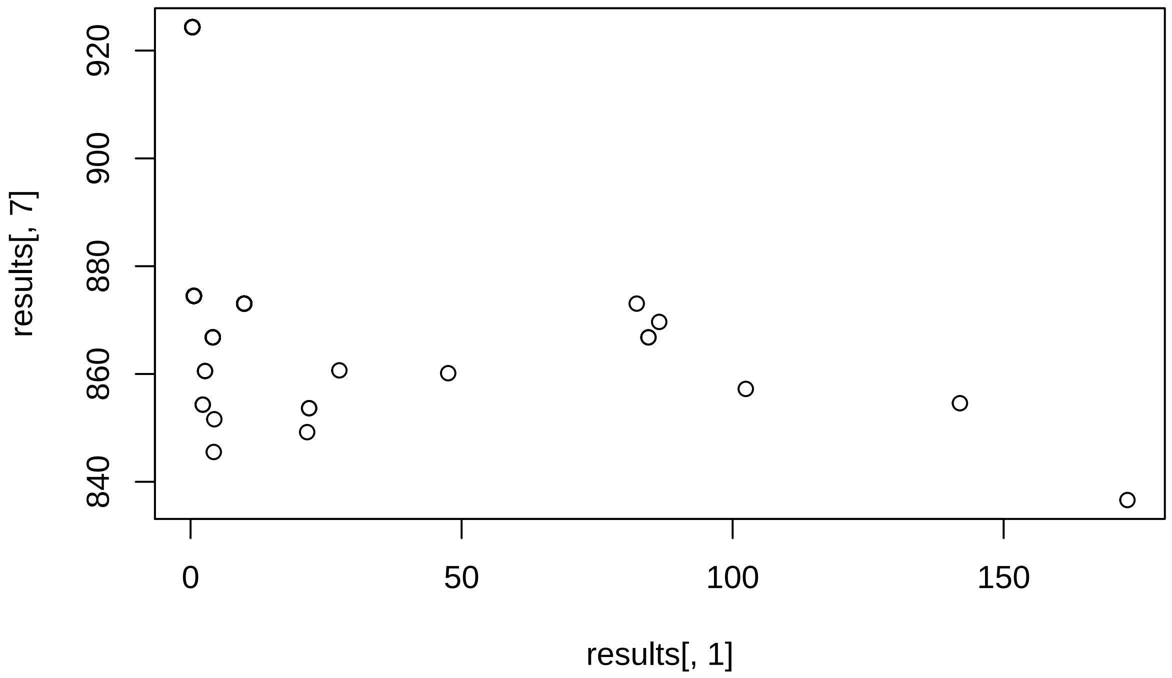
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

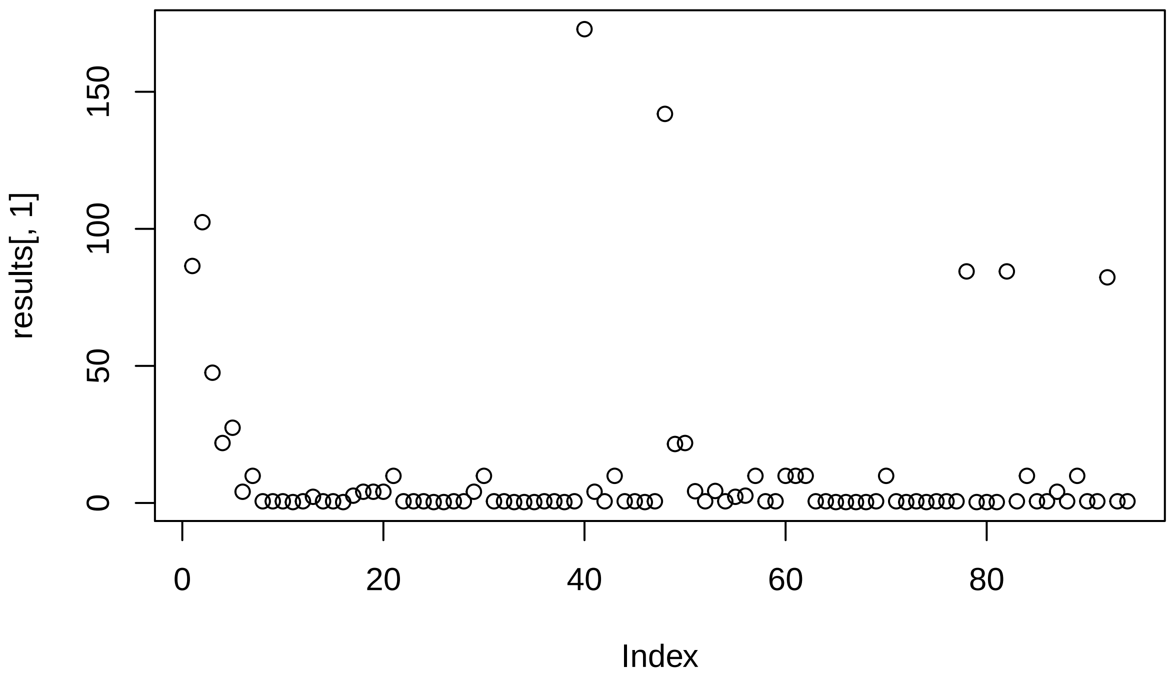




```
##  1 factor completed in 0.00448 minutes.    Estimated time of completion: 2020-04-28 10:40:16    [1]
##  [7]  80.0000010  60.0000010  78.6860377 924.3483996 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

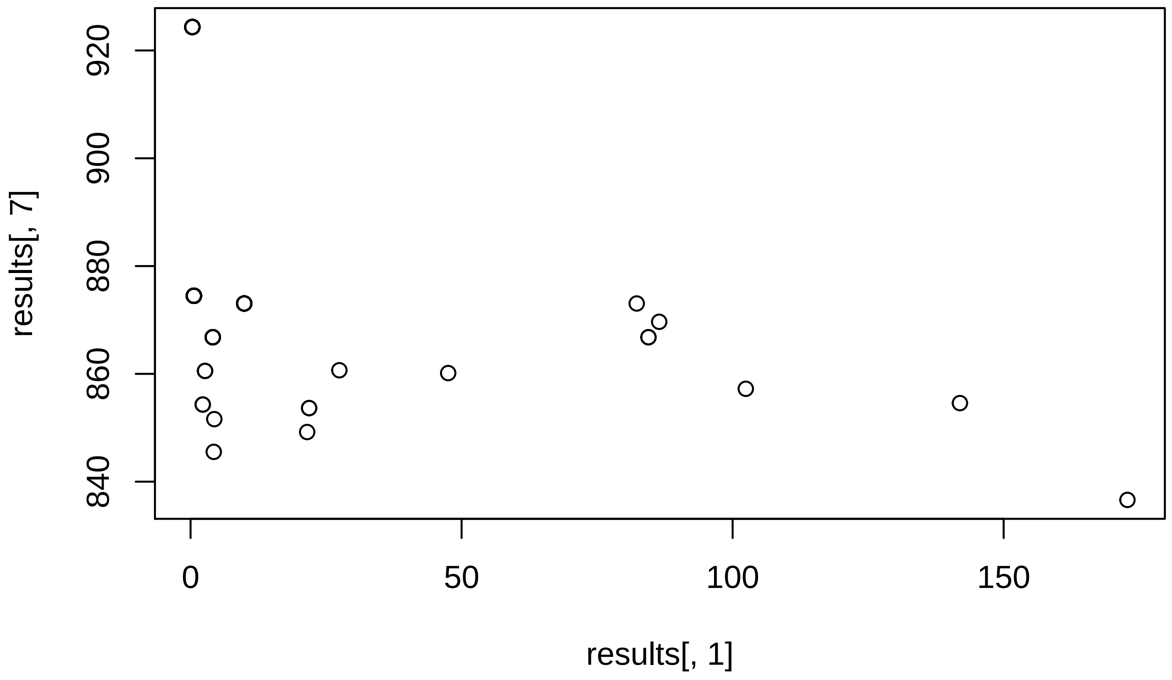
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

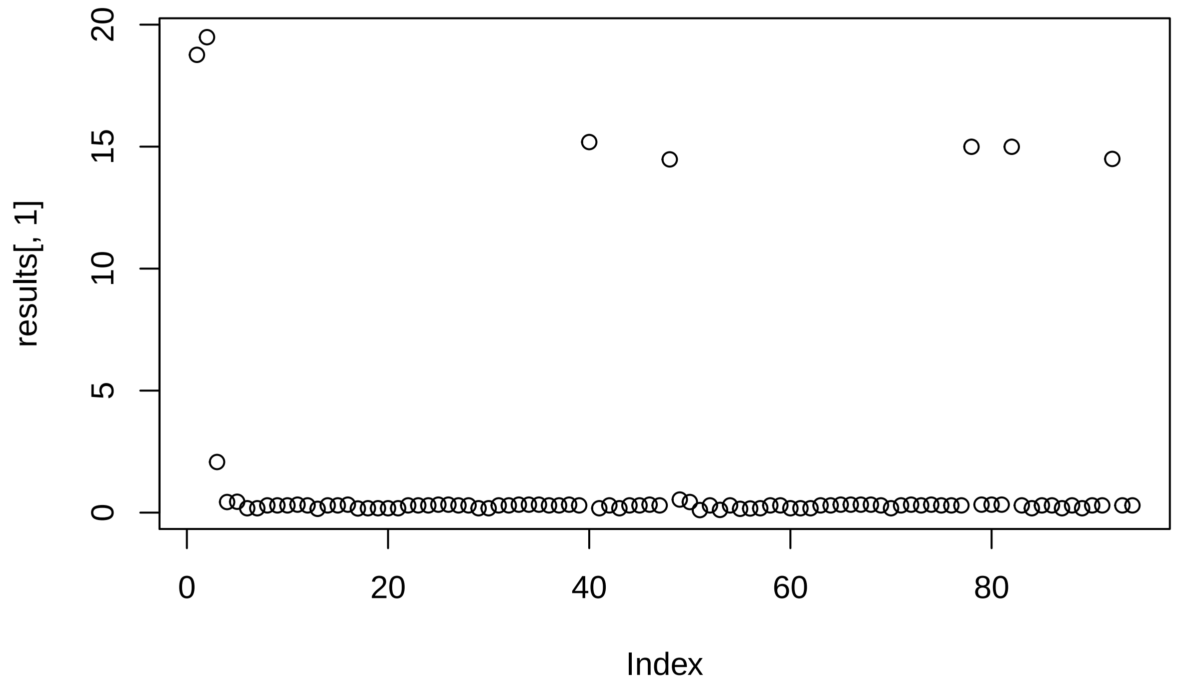




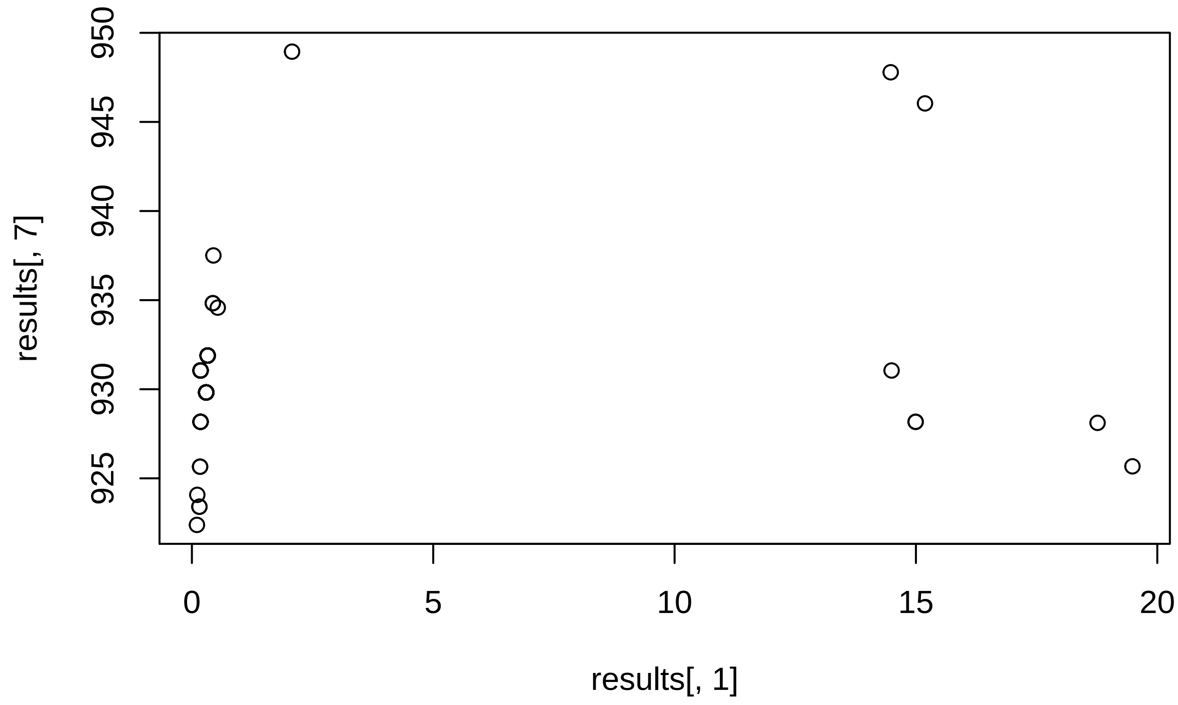
```
##  1 factor completed in 0.00428 minutes.    Estimated time of completion: 2020-04-28 10:40:18    [1]
##  [7]  90.0000010  60.0000010  78.6860377 924.3483996 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 70.000001"
## [1] "j = 1e-06"
```

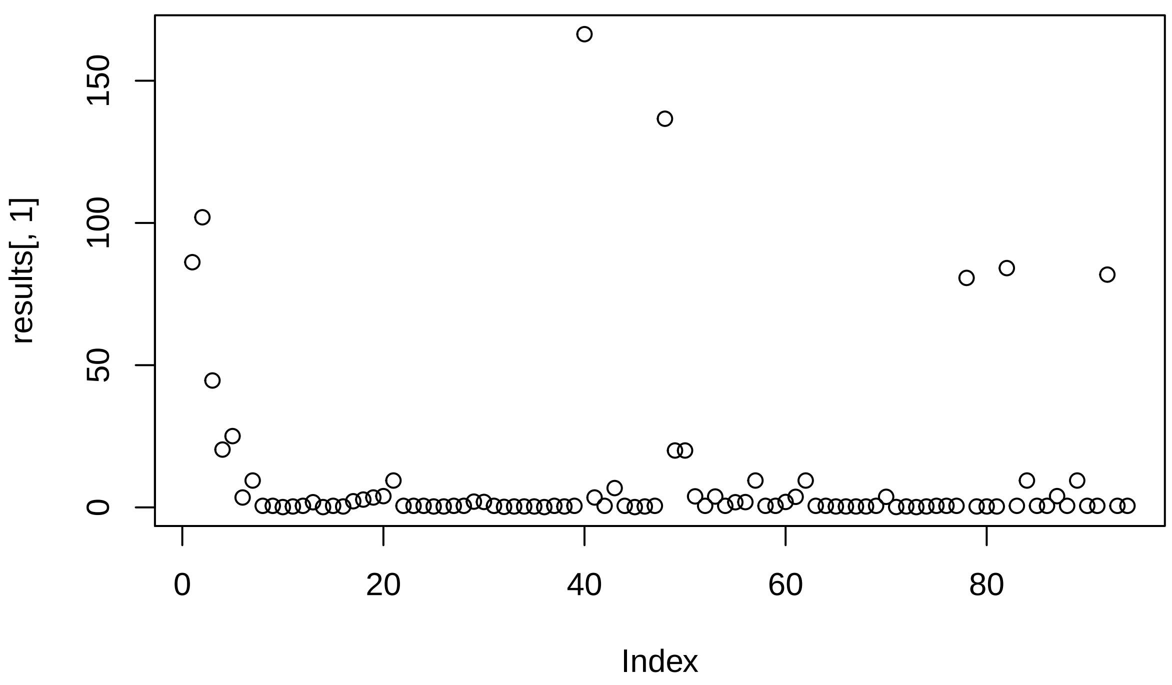
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





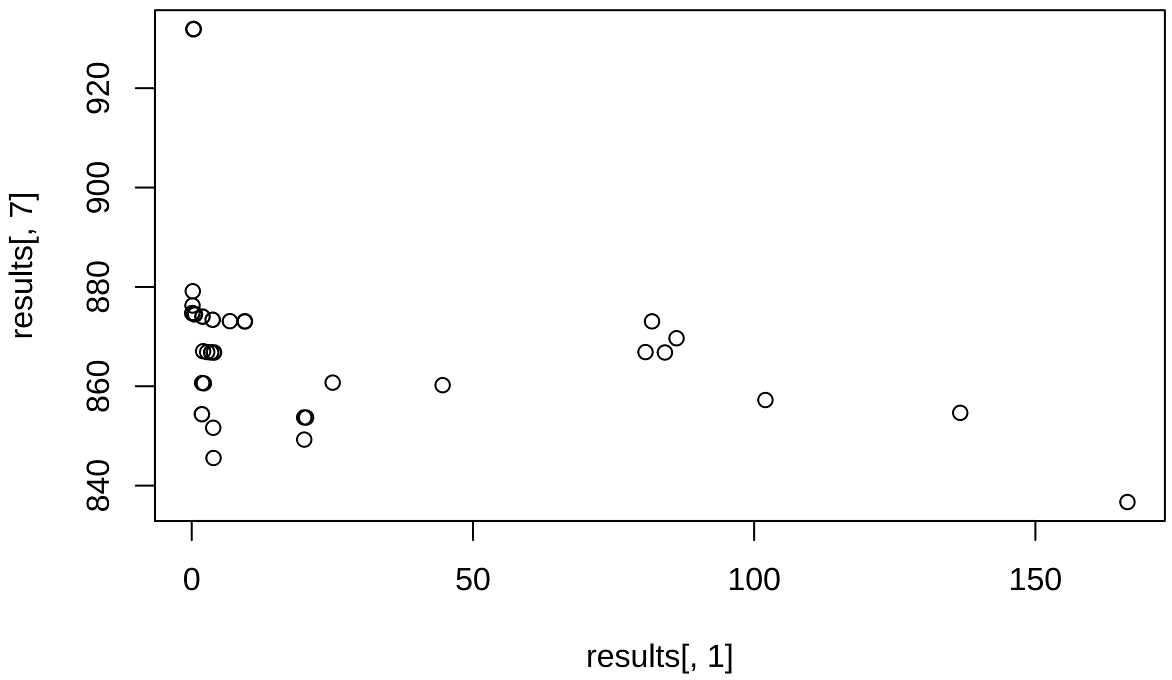
```
##  1 factor completed in 0.0058 minutes.    Estimated time of completion: 2020-04-28 10:40:20    [1]
##  [7]   0.0000010  70.0000010  78.6860377 922.3868283   0.1042628 922.3868283
## [13]  51.0000000  51.0000000
## [1] "j = 10.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

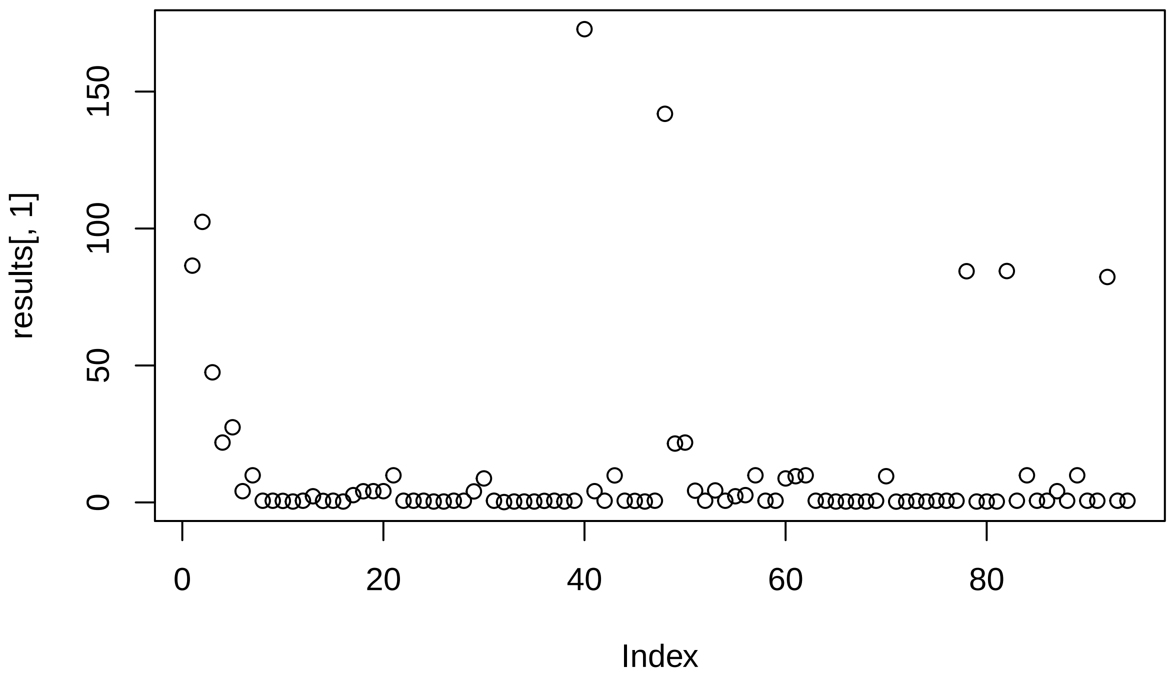




```
##  1 factor completed in 0.00445 minutes.    Estimated time of completion: 2020-04-28 10:40:23    [1]
##  [7]  10.0000010  70.0000010  78.6860377 874.7152324 166.3674876 836.6951841
## [13]  10.0000000  40.0000000
## [1] "j = 20.000001"
```

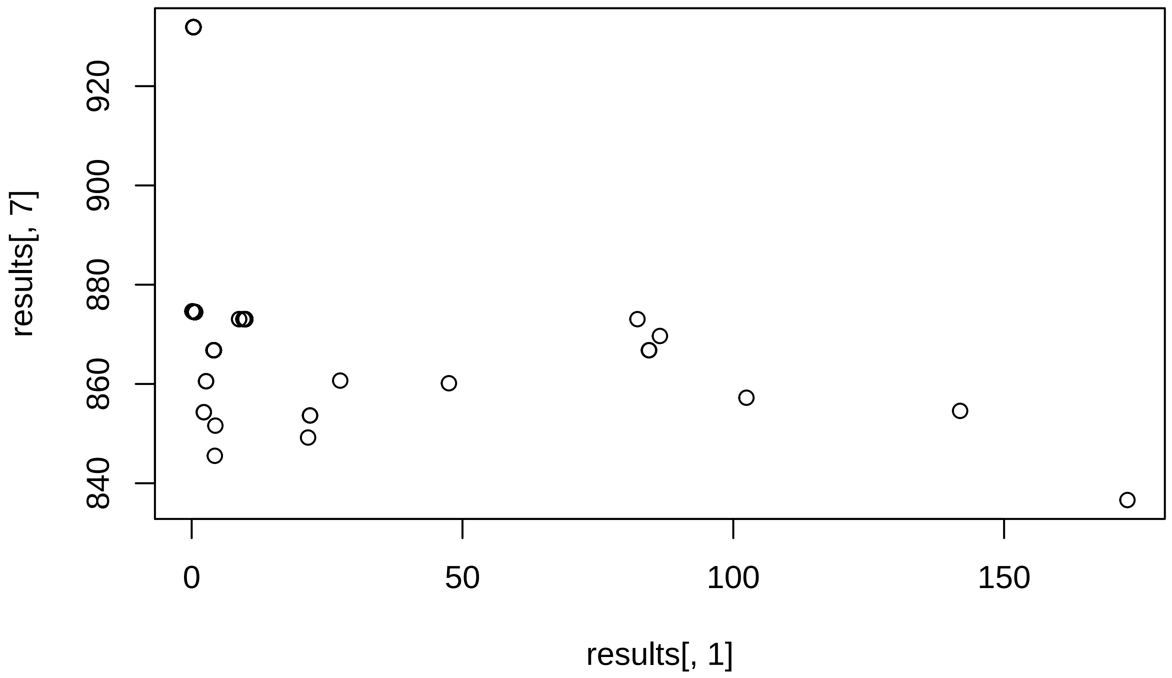
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

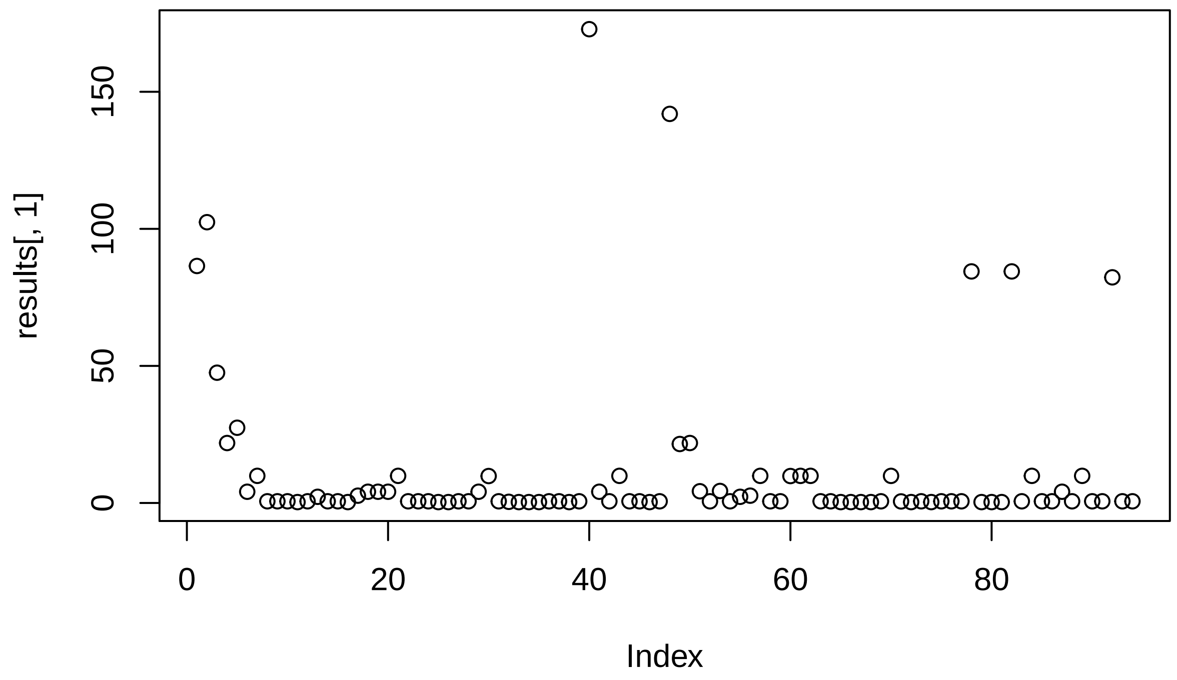




```
##  1 factor completed in 0.00556 minutes.    Estimated time of completion: 2020-04-28 10:40:25    [1]
##  [7]  20.0000010  70.0000010  78.6860377 874.6560759 172.7762783 836.6163412
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

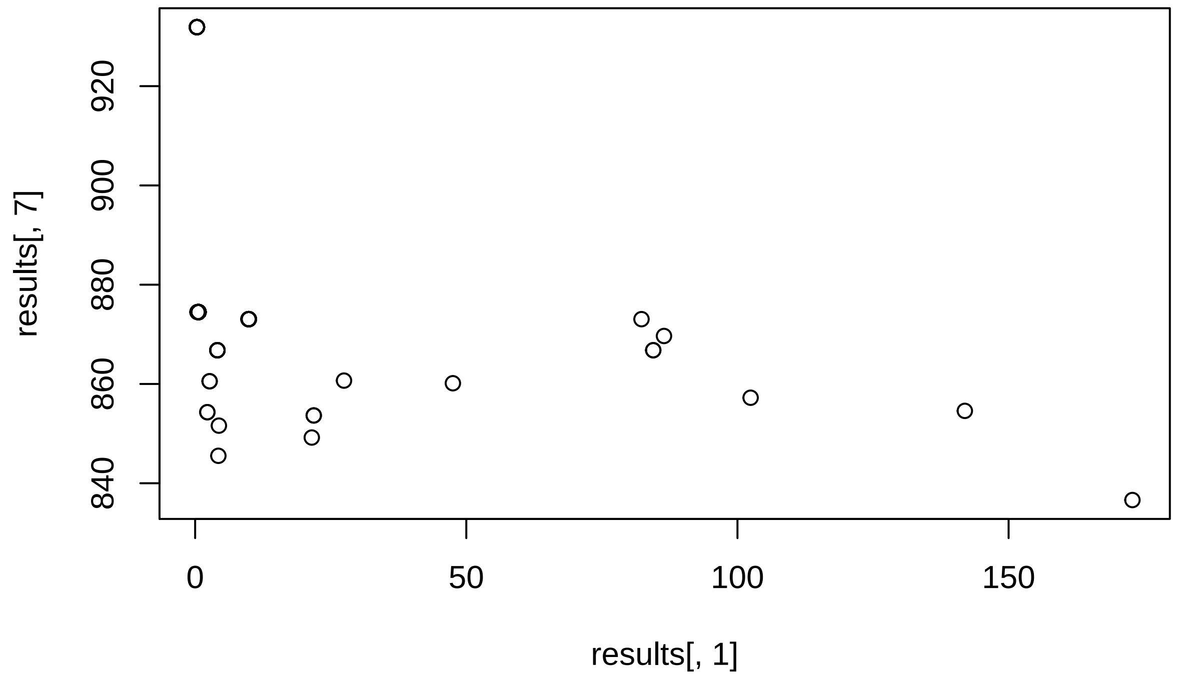
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

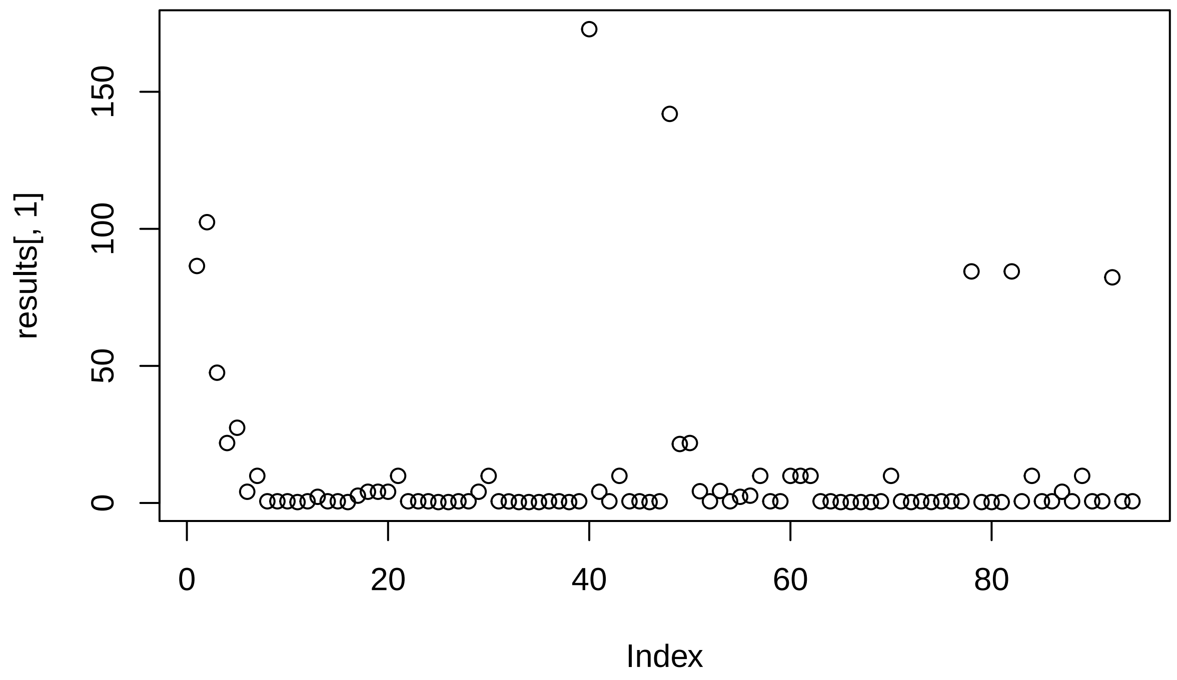




```
##  1 factor completed in 0.00451 minutes.    Estimated time of completion: 2020-04-28 10:40:27    [1]
##  [7]  30.0000010  70.0000010  78.6860377 931.8890324 172.8348926 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

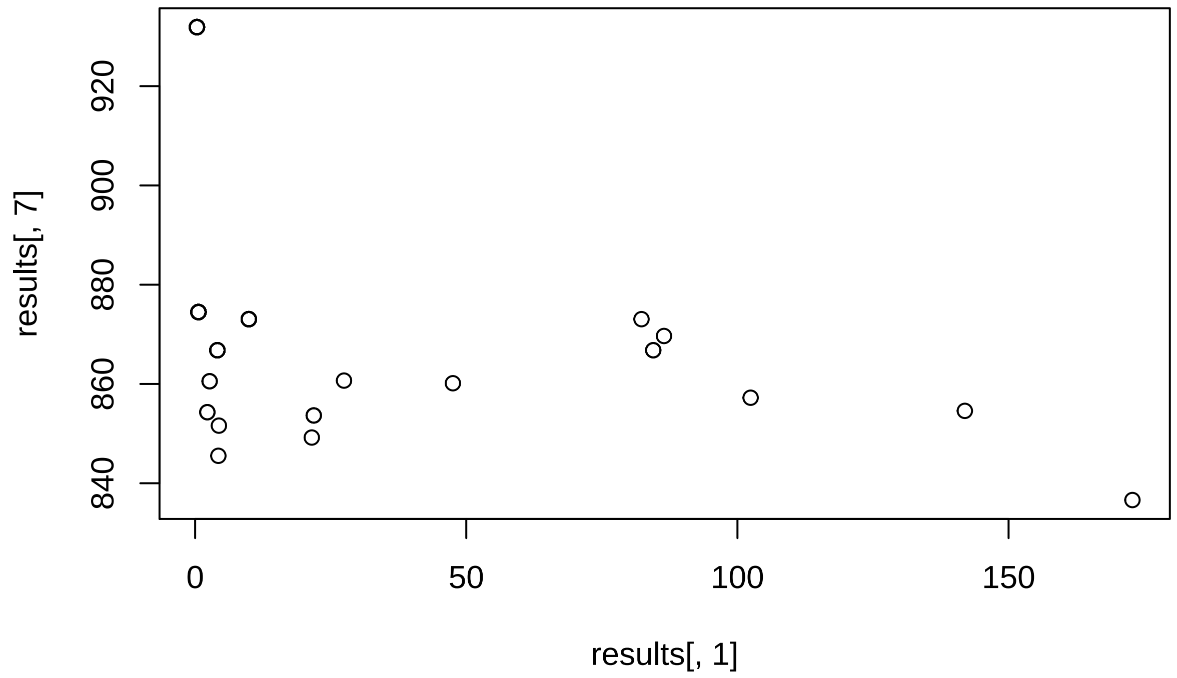
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

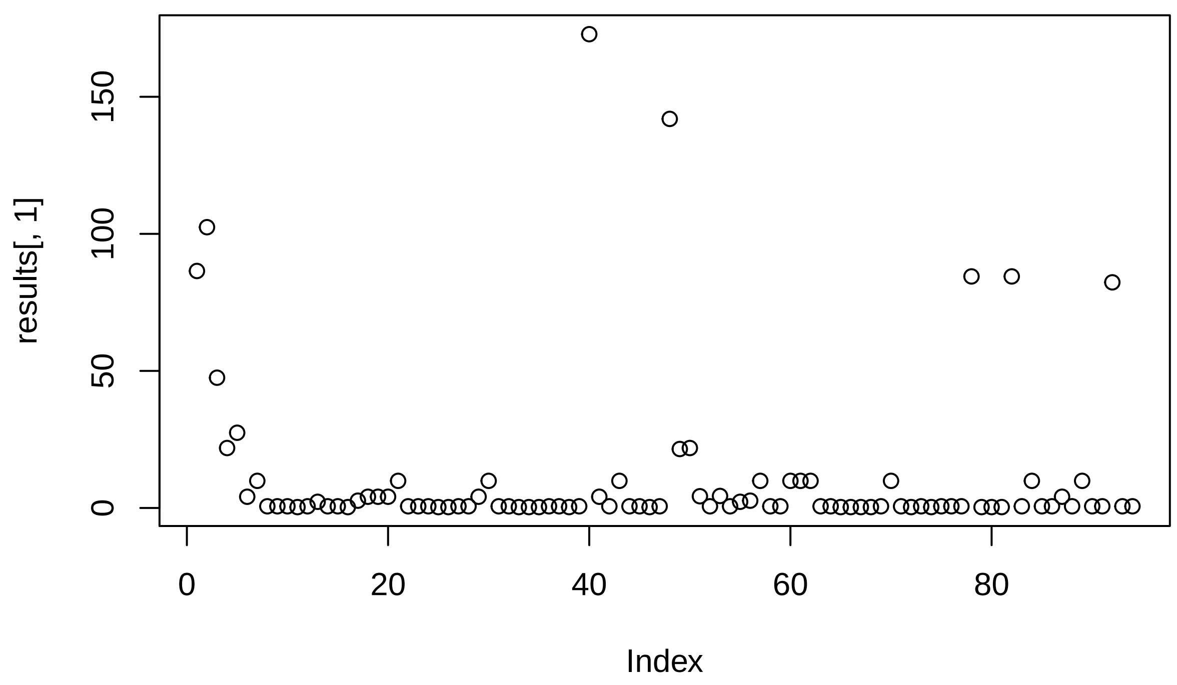




```
##  1 factor completed in 0.00426 minutes.    Estimated time of completion: 2020-04-28 10:40:30    [1]
##  [7]  40.0000010  70.0000010  78.6860377 931.8890324 172.8354141 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

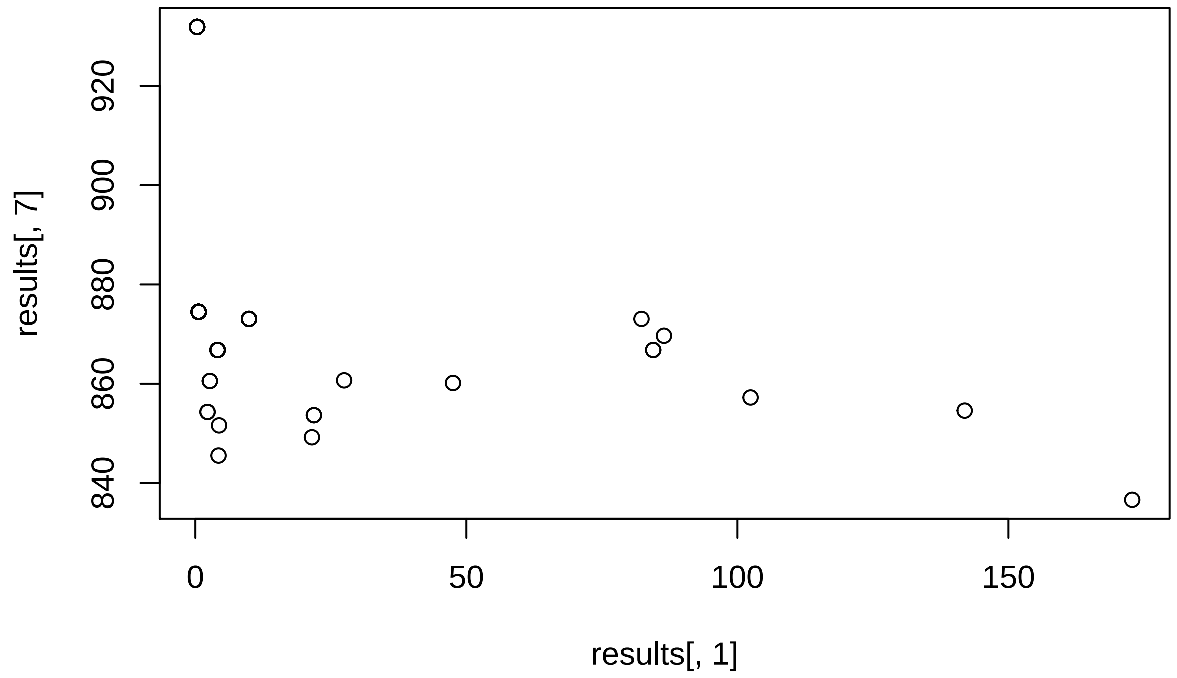
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

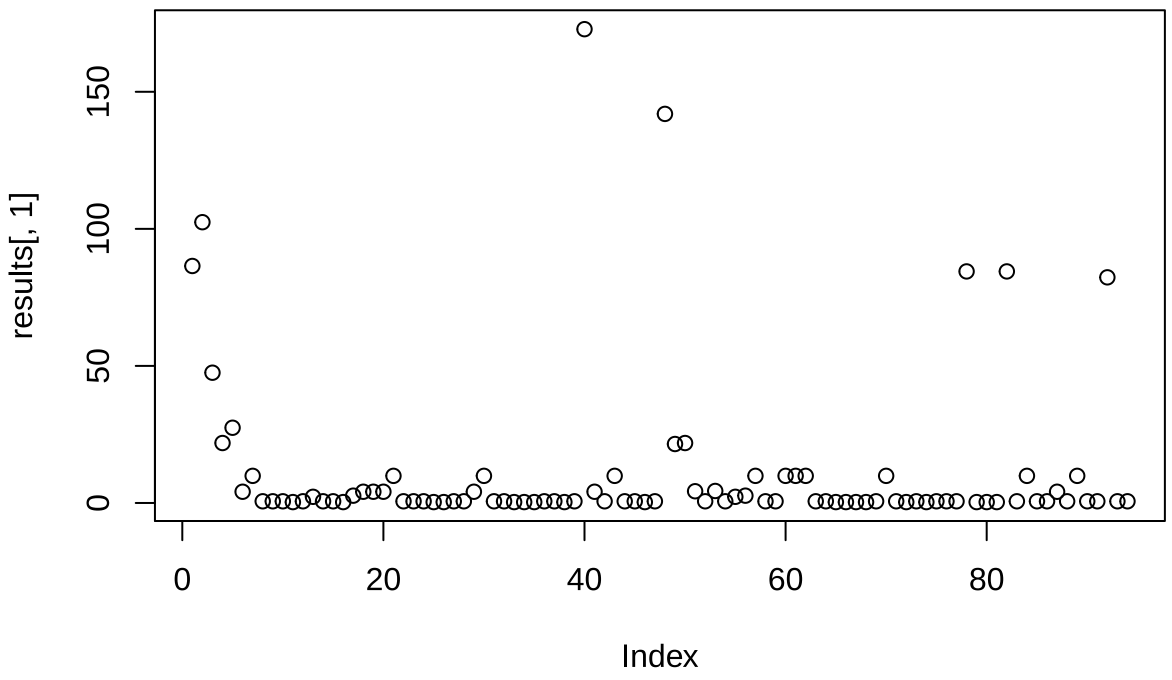




```
##  1 factor completed in 0.00413 minutes.    Estimated time of completion: 2020-04-28 10:40:32    [1]
##  [7]  50.0000010  70.0000010  78.6860377 931.8890324 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

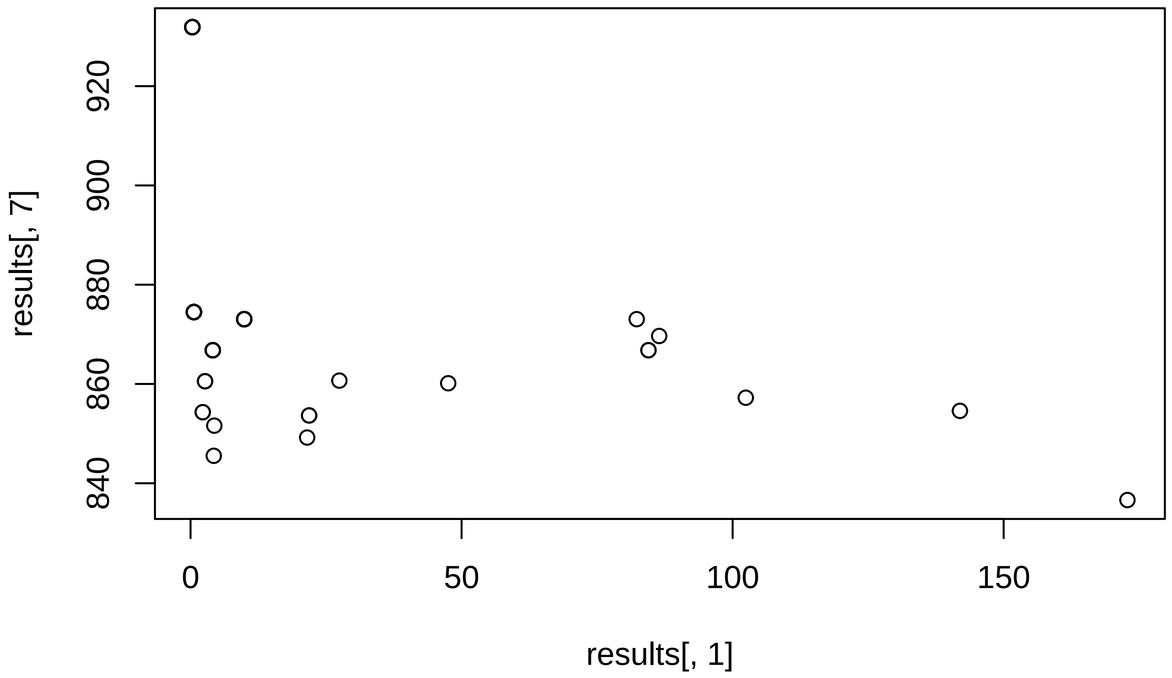
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

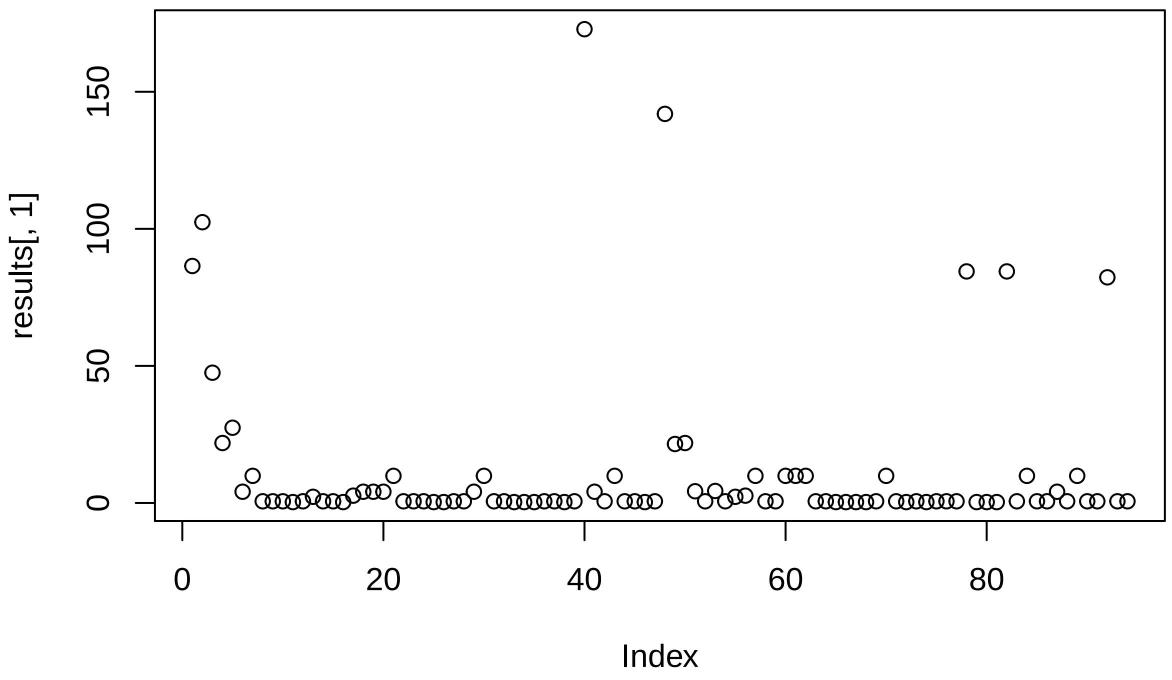




```
##  1 factor completed in 0.00464 minutes.    Estimated time of completion: 2020-04-28 10:40:34    [1]
##  [7]  60.0000010  70.0000010  78.6860377 931.8890324 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

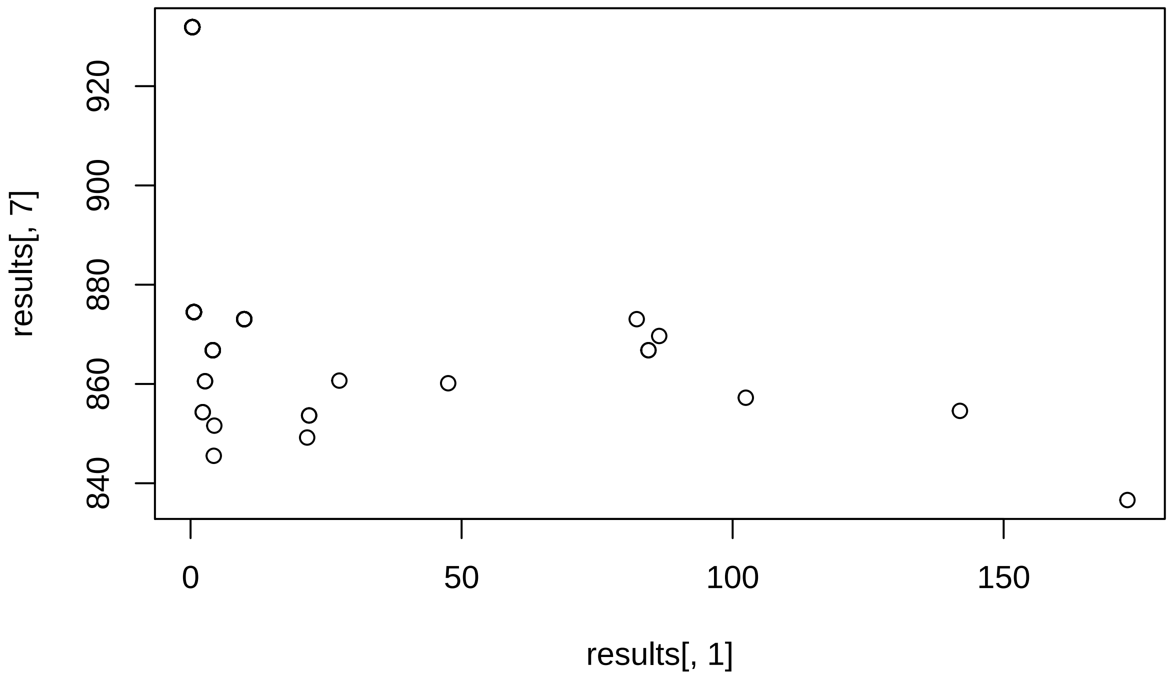
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

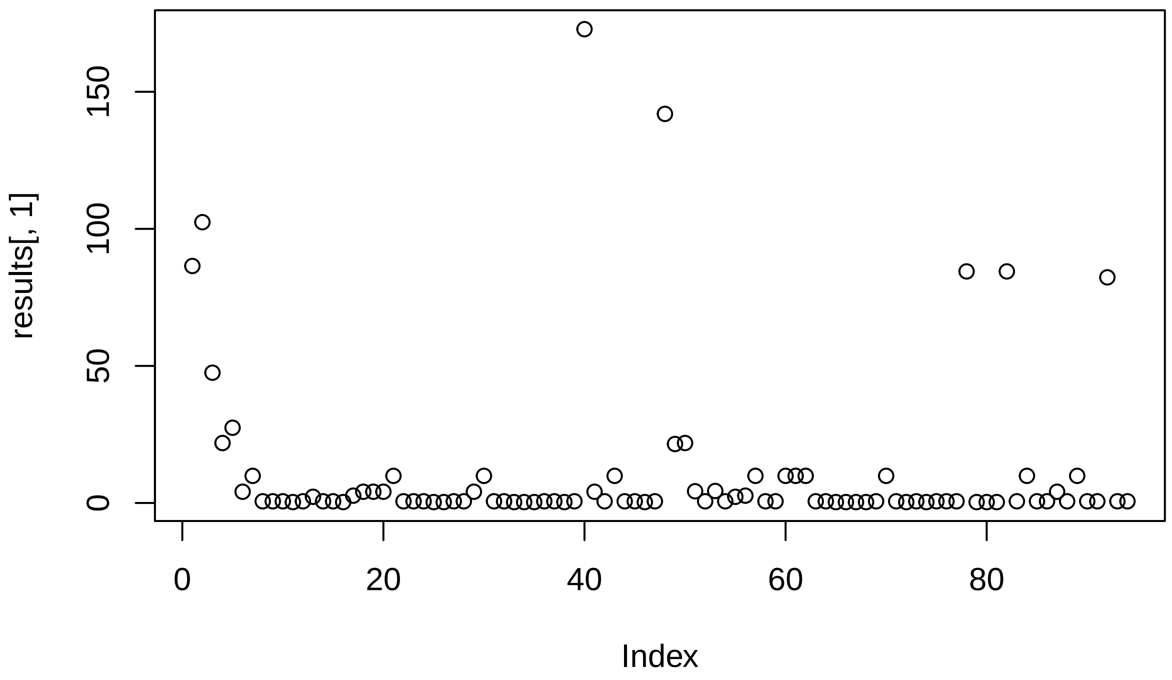




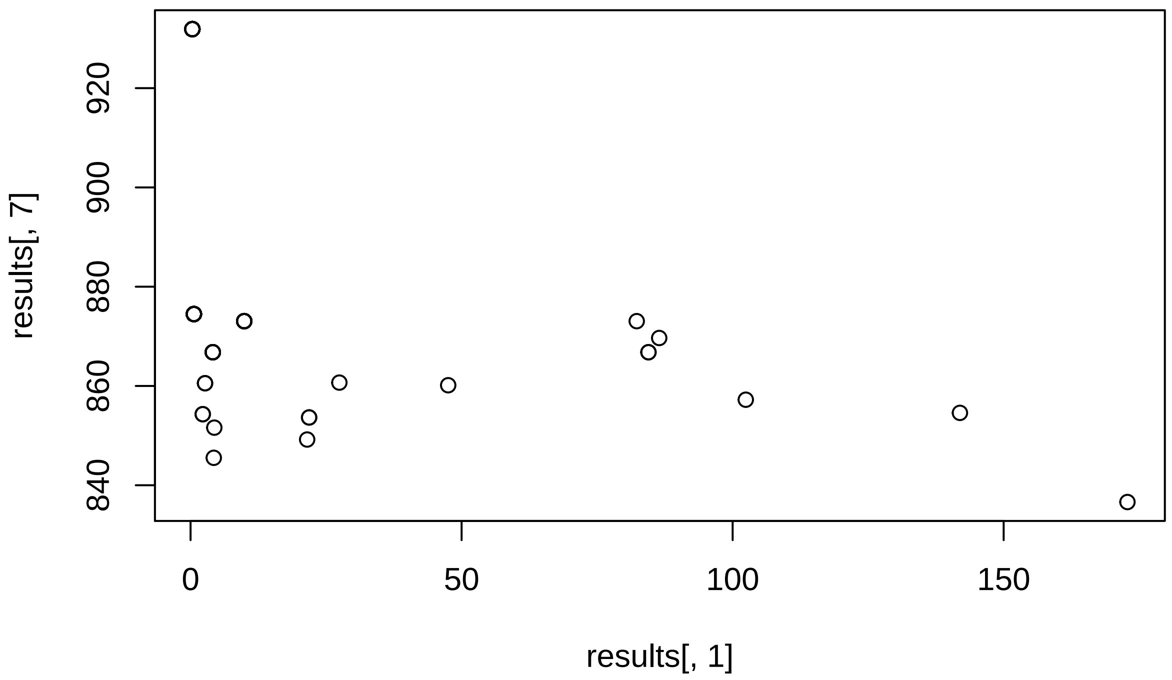
```
##  1 factor completed in 0.00573 minutes.    Estimated time of completion: 2020-04-28 10:40:37    [1]
##  [7]  70.0000010  70.0000010  78.6860377 931.8890324 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

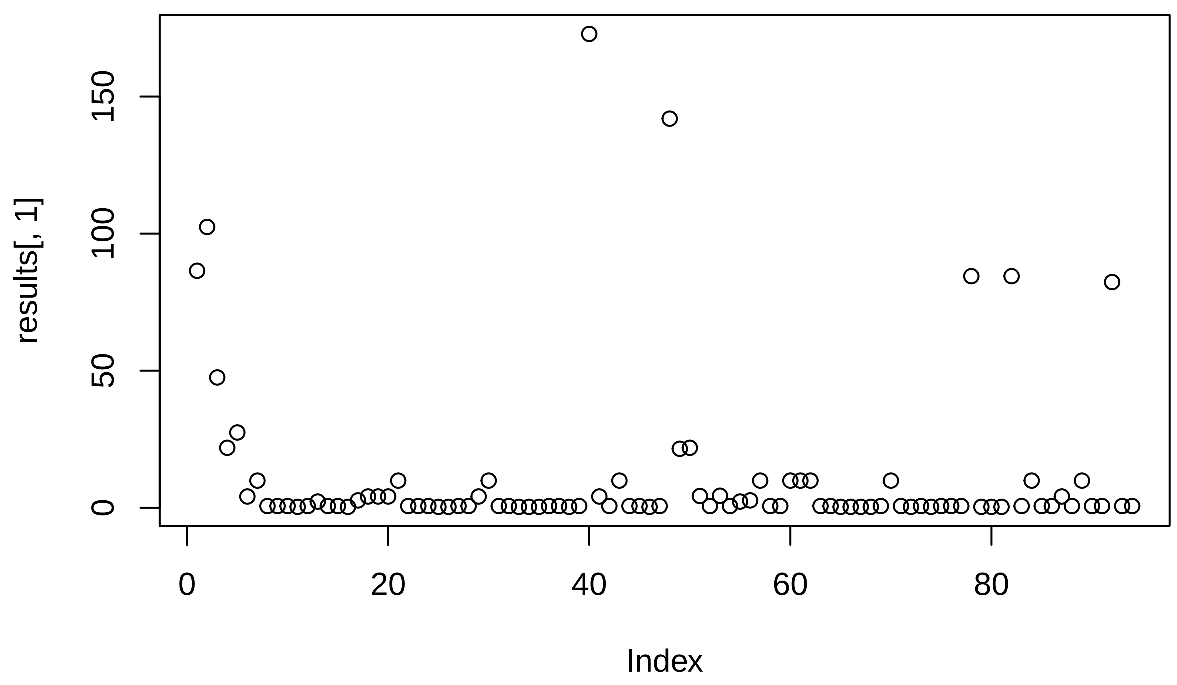
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





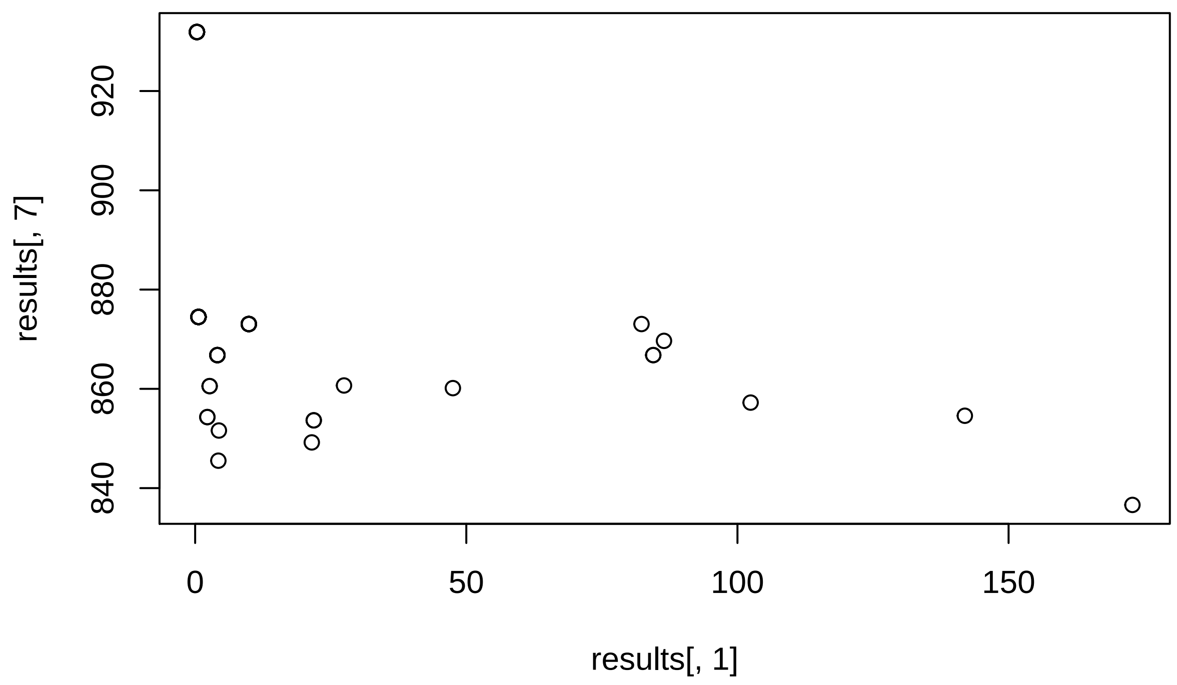
```
##  1 factor completed in 0.00561 minutes.    Estimated time of completion: 2020-04-28 10:40:40    [1]
##  [7]  80.0000010  70.0000010  78.6860377 931.8890324 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

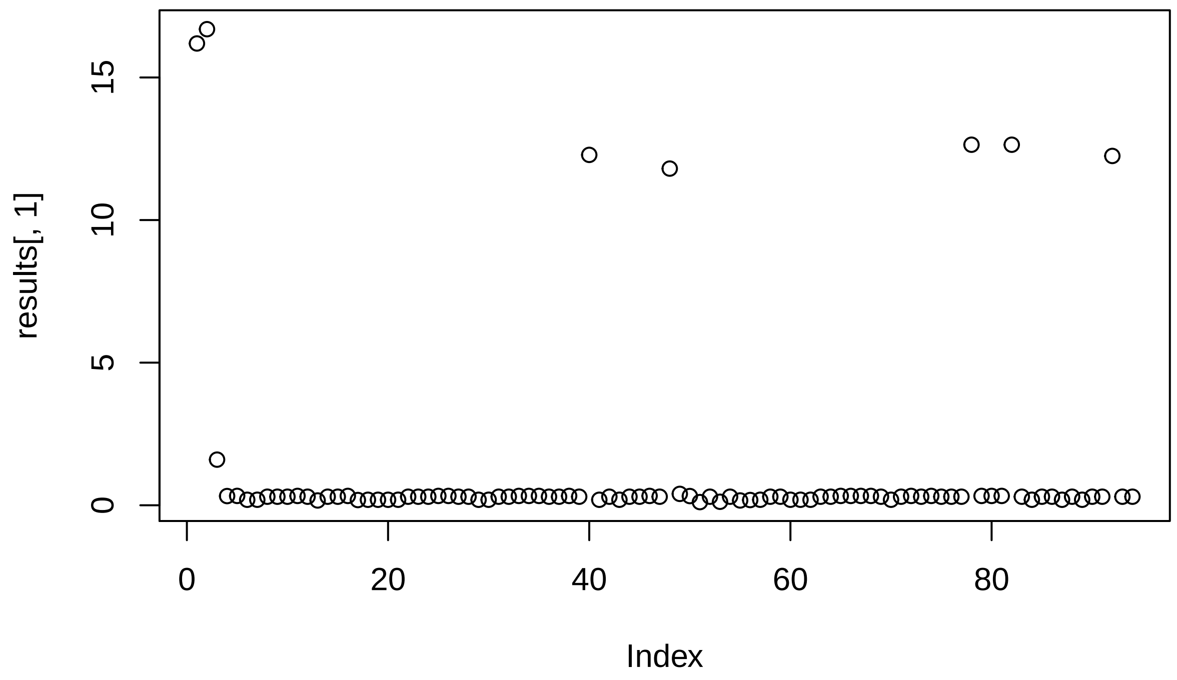




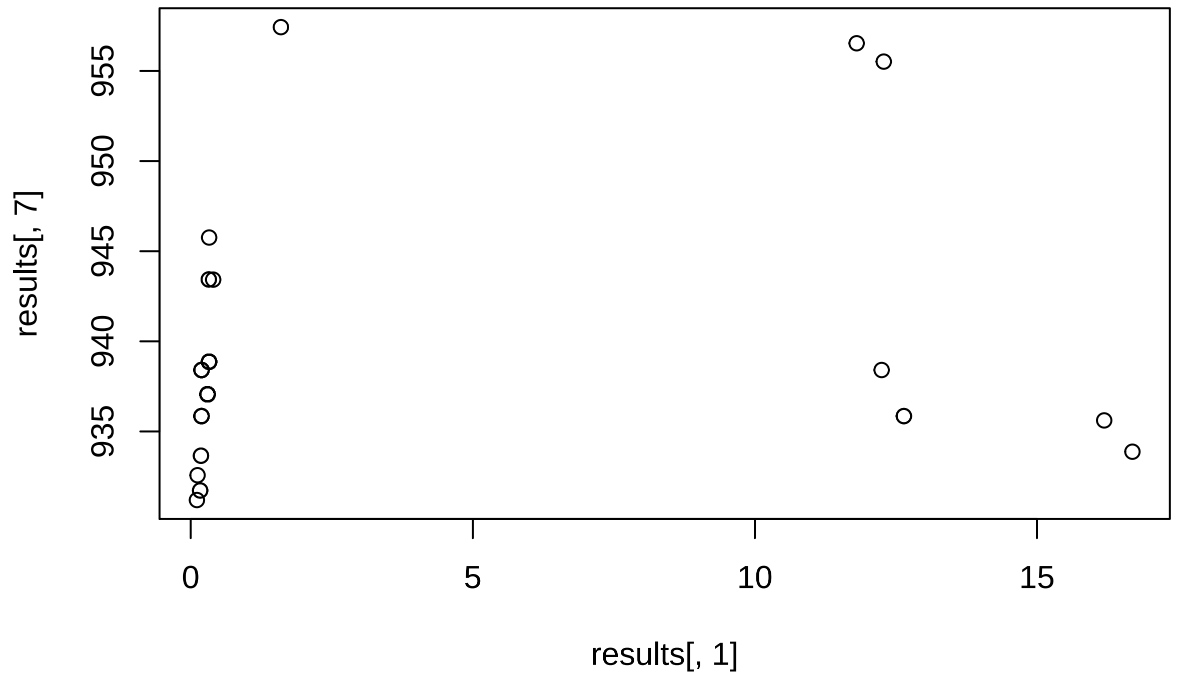
```
##  1 factor completed in 0.00533 minutes.    Estimated time of completion: 2020-04-28 10:40:43    [1]
##  [7]  90.0000010  70.0000010  78.6860377 931.8890324 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 80.000001"
## [1] "j = 1e-06"
```

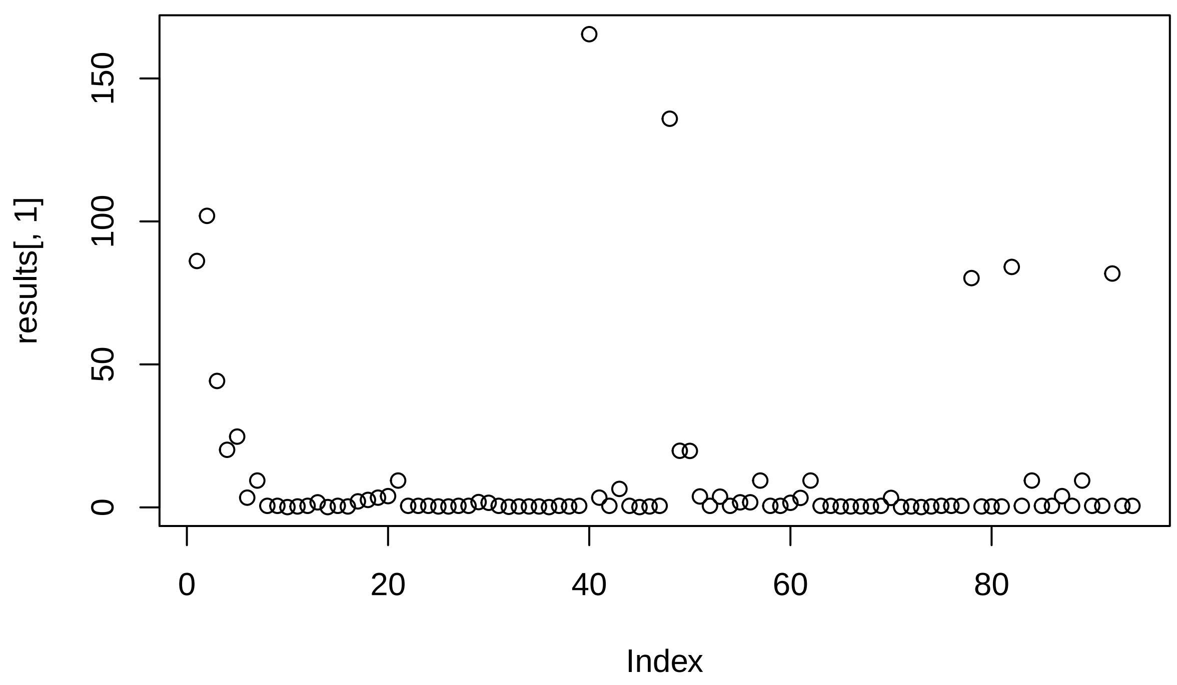
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





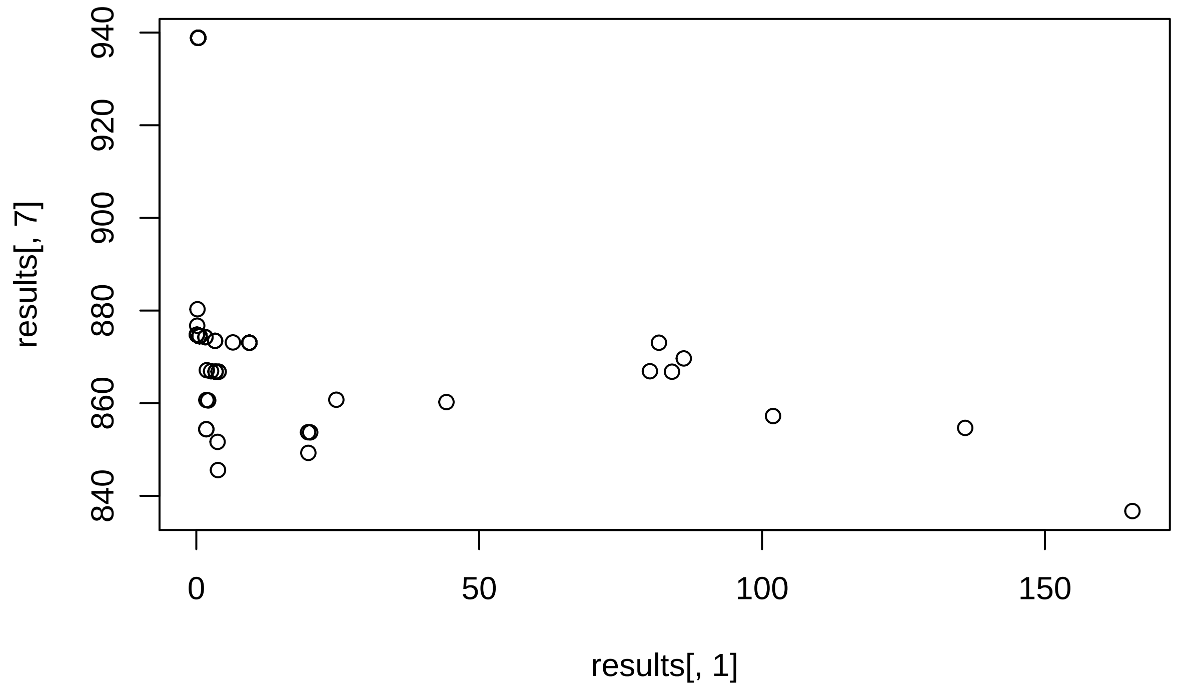
```
##  1 factor completed in 0.00453 minutes.    Estimated time of completion: 2020-04-28 10:40:45    [1]
##  [7]   0.0000010  80.0000010  78.6860377 931.1961875   0.1111507 931.1961875
## [13]  51.0000000  51.0000000
## [1] "j = 10.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

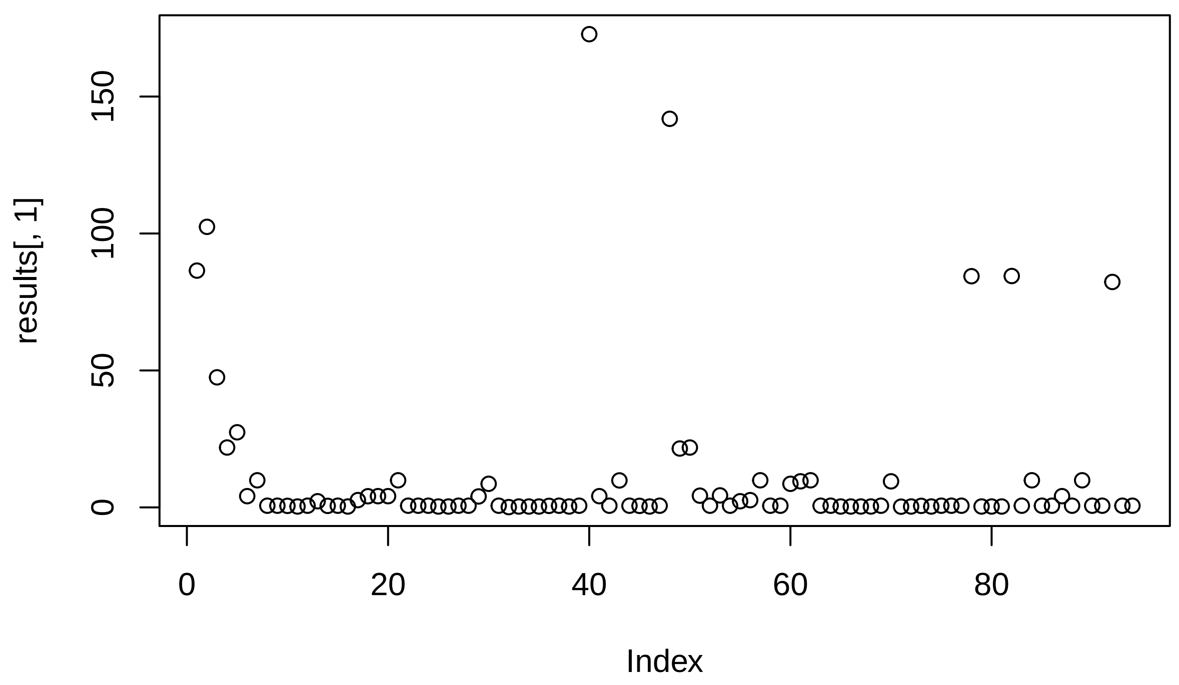




```
##  1 factor completed in 0.00417 minutes.    Estimated time of completion: 2020-04-28 10:40:48    [1]
##  [7]  10.0000010  80.0000010  78.6860377 874.7806478 165.4724182 836.7187907
## [13]  10.0000000  40.0000000
## [1] "j = 20.000001"
```

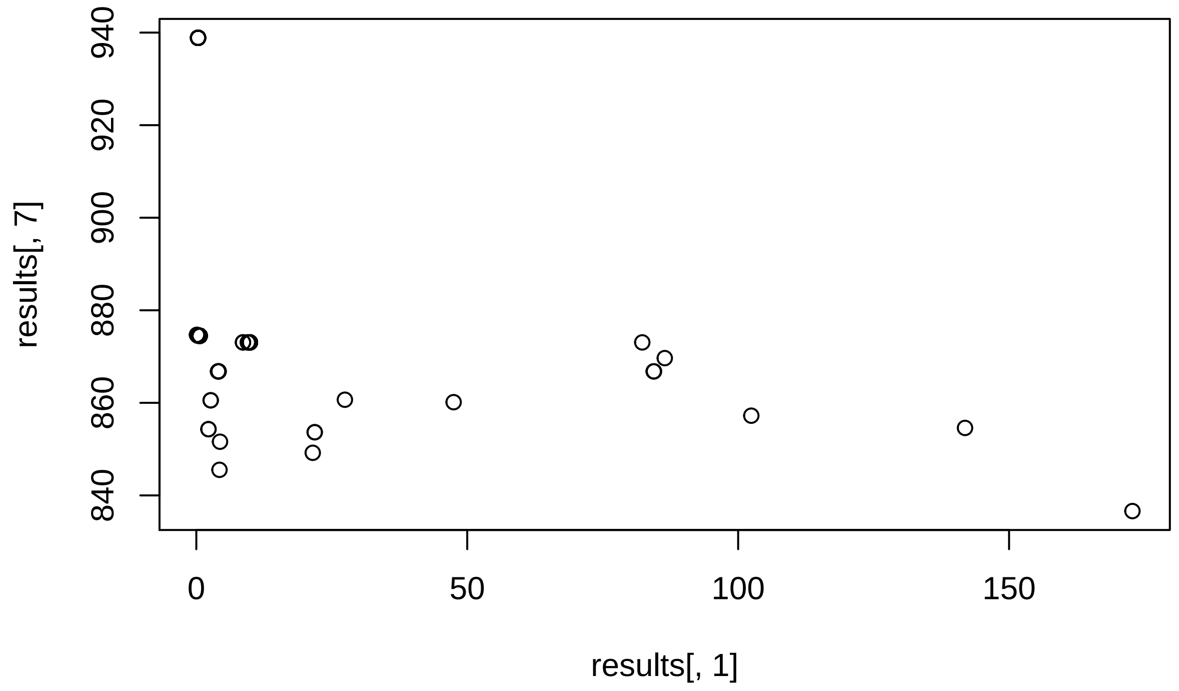
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

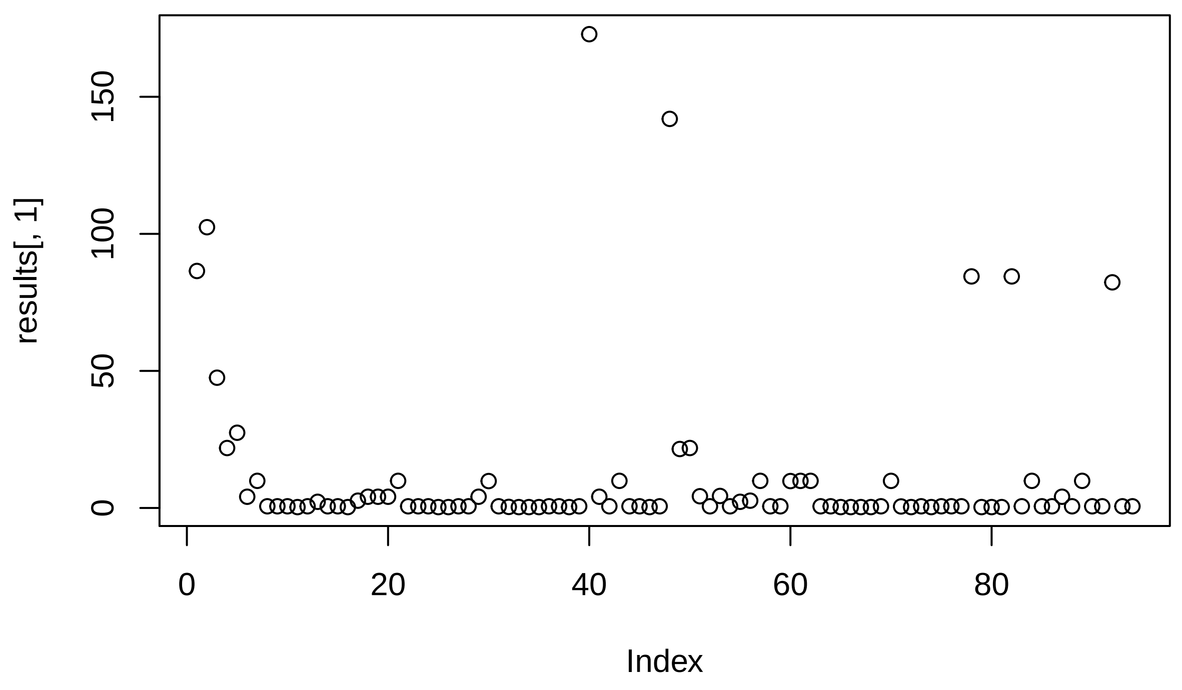




```
##  1 factor completed in 0.00404 minutes.    Estimated time of completion: 2020-04-28 10:40:50    [1]
##  [7]  20.0000010  80.0000010  78.6860377 874.7048610 172.7678320 836.6163432
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

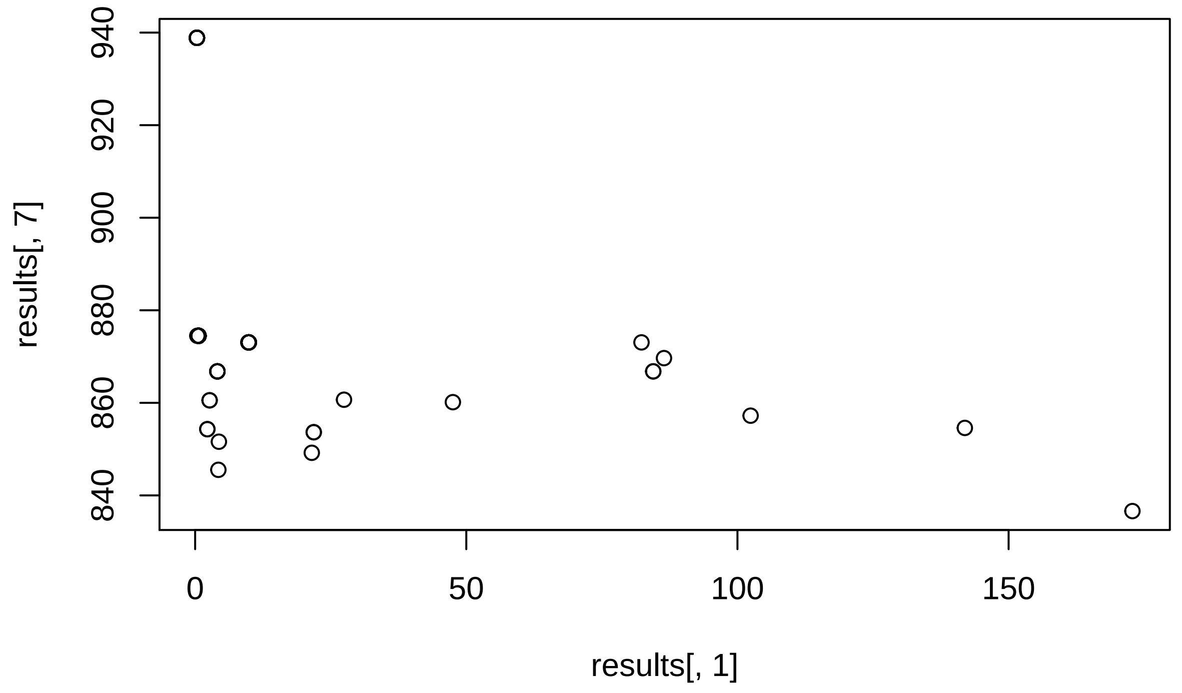
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

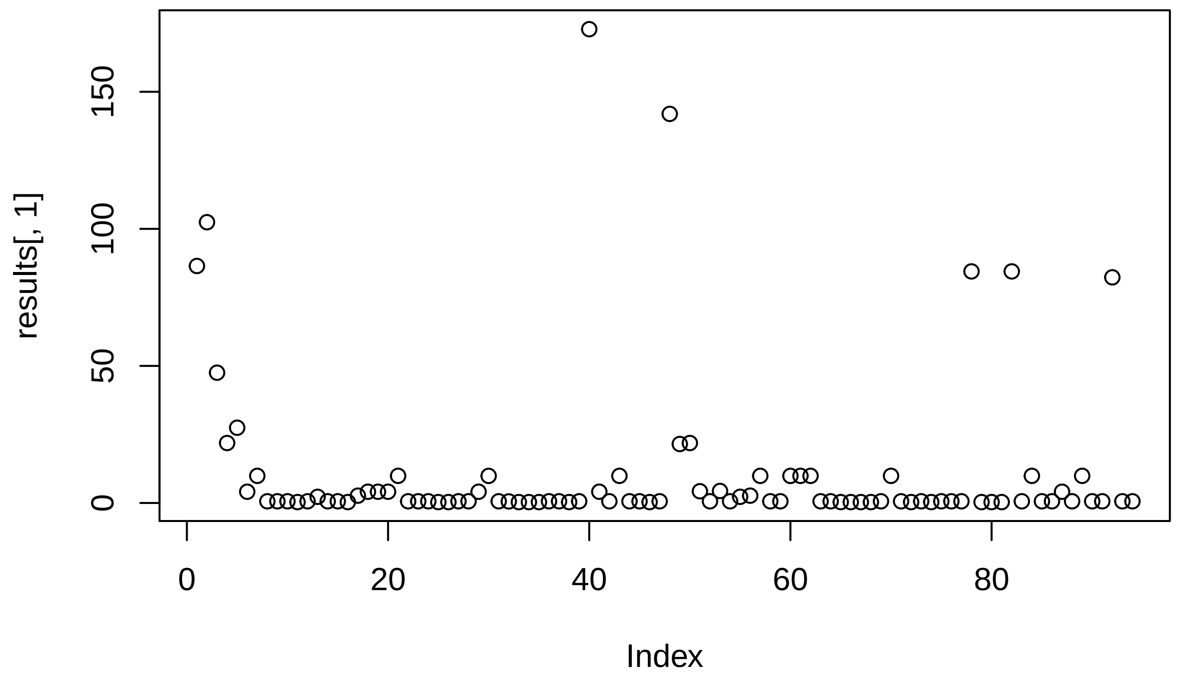




```
##  1 factor completed in 0.00395 minutes.    Estimated time of completion: 2020-04-28 10:40:52    [1]
##  [7]  30.0000010  80.0000010  78.6860377 938.8679479 172.8348174 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"
```

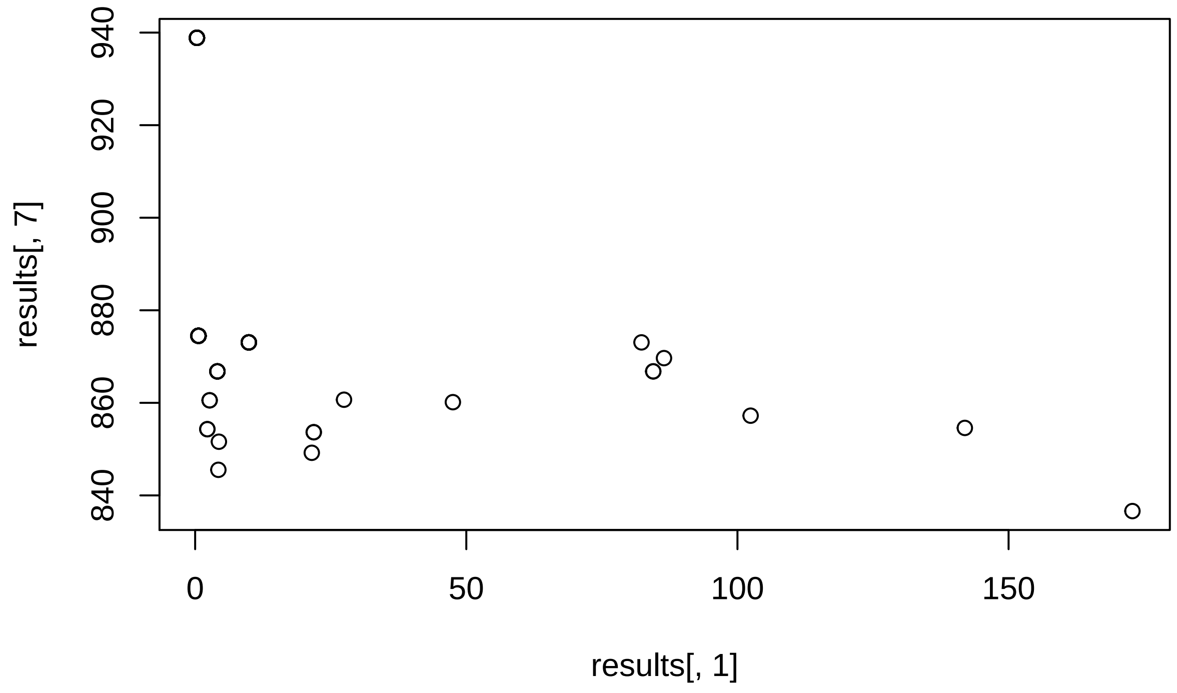
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

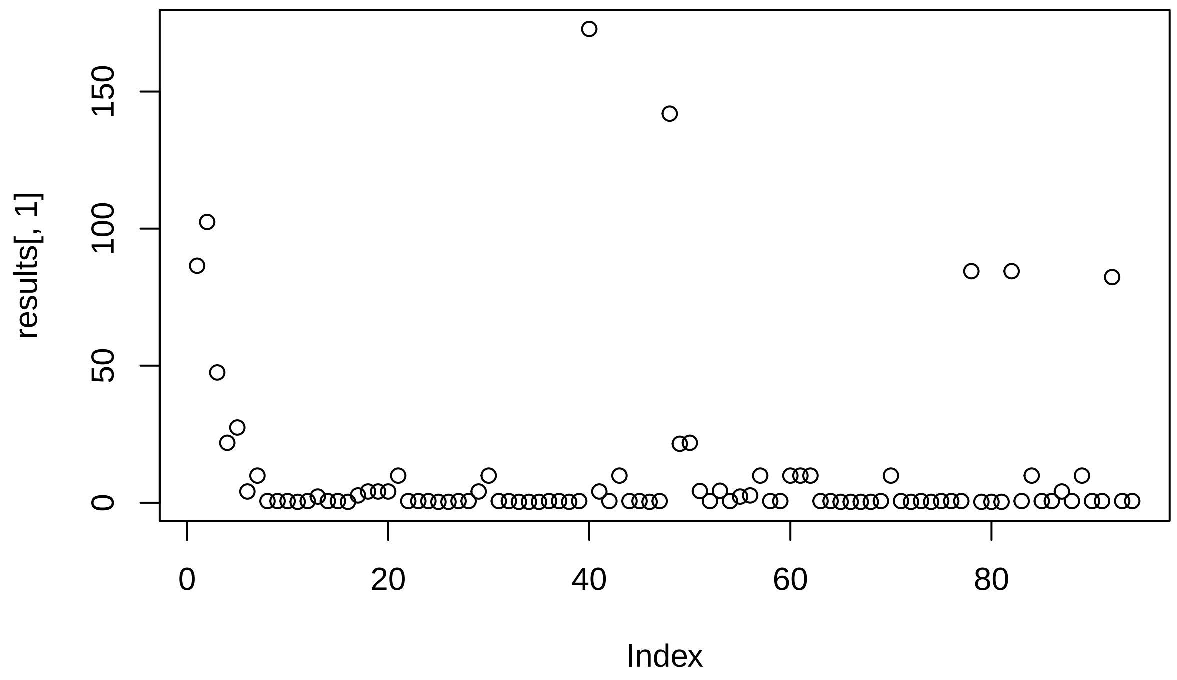




```
##  1 factor completed in 0.00399 minutes.    Estimated time of completion: 2020-04-28 10:40:54    [1]
##  [7]  40.0000010  80.0000010  78.6860377 938.8679479 172.8354134 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

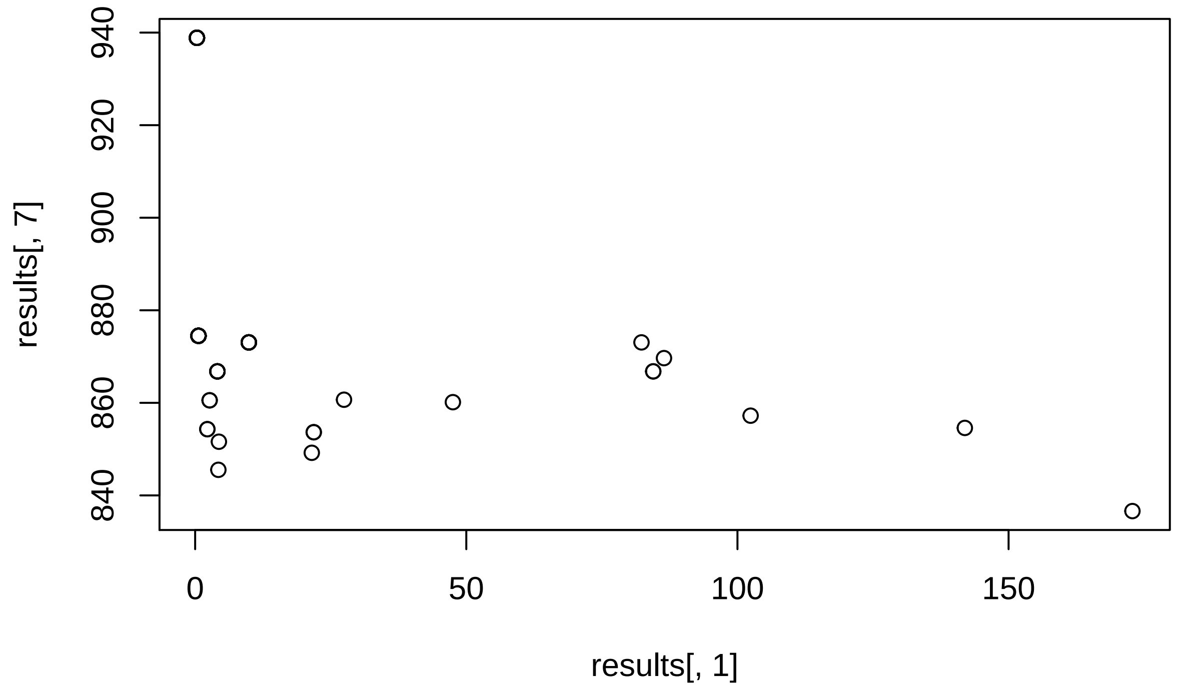
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

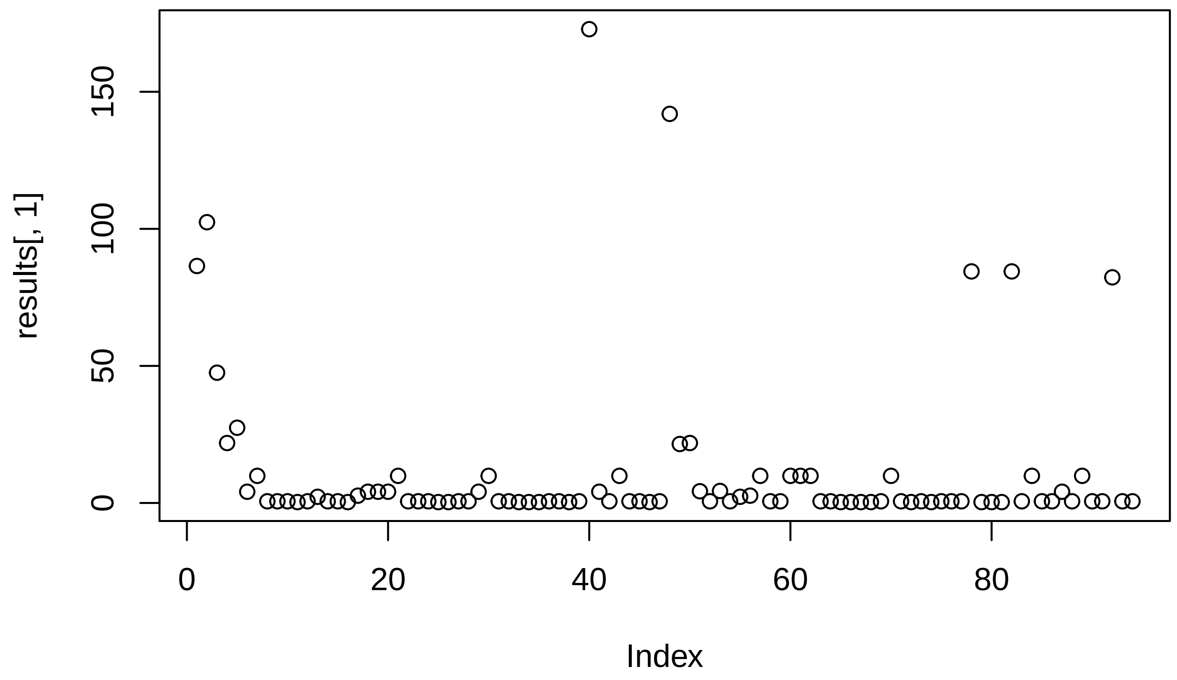




```
##  1 factor completed in 0.00415 minutes.    Estimated time of completion: 2020-04-28 10:40:56    [1]
##  [7]  50.0000010  80.0000010  78.6860377 938.8679479 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

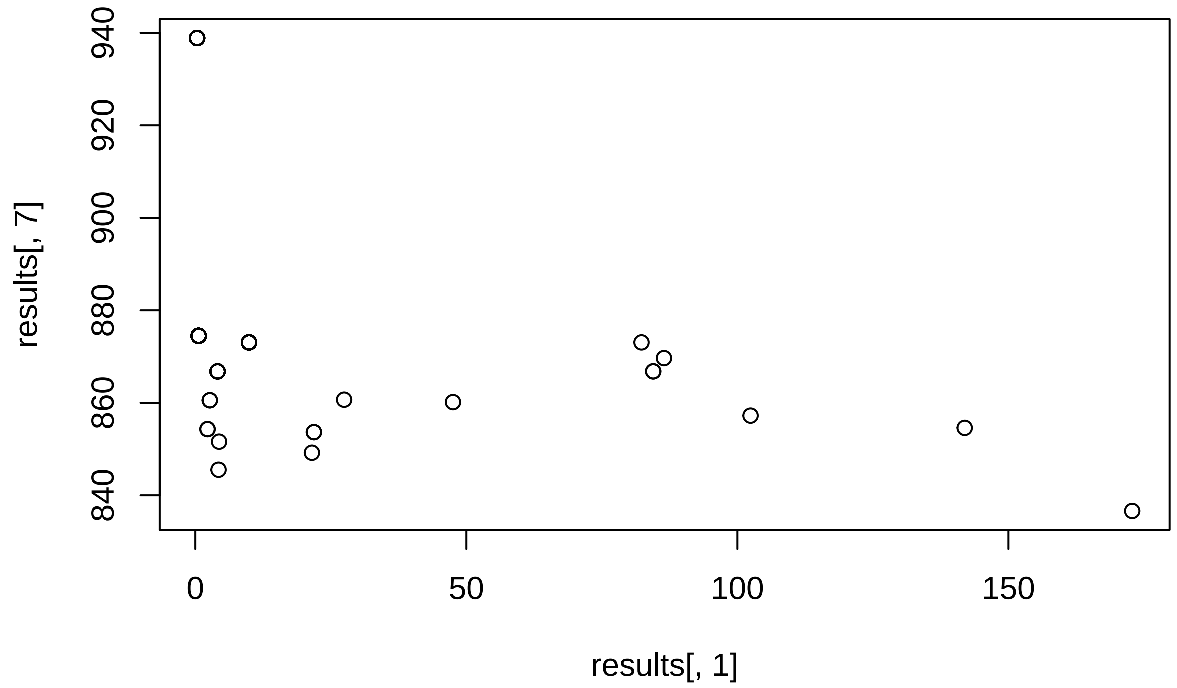
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

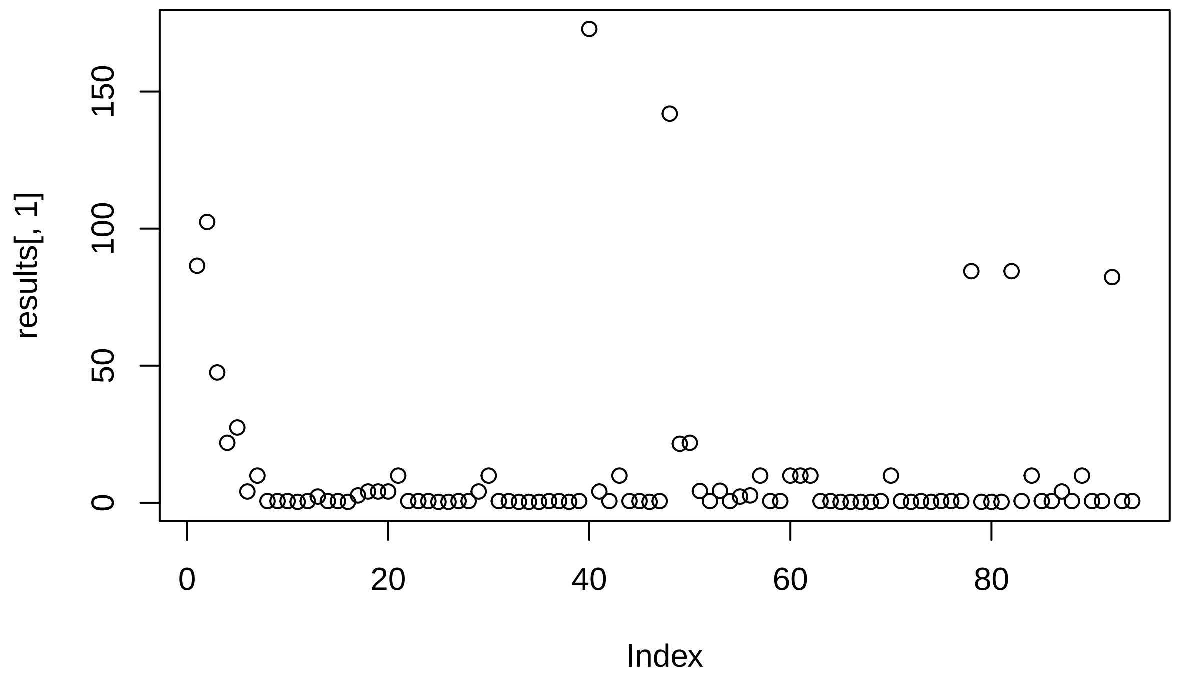




```
##  1 factor completed in 0.0043 minutes.    Estimated time of completion: 2020-04-28 10:40:58    [1]
##  [7]  60.0000010  80.0000010  78.6860377 938.8679479 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

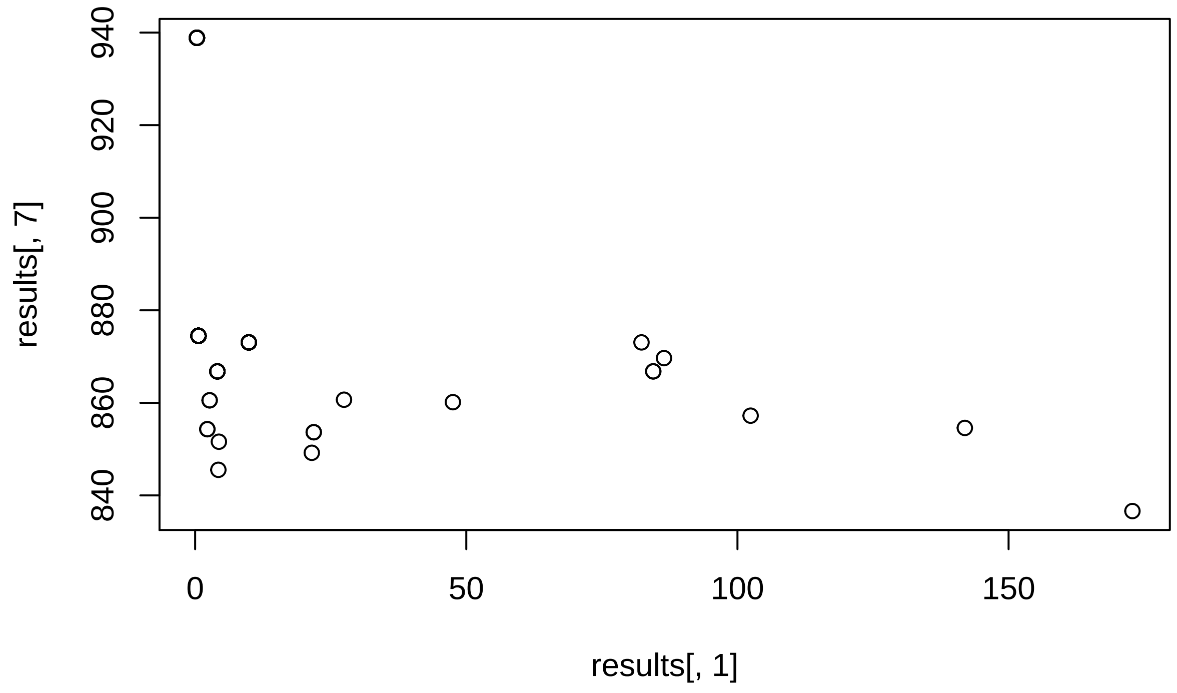
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

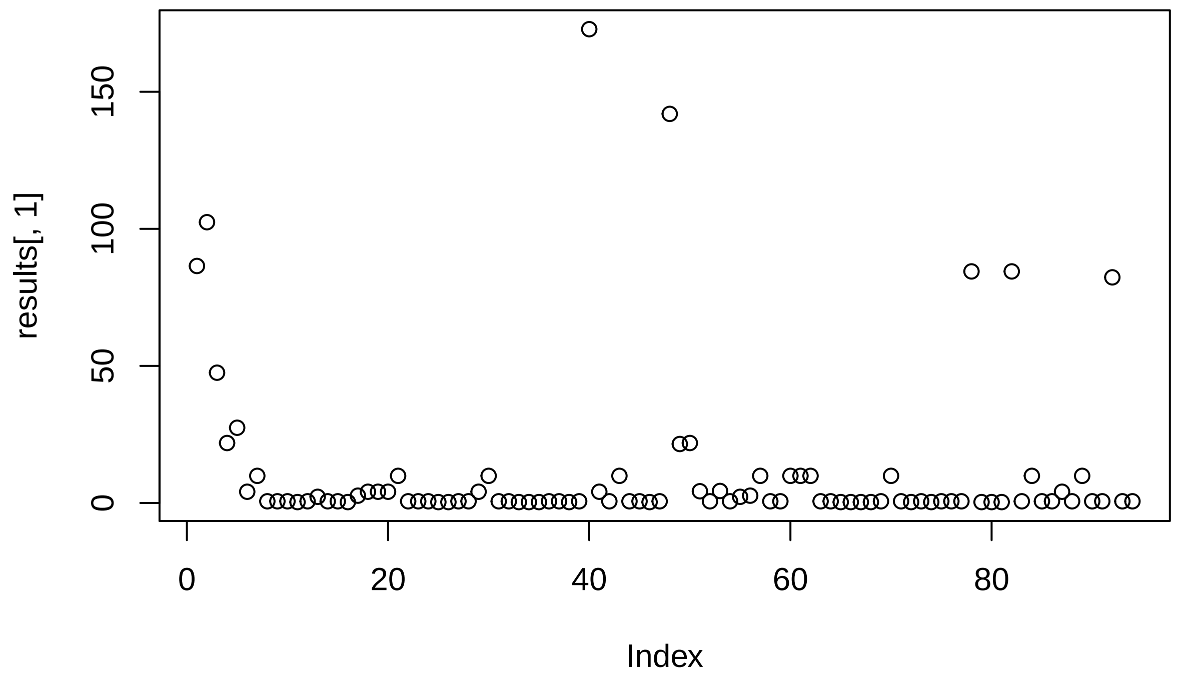




```
##  1 factor completed in 0.00471 minutes.    Estimated time of completion: 2020-04-28 10:41:00    [1]
##  [7]  70.0000010  80.0000010  78.6860377 938.8679479 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

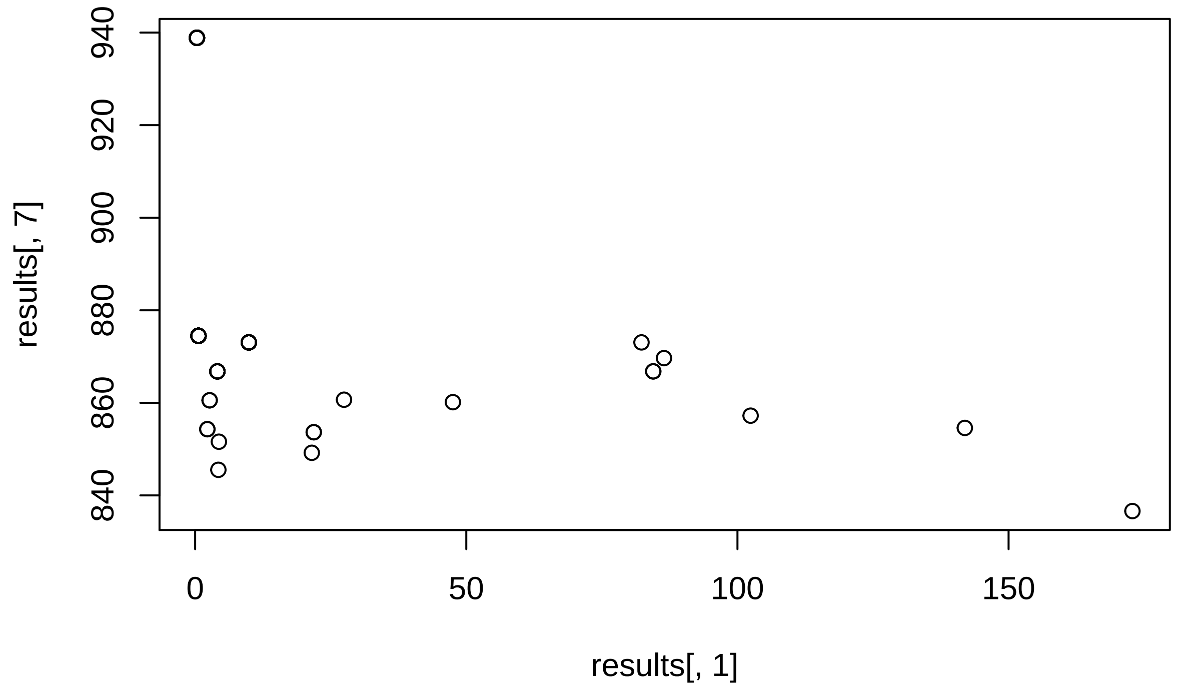
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

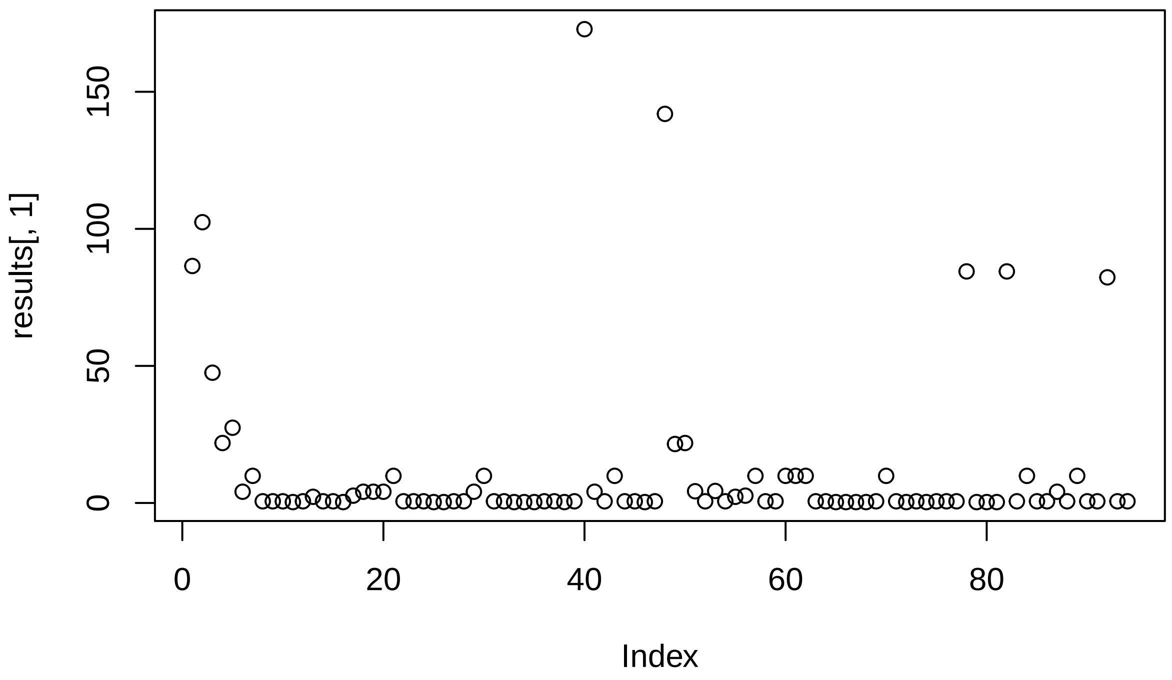




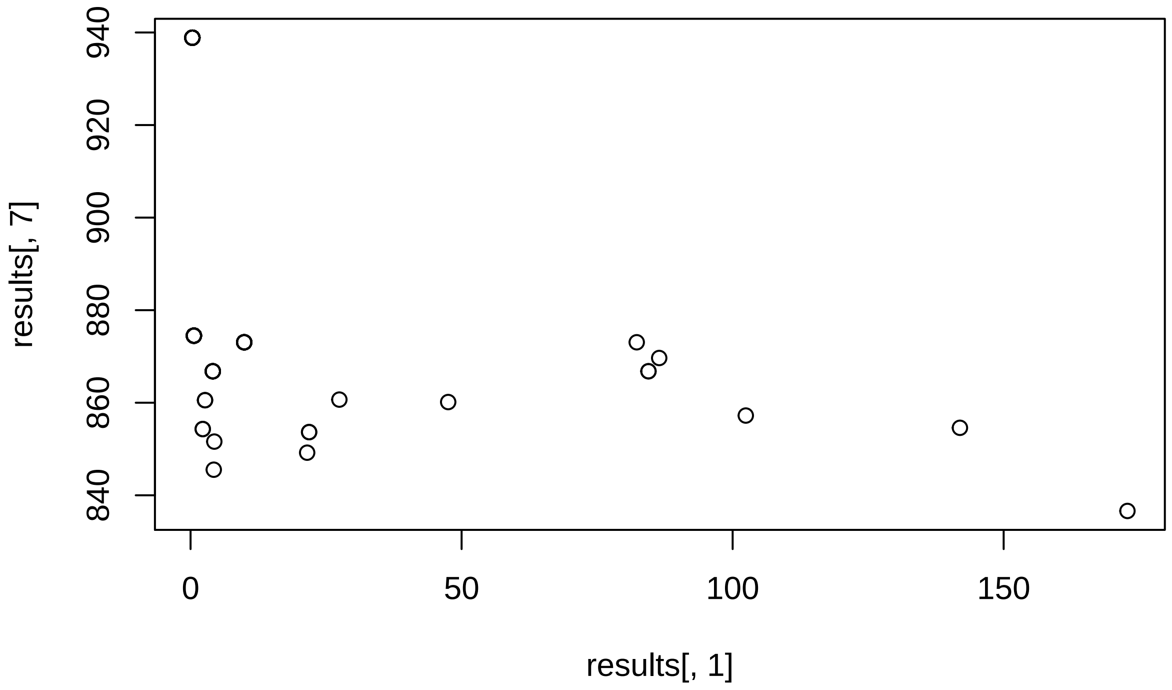
```
##  1 factor completed in 0.00482 minutes.    Estimated time of completion: 2020-04-28 10:41:03    [1]
##  [7]  80.0000010  80.0000010  78.6860377 938.8679479 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

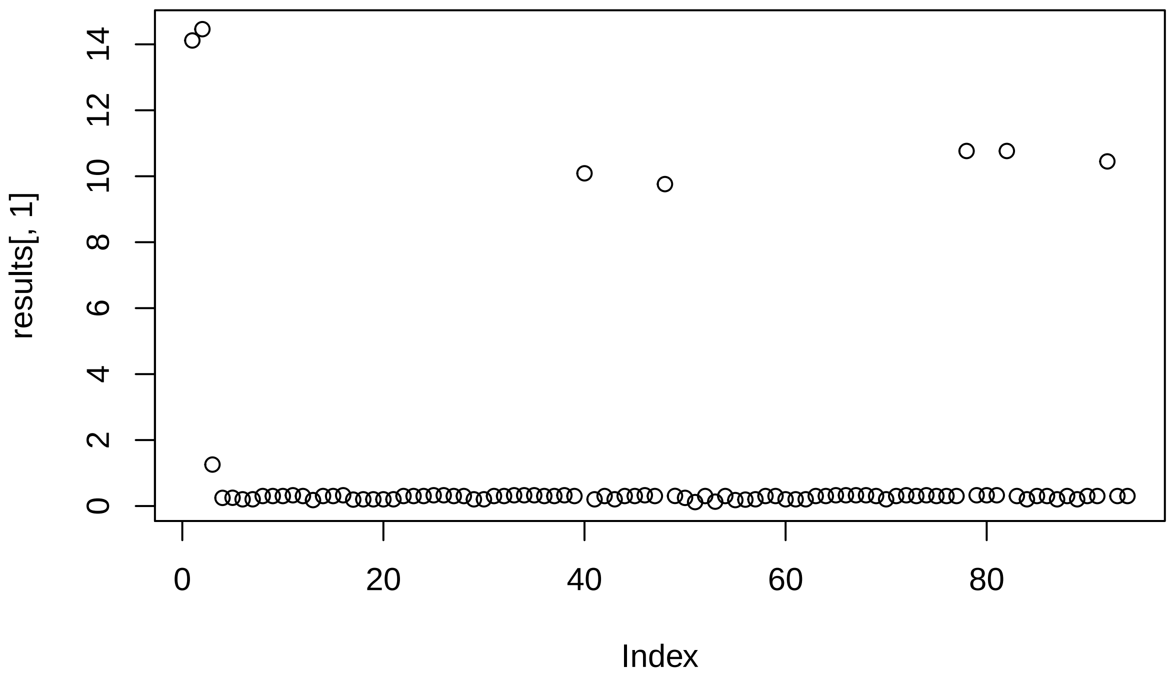
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





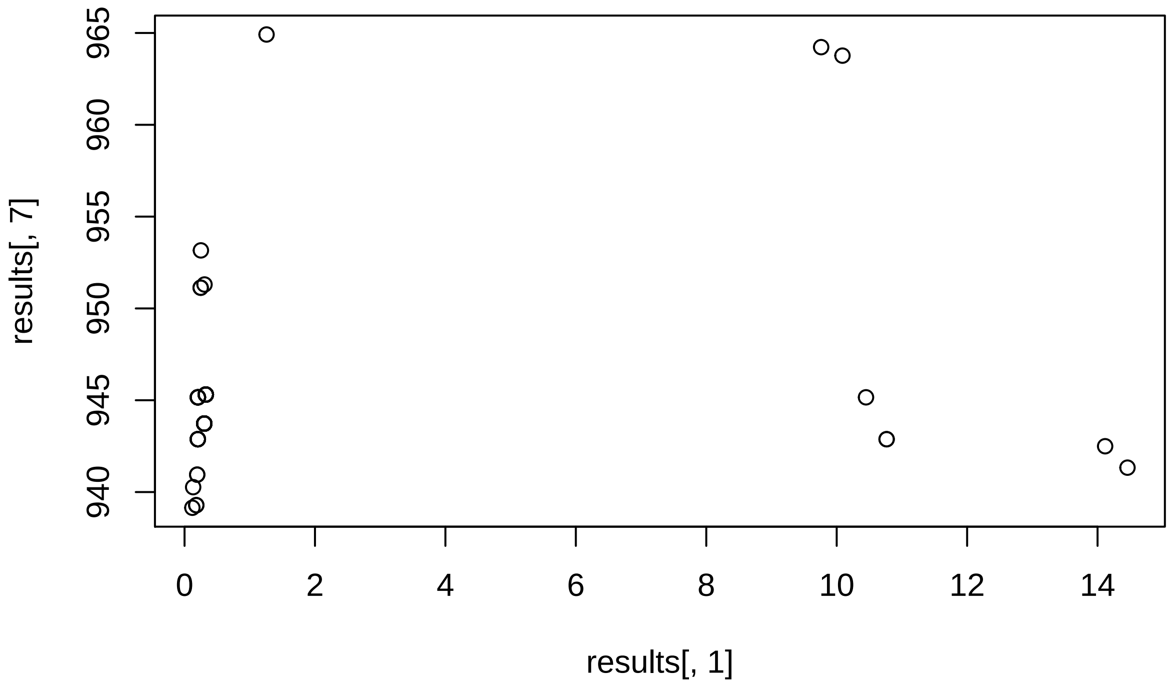
```
##  1 factor completed in 0.0051 minutes.    Estimated time of completion: 2020-04-28 10:41:05    [1]
##  [7]  90.0000010  80.0000010  78.6860377 938.8679479 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "X = 90.000001"
## [1] "j = 1e-06"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

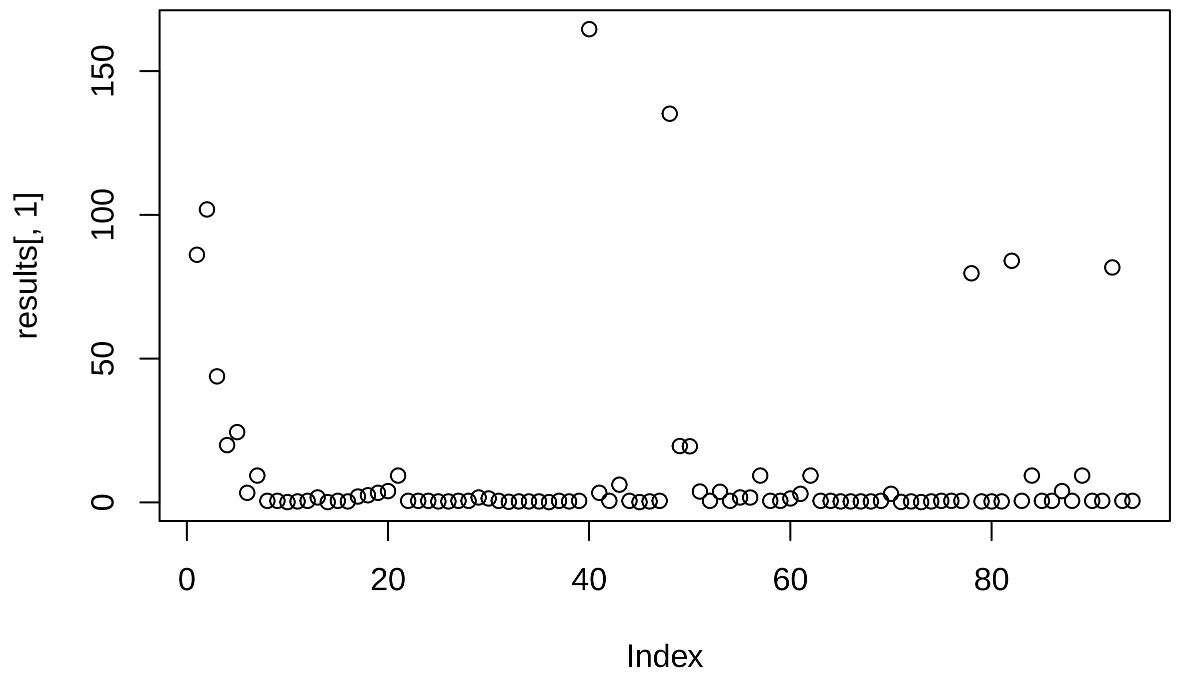




```
##  1 factor completed in 0.00955 minutes.    Estimated time of completion: 2020-04-28 10:41:08    [1]
##  [7]   0.0000010  90.0000010  78.6860377 939.1458906   0.1196251 939.1458906
## [13]  51.0000000  51.0000000
## [1] "j = 10.000001"
```

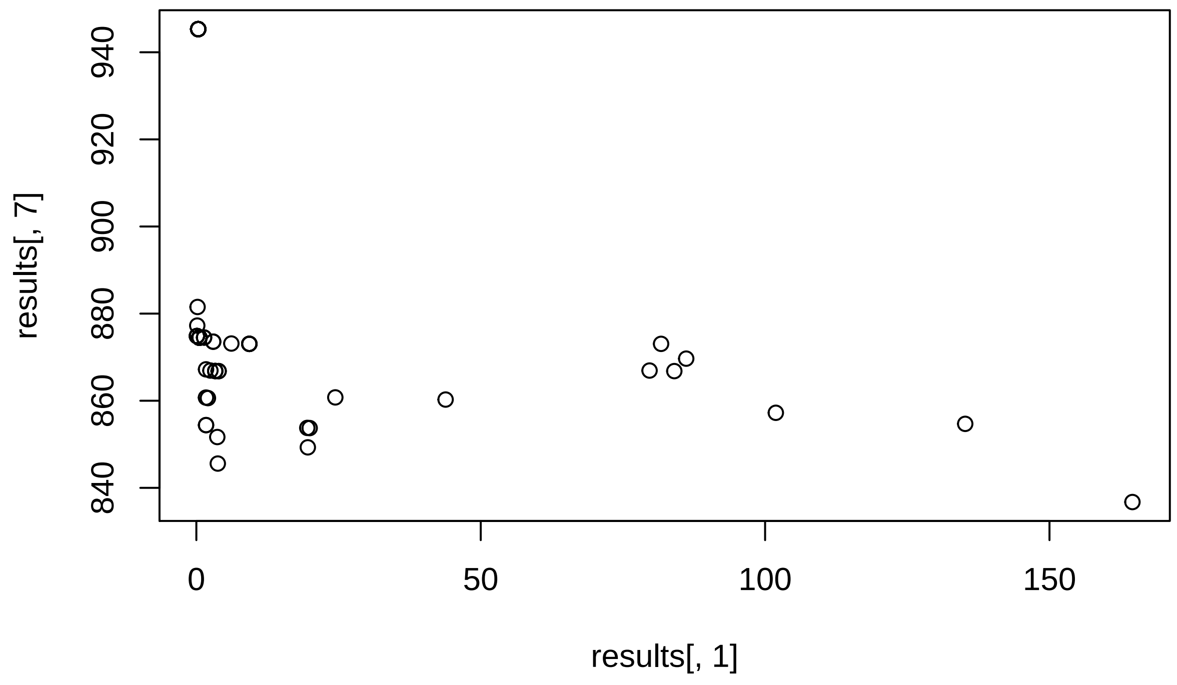
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

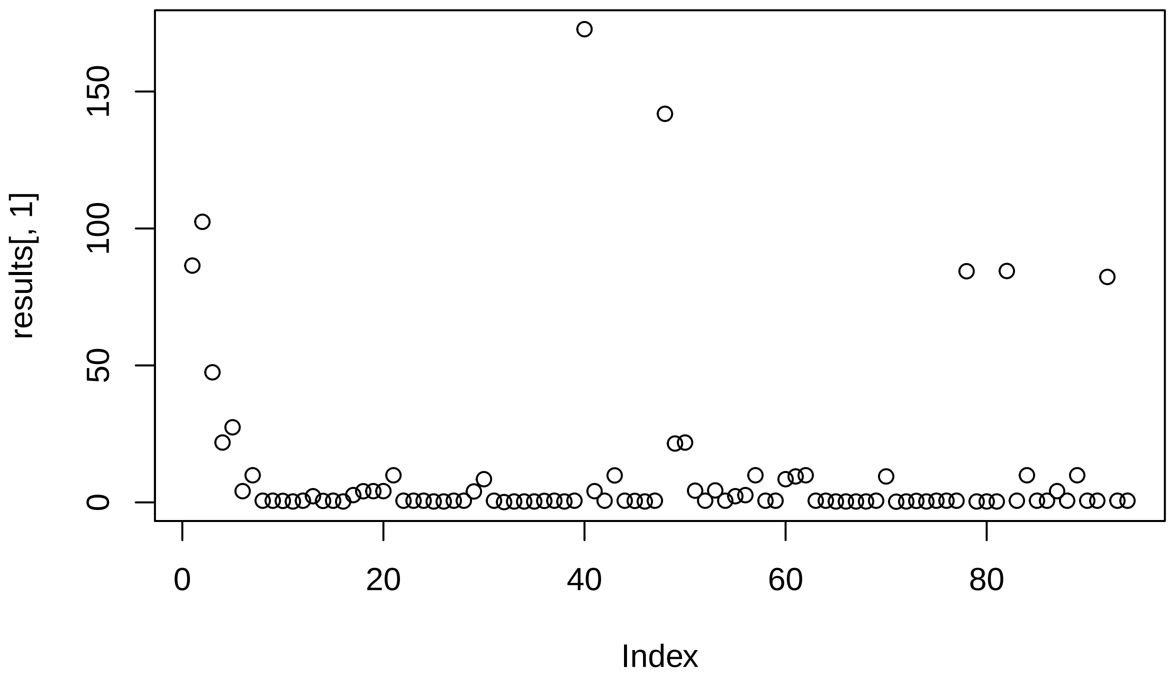




```
##  1 factor completed in 0.00443 minutes.    Estimated time of completion: 2020-04-28 10:41:10    [1]
##  [7]  10.0000010  90.0000010  78.6860377 874.8533861 164.5843546 836.7453386
## [13]  10.0000000  40.0000000
## [1] "j = 20.000001"
```

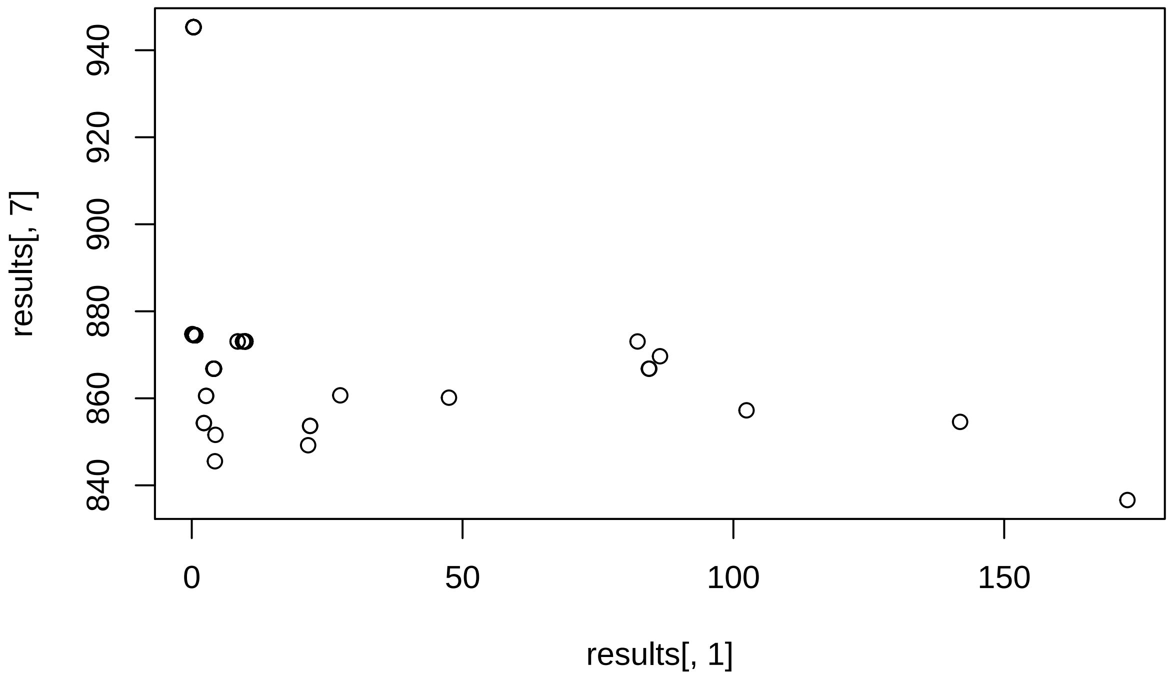
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

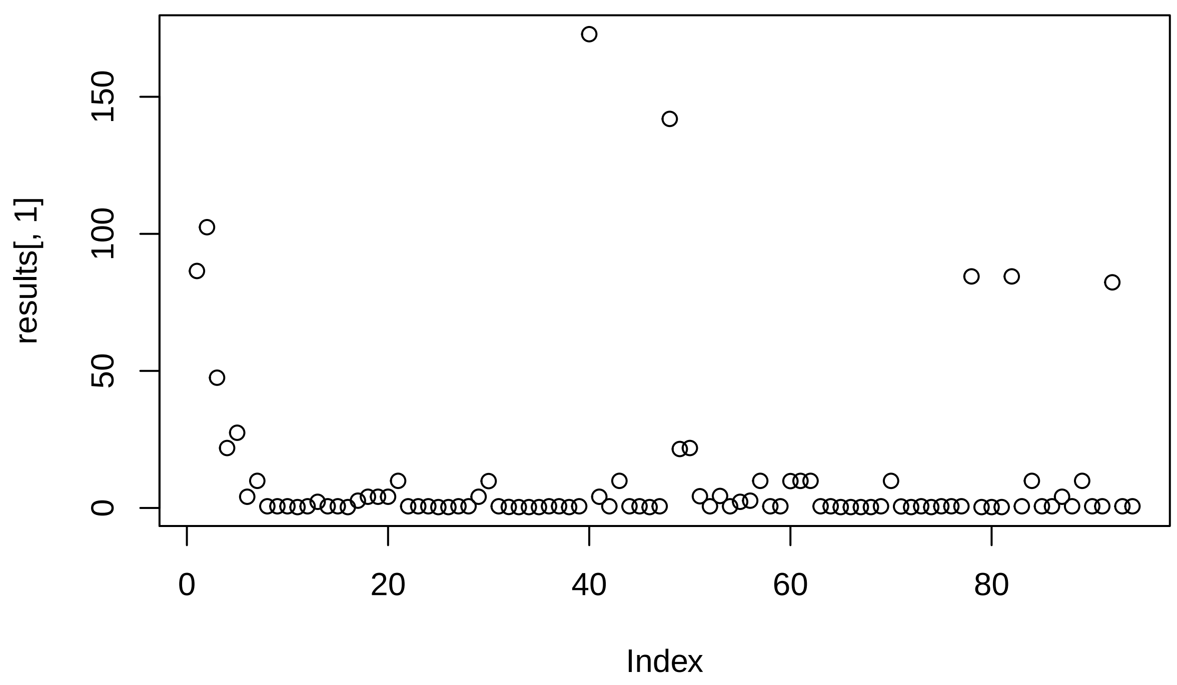




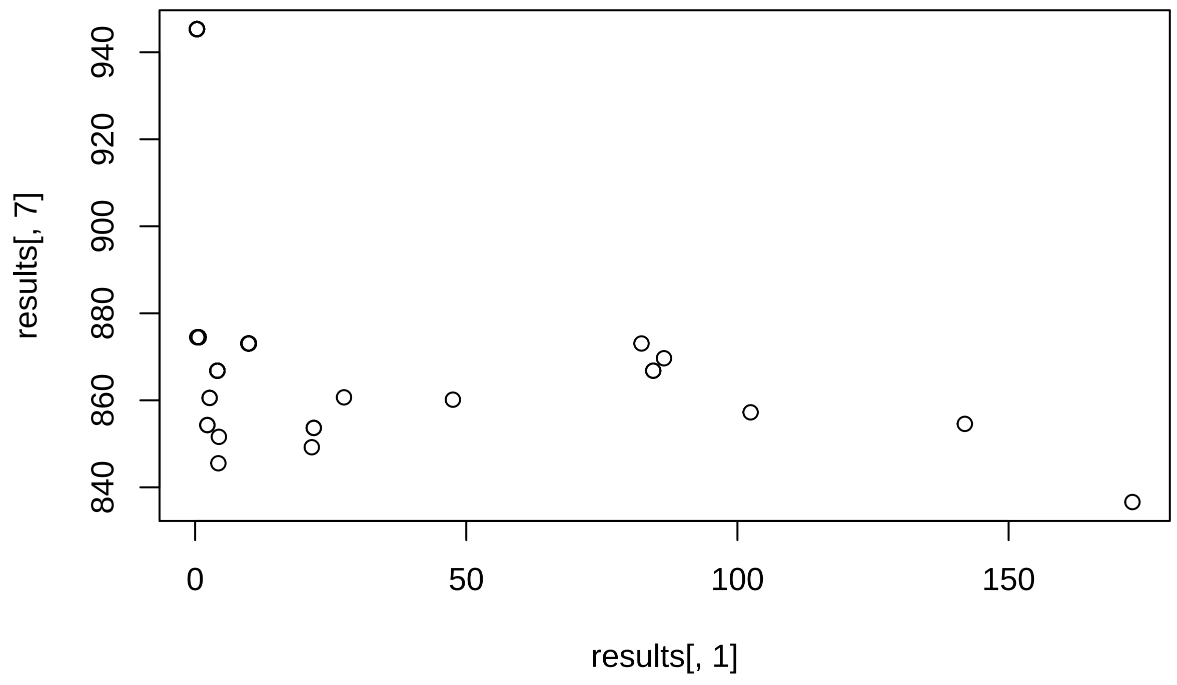
```
##  1 factor completed in 0.00611 minutes.    Estimated time of completion: 2020-04-28 10:41:13    [1]
##  [7]  20.0000010  90.0000010  78.6860377 874.7591659 172.7593864 836.6163454
## [13]  32.0000000  40.0000000
## [1] "j = 30.000001"
```

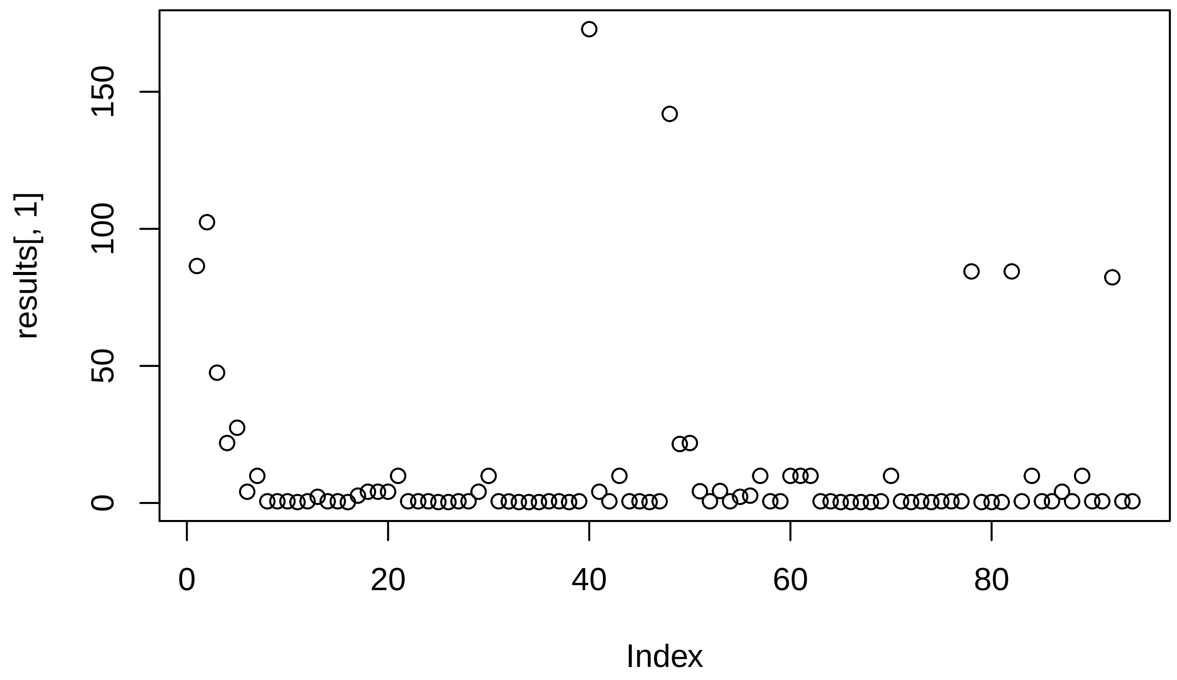
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```





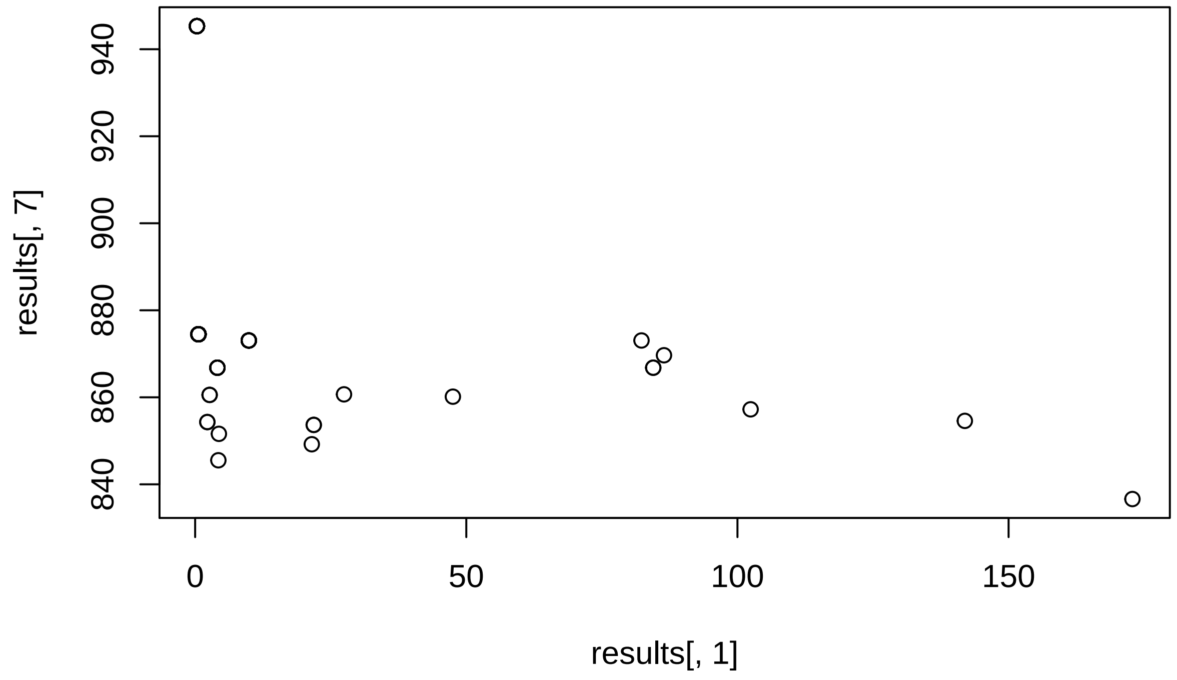
```
##  1 factor completed in 0.00424 minutes.    Estimated time of completion: 2020-04-28 10:41:15    [1]
##  [7]  30.0000010  90.0000010  78.6860377 945.3083537 172.8347422 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 40.000001"

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

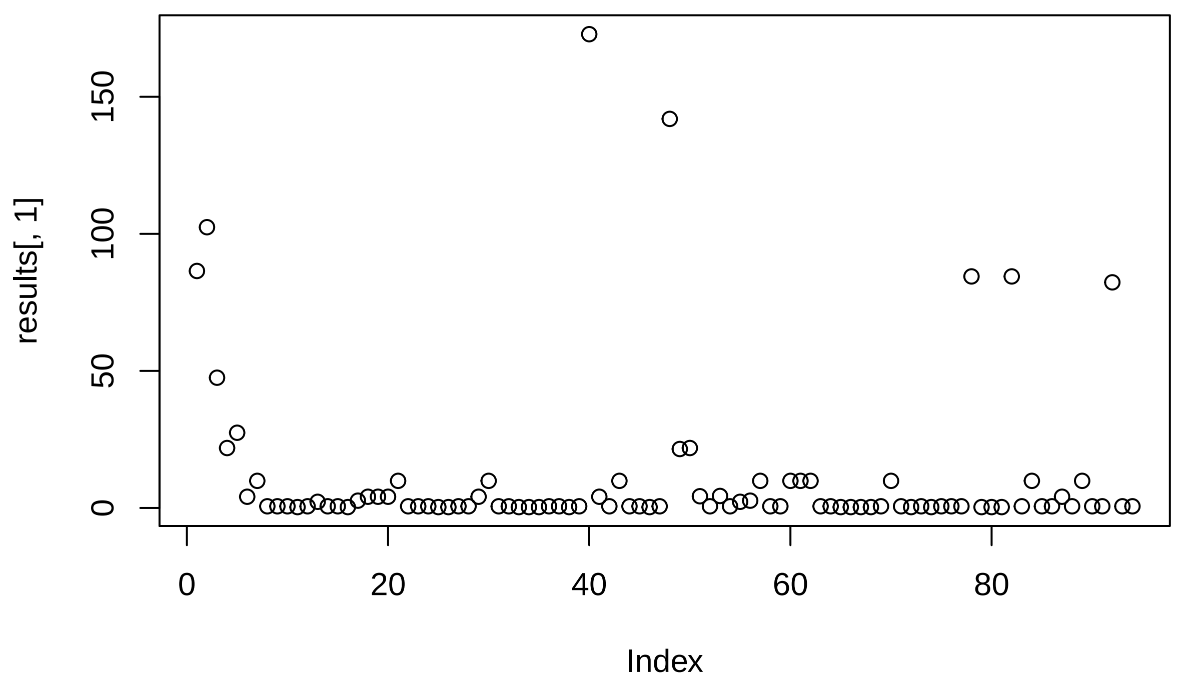




```
##  1 factor completed in 0.00435 minutes.    Estimated time of completion: 2020-04-28 10:41:17    [1]
##  [7]  40.0000010  90.0000010  78.6860377 945.3083537 172.8354127 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 50.000001"
```

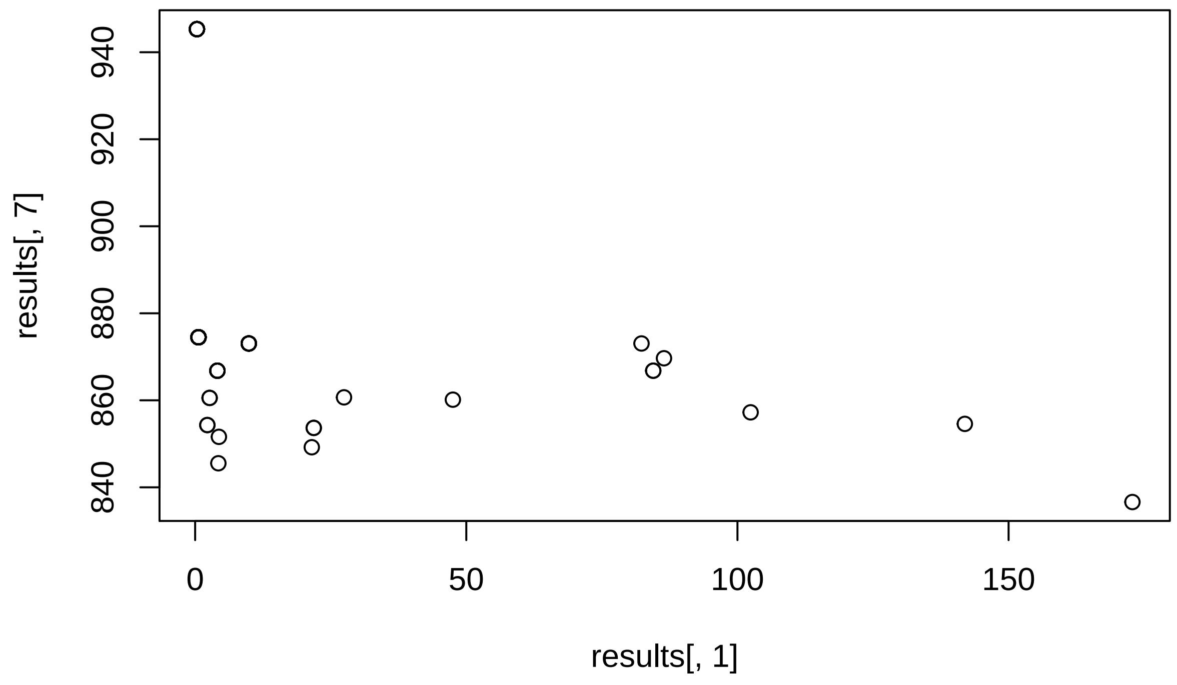
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

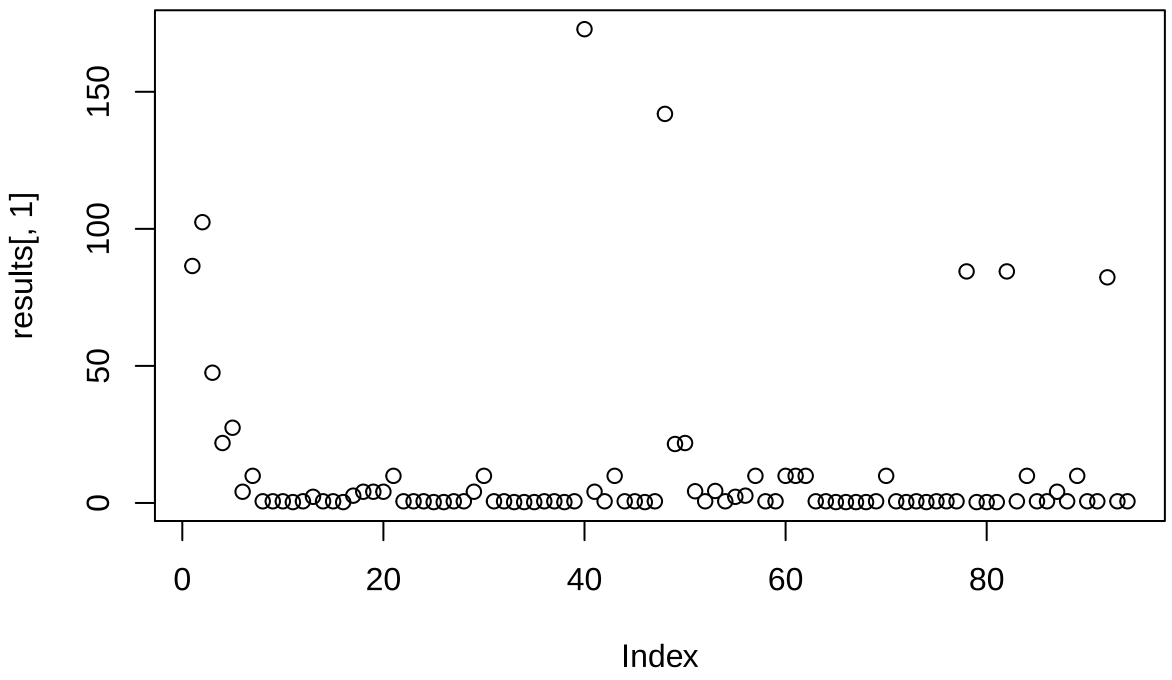




```
##  1 factor completed in 0.00406 minutes.    Estimated time of completion: 2020-04-28 10:41:19    [1]
##  [7]  50.0000010  90.0000010  78.6860377 945.3083537 172.8354187 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 60.000001"
```

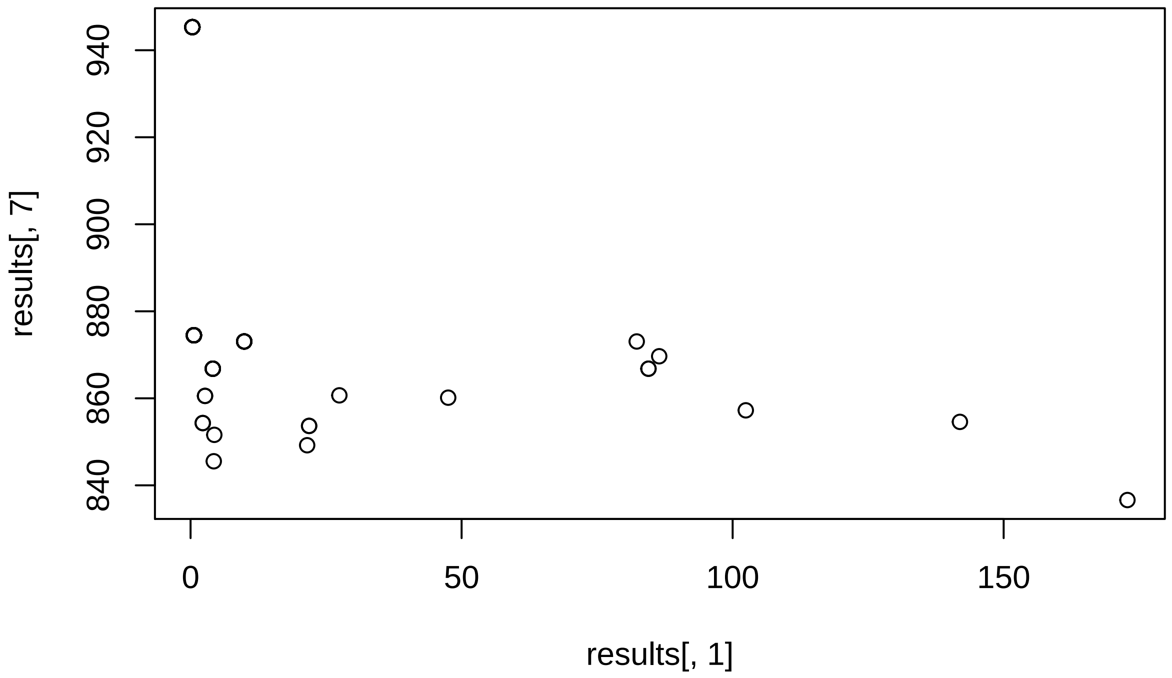
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

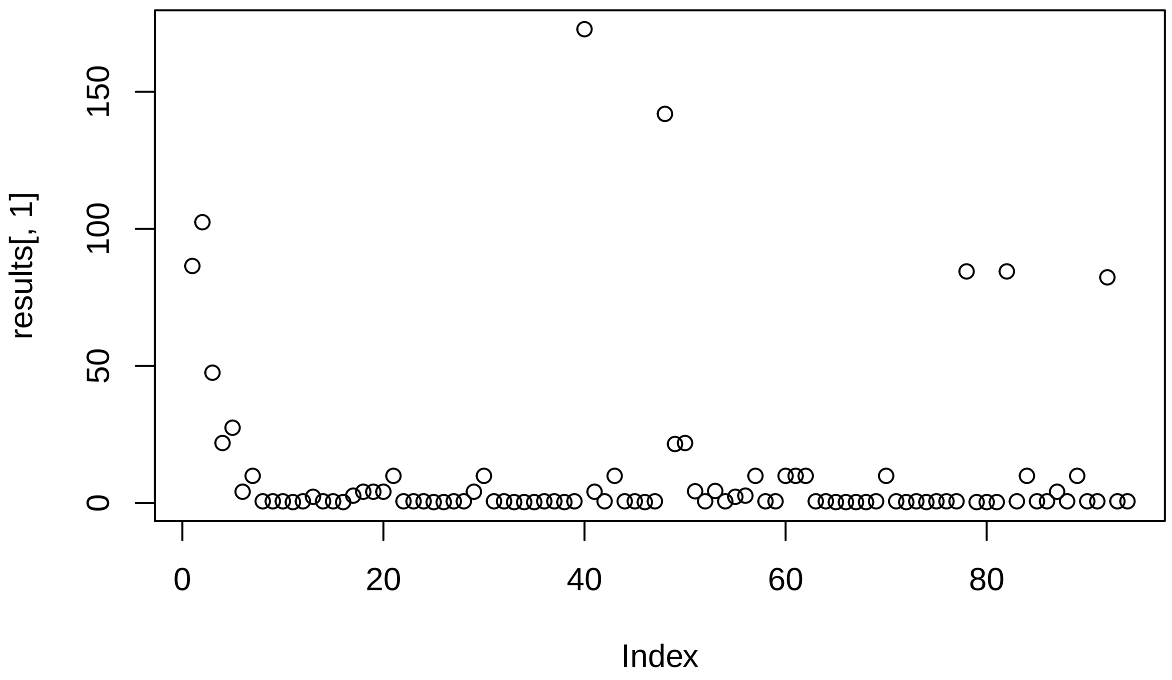




```
##  1 factor completed in 0.00436 minutes.    Estimated time of completion: 2020-04-28 10:41:21    [1]
##  [7]  60.0000010  90.0000010  78.6860377 945.3083537 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 70.000001"
```

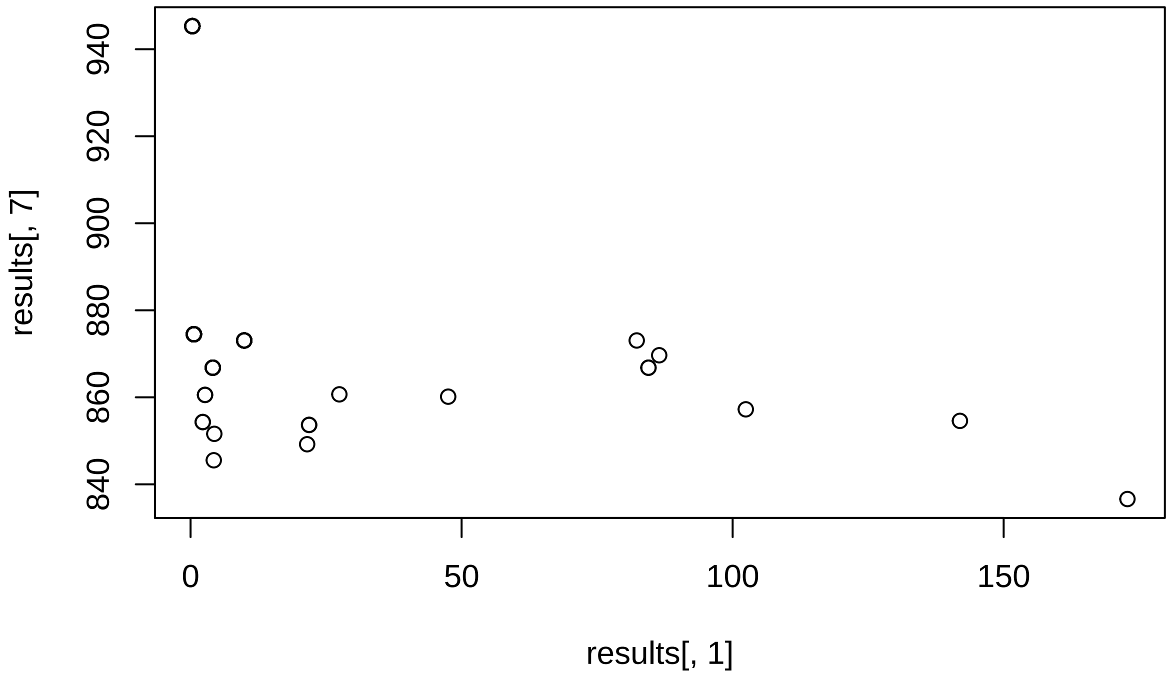
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

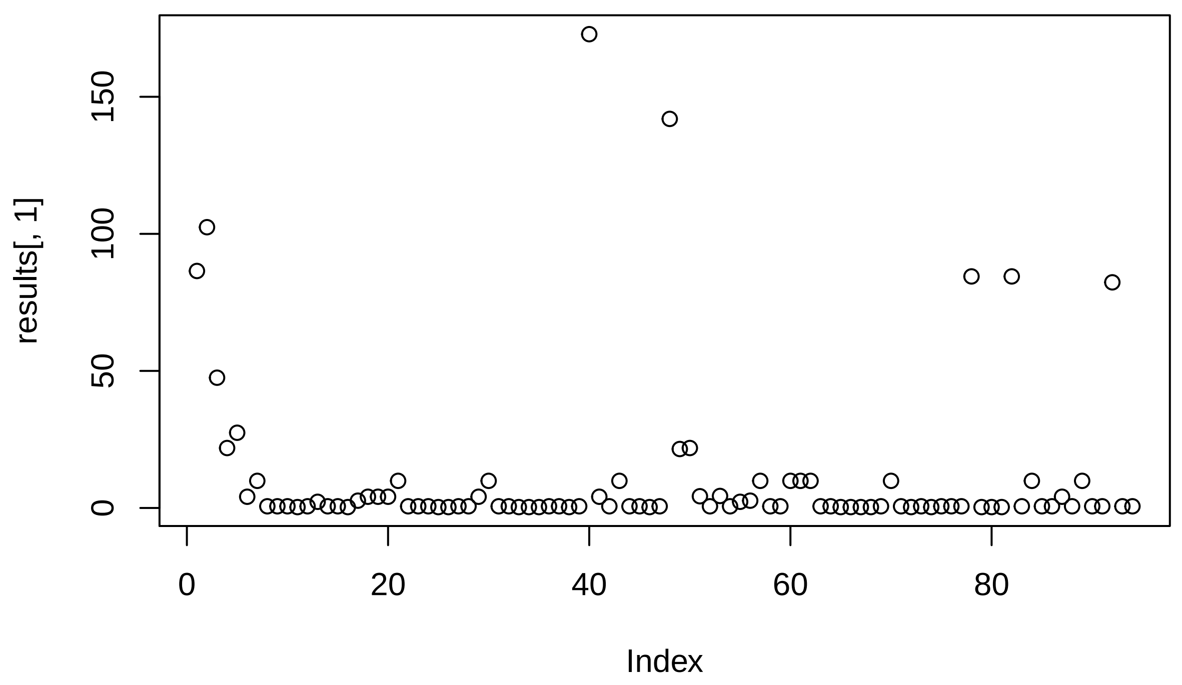




```
##  1 factor completed in 0.00419 minutes.    Estimated time of completion: 2020-04-28 10:41:23    [1]
##  [7]  70.0000010  90.0000010  78.6860377 945.3083537 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 80.000001"
```

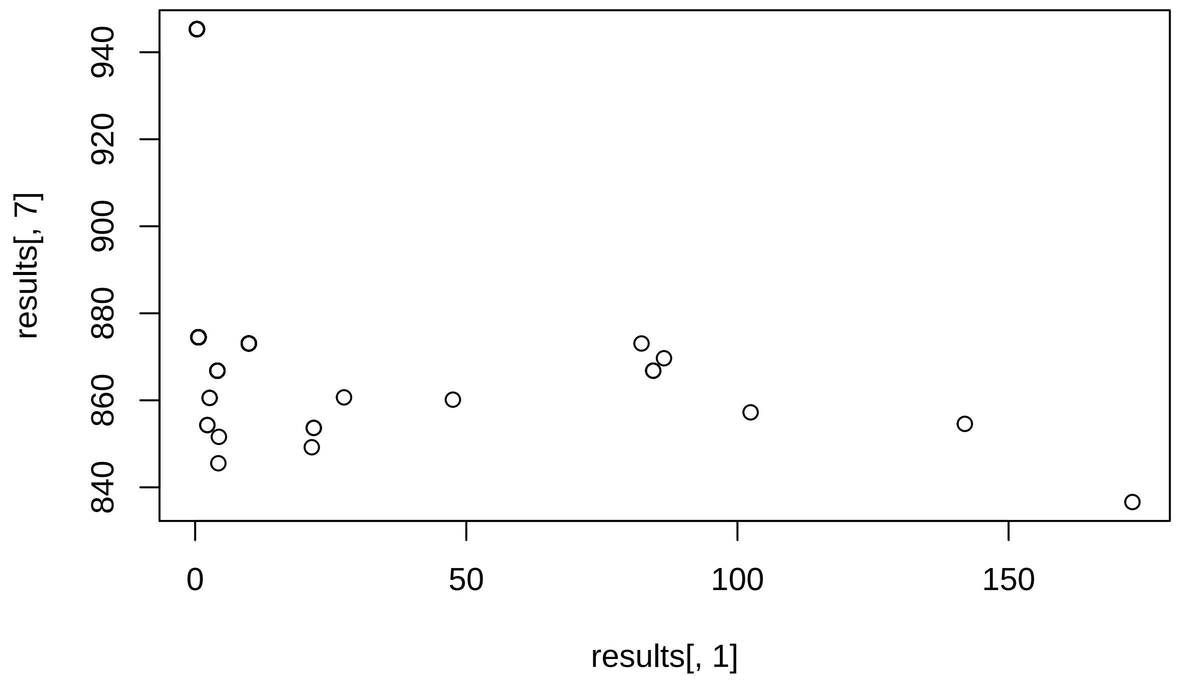
```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

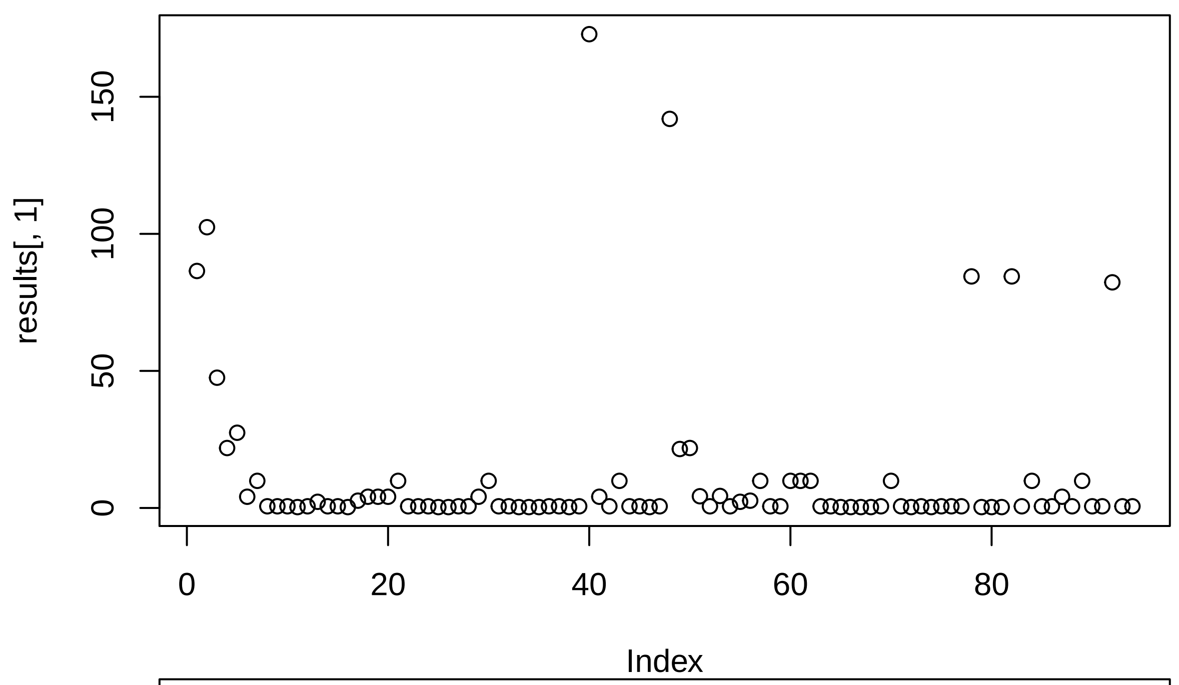


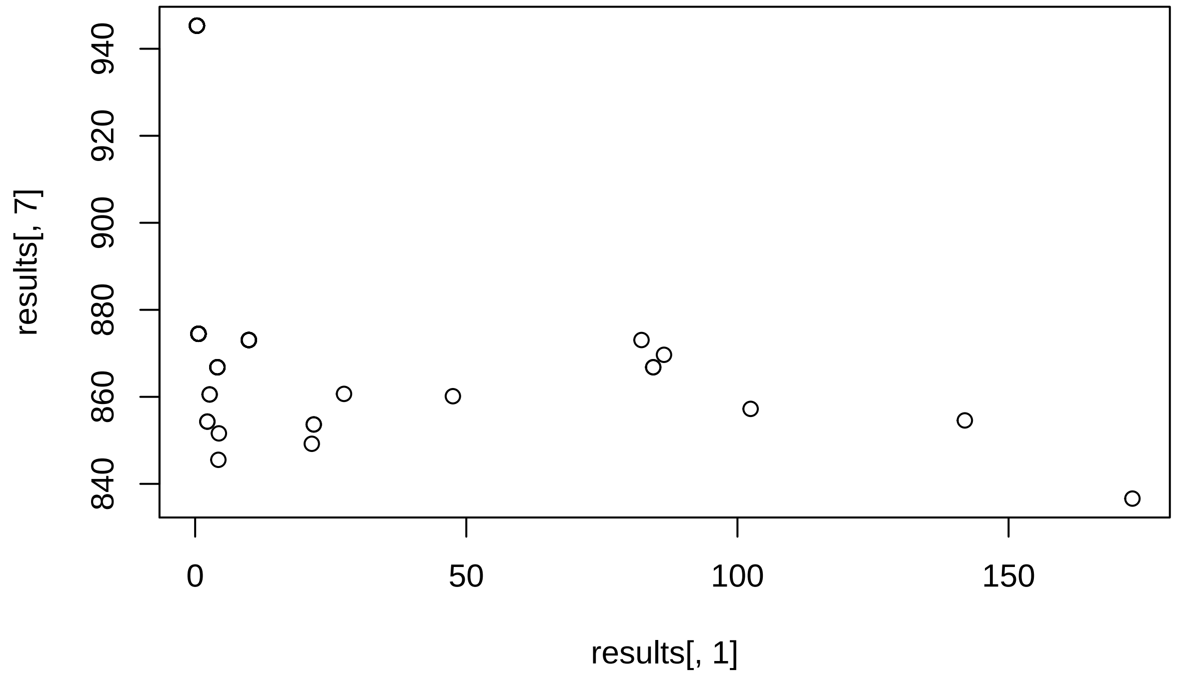


```
##  1 factor completed in 0.00408 minutes.    Estimated time of completion: 2020-04-28 10:41:25    [1]
##  [7]  80.0000010  90.0000010  78.6860377 945.3083537 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
## [1] "j = 90.000001"
```

```
## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.

## Warning in test_BodySize[, "intercept"] * beta_ridge[2] + t(basis_map) * : Recycling array of length
##   Use c() or as.vector() instead.
```

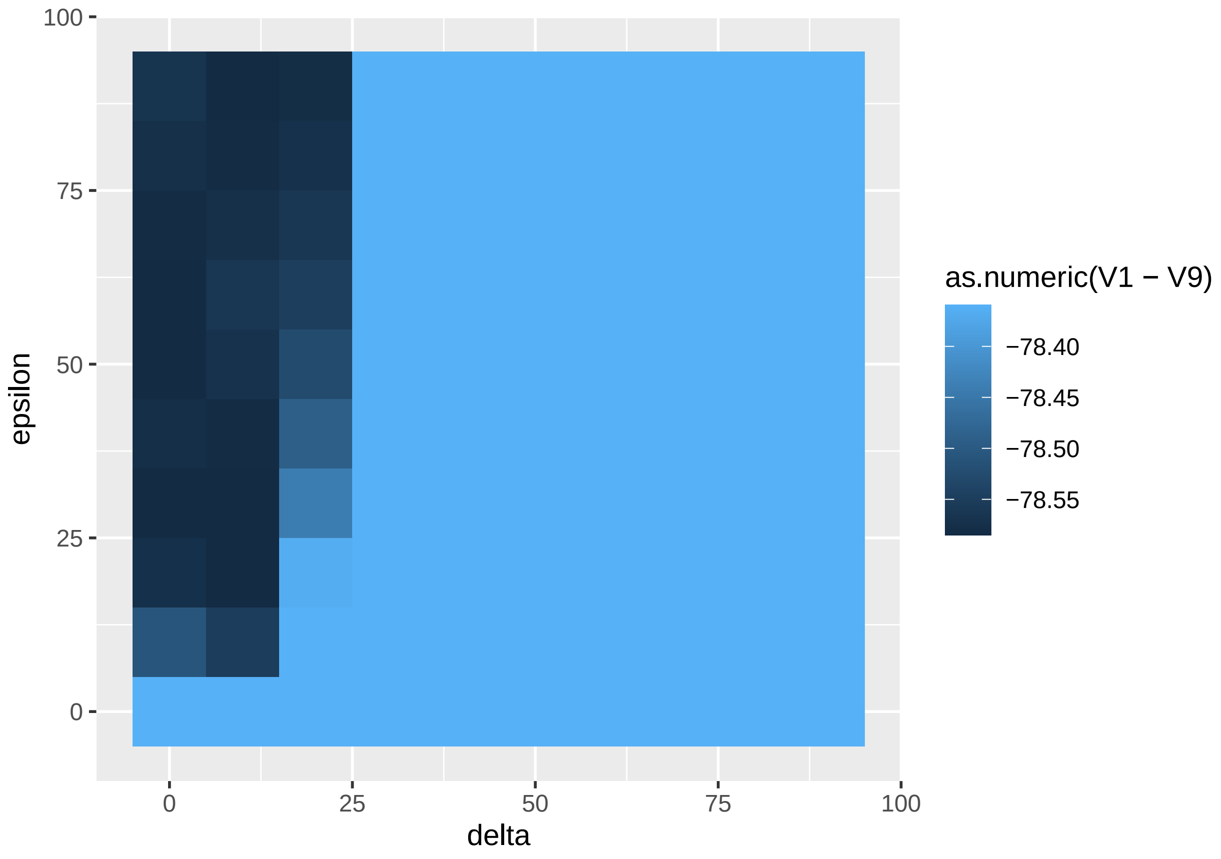




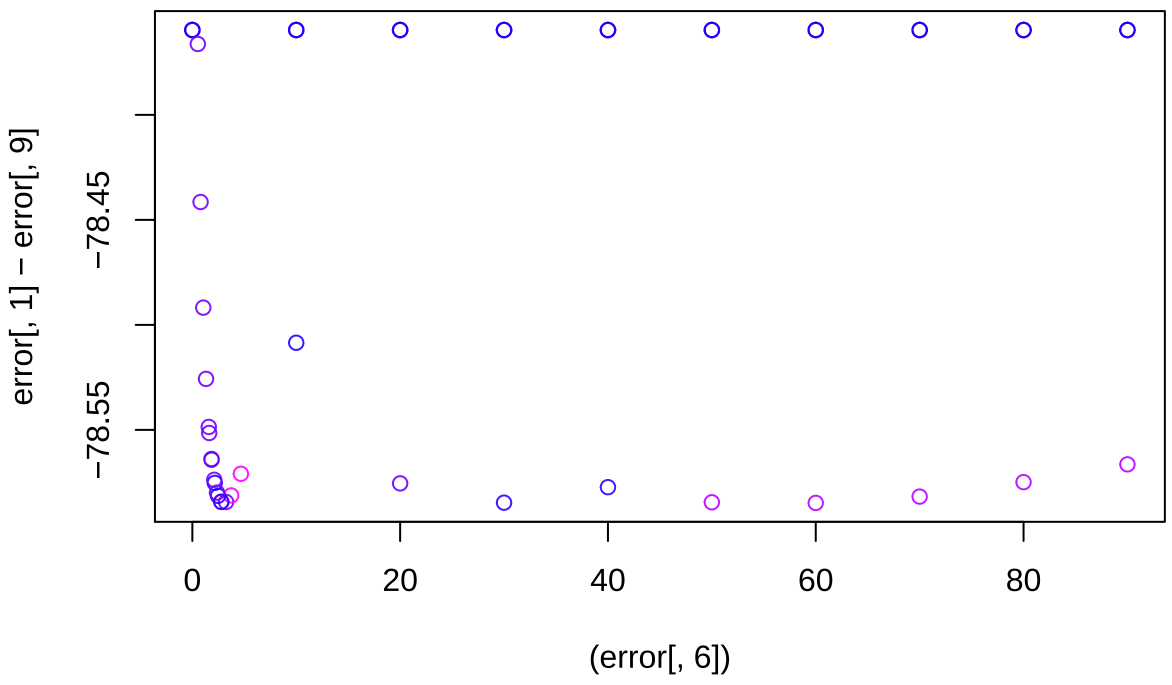


```
##  1 factor completed in 0.00438 minutes.    Estimated time of completion: 2020-04-28 10:41:27    [1]
##  [7]  90.0000010  90.0000010  78.6860377 945.3083537 172.8354188 836.6163348
## [13]  11.0000000  40.0000000
```

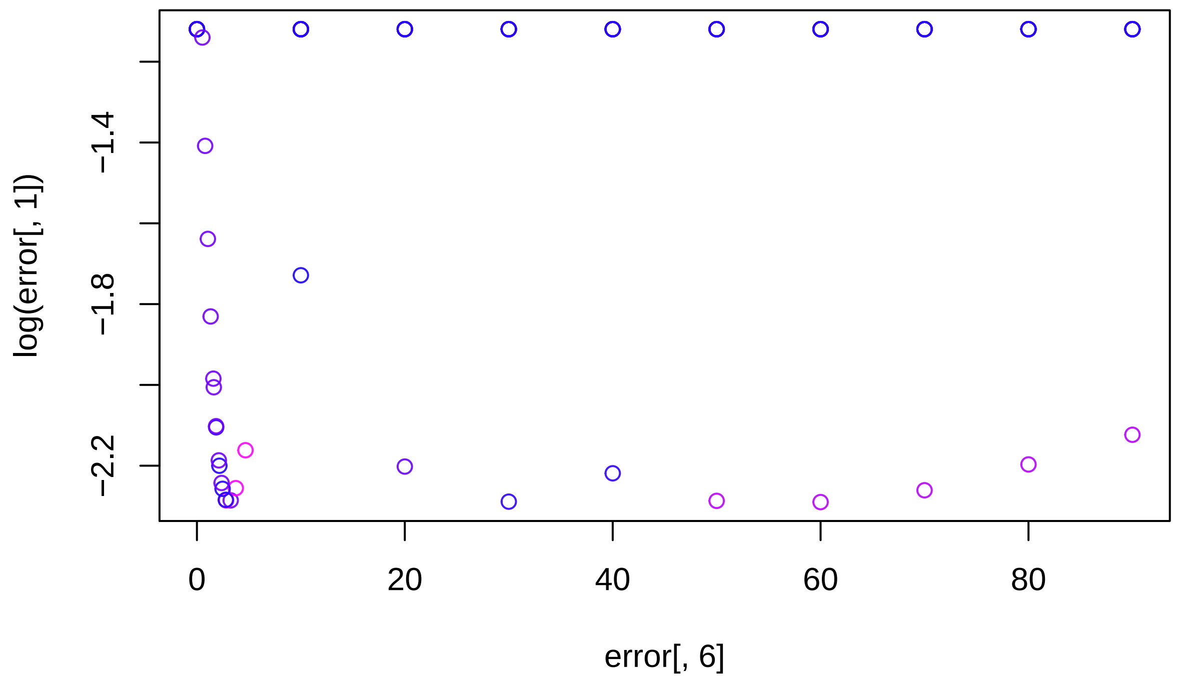
```
error %>%
  as.data.frame() %>%
  ggplot2::ggplot(aes(V7, V8)) +
  ggplot2::geom_tile(aes(fill= as.numeric(V1 - V9) )) +
  ggplot2::xlab("delta") +
  ggplot2::ylab("epsilon")
```



```
plot(y = error[,1] - error[,9], x = (error[,6]), col = rgb(red = error[,13]/max(error[,13]), green = 0,
```



```
plot(y = log(error[,1]), x = error[,6], col = rgb(red = error[,13]/max(error[,13]), green = 0, blue = 1
```



```
error[,13]
```

```
##   [1] 11 11 11 11 11 11 11 11 11 11  8 32 11 11 11 11 11 11 11 11 29 32 32 11 11
##  [26] 11 11 11 11 11 13 71 32 11 11 11 11 11 11 11 13 71 32 11 11 11 11 11 11 11
##  [51] 53 71 32 11 11 11 11 11 11 11 51 10 32 11 11 11 11 11 11 11 51 10 32 11 11
##  [76] 11 11 11 11 11 51 10 32 11 11 11 11 11 11 11 51 10 32 11 11 11 11 11 11 11
```

```
error[,14]
```

```
##   [1] 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 51 40 40 40 40
##  [26] 40 40 40 40 40 51 40 40 40 40 40 40 40 40 40 51 40 40 40 40 40 40 40 40 40
##  [51] 51 40 40 40 40 40 40 40 40 40 51 40 40 40 40 40 40 40 40 40 51 40 40 40 40
##  [76] 40 40 40 40 40 51 40 40 40 40 40 40 40 40 40 51 40 40 40 40 40 40 40 40 40
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[1]][[1]][[1]],]$Species
```

```
## [1] "3"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[2]][[1]][[1]],]$Species
```

```
## [1] "1"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[3]][[1]][[1]],]$Species
```

```
## [1] "15" "16" "17" "18"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[4]][[1]][[1]],]$Species
```

```
## [1] "14" "15" "16" "17" "18" "19"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[5]][[1]][[1]],]$Species
```

```
## [1] "5"  "6"  "7"  "8"  "9"  "10" "11" "12"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[6]][[1]][[1]],]$Species
```

```
## [1] "34" "35" "36"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[7]][[1]][[1]],]$Species
```

```
## [1] "25" "26" "27" "28" "29" "30" "31" "32" "33"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[8]][[1]][[1]],]$Species
```

```
##  [1] "24" "25" "26" "27" "28" "29" "30" "31" "32" "33"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[9]][[1]][[1]],]$Species
```

```
## [1] "3" "4"
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[10]][[1]][[1]],]$Species
```

```
## character(0)
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[11]][[1]][[1]],]$Species
```

```
## character(0)
```

```
train_BodySize[getPhyloGroups(train_tree)[unique(error[,13])[12]][[1]][[1]],]$Species
```

```
## character(0)
```

```
#theta_2 =  epsilon * exp(-delta * N1 * N2)

BodySize[clade2,]$Species
```

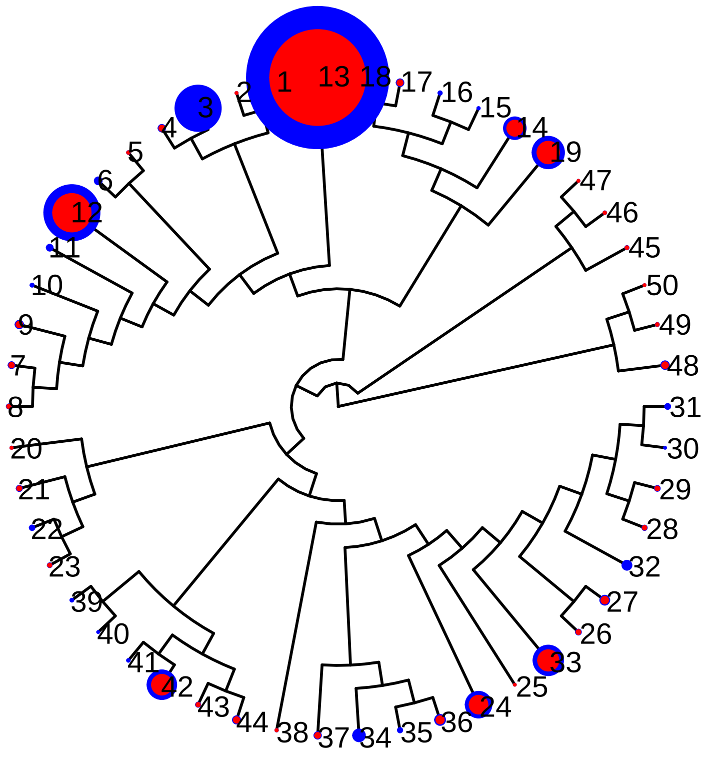
```
##  [1] "20" "21" "22" "23" "24" "25" "26" "27" "28" "29" "30" "31" "32" "33" "34"
## [16] "35" "36" "37" "38" "39" "40" "41" "42" "43" "44" "45" "46" "47" "48" "49"
## [31] "50"
```

```
BodySize[clade1,]$Species
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
## [16] "16" "17" "18" "19"
```

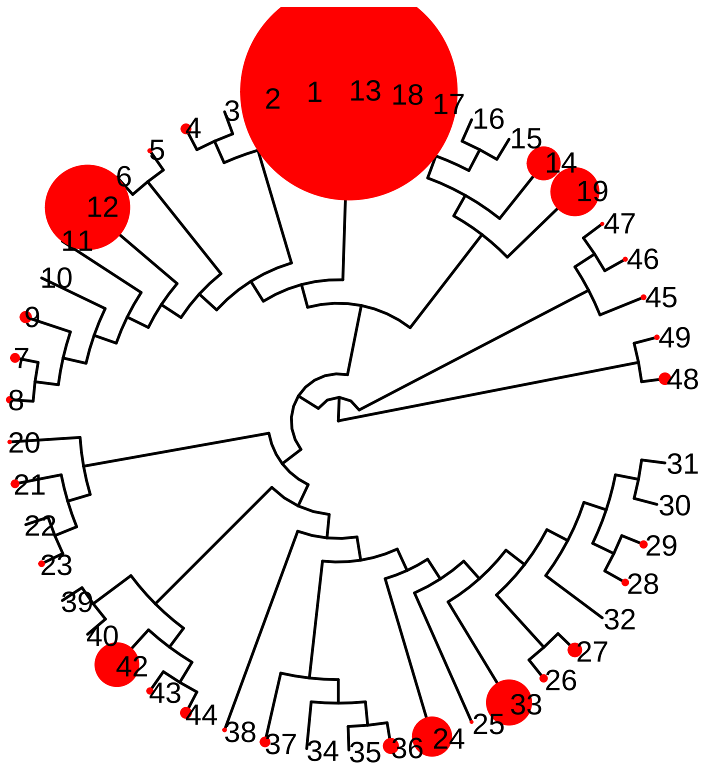
```
ggtree::ggtree(tree, branch.length = 'none', layout = 'circular') +
  ggtree::geom_tippoint(size=.15*as.numeric(BodySize$BodySize),col='blue')  +
  ggtree::geom_tippoint(size=.10*as.numeric(as.character(BodySize$BodySize_miss)),col='red') +
  ggtree::geom_tiplab()
```

```
## Warning: Removed 16 rows containing missing values (geom_point_g_gtree).
```

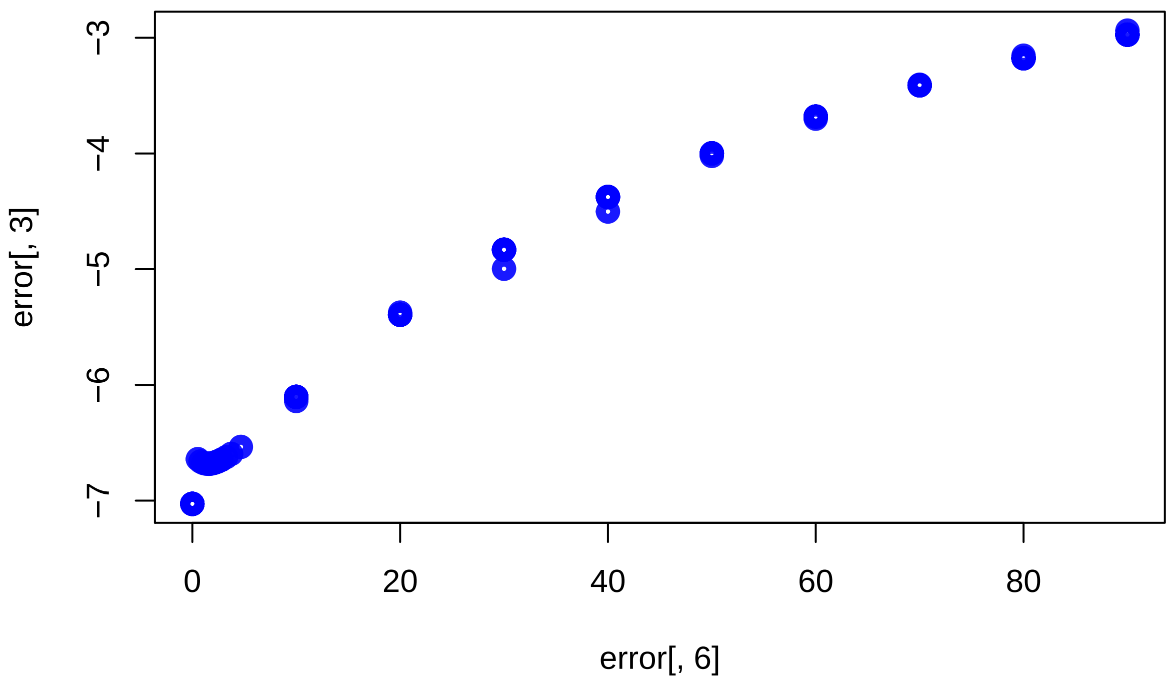


```r
ggtree::ggtree(train_tree, branch.length = 'none', layout = 'fan') +
  ggtree::geom_tippoint(size=.23*as.numeric(as.character(train_BodySize$BodySize_miss)),col='red') +
  ggtree::geom_tiplab()
```

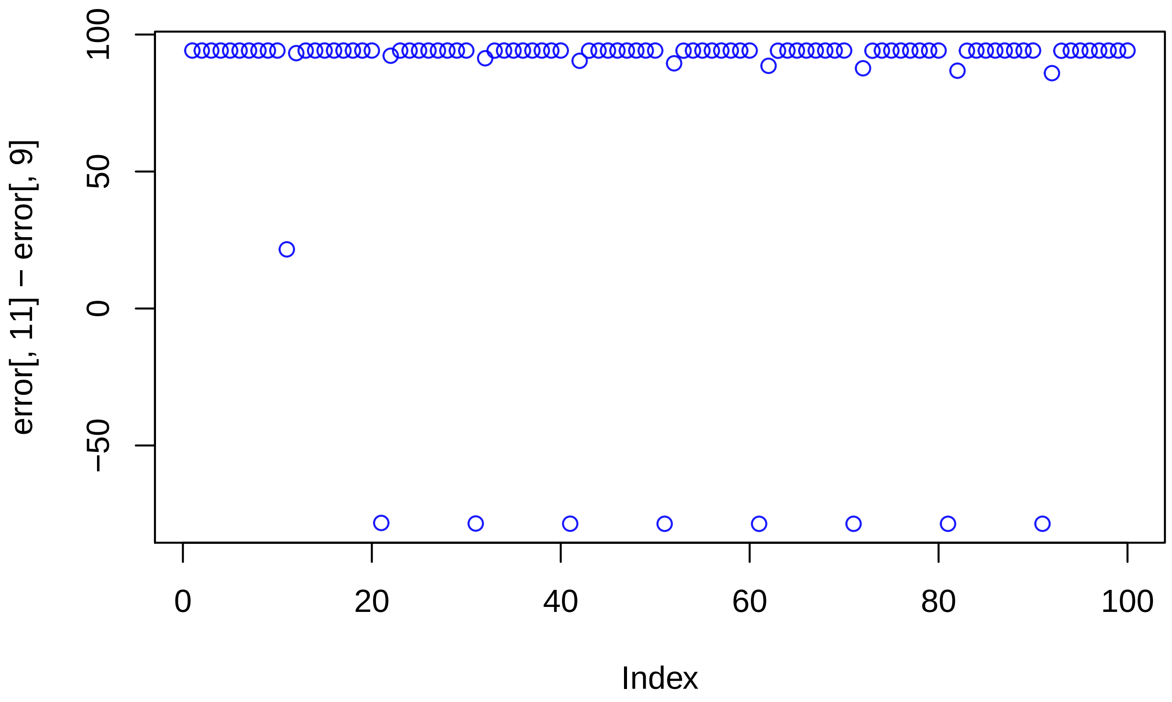
```
## Warning: Removed 15 rows containing missing values (geom_point_g_gtree).
```



```
# plot((error[,1]), type = 'l', lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
# plot(error[,1], error[,7], lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
# plot(error[,1], error[,8], lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
# plot(error[,1], error[,6], lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
#
plot(y =error[,3], x=error[,6], lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
```



```
#
# plot((error[,10]), type = 'l', lwd = 5, col = rgb(red = 1, green = 0, blue = 1, alpha = 0.9))
#
# plot((error[,11]), type = 'l', lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
# plot((error[,12]), type = 'l', lwd = 5, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))

plot(error[,11] - error[,9], col = rgb(red = 0, green = 0, blue = 1, alpha = 0.9))
```



error[, 11] − error[, 9]
100
50
0
−50
0
20
40
60
80
100
Index