

# Package: svrpath (via r-universe)

September 14, 2024

**Type** Package

**Title** The SVR Path Algorithm

**Version** 0.1.2

**Description** Computes the entire solution paths for Support Vector Regression(SVR) with respect to the regularization parameter, lambda and epsilon in epsilon-intensive loss function, efficiently. We call each path algorithm svrpath and epspath. See Wang, G. et al (2008) <doi:10.1109/TNN.2008.2002077> for details regarding the method.

**Imports** quadprog, svmpath

**Depends** R (>= 3.4.0)

**License** GPL (>= 2)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Do Hyun Kim [aut, cre], Seung Jun Shin [aut]

**Maintainer** Do Hyun Kim <09dohkim@gmail.com>

**Date/Publication** 2018-06-29 14:07:55 UTC

**Repository** https://09dohkim.r-universe.dev

**RemoteUrl** https://github.com/cran/svrpath

**RemoteRef** HEAD

**RemoteSha** 9d32f1832a1066855bc72c3df95acc511a95f7e5

## Contents

epspath . . . . .	2
plot.epspath . . . . .	3
plot.svrpath . . . . .	4
predict.epspath . . . . .	4

predict.svrpath . . . . .	5
solve.svr . . . . .	6
svrpath . . . . .	7

<b>Index</b>	<b>9</b>
--------------	----------

---

epspath	<i>Fit the entire epsilon path for Support Vector Regression</i>
---------	--

---

### Description

The Support Vector Regression (SVR) employs epsilon-intensive loss which ignores errors smaller than epsilon. This algorithm computes the entire paths for SVR solution as a function of epsilon at a given regularization parameter lambda, which we call epsilon path.

### Usage

```
epspath(x, y, lambda = 1, kernel.function = radial.kernel,
        param.kernel = 1, ridge = 1e-08, eps = 1e-07, eps.min = 1e-08, ...)
```

### Arguments

x	The data matrix (n x p) with n rows (observations) on p variables (columns)
y	The real number valued response variable
lambda	The regularization parameter value.
kernel.function	User defined kernel function. See svmpath.
param.kernel	Parameter(s) of the kernels. See svmpath.
ridge	Sometimes the algorithm encounters singularities; in this case a small value of ridge can help, default is ridge = 1e-8
eps	A small machine number which is used to identify minimal step sizes
eps.min	The smallest value of epsilon for termination of the algorithm. Default is eps.min = 1e-8
...	Generic compatibility

### Value

An 'epspath' object is returned.

### Author(s)

Do Hyun Kim, Seung Jun Shin

### See Also

[predict.epspath](#), [plot.epspath](#), [svrpath](#)

**Examples**

```

set.seed(1)
n <- 30
p <- 50

x <- matrix(rnorm(n*p), n, p)
e <- rnorm(n, 0, 1)
beta <- c(1, 1, rep(0, p-2))
y <- x %*% beta + e
lambda <- 1
eobj <- epspath(x, y, lambda = lambda)

```

---

plot.epspath

*plot the epspath, solution paths of SVR as a function of epsilon*


---

**Description**

produces a plot of the SVR epsilon path.

**Usage**

```

## S3 method for class 'epspath'
plot(x, intercept = FALSE, ...)

```

**Arguments**

x	The epspath object
intercept	If it is TRUE, then an intercept path plot is given.
...	Generic compatibility

**Value**

The entire solution path of SVR solution as a function of epsilon.

**Author(s)**

Do Hyun Kim, Seung Jun Shin

**Examples**

```

# The 'eobj' is given by examples description of epspath().
plot(eobj, lty = 2, lwd = 2, col = 2, cex.lab = 1.5)

```

---

plot.svrpath	<i>plot the svrpath, solution paths of SVR as a function of lambda</i>
--------------	--

---

**Description**

produces a plot of the SVR lambda path.

**Usage**

```
## S3 method for class 'svrpath'
plot(x, intercept = FALSE, ...)
```

**Arguments**

x	The svrpath object
intercept	If it is TRUE, then an intercept path plot is given.
...	Generic compatibility

**Value**

The entire solution path of SVR solution as a function of lambda.

**Author(s)**

Do Hyun Kim, Seung Jun Shin

**Examples**

```
# The 'obj' is given by examples description of svrpath().
plot(obj, lty = 2, lwd = 2, col = 2, cex.lab = 1.5)
```

---

predict.epspath	<i>Make predictions from an "epspath" object</i>
-----------------	--

---

**Description**

Provides a prediction value at a given epsilon from epspath object.

**Usage**

```
## S3 method for class 'epspath'
predict(object, newx, svr.eps = 1, ...)
```

**Arguments**

object	The epspath object
newx	Values of x to be predicted. This is a matrix with observations per row. Default is x in the epspath object.
svr.eps	The value of the "epsilon-insensitive loss" paramter, epsilon.
...	Generic compatibility

**Value**

In each case, the desired prediction.

**Author(s)**

Do Hyun Kim, Seung Jun Shin

**Examples**

```
# The 'eobj' is given by examples description of epspath().
predict(eobj, svr.eps = .1)
```

---

predict.svrpath      *Make predictions from a "svrpath" object*

---

**Description**

Provides a prediction value at a given lambda from svrpath object.

**Usage**

```
## S3 method for class 'svrpath'
predict(object, newx, lambda = NULL, criterion = "sic",
        ...)
```

**Arguments**

object	The svrpath object
newx	Values of x to be predicted. This is a matrix with observations per row. Default is x in the epspath object.
lambda	The value of the regularization paramter, lambda.
criterion	It provides predictions at an optimal lambda selected by SIC or GACV. "sic" or "gacv".
...	Generic compatibility

**Value**

In each case, the desired prediction.

**Author(s)**

Do Hyun Kim, Seung Jun Shin

**Examples**

```
# The 'eobj' is given by examples description of epath().
predict.svrpath(obj, lambda = 10) # or
predict(obj, criterion = 'sic')
```

---

solve.svr

*QP solver for SVR*

---

**Description**

solves quadratic programming(QP) for SVR.

**Usage**

```
## S3 method for class 'svr'
solve(a, b, lambda = 1, svr.eps = 1,
      kernel.function = radial.kernel, param.kernel = 1, ...)
```

**Arguments**

a	The data matrix (n x p) with n rows (observations) on p variables (columns)
b	The real number valued response variable
lambda	The regularization parameter
svr.eps	Epsilon in epsilon-insensitive loss function
kernel.function	User defined kernel function. See svmpath.
param.kernel	Parameter(s) of the kernels. See svmpath.
...	Generic compatibility

**Value**

SVR solution at a given lambda and epsilon

**Author(s)**

Dohyun Kim, Seung Jun Shin

**Examples**

```
# set.seed(1)
n <- 30
p <- 50

x <- matrix(rnorm(n*p), n, p)
e <- rnorm(n, 0, 1)
beta <- c(1, 1, rep(0, p-2))
y <- x %*% beta + e
solve.svr(x, y)
```

svrpath

*Fit the entire regularization path for Support Vector Regression***Description**

This algorithm computes the entire regularization path for the support vector regression with a relatively low cost compared to quadratic programming problem.

**Usage**

```
svrpath(x, y, svr.eps = 1, kernel.function = radial.kernel,
        param.kernel = 1, ridge = 1e-08, eps = 1e-08, lambda.min = 1e-08, ...)
```

**Arguments**

<code>x</code>	The data matrix (n x p) with n rows (observations) on p variables (columns)
<code>y</code>	The real number valued response variable
<code>svr.eps</code>	An epsilon in epsilon-insensitive loss function
<code>kernel.function</code>	This is a user-defined function. Provided are <code>poly.kernel</code> (the default, with parameter set to default to a linear kernel) and <code>radial.kernel</code>
<code>param.kernel</code>	The parameter(s) for the kernel. For this radial kernel, the parameter is known in the fields as "gamma". For the polynomial kernel, it is the "degree"
<code>ridge</code>	Sometimes the algorithm encounters singularities; in this case a small value of ridge can help, default is <code>ridge = 1e-8</code>
<code>eps</code>	A small machine number which is used to identify minimal step sizes
<code>lambda.min</code>	The smallest value of lambda for termination of the algorithm. Default is <code>lambda = 1e-8</code>
<code>...</code>	Generic compatibility

**Value**

A 'svrpath' object is returned, for which there are lambda values and corresponding values of the theta for each data point.

**Author(s)**

Do Hyun Kim, Seung Jun Shin

**See Also**

[predict.svrpath](#), [plot.svrpath](#), [eps.path](#)

**Examples**

```
set.seed(1)
n <- 30
p <- 50

x <- matrix(rnorm(n*p), n, p)
e <- rnorm(n, 0, 1)
beta <- c(1, 1, rep(0, p-2))
y <- x %*% beta + e
svr.eps <- 1
obj <- svrpath(x, y, svr.eps = svr.eps)
```



# Index

`epspath`, [2](#), [8](#)

`plot.epspath`, [2](#), [3](#)

`plot.svrpath`, [4](#), [8](#)

`predict.epspath`, [2](#), [4](#)

`predict.svrpath`, [5](#), [8](#)

`solve.svr`, [6](#)

`svrpath`, [2](#), [7](#)