User Guide for SLANTS

Jiaqi Liu

4/26/2019

Generate Data by Hand

The purpose of this experiment is to show the performance of SLANTS in stationary environment where the data generating model is fixed over time. We will generate two-dimensional time series data and treat X2 as our y. We want to detect the true relationship between the two dimensional time series and y.
The following function is used to creating lagged version of a time series vector.

```
utils.lag <- function(ts, lag = 1, pad = NA) {
  # return the lagged version of a time series vector
  return(c(rep(pad, lag), ts[1:(length(ts)-lag)]))
}
```

Here we generate two-dimensional time series with lag = 8, Gaussian noise and 2000 observations. D here stands for dimension and L here stands for lag.

```
Ex1 <- (function(N=2000, D=2, L=8){
  err <- matrix(rnorm(D*(N+L)), N+L, D)
  X <- err[ ,1]
  X <- cbind(X, 0.5 * utils.lag(X,1)^2 - 0.8 * utils.lag(X,7) + 0.2 * err[,2])
  list(N=N, D=D, L=L, X=X[-(1:L),], y=X[-(1:L),2])
})()
```

We set parameters for data preprocessing and model training. *Lambda parameter* we used here is LASSO penalty coefficient which allows to control the shrinkage speed.It is also a tuning parameter. *SrhinkStepSize* is the shrink step size, here we define it as $\frac{1}{t}$. *Order* is the order for b spline as well as the *nBspline* is the number of b splines that we want to fit the nolinear model. *SpaTol_gamma* is related to the tolerance of three gamma channels. We use it to choose the optimal channel. *MoveSize* is the ratio between the channels. It controls the distance between channels. *gamma_init* is the initial value for three channels,due to the randomdization coefficient in the algorithm. Some times Users need to choose a good initial value(smaller one). And the *alpha2_init* is the initial value for innovation parameter which helps to decompose the problem in EM algorithm.

```
Ex1_algo <- (function(experiment_config){
  ec <- experiment_config # short name
  lambda <- 1/c(1:ec$N) # same as batch
  shrinkStepSize <- 1/c(1:ec$N)
  list(
    order = 3,
    nBspline = 10,
    lambda = lambda,
    shrinkStepSize = shrinkStepSize,
    # if the performance does not exceed this ratio, the sparser (larger gamma) is preferred
    spaTol_gamma = 1.1,
    moveSize = 10^(0.4), # multipler to move among channels
    safeShrink_gamma = 10^(0.1), # to adjust gamma before they get too crazy
    gamma_init = 0.0001, #lambda in the paper,three channel
    alpha2_init = 0.05 #tao in the paper
```

```r
  )
})(Ex1)
```

getPreprocess helps to convert two-dimensional data into N x 2*8(DL) matrix and also get knots for b splines.

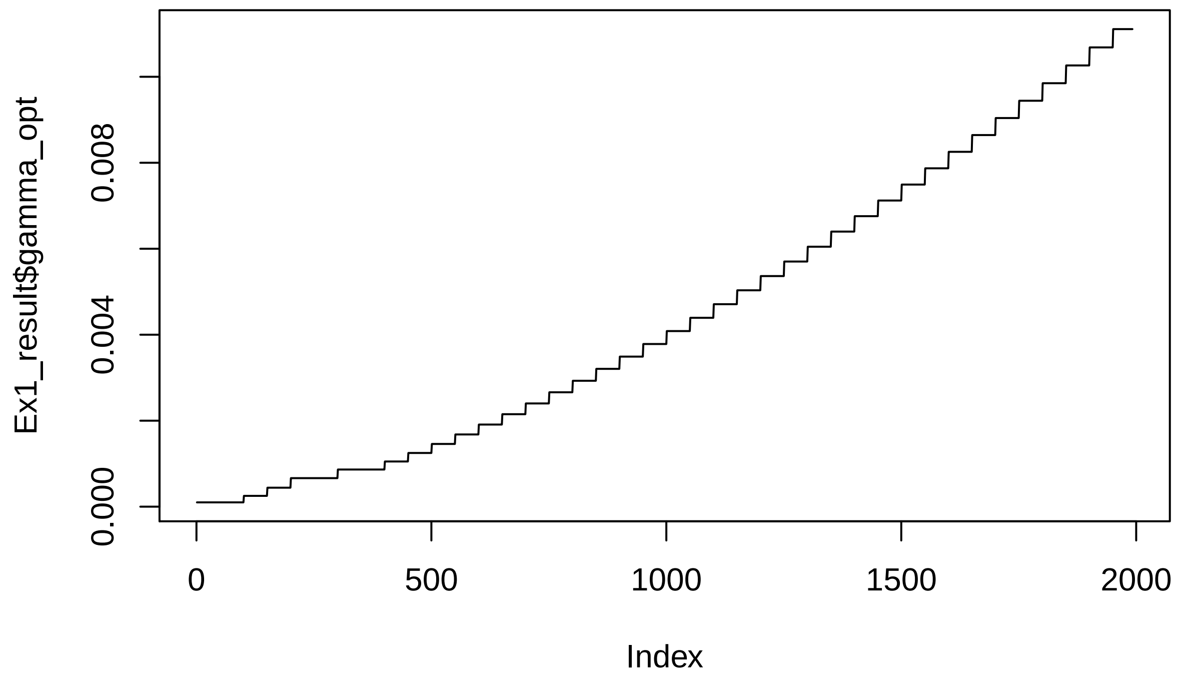
```r
Ex1_algo <- c(Ex1_algo, do.call(getPreprocess, c(Ex1, Ex1_algo)))
```

getSequentialNonlinearModel helps to use EM algorithm and B splines to get the model fitted. Parameter optimizing history, best coefficient and coefficient history are all returned in EX1_result.

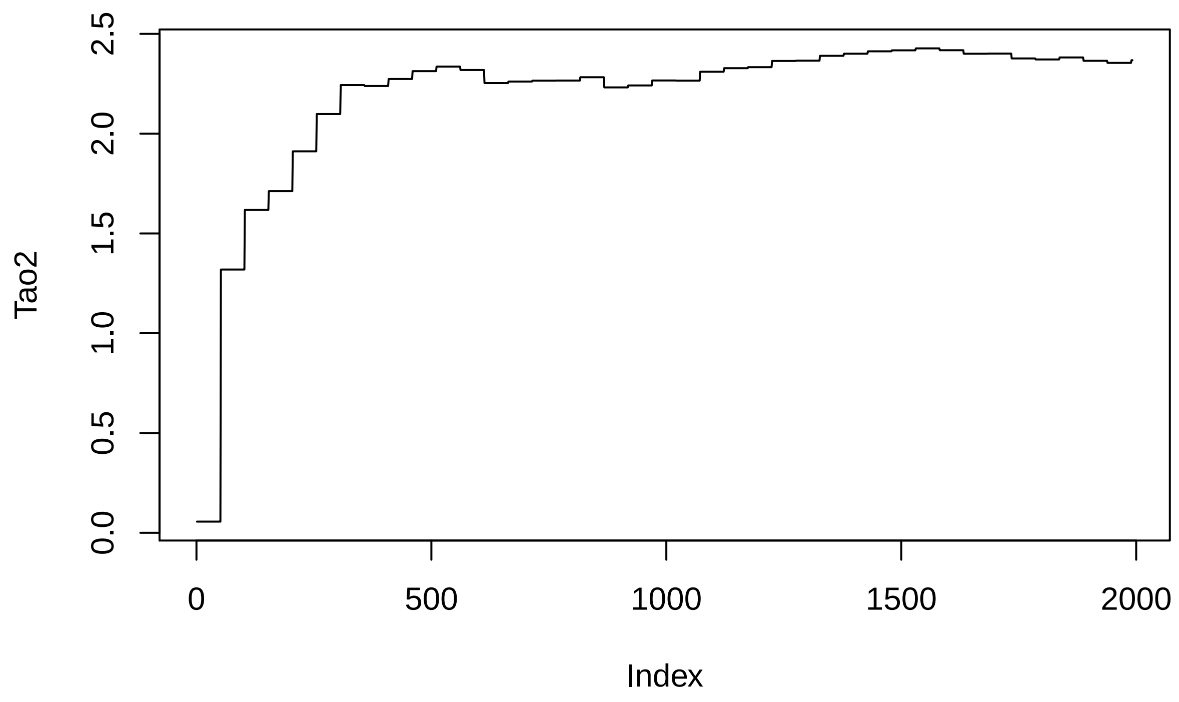
```r
Ex1_result <- do.call(getSequentialNonlinearModel, c(list(ifPrint=TRUE, testSize=50),Ex1, Ex1_algo))
```

Now Users might also want to plot the patameter convergence.

```r
plot(Ex1_result$gamma_opt,type = "l")
```

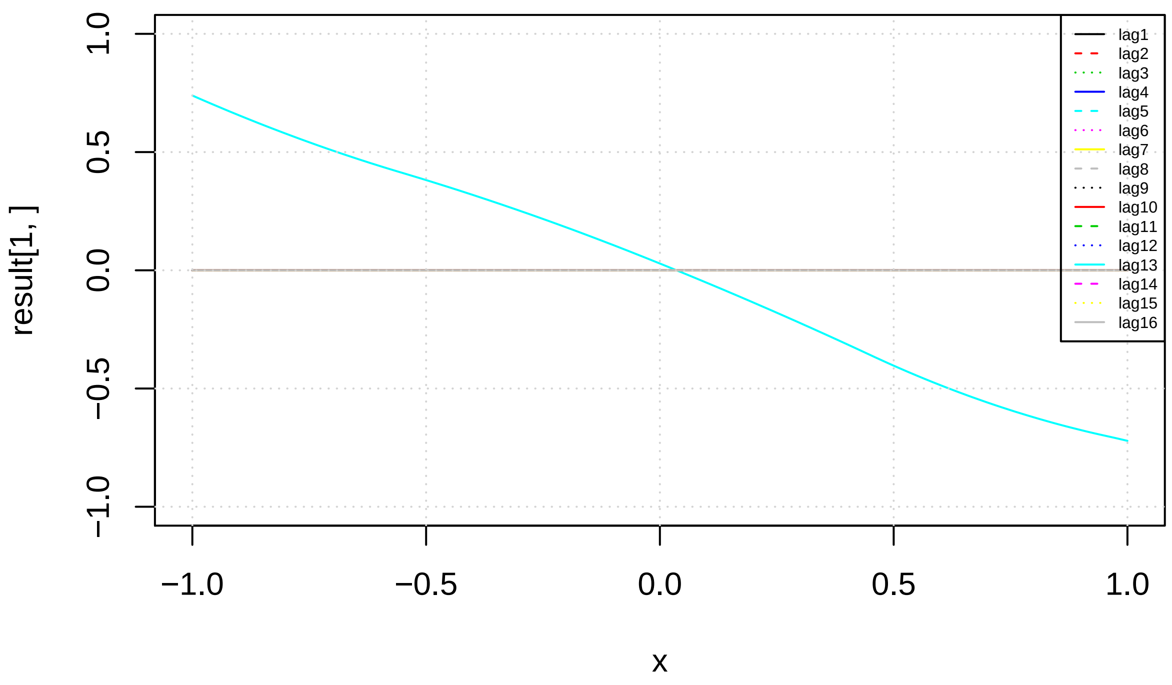


```r
plot(Ex1_result$alpha_opt,type = "l",ylab = "Tao2")
```



Plot the coefficient.

```
plotcoeff(Ex1_result$beta_opt,Ex1_algo$knots,Ex1_algo$nBspline)
```



result[1,]
x
−1.0
−0.5
0.0
0.5
1.0
lag1
lag2
lag3
lag4
lag5
lag6
lag7
lag8
lag9
lag10
lag11
lag12
lag13
lag14
lag15
lag16