Universidad de Santiago de Chile
Facultad de Ingeniería
Depto. de Ingeniería Informática

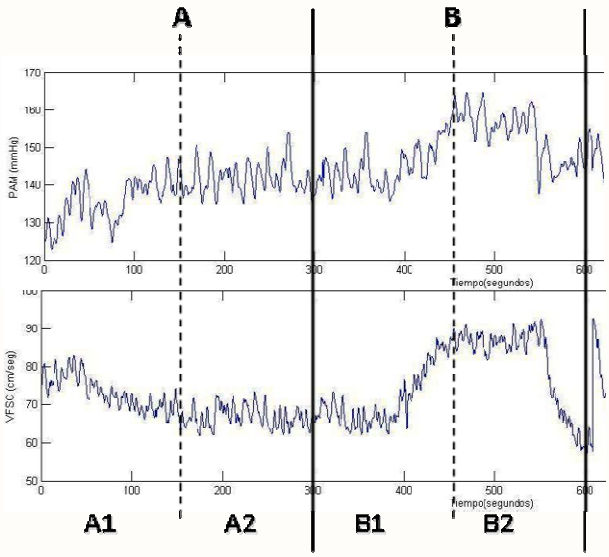
Taller de minería de datos avanzada
Capítulo VII
"SVR"

Profesor: Dr. Max Chacón.

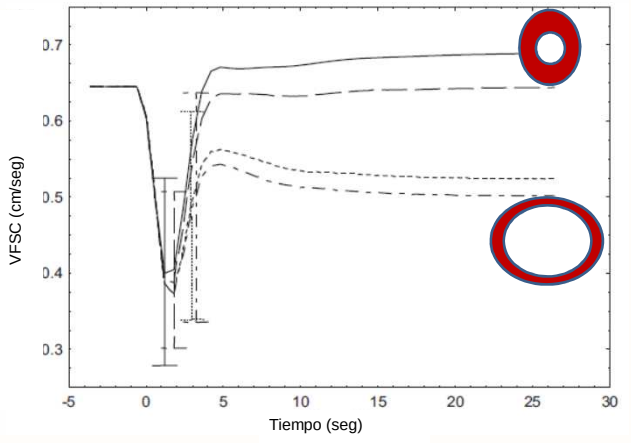
Contenidos

- Autorregulación cerebral en humanos.
- SVM en Regresión
- Modelos tipo FIR
- Paralelización de búsqueda de hyper parámetros

Autorregulación Dinámica

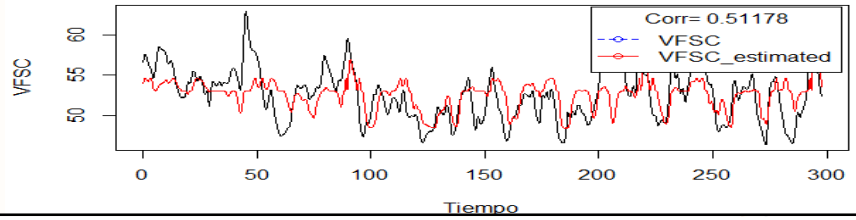


Autorregulación Dinámica



Regresión simple con SVM

- require(e1071)
- datos=read.csv("ANCO2000.csv")
- attach(datos)
- Ts=0.2
- Tiempo=seq(Ts,(length(VFSC))*Ts,Ts)
- formula=VFSC ~ PAM
- model <- svm(formula , datos)
- VFSC_estimated <- predict(model, PAM)
- eficiencia<-cor(VFSC_estimated,VFSC,method = "pearson")
- plot(Tiempo,VFSC,type="l")
- lines(Tiempo,VFSC_estimated, col = "red")
- legend("topright", c("VFSC","VFSC_estimated"), title = paste("Corr=",round(eficiencia,digits=5)), pch = 1, col=c("blue","red"),lty=c(2,1),inset = 0.01)

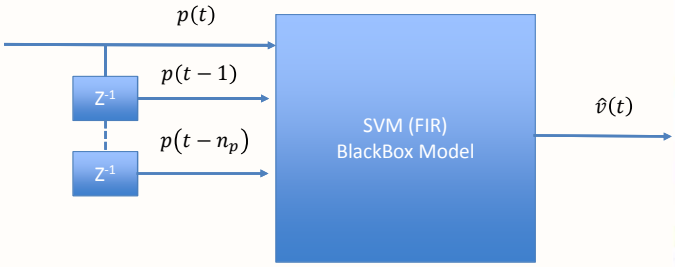


Tune en Regresión

- require(e1071)
- datos=read.csv("ANCO2000.csv")
- attach(datos)
- Ts=0.2
- Tiempo=seq(Ts,(length(VFSC))*Ts,Ts)
- formula=VFSC ~ PAM

- tuneResult <- tune(svm, formula, data = datos, ranges = list(nu = seq(0.1,0.1,0.1), cost = 2^(-4:-4), **type="nu-regression"**))
- tunedModel <- tuneResult$best.model
- VFSC_tunedModel <- predict(tunedModel, PAM)
- eficienciaTuned <- cor(VFSC_tunedModel, VFSC, method = "pearson")
- plot(Tiempo, VFSC, type="l")
- lines(Tiempo, VFSC_estimated, col = "red")
- lines(Tiempo, VFSC_tunedModel, col = "blue")
- legend("topright", c("VFSC",paste("VFSC_estimated corr=", round(eficiencia,5)), paste("VFSC_Tuned corr", round(eficienciaTuned,5))), title = "Correlacion", pch = 1, col=c("blue","red"),lty=c(2,1),inset = 0.01)

Modelos tipo FIR



For hyper parameter's configurations C, ν, Gamma and n_p a grid search is realized and using a **RBF** Kernel

Modelos tipo FIR

```
retardos_multi <- function( signalData, lags){
  signal.uni <- signalData
  max.lag <- max(unlist(lags)) + 1
  indices <- 1:nrow(signal.uni)
  lag.mat <- embed(indices, max.lag)
  col.names <- list("PAMn","VFSCn")
  columns <- NULL
  lagged.columns.names <- c()
  for(colname in col.names){
        lag.order <- lags[[colname]]
        columns[[colname]] <- signal.uni[lag.mat[, 1], colname]
        if(!is.null(lag.order) && lag.order > 0)
              for(i in 1:lag.order){
              new.colname <- paste(colname, paste0("lag", i), sep = ".")
              lagged.columns.names <- c(lagged.columns.names, new.colname)
              columns[[new.colname]] <- signal.uni[lag.mat[, i+1], colname]
              }
  }
  folded.signal <- data.frame(columns)
  sorting <- order(lag.mat[, 1])
  folded.signal <- folded.signal[sorting, ]
  list(folded.signal = folded.signal, lagged.columns.names = lagged.columns.names)
}
```

Paralelización en R

```
require(doParallel)
require(e1071)
registerDoParallel(cores = 7)
cost <- 2^seq(-4, 12, 2)
nu <- seq(0.1, 0.9, 0.4)
gamma<-2^seq(-4, 12, 2)
lagsList<-seq(1,5,1)
datos=read.csv("ANCO2000.csv")
PAMn<-(datos$PAM-min(datos$PAM))/(max(datos$PAM)-min(datos$PAM))
VFSCn<-(datos$VFSC-min(datos$VFSC))/(max(datos$VFSC)-min(datos$VFSC))
data <- data.frame(PAMn,VFSCn)
Ts=0.2
parms <- expand.grid(lagsList=lagsList, cost = cost, nu = nu, gamma=gamma)
```

Paralelización en R

```
salida <- (c( foreach(i = 1:nrow(parms),  combine = rbind, .inorder = FALSE)
%dopar% {
  c <- parms[i, ]$cost
  n <- parms[i, ]$nu
  g <- parms[i, ]$gamma
  l <- parms[i, ]$lagsList
  lag<-list(PAMn = l,VFSCn = 0)
  signal.train <- retardos_multi(data, lag)
  retDatos=signal.train$folded.signal
  x=subset(retDatos, select = -VFSCn)
  y=retDatos$VFSCn
  modelo <- e1071::svm(x, y, type = "nu-regression", kernel = "radial", cost =
 c, nu = n, gamma=g)
  pred <- predict(modelo, x)
  corr_pred<-cor(pred,y,method = "pearson")
  c(l, c, n, g, corr_pred)
}))
output <- matrix(unlist(salida), ncol = 5, byrow = TRUE)
mejoresModelos<-output[order(output[,5], decreasing = TRUE),]
print(mejoresModelos)
```

Paralelización en R

```
> print(mejoresModelos)
        [,1]      [,2] [,3]      [,4]      [,5]
 [1,]    5   64.0000  0.9 1024.0000 1.0000000
 [2,]    5 4096.0000  0.9 1024.0000 1.0000000
 [3,]    5   16.0000  0.9 1024.0000 1.0000000
 [4,]    5 1024.0000  0.9 1024.0000 1.0000000
 [5,]    4   64.0000  0.9 1024.0000 1.0000000
 [6,]    5  256.0000  0.9 1024.0000 1.0000000
 [7,]    4 4096.0000  0.9 1024.0000 1.0000000
 [8,]    4  256.0000  0.9 1024.0000 1.0000000
 [9,]    3 1024.0000  0.9 4096.0000 1.0000000
[10,]    3   64.0000  0.9 4096.0000 1.0000000
```

Selección de modelos y prueba fisiológica

- inverseStep=matrix(1,180/Ts,1)
- inverseStep[(90/Ts):(180/Ts),1]=0

- for (i in 1:length(mejoresModelos[,1])){

```
PAMn<-(datos$PAM-min(datos$PAM))/(max(datos$PAM)-min(datos$PAM))
VFSCn<-(datos$VFSC-min(datos$VFSC))/(max(datos$VFSC)-min(datos$VFSC))
data <- data.frame(PAMn,VFSCn)
lag<-list(PAMn = mejoresModelos[i,1],VFSCn = 0)
signal.train <- retardos_multi(data, lag)
retDatos=signal.train$folded.signal
x=subset(retDatos, select = -VFSCn)
y=retDatos$VFSCn
mejorModelo <- svm(x, y, kernel = "radial",type = "nu-regression", cost = mejoresModelos[i,2],
nu = mejoresModelos[i,3], gamma=mejoresModelos[i,4])
```

Selección de modelos y prueba fisiológica

```
PAMn=inverseStep
VFSCn=inverseStep
data <- data.frame(PAMn,VFSCn)
lag<-list(PAMn = mejoresModelos[i,1],VFSCn = 0)

signal.train <- retardos_multi(data, lag)
retDatos=signal.train$folded.signal
x=subset(retDatos, select = -VFSCn)
y=retDatos$VFSCn
stepTime=seq(Ts,(length(retDatos$PAMn))*Ts,Ts)

stepResponse <- predict(mejorModelo, x )

plot(stepTime,retDatos$PAMn,type="l", col="red")
lines(stepTime,stepResponse, col = "blue")
legend("topright", c("Escalon de presión", "respuesta al escalon"), title = "autorregulacion", pch =
1, col=c("red","blue"),lty=c(1,1),inset = 0.01)
print(paste("corr=",mejoresModelos[i,5]))
readline(prompt="Press [enter] to continue")
}
```

```
+ stepTime=seq(Ts,(length(retDatos$PAMn))*Ts,Ts)
+ stepResponse <- predict(mejorModelo, x )
+ plot(stepTime,retDatos$PAMn,type="l", col="red")
+ lines(stepTime,stepResponse, col = "blue")
+ legend("topright", c("Escalon de presión", "respuesta al escalon"),
=c("red","blue"),lty=c(1,1),inset = 0.01)
+ print(paste("corr=",mejoresModelos[i,5]))
+ readline(prompt="Press [enter] to continue")
+ }
[1] "corr= 0.999999977425459"
Press [enter] to continue
```