Study for SIR

Choi TaeYoung

2020-04-29

차원축소 분석 예시(SIR)

```
library(dr)
data(ais)
attach(ais)  # the Australian athletes data
#fit dimension reduction using sir
m1 <- dr(LBM~Wt+Ht+RCC+WCC, method="sir", nslices = 8)
summary(m1)
```

```
##
## Call:
## dr(formula = LBM ~ Wt + Ht + RCC + WCC, method = "sir", nslices = 8)
##
## Method:
## sir with 8 slices, n = 202.
##
## Slice Sizes:
## 25 25 25 25 27 27 30 18
##
## Estimated Basis Vectors for Central Subspace:
##         Dir1      Dir2    Dir3     Dir4
## Wt  -0.11412 -0.016473 -0.3759  0.01182
## Ht  -0.05112 -0.003916  0.6108 -0.04842
## RCC -0.99086  0.999513 -0.4675  0.12609
## WCC  0.05060 -0.026227  0.5167  0.99077
##
##                Dir1   Dir2    Dir3    Dir4
## Eigenvalues 0.8769 0.1592 0.04233 0.01313
## R^2(OLS|dr) 0.9987 0.9988 0.99997 1.00000
##
## Large-sample Marginal Dimension Tests:
##                 Stat df  p.value
## 0D vs >= 1D 220.484 28 0.000000
## 1D vs >= 2D  43.354 18 0.000713
## 2D vs >= 3D  11.201 10 0.342038
## 3D vs >= 4D   2.651  4 0.617748
```

```
# repeat, using save:

m2 <- update(m1,method="save")
summary(m2)
```

```
##
```

```
## Call:
## dr(formula = LBM ~ Wt + Ht + RCC + WCC, method = "save", nslices = 8)
##
## Method:
## save with 8 slices, n = 202.
##
## Slice Sizes:
## 25 25 25 25 27 27 30 18
##
## Estimated Basis Vectors for Central Subspace:
##          Dir1    Dir2     Dir3     Dir4
## Wt  -0.127116 -0.2458  0.01335 -0.03206
## Ht  -0.009194  0.4430  0.01266  0.03128
## RCC -0.986743 -0.6318 -0.99192  0.93635
## WCC  0.100479  0.5867 -0.12551 -0.34821
##
##               Dir1   Dir2   Dir3    Dir4
## Eigenvalues 0.8061 0.3525 0.1783 0.08903
## R^2(OLS|dr) 0.9931 1.0000 1.0000 1.00000
##
## Large-sample Marginal Dimension Tests:
##                Stat df(Nor) p.value(Nor) p.value(Gen)
## 0D vs >= 1D 144.020      70    4.742e-07     0.002279
## 1D vs >= 2D  57.687      42    5.410e-02     0.321950
## 2D vs >= 3D  24.982      21    2.480e-01     0.575637
## 3D vs >= 4D   5.669       7    5.789e-01     0.667594
```

```
# repeat, using phd:

m3 <- update(m2, method="phdres")
summary(m3)
```

```
##
## Call:
## dr(formula = LBM ~ Wt + Ht + RCC + WCC, method = "phdres", nslices = 8)
##
## Method:
## phdres, n = 202.
##
## Estimated Basis Vectors for Central Subspace:
##         Dir1     Dir2     Dir3    Dir4
## Wt   0.02246 -0.02482 -0.13048  0.1463
## Ht  -0.00981  0.19216  0.00483 -0.2390
## RCC -0.99955 -0.22063 -0.60190  0.9164
## WCC  0.01705  0.95592  0.78783  0.2859
##
##                 Dir1   Dir2   Dir3    Dir4
## Eigenvalues -1.51312 1.4274 1.1651 -0.4919
## R^2(OLS|dr)  0.00701 0.3283 0.9613  1.0000
##
## Large-sample Marginal Dimension Tests:
##                Stat df Normal theory Indep. test General theory
## 0D vs >= 1D 42.341 10     6.521e-06    0.001055        0.02629
## 1D vs >= 2D 25.984  6     2.242e-04          NA        0.01689
## 2D vs >= 3D 11.427  3     9.626e-03          NA        0.05027
```

```
## 3D vs >= 4D  1.729  1     1.886e-01           NA         0.16545
```

```
summary(s0 <- dr(LBM~log(SSF)+log(Wt)+log(Hg)+log(Ht)+log(WCC)+log(RCC)+log(Hc)+log(Ferr), data=ais, sli
```

```
##
## Call:
## dr(formula = LBM ~ log(SSF) + log(Wt) + log(Hg) + log(Ht) + log(WCC) +
##     log(RCC) + log(Hc) + log(Ferr), data = ais, slice.function = dr.slices.arc,
##     nslices = 8, chi2approx = "wood", numdir = 4, method = "sir")
##
## Method:
## sir with 8 slices, n = 202.
##
## Slice Sizes:
## 26 26 25 25 25 27 30 18
##
## Estimated Basis Vectors for Central Subspace:
##                Dir1      Dir2     Dir3      Dir4
## log(SSF)   0.158016 -0.075965  0.15503 -0.042384
## log(Wt)   -0.970701 -0.022829 -0.24362  0.258583
## log(Hg)   -0.139764  0.346539  0.54270 -0.008597
## log(Ht)   -0.087587 -0.331604  0.30867 -0.630151
## log(WCC)   0.006682 -0.014914 -0.00581 -0.024479
## log(RCC)  -0.010892  0.502020  0.71198  0.343766
## log(Hc)    0.073437 -0.715120 -0.07453 -0.643773
## log(Ferr) -0.003117  0.003869 -0.11969 -0.030918
##
##               Dir1   Dir2   Dir3    Dir4
## Eigenvalues 0.9380 0.2046 0.0929 0.06665
## R^2(OLS|dr) 0.9987 0.9988 0.9988 0.99898
##
## Large-sample Marginal Dimension Tests:
##               Stat df   p.value
## 0D vs >= 1D 269.50 56 0.0000000
## 1D vs >= 2D  80.02 42 0.0003665
## 2D vs >= 3D  38.69 30 0.1327694
## 3D vs >= 4D  19.93 20 0.4624789
```

```
## 카테고리 그룹을 포함(pool 옵션의 디폴트값이 FALSE이다. 뜻은 합동분산 추정관련)
summary(s1 <- update(s0, group=~Sex))
```

```
##
## Call:
## dr(formula = LBM ~ log(SSF) + log(Wt) + log(Hg) + log(Ht) + log(WCC) +
##     log(RCC) + log(Hc) + log(Ferr), data = ais, group = ~Sex,
##     slice.function = dr.slices.arc, nslices = 8, chi2approx = "wood",
##     numdir = 4, method = "sir")
##
## Method:
## psir with 8 8 slices, n = 202.
##
## Slice Sizes:
## 13 13 13 13 12 12 12 12 13 17 14 16 13 14 12 3
##
## Estimated Basis Vectors for Central Subspace:
```

```
##                Dir1     Dir2     Dir3      Dir4
## log(SSF)   0.117498  0.04262 -0.09604  0.093277
## log(Wt)   -0.954097 -0.16172  0.09756 -0.095106
## log(Hg)   -0.107177 -0.52877 -0.90198  0.018578
## log(Ht)   -0.075489  0.51072  0.31713 -0.050511
## log(WCC)   0.036513  0.04380  0.12594 -0.053962
## log(RCC)  -0.126238 -0.08432 -0.04154 -0.692498
## log(Hc)    0.203441  0.64997 -0.18122  0.704890
## log(Ferr)  0.007086 -0.01077  0.12924  0.004844
##
##               Dir1   Dir2   Dir3   Dir4
## Eigenvalues 1.8335 0.4492 0.3172 0.2030
## R^2(OLS|dr) 0.9904 0.9922 0.9930 0.9952
##
## Large-sample Marginal Dimension Tests:
##              Stat  df   p.value
## 0D vs >= 1D 621.8 112 0.000e+00
## 1D vs >= 2D 251.4  91 1.110e-16
## 2D vs >= 3D 160.7  72 1.020e-08
## 3D vs >= 4D  96.6  55 4.509e-04
```

```
data(we8there)

dim(we8thereCounts)
```

```
## [1] 6166 2640
```

```
dim(we8thereRatings)
```

```
## [1] 6166    5
```

```
as.matrix(we8thereCounts)[12,400] ## count for bigram 400 in review 12
```

```
## [1] 0
```

```
##12th review에서 bigram >0 뽑아내기
mm <- as.matrix(we8thereCounts)[12]
which(mm>0)
```

```
## integer(0)
```

```
## get to know what's in the matrix
g1 <- min(as.matrix(we8thereCounts)[,]) ## min count over reviews/bigrams
g2 <- max(as.matrix(we8thereCounts)[,]) ## max count over reviews/bigrams
g1
```

```
## [1] 0
```

```
g2 ## a certain bigram was mentioned in a certain review 13 times
```

```
## [1] 13
```

```
## here we look at the frequencies of the bigram in column 1000
## the data are extremely sparce
hh <- as.matrix(we8thereCounts)[,1000]


## overall rating
overall <- as.matrix(we8thereRatings[,1:5])
```

```r
summary(we8thereRatings[,1:5])
```

```
##       Food          Service          Value         Atmosphere
##  Min.   :1.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:4.000   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:3.000
##  Median :5.000   Median :4.000   Median :4.000   Median :4.000
##  Mean   :4.093   Mean   :3.897   Mean   :3.898   Mean   :3.951
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
##     Overall
##  Min.   :1.000
##  1st Qu.:3.000
##  Median :5.000
##  Mean   :3.945
##  3rd Qu.:5.000
##  Max.   :5.000
```

```r
## Multinomial logistic regression and fitted reduction
## we8mnlm=mnlm(we8thereCounts,overall,bins=5)
## bins: for faster inference if covariates are factors
## covariate is a factor with 5 levels
cl <- NULL
we8mnlm <- mnlm(cl,covars=overall,counts=we8thereCounts,bins=5)
```

```
## Warning in (function (x, y, family = c("gaussian", "binomial", "poisson"), :
## numerically perfect fit for some observations.
```

```
## Warning in (function (x, y, family = c("gaussian", "binomial", "poisson"), :
## numerically perfect fit for some observations.
```

```r
## 여기 mnlm으로 얻는 값들의 구조가 뭔가? 우리가 흔히 하는 coefficient 추정값들

## we8mnlm$intercept
## estimates of alphas
## we8mnlm$loadings
## estimates of betas
# fitted(we8mnlm)
# as.matrix(fitted(we8mnlm))[1,]
## fitted counts for first review
## extract coefficients
B <- coef(we8mnlm) ##2 x 2640 sparse Matrix
B[1,1:10] ## estimates of alpha
```

```
##      veri good        go back      dine room    dine experi     great food
##      -5.169880      -4.697690      -6.037884      -5.555694      -9.723688
##     food great    realli good      ice cream high recommend    great place
##      -8.734836      -6.099053      -6.570905     -10.226808      -8.008069
```

```r
B[2,1:10] ## estimates of beta
```

```
##      veri good        go back      dine room    dine experi     great food
##      0.1530334      0.0000000      0.1228750      0.0000000      0.3949753
##     food great    realli good      ice cream high recommend    great place
##      0.4960973      0.1160693      0.0000000      0.2574008      0.0000000
```

```r
mean(B[2,]==0) ## sparsity in loadings
```

```
## [1] 0.7469697
```

```
## some big loadings in IR
order(B[2,][1:5]) #2 3 4 1 5(제일 작은 값이 2번째 있고 제일 큰 값이 5번째에 있음)
```

```
## [1] 2 4 3 1 5
```

```
B[2,order(-B[2,])[1:10]] #제일 큰 coef 1~10번째
```

```
##     cannot wait      food fabul     food superb outstand servic    around world
##        3.571854        2.739032        2.723090        2.670907        2.595806
##      best sushi   francisco bay     food awesom     mouth water     kept secret
##        2.448667        2.177312        2.002595        1.778145        1.764538
```

```
## following provides fitted multinomial probabilities
pred <- predict(we8mnlm,overall,type="response")
pred_1 <- pred[1,] ## predicted multinomial probs for review 1
sum(pred[1,]) ## must add to one
```

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

```
## following predicts inverse prediction (fitted reduction)
## predinv=predict(we8mnlm,we8thereCounts,type="reduction")
predinve <- srproj(B,we8thereCounts) #MNIR projection onto factors


## 여기 srproj 함수를 뜯어내서

predinv=predinve[,1]
predinv[1:10] ## prints predicted ratings for first 10 reviews
```

```
##           1           2           5          11          12          13
##  0.402855446  0.000000000  0.000000000  0.032002746  0.446481751  0.035465620
##          14          15          17          18
## -0.166802810  0.000197994  0.000000000  0.000000000
```

```
# NOT RUN {
library(MASS)
data(fgl)

## make your cluster
## FORK is faster but memory heavy, and doesn't work on windows.
cl <- makeCluster(2,type=ifelse(.Platform$OS.type=="unix","FORK","PSOCK"))
print(cl)
```

```
## socket cluster with 2 nodes on host 'localhost'
```

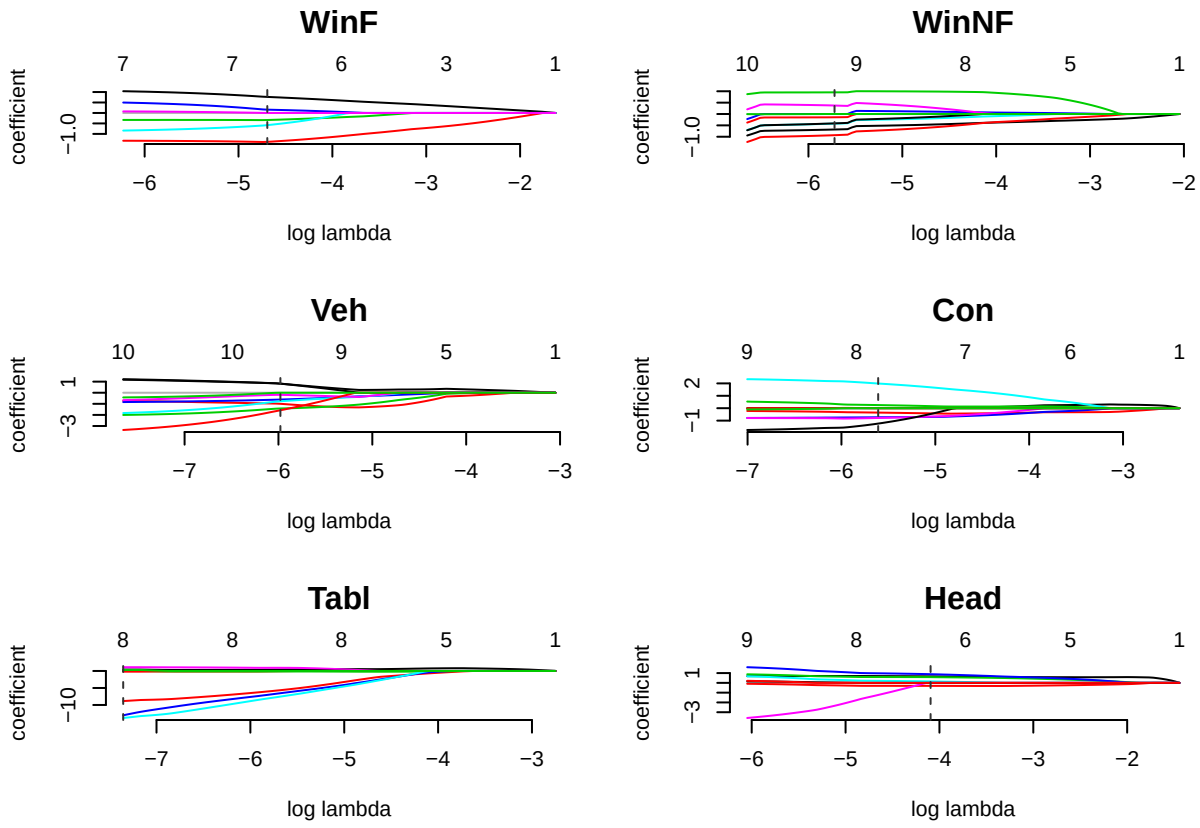
```
## fit in parallel
fits <- dmr(cl, fgl[,1:9], fgl$type, verb=1)
```

```
## fitting 214 observations on 6 categories, 9 covariates.
## converting counts matrix to column list...
## distributed run.
## socket cluster with 2 nodes on host 'localhost'
```

```
## 첫함수 <- cl이 군집들
## 둘째 <- fgl들이 값들(상관계수?(covariate))
## 셋째 <- 타입(여기선 2개)
```

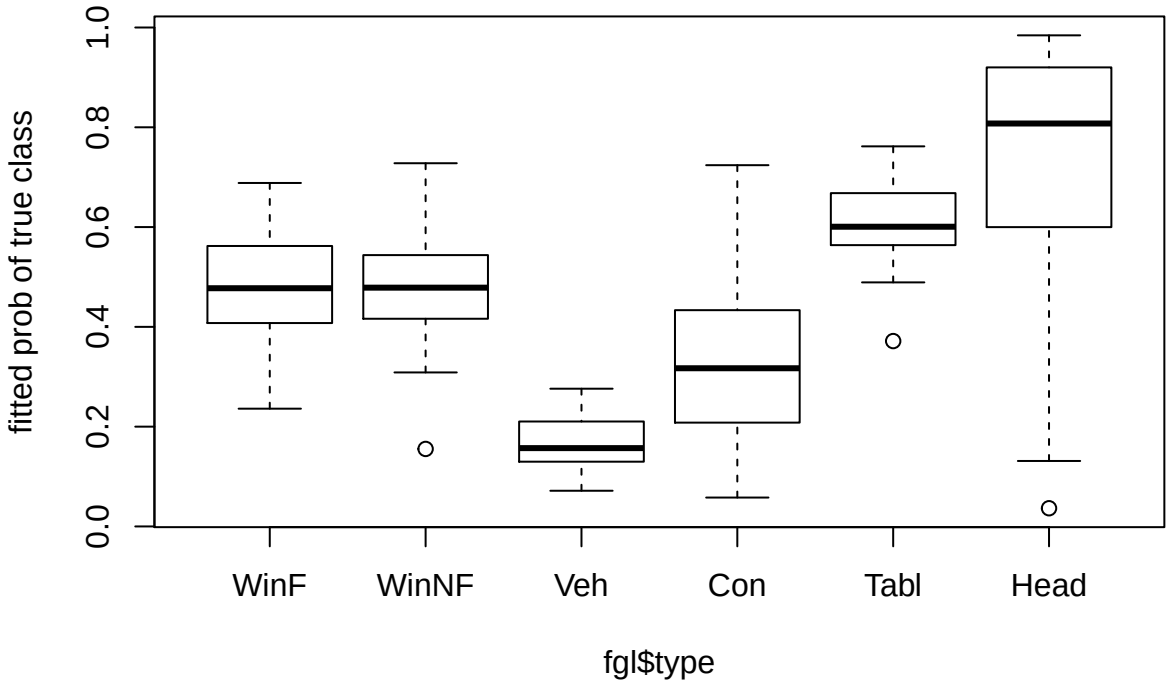
```
## its good practice stop the cluster once you're done
stopCluster(cl)

## Individual Poisson model fits and AICc selection
par(mfrow=c(3,2))
for(j in 1:6){
    plot(fits[[j]])
    mtext(names(fits)[j],font=2,line=2) }
```




```
##  AICc model selection
B <- coef(fits)

## Fitted probability by true response
par(mfrow=c(1,1))
P <- predict(B, fgl[,1:9], type="response")
boxplot(P[cbind(1:214,fgl$type)]~fgl$type,
    ylab="fitted prob of true class")
```




```
# }
```

SIR 적용

- x_i가 텍스트로 되어 있는 문서 -> sparse matrix
- 종속변수 y_i를 예측 -> $v_i(=y_i)$에서 Φ라는 계수(coef)를 추정.
- 여기서 i는 관측값의 수
- 로지스틱 회귀를 사용가능(역 조건부 분포 활용)
- y_i에 대한 정보를 보존 -> 저차원 문서 점수(SR score)를 얻는 역회귀 제안

$$x_i \sim MN(m_i, q_i) \text{ with } q_{ij} = \frac{\exp(\eta_{ij})}{\sum_{i=1^p} \exp(\eta_{ij})},\ j = 1, \dots, p$$

- where $\eta_{ij} = \alpha_j + u_{ij} + v_i^T \varphi_j$.
- 여기서 $\alpha_j \sim N(0,1)$라고 가정한다.
- 여기서 φ_{jk}는 독립 라플라스 사전분포를 갖는다. 즉, $\pi(\varphi_{jk}) = \lambda_{jk}/2 \exp(-\lambda_{jk}|\varphi_{jk}|)$ for $j = 1, \dots, p$ and $k = 1, \dots, K$
- 여기서 내포된 φ_{jk}의 사전 표준편차는 $\sqrt{2}/\lambda_{jk}$
- 여기서 각 λ_{jk}에 할당된 공액 감마 초모수의 분포는 $\text{Gamma}(\lambda_{jk}; s, r) = r^s/\Gamma(s)\lambda_{jk}^{s-1}e^{-r\lambda_{jk}}$

- 여기서 s은 shape parameter, r은 rate parameter, 평균은 s/r, 분산은 s/r^2 이다.
- 결국 우리가 관심있는 사후 분포는 아래와 같다.

$$p(\boxtimes, \boxtimes, \boxtimes, \mathsf{U}|\mathsf{X}, \mathsf{V}) \propto \prod_{i=1}^{n}\prod_{j=0}^{p} q_{ij}^{x_{ij}}\pi(u_{ij})N(\alpha_j; 0, \sigma_\alpha^2)\prod_{k=1}^{K}\mathsf{GL}(\varphi_{jk}, \lambda_{jk})$$

- 여기서 감마-라쏘의 사후분포를 $c(\varphi_{jk}) = s\log(1 + |\varphi_{jk}|/r)$로 나타낼 수 있다.
- 이 모든 것을 로그를 취하고, 음수를 취한 뒤, 상수항을 제거한 식은 $l(\alpha_j, \varphi_j) + \sum_{j=1}^{p}(\alpha_j/\sigma_\alpha)^2 + c(\Phi)$인 이 식을 최소화하는 $\hat{\alpha}, \hat{\varphi}$를 찾으면 된다.
- 여기서 $l(\alpha_j, \varphi_j) = \sum_{i=1}^{n}[m_i e^{\alpha_j + v_i^T\varphi_j} - x_{ij}(\alpha_j + v_i^T\varphi_j)]$

```
x.i <- as.matrix(we8thereCounts)
v.i <- as.matrix(we8thereRatings)
```

1. SIR coefficient들 찾기(알파, 베타?) ex) mnir에서는 6x2640 차원 스파스 매트릭스 만들어짐
2. 그 값들을 메트릭스로 겹치는 차원으로 만들기 ex) 19개 컬럼이름
3. x corv 메트릭스 생성 -> 충분차원 축소 상황 발생
4. SR 스코어?로 해석하는 건지