Statistical Learning

Lecture 06a

ANU - RSFAS

Last Updated: Fri Apr 1 08:35:04 2022

Dimension Reduction Methods

- The methods that we have discussed have involved fitting linear regression models, via least squares or a shrunken approach, using the original predictors, $X_1, X_2, \ldots, X_p$.
- We now explore a class of approaches that:
  - transform the predictors
  - then fit a least squares model using the transformed variables.
- We will refer to these techniques as dimension reduction methods.

Dimension Reduction Methods

- Let $Z_1, \ldots, Z_M$ represent $M < p$ linear combinations of our original p predictors.

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

for some constants $\phi_{m1}, \phi_{m2}, \ldots, \phi_{mp}$.

- We can then fit the linear regression model using OLS:

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i$$

- If the constants $\phi_{m1}, \phi_{m2}, \ldots, \phi_{mp}$ are chosen wisely, then such dimension reduction approaches can often outperform standard regression on the original covariates.

- Notice

$$\begin{aligned}\sum_{m=1}^{M} \theta_m z_{im} &= \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{mj} x_{ij} \\ &= \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{mj} x_{ij} \\ &= \sum_{j=1}^{p} \beta_j x_{ij}\end{aligned}$$

- So $\beta_j = \sum_{m=1}^{M} \theta_m \phi_{mj}$.

- The dimension reduction model can be thought of as a special case of the original linear regression model.

- Dimension reduction serves to constrain the estimated β_j coefficients.

- This may lead to a win in the bias-variance trade-off.

Principal Components Regression

- Here we apply principal components analysis (PCA) to define the linear combinations of the predictors, for use in our regression.
- The first principal component is that (normalized) linear combination of the variables with the largest variance.
- The second principal component has largest variance, subject to being uncorrelated with the first.
- And so on . . .
- Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation.

First Principle Component

- The first principal component of a set of covariates $X_1, X_2, \ldots, X_P$ is the normalized linear combination of the features:

$$Z_1 = \phi_{11} X_1 + \phi_{12} X_2 + \cdots + \phi_{1p} X_p$$

  that has the largest variance.

- Normalized means that $\sum_{j=1}^{p} \phi_{1j}^2 = 1$

- $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ are called the loadings.

- We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

Computation

- Suppose we have a $n \times p$ set of covariates X.
- We assume that each of the variables in X has been centered to have mean zero (that is, the column means of X are zero).
- Since each x_{ij} has mean zero, then so does z_{i1}.

$$z_{i1} = \phi_{11}x_{i1} + \phi_{12}x_{i2} + \cdots + \phi_{1p}x_{ip}$$

- This leads to:

$$\begin{aligned} Var(z_{i1}) &= E(z_{i1}^2) - E(z_{i1})^2 \\ &= E(z_{i1}^2) - 0 \\ &= E(z_{i1}^2) \end{aligned}$$

- So the sample variance can be written as:

$$\frac{1}{n}\sum_{i=1}^{n} z_{i1}^2$$

- The first principal component loading vector solves the optimization problem

$$\underset{\phi_{11},\phi_{12},\ldots,\phi_{1p}}{\text{maximize}} \quad \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{1j}x_{ij}\right)^2 \quad \text{subject to } \sum_{j=1}^{p}\phi_{1j}^2 = 1$$

- This problem can be solved via a singular-value decomposition of the matrix X. This approach also provides the other principle components.

Proportion Variance Explained

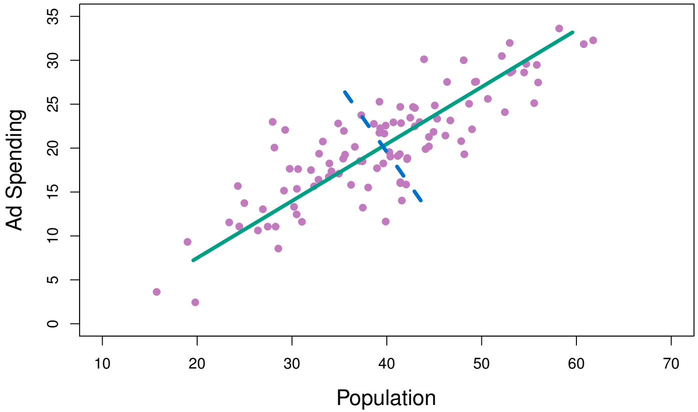
- To understand the strength of each principle component, we are interested in knowing the proportion of variance explained (PVE) by each one.
- The variance explained by the m^{th} principal component is

$$V(Z_m) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2$$

- Therefore, the proportion of variance explained (PVE) by the m^{th} principal component is:

$$\frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{M} \sum_{i=1}^{n} z_{im}^2}$$

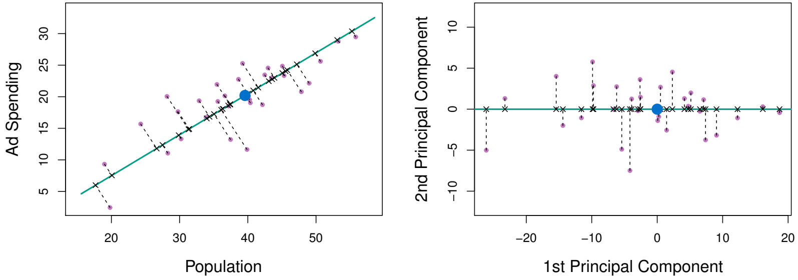
- This quantity is positive and between 0 and 1.



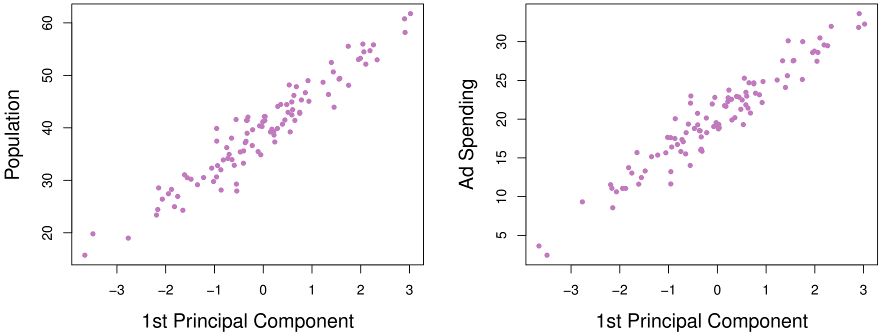
- The **population size** (pop) and **ad spending** (ad) for 100 different cities are shown as purple circles.
- The **green** solid line indicates the first principal component, and the **blue** dashed line indicates the second principal component.

- For a particular case in our data set we have:

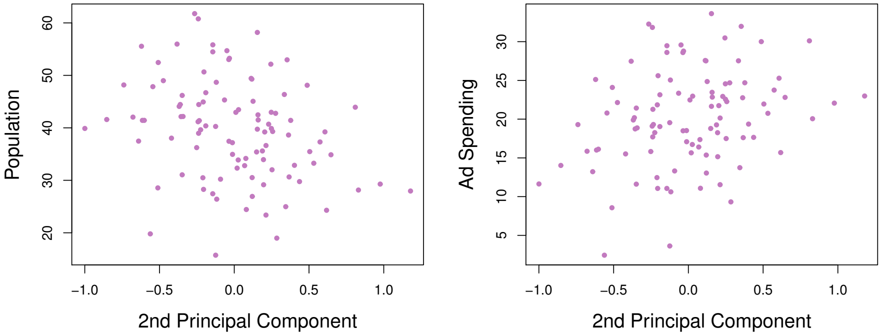
$$z_{i1} = 0.839 \times (\mathrm{pop}_i - \bar{\mathrm{pop}}) + 0.544 \times (\mathrm{ad}_i - \bar{\mathrm{ad}})$$



- Left: The first principal component, chosen to minimize the sum of the squared perpendicular distances to each point, is shown in green.
- Right: Showing that the first and second principle components are orthogonal.



- The relationships are strong.



- The relationships are weak.

Credit Card Balance Data

```r
bal <- read.csv("Credit.csv", header=TRUE)[,-1]

summary(bal)
```

```
##      Income           Limit           Rating          Cards      
##  Min.   : 10.35   Min.   :  855   Min.   : 93.0   Min.   :1.000  
##  1st Qu.: 21.01   1st Qu.: 3088   1st Qu.:247.2   1st Qu.:2.000  
##  Median : 33.12   Median : 4622   Median :344.0   Median :3.000  
##  Mean   : 45.22   Mean   : 4736   Mean   :354.9   Mean   :2.958  
##  3rd Qu.: 57.47   3rd Qu.: 5873   3rd Qu.:437.2   3rd Qu.:4.000  
##  Max.   :186.63   Max.   :13913   Max.   :982.0   Max.   :9.000  
##       Age          Education        Gender            Student         
##  Min.   :23.00   Min.   : 5.00   Length:400         Length:400        
##  1st Qu.:41.75   1st Qu.:11.00   Class :character   Class :character  
##  Median :56.00   Median :14.00   Mode  :character   Mode  :character  
##  Mean   :55.67   Mean   :13.45                                        
##  3rd Qu.:70.00   3rd Qu.:16.00                                        
##  Max.   :98.00   Max.   :20.00                                        
##    Married           Ethnicity            Balance       
##  Length:400         Length:400         Min.   :   0.00  
##  Class :character   Class :character   1st Qu.:  68.75  
##  Mode  :character   Mode  :character   Median : 459.50  
##                                        Mean   : 520.01  
##                                        3rd Qu.: 863.00  
##                                        Max.   :1999.00  
```

```r
library(pls)
set.seed(2)
pcr.fit <- pcr(Balance ~ ., data=bal, scale=TRUE,
               validation="CV")
```

```r
summary(pcr.fit)
```

```
## Data:    X dimension: 400 11
##  Y dimension: 400 1
## Fit method: svdpc
## Number of components considered: 11
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           460.3    300.3    299.2    293.7    292.2    291.8    281.2
## adjCV        460.3    300.0    298.9    290.8    292.3    293.0    285.0
##        7 comps  8 comps  9 comps  10 comps  11 comps
## CV       264.2    264.4    266.0     100.2    100.11
## adjCV    263.6    264.0    265.8     100.1     99.97
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          25.05    39.64    49.73    59.74    68.89    77.73    86.43    93.91
## Balance    58.07    58.37    60.78    60.90    61.46    63.11    68.70    68.71
##          9 comps  10 comps  11 comps
## X          97.60     99.98    100.00
## Balance    68.72     95.47     95.51
```

```r
summary(lm(Balance~., data=bal))
```

```
##
## Call:
## lm(formula = Balance ~ ., data = bal)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -161.64  -77.70  -13.49   53.98  318.20
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -489.86112   35.80118 -13.683  < 2e-16 ***
## Income               -7.80310    0.23423 -33.314  < 2e-16 ***
## Limit                 0.19091    0.03278   5.824 1.21e-08 ***
## Rating                1.13653    0.49089   2.315   0.0211 *
## Cards                17.72448    4.34103   4.083 5.40e-05 ***
## Age                  -0.61391    0.29399  -2.088   0.0374 *
## Education            -1.09886    1.59795  -0.688   0.4921
## GenderMale           10.65325    9.91400   1.075   0.2832
## StudentYes          425.74736   16.72258  25.459  < 2e-16 ***
## MarriedYes           -8.53390   10.36287  -0.824   0.4107
## EthnicityAsian       16.80418   14.11906   1.190   0.2347
## EthnicityCaucasian   10.10703   12.20992   0.828   0.4083
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 98.79 on 388 degrees of freedom
## Multiple R-squared:  0.9551, Adjusted R-squared:  0.9538
## F-statistic: 750.3 on 11 and 388 DF,  p-value: < 2.2e-16
```

```r
pcr.fit$loadings
```

```
##
## Loadings:
##                    Comp 1 Comp 2 Comp 3 Comp 4 Comp 5 Comp 6 Comp 7 Comp 8
## Income             -0.542
## Limit              -0.586
## Rating             -0.587
## Cards                             0.479 -0.272        -0.283  0.771 -0.109
## Age                -0.123         0.107 -0.479 -0.295 -0.584 -0.359  0.413
## Education                        -0.475  0.199 -0.583 -0.402  0.216 -0.418
## GenderMale                        0.334        -0.746  0.514  0.102  0.227
## StudentYes                 0.125 -0.619 -0.296         0.202  0.428  0.534
## MarriedYes                        0.126  0.739        -0.324  0.136  0.537
## EthnicityAsian             0.697  0.106
## EthnicityCaucasian        -0.687 -0.100  0.134                0.103
##                    Comp 9 Comp 10 Comp 11
## Income                     0.836
## Limit                     -0.379   0.705
## Rating                    -0.374  -0.708
## Cards
## Age                       -0.103
## Education
## GenderMale
## StudentYes
## MarriedYes          0.119
## EthnicityAsian     -0.707
## EthnicityCaucasian -0.695
##
##                Comp 1 Comp 2 Comp 3 Comp 4 Comp 5 Comp 6 Comp 7 Comp 8 Comp 9
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.091  0.091  0.091  0.091  0.091  0.091  0.091  0.091  0.091
## Cumulative Var  0.091  0.182  0.273  0.364  0.455  0.545  0.636  0.727  0.818
##                Comp 10 Comp 11
## SS loadings      1.000   1.000
## Proportion Var   0.091   0.091
## Cumulative Var   0.909   1.000
```

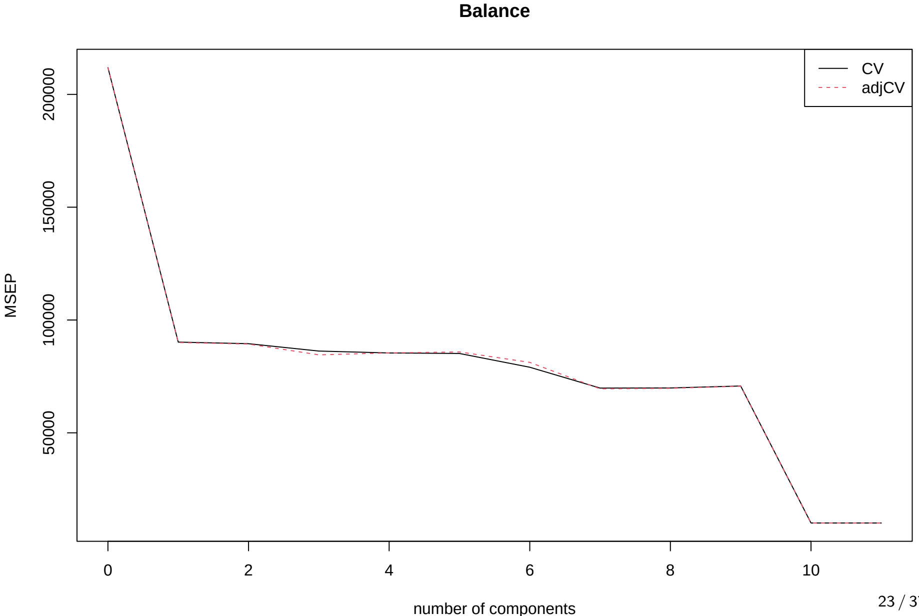
```r
z1 <- pcr.fit$loadings[,1]
sum(z1^2)
```

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

Choosing M

```r
plot(pcr.fit, "validation", val.type = "MSEP",
     legendpos = "topright")
```

Choosing M


Balance
CV
adjCV
MSEP
200000
150000
100000
50000
0
2
4
6
8
10
number of components

- Adjusted MSE attempts to correct for the fact that the estimates are "trained" on subsets of the data and not the whole data set, which can overestimate the MSE.
- See the following paper on Wattle for more information:

"Mean squared error of prediction (MSEP) estimates for principal component regression (PCR) and partial least squares regression (PLSR)" (2004); Bjorn-Helge Mevik and Henrik Rene Cederkvist.

Diabetes Data

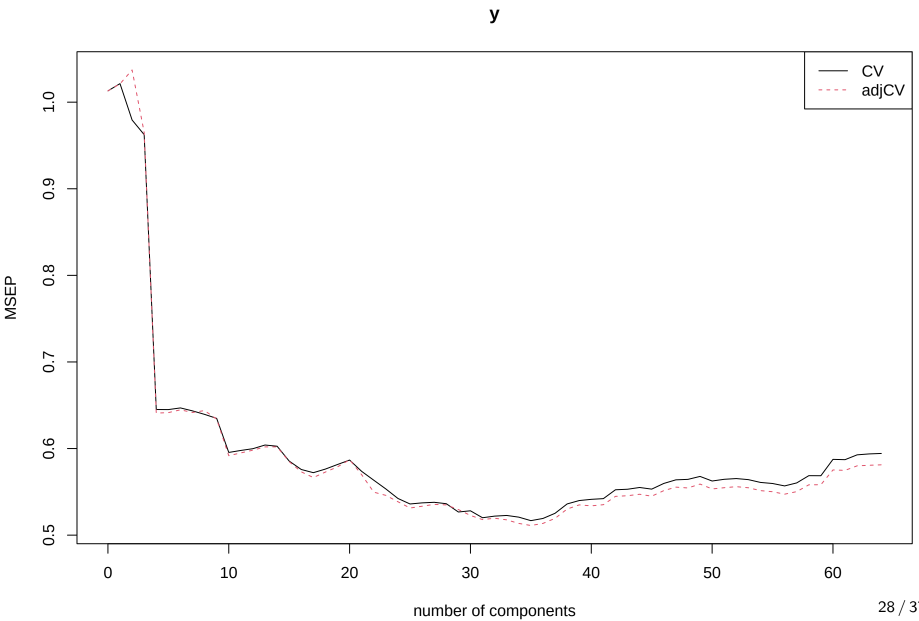
```r
diabTrain <- read.table("diabTrain.dat", header=T)
diabTest <- read.table("diabTest.dat", header=T)

X <- as.matrix(diabTrain[,-1])
y <- diabTrain[,1]

X.test <- as.matrix(diabTest[,-1])
y.test <- diabTest[,1]
```

```r
set.seed(2)
pcr.fit <- pcr(y ~ X, scale=FALSE,
               validation="CV")
```

```r
summary(pcr.fit)
```

```
## Data:    X dimension: 342 64
##  Y dimension: 342 1
## Fit method: svdpc
## Number of components considered: 64
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.006    1.011   0.9897   0.9811   0.8032   0.8031   0.8043
## adjCV        1.006    1.011   1.0184   0.9824   0.8006   0.8009   0.8029
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.8022   0.7997   0.7968    0.7717    0.7732    0.7745    0.7772
## adjCV   0.8010   0.8024   0.7962    0.7693    0.7713    0.7734    0.7758
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
## CV       0.7763    0.7652    0.7588    0.7564    0.7592    0.7626    0.7660
## adjCV    0.7759    0.7648    0.7570    0.7527    0.7569    0.7608    0.7662
##        21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
## CV       0.7574    0.7506    0.7439    0.7364    0.7321    0.7330    0.7334
## adjCV    0.7548    0.7414    0.7389    0.7339    0.7289    0.7303    0.7318
##        28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
## CV       0.7323    0.7258    0.7267    0.7212    0.7225    0.7230    0.7216
## adjCV    0.7314    0.7277    0.7230    0.7197    0.7208    0.7195    0.7166
##        35 comps  36 comps  37 comps  38 comps  39 comps  40 comps  41 comps
## CV       0.7189    0.7206    0.7247    0.7321    0.7348    0.7358    0.7363
## adjCV    0.7149    0.7166    0.7207    0.7281    0.7314    0.7306    0.7315
##        42 comps  43 comps  44 comps  45 comps  46 comps  47 comps  48 comps
## CV       0.7432    0.7436    0.7450    0.7437    0.7481    0.7509    0.7513
## adjCV    0.7381    0.7386    0.7397    0.7382    0.7426    0.7453    0.7446
##        49 comps  50 comps  51 comps  52 comps  53 comps  54 comps  55 comps
## CV       0.7535    0.7500    0.7513    0.7519    0.7511    0.7489    0.7481
## adjCV    0.7476    0.7439    0.7448    0.7456    0.7448    0.7426    0.7417
##        56 comps  57 comps  58 comps  59 comps  60 comps  61 comps  62 comps
## CV       0.7462    0.7485    0.7541    0.7541    0.7665    0.7663    0.7699
## adjCV    0.7398    0.7418    0.7471    0.7471    0.7585    0.7582    0.7616
##        63 comps  64 comps
```

```r
plot(pcr.fit, "validation", val.type = "MSEP",
     legendpos = "topright")
```



- The lowest MSE occurs at $m = 30$. Not shown in the table but this explains 92.55% of X.

```r
pcr.pred <- predict(pcr.fit, X.test, ncomp=30)
mean((pcr.pred-y.test)^2)
```

```
## [1] 0.5708273
```

Thoughts

- PCR identifies linear combinations, or directions, that best represent the predictors $X_1, \ldots, X_p$.
- These directions are identified in an unsupervised way, since the response Y is not used to help determine the principal component directions.
- That is, the response does not supervise the identification of the principal components.
- PCR suffers from a potentially serious drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

Partial Least Squares

- PLS is a dimension reduction method, which first identifies a new set of features $Z_1, \ldots, Z_M$ that are linear combinations of the original features, and then fits a linear model via OLS using these M new features.
- PLS identifies these new features in a supervised way:
  - It makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response.
- The PLS approach attempts to find directions that help explain both the response and the predictors.

Details of Partial Least Squares

- After standardizing the p predictors, PLS computes the first direction Z_1 by setting each ϕ_{1j} equal to the coefficient from the simple linear regression of Y onto X_j (i.e. one by one).

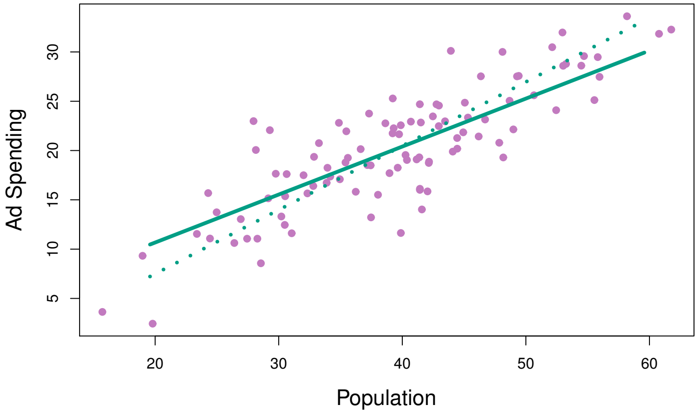
- We can show that this coefficient is proportional to the correlation between Y and X_j.

- To compute $Z_1 = \sum_{j=1}^{p} \phi_{1j} x_j$:

- To find Z_2:
  - Regress each x_j on Z_1, then get the residuals ϵ_j.
  - Then regress Y on each ϵ_j (one by one - simple linear regression).

- To find $Z_3, \ldots, Z_M$ repeat the approach.

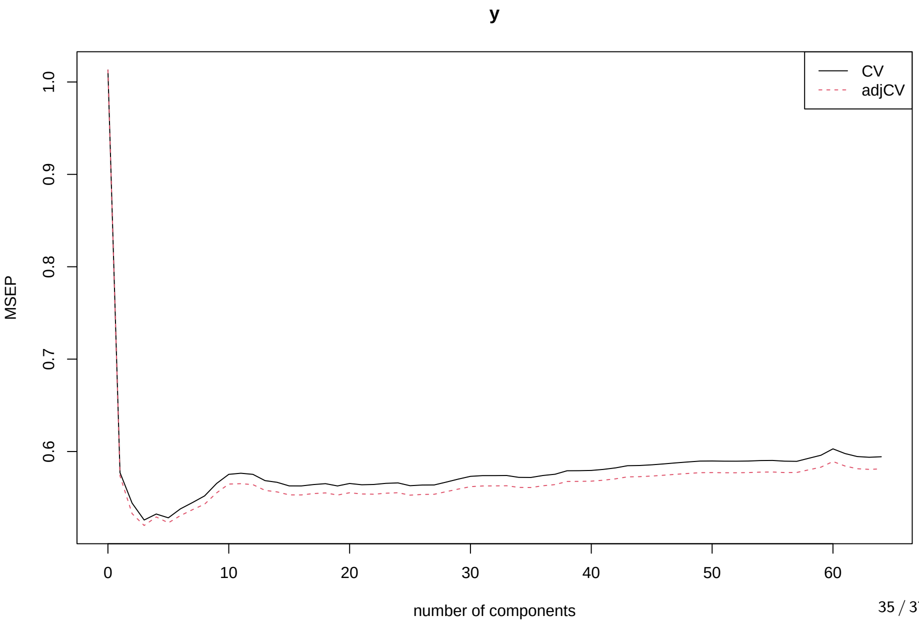
- When $M = p$ we get a solution that is the same as standard regression.



- First Component - PLS (solid) & PCR (dashed)

```r
set.seed(2)
pls.fit <- plsr(y ~ X, scale=FALSE,
                validation="CV")
```

```r
plot(pls.fit, "validation", val.type = "MSEP",
     legendpos = "topright")
```



```r
summary(pls.fit)
```

```
## Data:    X dimension: 342 64
##  Y dimension: 342 1
## Fit method: kernelpls
## Number of components considered: 64
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.006   0.7597   0.7377   0.7252   0.7296   0.7268   0.7334
## adjCV        1.006   0.7574   0.7299   0.7211   0.7275   0.7231   0.7285
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.7381   0.7429   0.7520    0.7585    0.7593    0.7585     0.754
## adjCV   0.7328   0.7369   0.7452    0.7515    0.7518    0.7511     0.747
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
## CV       0.7528    0.7502    0.7502    0.7511    0.7518    0.7502    0.7519
## adjCV    0.7458    0.7437    0.7437    0.7446    0.7451    0.7436    0.7452
##        21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
## CV       0.7510    0.7512    0.7520    0.7524    0.7503    0.7508    0.7509
## adjCV    0.7444    0.7442    0.7449    0.7453    0.7435    0.7440    0.7441
##        28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
## CV       0.7529    0.7551    0.7571    0.7576    0.7576    0.7576    0.7563
## adjCV    0.7460    0.7479    0.7496    0.7501    0.7502    0.7503    0.7491
##        35 comps  36 comps  37 comps  38 comps  39 comps  40 comps  41 comps
## CV       0.7563    0.7577    0.7586    0.7610    0.7611    0.7613    0.7620
## adjCV    0.7490    0.7503    0.7511    0.7534    0.7535    0.7536    0.7543
##        42 comps  43 comps  44 comps  45 comps  46 comps  47 comps  48 comps
## CV       0.7631    0.7646    0.7648    0.7652    0.7659    0.7666    0.7673
## adjCV    0.7553    0.7567    0.7568    0.7573    0.7579    0.7585    0.7591
##        49 comps  50 comps  51 comps  52 comps  53 comps  54 comps  55 comps
## CV       0.7679    0.7680    0.7678    0.7678    0.7679    0.7683    0.7683
## adjCV    0.7597    0.7597    0.7596    0.7596    0.7597    0.7601    0.7602
##        56 comps  57 comps  58 comps  59 comps  60 comps  61 comps  62 comps
## CV       0.7678    0.7677    0.7699    0.7720    0.7765    0.7732    0.7711
## adjCV    0.7597    0.7598    0.7617    0.7636    0.7676    0.7644    0.7625
##        63 comps  64 comps
## CV       0.7706    0.7709
```

- Looks like we should use 4 components.

```
pls.pred <- predict(pls.fit, X.test, ncomp=4)
mean((pls.pred-y.test)^2)
```

```
## [1] 0.5599512
```

- In this case, regularization approaches did better on the test data compared to dimension reduction approaches.