Multivariate Statistical Methods for Big Data Analysis and Process Improvement

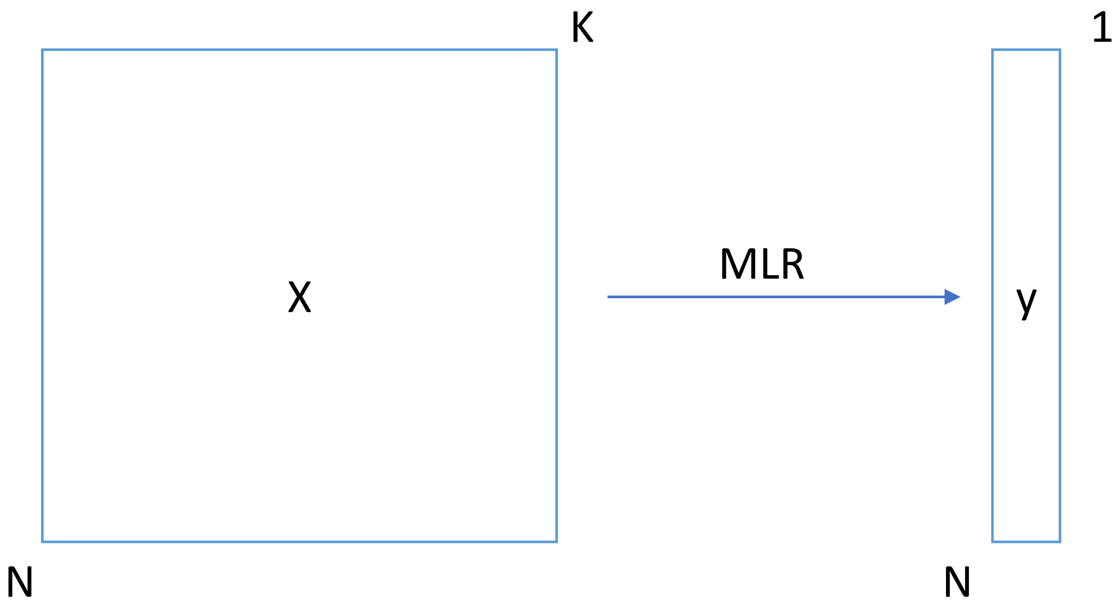
Instructor: Dr. Brandon Corbett

Lecture 7 for ChE 765 | Sep 767, McMaster University

Paper review presentation

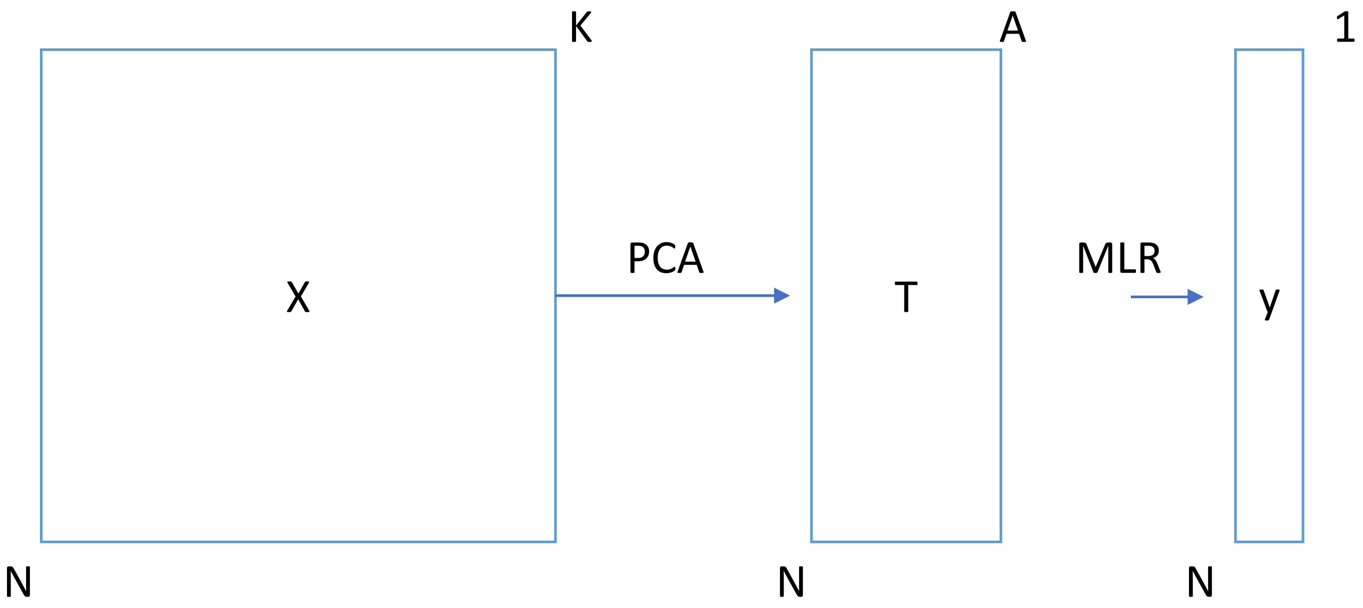
Recap – Two blocks of data

Review: Multiple linear regression



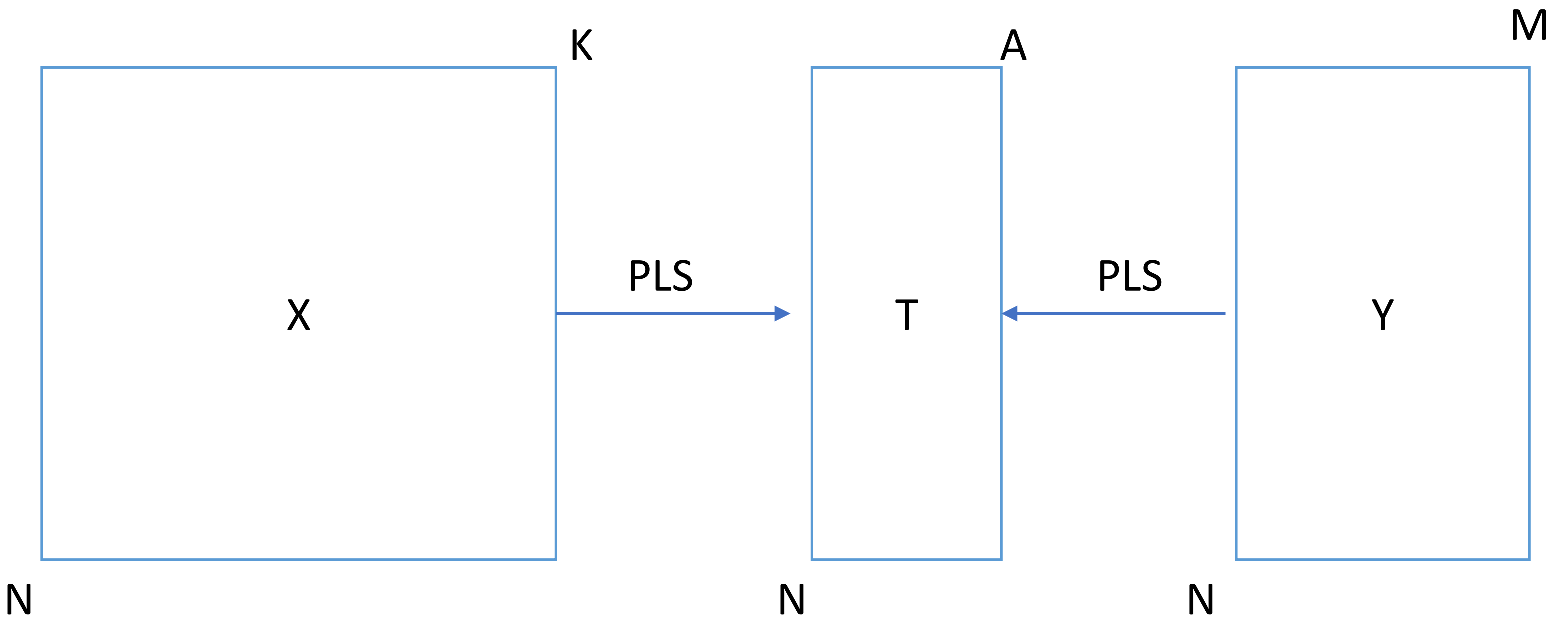
$$y = Xb$$
$$b = (X^T X)^{-1} X^T y$$

Review: principal component regression (PCR)



$$T = XP$$
$$\hat{y} = Tb$$

Projection to Latent Structures (PLS)



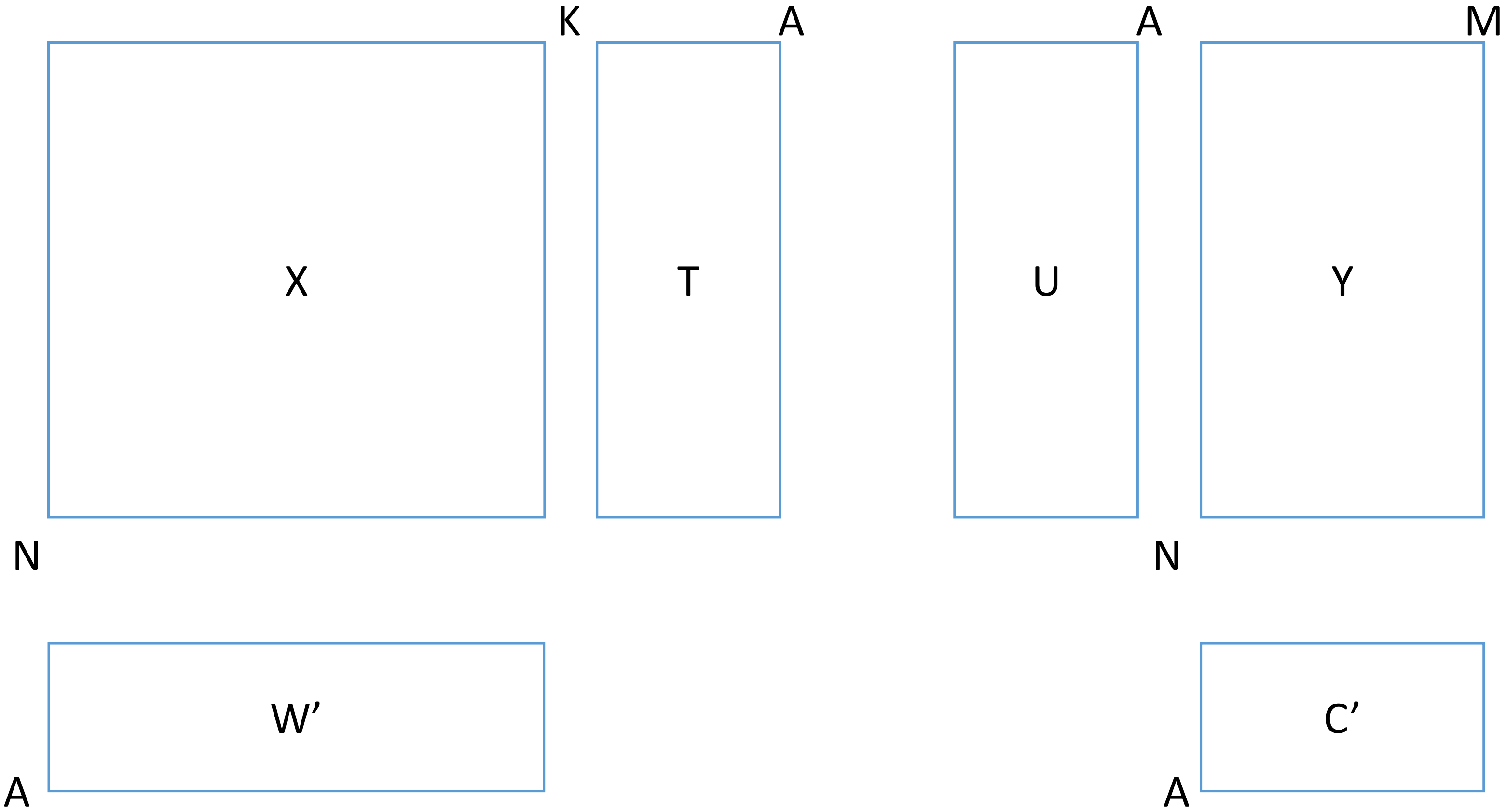
- 2 blocks of data
- Often used to predict Y given X
- Also used for monitoring, optimization, product development

PLS Objective

- Objective of PCA: best explanation of the X-space

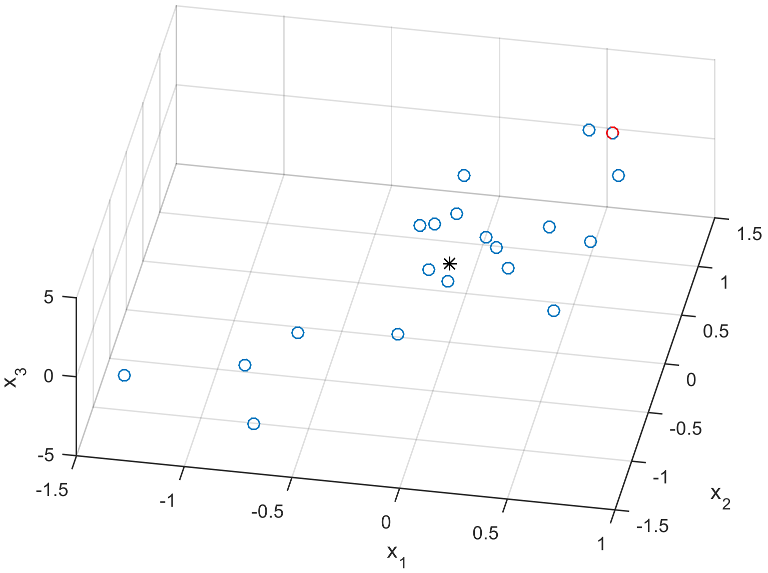
- What we want from PLS:
  1. Best explanation of X-space
  2. Best explanation of Y-space
  3. Maximize relationship between X and Y spaces

PLS: Notation

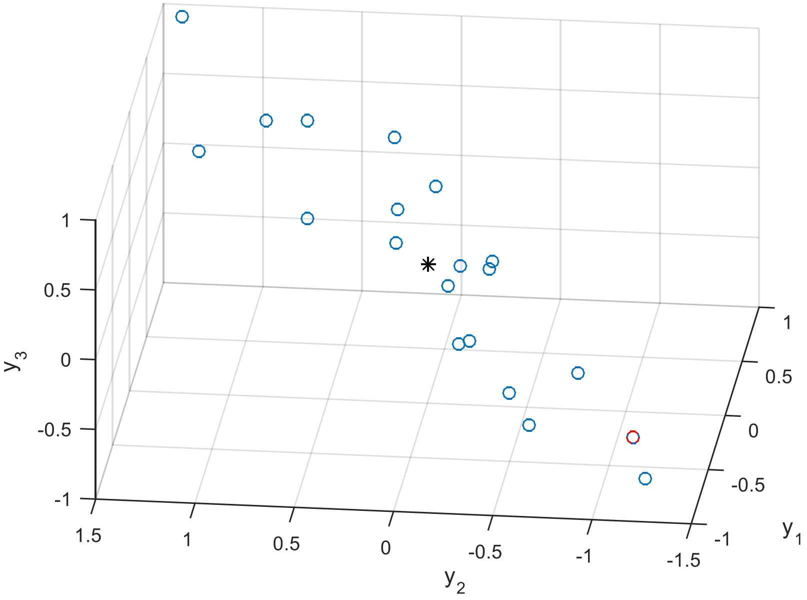


Geometric interpretation

X-Space

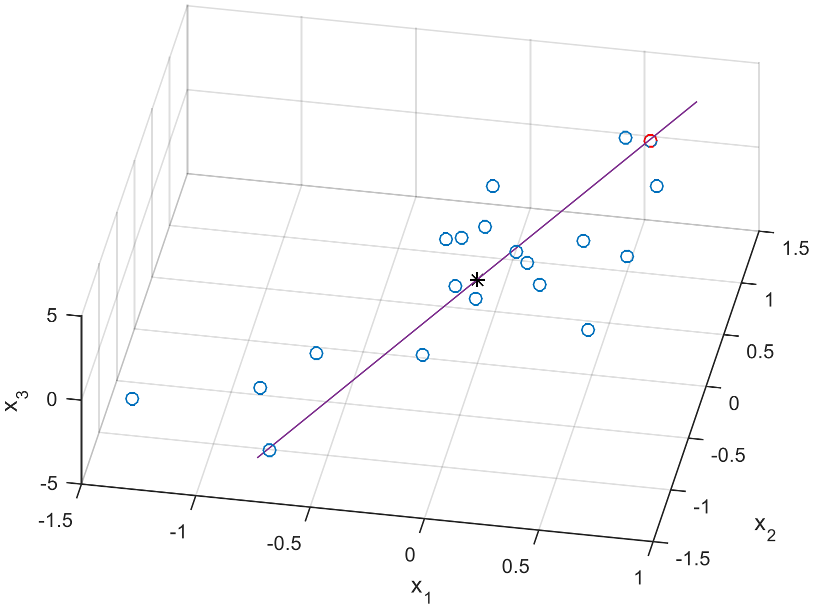


Y-Space

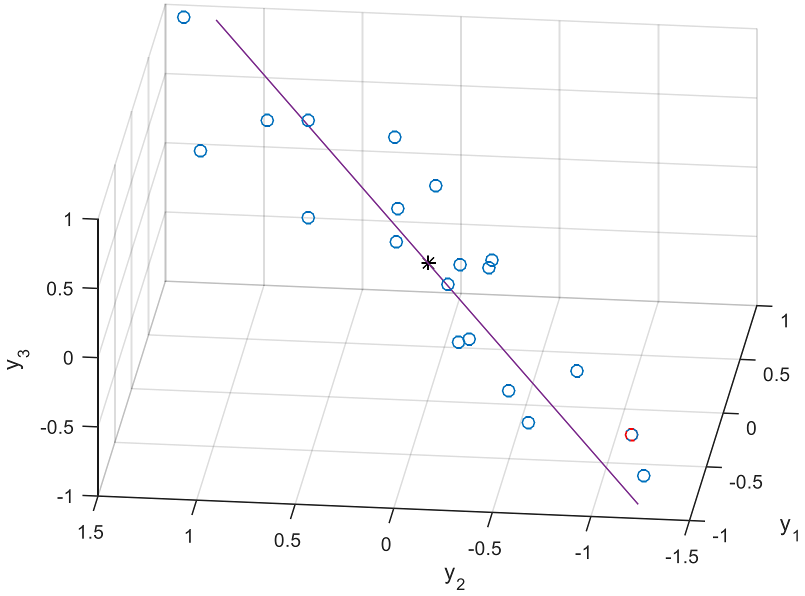


Geometric interpretation – determine weightings

X-Space



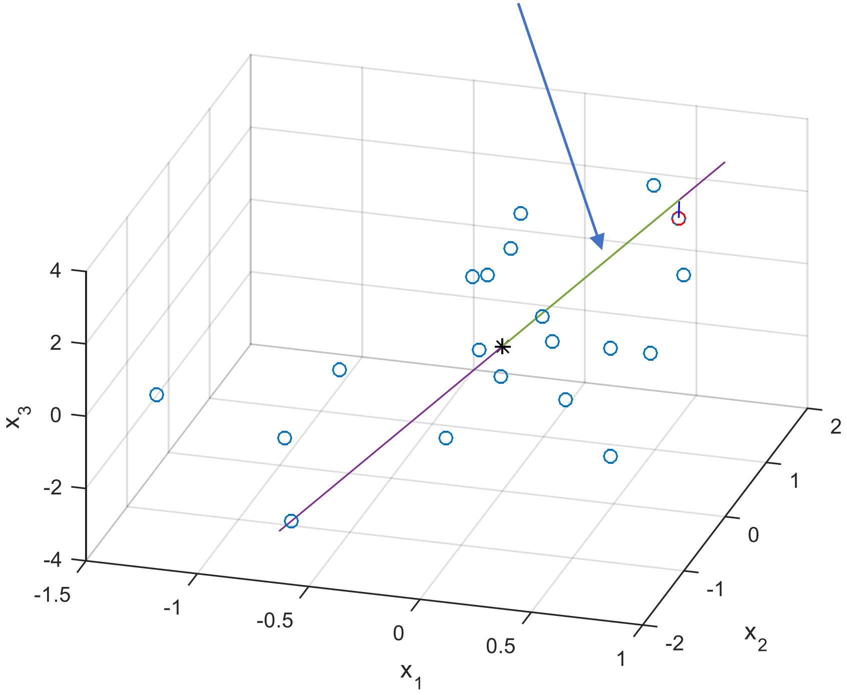
Y-Space



Geometric interpretation – Determine scores

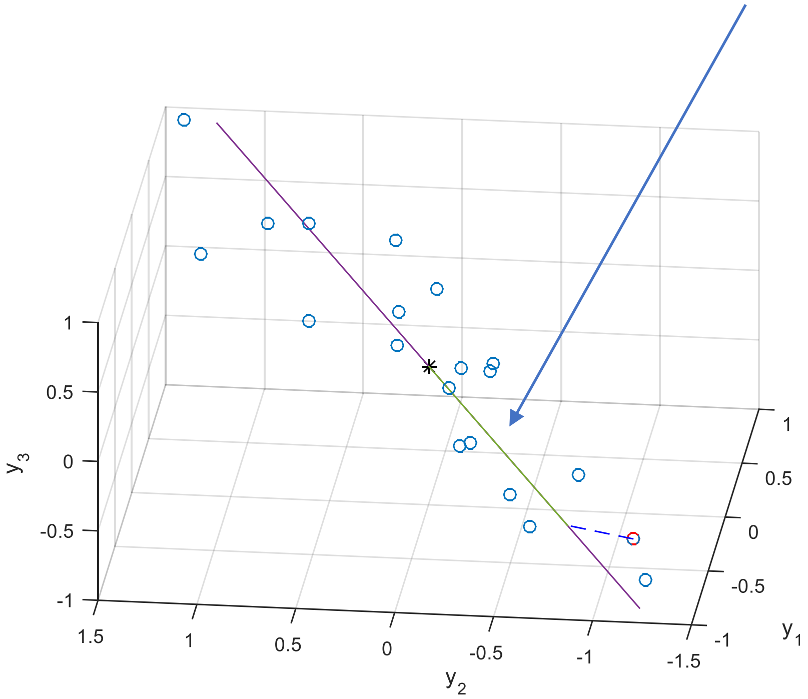
X-Space

t

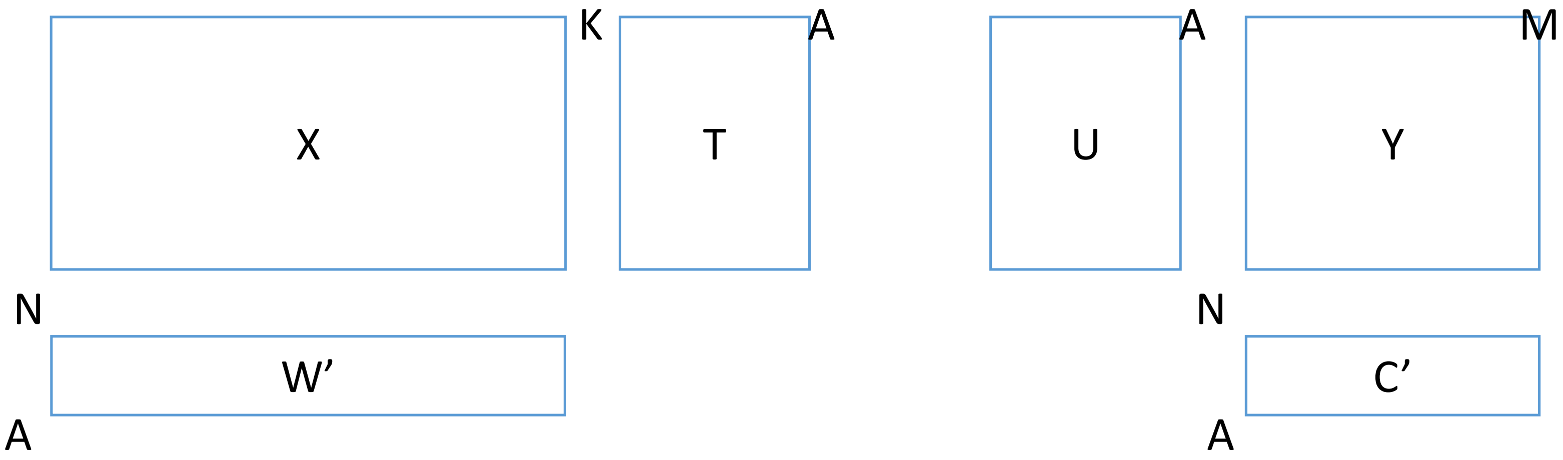


Y-Space

u



Simple PLS (SIMPLS)



- PLS scores explain X:
  - $t_a = X_a w_a$
  - $t_a^T t_a$ subject to $w_a^T w_a = 1.0$
- PLS scores also explain Y:
  - $u_a = Y_a c_a$
  - Max: $u_a^T u_a$ subject to c_a^T
- Maximize covariance (discussion on board)
  - $cov(t_a, u_a) = corr(t_a, u_a) \cdot \sqrt{t_a^T t_a} \cdot \sqrt{u_a^T u_a} \cdot \frac{1}{N}$

1 objective is 3 objectives

Objective for PLS: Maximize covariance of t_a and u_a

$$cov(t_a, u_a) = corr(t_a, u_a) \cdot \sqrt{t_a^T t_a} \cdot \sqrt{u_a^T u_a} \cdot \frac{1}{N}$$

1. Explaining X-space is given by $t_a^T t_a$
2. Explaining Y-space is given by $u_a^T u_a$
3. Maximizing relationship between X and Y space $Corr(t_a, u_a)$

Notes:

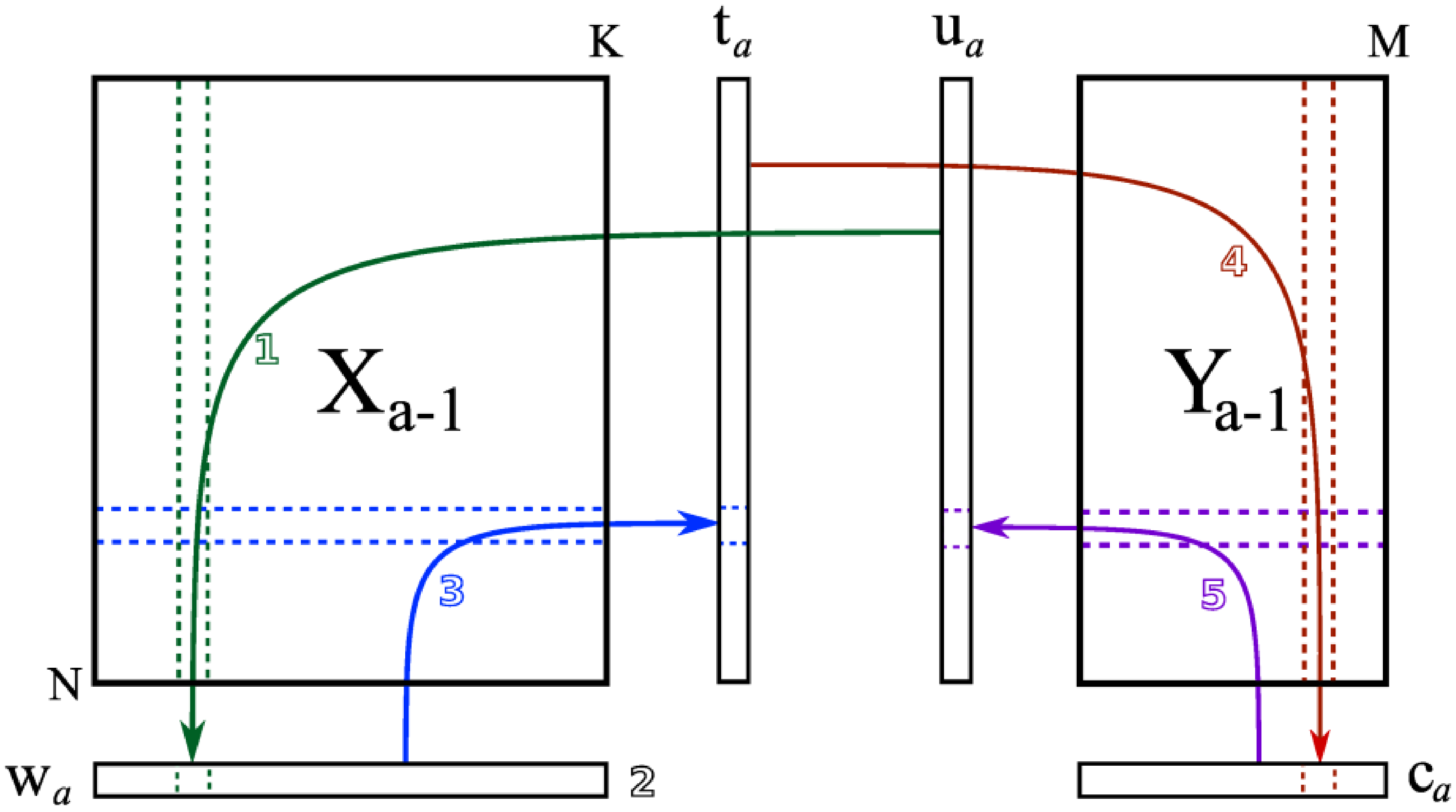
- The above description is for SIMPALS
- We will use NIPALS which is a little different (used by ProMV)
- SIMPALS = NIPALS when M = 1 (ie only one Y variable)

NIPALS: NonLinear Iterative Partial Least Squres

Remarks:

- Very similar both conceptually and in steps to NIPALS for PCA
- In most cases, converges faster than PCA
- May look complicated at first

NIPALS Algorithm



NIPALS Algorithm

- Start with X and Y: preprocessed matrix of raw data
- Call them X_0 and Y_0
- Indicates that no components have been calculated yet

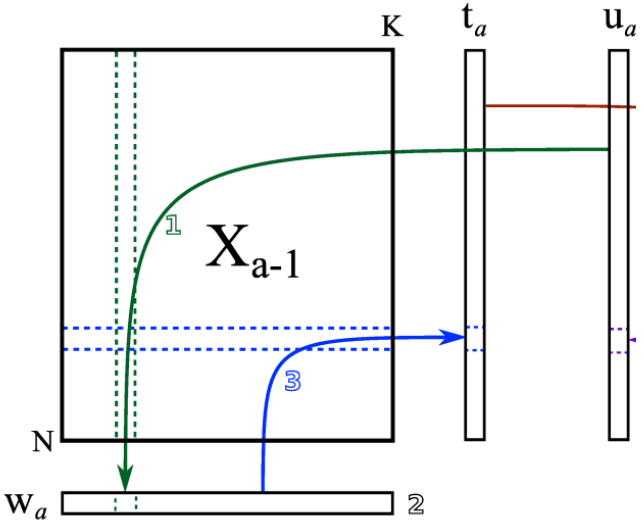
For a = 1,2, …, A:

1. Select an arbitrary initial column for u_a
2. In a while-loop, until convergence:
    1. Regress columns from X_{a-1}onto u_a to get weights w_a
    2. Normalize the weights
    3. Regress rows from X_{a-1}onto $w_a$to get scores t_a
    4. Regress columns from Y_{a-1} onto $t_a$to get weights c_a
    5. Regress rows from Y_{a-1} onto c_a to get scores u_a
3. Deflate component from X_{a-1} and Y_{a-1}

NIPALS Algorithm

Step 2.1 Regress every column from X_{a-1} (call it x_k) onto u_a

- Recall terminology (“regress y onto x”)
- Store regression coefficients as entry in $w_{k,a}$



- Recall least squares for centered data:
- $\hat{y} = \beta x$ and $\beta = \frac{x^T y}{x^T x}$
- In this case: $w_{k,a} = \frac{u_a^T x_k}{u_a^T u_a}$

NIPALS Algorithm

Step 2.1

- Repeat regression for every column in X_{a-1}
- Can calculate regression all at once (if not missing)

$$w_a^T = \frac{1}{u_a^T u_a} u_a^T X_{a-1}$$

u_a is an N x 1 column vector

X_{a-1} is an N x K matrix

w_a is an K x 1 vector

NIPALS algorithm

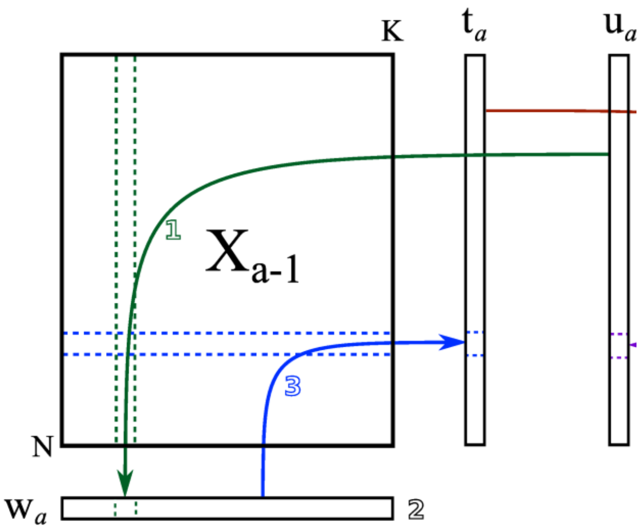
Step 2.2 Normalize the weightings

- w_a won’t have unit length
- Rescale it to magnitude 1.0
- $w_a^T = \frac{1}{\sqrt{w_a^T w_a}} w_a^T = \frac{w_a^T}{\|w_a^T\|}$

NIPALS Algorithm

Step 2.3 Regress every row X_{a-1} onto w_a^T

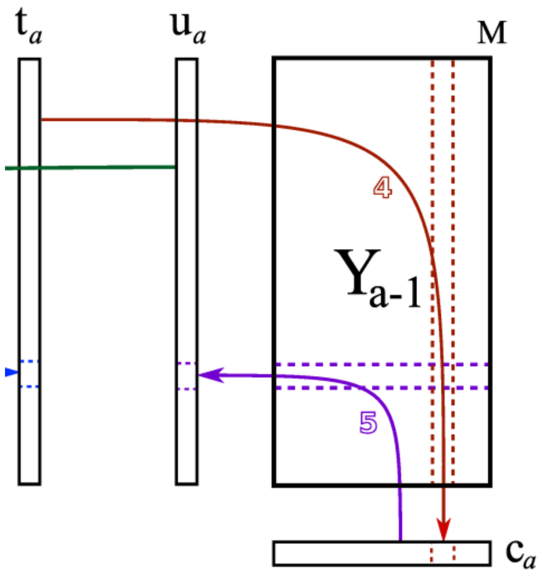
- Regress x_i onto w_a^T
- Store regression coefficients as entry in $t_{i,a}$



- In practice: $t_a = \frac{1}{w_a^T w_a} \cdot X_{a-1} w_a$

NIPALS Algorithm

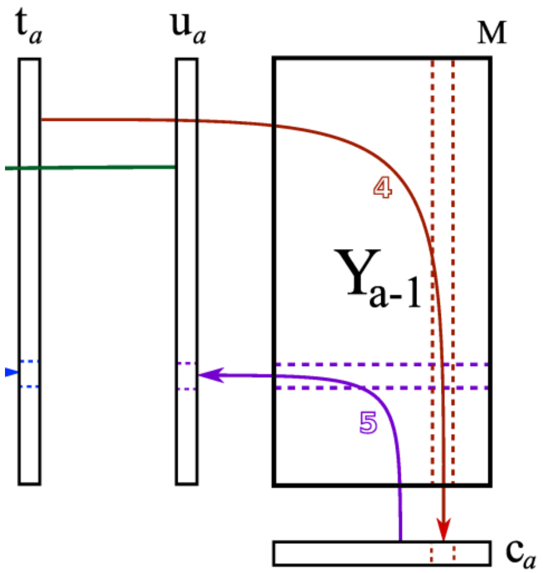
Step 2.4 Regress every column of Y_{a-1} onto t_a to get loadings c_a



- In practice: $c_a^T = \frac{1}{t_a^T t_a} \cdot t_a^T Y_{a-1}$

NIPALS Algorithm

Step 2.5 Regress every row of Y_{a-1} onto c_a to get scores u_a



- In practice: $u_a = \frac{1}{c_a^T c_a} \cdot Y_{a-1} c_a$

Have we converged?

- Compare u_a from previous iteration
- Stop if change is less than $\sqrt{eps} = 1.5 \times 10^{-8}$
- Could aslo check on t_a

- Stop if iterations > 300

At Convergence:

t_a, w_a, u_a and c_a jointly form the ath component
Store them as columns in matrices $T, W, U, and\ C$

Deflation in NIPALS-PLS

We cannot deflate using w_a!

Calculate the loadings matrix P

- Regress columns from X_{a-1}onto converged t_a to get p_a
- $p_a = \frac{1}{t_a^T t_a} X_{a-1}^T t_a$

Deflation in NIPALS-PLS

Remove predicted variability from X_{a-1} and $Y_{a-1}$to get residuals:

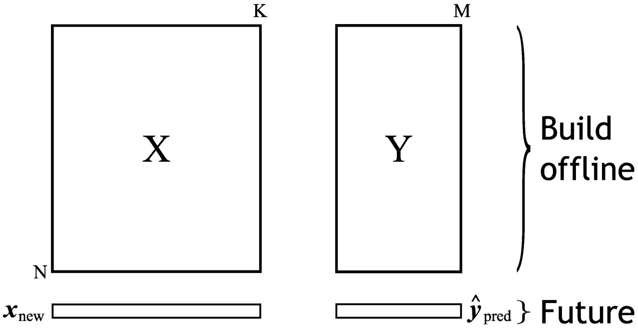
- Deflate X_{a-1}:
  - $E_a = X_{a-1} - \hat{X}_{a-1} = X_{a-1} - t_a p_a^T$
  - $X_a = E_a$
  - Use this $X_a$to fit the next component

- Deflate Y_{a-1}:
  - $F_a = Y_{a-1} - \hat{Y}_{a-1} = Y_{a-1} - t_a c_a^T$
  - $Y_a = F_a$
  - Use this $Y_a$to fit the next component

The weights in PLS

- Scores are calculated from deflated matrices:
  - $\mathbf{t}_1 = \mathbf{X}_{a=0}\mathbf{w}_1 = \mathbf{X}_0\mathbf{w}_1$
  - $\mathbf{t}_2 = \mathbf{X}_{a=1}\mathbf{w}_2 = (\mathbf{X}_0 - \mathbf{t}_1\mathbf{p}_1)\,\mathbf{w}_2$
- $\mathbf{w}_2$: relates score $\mathbf{t}_2$ to $\mathbf{X}_{a=1}$, the deflated matrix
- This is hard to interpret. We would like instead:
  - $\mathbf{t}_1 = \mathbf{X}_{a=0}\mathbf{w}{*}_1 = \mathbf{X}_0\mathbf{w}{*}_1$
  - $\mathbf{t}_2 = \mathbf{X}_{a=0}\mathbf{w}{*}_2 = \mathbf{X}_0\mathbf{w}{*}_2$
  - *etc*
- We calculate matrix $\mathbf{W}^* = \mathbf{W}\left(\mathbf{P}'\mathbf{W}\right)^{-1}$
- So $\mathbf{T} = \mathbf{X}_0\mathbf{W}^*$, or simply: $\boxed{\mathbf{T} = \mathbf{X}\mathbf{W}^*}$
  - $\mathbf{w}{*}_1 = \mathbf{w}_1$
  - $\mathbf{w}{*}_a \neq \mathbf{w}_a$ for $a > 1$
- We get a clearer interpretation of the variable relationships using $\mathbf{W}^*$ instead of $\mathbf{W}$

Using PLS on new data

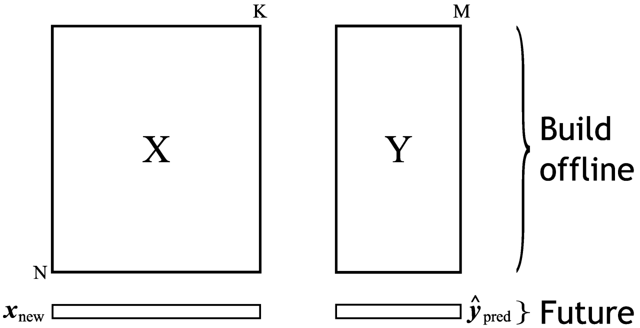


$$\begin{aligned}
t_{1,\text{new}} &= \mathbf{x}'_{\text{new}}\mathbf{w}_1 \\
\mathbf{x}'_{\text{new}} &= \mathbf{x}'_{\text{new}} - t_{1,\text{new}}\mathbf{p}'_1 \qquad \text{(deflate)} \\
t_{2,\text{new}} &= \mathbf{x}'_{\text{new}}\mathbf{w}_2 \\
\mathbf{x}'_{\text{new}} &= \mathbf{x}'_{\text{new}} - t_{2,\text{new}}\mathbf{p}'_2 \\
\textit{etc}
\end{aligned}$$

Collect all the $t_{a,\text{new}}$ score values in $\mathbf{t}_{\text{new}}$

Alternatively use $\mathbf{t}_{\text{new}} = \mathbf{x}'_{\text{new}}\mathbf{W}^*$ to get $\mathbf{t}_{\text{new}}$ without deflation

Using PLS on new data



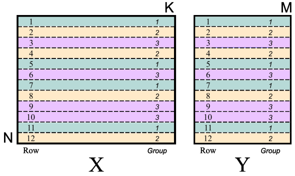
$$
\begin{aligned}
\widehat{\mathbf{y}}'_{\text{new}} &= \mathbf{t}'_{\text{new}}\mathbf{C}' \\
\widehat{\mathbf{y}}'_{\text{new}} &= \mathbf{x}'_{\text{new}}\mathbf{W}^{*}\mathbf{C}'
\end{aligned}
$$

- Then uncenter and unscale the $\widehat{\mathbf{y}}'_{\text{new}}$

Cross-validation to calculate Q^2

Similar procedure as with PCA

Split the rows in $\mathbf{X}$ and $\mathbf{Y}$ into G groups.

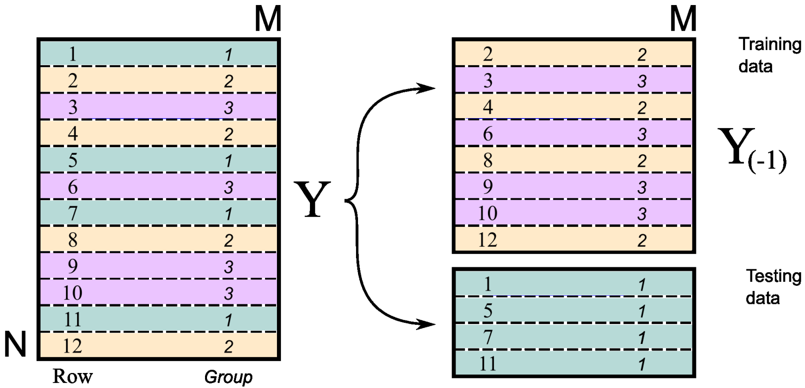


$G = 3$ in this illustration

- Typically $G \approx 7$ [ProSensus, Simca-P use $G = 7$]
- Rows can be randomly grouped, or
- ordered *e.g.* 1, 2, 3, 1, 2, 3, ...
- ordered *e.g.* 1, 1, 2, 2, 3, 3, ...

Cross-validation concept for PLS

Fit a PLS model using $\mathbf{X}_{(-1)}$ and $\mathbf{Y}_{(-1)}$; use $\mathbf{X}_{(1)}$ as testing data



Row	Group
1	1
2	2
3	3
4	2
5	1
6	3
7	1
8	2
9	3
10	3
11	1
12	2

Training data

2	2
3	3
4	2
6	3
8	2
9	3
10	3
12	2

Testing data

1	1
5	1
7	1
11	1

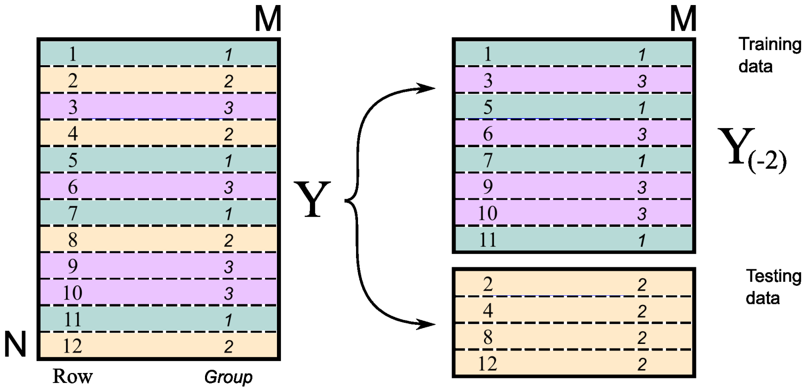
Split the X-matrix along the same rows,
but only calculate PRESS using F matrix.

$$F_{(1)} = Y_{(1)} - \hat{Y}_{(1)}$$

$\mathbf{F}_{(1)}$ = prediction error for testing group 1

Cross-validation concept for PLS

Fit a PLS model using $\mathbf{X}_{(-2)}$ and $\mathbf{Y}_{(-2)}$; use $\mathbf{X}_{(2)}$ as testing data



Row	Group
1	1
2	2
3	3
4	2
5	1
6	3
7	1
8	2
9	3
10	3
11	1
12	2

Training data

1	1
3	3
5	1
6	3
7	1
9	3
10	3
11	1

Testing data

2	2
4	2
8	2
12	2

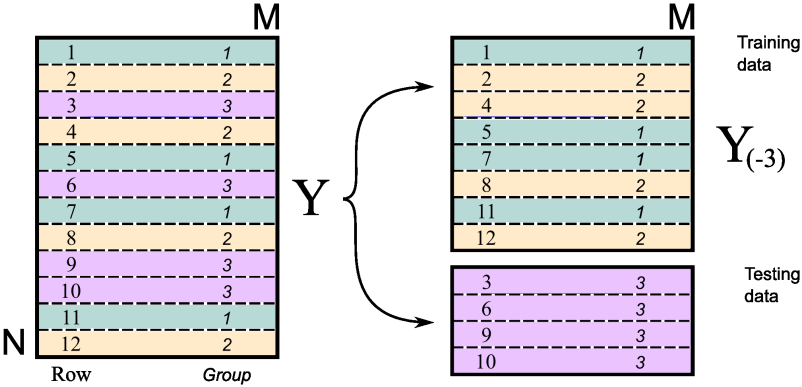
Split the X-matrix along the same rows, but only calculate PRESS using F matrix.

$$F_{(2)} = Y_{(2)} - \hat{Y}_{(2)}$$

$\mathbf{F}_{(2)}$ = prediction error for testing group 2

Cross-validation concept for PLS

Fit a PLS model using $\mathbf{X}_{(-3)}$ and $\mathbf{Y}_{(-3)}$; use $\mathbf{X}_{(3)}$ as testing data



Row	*Group*
1	*1*
2	*2*
3	*3*
4	*2*
5	*1*
6	*3*
7	*1*
8	*2*
9	*3*
10	*3*
11	*1*
12	*2*

Training data

1	*1*
2	*2*
4	*2*
5	*1*
7	*1*
8	*2*
11	*1*
12	*2*

Testing data

3	*3*
6	*3*
9	*3*
10	*3*

Split the X-matrix along the same rows,
but only calculate PRESS using F matrix.

$$F_{(3)} = Y_{(3)} - \hat{Y}_{(3)}$$

$\mathbf{F}_{(3)}$ = prediction error for testing group 3

Cross-validation concept for PLS

- PRESS $= \text{ssq}(\mathbf{F}_{(1)}) + \text{ssq}(\mathbf{F}_{(2)}) + \ldots + \text{ssq}(\mathbf{F}_{(G)})$
- PRESS = prediction error sum of squares from each prediction group
- $Q^2 = 1 - \dfrac{\mathcal{V}(\text{predicted } \mathbf{F}_A)}{\mathcal{V}(\mathbf{Y})} = 1 - \dfrac{\text{PRESS}}{\mathcal{V}(\mathbf{Y})}$
- Q^2 is calculated and interpreted in the same way as R^2
- Q^2_k can be calculated for variable $k = 1, 2, \ldots K$
- You should always find $Q^2 \leq R^2$
- If $Q^2 \approx R^2$: that component is useful and predictive in the model
- If Q^2 is "small": that component is likely fitting noise

To read: Esbensen and Geladi, 2010, "Principles of proper validation"

PLS plots

- Score plots: **t** and **u** show relationship between rows
- Weight plots: **w**: relationship between **X** columns
- Loading plots: **c**: relationship between **Y** variables
- Weight and loading plots: $\mathbf{w}^*\mathbf{c}$: relationship between **X** and **Y**
- SPE plots (X-space, Y-space)
- Hotelling's T^2 plot
- Coefficient plots
- VIP plot
- R^2 plots (X-space, Y-space, per variable)

Variable importance to prediction

Important variables in the model?

- Have large (absolute) weights: why?
- Come from a component that has a high R^2

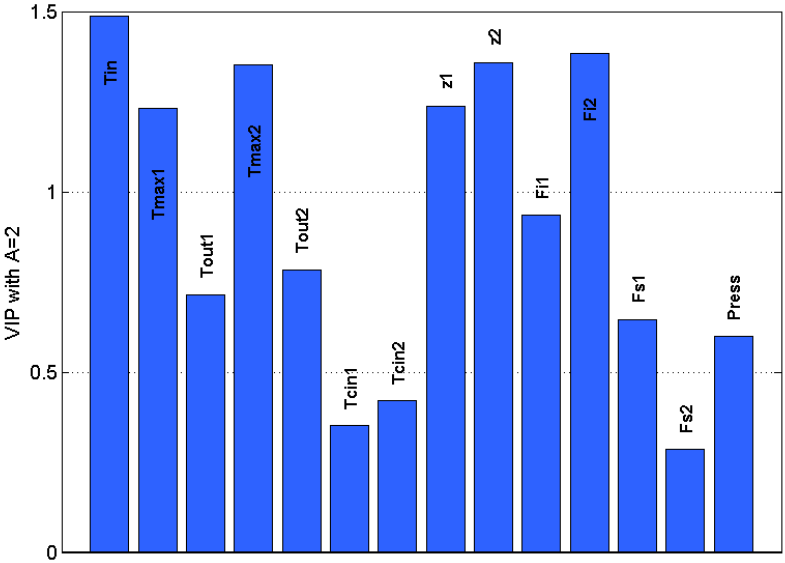
Combining these two concepts we calculate *for each variable*:

Importance of variable k using A components in PLS

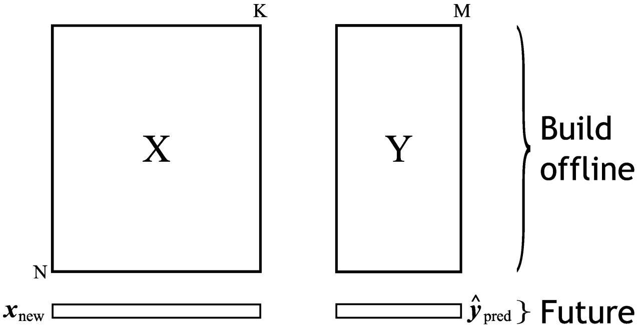
$$VIP^{\mathbf{2}}_{A,k} = \frac{K}{\mathrm{SSX}_0 - \mathrm{SSX}_A} \cdot \sum_{a=1}^{A} \left(\mathrm{SSX}_{a-1} - \mathrm{SSX}_a\right) W^2_{a,k}$$

- SSX_a = sum of squares in the $\mathbf{X}$ matrix after a components
- $\frac{\mathrm{SSX}_{a-1} - \mathrm{SSX}_a}{SSX_A}$ = incremental R^2 for a^{th} component
- $\frac{\mathrm{SSX}_0 - \mathrm{SSX}_A}{SSX_A} = R^2$ for model using A components
- Messy, but you can show that $\sum_k VIP^2_{A,k} = K$
- Reasonable cut-off = 1
- VIP for PCA models: use $P^2_{a,k}$ instead of $W^2_{a,k}$

Variable importance to prediction



Coefficient plot

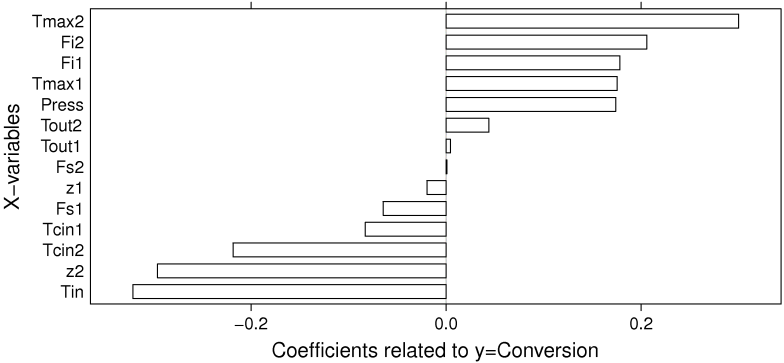


$$
\begin{aligned}
\widehat{\mathbf{y}}'_{\text{new}} &= \mathbf{t}'_{\text{new}}\mathbf{C}' \\
\widehat{\mathbf{y}}'_{\text{new}} &= \mathbf{x}'_{\text{new}}\mathbf{W}^{*}\mathbf{C}' \\
\widehat{\mathbf{y}}'_{\text{new}} &= \mathbf{x}'_{\text{new}}\boldsymbol{\beta}
\end{aligned}
$$

- $\boldsymbol{\beta}$ is a $K \times M$ matrix
- Each column in $\boldsymbol{\beta}$ contains the regression coefficients for column m from **Y** matrix
- **Never implement PLS using $\boldsymbol{\beta}$ matrix**

Coefficient plot

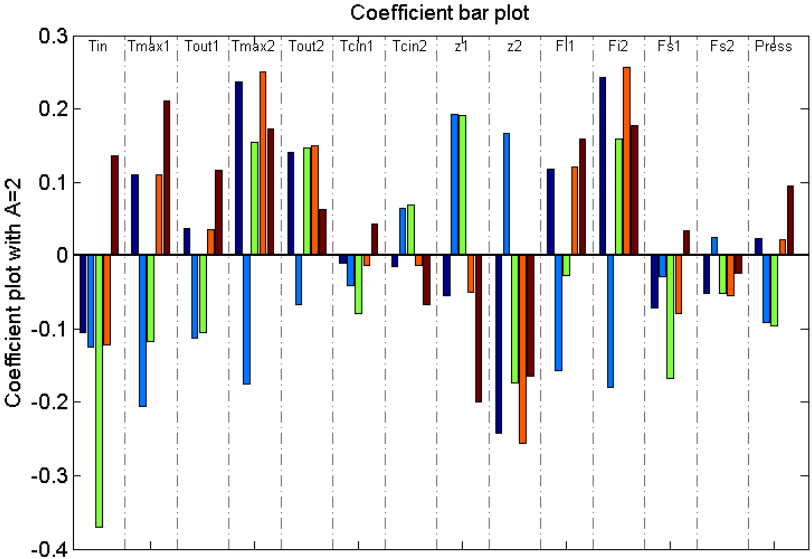
For a single *y*-variable:



- $\hat{y} = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_K x_K$
- where x_k and $\hat{y}$ are the preprocessed values
- *Again* – never implement PLS this way.

Coefficient plot

For all **Y**-variables



Jackknifing

We re-calculate the model $G + 1$ times during cross-validation:

- G times, once per group
- The "+1" is from the final round, where we use **all** observations

We get $G + 1$ estimates of the model parameters:

- loadings
- VIP values
- coefficients

for every variable $(1, 2, \ldots K)$.

Calculate "reliability intervals" (don't call them confidence intervals)

- Martens and Martens (paper 43) describe jackknifing.
- Efron and Tibshirani describe the bootstrap and jackknife.