

# Latent Variable Methods Course

## Learning from data

Instructor: Kevin Dunn  
kevin.dunn@connectmv.com  
<http://connectmv.com>

© Kevin Dunn, ConnectMV, Inc. 2011

Revision: 268:adfd compiled on 15-12-2011

# Copyright, sharing, and attribution notice

This work is licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported License. To view a copy of this license, please visit

<http://creativecommons.org/licenses/by-sa/3.0/>



This license allows you:

- ▶ **to share** - to copy, distribute and transmit the work
- ▶ **to adapt** - but you must distribute the new result under the same or similar license to this one
- ▶ **commercialize** - you are allowed to create commercial applications based on this work
- ▶ **attribution** - you must attribute the work as follows:
  - ▶ "Portions of this work are the copyright of ConnectMV", *or*
  - ▶ "This work is the copyright of ConnectMV"

We appreciate:

- ▶ if you let us know about **any errors** in the slides
- ▶ **any suggestions to improve the notes**
- ▶ telling us if you use the slides, especially commercially, so we can inform you of major updates
- ▶ emailing us to ask about different licensing terms

All of the above can be done by writing us at

**[courses@connectmv.com](mailto:courses@connectmv.com)**

If reporting errors/updates, please quote the current revision number: 268:adfd

# Advanced Preprocessing

In this next section we will look at data preprocessing

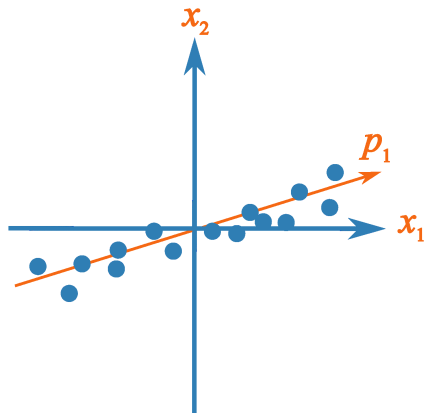
- ▶ Centering
- ▶ Scaling
- ▶ Block scaling
- ▶ Trimming and Winsorizing
- ▶ Univariate transformations
- ▶ Calculated variables
- ▶ Integer and categorical variables (qualitative variables)
- ▶ Process dynamics handled by lagging columns in **X** and/or **Y**
- ▶ Dealing with nuisance variation
- ▶ Meaningful feature extraction

# Centering

- ▶ Removes arbitrary offset from each column
- ▶ Not centering: sometimes results in extra component
- ▶ If centering: sometimes get a better fit
- ▶ Centering: can have a major impact on model's interpretation
- ▶ May be done around a “natural” reference: e.g. setpoint, instead of the mean
- ▶ For robustness: center about the median instead

More details from this very insightful paper: [Bro and Smilde](#):  
“Centering and scaling in component analysis”

# Scaling

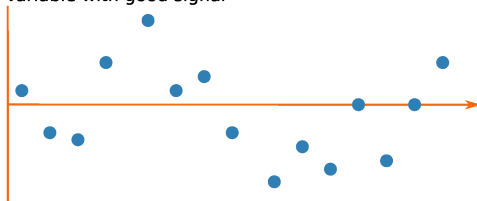


- ▶ If no prior knowledge: scale each column to unit variance
- ▶ Need to emphasize/deemphasize a variable?
  - ▶ First scale all columns to unit variance
  - ▶ Then upscale/downscale the column(s) as appropriate

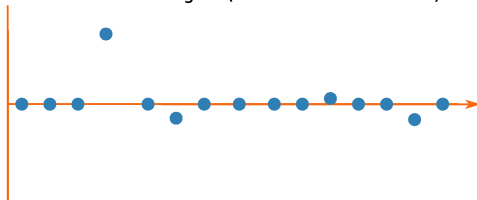
# Scaling

Be careful of inflating noise in variables with little signal

Variable with good signal



Variable with little signal (low standard deviation)

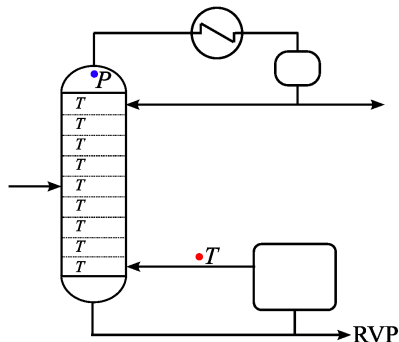


- ▶ Use robust scaling instead (scale by MAD)
- ▶ Heuristic: don't scale  $x_k$  if  $\text{stdev}(x_k) < 4 \times \text{its measurement noise}$

$$\text{median absolute deviation}_k = 1.4826 \cdot \text{median} \{ |x_k - \text{median} \{x_k\}| \}$$

# Block-scaling, or group-scaling

Distillation column example: 80 tags



- ▶ 50 tray temperatures
- ▶ 30 other tags available

What will be the prominent direction in PCA or PLS model?

Block scaling:

- ▶ Do normal preprocessing
- ▶ Then divide group of tags (called a block of variables) by  $\sqrt{K_b}$



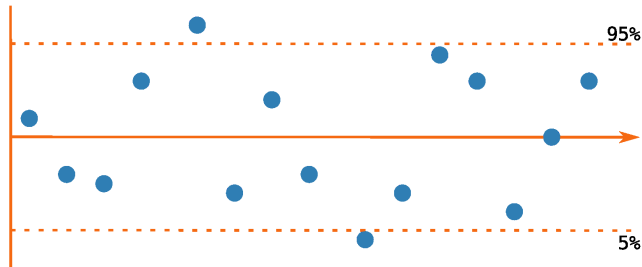
# Robust centering and scaling

- ▶ Use robust preprocessing to emphasize outliers
- ▶ Interesting project topic
  - ▶ Does robust preprocessing complicate or help the subsequent PCA/PLS model-building?
  - ▶ What about robust preprocessing followed by robust PCA or robust PLS?
  - ▶ What if we modify the NIPALS iterations with a robust alternative?
    - ▶ Use the hat-values to down-weight outliers?
    - ▶ Several robust LS methods: e.g. Tukey, quantiles-and-medians, and others
    - ▶ Will the NIPALS algorithm still converge?

Very readable paper: [Filzmoser and Todorov, 2011](#): “Review of robust multivariate statistical methods in high dimension”

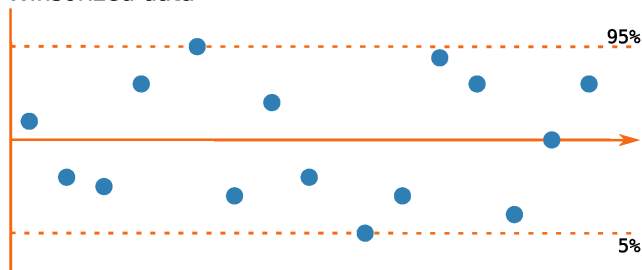
# Dealing with outliers: winsorizing

Raw data



Replace outliers beyond a chosen  $\alpha$  level at their respective bounds

Winsorized data

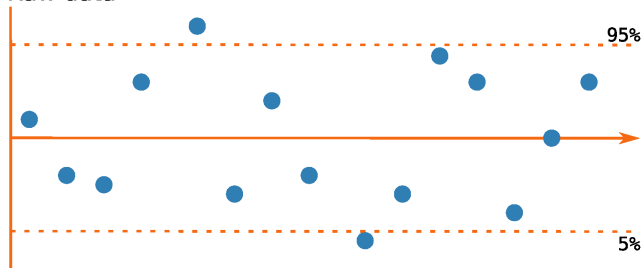


e.g.  $\alpha = 5\%$  shown here

Can break multivariate relationships between variables

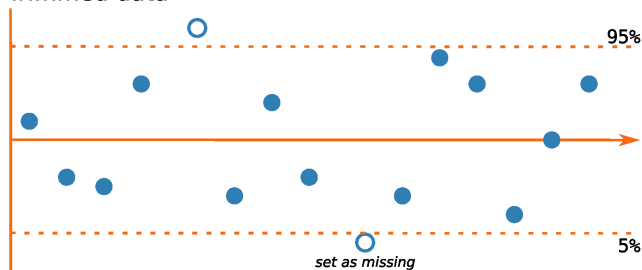
# Dealing with outliers: trimming

Raw data



Convert outliers beyond a chosen  $\alpha$  level to missing values

Trimmed data



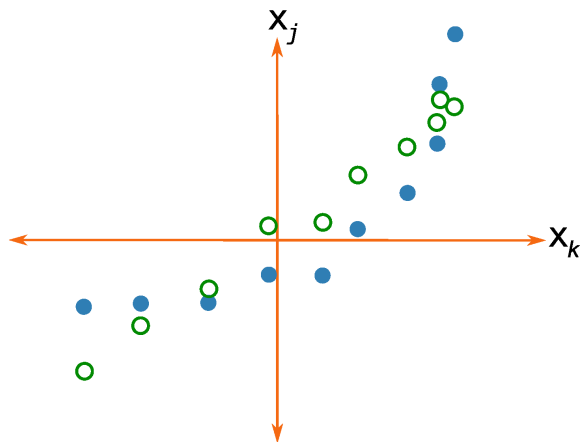
e.g.  $\alpha = 5\%$  shown here

Better (more honest) alternative to winsorizing

# Univariate transformations

Univariate transformation's often motivated by first principle's knowledge:

- Original data, then centered and scaled
- Transformed data, then centered and scaled



Examples:

- ▶ Use  $\log(T)$  instead of temperature,  $T$
- ▶ Use  $1/P$  instead of pressure,  $P$
- ▶  $\sqrt{F}$  instead of flow,  $F$  (as shown in figure)

# Univariate transformations

- ▶ Pre-bias data to avoid negative and zero values
- ▶ e.g. if  $T$  is measured in Celcius and could be negative, use  $\log(T + c)$
- ▶ where  $c$  is large enough to avoid negative logs
  
- ▶ Can be used on **X and Y** variables
  - ▶ e.g. use  $\log(y)$  to get better predictions: software takes care of transforming and un-transforming

# Expanding the **X**-matrix with calculated variables

Add extra columns to **X**:

- ▶ Heat balance:
  - ▶ add column for sum of all heat inputs
  - ▶ add column for sum of heat outputs
  - ▶ add column for sum of heat created/lost by reaction
- ▶ As above, but for mass balance
  - ▶ See “Data reconciliation” (Tong and Crowe's work)
- ▶ Key performance indicators for your particular industry
- ▶ Add dimensionless numbers:
  - ▶ Reynolds number =  $\frac{\rho v D}{\mu}$
  - ▶ Power number =  $\frac{P}{\rho n^3 d^5}$

Soft-sensor applications:

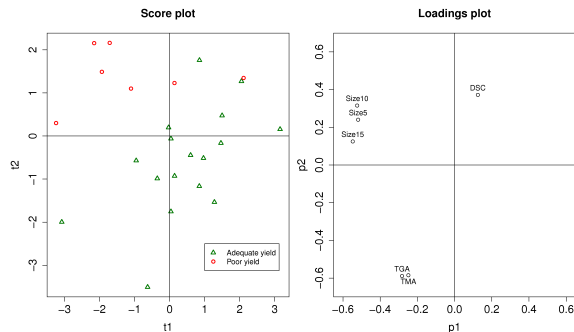
- ▶ Add columns from first-principles equations, e.g. Antoine equation:  $\log(\text{RVP}) = \log P - B \left[ \frac{1}{C} - \frac{1}{T + C} \right]$

# Handling qualitative variables

## Binary variables

- ▶ (yes/no) **or** (on/off) **or** (present/absent)
- ▶ Also called “dichotomous variables”
- ▶ Included and used like any other variable
- ▶ Centering and scaling affected by the relative number of rows from each category
- ▶ *illustration on the board*

Or just use variable to colour-code scores by:



## Handling qualitative variables

*Unordered* indicators must be expanded into extra columns

- ▶ aka “dummy variables”, or “categorical variables”
- ▶ can be done with **X**-space and/or **Y**-space variables

e.g. reactor T, reactor D, reactor H

1	0	0	← T
0	1	0	← D
0	0	1	← H

Should then block scale the group of columns, especially if number of levels is high

We will use this in the class on “Classification”



## Ordered categorical variable: ordinal variable

e.g. Primary school, High school, College, Bachelor's, Masters

- ▶ Can convert them to a single column of integers. e.g.
  - ▶ 1 = Primary school
  - ▶ 3 = College
  - ▶ 5 = Masters
- ▶ You may choose to leave them as a single column then
  - ▶ e.g. months of the year: Jan=01, Feb=02, etc
- ▶ Loadings interpretation: same as a continuous variable
- ▶ As a predictor in the **Y**-space: round prediction to closest integer

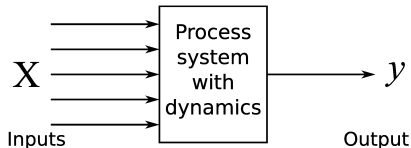
**Caution:** In many cases the gap from say 1 to 2 is not the same as the gap from say 3 to 4.

Rather expand into columns then.

# Handling process dynamics with lagging

Time series modelling is a well-studied topic\*

- ▶ Inputs:  $\mathbf{x}(t)$
- ▶ Outputs:  $\mathbf{y}(t)$



- ▶ Time series models can be built to predict  $\mathbf{y}_t$  when given:
  - ▶  $\mathbf{x}_t, \mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \dots$
  - ▶  $\mathbf{y}_{t-1}, \mathbf{y}_{t-2}, \dots$
- ▶ These models are then used for:
  - ▶ forecasting: to make decisions
  - ▶ control: e.g. feedback control
  - ▶ process monitoring
- ▶ Can we include these ideas in PCA/PLS?

\* Standard reference: Box and Jenkins; book by Chatfield is very good

## Time series example

- ▶ First order system's transfer function:  $\frac{y(s)}{x(s)} = \frac{K}{\tau s + 1}$
- ▶ Time domain:  $\tau \frac{dy}{dt} + y(t) = Kx(t)$
- ▶ Take samples  $\Delta t$  time units apart:

$$y_t = \delta y_{t-1} + K(1 - \delta)x_t \quad \text{where} \quad \delta = e^{-\frac{\Delta t}{\tau}}$$

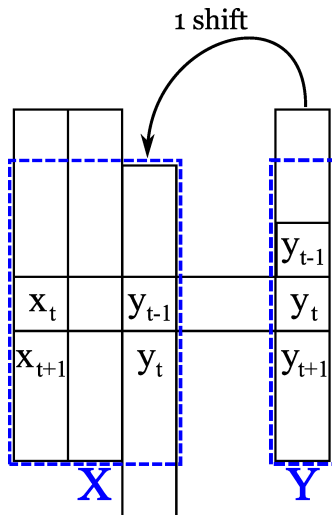
General form:

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \dots + a_m y_{t-m} + b_0 x_t + b_1 x_{t-1} + \dots + b_n x_{t-n}$$

- ▶ function of past y's:  $y_{t-1}, y_{t-2}, \dots$
- ▶ function of current and past x's:  $x_t, x_{t-1}, \dots$

# Time series example: lagging Y's

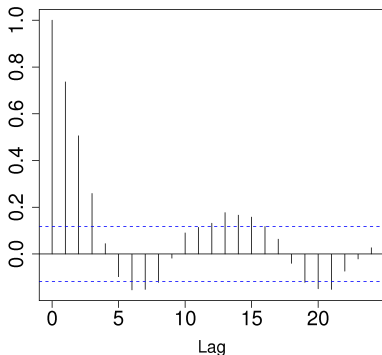
- ▶ How to approximate this with a latent variable model?
- ▶ Copy and shift columns to get consistency within a row
- ▶ So add columns to  $\mathbf{X}$  to get the time-history of  $y$
- ▶  $y(t) = a_1 y(t-1) + b_0 x_t + \dots$



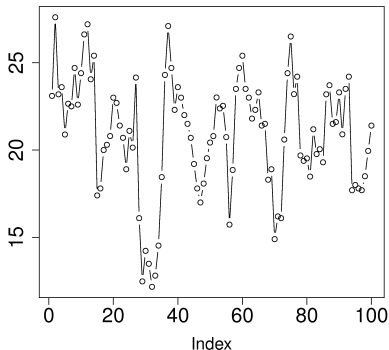
## Time series example: lagging Y's

- ▶ Lagging adds strong correlations among columns in **X**
- ▶ That's OK: PLS can handle it.
- ▶ What if we don't know how many lags to add?
- ▶ One approach: use the autocorrelation function: `acf(y)`

**Autocorrelation function**



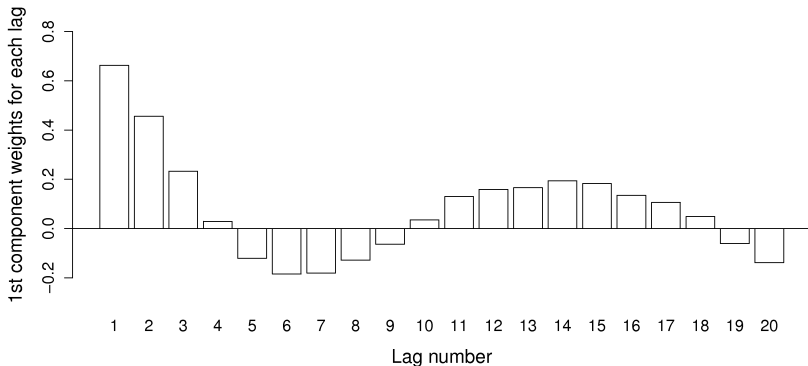
**Sample of the raw data**



# Time series example: lagging Y's

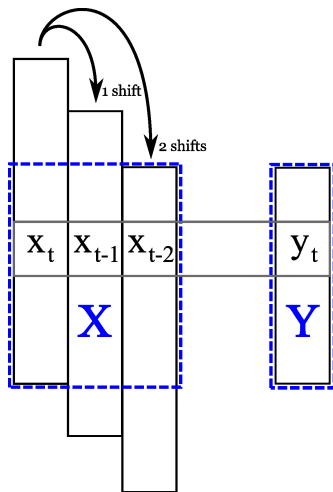
Other approaches:

- ▶ look at the PLS coefficients and jack-knifed reliability intervals
- ▶ look at the **X**-space weights:  $\mathbf{w}^*$

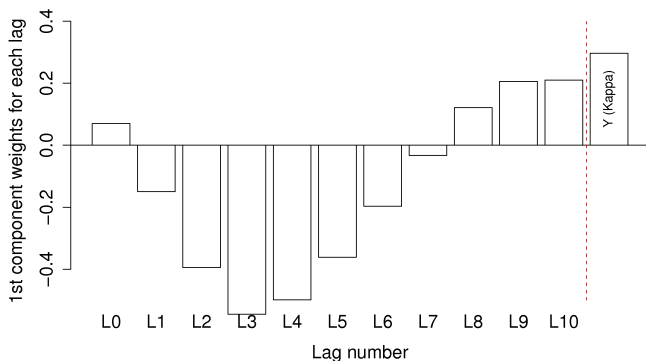


## Time series example: lagging X's

- ▶ Previous history in **X** variables might be useful
- ▶ Especially in processes with long mixing times/residence times
- ▶  $y_t = b_0x_t + b_1x_{t-1} + b_2x_{t-2} + \dots$
- ▶ Unsure how many? Add lags then check coefficient and/or **w \* c** plots
- ▶ Use lags which are large



# Lagging



- ▶ Sometimes we find counter-intuitive lags
- ▶ Sometimes the lags are all small and “smeared” over many columns
  - ▶ Then use an average over all important lags instead
  - ▶ Better still: block-scale all lags to have equivalent influence of 1 variable



## Dealing with nuisance variation

Sometimes we have dominant variation that we are not interested in: e.g. **throughput**

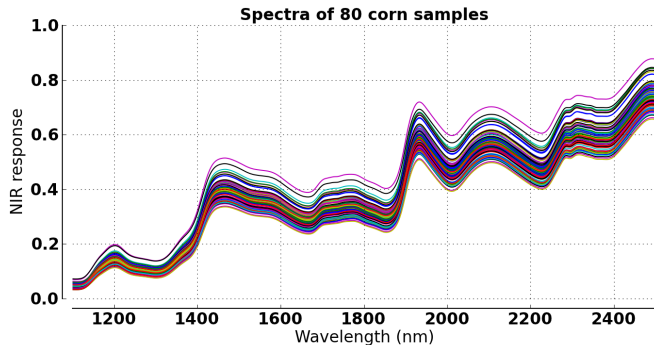
This can dominate a component, and cause false alarms for monitoring. Can't remove throughput variable, because it helps the model. Options?

1. If it is just in one score, e.g.  $t_2$ , just skip it in  $T^2$  calculation
2. Regress out the throughput effect:
  - ▶ Regress the throughput variable on all other  $X$ -variables
  - ▶ Take residuals (which are mostly orthogonal now to throughput)
  - ▶ Use these residuals as your  $X$
  - ▶ Model is hard to interpret though; good for predictions
3. Preprocessing option: divide some  $X$ -variables by throughput to normalize out throughput effect; still keep the single throughput variable.

# Spectral data

Spectral data have many specific preprocessing tools:

- ▶ Savitzky-Golay filtering/smoothing
- ▶ Multiplicative Scatter Correction (MSC)
- ▶ Standard Normal Variate (SNV) transformation
- ▶ Taking first derivatives wrt wavelength axis



Interesting project topic for someone working with spectra

## Feature extraction: FFT and wavelet features

- ▶ Take FFT and/or wavelet features time-based signals
- ▶ Use these features in **X**, instead of time-domain signals

Some features to consider:

- ▶ Amplitude in the time-domain signal
- ▶ Duration of a certain time-base signal
- ▶ Mean height of FFT peaks
- ▶ Area under certain/all FFT peaks

## Kamyr digester exercise



- ▶ From: <http://www.pulpandpaper-technology.com/contractors/steel/avesta/>
- ▶ Hainan Jinhai Pulp and Paper Co. Ltd., China

## Objective

The **critical quality variable**:  $y$  = Kappa number

- ▶ Higher Kappa number implies more lignin (e.g. cardboard)
- ▶ Lower Kappa number if characteristic of bleachable pulps

Measured at the end of the process

But long delays (over 3 hours) exist between variables that affect  $y$

This creates problems with feedback control and adjustment to keep  $y$  on target.

### Our aim

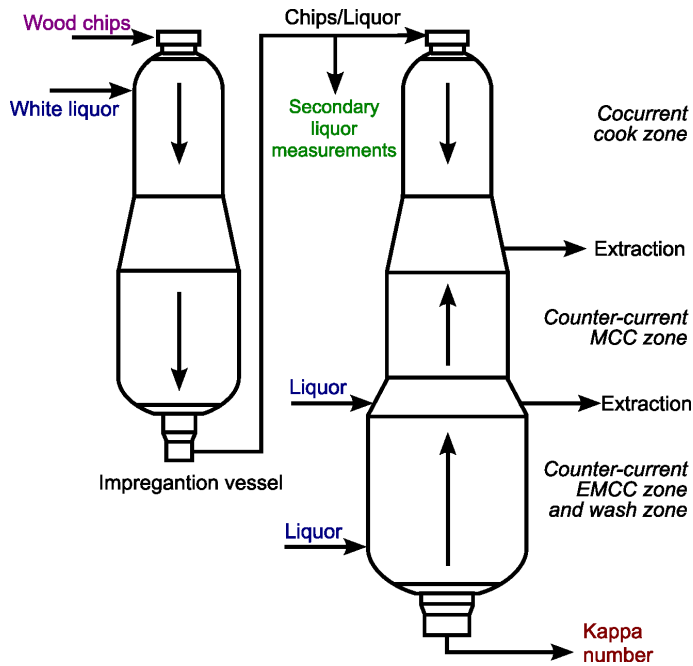
1. Build a good predictive model for Kappa number
2. Understand which variables affect  $y$  = Kappa number, because company wants to reduce variability in Kappa number

# Data available

*More details on next slide also*

- ▶ Data is from a bleach kraft mill in Alberta
- ▶ Slow variables sampled once per hour; faster variables: we have their hourly average
- ▶ Time delays from some  $X$  variables to  $y$  are known, and those columns have been shifted already
  - ▶ ChipLevel14: is the chip level, lagged by 4 hours
  - ▶ BlackFlow2: flow of black liquor, lagged by 2 hours
- ▶ So data are already “aligned” within each row

# Kamyr digester exercise



## Suggested steps

Feel free to examine the data in other ways.

1. Use all the data initially to clean up any outliers.
2. Examine the loadings plot to understand the relationships between variables.
3. Confirm the relationships in the raw data and coefficient plot. Are the interpretations the same?
4. How well are you able to predict Kappa number ( $R_y^2$  and RMSEE)?
  - ▶ Plot an observed vs predicted scatter plot
  - ▶ More usefully: plot observed *and* predicted on a time-series plot. Notice any problems?
5. Add a one hour lag of Kappa to the  $X$ -space. Report how the model has improved its  $R_y^2$  and RMSEE.
6. Try adding a second lag.
7. How would you implement such a model in practice?
8. Use test set split strategy to estimate RMSEP.



## Empirical models: outline for this section

1. Cautions when interpreting empirical models
2. Correlation vs causation
3. Designed experiments are the only way to break the problem

# Models

## Famous quote

“Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful.”

George Box and Norman Draper, *Empirical Model-Building and Response Surfaces*, 1987, p 74.

► Models are useful:

1. we can learn more about a process
2. helpful for troubleshooting (model “before” and “after”)
3. make predictions from them
4. used for monitoring
5. sometimes used to optimize a process, especially cause-and-effect models

# Models

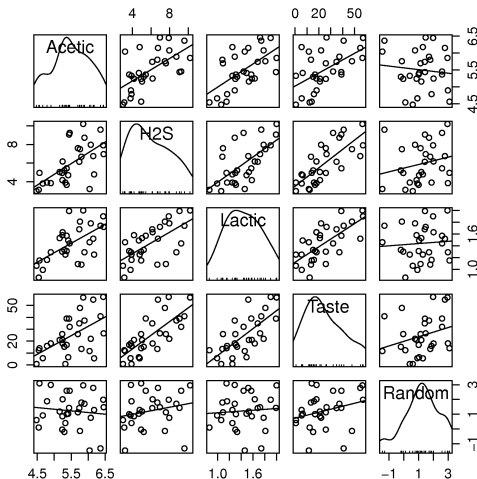
- ▶ Empirical model's **trends** often match reality
  - ▶ though exact values might not always be equal
- ▶ High fidelity first-principles models used with greater certainty
  - ▶ but are more costly to develop than empirical models

## Aim

This section looks at the risks of using empirical models

# Empirical model interpretation

By example: slightly modified **cheese data set**



Correlation matrix:

	Acetic	H2S	Lactic	Taste	Random
Acetic	1.000	0.618	0.6043	0.550	-0.1080
H2S	0.618	1.000	0.6439	0.756	0.2172
Lactic	0.604	0.644	1.0000	0.703	0.0735
Taste	0.550	0.756	0.7035	1.000	0.2790
Random	-0.108	0.217	0.0735	0.279	1.0000

- ▶ 3+1  
X-variables
- ▶ We added a  
random  
variable
- ▶  $y = \text{Taste}$
- ▶  $N = 30$

PCA model on the first 3 X-variables: how many components expected?

## Some models for these data

### Multiple linear regression

$$\hat{y} = -38 + \underbrace{2.2x_{\text{acetic}}}_{-7.3 \leq \beta_{\text{acetic}} \leq 12} + \underbrace{3.4x_{\text{H2S}}}_{1.9 \leq \beta_{\text{H2S}} \leq 6.0} + \underbrace{19x_{\text{lactic}}}_{1.9 \leq \beta_{\text{lactic}} \leq 37} + \underbrace{2.3x_{\text{rand}}}_{-1.2 \leq \beta_{\text{rand}} \leq 5.8}$$

- ▶ RMSEE = 9.4 and  $R^2 = 0.65$
- ▶  $\text{RMSEP}_{TSS} = 11.7 \leftarrow \text{test set switch} = 0.5(11.4 + 11.9)$

### Variable selection: picks H2S

$$\hat{y} = -9.8 + \underbrace{5.8x_{\text{H2S}}}_{3.8 \leq \beta_{\text{H2S}} \leq 7.7}$$

- ▶ RMSEE = 10.5 and  $R^2 = 0.57$
- ▶  $\text{RMSEP}_{TSS} = 11.0$

### Principal components regression:

$$\hat{y} = \text{const} + 8.4x_{\text{acetic}} + 2.4x_{\text{H2S}} + 16x_{\text{lactic}} + 0.6x_{\text{rand}}$$

- ▶ RMSEE = 9.9 and  $R^2 = 0.62$
- ▶  $\text{RMSEP}_{TSS} = 10.6$
- ▶ PCA loading =  $\mathbf{p}_1 = [0.56, 0.59, 0.58, 0.09]$ , explains 72.7%

## Some models for these data

### Projection to latent structures:

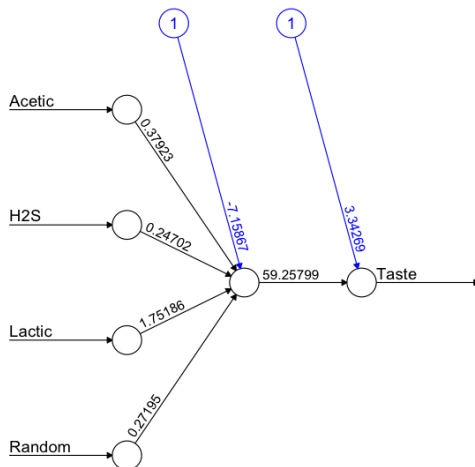
$$\hat{y} = \text{const} + 9.8x_{\text{acetic}} + 2.4x_{\text{H}_2\text{S}} + 13.3x_{\text{lactic}} + 1.8x_{\text{rand}}$$

- ▶ RMSEE = 9.4 and  $R^2 = 0.65$
- ▶ RMSEP<sub>TSS</sub> = 10.7
- ▶ PLS weights =  $\mathbf{w}_1 = [0.46, 0.63, 0.59, 0.23]$

### Neural network model: (also see next slide)

- ▶ RMSEE = 8.8 and  $R^2 = 0.70$
- ▶ RMSEP<sub>TSS</sub> = 14.6
- ▶ Weights: as shown on next page
- ▶ Confidence intervals for weights: all spanned zero

# Neural network models



Error: 1166.392458 Steps: 16725

Trained in R. I will post the R script on the course website; feel free to make a better neural net model - my knowledge about them is limited.

# Which model is appropriate?

For the purposes of

1. learning from the data?
2. troubleshooting?
3. making predictions?
4. process monitoring?
5. optimization?

We will discuss each of the above models in the context of these 5 purposes.



## Process optimization

What if we wanted to improve taste (higher value)? Product development specialist suggests:

- ▶ high lactic acid: 9.0
- ▶ low  $\text{H}_2\text{S}$ : 4.0
- ▶ high acetic acid: 1.80
- ▶ random = 0

Model predictions are:

- ▶ MLR: 29.9
- ▶ OLS: 13.3
- ▶ PCR: 54.3
- ▶ PLS: 48.2
- ▶ NN: 38.8
- ▶ Which model is correct?
- ▶ None of them are correct! The correlation structure is broken.

- ▶ SPE from PCR and PLS will be well above the limits

# Inferring causality

- ▶ Only way to infer causality: designed experiment
  - ▶ Create  $2^3 = 8$  new cheeses
  - ▶ Rebuild model
- ▶ Use your process knowledge to help infer causality
  - ▶ Some empirical models are causal: e.g. spectra related to corn
  - ▶ Digester data set we looked at earlier

# Inferring causality

- ▶ Empirical models: only look at the correlation structure
- ▶ If correlation structure changes: rebuild the model
- ▶ Collinearity between variables:
  - ▶ PLS will spread the weight over all variables
  - ▶ MLR: cannot always invert  $\mathbf{X}'\mathbf{X}$  with extreme collinearity

# Noise in variables

- ▶ All models place less weight on noisy variables

## Section: adaptive methods

Online models are usually either for prediction, or for monitoring.

Why use adaptive models?

- ▶ We may notice a greater number of SPE limit violations
- ▶ Higher number of false alarms
- ▶ Contribution plots less accurate in identifying the variables (monitoring)
- ▶ Can we get better predictions if we
  - ▶ put more weight on recently acquired data
  - ▶ and downweight older data

Another reason for adaptive models:

- ▶ We cannot build our model in one go: too many rows to load in RAM
- ▶ An excellent course project

# What has changed?

Many companies report their online models need regular maintenance (like a car)

But a model is coded in software, which doesn't change. So what has changed?

## **The systems we model are never constant**

- ▶ sensors drift with time, and drift at different rates (slow and fast drift)
- ▶ sensors re-calibrated/replaced: spikes and offsets (very fast)
- ▶ fouling (e.g. pipes and heat exchangers) causes slow changes (couple of months)
- ▶ feedstock change-overs, especially in petrochemical processes (changes over a couple of days)
- ▶ operators change and adjust set-points and operating conditions (fast)

Do you do continual, incremental maintenance. Or do you run a complete rebuild (e.g. buy a new car)?

# First eliminate other potential problems

Before deciding to use adaptive methods, first ensure problems are not due to:

- ▶ Missing values from one or more important (high VIP) variables in the model can cause biased and/or poor predictions
- ▶ Check contributions to ensure you are not in a prolonged abnormal state
  - ▶ Don't want to adapt to abnormal situations

# What's in a model?

To understand what might need to change in the model, we need to know what a latent variable model consists of.

- ▶ Preprocessing: centering and scaling vectors
- ▶ Model parameters: **P**, **W**, **C**,
- ▶ Model limits for  $T^2$  and SPE
- ▶ Number of components,  $A$



## Adaptive preprocessing: centering

Update the centering vector using a moving window of width  $w$  samples

- ▶  $\bar{x}_t = \frac{1}{w}x_{t-1} + \frac{1}{w}x_{t-2} + \dots + \frac{1}{w}x_{t-w}$
- ▶ Ensure width is wide enough to avoid outlier bias
- ▶ Better still: use an adaptive median

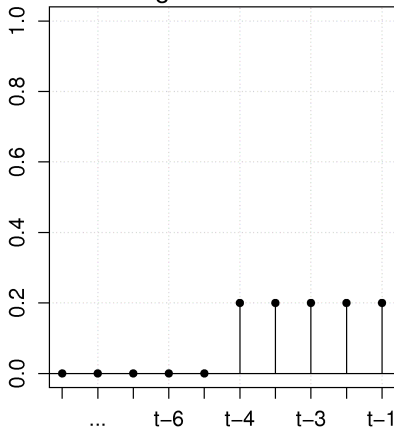
## Adaptive preprocessing: centering

To discount older data in the centering:

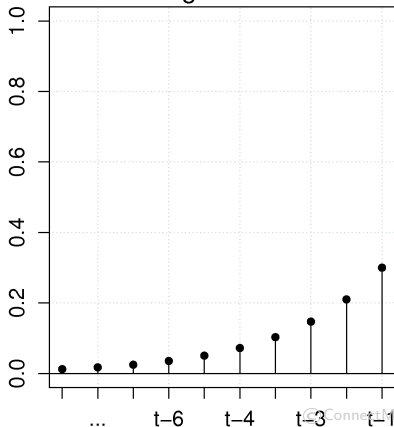
### Exponentially Weighted Moving Average (EWMA)

- ▶ heavier weights for recent observations
- ▶ small weights further back

MA weights when width=5



EWMA weights when  $\lambda = 0.3$



## Adaptive centering

For a single tag in the model (either  $x$ - or  $y$ -variable), but we will use the notation  $x_t$

$x_t$  = value of the tag at time  $t$

$m_t$  = adaptive mean as used at time  $t$

$m_{t+1} = m_t + \lambda(x_t - m_t)$

$m_{t+1} = (1 - \lambda)m_t + \lambda x_t$

- ▶  $m_0$  = median (or mean) of the training data
  - ▶  $\lambda = 0$  implies we don't update that variable
  - ▶ We update that tag faster as  $\lambda \rightarrow 1$
  - ▶ You should use different  $\lambda$  values for different tags
- 
- ▶ How do we determine  $\lambda$ ?
  - ▶ How do we prevent outliers from influencing us?

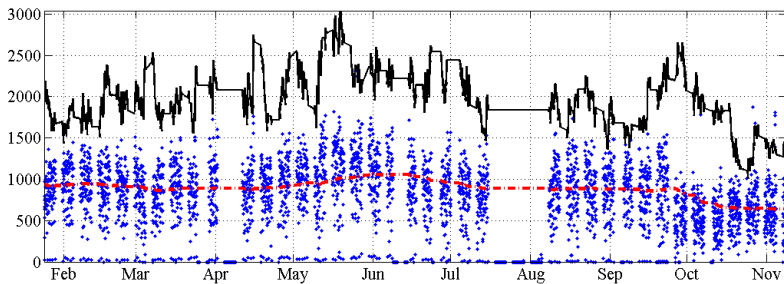
## Adaptive scaling

Models are usually less sensitive to scaling. But if adapted, use the **Exponentially Weighted Moving Variance (EWMV)**

$$\begin{aligned}x_t &= \text{value of the tag at time } t \\s_t &= \text{adaptive scaling as used at time } t \\s_{t+1} &= \sqrt{(1 - \alpha)s_t^2 + \alpha(x_t - m_t)^2}\end{aligned}$$

- ▶  $s_0$  = scaling vector of the training data
- ▶  $\alpha = 0$  implies no scaling update
- ▶ We update faster as  $\alpha \rightarrow 1$
- ▶ You can use different  $\alpha$  values for different tags, but probably easier to just use same  $\alpha$
- ▶ Note that you adapt the scaling around the moving center

## Example



- ▶ **Dashed**: adaptive mean centering values
- ▶ **Solid**: adaptive scaling values
- ▶ **Dots**: raw data
- ▶ Notice the outliers around 0.0

Scaling values modified to fit on same plot axes

# Model updating

The preprocessing (centering and scaling) updates will fix most drift and adaptive problems.

But if the *correlation structure* between the variables is *also* changing, then you must adapt the model.

We will consider a least squares model update first, then look at PCA and PLS.

# Recursive Least Squares

Suppose we have a model:  $y = \mathbf{b}\mathbf{x}$  at time  $t - 1$

At time  $t$  we can make a prediction  $\hat{y}_t = \mathbf{b}\mathbf{x}_t$  given a new  $\mathbf{x}_t$

Later we obtain the true  $y_t$ . So now we want to update the model:  
i.e. **re-estimate the value of  $b$**

The equations to re-estimate  $b$  without requiring the previous training data are called *recursive least squares equations*:

- ▶  $\mathbf{b}_t = \mathbf{b}_{t-1} + \mathbf{K}_t \mathbf{e}_t$  = updated  $\mathbf{b}$  at time  $t$
- ▶  $\mathbf{e}_t = y_t - \mathbf{x}_t \mathbf{b}_{t-1}$  = prediction error at time  $t$
- ▶  $\mathbf{K}_t = \frac{\mathbf{L}_{t-1}}{\lambda + \mathbf{x}_t \mathbf{L}_{t-1} \mathbf{x}_t} \mathbf{x}_t^T$  = gain matrix
- ▶  $\mathbf{L}_t = \frac{(\mathbf{I} - \mathbf{K}_t \mathbf{x}_t) \mathbf{L}_{t-1}}{\lambda}$

and we initialize with  $\mathbf{L}_{t=0} = (\mathbf{X}^T \mathbf{X})^{-1}$ .

$\lambda$  = discounting factor, where  $\lambda = 1.0$  no discounting.

## Aside: Keep lab's $y$

After learning about soft sensors, you might be tempted to remove the (expensive) mechanism used to acquire  $y$ .

- ▶ Rather just reduce the frequency of using the lab
- ▶ Use your soft sensor to get predictions at a higher rate
- ▶ We will need the lab's  $y$ -values later:
  - ▶ track  $SPE_y = (y_{lab} - \hat{y})$  to see if it grows over time: indicates model needs rebuilding
  - ▶  $SPE_y$  has a confidence limit: from the training data
  - ▶ You will need the true  $y$ -values to rebuild model or for model adaption



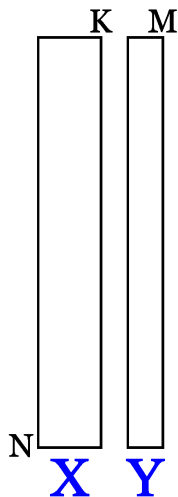
## Recursive PCA and PLS

Based on the least squares recursive equations, is it possible to get:

$$(\text{LVmodel})_{t+1} = \mu(\text{LVmodel})_t + (1 - \mu)(\text{new data}_t)$$

**AIM:** Update our model with new data from the process

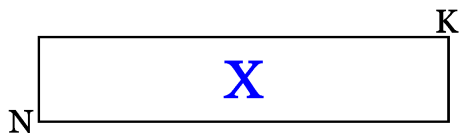
# Kernel matrices and kernel methods



$$X^T X \quad \begin{matrix} K \\ K \end{matrix}$$

$$X^T Y \quad \begin{matrix} M \\ K \end{matrix}$$

$$X^T Y Y^T X \quad \begin{matrix} K \\ K \end{matrix}$$



$$X X^T \quad \begin{matrix} N \\ N \end{matrix}$$

# Kernel matrices and kernel methods

We can create the kernel matrices easily when there are no missing values

**PCA:**

$$\begin{aligned}\mathbf{P} &= \text{eigenvectors of } \mathbf{X}^T \mathbf{X} \\ \mathbf{T} &= \mathbf{X} \mathbf{P} \\ &\quad (\text{use only the first } A \text{ eigenvectors})\end{aligned}$$

**PLS:**

$$\mathbf{w} = \text{eigenvectors of } \mathbf{X}^T \mathbf{Y} \mathbf{Y}^T \mathbf{X}$$

remaining PLS model parameters can be found (follow NIPALS algorithm for 1 iteration)

- ▶ See [Dayal and MacGregor](#): Recursive PLS
- ▶ and [Dayal and MacGregor](#): Improved PLS algorithms

# Updating the latent variable model

Update the model with new data:

$$\begin{aligned}(\mathbf{X}^T \mathbf{X})_{t+1} &= (1 - \mu)(\mathbf{X}^T \mathbf{X})_t + \mu \mathbf{x}_{\text{new}}^T \mathbf{x}_{\text{new}} \\ (\mathbf{X}^T \mathbf{Y})_{t+1} &= (1 - \mu)(\mathbf{X}^T \mathbf{Y})_t + \mu \mathbf{x}_{\text{new}}^T \mathbf{y}_{\text{new}}\end{aligned}$$

giving relative weight of  $\mu$  to the new data.

- ▶ Calculate new model loadings and weights from updated covariance matrices
- ▶ This gives declining weight to older information
- ▶ Higher values of  $\mu$  give more weight to recent data
- ▶ Adapt too fast? “lose” memory of old operation
- ▶ Adapt too fast? monitoring model adapts to abnormal operation and doesn't alarm
- ▶ Adapt too slowly? predictions are still poor

## Reintroduce some original variation

To counteract too-fast adaption, and keep model relevant:

$$\begin{aligned}(\mathbf{X}^T \mathbf{X})_{t+1} &= (1 - \mu)(\mathbf{X}^T \mathbf{X})_t + \mu \mathbf{x}_{\text{new}}^T \mathbf{x}_{\text{new}} + f_x \cdot (\mathbf{X}^T \mathbf{X})_{t=0} \\ (\mathbf{X}^T \mathbf{Y})_{t+1} &= (1 - \mu)(\mathbf{X}^T \mathbf{Y})_t + \mu \mathbf{x}_{\text{new}}^T \mathbf{y}_{\text{new}} + f_y \cdot (\mathbf{X}^T \mathbf{Y})_{t=0} \\ f_x &= \gamma \frac{\|\mu \mathbf{x}_{\text{new}}^T \mathbf{x}_{\text{new}}\|}{\|(\mathbf{X}^T \mathbf{X})_{t=0}\|} \\ f_y &= \gamma \frac{\|\mu \mathbf{x}_{\text{new}}^T \mathbf{y}_{\text{new}}\|}{\|(\mathbf{X}^T \mathbf{Y})_{t=0}\|}\end{aligned}$$

The  $\gamma$  tuning value is the amount by which we “re-introduce” original variation into the model. We introduce it in proportion to the ratio of new vs old information.

**Reference available on request:** Dunn et al., Unpublished research paper

# Tuning the adaptive parameters

How to pick:  $\lambda$ ,  $\alpha$ ,  $\mu$  and  $\gamma$ ?

Use a grid search, or a derivative-free optimization technique to achieve your objective. For example:

- ▶ 2003 = build data set
- ▶ 2004 = testing data set

Suitable objectives for the optimizer?

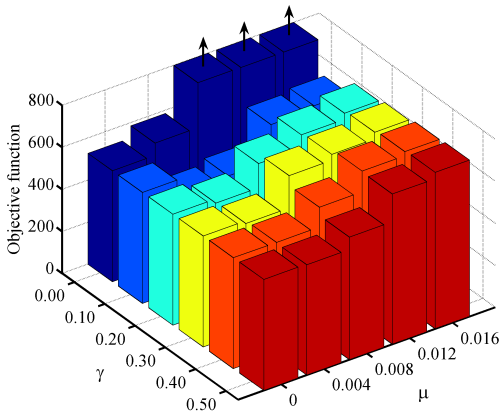
- ▶ Monitoring: minimize the type I and type II errors in 2004
- ▶ Prediction: minimize RMSEP on 2004 data

by varying the adaption parameters

# Tuning the adaptive parameters

All adaption parameters can be varied to “tune” adaptive model

But you don't want too many free variables in your grid search



- ▶ Pick  $\lambda$  and  $\alpha$  so that centering and scaling tracking look good visually
- ▶ Vary  $\mu$ : update rate for covariance matrices
- ▶ Vary  $\gamma$ : rate at which original data is re-introduced into covariance matrices

## How much has the model changed?

We want to avoid adapting too fast (which will lead us to track transients), or too slow.

So it's helpful to know how much our model has changed from the base case. I developed this metric in 2003 to 2005 on an adaptive monitoring project.

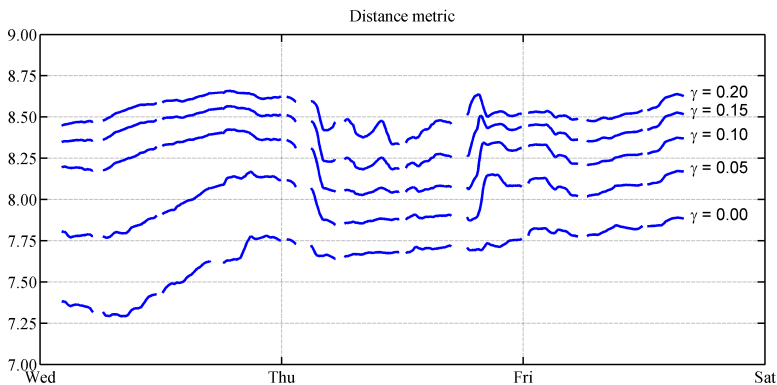
Uses the sum of squares of angle cosines between all loading vectors (inspired by [Krzanowski's paper](#))

$$\begin{aligned}d_{t,\text{PCA}} &= \text{trace}(\mathbf{P}_B^T \mathbf{P}_t \mathbf{P}_t^T \mathbf{P}_B) \\d_{t,\text{PLS}} &= \text{trace}(\mathbf{W}_B^T \mathbf{W}_t \mathbf{W}_t^T \mathbf{W}_B)\end{aligned}$$

- ▶ Minimum distance =  $d_t = 0$ : current loadings,  $\mathbf{P}_t$  are completely orthogonal to base model's loadings,  $\mathbf{P}_B$
- ▶ Maximum =  $d_t = A$ : current loadings are completely aligned with base model's loadings (but may be in different order)

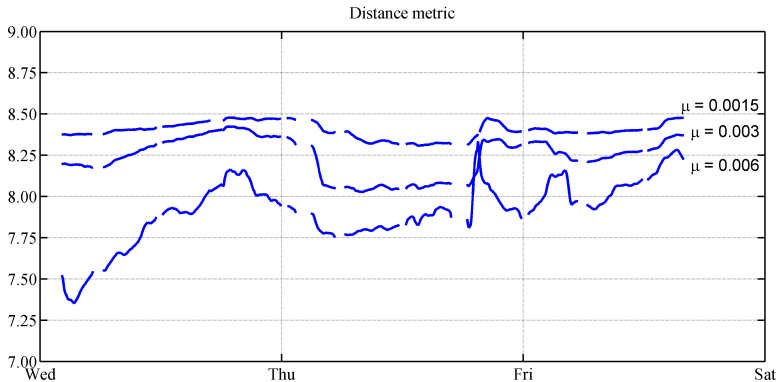


## Distance metric example ( $A = 9$ )



Gaps are during out-of-control operation: so no updating occurred

## Distance metric example ( $A = 9$ )



- ▶ Notice the more sluggish adaption for smaller values of  $\mu \approx 0.0015$
- ▶ Want to avoid rapid updating, especially in monitoring systems. Rate of change of  $d_t$  on Thursday night for  $\mu = 0.006$  is too fast for a reasonable monitoring system

## Adaptive model limits

- ▶ Limits for  $T^2$  will not change, if  $A$  is fixed
- ▶ Can be adjusted, based on the eigenvalues from the covariance matrix

## Tip: knowing what to adapt

Example:

- ▶ Build a model on 2003 data
- ▶ Apply it on data from 2004 and realize that predictions/monitoring gets poorer with time
- ▶ Build a model on 2003 and 2004 data. Compare:
  - ▶ centering and scaling
  - ▶ model weights and loadings (use the distance metric)
  - ▶ model limits
  - ▶ number of components

Seeing what has changed will guide you on what parts of the model need to adapt.

## Other tips: avoiding bad adaption

- ▶ Screen incoming data for outliers, especially for updating the centering vector
- ▶ Don't adapt when the observation has very high SPE and very high  $T^2$ , e.g. above 99% limits
- ▶ Conversely: **no need to adapt** if SPE and  $T^2$  are below, say, 50% limit: no new information coming in
- ▶ Adapt in a mid-region, e.g. between 50% and 95% limits
- ▶ You can speed up adaption rate for centering and scaling after a plant shutdown to rapidly get the model back on track

# References

- ▶ Good review paper: Kadlec et al., 2011
- ▶ Recursive PLS: Dayal and MacGregor
- ▶ Improved PLS algorithms: Dayal and MacGregor
- ▶ Adaptive PCA and PLS: Wold
- ▶ Other papers: related to kernel methods