Introduction to Soft Sensors

- I want to know how many inches of rain fell today
- Easy to count how many people brought an umbrella to class

Procedure:

1. Collect historical data (records of rainfall and number of people carrying umbrellas
2. Build a model between two data sets
3. For following days, number of people carrying umbrellas can tell me how much rain fell

The problem

Certain measurements/final properties [called y's] are

1. expensive, and/or
2. time-consuming

to obtain from many practical systems.

Aim

Get a prediction of y cheaply and quickly

How can we get predictions?

First-principles models are obviously more desirable.

- based on cause and effect
- can be used to later improve and optimize the process also

But ....

- may be costly to identify the values of parameters in the model
- ... impossible even
- high-fidelity models may take a long time to execute: too slow for real-time use

all of which negate their usefulness.

You might see any number of other models in practice: Kalman filters, neural networks, Bayesian networks, *etc*.

Principle and motivation

Principle

Use **any predictive model** to predict the value of interest, $\hat{y}$, from any other available data, $\mathbf{X}$

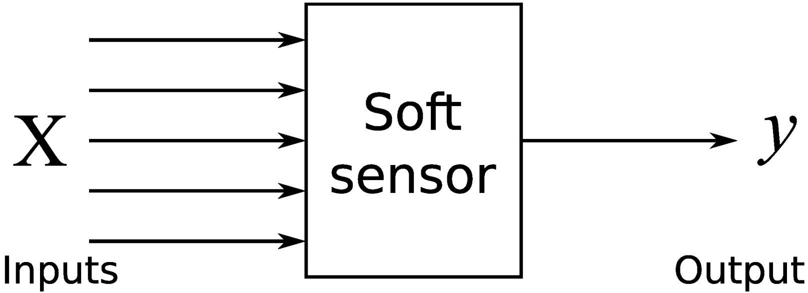
This predictive model is called:

- a soft sensor, or
- an inferential sensor

These predictions are mainly used for:

- monitoring of that $\hat{y}$
- feedback control to maintain $\hat{y}$ at target

Soft sensor structure



Value of interest, y, should ideally be measured on the process

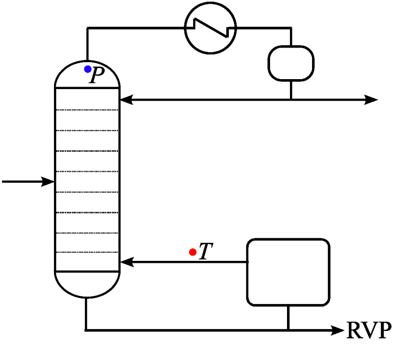
Data used in **X**:

- real-time process measurements
- calculated variables from the real-time measurements
- spectra, acoustical data, image data
- categorical variables

... anything that is readily available as input into the model to predict y

Example: distillation column purity prediction

- Predict purity from distillation column bottom = y = RVP
- Inputs to model:
  - Various temperatures on the column
  - Pressure measurements
  - Flow rates through the column (column demand)
  - External factors: ambient temperature



Example: distillation column purity prediction

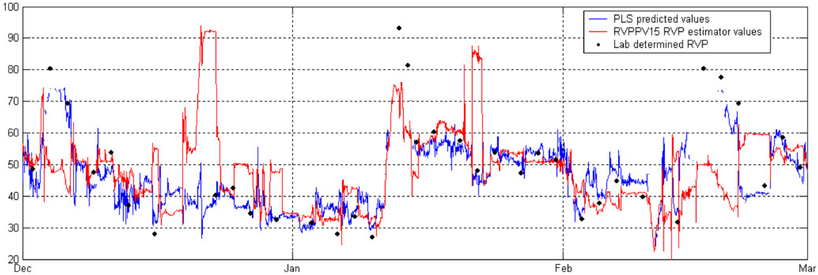
We know the Antoine equation applies here:

$$\log(\text{RVP}) = \log P - B\left[\frac{1}{C} - \frac{1}{T+C}\right]$$

Actual use: $\log(\text{RVP}) = \log P - B\left[\frac{1}{C} - \frac{1}{T+C}\right] + \text{bias}$

Bias is updated 3 times per week to "correct" predictions (accuracy)

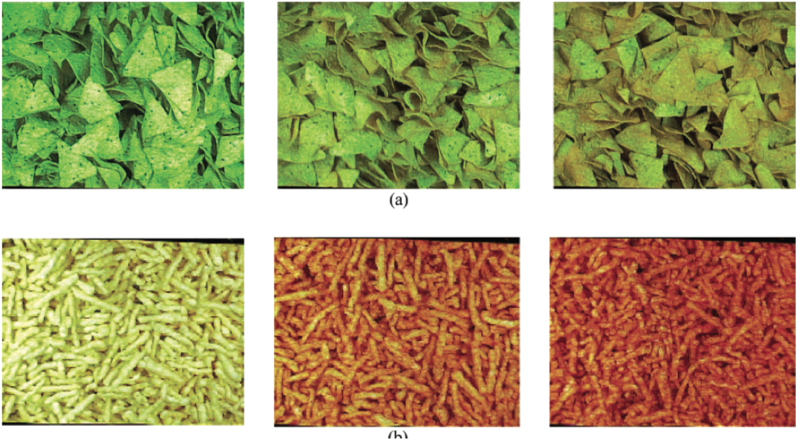
Example: distillation column purity prediction



- `RVPPV15 RVP` is the Antoine estimate
- It occasionally gives really poor predictions. *Why*?

Example: snack food seasoning prediction

- Work by Honglu Yu: http://digitalcommons.mcmaster.ca/opendissertations/866/
- Image data used as **X**



(a)

(b)

Example: snack food seasoning prediction

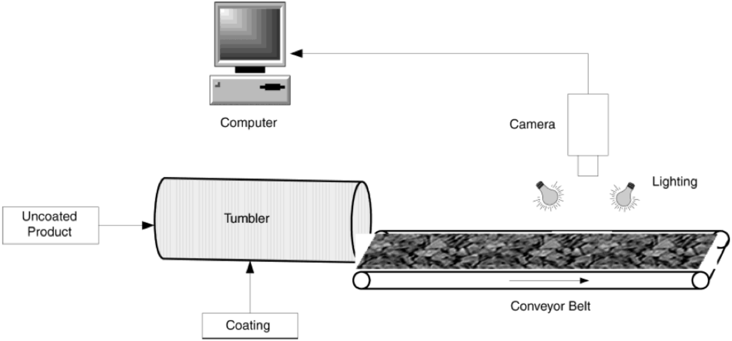
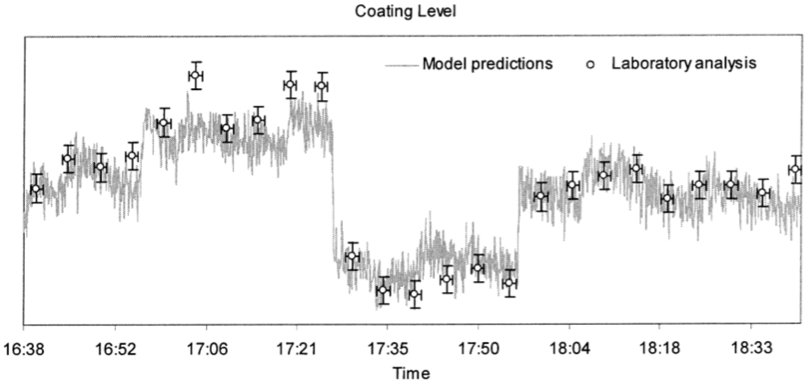


Figure 8. Schematic of the processes and imaging systems.

See http://dx.doi.org/10.1021/ie020941f

Example: snack food seasoning prediction

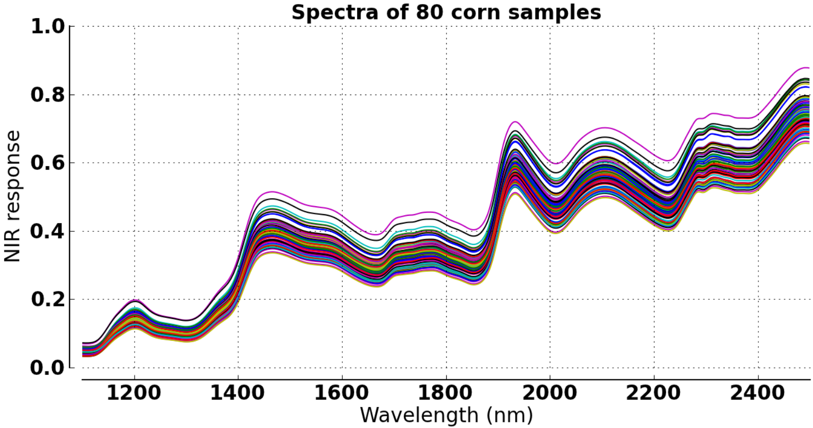


See http://dx.doi.org/10.1021/ie020941f

We will go into the details about this application in the section on "Image Processing"

Prediction of multiple properties of corn from NIR

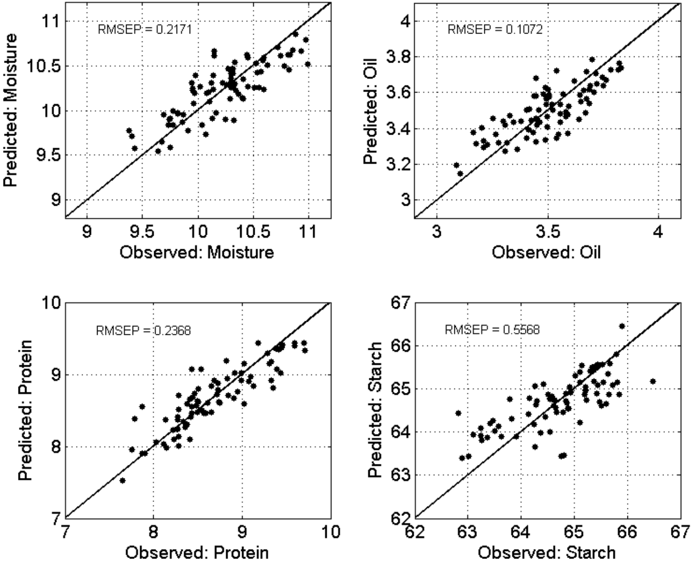
Source data are NIR spectra **X**. There are 4 properties: y



Source: http://eigenvector.com/data/Corn/index.html

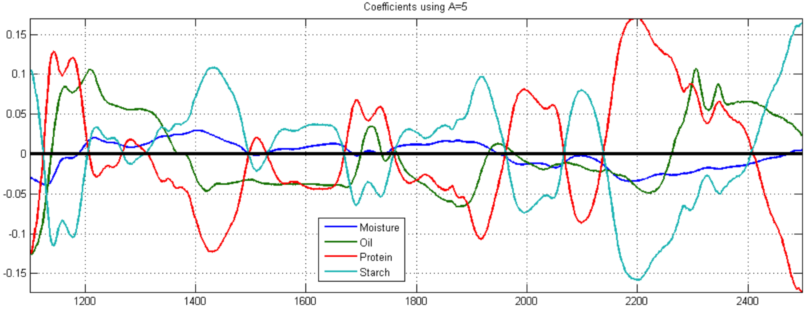
Prediction of multiple properties of corn from NIR

Prediction ability with 5 components:



Prediction of multiple properties of corn from NIR

Coefficients, **b**, for prediction: $y = \mathbf{x}_{\text{new}}\mathbf{b}$


Coefficients using A=5

Further advantages of soft sensors

Advantages include:

- That **X** data is more frequently available than **y** (real-time)
- **X** often has greater accuracy than the lab and sampling error in **y**
- Get a free process monitoring check from SPE and T^2 if you use PLS (not for other prediction models)
- Can handle missing values in **X** with PLS
- A model such as PCR or PLS will handle collinearity in **X**
- We also would like a realistic confidence interval for our prediction

Inferential predictions are only a *supplement* to the more expensive/time-consuming way of getting to the value of interest.

- We can reduce laboratory costs
- But, we should never eliminate the lab values though!

Calibration vs soft-sensors

Sometime you might see the term “multivariate calibration”

- Calibration: implies model is built from samples which are specifically acquired for the purpose of building the predictive model
- Soft sensors: built from historical, happenstance data (i.e. regular day-to-day data)

Calibration data is collected with the intention of building a predictive model.

Good reference: Martens and Næs

Concepts

Like we saw in process monitoring, there are 2 phases to building a soft sensor

1. Model building phase (training)
2. Testing the model's performance

Then you can go ahead and apply it on-line.

Concepts: judging model's performance

- **RMSEE** = root mean square error of estimation = $\sqrt{\frac{1}{N} \cdot \sum_i^N \left(y_i - \hat{y}_i\right)^2} \quad \longleftarrow$ when building the model
- **RMSEP** = root mean square error of prediction = $\sqrt{\frac{1}{N} \cdot \sum_i^N \left(y_{i,\text{new}} - \hat{y}_{i,\text{new}}\right)^2} \longleftarrow$ testing model on new data
- Bias = average offset = $\dfrac{1}{N} \cdot \left(\displaystyle\sum_i^N y_{i,\text{new}} - \hat{y}_{i,\text{new}} \right)$

These 3 metrics are easy to use: they are all in the original units of y, and we want them as small as possible.

Concepts: judging model's performance

Ideally: we have a completely separate test set for which we *minimize* its prediction error.

- Build predictive model
- Use model on testing set and calculate residuals
- Are residuals small enough? Acceptable overall performance?

We can't always afford a testing set, so the next best option: **test set switch**

- Split data in half
- Build model on part 1, test on part 2 to calculate RMSEP_2
- Build model on part 2, test on part 1 to calculate RMSEP_1

Once finished:

- Calculate average RMSEP: aim to minimize this.
- Finally use model built from all data.

Exercise

Time for a calibration exercise ("sawdust")

Cross-validation for soft-sensors

Why is cross-validation **not suitable** for *soft-sensor* models?

- Can it be made more suitable?

Helpful references on soft sensors

1. Systematic and robust procedure described by: Lin *et al.* (paper 107)
2. Kresta, Marlin and MacGregor: Development of inferential process models using PLS
   - Describes some cautions when using soft sensors

Systematic procedure to **build** a soft sensor (phase 1)

1. Start with available data that is representative of process operation
2. **Remove outliers***: plant shutdowns, gross outliers
3. Build initial model
4. Remove outliers in T^2 and SPE
5. Rebuild model (may need to iterate back to step 2)
6. From here on: aim is to maximize RMSEP on a completely separate testing set
7. What data transformations might improve predictions? See next section on "Advanced Preprocessing"
8. Drop out noisy variables (small weights, small VIP)
9. Iteratively build your models until you get desirable performance
10. Your final model should make physical sense
    - known important variables should have large weights

* Try to ensure you can use the same cleaning routine on-line (phase 2)

Online procedure for a software sensor I

Phase 2: using a soft sensor online

1. Collect the input variables required
2. Do univariate cleaning and checks (the same as when building the model)
3. Perform any calculations, feature extraction, etc, on variables that will create new columns
4. You should now have assembled your vector: $x_{\text{new, raw}}$ (might have missing values)
5. Preprocess this vector (the same as when building the model), to get $\mathbf{x}_{\text{new}}$
6. Project this vector on to the latent variable directions, $\mathbf{W}^*$, to get $\mathbf{t}_{\text{new}} = \mathbf{x}_{\text{new}}\mathbf{W}^*$
7. Calculate projected $\hat{\mathbf{x}}_{\text{new}} = \mathbf{t}_{\text{new}}P'$
8. Calculate residuals: $\mathbf{e}_{\text{new}} = \mathbf{x}_{\text{new}} - \hat{\mathbf{x}}_{\text{new}}$

Online procedure for a software sensor II

9. Calculate SPE from $\mathbf{e}_{\text{new}}$
   - Below the limit: *continue*
   - If not: investigate contributions* to diagnose problem; *you really shouldn't continue*
10. Calculate Hotelling's T^2 from $\mathbf{t}_{\text{new}}$
    - Below the limit: *continue*
    - If not: investigate contributions* to diagnose problem; *continue with caution* if not a huge outlier (subjective)
11. Make your predictions: $\hat{\mathbf{y}}_{\text{new}} = \mathbf{t}_{\text{new}} C'$
12. Un-preprocess your predictions back to real-world units
13. Check your prediction quality:
    - Run your prediction model in parallel to lab values initially
    - Check model predictions against lab values to determine if it needs updating.

* **Note**: a useful procedure if only a handful of variables are flagged in the contributions: set them as missing values and repeat the prediction. If this doesn't work, then you will need to rebuild the soft sensor.

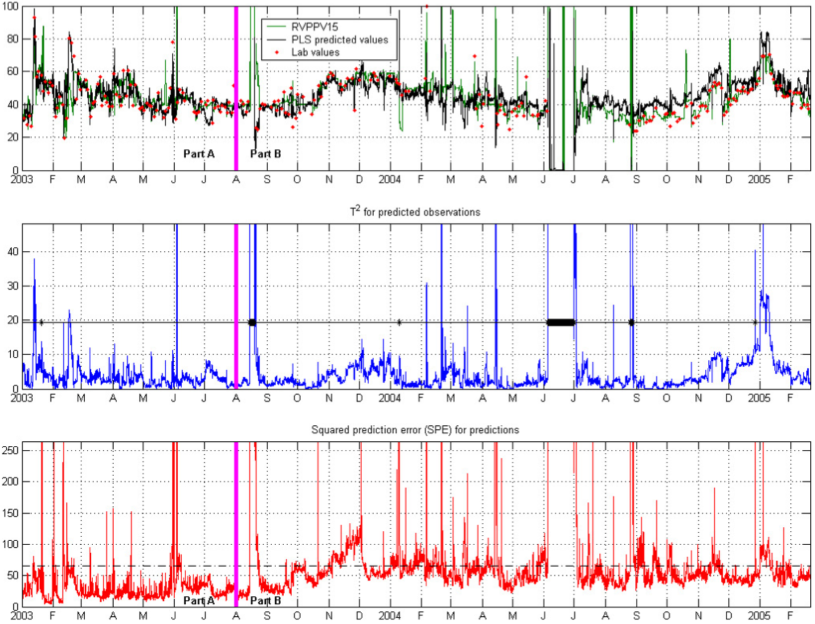
Dealing with poor online prediction performance

Predictions from the model are valid as long as the correlation structure is consistent.

- Track SPE: above the 95% limit indicates that things have changed (see next slide)
- Track T^2: increased throughput? Adjust mean-centering and perhaps the scaling
- Drift in some variables*: adjust mean-centering vector
- Still not solved: may need to rebuild model, especially after major events (plant shutdown and clean; new equipment installed)

This is another reason why we

* We will deal with this topic in a later class on adaptive kernel methods



Degraded performance later on (part B).

Caution: Missing values

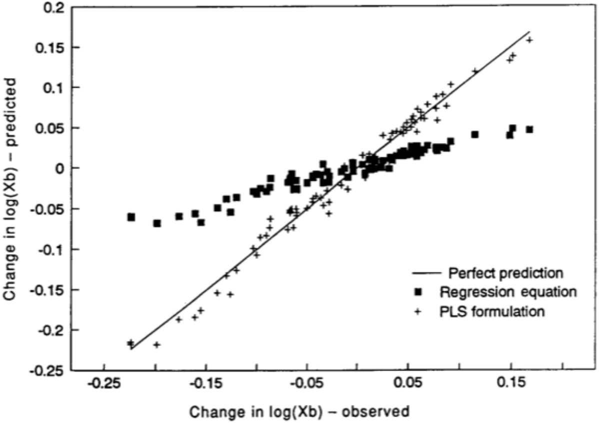
- Sensors go offline from time-to-time
- Often missing data are systematic: uneven sampling frequency between variables

The last case should be carefully checked:

- Try different missing value algorithms (project topic!)
- Build model only on rows where most data is available and compare to full model.

Caution: Missing values

- From paper by Kresta, Marlin and MacGregor



- Regression model replaced missing value by the mean
- PLS model used usual missing data handling

Caution: Have adequate variation in training data

- Have adequate variation in the data

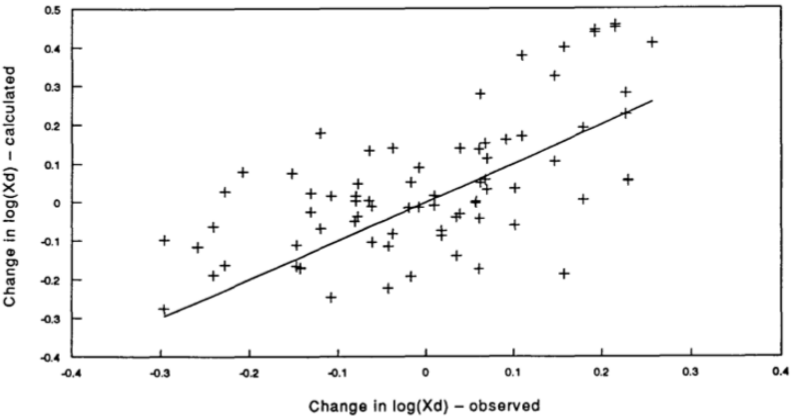


Fig. 6. Prediction of the methanol composition in the distillate for the MAW column using a model developed from a data set containing only variation in the manipulated variables. This shows the degradation of the prediction when an improper data set is used.

Caution: Have adequate variation in training data

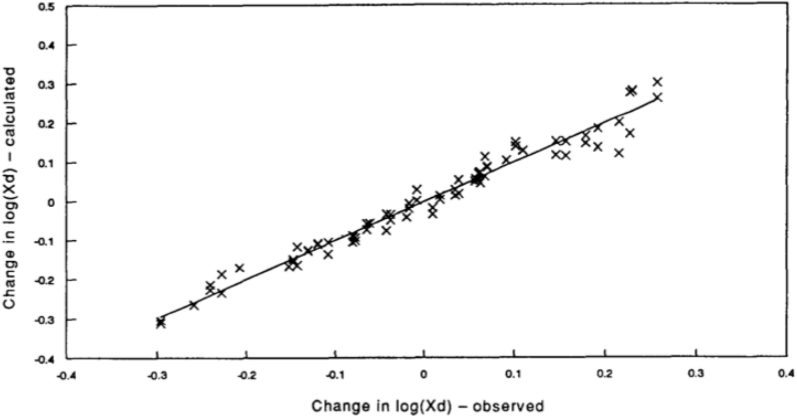
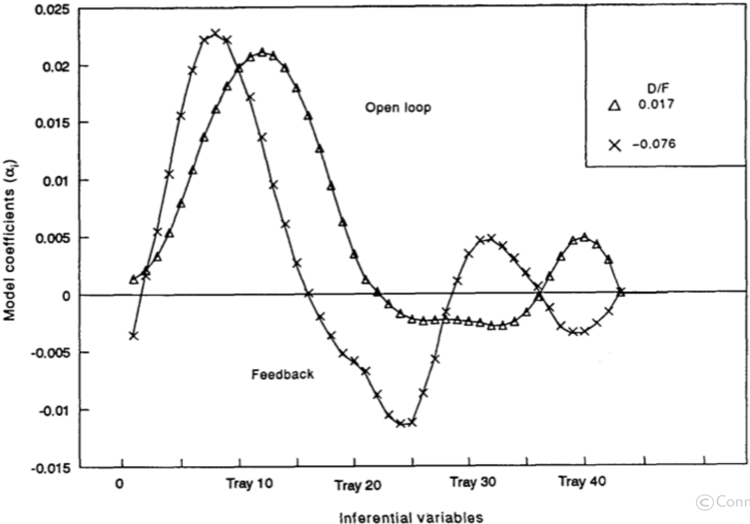


Fig. 5. Prediction of the methanol composition in the distillate for the MAW column using a model developed from a proper data set, one containing typical variation of both manipulated and disturbance variables.

- The model-building data must represent all expected conditions under which it will be applied.

Caution: Dealing with feedback control

- Feedback control changes the correlation structure among variables
- Recall: if correlation structure changes - **rebuild model**


Model coefficients (α_i)
Open loop
Feedback
D/F
△ 0.017
× -0.076
Tray 10
Tray 20
Tray 30
Tray 40
Inferential variables