Hydroinformatics for hydrology: data-driven and hybrid techniques

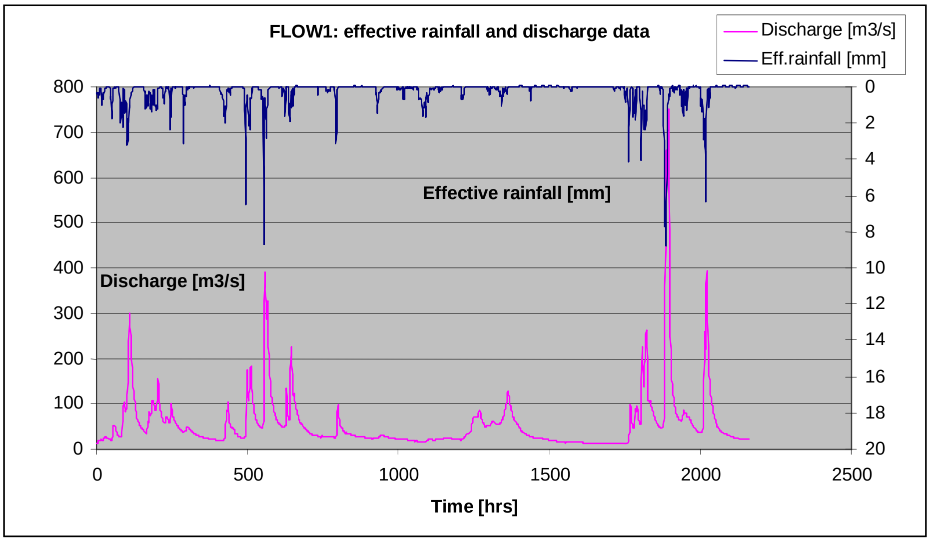
Dimitri P. Solomatine

UNESCO-IHE Institute for Water Education
Hydroinformatics Chair

Outline of the course

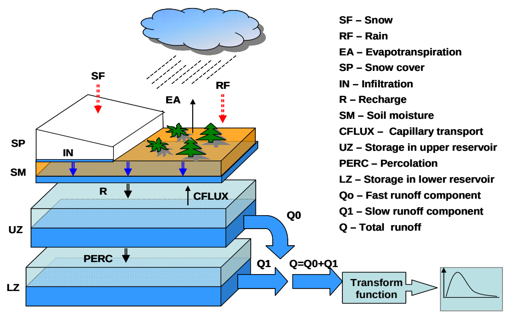
- Notion of data-driven modelling (DDM)
- Data
- Introduction to some methods
- Combining models - hybrid models
- Demonstration of applications
  - Rainfall-runoff modelling
  - Reservoir optimization

Quick start: rainfall-runoff modelling

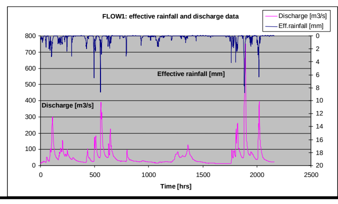


Quick start: how to calculate runoff one step ahead?

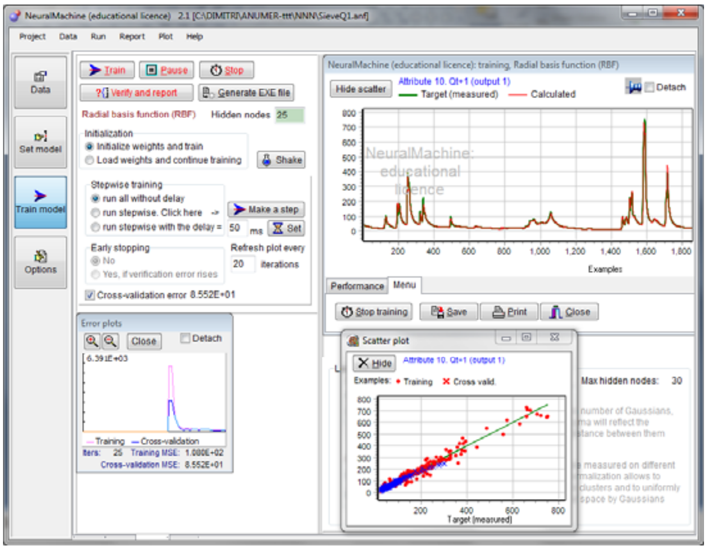
- A. Lumped conceptual model



- B. **Data-driven (regression) model**, based on past data:
- $Q\,(t+1) = f\,(RE_t, RE_{t\text{-}1}, RE_{t\text{-}2}, RE_{t\text{-}3}, RE_{t\text{-}4}, RE_{t\text{-}5}, \; Q_t, Q_{t\text{-}1}, Q_{t\text{-}2})$
- where f is a non-linear function



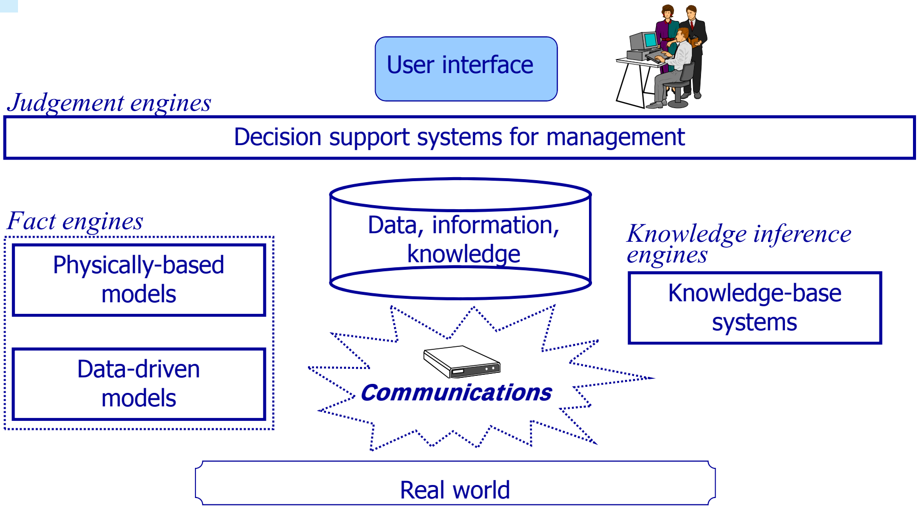
Quick start: Neural netowrk model of the rainfall-runoff relationship (NeuralMachine)



Why data-driven now?

- Measuring campaigns using automatic computerised equipment: a lot of data became available
- important breakthroughs in computational intelligence and machine learning methods
- penetration of “computer sciences” into civil engineering (e.g., hydroinformatics, geo-informatics etc.)

Hydroinformatics system: typical architecture

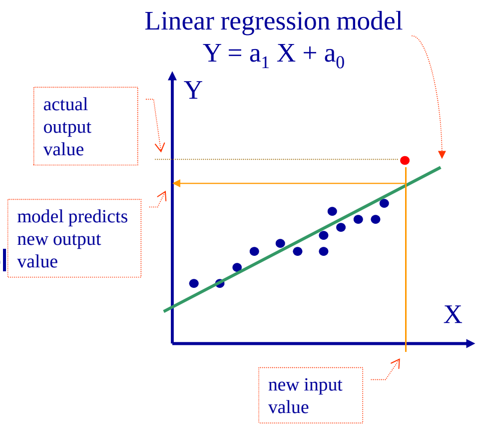


Modelling

- Model is …
  - a simplified description of reality
  - an *encapsulation of knowledge* about a particular physical or social process in electronic form
- Goals of modelling are:
  - understand the studied system or domain (understand the past)
  - predict the future
    - predict the future values of some of the system variables, based on the knowledge about other variables
  - use the results of modelling for making decisions (change the future)

Example of a simple data-driven model

- independent variable X (input) and dependent variable Y (output)
- linear regression roughly describes the observed relationship
- parameters a_1 and a_2 are unknown and are found by feeding the model with data and solving an optimization problem (training)
- the model then predicts output for the new input without actual knowledge of *what* drives Y

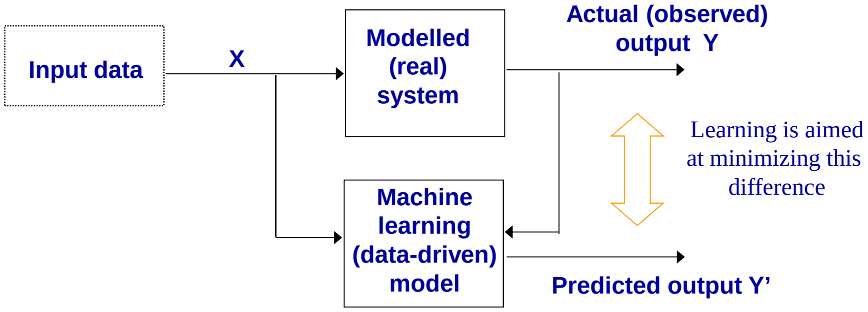


Data: attributes, inputs, outputs

- set *K* of examples (or instances) represented by the duple <$\mathbf{x}_k$, $\mathbf{y}_k$>, where $k = 1,\ldots, K$, vector $\mathbf{x}_k = \{x_1,\ldots,x_n\}_k$, vector $\mathbf{y}_k = \{y_1,\ldots,y_m\}_k$, n = number of inputs, m = number of outputs.
- The process of building a function ('model') $\mathbf{y} = f(\mathbf{x})$ is called *training*.
- Often only one output is considered, so $m = 1$.

Measured data	**Attributes**					**Model output**
	Inputs				*Output*	$y^* = f(\boldsymbol{x})$
Instances	$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	…	$\boldsymbol{x}_n$	$\boldsymbol{y}$	$\boldsymbol{y}^*$
Instance 1	x_{11}	x_{12}		x_{1n}	y_1	y_1*
Instance 2	x_{21}	x_{21}		$x_{2\,n}$	y_2	y_2*
…						…
Instance K	x_{K1}	x_{K2}		$x_{K\,n}$	y_K	y_K*

Data-driven model



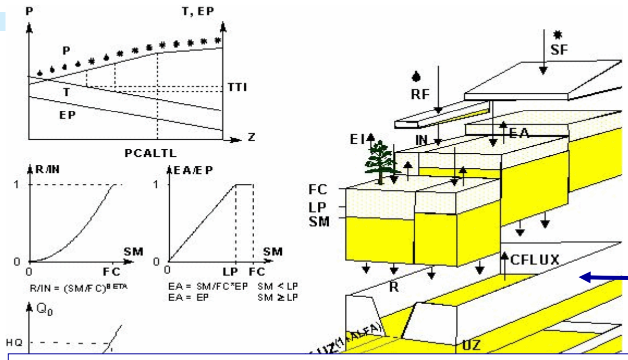
- DDM "learns" the *target function* Y=*f* (X) describing how the real system behaves
- Learning = process of minimizing the difference between observed data and model output. X and Y may be non-numeric
- After learning, being fed with the new inputs, DDM can generate output close to what the real system would generate

Data-driven models vs Knowledge-driven (physically-based) models (1)

- "Physically-based", or "knowledge-based" models are based on the understanding of the underlying processes in the system
  - examples: river models based on main principles of water motion, expressed in differential equations, solved using finite-difference approximations
- "Data-driven" model is defined as a model connecting the system state variables (input, internal and output) without much knowledge about the "physical" behaviour of the system
  - examples: regression model linking input and output
- Current trend: combination of both ("hybrid" models)

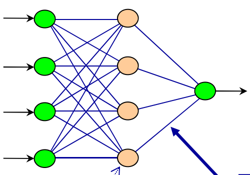
Data-driven models vs Physically-based models

physically-based (conceptual) hydrological rainfall-runoff model



$$P - E - Q = \frac{d}{dt}[SP + SM + UZ + LZ + lakes]$$

Coefficients to be identified by calibration

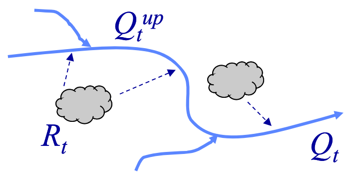


connections with weights to be identified by training

data-driven rainfall-runoff model (artificial neural network)

Using data-driven methods in rainfall-runoff modelling

- Available data:
  - rainfalls R_t
  - runoffs (flows) Q_t
- Inputs: lagged rainfalls $R_t\ R_{t-1} \ldots R_{t-L}$
- Output to predict: Q_{t+T}



- Model: $Q_{t+T} = F\ (R_t\ \ R_{t-1} \ldots R_{t-L} \ldots Q_t\ \ Q_{t-1}\ Q_{t-A} \ldots Q_t^{up}\ Q_{t-1}^{up} \ldots)$
  (past rainfall) (autocorrelation) (routing)

- Questions:
  - how to find the appropriate lags? (lags embody the physical properties of the catchment)
  - how to build non-linear regression function F ?

Steps in modelling process: details

- State the problem (why do the modelling?)
- Evaluate data availability, data requirements
- Specify the modelling methods and choose the tools
- *Build* (identify) the model:
  - Choose variables that reflect the physical processes
  - Collect, analyse and prepare the data
  - Build the model
  - Choose objective function for model performance evaluation
  - Calibrate (identify, estimate) the model parameters:
    - if possible, maximize model performance by comparing the model output to past measured data and adjusting parameters
- *Evaluate* the model:
  - Evaluate the model uncertainty, sensitivity
  - Test (validate) the model using the "unseen" measured data
- *Apply* the model (and possibly assimilate real-time data)
- Evaluate results, refine the model

Suppliers of methods for data-driven modelling

- Statistics
- Machine learning
- Soft computing (fuzzy systems)
- Computational intelligence
- Artificial neural networks
- Data mining
- Non-linear dynamics (chaos theory)

Suppliers of methods for data-driven modelling (1) Machine learning (ML)

- ML = constructing computer programs that automatically improve with experience
- Most general paradigm for DDM
- ML draws on results from:
  - statistics
  - artificial intellingence
  - philosophy, psychology, cognitive science, biology
  - information theory, computational complexity
  - control theory
- For a long time concentrated on categorical (non-continuous) variables

Suppliers of methods for data-driven modelling (2) Soft computing

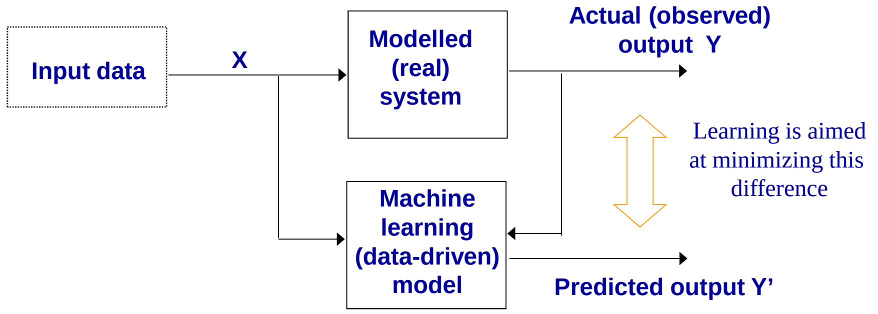
- Soft computing - tolerant for imprecision and uncertainty of data (Zadeh, 1991). Currently includes almost everything:
  - fuzzy logic
  - neural networks
  - evolutionary computing
  - probabilistic computing (incl. belief networks)
  - chaotic systems
  - parts of machine learning theory

Suppliers of methods for data-driven modelling (3) Data mining

- Data mining (preparation, reduction, finding new knowledge):
  - automatic classification
  - identification of trends (eg. statistical methods like ARIMA)
  - data normalization, smoothing, data restoration
  - association rules and decision trees
    - IF (WL>1.2 @3 h ago, Rainfall>50 @1 h ago) THEN (WL>1.5 @now)
  - neural networks
  - fuzzy systems
- Other methods oriented towards optimization:
  - automatic calibration (with a lot of data involved, makes a physically-driven model partly data-driven)

Machine learning: Learning from data

Data-driven modelling: (machine) learning



- DDM tries to "learn" the *target function* $Y=f(X)$ describing how the real system behaves
- Learning = process of minimizing the difference between observed data and model output. X and Y may be non-numeric
- After learning, being fed with the new inputs, DDM can generate output close to what the real system would generate

Training (calibration), cross-validation, testing

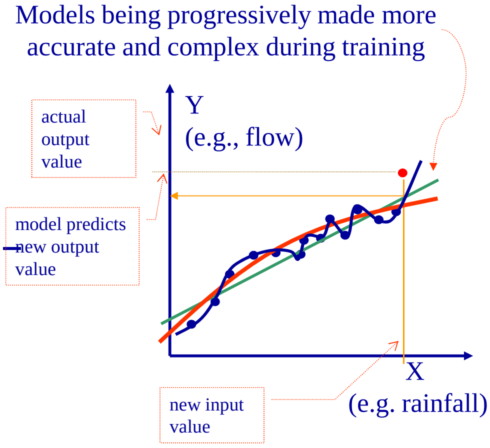
- Ideally, the observed data has to be split in three data sets:
  - training (calibration)
  - cross-validation (*imitates test set and model operation, used to test model performance during calibration process*)

    → used to build the model
  - testing (*imitates model operation, used for final model test, should not be seen by developer*)

    → used to test the model **after it is built**
- One should distinguish
  - minimizing the error during the model calibration, cross-validation
  - minimizing the error during model operation (or on the *unseen* test set)
- Ideally, we should aim at minimizing the cross-validation error – since this will give hopes that error on test set will be also small.
- In practice, process of training uses training set, and cross-validation set is used to check periodically the model error and stop training

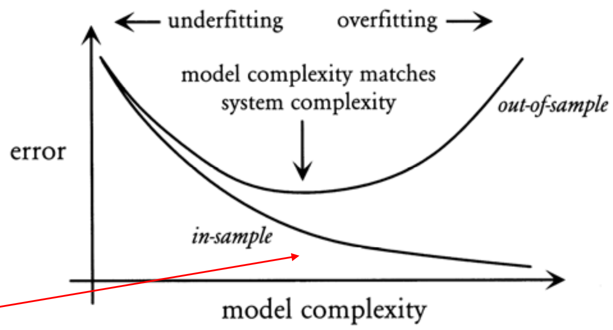
What is a "good" model?

- Consider a model being progressively made more accurate (and complex): **Green → Red → Blue**
- Green (linear) model is simple – but it is not accurate enough
- Blue model is the most accurate but is it "the best"?
- Red model: less accurate than the Blue one, but *captures the trend in data*. It will *generalise* well.
- Question: how to determine during training when to stop improving the model?



Which model is "better": green, red or blue?

Necessity of cross-validation during training



good moment to stop training

in-sample = training

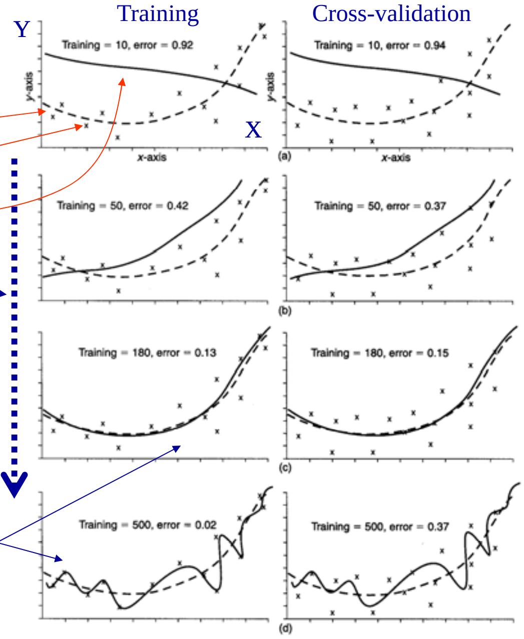
out-of-sample = cross-validation, or verification (testing)

Training, cross-validation, verification (2)

Relationship Y=F(X) to be modelled (unknown)

x Data $\{x_i, y_i\}$collected about this relationship

Data-driven model



- Training: iterative refinement of the model – with each iteration model parameters are changed to reduce error on training set
- → error on training set gets lower and lower
- → error on test set gets lower, then starts to increase
- This may lead to overfitting and high error on cross-validation set
- The moment to stop training

Data-Driven Modelling: care is needed

- Difficulties with extrapolation (working outside the variables' range)
  - A solution: exhaustive data collection, optimal construction of the calibration set
- Care needed if the time series is not stationary
  - A solution: to build several models responsible for different regimes
- Need to ensure that the relevant physical variables are included
  - A solution: use correlation and average mutual information analysis

Data

Data: attributes, inputs, outputs

- set K of examples (or instances)
  represented by the duple $<\mathbf{x}_k, \mathbf{y}_k>$, where $k = 1,\ldots, K$,
  vector $\mathbf{x}_k = \{x_1,\ldots,x_n\}_k$, vector $\mathbf{y}_k = \{y_1,\ldots,y_m\}_k$,
  n = number of inputs, m = number of outputs.
- The process of building a function ('model') $\mathbf{y} = f(\mathbf{x})$ is called *training*.
- Often only one output is considered, so $m = 1$.

Measured data	**Attributes**					**Model output**
	Inputs				*Output*	$y^* = f(\boldsymbol{x})$
Instances	$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	…	$\boldsymbol{x}_n$	$\boldsymbol{y}$	$\boldsymbol{y}^*$
Instance 1	x_{11}	x_{12}		x_{1n}	y_1	$y_1{}^*$
Instance 2	x_{21}	x_{21}		x_{2n}	y_2	$y_2{}^*$
…						…
Instance K	x_{K1}	x_{K2}		x_{Kn}	y_K	$y_K{}^*$

Types of data (roughly)

- Class (category, label)
- Ordinal (order)
- Numeric (real-valued)
- etc. (not considered here)

- (Time) series data: numerical data which values have associated index variable with it.

Four styles of learning

- Classification
  - on the basis of classified examples, a way of classifying unseen examples is to be found
- Association
  - association between features (which combinations of values are most frequent) is to be identified
- Clustering
  - groups of objects (examples) that are "close" are to be identified
- Numeric prediction
  - outcome is not a class, but a numeric (real) value
  - often called *regression*

Instances (examples)

- Instances = examples of input data.
- Instances that can be stored in a simple rectangular table (only these will be mainly considered):
  - individual unrelated customers described by a set of attributes
  - records of rainfall, runoff, water level taken every hour
- Instances that cannot be stored in a table, but require more complex structures:
  - instances of pairs that are *sisters,* taken from a family tree
  - related tables in complex databases describing staff, their ownership, involvement in projects, borrowing of computers, etc.

Data preparation results in:

- training data set - raw data is presented in a form necessary to to train the DDM;
- cross-validation data set - needed to detect overtraining;
- testing, or validation data set - it is needed to validate (test) the model's predictive performance;
- algorithms and software to perform pre-processing (eg., normalization);
- algorithms and software to perform post-processing (eg., denormalization).

Important steps in data preparation

- Replace missing, empty, inaccuarate values
- Handle issue of spatial and temporal resolution
- Linear scaling and normalization
- Non-linear transformations
- Transform the distributions
- Time series:
  - Fourier and wavelet transforms
  - Identification of trend, seasonality, cyles, noise
  - Smoothing data
- Finding relationships between attributes (eg. correlation, average mutual information - AVI)
- Discretizing numeric attributes into {low, medium, high}
- Data reduction (Principal components analysis - PCA)

Replacing missing and empty values

- What to do with the outliers? How to reconstruct missing values?
- *Estimator* is a device (algorithm) used to make a justifiable guess about the value of some particular variable, that is, to produce an estimate
- *Unbiased* estimator is a method of guessing that does not change important characteristics of the data set when the estimates are included with the existing values
- Example: Dataset 1 2 3 x 5
  - Estimators:
    - 2.750, if the mean is to be unbiased;
    - 4.659, if the standard deviation is to be unbiased;
    - 4.000, if the step-wise change in the variable value (trend) is to be unbiased (that is, linear interpolation is used $x_i = (x_{i+1} + x_{i-1}) / 2$)

Issue of spatial and temporal resolution

- Examples:
  - in a harbour sedimentation is measured once in two weeks at one locations and once a month at other two locations, and never at other (maybe important) locatons
  - in a catchment the rainfall data that was manually collected at a three gauging stations for 20 years once a day, and 3 years ago the measurements started also at 4 new automatic stations as well, with the hourly frequency
- Solutions:
  - filling-in missing data
  - introducing an artificial resolution being equal to the maximum for all variables

Linear scaling and normalization

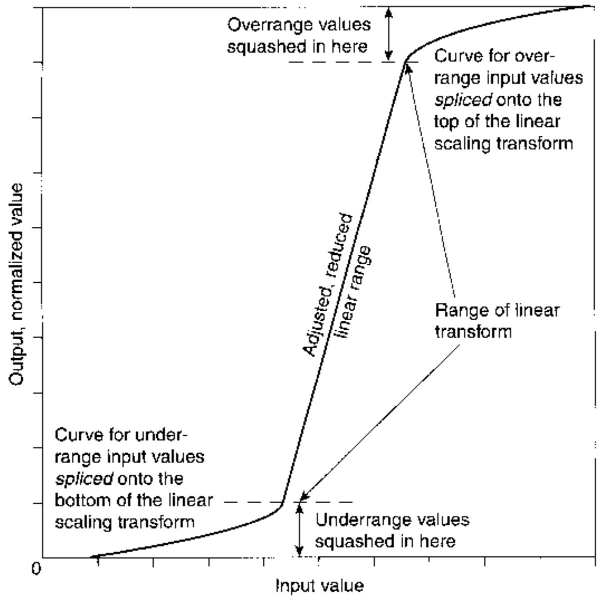
- General form:
- $x'_i = a\, x_i + b$
- to keep data positive:
- $x'_i = x_i + min\ (x_1...x_n) + SmallConst$
- Squashing data into the range [0, 1]

$$x'_i = \frac{x_i - \min(x_1...x_n)}{\max(x_1...x_n) - \min(x_1...x_n)}$$

Non-linear transformations

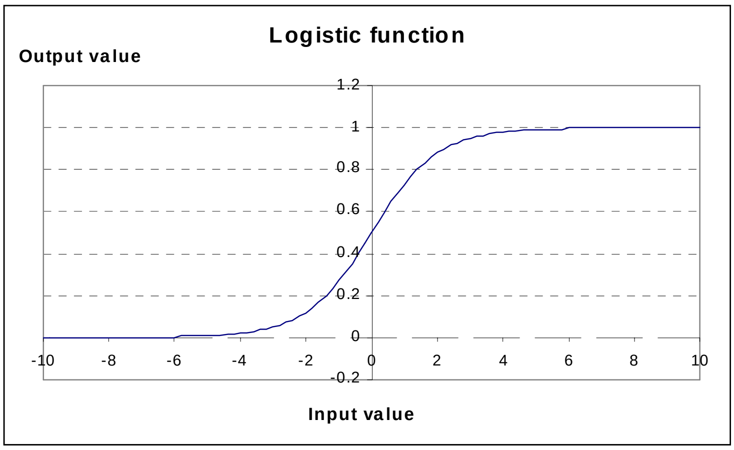
- Logarithmic
- $x'_i = log\,(x_i)$

- Softmax scaling:



Logistic function

$$L(x) = \frac{1}{1 + e^{-\alpha x}}$$

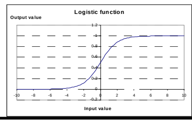


Softmax function

- 1. {x} should be first transformed linearly to vary around the mean m_x :

$$x_i' = \frac{x_i - E(x)}{\lambda\,(\sigma_x / 2\pi)}$$

  - where
  - $E(x)$ is mean value of variable x;
  - σ_x is the standard deviation of variable x;
  - λ is linear response measured in standard deviations – for example ±σ (that is σ on either side of the central point of the distribution) cover 68% of the total range of x, ±2σ cover 95.5%, ±3σ cover 99.7%.
  - π ≈ 3.14
- 2. logistic function applied L(x′)



Transforming the distributions: Box-Cox transform

- The first step uses the power transform to adjust the changing variance:

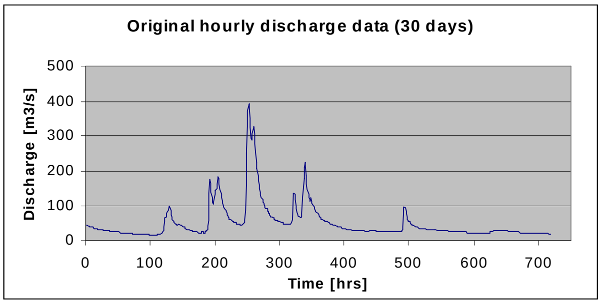
$$x_i' = \frac{x_i^{\lambda} - 1}{\lambda}$$

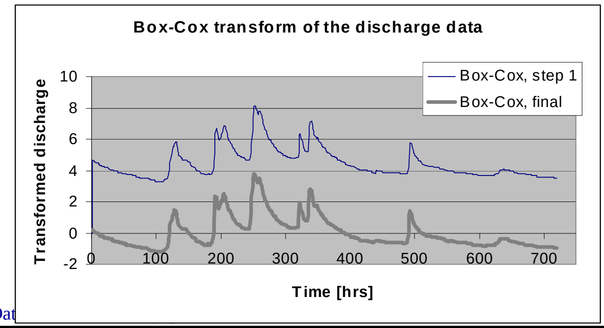
- where
  - x_i is the original value,
  - x'_i is the transformed value,
  - λ is a user-selected value.
- The second step balances the distribution by subtracting the mean and dividing the result by the standard deviation:

$$x_i'' = \frac{x_i' - E(x')}{\sigma_{x'}}$$

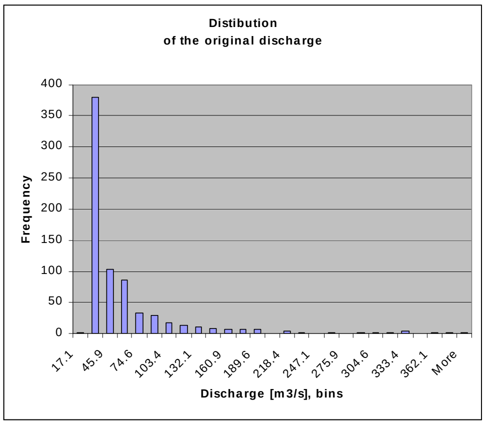
  - where
    - x'_i is the value after the first transform,
    - x''_i is (final) standardized value,
    - $E(x')$ is mean value of variable x'
    - $\sigma_{x'}$ is standard deviation of variable x'.

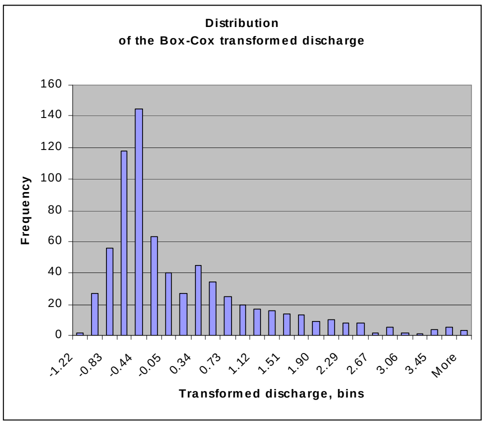
Box-Cox transform of the rainfall data



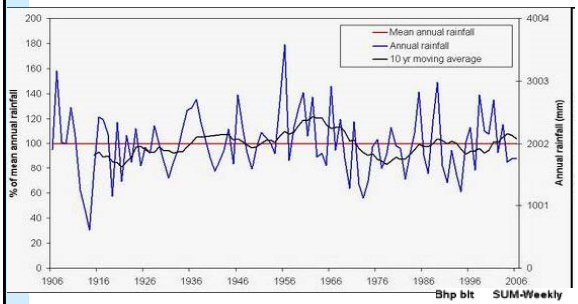


Box-Cox transform: original and resulting histograms

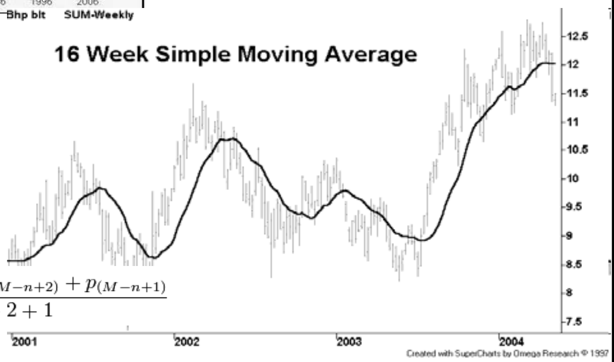




Smooth data?



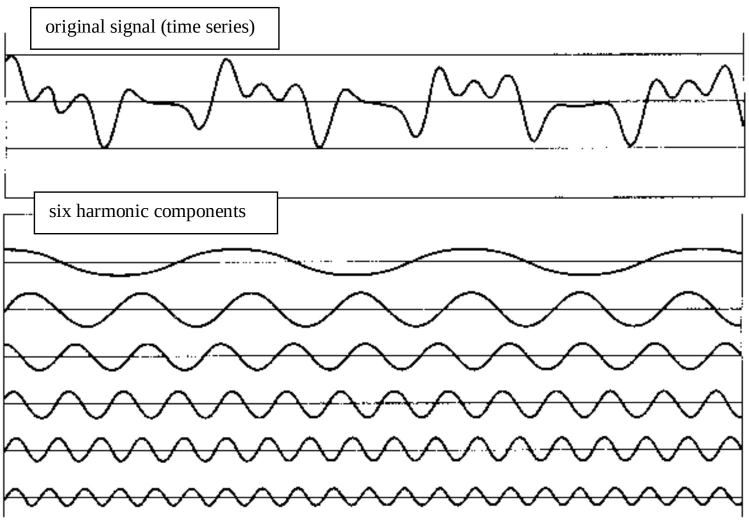
- Simple and Weighted Moving Averages
- Savitzky–Golay filter: builds local polynomial regression (of degree k) on a series of values
- other filters (Gaussian, Fourier, etc.)



$$SMA = \frac{p_M + p_{M-1} + \cdots + p_{M-9}}{10}$$

$$\text{WMA}_M = \frac{np_M + (n-1)p_{M-1} + \cdots + 2p_{(M-n+2)} + p_{(M-n+1)}}{n + (n-1) + \cdots + 2 + 1}$$

Fourier transform can be used to smooth data (extract only low-frequency harmonics)



Finding relationships between i/o variables

- **Correlation coefficient R**

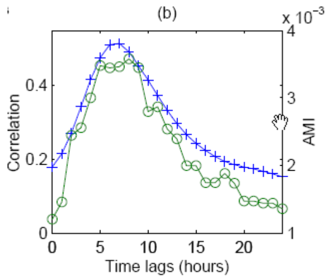
$$R = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

- **Average mutual information (AMI)**. It represents the measure of information that can be learned from one set of data having knowledge of another set of data.

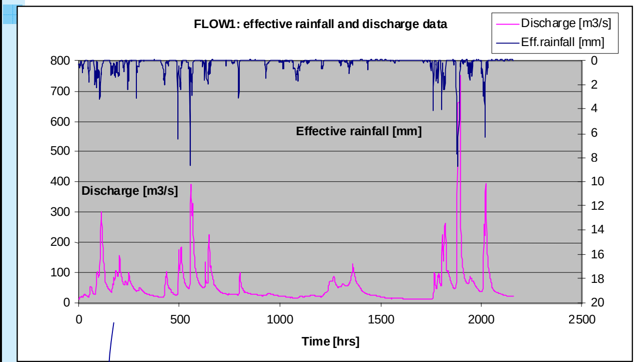
$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log_2 \left[\frac{P(x,y)}{P(x)P(y)} \right]$$

- where $P(x,y)$ is the joint probability for realisation x of X and y of Y; and $P(x)$ and $P(y)$ are the individual probabilities of these realisations
- If X is completely independent of Y then AMI $I(X;Y)$ is zero.

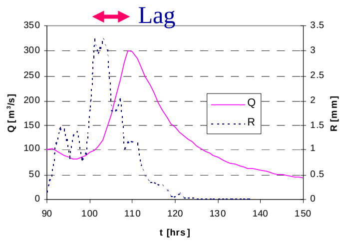
AMI can be used to identify the optimal time lag for a data-driven rainfall-runoff model



Us of AMI: relatedness between flow and past rainfall

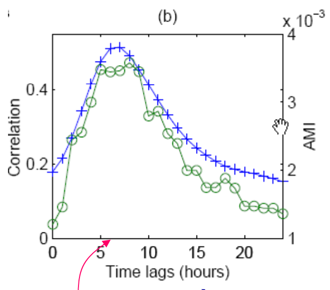


Zoom in:



- Consider future discharge Q_{t+1} and past rainfalls R_{t-L}. What is lag L such that the relatedness is strongest ?
- AMI can be used to identify the optimal time lag to ensure

AMI between Q_{t+1} and past lagged rainfalls R_{t-L}



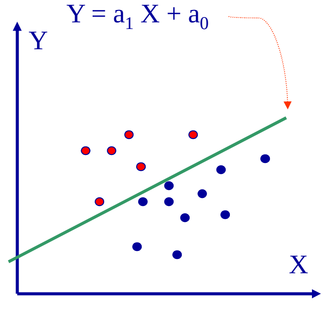
Max AMI → optimal lag

Introducing classification:

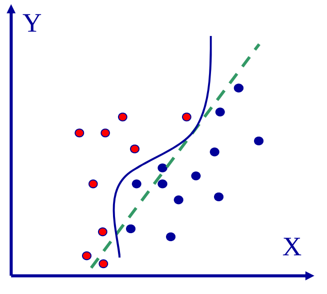
main ideas

Discriminating surfaces: a traditional method of classification from statistics

- surface (line, hyperplane) separates examples of different classes

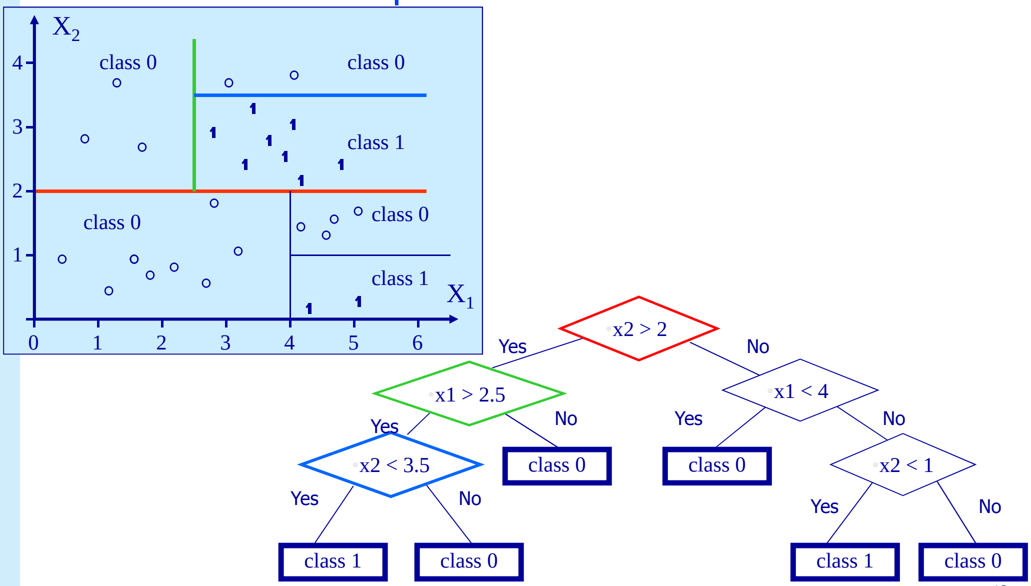


linearly separable examples



more difficult example: linear function will misclassify several examples,
so a non-linear function needed
(or transformation of space)

Decision tree: example with 2 numeric input variables, 2 output classes: 0 and 1



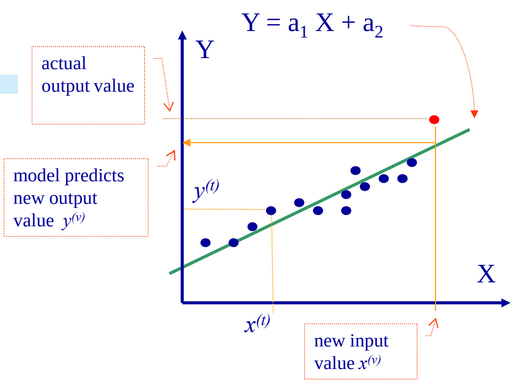
Numeric prediction (regression):

linear models
and their combinations in tree-like structures
(M5 model trees)

Models for numeric prediction

- Target function is real-valued
- There are many methods:
  - Linear and non-linear regression
  - ARMA (auto-regressive moving average) and ARIMA models
  - Artificial Neural Networks (ANN)
- ⇒ We will consider now:
  - Linear regression
  - Regression trees
  - Model trees

Linear regression



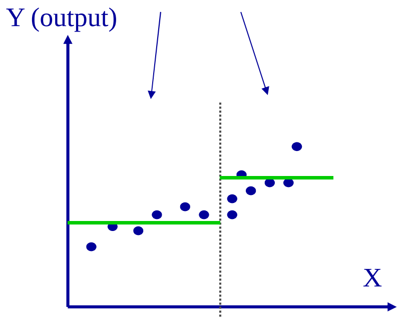
- Given measured (training) data: T vectors $\{x^{(t)},y^{(t)}\}$, $t = 1,\ldots T$.
- Unknown a_1 and a_2 are found by solving an optimization problem

$$E = \sum_{t=1}^{T}\left(y^{(t)} - (a_0 + a_1 x^{(t)})\right)^2 \rightarrow \min$$

- Then for the new V vectors $\{x^{(v)}\}$, $v = 1,\ldots V$ this equation can approximately reproduce the corresponding functions values $\{y^{(v)}\}$, $v = 1,\ldots V$

Numeric prediction by averaging in subsets (regression trees in 1D)

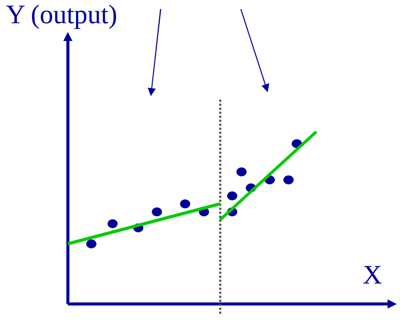
input X_1 is split into intervals;
averaging is performed in each interval



- input space can be split according to *standard deviation* in subsets

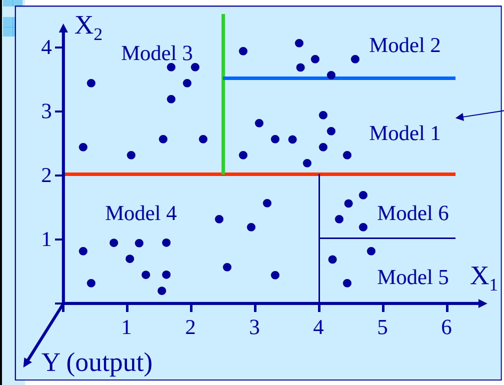
Numeric prediction by piece-wise linear models (model trees in 1D)

input X_1 is split into intervals;
separate linear models can be built
for each of the intervals



- question is: how to split the input space in an optimal way?

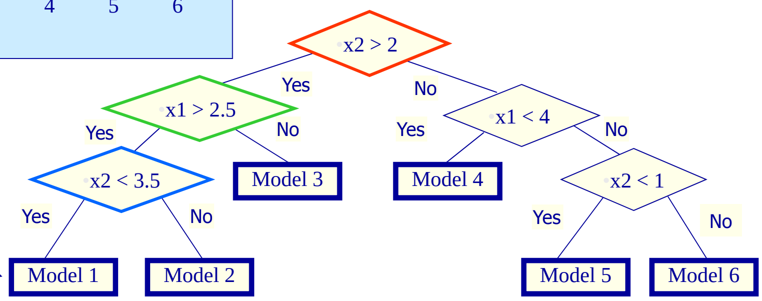
Regression and M5 model trees: building them in 2D



input space $X_1 \times X_2$ is split into regions; separate regression models can be built for each of the regions

- Tree structure where
  - nodes are splitting conditions
  - leaves are:
    - constants (→ regression tree)
    - linear regression models (→ M5 model tree)

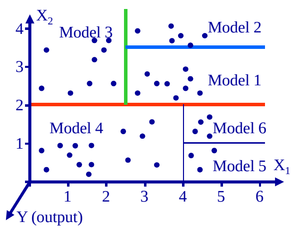
Example of Model 1:
- in a regression tree: $Y = 10.5$
- in M5 model tree: $Y = 2.1 * X_1 + 0.3 * X_2$



How to select an attribute for split in regression trees and M5 model trees

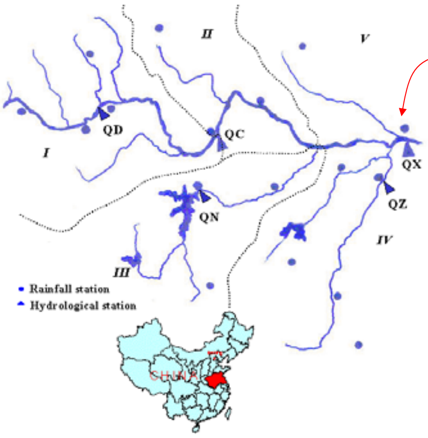
- regression trees: same as in decision trees (information gain)
- main idea:
  - choose the attribute that splits the portion T of the training data that reaches a particular node into subsets T_1, T_2,...
  - use the *standard deviation sd (T)* of the output values in T as a measure of error at that node (in decision trees - entropy *E(T)* was used)
  - split should result in subsets T_i with low standard deviation $sd(T_i)$
  - so model trees splitting criterion is SDR (standard deviation reduction) that has to be maximized:

$$SDR = sd(T) - \sum_i \frac{|T_i|}{|T|} \times sd(T_i)$$



Regression and model trees in numerical prediction: some applications

M5 tree in Rainfall-runoff modelling (Huai river, China)



Data (1976-1996)
Daily discharges (*QX, QC*)
Daily rainfall at 17 stations
Daily evaporation for 14 years (1976-1989) at 3 stations
Training data: 1976-89
Cross-valid. & testing: 1990-96

Model structure:
$Q_{t+T} = F(R_t\ R_{t-1} \ldots R_{t-L} \ldots Q_t\ Q_{t-1}\ Q_{t-A} \ldots Q_t^{up}\ Q_{t-1}^{up} \ldots)$

Variables considered
Output: predicted discharge QX_{t+1}
Inputs (with different time lags):
- daily areal rainfall (*Pa*)
- moving average of daily areal rainfall (*PaMov*)
- discharges (*QX*) and upstream (*QC*)

Smoothed variables have higher correlation coeff. with the output, e.g. 2-day-moving average of rainfall ($PaMov2_t$)

- Final model for the flood season:
- Output:
  - discharge the next day QX_{t+1}
- Inputs:
  - Pa_t Pa_{t-1} $PaMov2_t$ $PaMov2_{t-1}$ QC_t QC_{t-1} QX_t
- Techniques used: M5 model trees, ANN

Resulting M5 model tree with 7 models (Huai river)

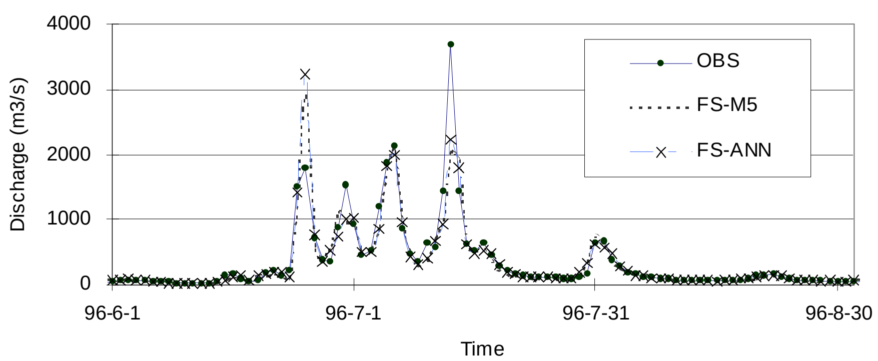
```
QXt <= 154 :
|   PaMov2t <= 4.5 : LM1 (1499/4.86%)
|   PaMov2t >  4.5 :
|   |   PaMov2t <= 18.5 : LM2 (315/15.9%)
|   |   PaMov2t >  18.5 : LM3 (91/86.9%)
QXt >  154 :
|   PaMov2t-1 <= 13.5 :
|   |   PaMov2t <= 4.5 : LM4 (377/15.9%)
|   |   PaMov2t >  4.5 : LM5 (109/89.7%)
|   PaMov2t-1 >  13.5 :
|   |   PaMov2t <= 26.5 : LM6 (135/73.1%)
|   |   PaMov2t >  26.5 : LM7 (49/270%)
```

Models at the leaves:

```
LM1:  QXt+1 = 2.28 + 0.714PaMov2t-1 - 0.21PaMov2t + 1.02Pat-1 + 0.193Pat
              - 0.0085QCt-1 + 0.336QCt + 0.771QXt
LM2:  QXt+1 = -24.4 - 0.0481PaMov2t-1 - 4.96PaMov2t + 3.91Pat-1 + 4.51Pat
              - 0.363QCt-1 + 0.712QCt + 1.05QXt
LM3:  QXt+1 = -183 + 10.3PaMov2t-1 + 8.37PaMov2t - 5.32Pat-1 + 1.49Pat
              - 0.0193QCt-1 + 0.106QCt + 2.16QXt
LM4:  QXt+1 = 47.3 + 1.06PaMov2t-1 - 2.05PaMov2t + 1.91Pat-1 + 4.01Pat
              - 0.3QCt-1 + 1.11QCt + 0.383QXt
LM5:  QXt+1 = -151 - 0.277PaMov2t-1 - 37.8PaMov2t + 31.1Pat-1 + 30.3Pat
              - 0.672QCt-1 + 0.746QCt + 0.842QXt
LM6:  QXt+1 = 138 - 5.95PaMov2t-1 - 39.5PaMov2t + 29.6Pat-1 + 35.4Pat
              - 0.303QCt-1 + 0.836QCt + 0.461QXt
LM7:  QXt+1 = -131 - 27.2PaMov2t-1 + 51.9PaMov2t + 0.125Pat-1 - 5.29Pat
              - 0.0941QCt-1 + 0.557QCt + 0.754QXt
```


Performance of M5 and ANN models (Huai river)

D.P. Solomatine and Y. Xue. M5 model trees compared to neural networks: application to flood forecasting in the upper reach of the Huai River in China. *ASCE Journal of Hydrologic Engineering*, 9(6), 2004, 491-501.



M5 and ANN, flood season data (testing, fragment)

M5 model trees and ANNs in rainfall-runoff modelling: predicting flow three hours ahead (Sieve catchment)

The model: $Q_{t+3} = f\ (RE_t, RE_{t-1}, RE_{t-2}, RE_{t-3}, Q_t, Q_{t-1})$

- Inputs:
- RE_t, RE_{t-1}, RE_{t-2}, RE_{t-3}, Q_t, Q_{t-1} (rainfall for 3 past hours, runoff for 2)
- ANN verification

RMSE=11.353

NRMSE=0.234

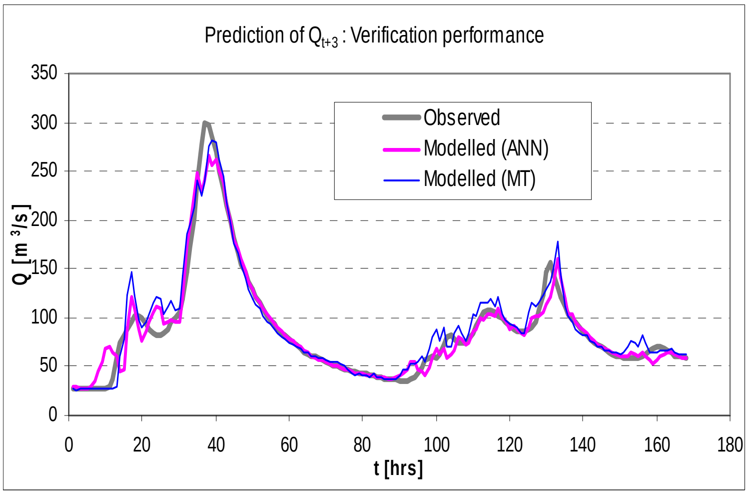
COE=0.9452

- MT verification

RMSE=12.548

NRMSE=0.258

COE=0.9331



Numerical prediction by M5 model trees: conclusions

- Transparency of trees: model trees is easy to understand (even by the managers)
- M5 model tree is a mixture of local accurate models
- Pruning (reducing size) allows:
  - to prevent overfitting
  - to generate a family of models of various accuracy and complexity