Industrial Evolutionary Computing

Arthur Kordon, Guido Smits, and Mark Kotanchek

The Dow Chemical Company

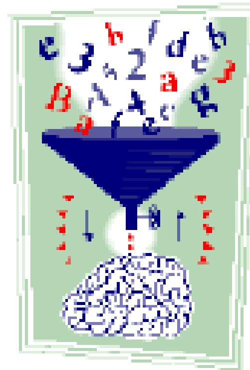
GECCO 2004

Overview

In theory, there is no difference between theory and practice. In practice, there is.
- Jan L.A. van de Snepscheut

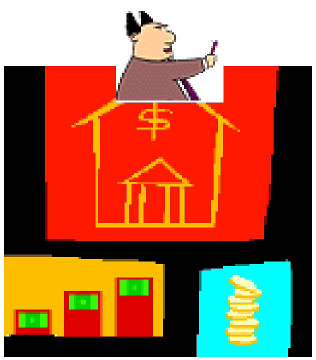
- Evolutionary Computing and the business model
- Implementation guidelines
- Integrate & Conquer
- Key application areas
- Open issues

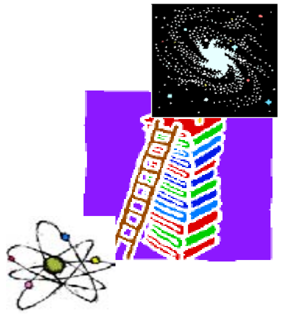
Academic vs. industrial data analysis



Transfer data into knowledge

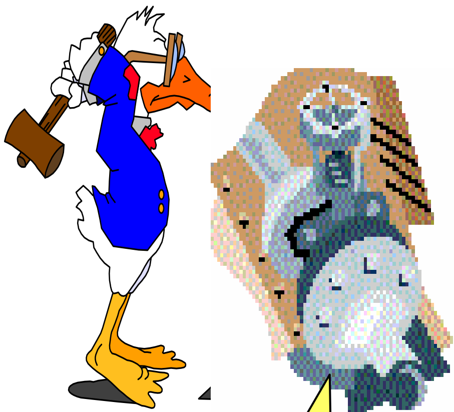
Transfer data into value





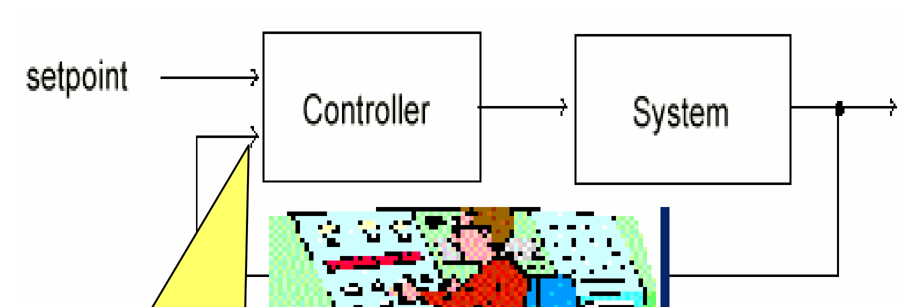
Special Features of Industrial Data Analysis 1

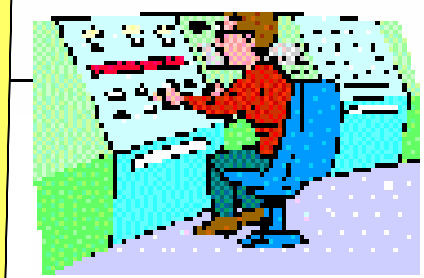
Operators intervention



Operators manually modify the process

Curse of closed loops



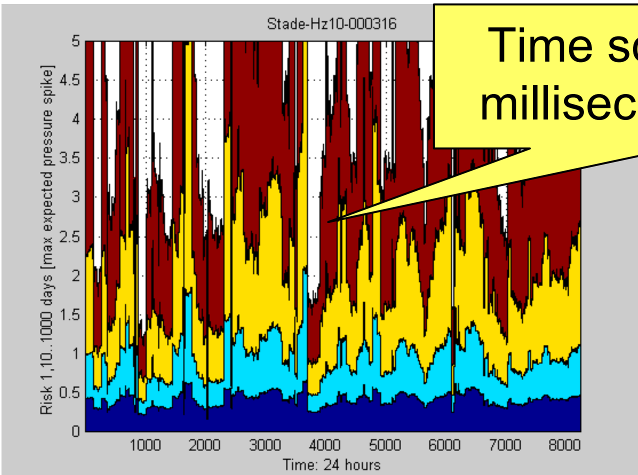


The majority of process variables are in closed loops and depend on controller adjustments

It's the context, stupid!

Special Features of Industrial Data Analysis 2

Multiple time scales



Time scales vary from milliseconds to months

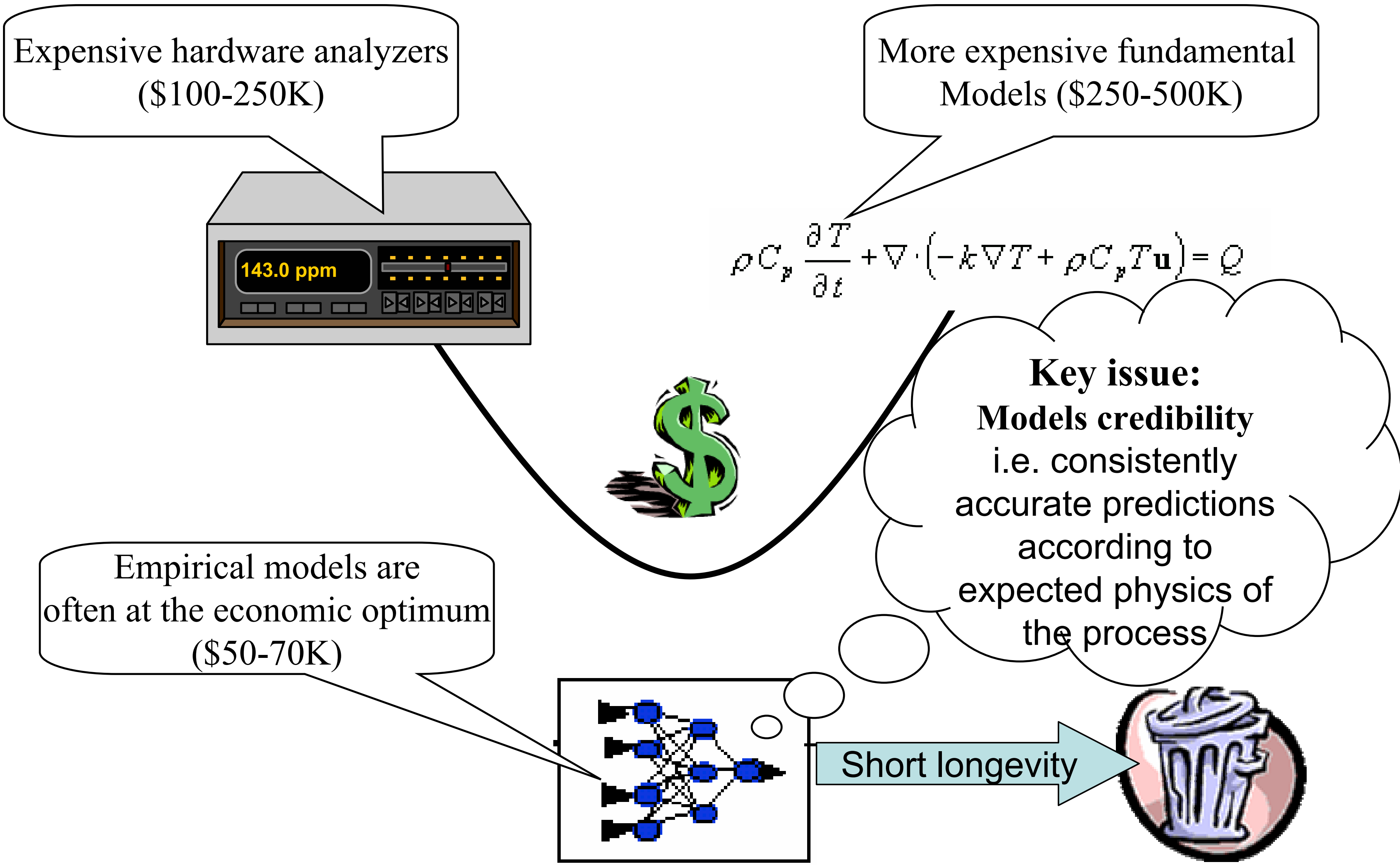
Real-time pressure

Most of models operate in real time

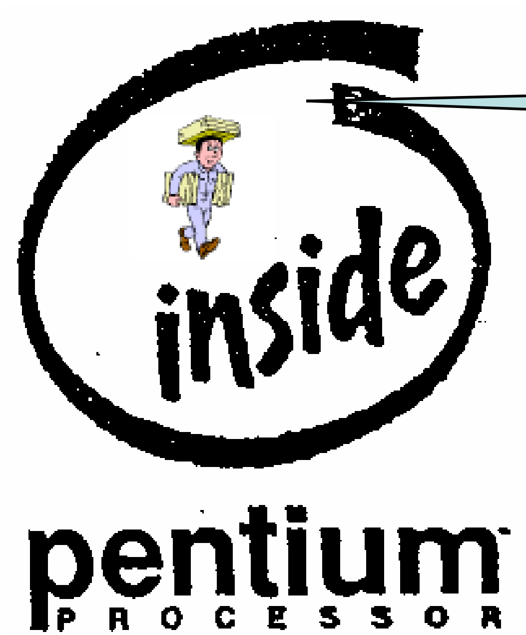
Data analysis must be explained clearly

Process engineers prefer to have a generic understanding of data analysis approaches they use

Economic advantage of data-driven models

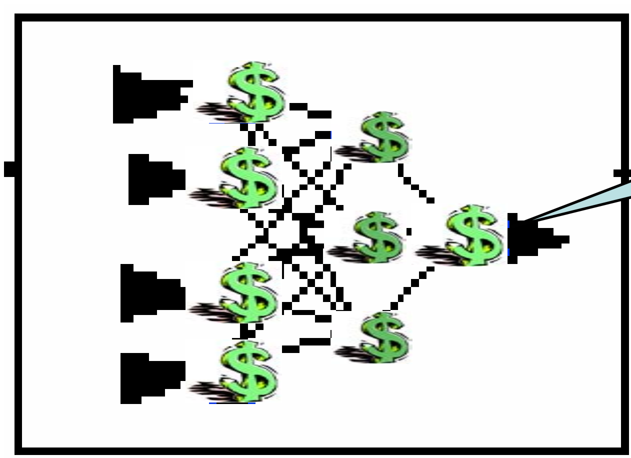


Intelligent Systems in Industrial Data Analysis: Lessons From the Past



The Expert Systems campaign (late 80s)
"We'll put engineers in the box"

- **static rule-based models not linked to numerical world**
- **the politics of knowledge acquisition**
- **the efforts of knowledge acquisition**



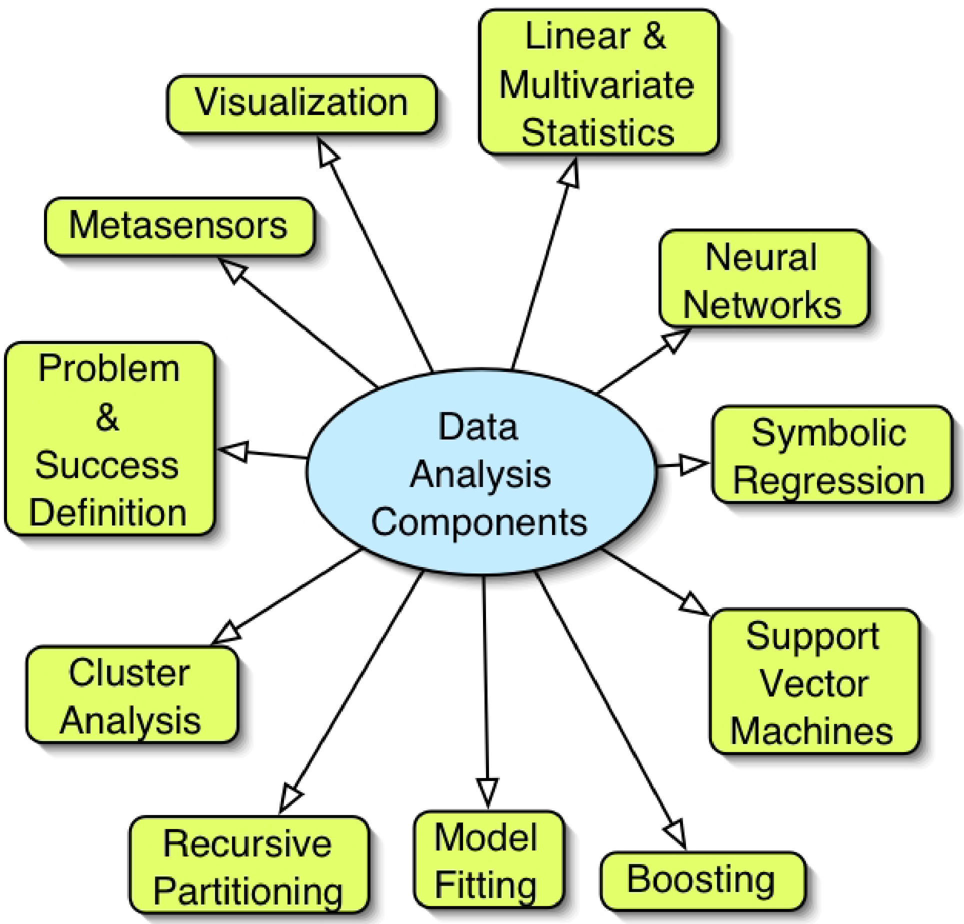
The Neural Networks campaign (early 90s)
"We'll turn data into gold"

- **black-box models with inefficient structure**
- **fragile models and model validation**
- **maintenance nightmare**

Application Issues in the Chemical Industry

- High dimensionality of the data
- Highly correlated data with time delays
- Outlier detection
- Multiple optima
- Intensive number crunching needed
- Too much or too little data
  - Often sparse, or "statistically insignificant" instances, but at the same time, physically meaningful or commercially viable
  - Often lots of redundant data

Industrial data analysis components



The role of evolutionary computing (symbolic regression) is to …

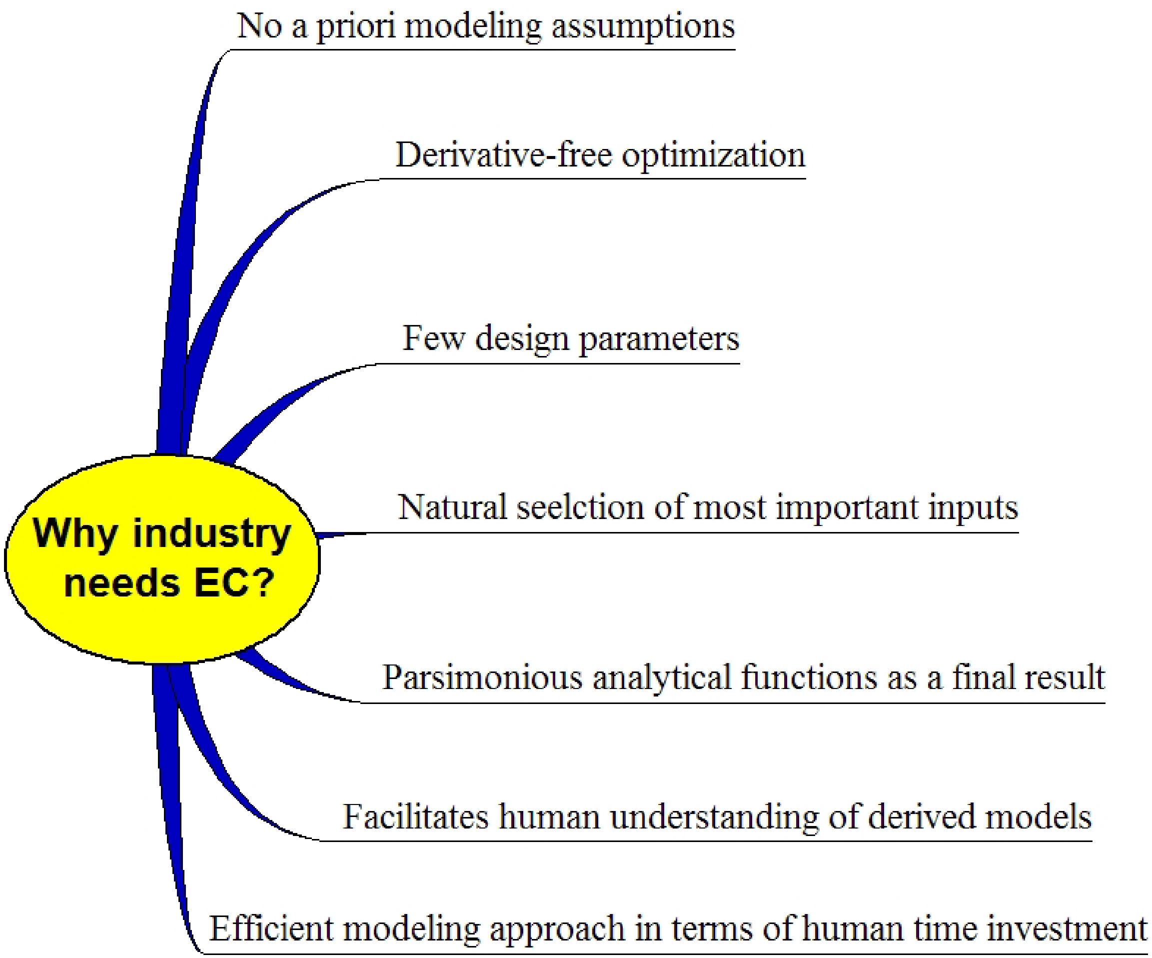
- Facilitate physical/mechanism insight and **understanding**
- **Summarize** data behavior
- Identify data **transforms** and metasensors
- Perform **variable selection**
- Enable response surface **exploration and optimization**
- **Visualize** behavior in the form of a symbolic expression

The overall goal is to achieve speed, accuracy & efficiency. Symbolic regression is part of an integrated methodology.

Selected Evolutionary Computing Approaches (used in industrial applications at Dow Chemical)

- **Applied EC approaches**
  - Genetic Programming for Symbolic Regression
  - Particle Swarm Optimization
  - Genetic Algorithms
- **Auxiliary Technologies**
  - Neural Networks
  - Support Vector Machines (for regression)
  - Context + Experts + Statistics + Physics

Why industry needs Evolutionary Computing?



Technical issues with Evolutionary Computing

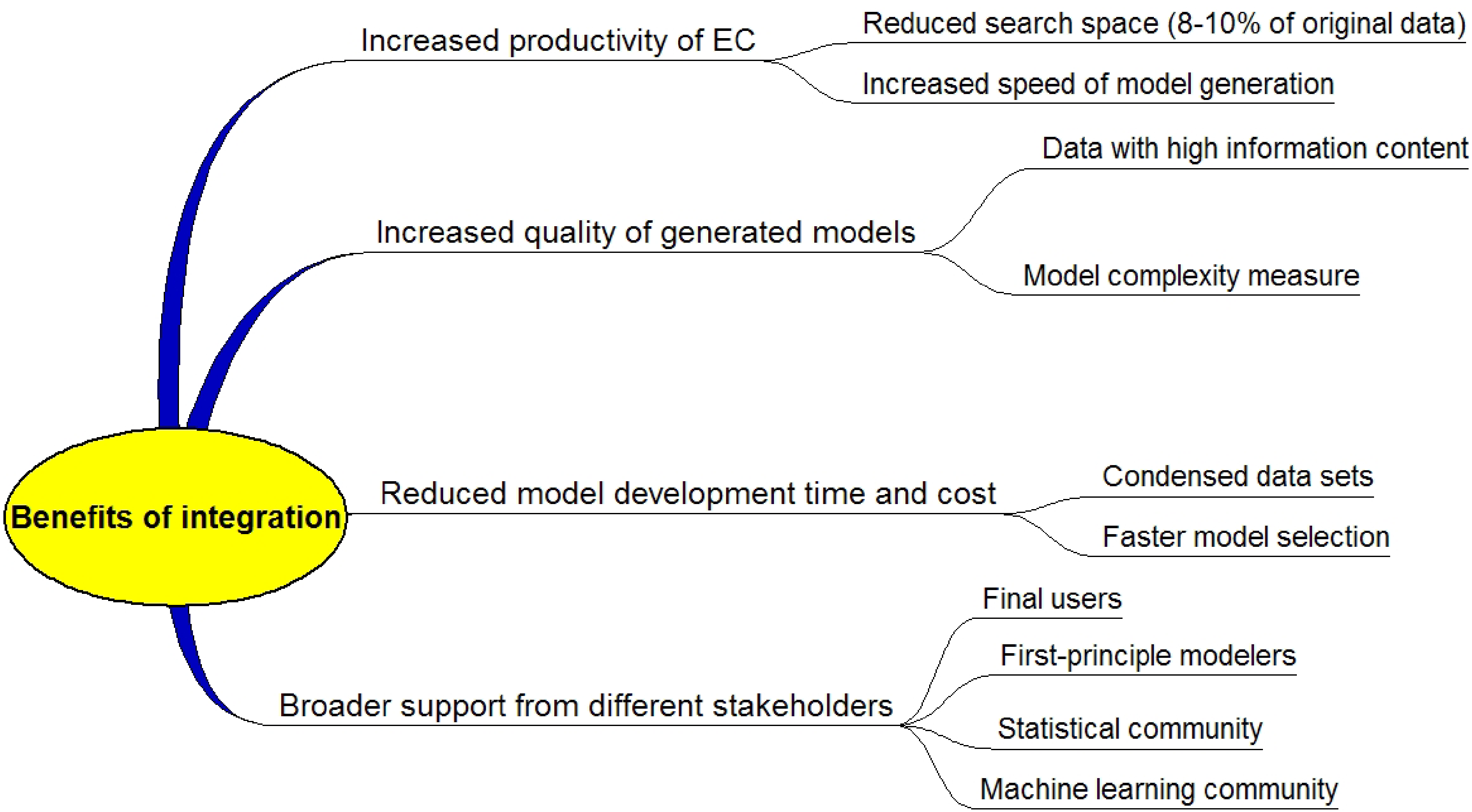
Typical size for
undesigned data is
~30 variables x ~1000 data points

- **Issues with EC**
  - **Slow model generation on large data sets**
  - **Non-trivial model selection**
    - Accuracy of prediction
    - Model complexity
    - Model interpretability
    - Model diversity
  - **Sensitive to data quality**
  - **User-friendly software not available**
  - **Relatively unknown approach to engineering community**
  - **Theoretical development still in progress**
  - **Applications in infancy**

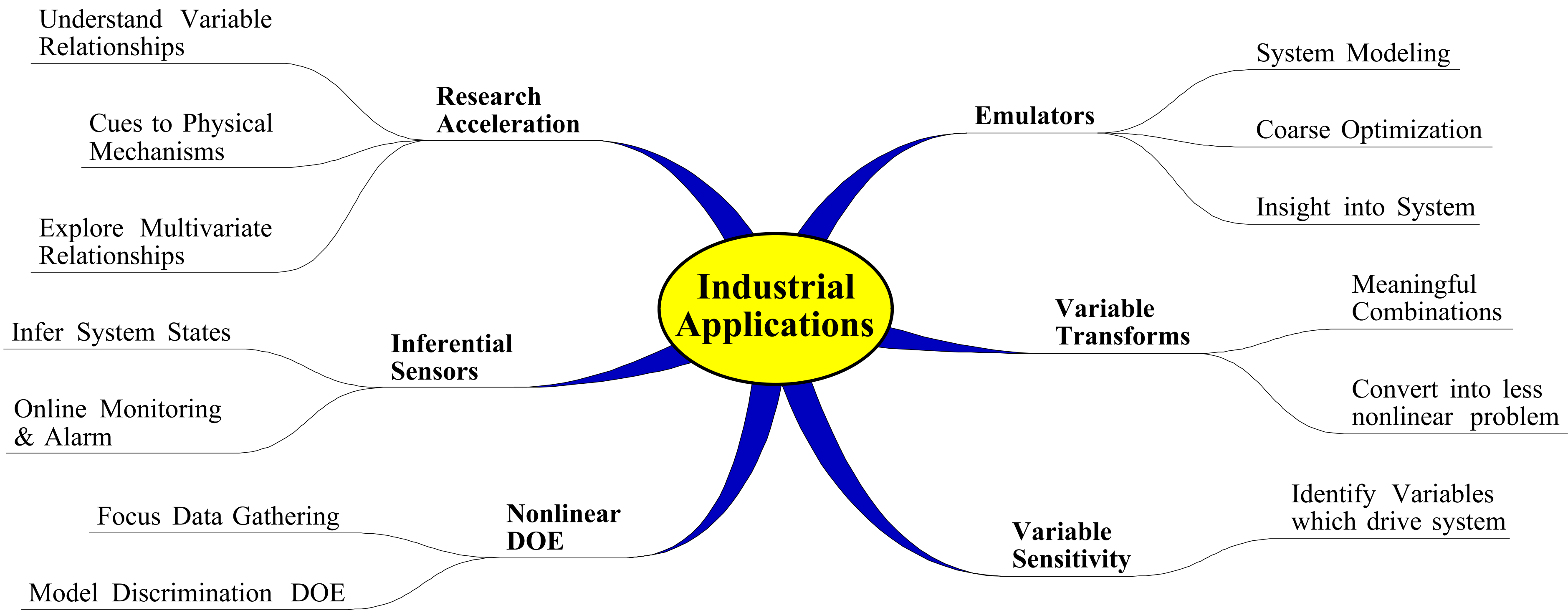
Economic benefits from Evolutionary Computing

- Resolve complex optimization problems (PSO/GA)
- Physical Interpretation & Insight (Symbolic Regression)
  - Suggestions for profitable directions for research/sensors/etc.
  - Accelerate research & development
  - Higher credibility in comparison to black-boxes
- Reduce model development cost
  - Significantly reduced development time relative to alternatives
- Reduce model exploitation cost
  - Minimal model implementation cost (no need for specialized software)
  - Reduced maintenance cost (less frequent re-training)
- Reduce cost of industrial experiments
  - Minimizes the number of additional experiments

Benefits of integrating Evolutionary Computing with other approaches



Application areas with impact

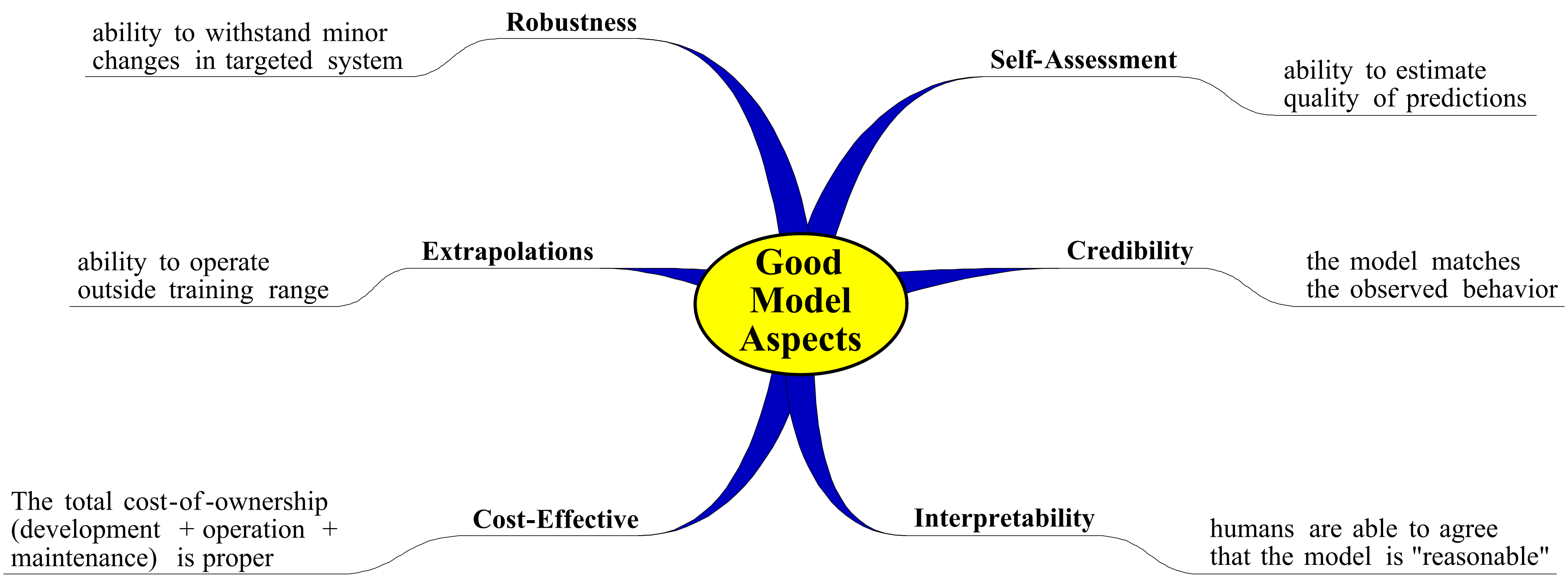


Implementation guidelines

- Requirements for successful empirical modeling
- Key issues to be overcome
- Implementation strategy
- Implementation tools

Requirements for successful data-driven modeling

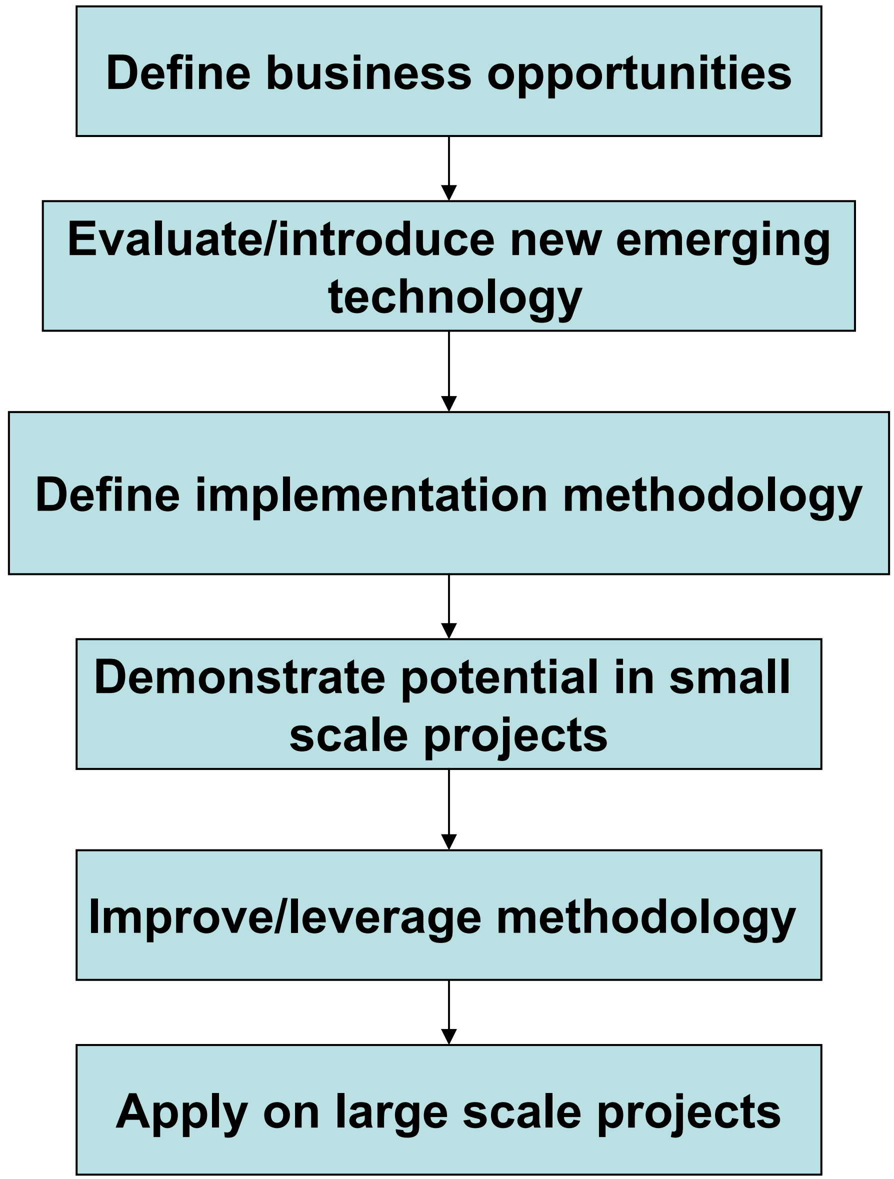
Objective function:
Minimizing modeling cost and maximizing data analysis efficiency under broad range of operating conditions



Key issues to overcome

- **Key implementation issues**
  - Data pre-processing and condensation
  - Model selection
  - User-friendly implementation tools
  - Marketing of EC to different modeling communities
  - Resistance to implement empirical models (inherited from black-box models)
  - Seamless integration into existing maintenance and support infrastructure
  - Critical mass of model developers familiar with EC

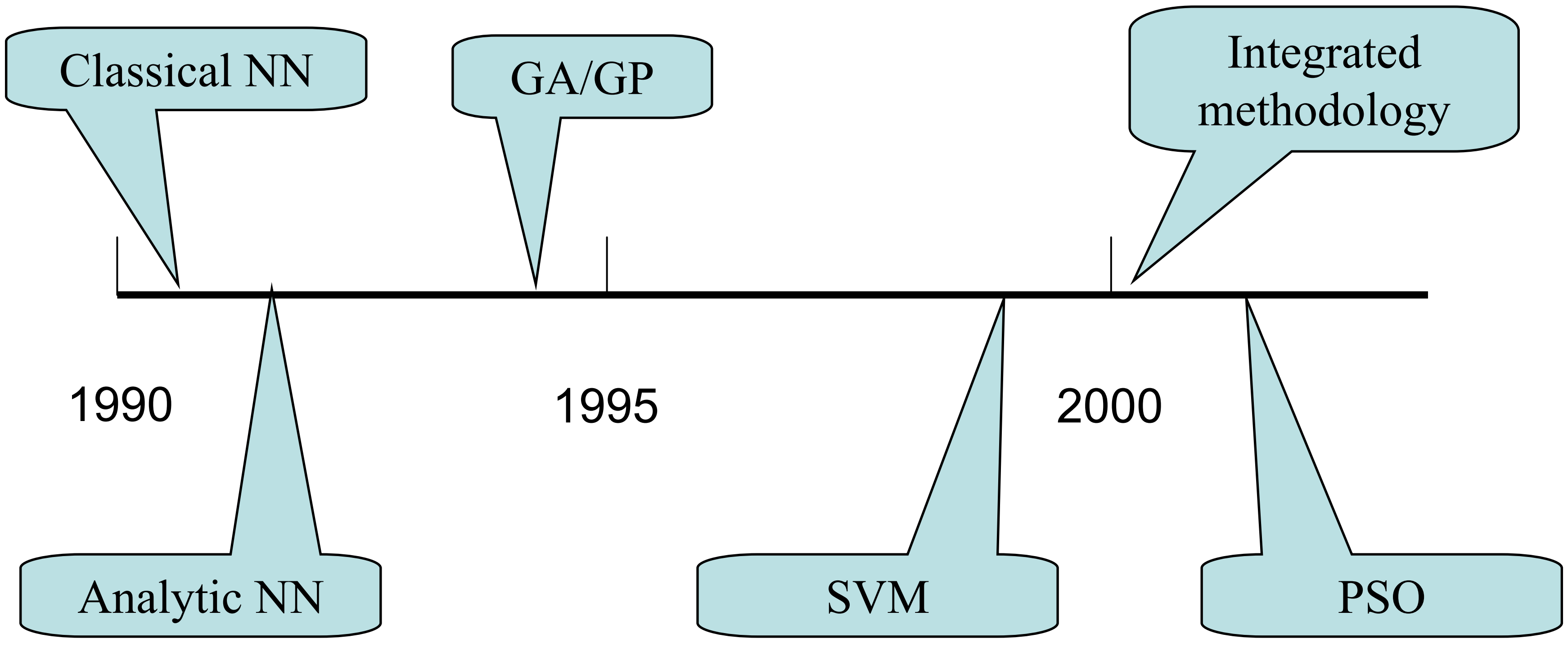
Implementation strategy



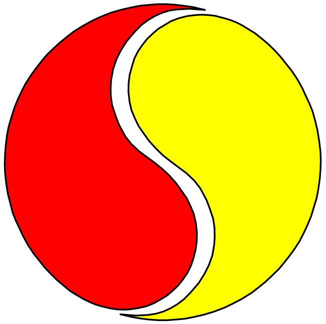
Implementation tools

- **MATLAB (Dow developed)**
  - **GA**
  - **GP**
  - **PSO (single objective and multi-objective)**
  - **Analytic neural networks**
  - **Support vector machines**
- **Mathematica (Dow developed)**
  - **Symbolic regression package**
  - **AutoAnalysisTools**
  - **Analytic neural networks**
  - **PSO**
  - **Group Methods of Data Handling (GMDH)**
- **Tools for model distribution**
  - **Delphi**
  - **Web Mathematica**
  - **Excel**
  - **Process control systems**

Exploitation/Implementation Sequence of Computational Intelligence Approaches in Dow Chemical

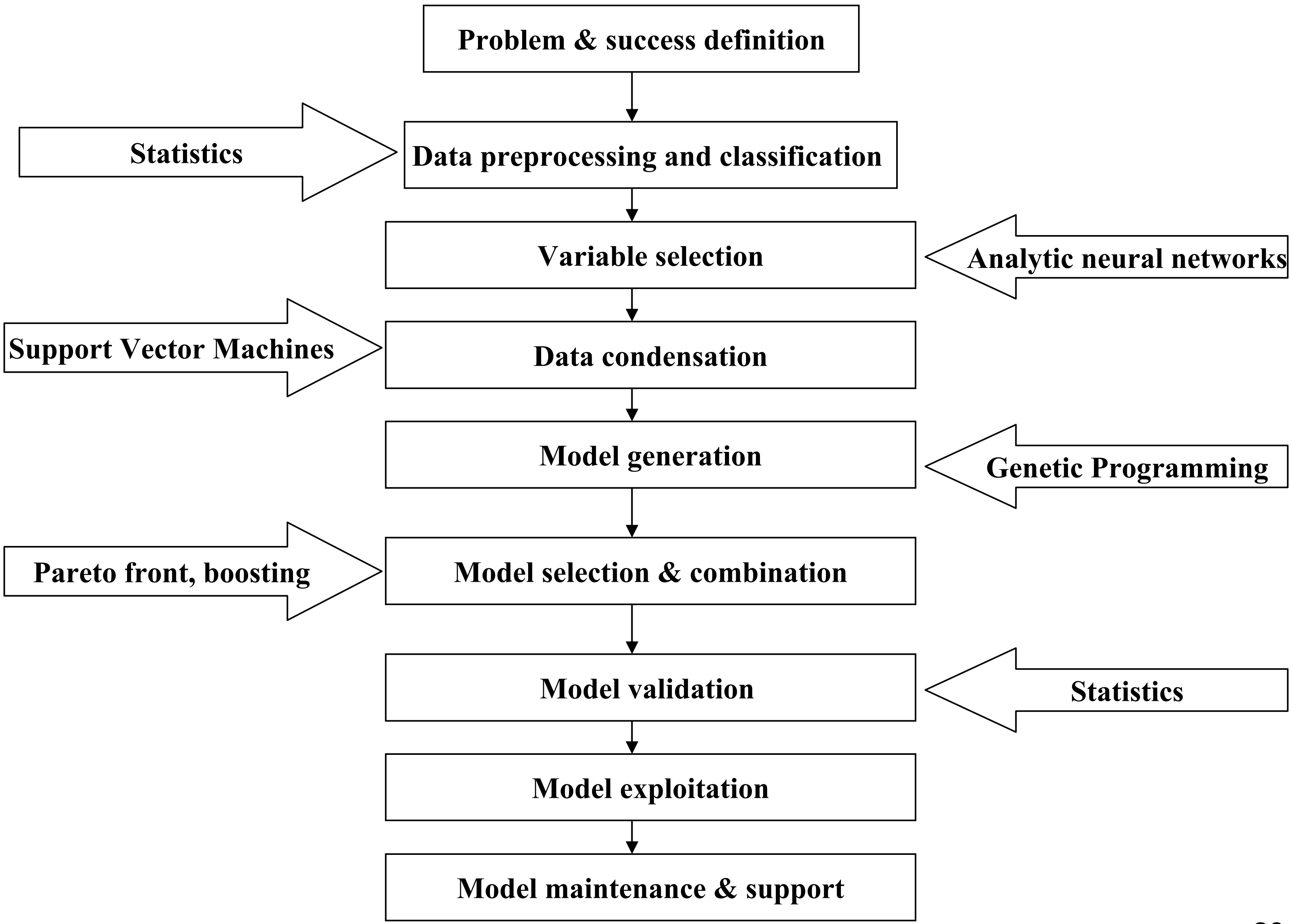


Integrate & Conquer

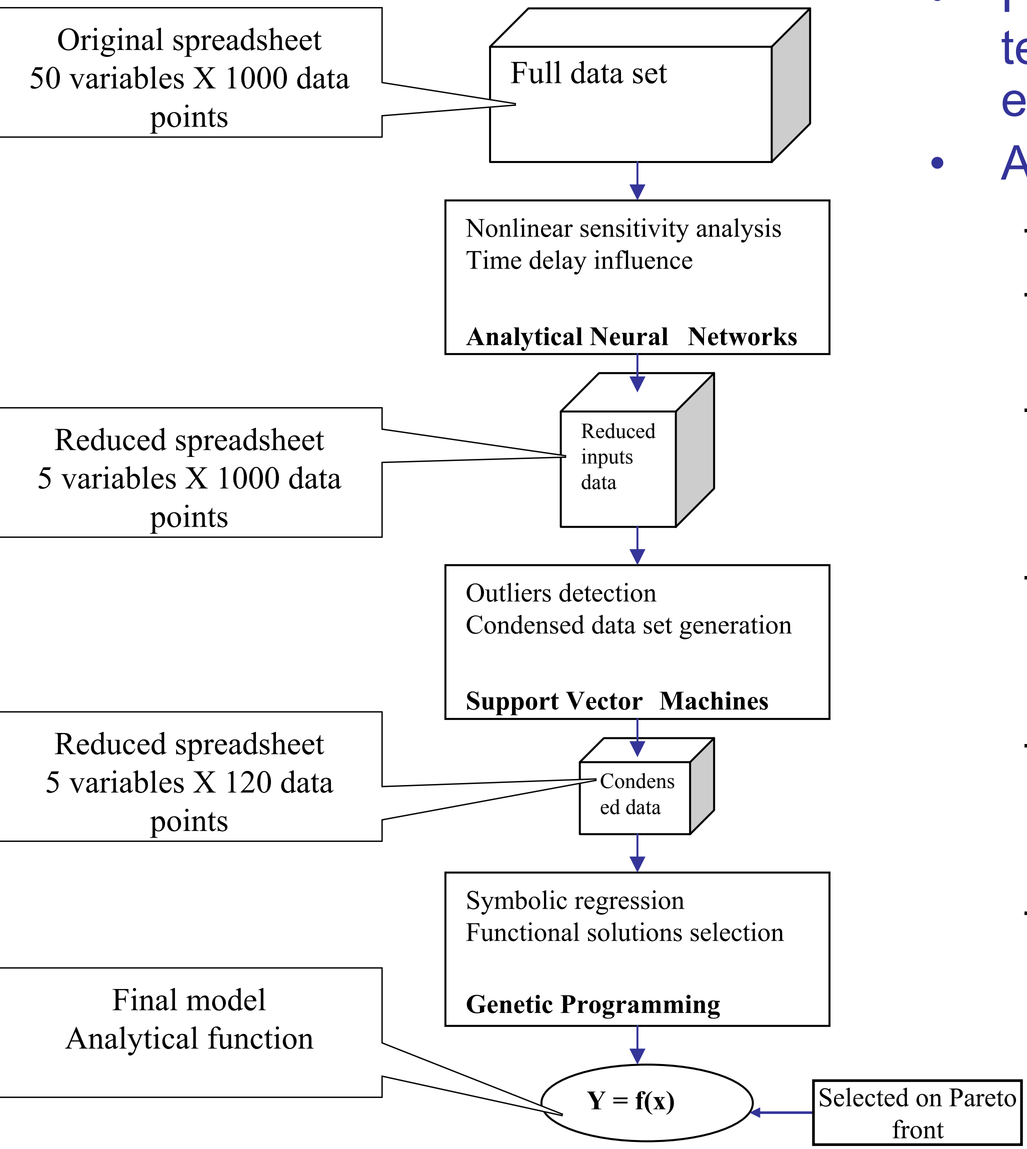


- Integrated methodology for successful EC implementation
- Related approaches
- A case study

Integrated Methodology



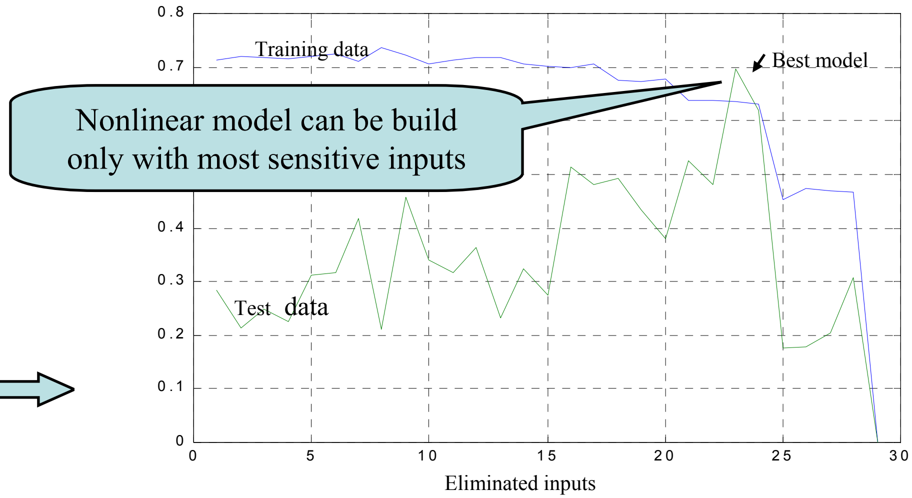
Integrated Methodology for Empirical Models Development

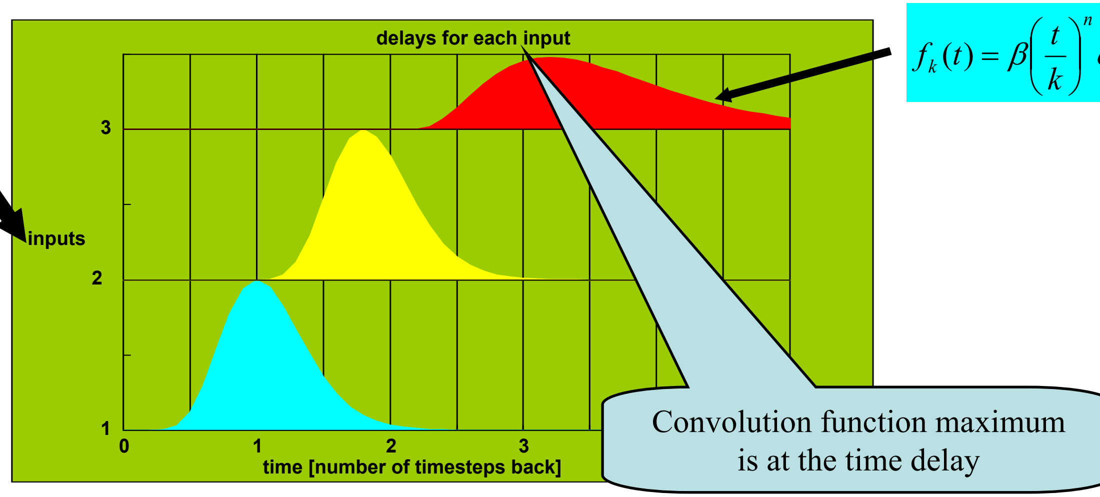


- Hybrid approach integrating multiple technologies exploits the strengths of each
- Advantages:
  - Fast development (days)
  - Robust performance (compact models)
  - Direct implementation in any Distributed Control System (no need for specialized software)
  - Very low capital cost (only if hardware for data collection is unavailable)
  - Low average cost of ownership (reduced development and maintenance cost)
  - Process engineers like it (preferable to black-box models)

Steps Based on Analytic Neural Nets

Representative data collection
↓
Data preprocessing and classification
↓
NN sensitivity analysis of all inputs
↓
Convolution parameters' estimation
↓
Outlier detection and data set condensation
↓
GP function generation
↓
Analytical function selection/verification
↓
On-line implementation
↓
Model maintenance



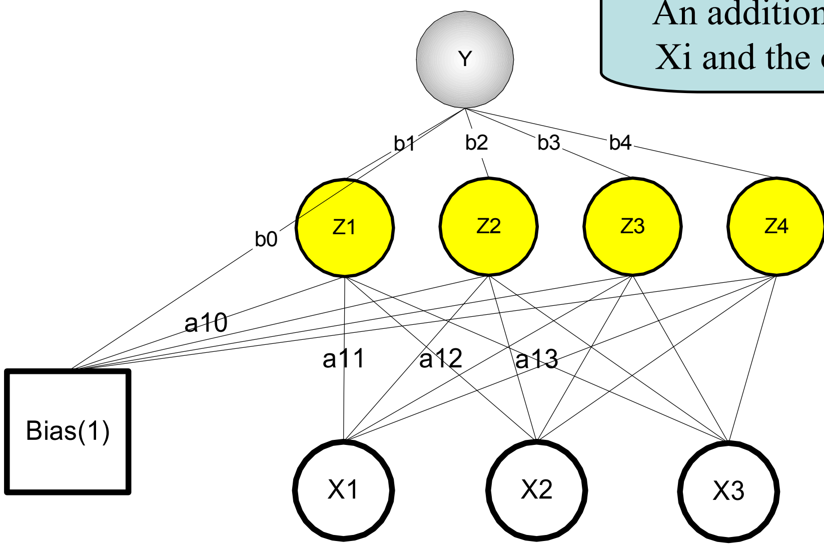


$$f_k(t) = \beta\left(\frac{t}{k}\right)^n e^{-n\left(\frac{t}{k}\right)}$$

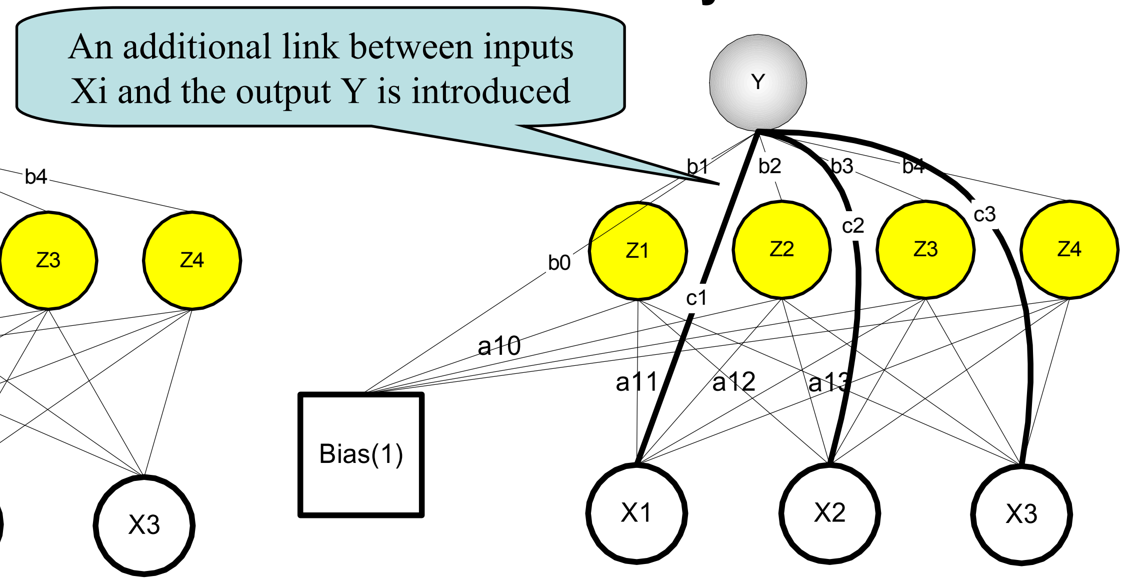
Objective: to supply GP with clean, informative, and parsimonious data set

Structural difference between classical and analytic neural networks

Classical NN



Analytical NN



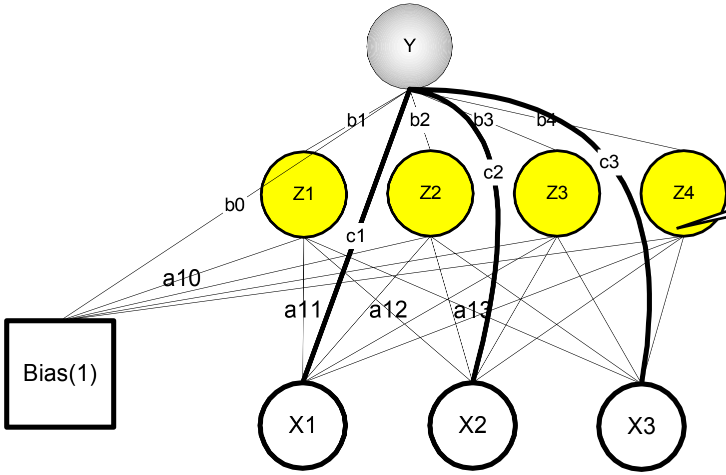
Hidden nodes calculation

$$Z_1 = F_h(a_{10} + a_{11}X_1 + a_{12}X_2 + a_{13}X_3)$$
$$Z_2 = F_h(a_{20} + a_{21}X_1 + a_{22}X_2 + a_{23}X_3)$$
$$Z_3 = F_h(a_{30} + a_{31}X_1 + a_{32}X_2 + a_{33}X_3)$$
$$Z_4 = F_h(a_{40} + a_{41}X_1 + a_{42}X_2 + a_{43}X_3)$$
$$Y = F_o(b_0 + b_1Z_1 + b_2Z_2 + b_3Z_3 + b_4Z_4)$$

$$Z_1 = F_h(a_{10} + a_{11}X_1 + a_{12}X_2 + a_{13}X_3)$$
$$Z_2 = F_h(a_{20} + a_{21}X_1 + a_{22}X_2 + a_{23}X_3)$$
$$Z_3 = F_h(a_{30} + a_{31}X_1 + a_{32}X_2 + a_{33}X_3)$$
$$Z_4 = F_h(a_{40} + a_{41}X_1 + a_{42}X_2 + a_{43}X_3)$$
$$Y = F_o(b_0 + b_1Z_1 + b_2Z_2 + b_3Z_3 + b_4Z_4 + c_1X_1 + c_2X_2 + c_3X_3)$$

Key idea behind analytic neural networks

If input-to-hidden layer weights aij are fixed, there is an analytical solution for the weights bi and ci



$$Z_1 = \mathrm{F}_h(a_{10} + a_{11}X_1 + a_{12}X_2 + a_{13}X_3)$$

$$Z_2 = \mathrm{F}_h(a_{20} + a_{21}X_1 + a_{22}X_2 + a_{23}X_3)$$

$$Z_3 = \mathrm{F}_h(a_{30} + a_{31}X_1 + a_{32}X_2 + a_{33}X_3)$$

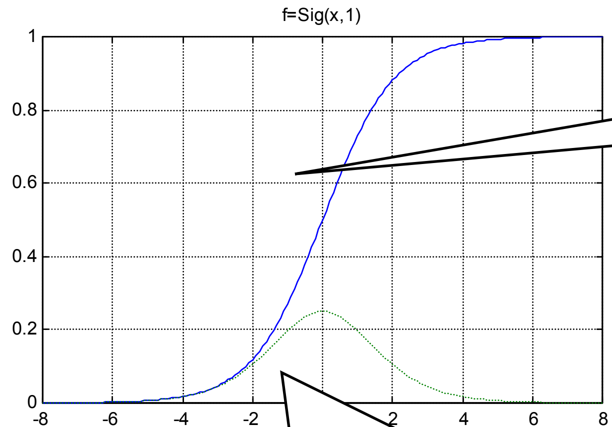
$$Z_4 = \mathrm{F}_h(a_{40} + a_{41}X_1 + a_{42}X_2 + a_{43}X_3)$$

$$Y = \mathrm{F}_o(b_0 + b_1Z_1 + b_2Z_2 + b_3Z_3 + b_4Z_4 + c_1X_1 + c_2X_2 + c_3X_3)$$

$$F_o^{-1}(Y) = [1\ X\ Z] * \begin{bmatrix} b_0 \\ c_i \\ b_j \end{bmatrix}$$

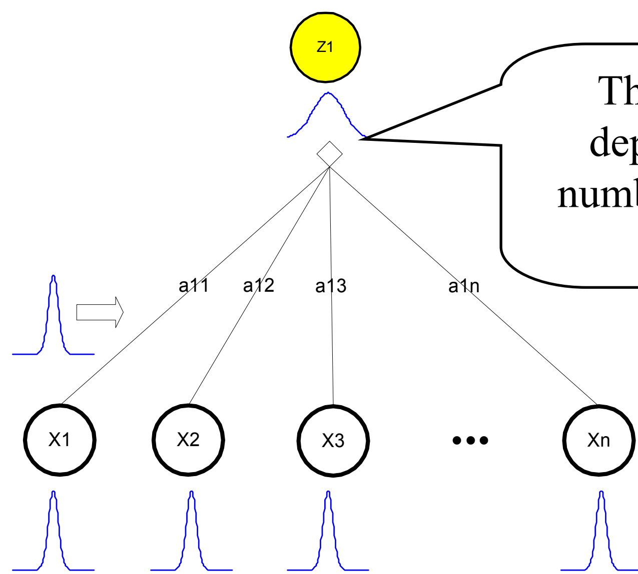
Standard linear regression problem
X – inputs data matrix **(known)**
Z – hidden layer values vector **(known)**
Unique least-squares solutions for bi and ci

Key technique for input-to-hidden layer initialization



Hidden nodes have to be within the active region of the nonlinear function

The width of the active zone is defined by the steepness of the function or the "temperature"



The "temperature" depends also on the number of inputs to the hidden node

Empirical expression for a normalized "temperature" of a sigmoid function

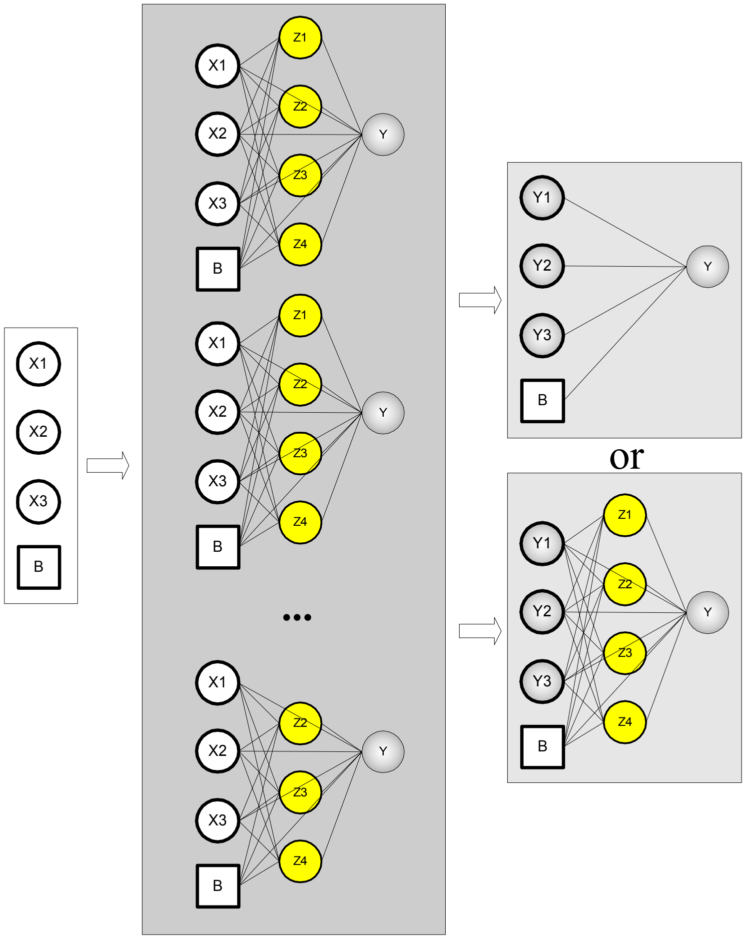
$$Tn = \eta.\frac{\log(2+\sqrt{3})}{\sqrt{ni-0.5}}$$

Weights from the input-to-hidden layer are Sampled from a normal distribution

Analytic Neural Network Benefits

- **Robust** algorithm
  - No tunable parameters
  - One **global** optimum
- Very **fast**,
  - possible to use a whole range of cross-validation principles from statistics
  - No longer an NP-complete problem
- Strong **theoretical foundation**
  - statistical learning theory
  - Direct measure for the model capacity (VC-dimension)

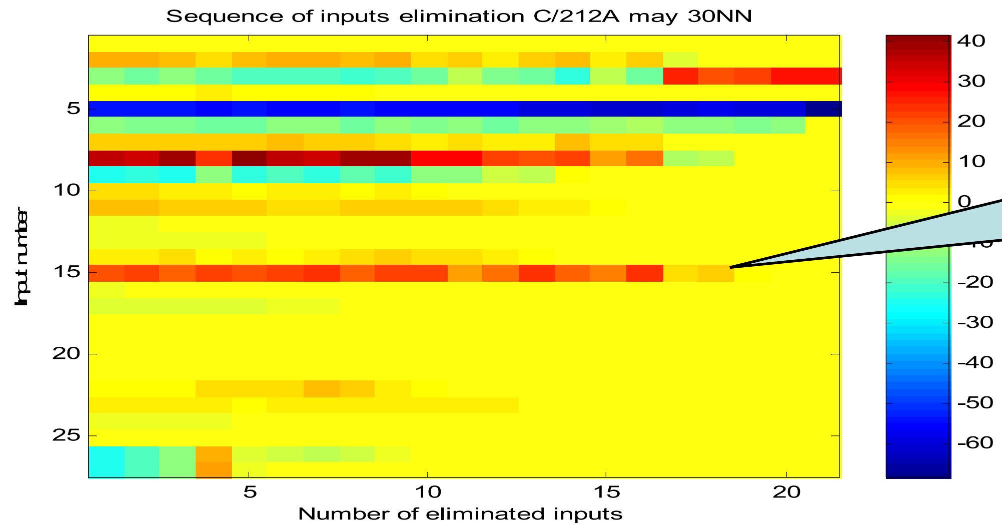
Stacked Analytic Neural Nets (SANN)



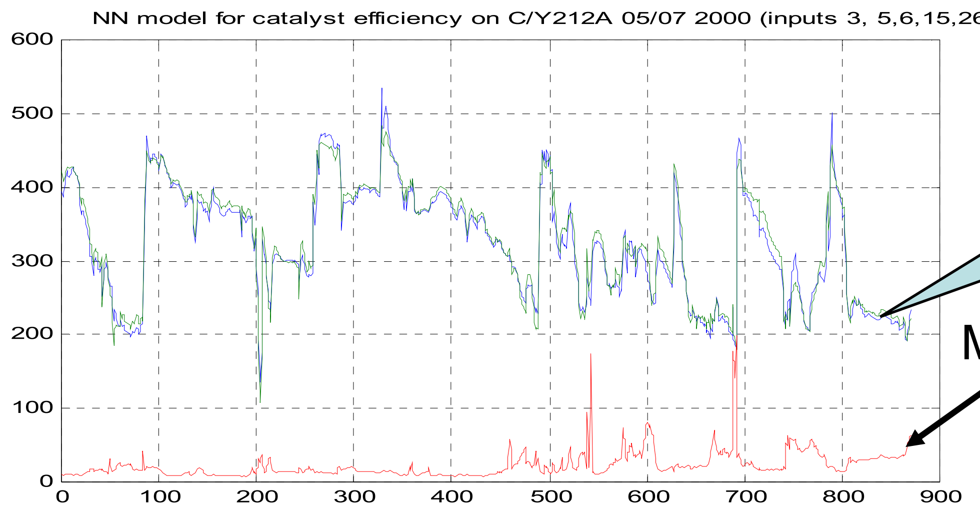
- Fast development
- Diverse subnet consensus indicator of model output quality
- Allows explicit calculations of input/output sensitivity
- Can handle time-delayed inputs by convolution functions
- Gives more reliable estimates based on multiple models statistics

Internally developed in Dow Chemical by Guido Smits

An example of stacked analytical NN application - a model for catalyst efficiency



Sensitivity analysis of various process parameters on catalyst efficiency

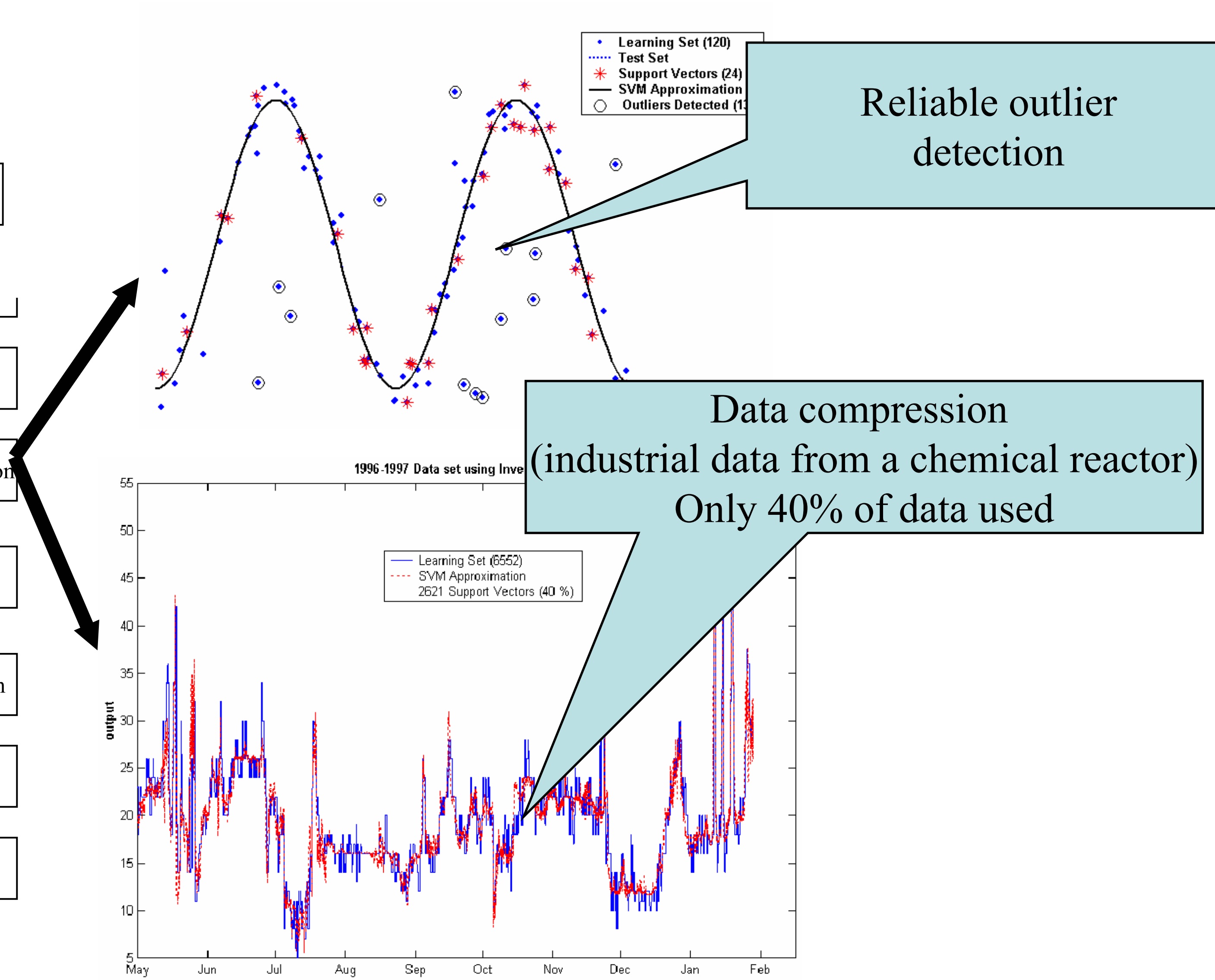


NN model performance with model disagreement indicator

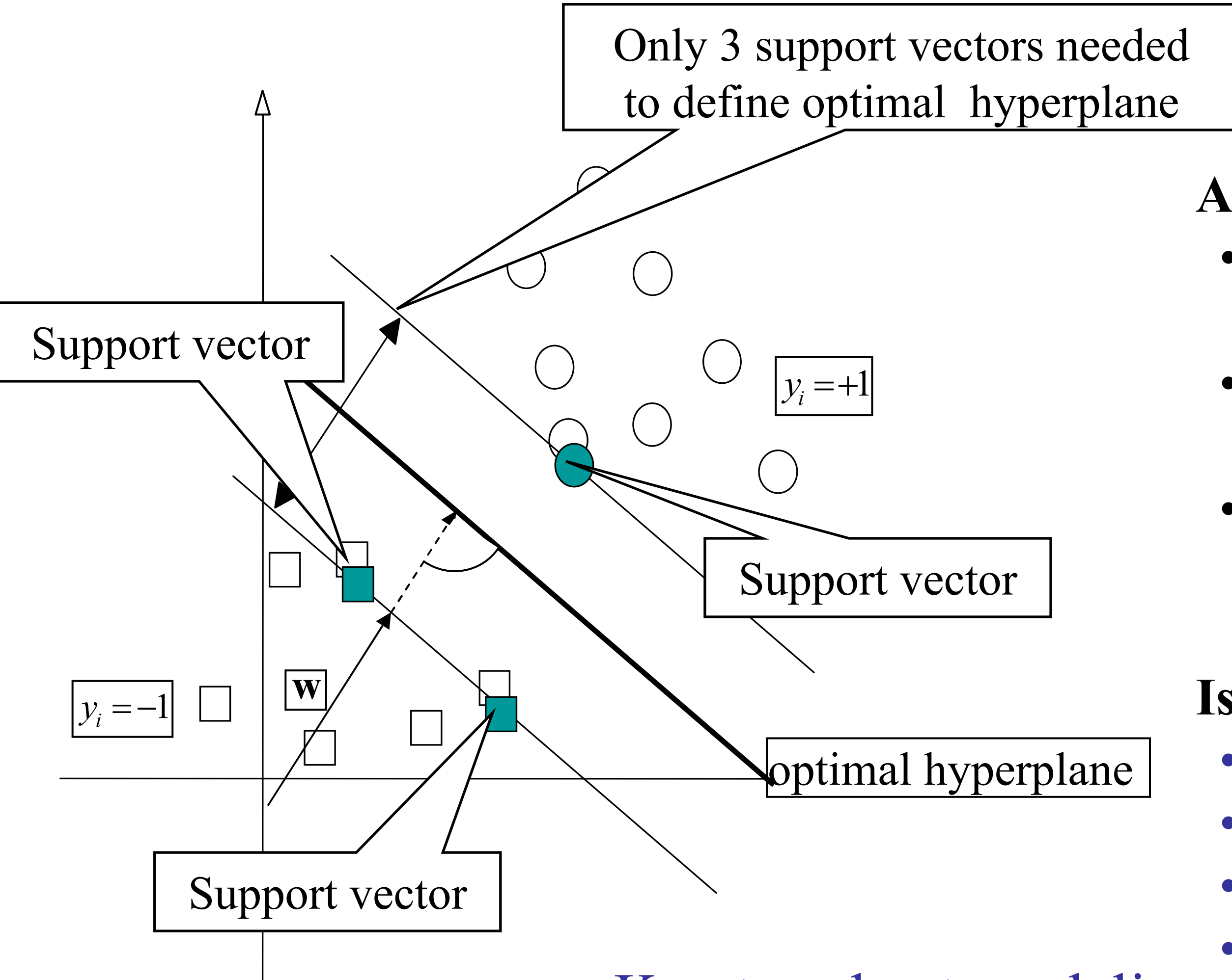
Model disagreement indicator

Steps Based on Support Vector Machines

- Representative data collection
- Data preprocessing and classification
- NN sensitivity analysis of all inputs
- Convolution parameters' estimation
- Outlier detection and data set condensation
- GP function generation
- Analytical function selection/verification
- On-line implementation
- Model maintenance



Support Vector Machines

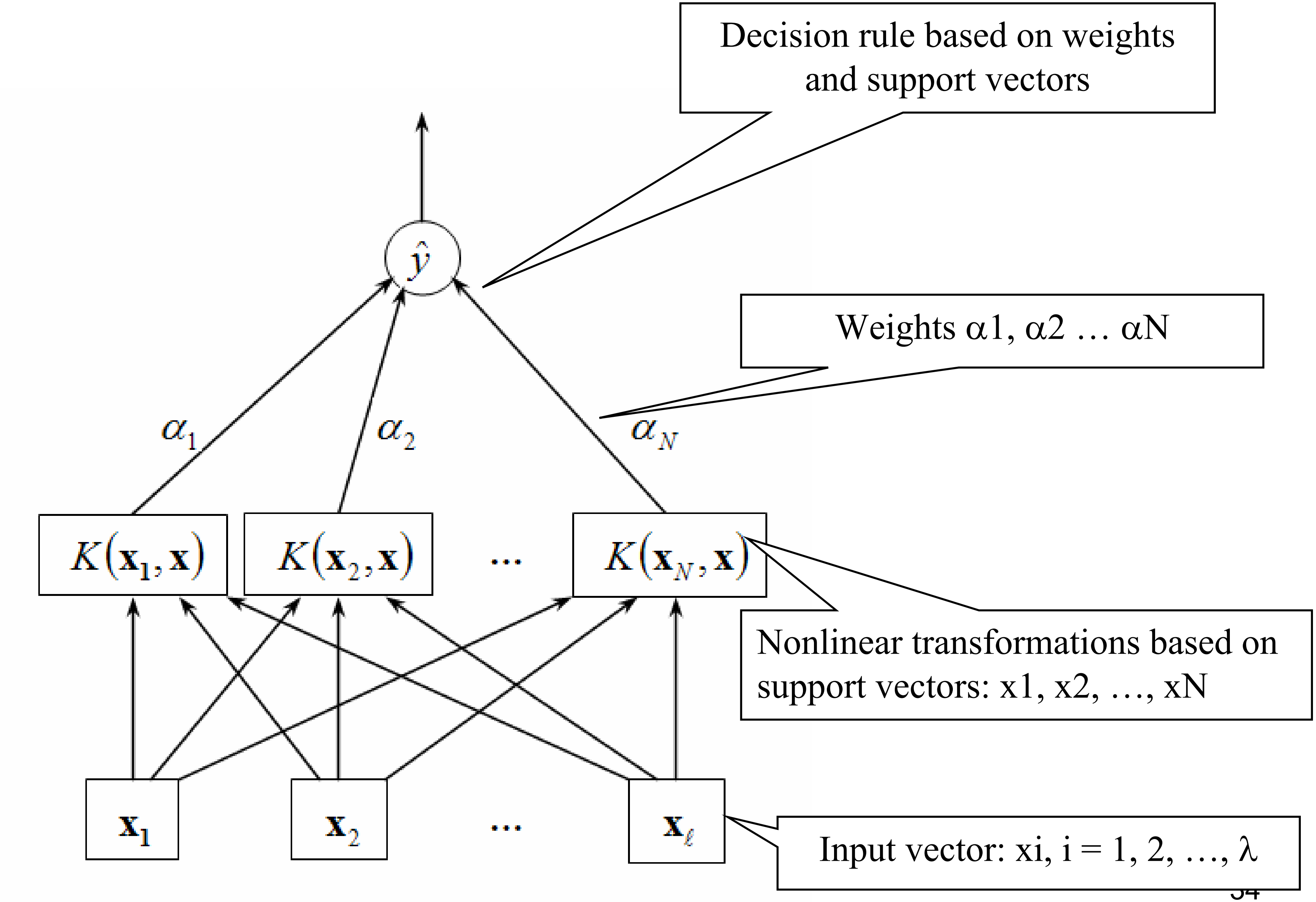


Advantages
- Solid theoretical basis => Statistical Learning Theory
- Model building is based on global optimum
- Explicit control over model complexity

Issues
- *ad hoc* Kernel selection
- Complex theory
- No commercial software
- Computationally intensive

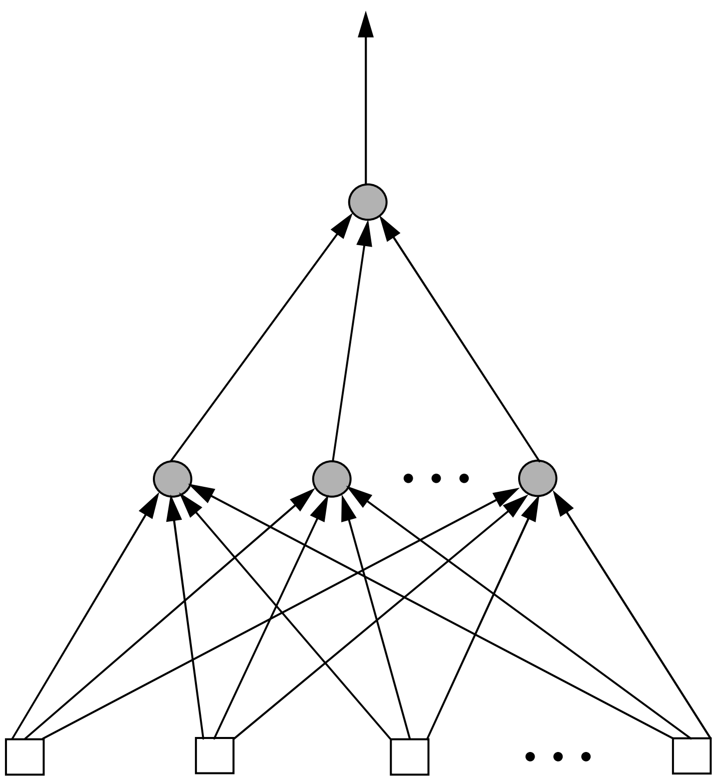
Key to robust modeling

The generic scheme of SVM

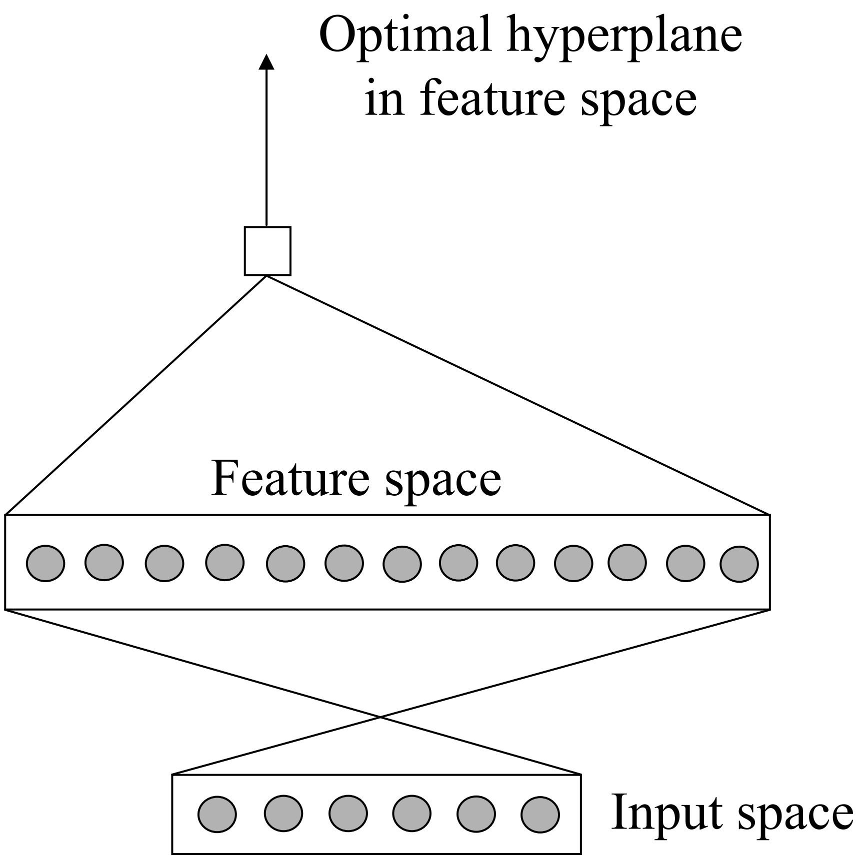


Support Vector Machines and Neural Networks

Neural Network



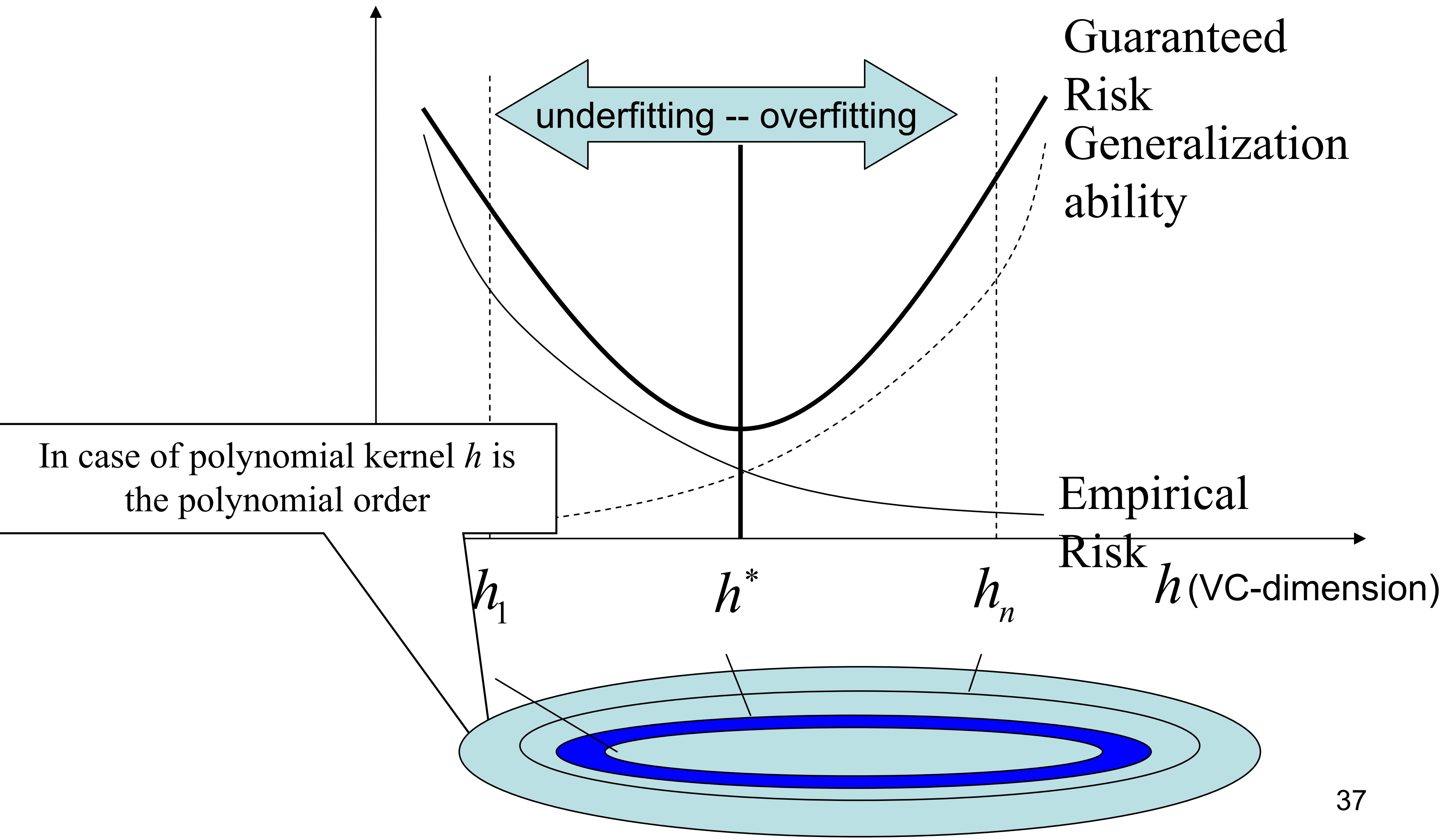
Support Vector Machine



VC-dimension

- In general, VC-dimension does not coincide with the number of parameters (can be larger or smaller)
- VC-dimension of the set of functions is responsible for the generalization ability of learning machines
- Opens remarkable opportunities to overcome the “curse of dimensionality” (large number of parameters, but low VC-dimension)

Structural Risk Minimization Principle



Structural Risk Minimization Principle

- Trade-off between quality of approximation of the given data and the complexity of the approximating function.
- The VC-dimension is now a controlling variable
- Chooses the set of functions with the lowest VC-dimension for which minimizing the empirical risk gives the best bound on the actual risk.
- Minimize

$$R(\alpha) \leq R_{\mathrm{emp}}(\alpha) + \Phi\left(\frac{\ell}{h}\right)$$

Prediction error

Complexity

Where α is the model parameter of interest, *l* is the sample size and *h* is the complexity measure

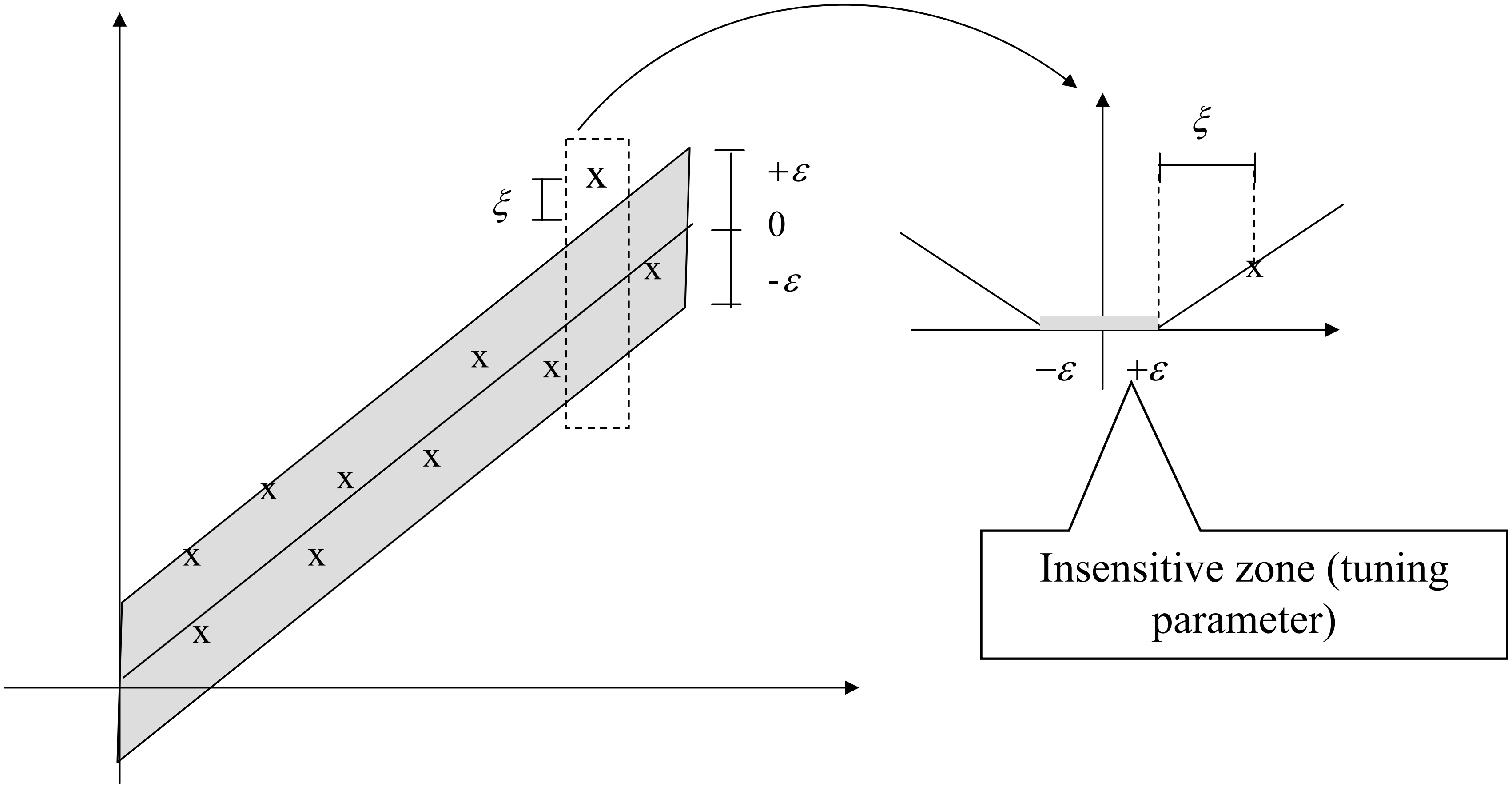
Structural Risk Minimization in learning algorithms

- Keep $\Phi\left(\frac{\ell}{h}\right)$ fixed, minimize $R_{\text{emp}}(\alpha)$
  - Neural Networks

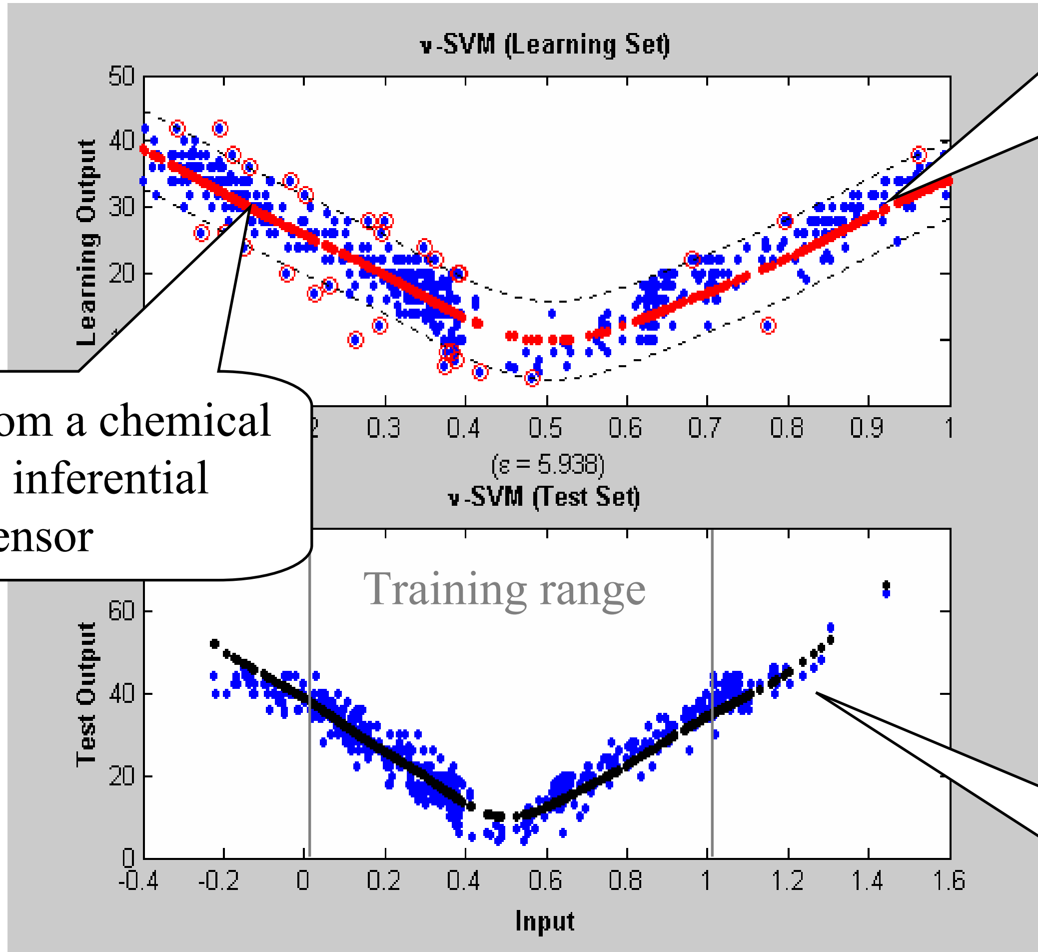
- Keep $R_{\text{emp}}(\alpha)$ fixed, minimize $\Phi\left(\frac{\ell}{h}\right)$
  - Support Vector Machines

Neural Networks and Support Vector Machines are two sides of the same coin

SVM for Regression: Constructing a tube



Generalization capabilities of SVM based on mixed kernels



Support Vector Machine model based on mixed polynomial and RBF kernels

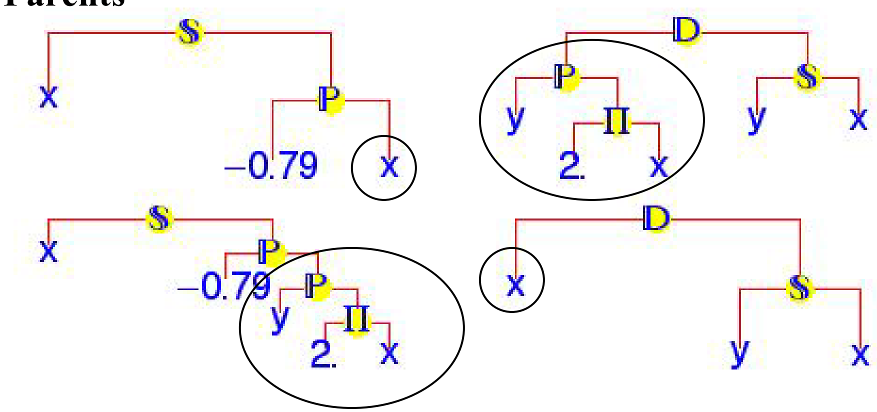
Data set from a chemical reactor inferential sensor

Impressive generalization 50% outside the training range!

Genetic Programming

Genome Tree Plots

Parents



Example of Crossover Operation

Phenotypes (Expressions)

Parents

$$-(-0.787701)^{x} + x \qquad \frac{y^{2}\,x}{-x+y}$$

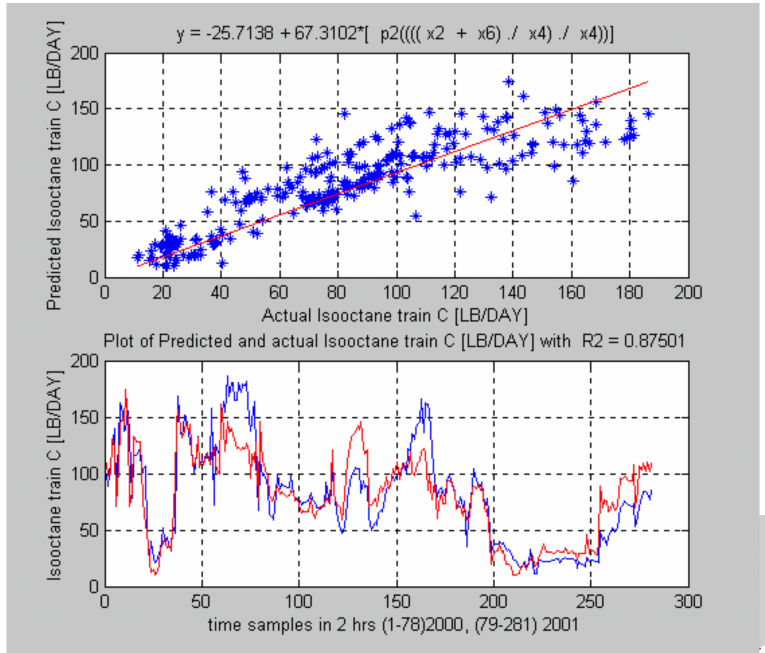
Children

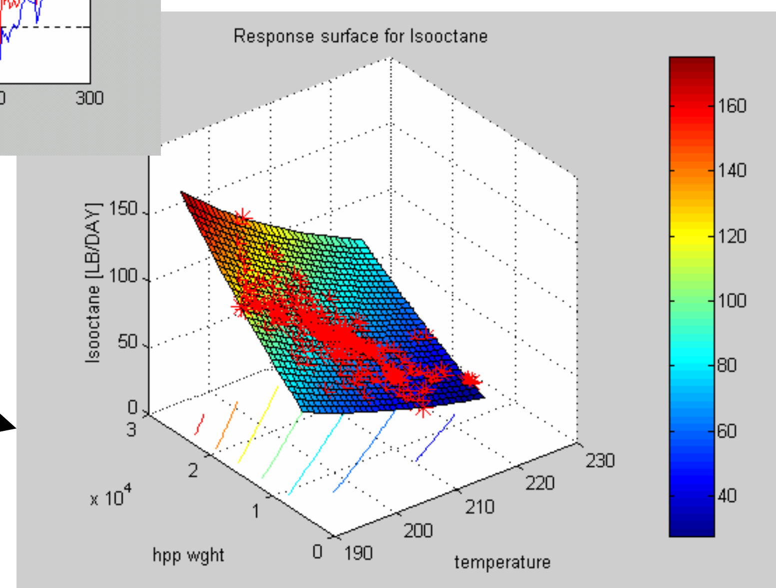
$$-(-0.787701)^{y^{2}\,x} + x \qquad \frac{x}{-x+y}$$

- Based on artificial evolution of millions of potential nonlinear functions => survival of the fittest
- Many possible solutions with different levels of complexity
- The final result is an explicit nonlinear function
- Better generalization capabilities than neural nets
- Low implementation requirements
- Time delays
- Sensitivity analysis of large data sets
- Relatively slow (several hours of computational time)

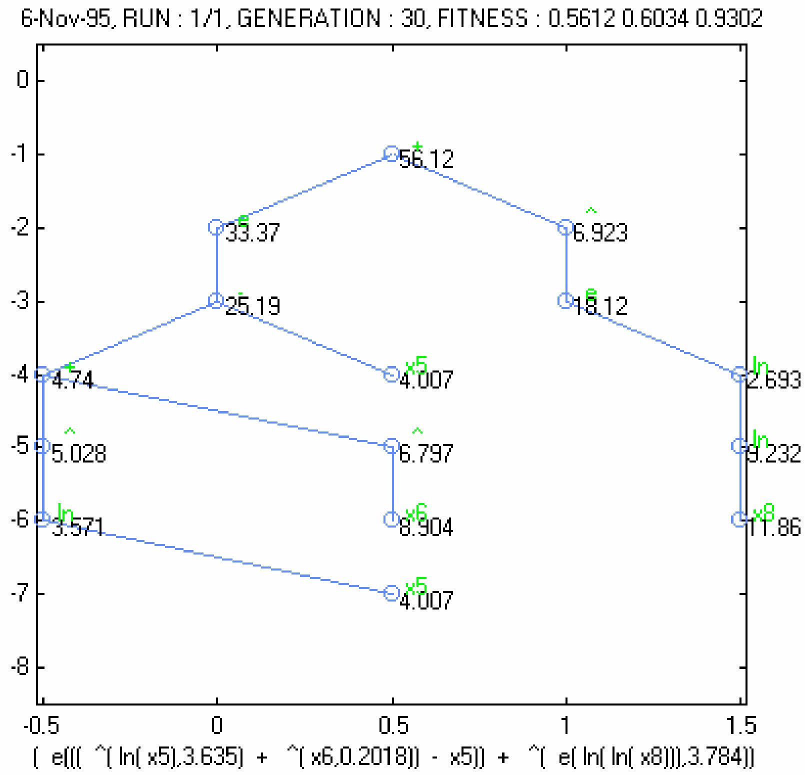
Steps Based on Genetic Programming

1. Representative data collection
2. Data preprocessing and classification
3. NN sensitivity analysis of all inputs
4. Convolution parameters' estimation
5. Outlier detection and data set condensation
6. GP function generation
7. Analytical function selection/verification
8. On-line implementation
9. Model maintenance





Problem 1: Where are the Building Blocks in GP? Does the Schema Theorem apply?

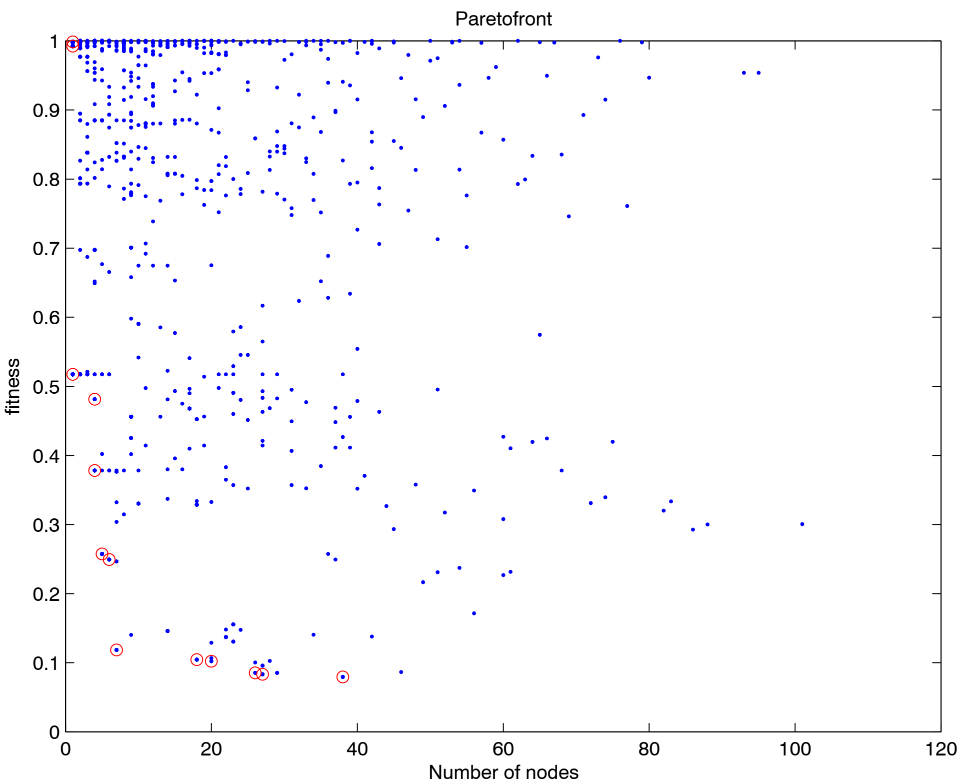


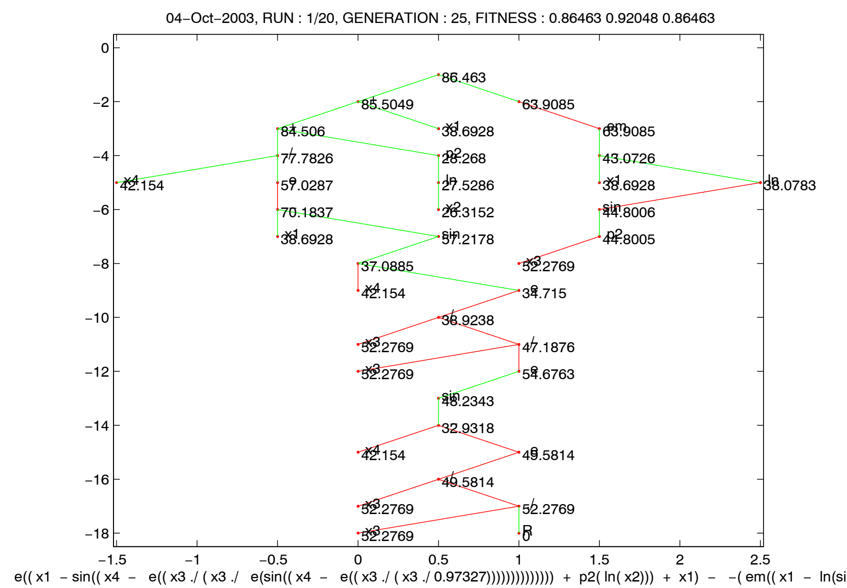
- We are working with dynamic structures that can arbitrarily grow in size.
- We're doing Empirical Risk Minimization on a small subset of the available information (we ignore all the sub equations).

Problem 2: How do we make sure the Structural Risk Minimization Principle Applies?

- SRM = Trade-off between quality of approximation of the given data en the complexity of the approximating function.
- Can we determine something like the VC-dimension for an arbitrary tree-structure?
- Can we choose the set of functions with the lowest VC-dimension for which minimizing the empirical risk gives the best bound on the actual risk?
- Minimize $R(\alpha) \leq R_{\text{emp}}(\alpha) + \Phi(\frac{\ell}{h})$

New Approach to GP: Optimize the Pareto front of Fitness vs. Complexity instead of just Fitness.

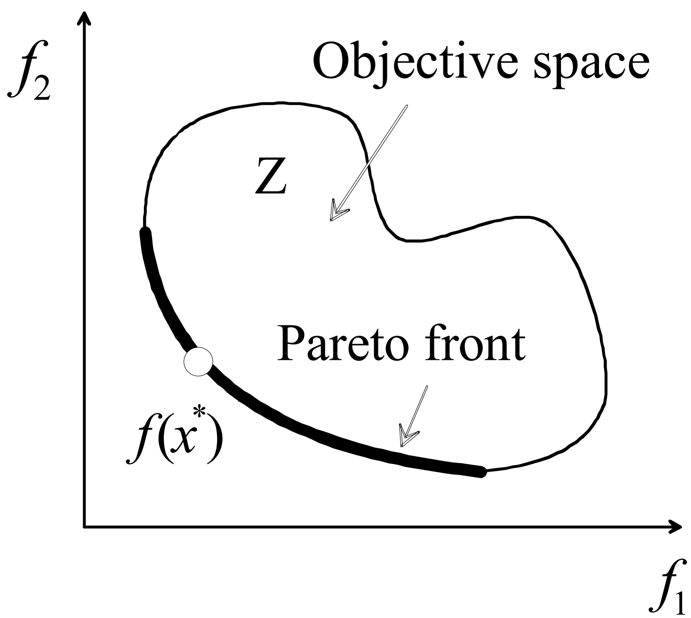




All sub-equations are also taken into account. This results in effective population sizes of a few thousand instead of a few hundred with no additional computational cost.

Pareto Optimality

A decision vector $x^* \in S$ is Pareto optimal if there does not exist another $x \in S$ such that $f_i(x) \leq f_i(x^*)$ for all $i = 1, \ldots, k$ and $f_j(x) < f_j(x^*)$ for at least one index *j*.



Task in (Multi-Objective Optimization Problem- MOOP): Determine the Pareto front

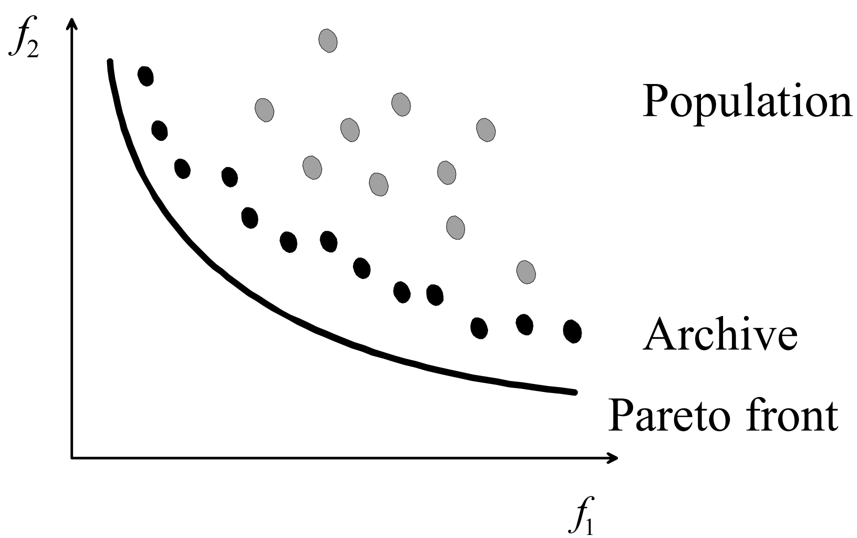
The Standard GP is Extended with an Archive

Archive (t) = set of *best* equations found so far during the run

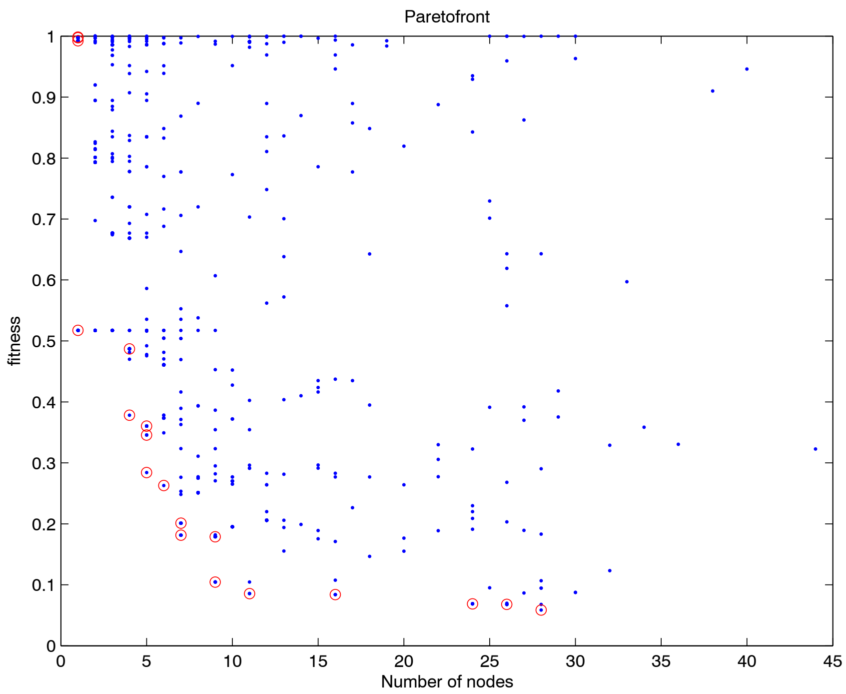
= *best* estimate of the Pareto front

$$\text{Archive}(0) = \textit{Best}(\text{Initial equations from the population})$$

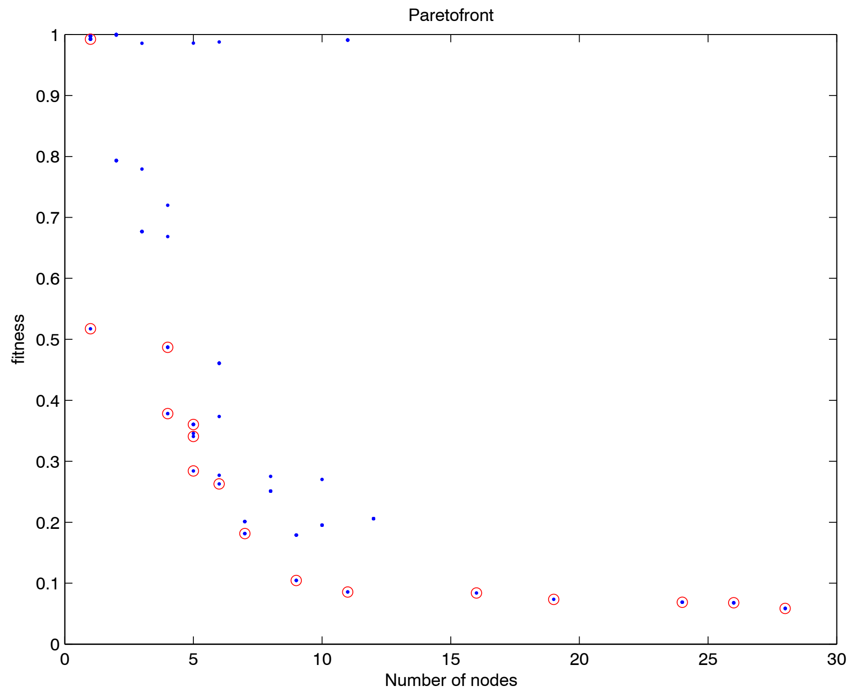
$$\text{Archive}(t+1) = \textit{Best}\ (\text{Archive}(t) \bigcup \text{Current paretofront of the population})$$



Crossover only occurs between the Members of the Population and the Archive



Population



Archive

- This ensures very quick propagation of the building blocks through the population.
- Population Diversity is always high by construction.

Both features result in a much more effective exploration of the function space.

Post run Analysis is much faster (The focus is on the Pareto Front Population)

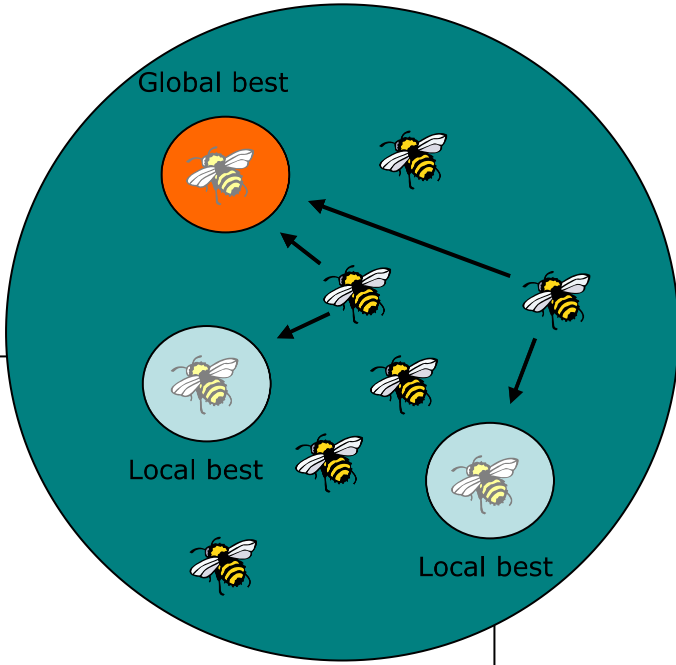
	Norm.	Adj.	Raw	Complexity	Vars	Function
1	0.	0.663	0.663	1	x2	x_2
2	0.205	0.731	0.731	5	x1 x2	$x_1 - x_2$
3	0.516	0.833	0.833	11	x2	$4.^{-x_2}\, x_2$
4	0.608	0.864	0.864	19	x2	$(1.955^{x_2})^{-x_2}\, x_2$
5	0.661	0.881	0.881	25	x1 x2	$\frac{x_1}{x_2^4 + 2. x_1 + 4.}$
6	0.767	0.916	0.916	33	x1 x2	$\frac{x_1 x_2}{x_2^5 + 2. x_1 x_2 + 1.955}$
7	0.86	0.947	0.947	41	x1 x2	$\frac{x_1}{x_2^4 + 2.^{x_1} + 1.955 + \frac{1.955}{x_2}}$
8	0.884	0.955	0.955	49	x1 x2	$\frac{x_1}{x_2^4 + 2.^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}}$
9	0.939	0.973	0.973	51	x1 x2	$\frac{x_1}{0.512^{x_1} x_1^{x_1} + x_2^4 + \frac{2.}{x_2} + 1.955}$
10	0.956	0.979	0.979	59	x1 x2	$\frac{x_1}{0.512^{x_1} x_1^{x_1} + x_2^4 - x_2 + \frac{1.955}{x_2} + 1.955}$
11	0.962	0.981	0.981	61	x1 x2	$\frac{x_1}{x_2^{1.955^{x_2}} + x_2 + 0.512^{x_1} x_1^{x_1} + \frac{1.955}{x_2}}$
12	0.98	0.986	0.986	69	x1 x2	$\frac{x_1}{x_2^{1.955^{x_2}} + 0.512^{x_1} x_1^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}}$
13	0.992	0.99	0.991	99	x1 x2	$\frac{x_1}{(x_2^4 + x_1 + 8.)\left(0.512^{x_1} x_1^{x_1} - x_1 + x_2^{x_2} + \frac{1.955}{x_2} + 1.955\right)}$
14	0.997	0.992	0.992	121	x1 x2	$\frac{x_1}{(4. x_1^{-x_1} + 2. x_1 + 1.955)\left(x_2^{1.955^{x_2}} + 0.512^{x_1} x_1^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}\right)}$
15	1.	0.993	0.994	133	x1 x2	$\frac{x_1}{\left(4.^{x_1} \left(\frac{1.}{x_1^2}\right)^{x_1} + 2. x_1 + 1.955\right)\left(x_2^{1.955^{x_2}} + 0.512^{x_1} x_1^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}\right)}$
16	1.	0.993	0.994	149	x1 x2	$\frac{x_1}{\left(4.^{x_1} \left(\frac{1.}{x_1^2}\right)^{x_1} + 1.762 x_1\right)\left(x_2^{1.955^{x_2}} + 0.512^{x_1} x_1^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}\right)}$
17	0.99	0.99	0.994	193	x1 x2	$\frac{x_1}{\left(x_2^{1.955^{x_2}} + 0.512^{x_1} x_1^{x_1} - x_1 + 1.955 + \frac{1.955}{x_2}\right)\left(4.^{x_1} \left(\frac{1.}{x_1^2}\right)^{x_1} + x_1 \left(1.512 + \frac{1.}{x_2^{x_2} + x_2 + x_1 + \frac{1.955}{x_2}}\right)\right)}$

All equations on the Pareto front in increasing order of complexity and fitness

Advantages of Pareto-front GP

- Initial results indicate 10-100 times increased efficiency vs. conventional GP.
- Building Blocks (Transforms) are generated automatically.
- Effective population sizes are much higher with no additional computational cost.
- The post-run Analysis is much faster – Only the functions in the Archive need to be inspected.
- No need anymore for multiple runs with different levels of parsimony control.

Particle swarm optimization



An efficient technique to find the global optimum
for model inversion and non-linear parameter estimation

At each time step *t*

For each particle *i*

Update the position change (velocity)

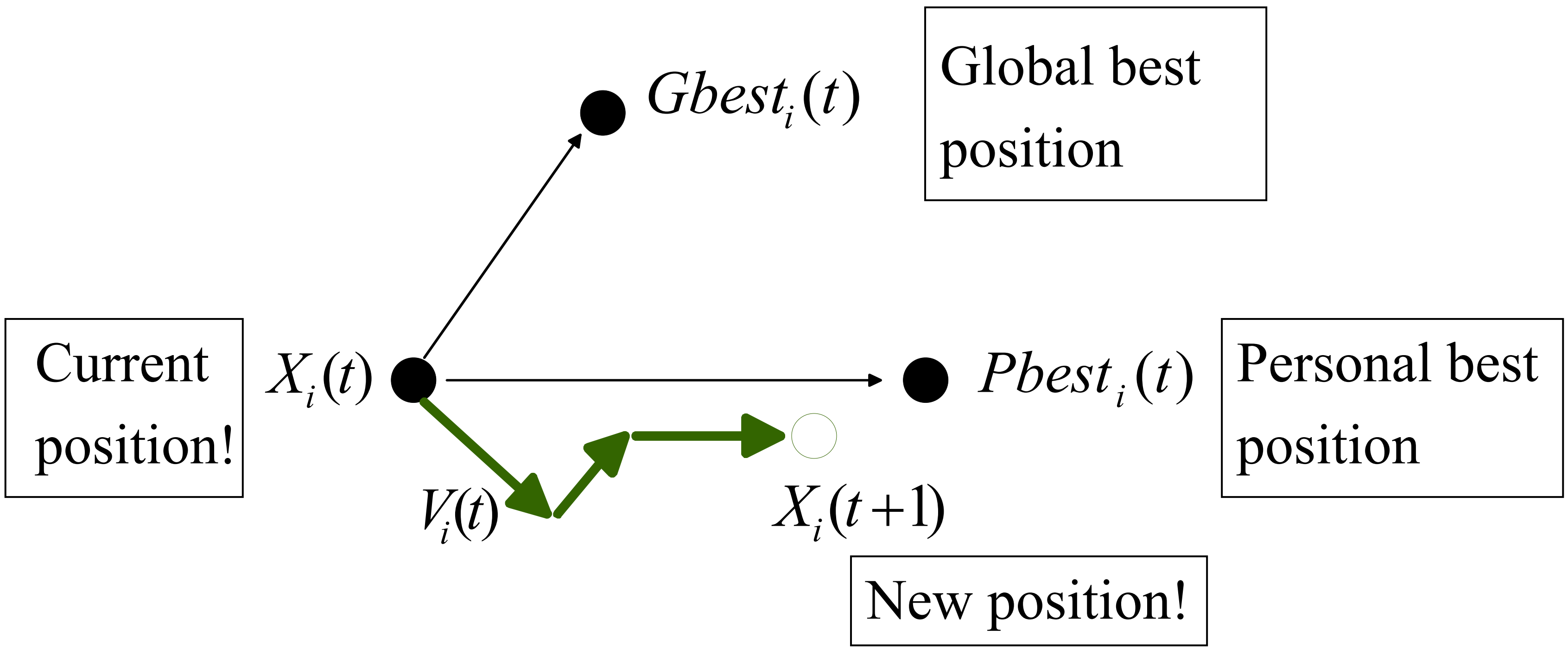
$$V_i(t+1) = \chi \cdot (V_i(t) + c_1 \cdot rand(0,1) \cdot (P_i(t) - X_i(t)) + c_2 \cdot rand(0,1) \cdot (P_g(t) - X_i(t))$$

Then move $X_i(t+1) = X_i(t) + V_i(t+1)$

Note: - stochastic component

- parameters c_1, c_2, χ default values (2.05, 2.05, 0.73)

Particle's Movement – A Compromise



Software tools

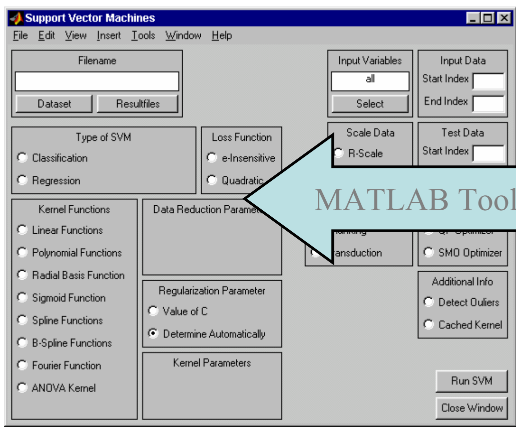
Representative data collection — DAP, Cave, IP21

Data preprocessing and classification — Excel, JMP, SIMCA, Mathematica

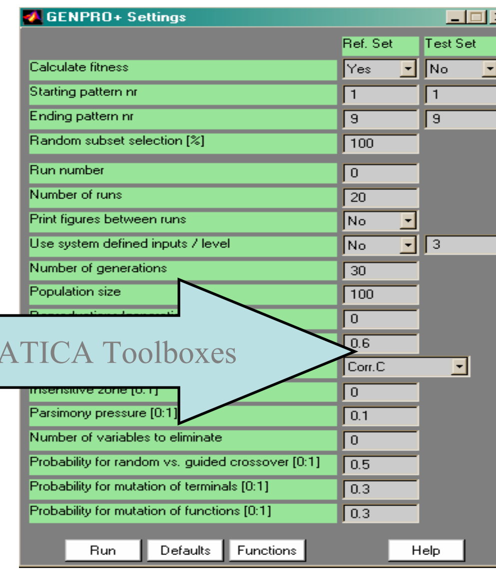
NN sensitivity analysis of all inputs — MATLAB, Excel

Convolution parameters' estimation

Outlier detection and data set condensation — MATLAB Toolbox



GP function generation — MATLAB &MATHEMATICA Toolboxes



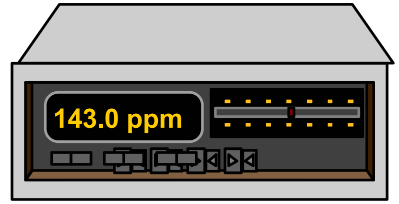
Analytical function selection/verification

On-line implementation — G2, MOD, IP21

Model maintenance — DAP, Cave, IP21

Case Study: Inferential Sensors

Key objective:
To predict difficult-to-measure parameter (melt index) from easy-to-measure data (temperature, pressure, flow, etc.)



$$\rho C_p \frac{\partial T}{\partial t} + \nabla \cdot \left(-k \nabla T + \rho C_p T \mathbf{u}\right) = Q$$

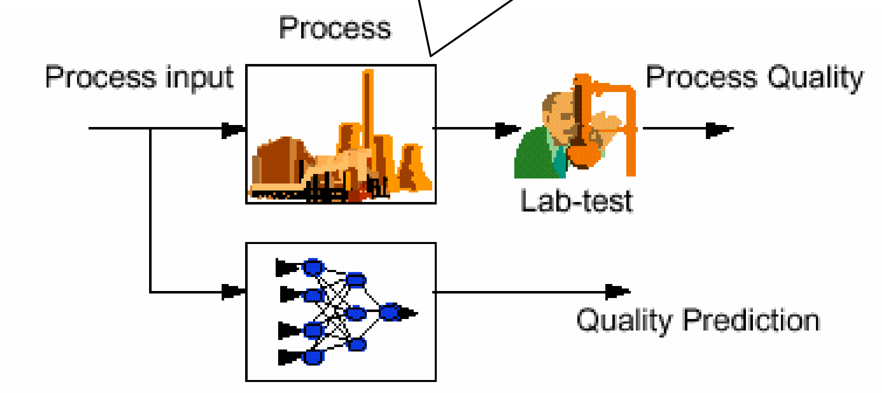
Training data

Inferential Sensors Development Software

Simple formulas

$$y = a + b \cdot \left(e \cdot \left|\frac{x3}{x5}\right| - d\right)^c$$

Easy On-Line implementation



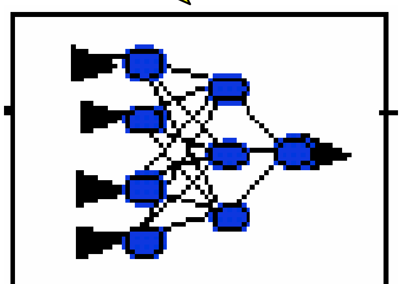
Inferential Sensor
An empirical model based on analytical equations with built-in self-assessment capability

Issues with neural net-based inferential sensors

Issues with existing neural net-based inferential sensors:

- High sensitivity to process changes
- Frequent re-training
- Complicated development & maintenance
- Low survival rate after 3 years in operation
- Engineers hate black-boxes

Black box



Specialized run-time software

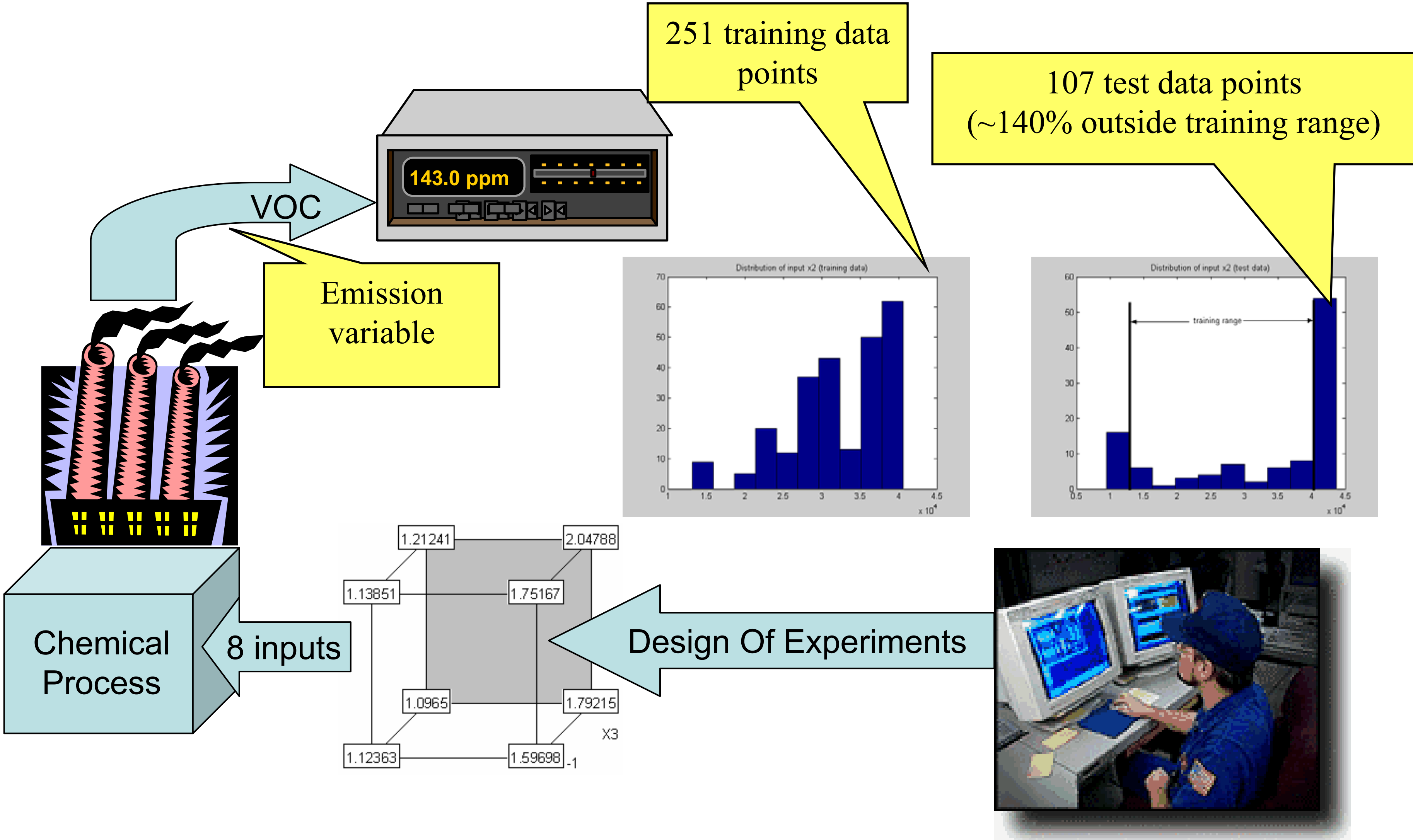
VS.

Analytical expression

$$\text{Func2} = \frac{\text{rate}^2\left(\text{vac} + \frac{\text{rate}*\text{hopp wt}}{\text{temp}}\right)\text{pllt wt temp}}{\text{density}*\text{temp}^2}$$

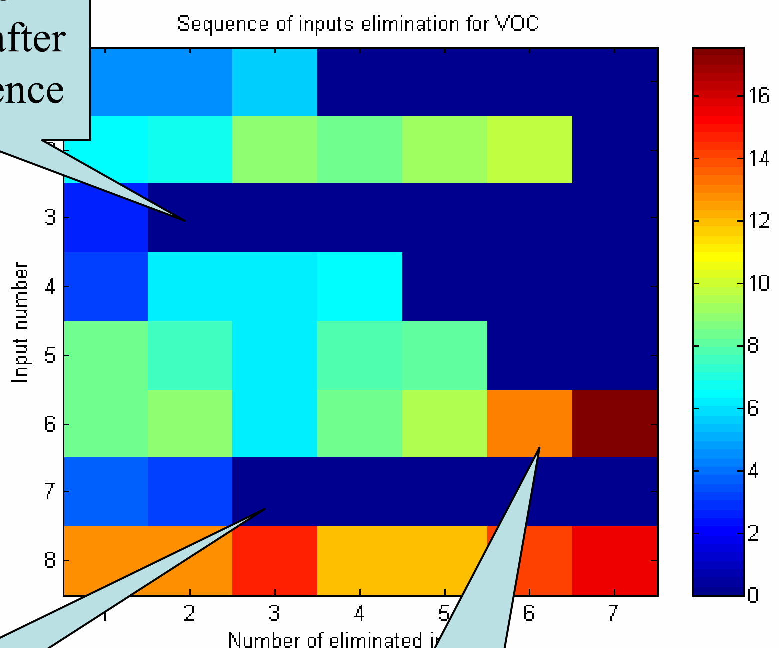
Directly coded into most on-line systems

Inferential sensor for emission monitoring: A case study
Data Collection



Inferential sensor for emission monitoring: A case study Sensitivity analysis by SANN

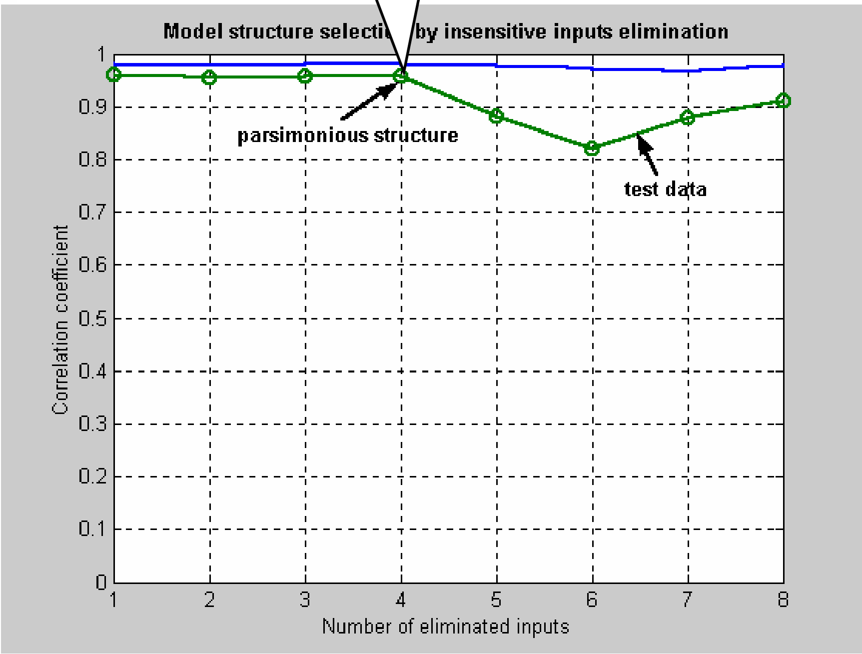
Input x3 removed after first sequence



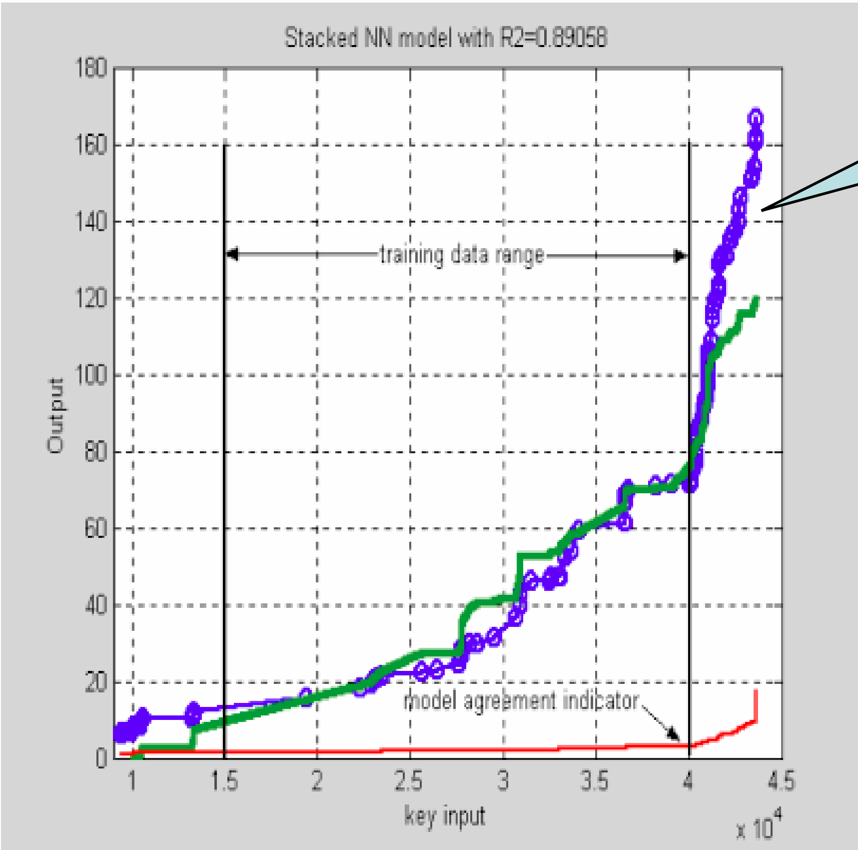
Input x7 removed after second sequence

Input x6 has the strongest sensitivity

A NN with 4 inputs: x2, x5, x6, and x8 is selected after discussion with process engineers



Inferential sensor for emission monitoring: A case study (SANN model performance)



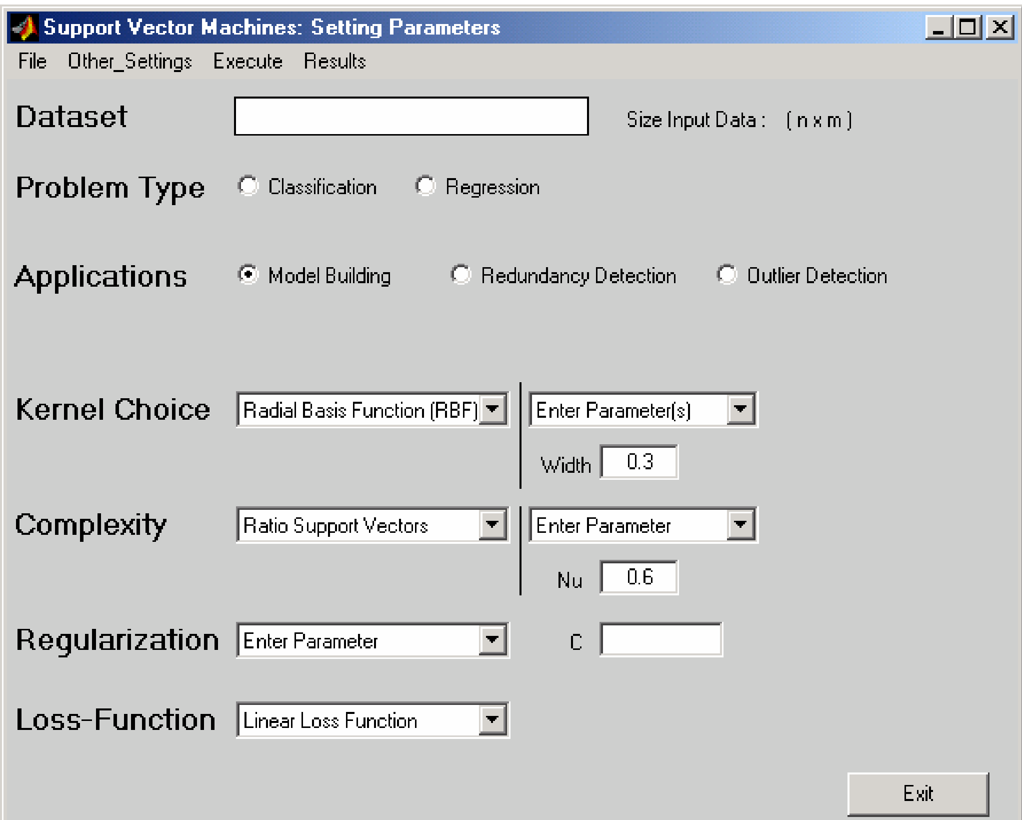
Bad extrapolation (test data is 140% outside the range of training data)

Model based on 30 stacked NN with 10 neurons in hidden layer

Reduced number of inputs from 8 to 4

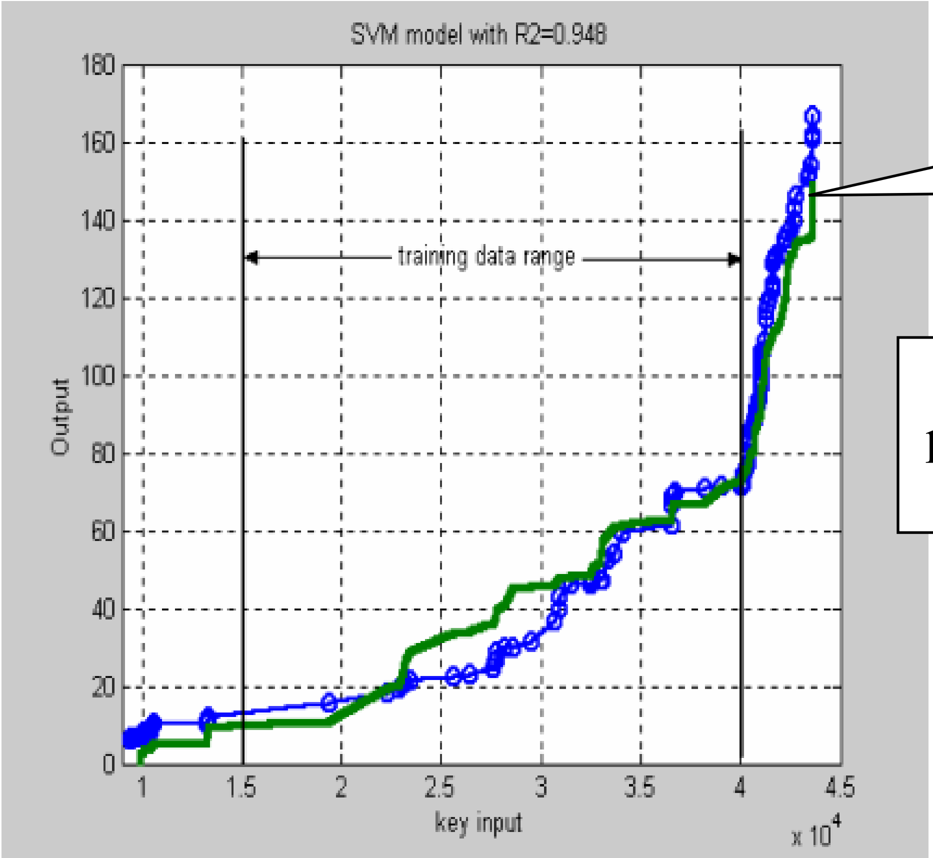
Fast test of the hypothesis about potential nonlinear relationship (in 20-30 min)

Inferential sensor for emission monitoring: A case study (SVM parameters)



Parameters:
% support vectors: 10
C = 10^6
Mixed Kernels: Polynomial and RBF
Range of Polynomial kernels: 1-3
Range of RBF kernel: 0.25-0.75
Range of ratio 0.5 – 0.99

Inferential sensor for emission monitoring: A case study (SVM model performance)

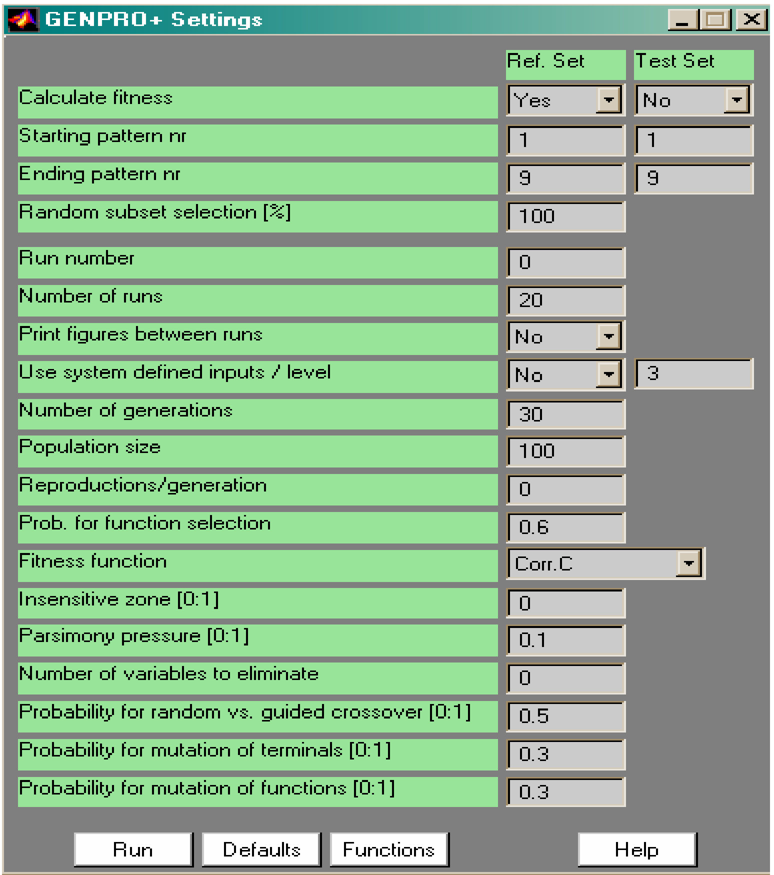


Impressive extrapolation (test data is 140% outside the range of training data)

Model based on a mixture of 2nd order polynomial global kernel and RBF local kernel with width of 0.5 and ratio of 0.95

Reduced number of training data points from 251 to 34 (based on support vectors)

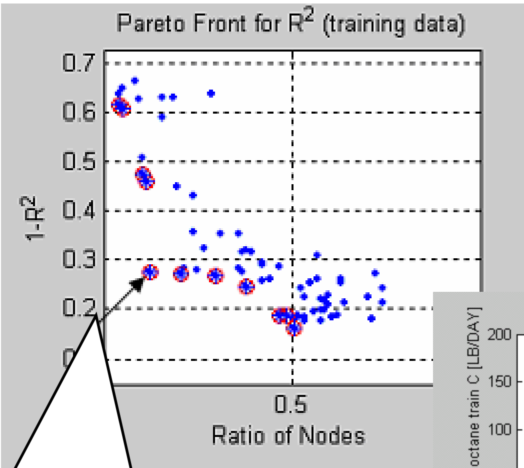
Inferential sensor for emission monitoring: A case study (GP parameters)



Parameters for a GP simulated evolution

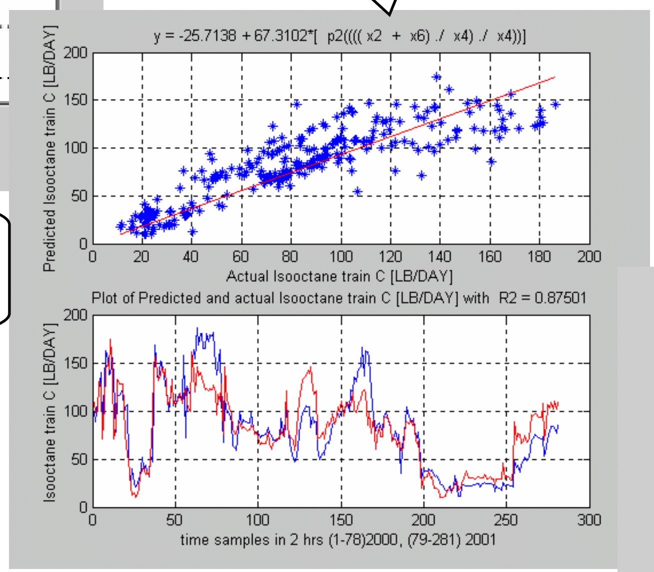
Reference data :34
Random subset selection [%] :100
Number of runs :20
Population size :500
Number of generations :100
Probability for function as next node :0.6
Optimization function :Corr.Coef
Parsimony pressure :0.1
Prob. for random vs guided crossover :0.5
Probability for mutation of terminals :0.3
Probability for mutation of functions :0.3

Inferential sensor for emission monitoring: A case study (Selected symbolic regression model)

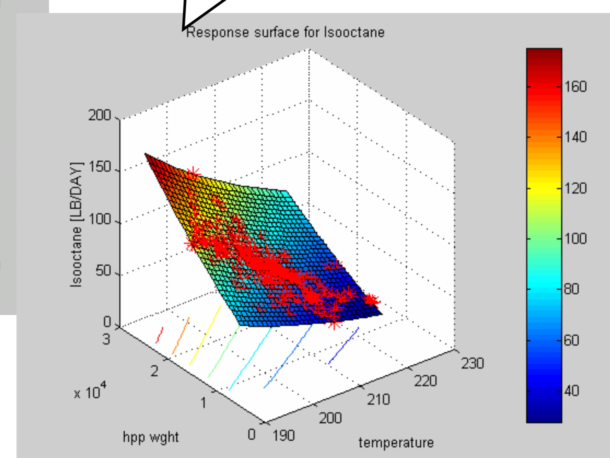


Selected model on Pareto front

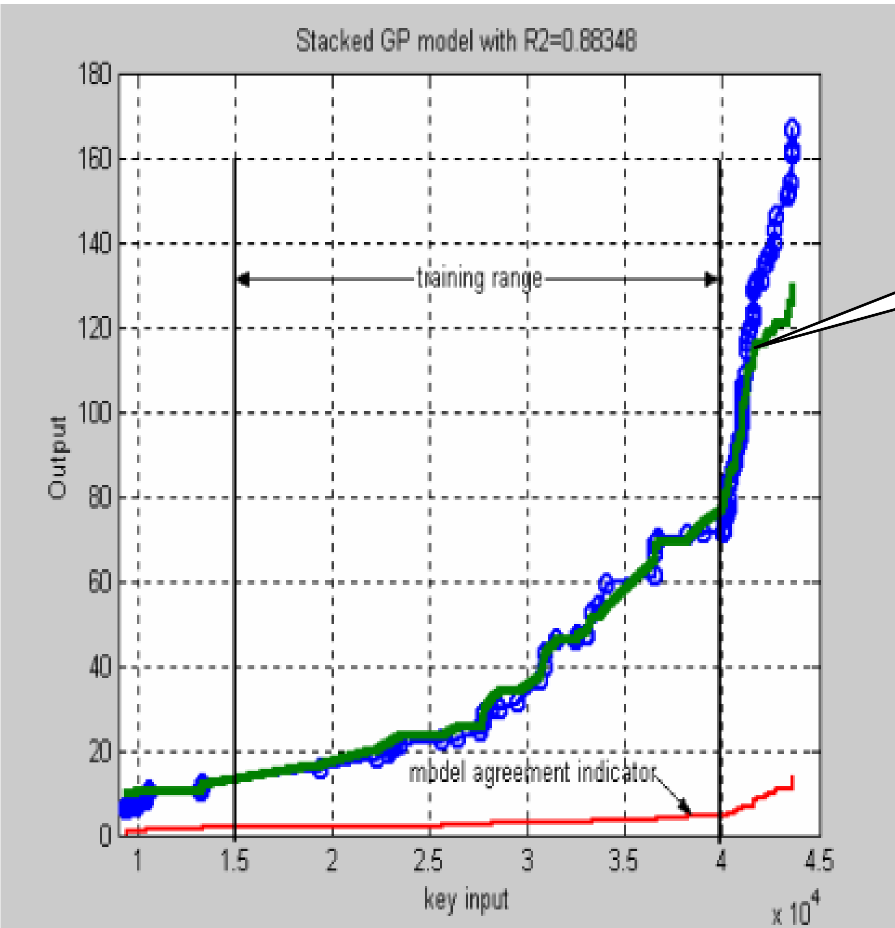
Simple expression with acceptable performance (R2 = 0.87)



Response surface of model according to process physics



Inferential sensor for emission monitoring: A case study (Final solution: Stacked Symbolic Regression model)

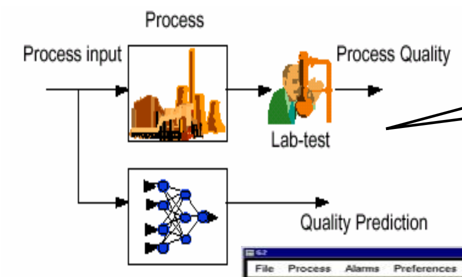


In operation since August 2001

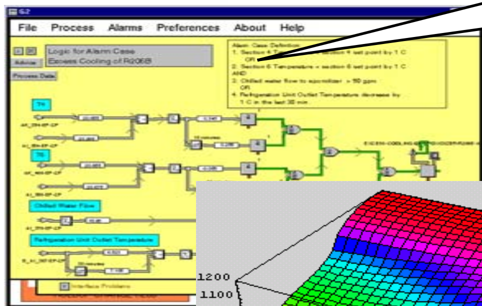
Model based on 8 Stacked Symbolic Predictors

Shorter evolutionary process based on 8.44% of the original training data set

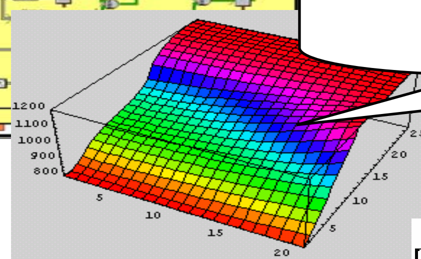
Key application areas



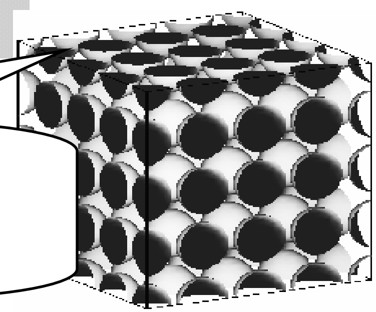
Robust Inferential Sensors
Mass-scale on-line empirical models



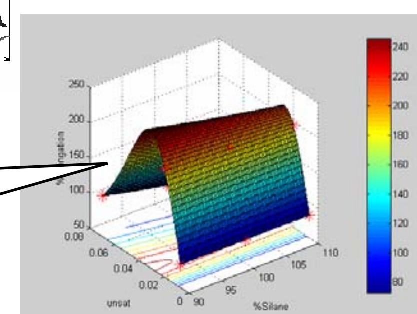
Automated Operating Discipline
Consistent intelligent on-line supervision



Empirical Emulators of Fundamental Models
Effective on-line process optimization



Fundamental model building based on GP
Accelerated new product development

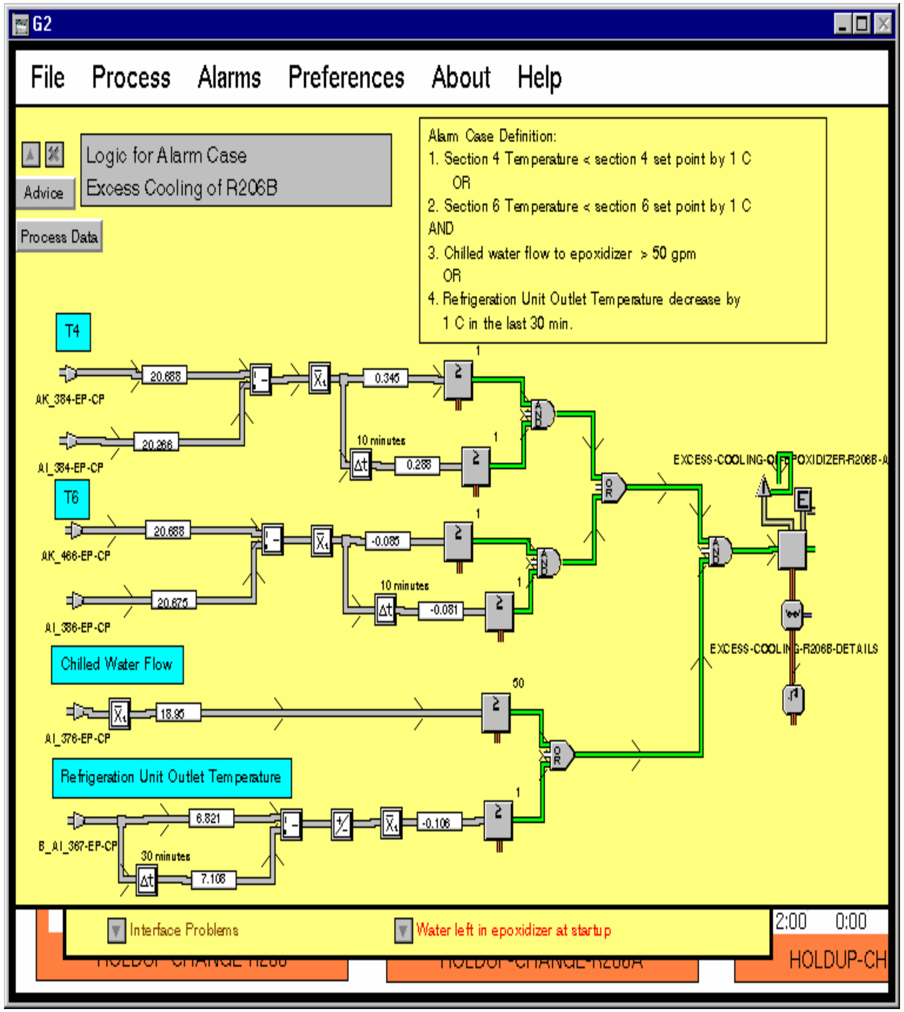


Nonlinear DOE based on GP
Minimizing expensive process experiments

EC Applications in Dow Chemical

Application Domains	Examples
Material Design	• Color Matching • Appearance Engineering • Polymer Design • Synthetic Leather
Materials Research	• Diverse Chemical Library Selection • Fundamental Model Building • Reaction Kinetics Modeling • Combi-Chem Catalyst Exploration • Combi-Chem Data Analysis
Production Design	• Acicular Mullite Emulator • EDC/VCM Nonlinear DOE • Bioreactor Optimization
Production Monitoring & Analysis	• Epoxy Holdup Monitoring • Isocyanate Level Estimation • FTIR Calibration Variable Selection • Poly-3 Volatile Emission Monitoring • Epoxy Intelligent Alarm Processing • PerTet Emulator for Online Optimization • Emissions Monitoring
Business Modeling	• Diffusion of Innovation • Hydrocarbon Trading & Energy Systems Optimization • Scheduling Heuristics • Plant Capacity Drivers

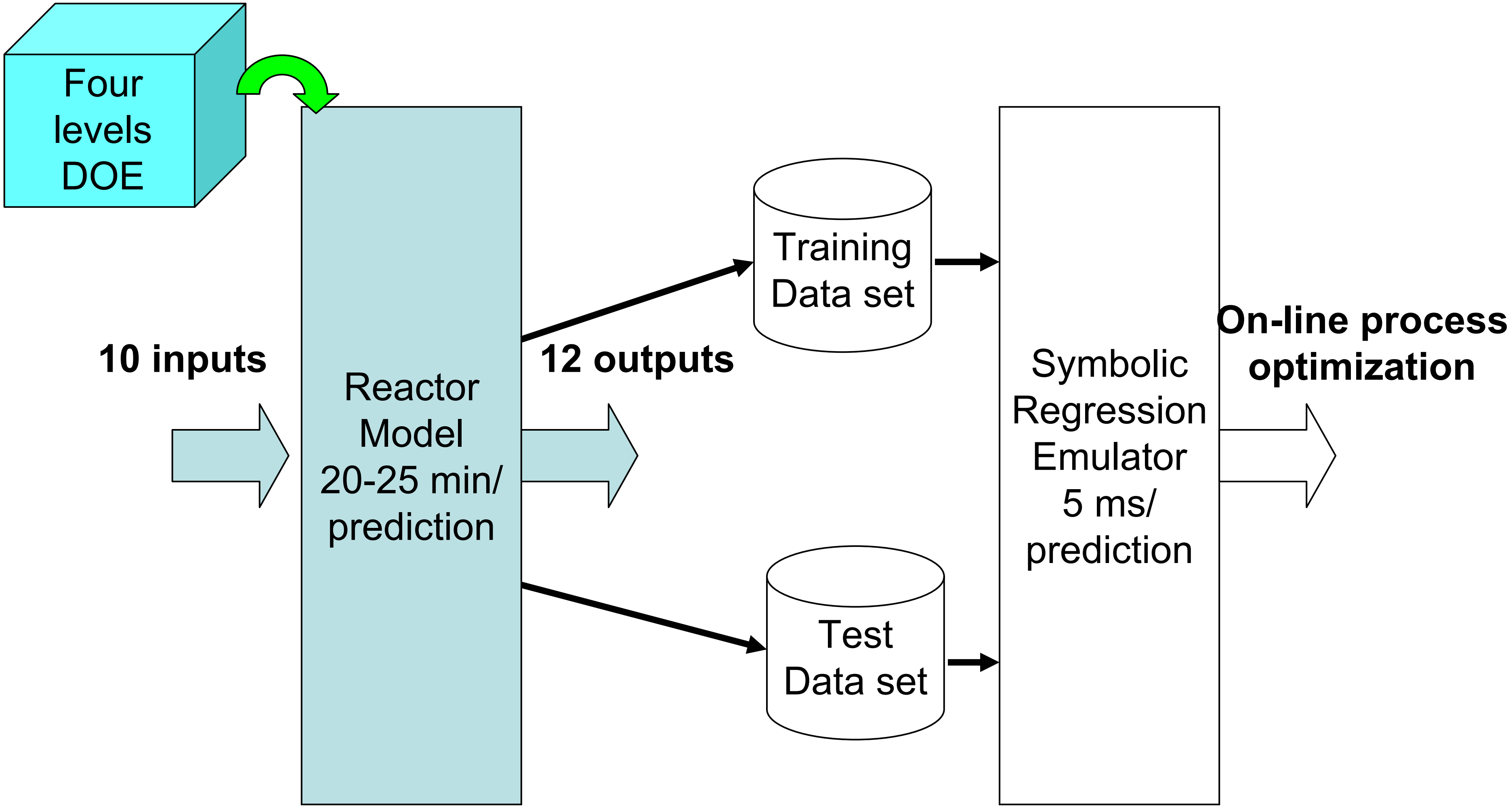
Automating Operating Discipline



- Heuristic rules defined verbally by process engineers/operators
- holdup predictor designed by stacked analytic NN and GP
- all decision blocks have fuzzy thresholds defined by membership functions
- simple empirical models and mass balances
- fundamental model predictions are used in the heuristic rules

- reduced major shutdowns
- reduced lab sampling

Emulator for optimization of an industrial chemical process

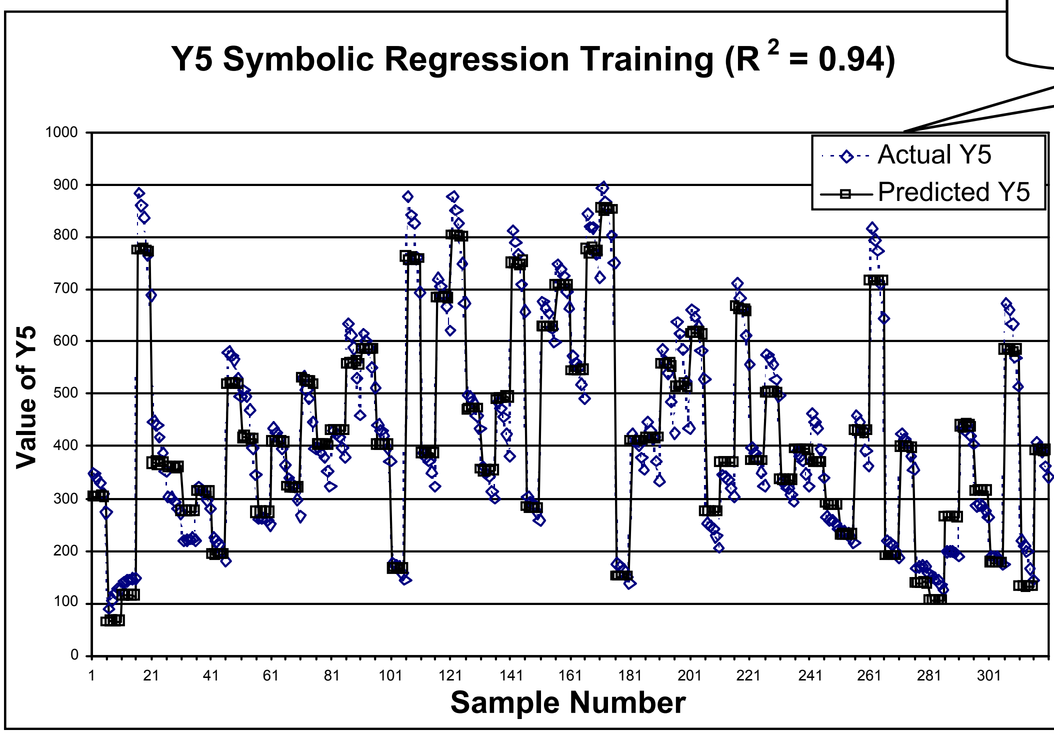


Symbolic regression-based emulator's performance

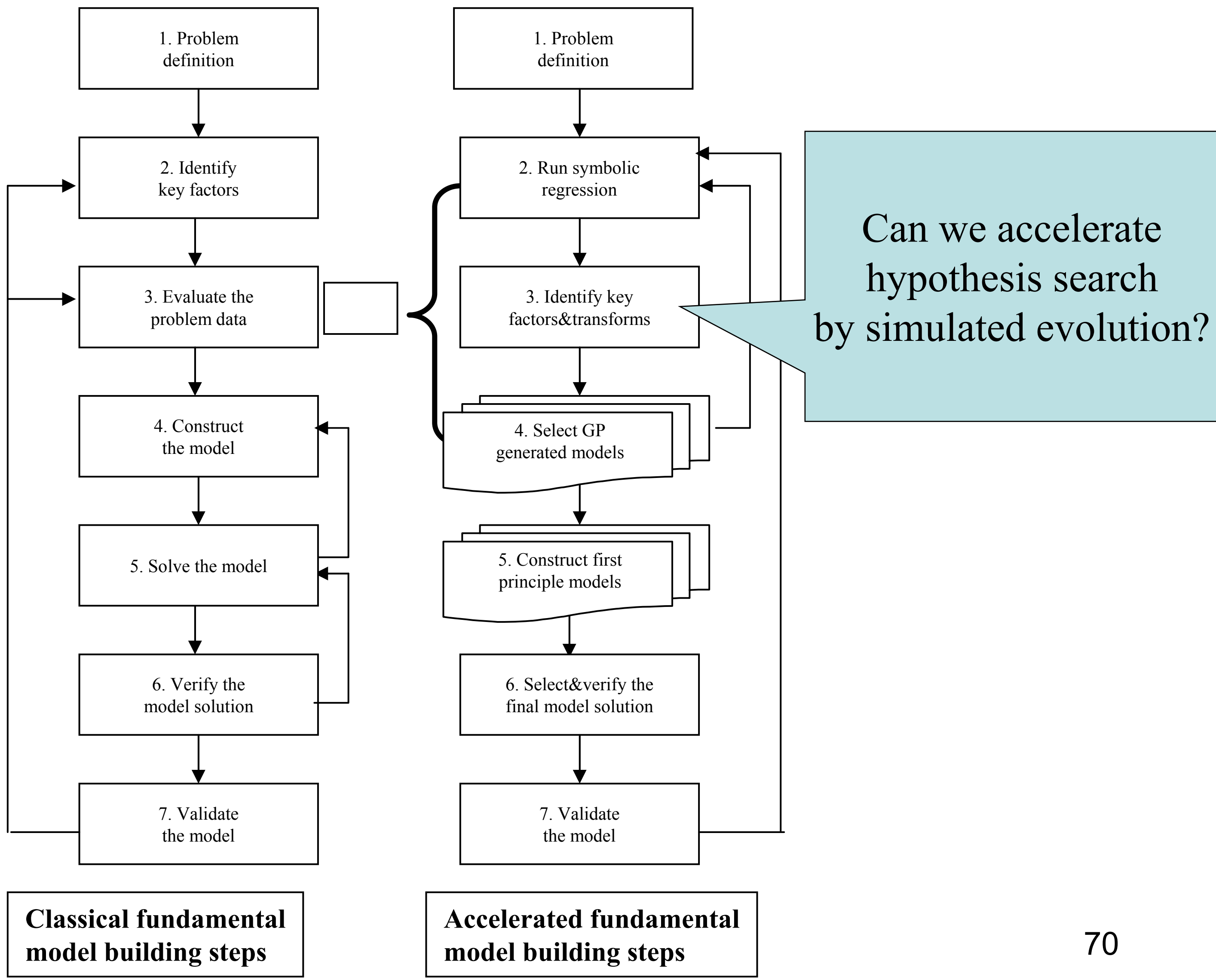
Simple expression for on-line implementation

$$Y5 = 3x_9 + \frac{6x_3 + x_4 + x_5 + 2x_6 + x_2 x_9 - 3x_{10} - \frac{x_2 - 3x_3 - x_5 + \sqrt{x_6} - 2x_7 - x_9 + x_{10}}{\mathrm{Log}[x_2]^2}}{x_2}$$

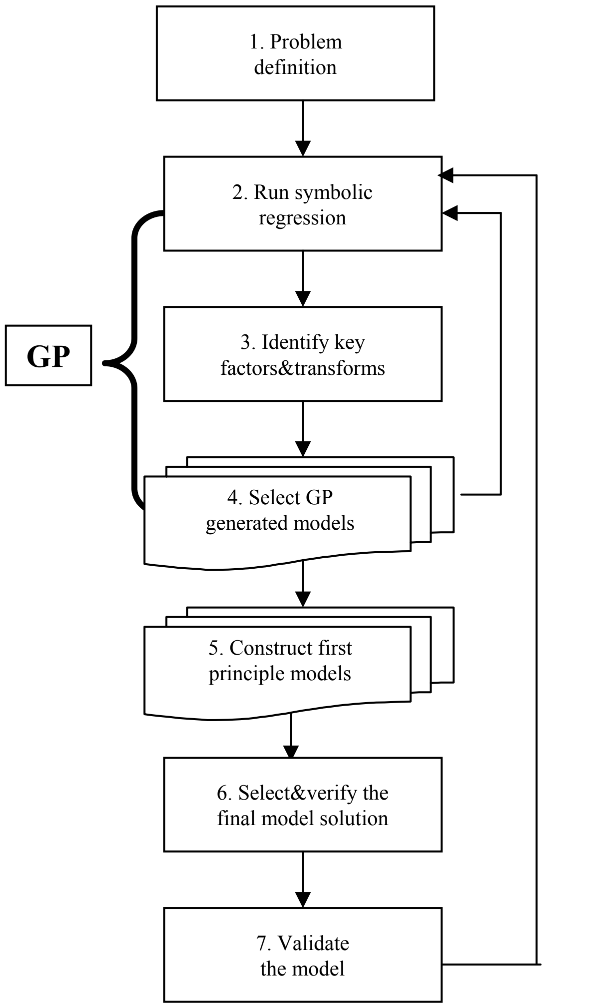
Acceptable performance



Accelerated Fundamental Model Building Based on GP

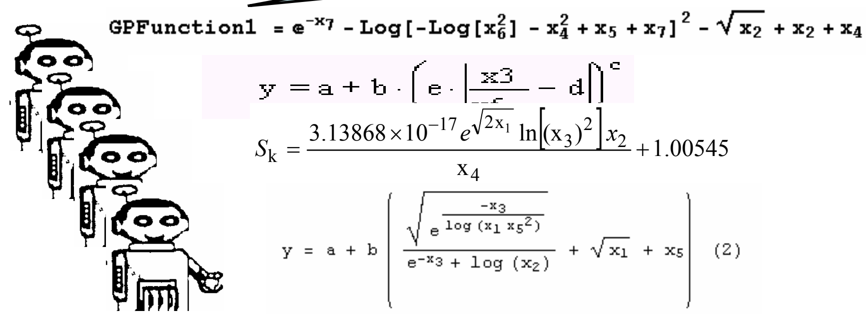


Fundamental Model Building Based on GP



Accelerated fundamental model building steps

Run simulated evolution before beginning fundamental modeling

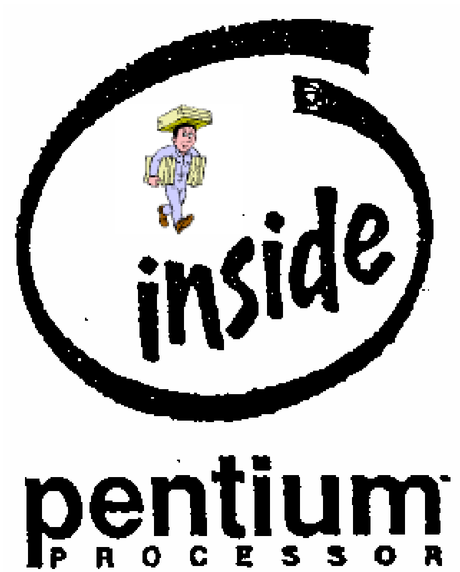


Virtual modelers

The evolutionary process identifies the key input variables as well as natural groupings & relationships. Combining this with a domain knowledge and first-principles insights is very powerful.

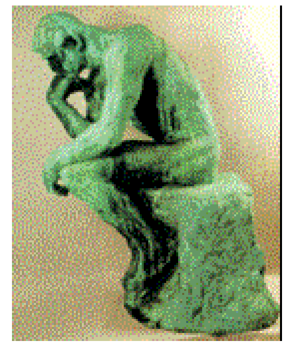
Approaches to accelerate fundamental model building process

AI approach



Mimic the expert

Reduce hypothesis search by GP



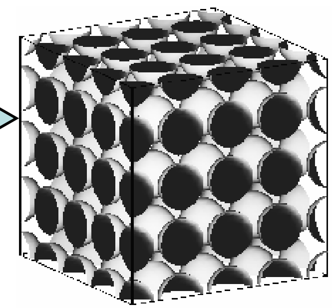
GP as automated invention machine

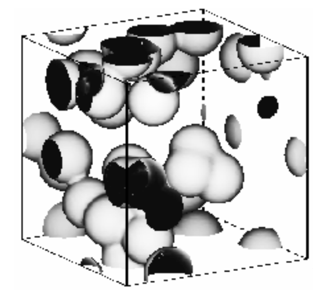
Eliminate the expert

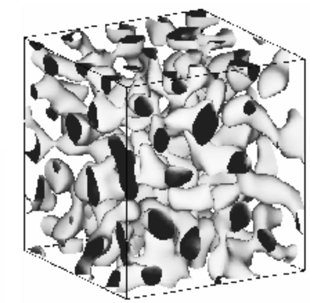
Maximize creativity of the expert

The problem of structure-properties in fundamental modeling

Material structure







Properties:
- molecular weight
- particle size
- crystallinity
- volume fraction
- material morphology
- etc.

Key modeling effort for new product development

Modeling issues:
• nonlinear interaction
• large number of preliminary expensive experiments required
• large number of possible mechanisms
• slow fundamental model building
• insufficient data for training neural nets

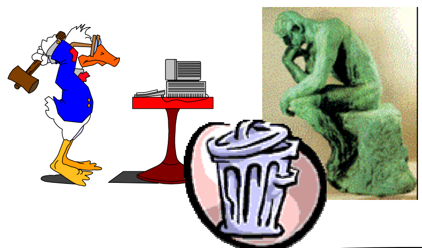
Case Study with Structure-Property Relationships

Fundamental Model Building

Theoretical Analysis

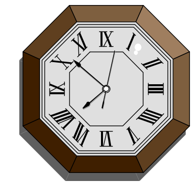
$$\frac{dT}{dt} = a\frac{\partial^2 T}{\partial z^2} - \frac{\Delta H}{C_p\,\rho}\frac{dc}{dt}$$

Hypothesis Search



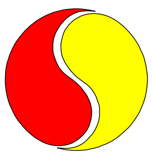
Fundamental model

$$y = a + [b\,x_1 + c\log(x_2)]\,e^{kx_3} + d\,x_5$$



3 months

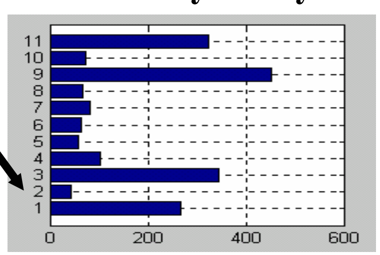
Structure-property data sets

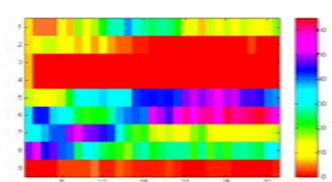


Fundamental Model Building + Symbolic Regression = Accelerated New Product Development

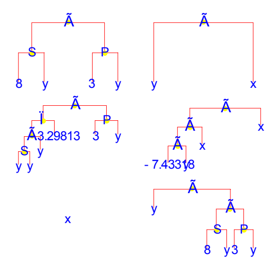
Symbolic Regression

Sensitivity Analysis



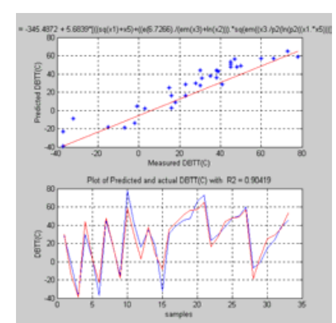


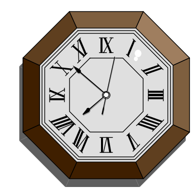
Simulated Evolution



Symbolic Regression Model

$$y = a + b\left(\sqrt{\frac{e^{\frac{-x_3}{\log(x_1 x_5^2)}}}{e^{-x_3} + \log(x_2)}} + \sqrt{x_1} + x_5\right) \quad (2)$$





10 hours

Results from hypothesis search

Key transforms

Transform 1: $\frac{X_3}{x_1}$ with R^2 of 0.74;

Transform 2: $\frac{X_3}{\sqrt{x_5}}$ with R^2 of 0.81;

Transform 3: $\frac{X_1 X_3}{\sqrt{x_5}}$ with R^2 of 0.84;

Physical interpretation found

Results from hypothesis search

Selected symbolic regression empirical model

Fundamental model

$$y = a + [b\,x_1 + c\log(x_2)]\,e^{kx_3} + d\,x_5$$

Selected empirical model

GP-generated empirical model captured correctly the functional forms of the fundamental model

$$y = a + b\left(\frac{\sqrt{e^{\frac{-x_3}{\log(x_1 x_5^2)}}}}{e^{-x_3} + \log(x_2)} + \sqrt{x_1} + x_5\right)$$

Square root form for x1

Linear form for x5

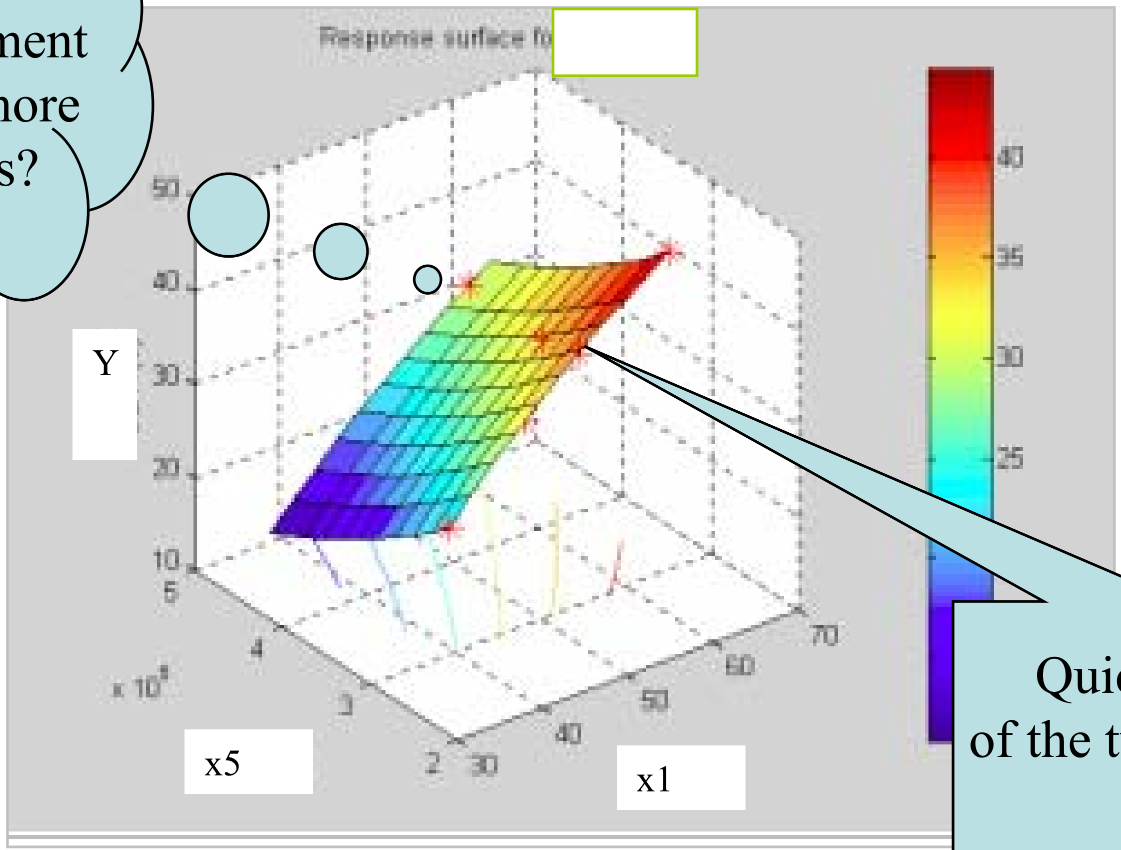
Exponential form for x3

Logarithmic form for x2

Results from hypothesis search

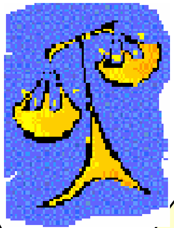
Response surface analysis of the selected symbolic regression empirical model

Quick assessment do we need more experiments?



Quick assessment of the type of nonlinear behavior

Comparative Analysis of Symbolic Regression in Fundamental Model Building



Advantages of Symbolic Regression

Model Development Speed
- 10 hours vs. 3 months

Summarize Multivariate Data
- convert data into equations to facilitate human insight
- can explore parameter sensitivity and play what-if games

Accuracy
- achieved > 90% correlation with experimental results

Identify Key Variables and Transforms
- with the exception of x1, symbolic regression captured correctly all other functional forms in the model

Suggest Physical Mechanisms
- evolved expressions and equation "building blocks" may be interpreted from a first-principles viewpoint

Suggest Future Experiments
- optima in evolved expressions may be validated in future experiments

Disadvantages of Symbolic Regression

Blind to Physics and Chemistry
- genetic programming does not currently take into account the physical or chemical laws
- expressions may have no physical meaning - mathematical consistency is how fitness is defined
- inclusion of physical constraints is a research topic

Garbage-In/Garbage Out
- appropriate variables must be supplied
- data is assumed to be accurate
- operational range should be covered

Experts (Scientists) are Still Required

domain expert is an absolute must for interpretation of evolved expressions

The domain expert delivers the final fundamental model

GP and Design Of Experiments (DOE) Models Showing Lack of Fit

Situations of Lack of Fit

1. Simple factorial DOE

Enough experiments to fit first order model

$$y = \beta_o + \sum_{i=1}^{k} \beta_i x_i + \sum_{i<j}\sum \beta_{ij} x_i x_j$$

Classical approach if LOF add experiments to fit second order model

$$S_k = \beta_o + \sum_{i=1}^{k} \beta_i x_i + \sum \beta_{ii} x_i^2 + \sum_{i<j}\sum \beta_{ij} x_i x_j$$

More costly experiments

2. A response surface DOE

already had all experiments to fit second order model

$$S_k = \beta_o + \sum_{i=1}^{k} \beta_i x_i + \sum \beta_{ii} x_i^2 + \sum_{i<j}\sum \beta_{ij} x_i x_j$$

Classical approach if LOF no alternative (use model as it is)

Suggested approach: Use GP to transform inputs

1. Generate GP models

$$S_k = \frac{3.13868\times 10^{-17} e^{\sqrt{2x_1}} \ln\left[(x_3)^2\right] x_2}{x_4} + 1.00545 \quad (2)$$

Selected solution

2. Generate input transforms

Variable transformations suggested by GP model

Original Variable	Transformed Variable
x_1	$Z_1 = \exp\left(\sqrt{2x_1}\right)$
x_2	$Z_2 = x_2$
x_3	$Z_3 = \ln[(x_3)^2]$
x_4	$Z_4 = x_4^{-1}$

3. Fit response surface model in transformed variables

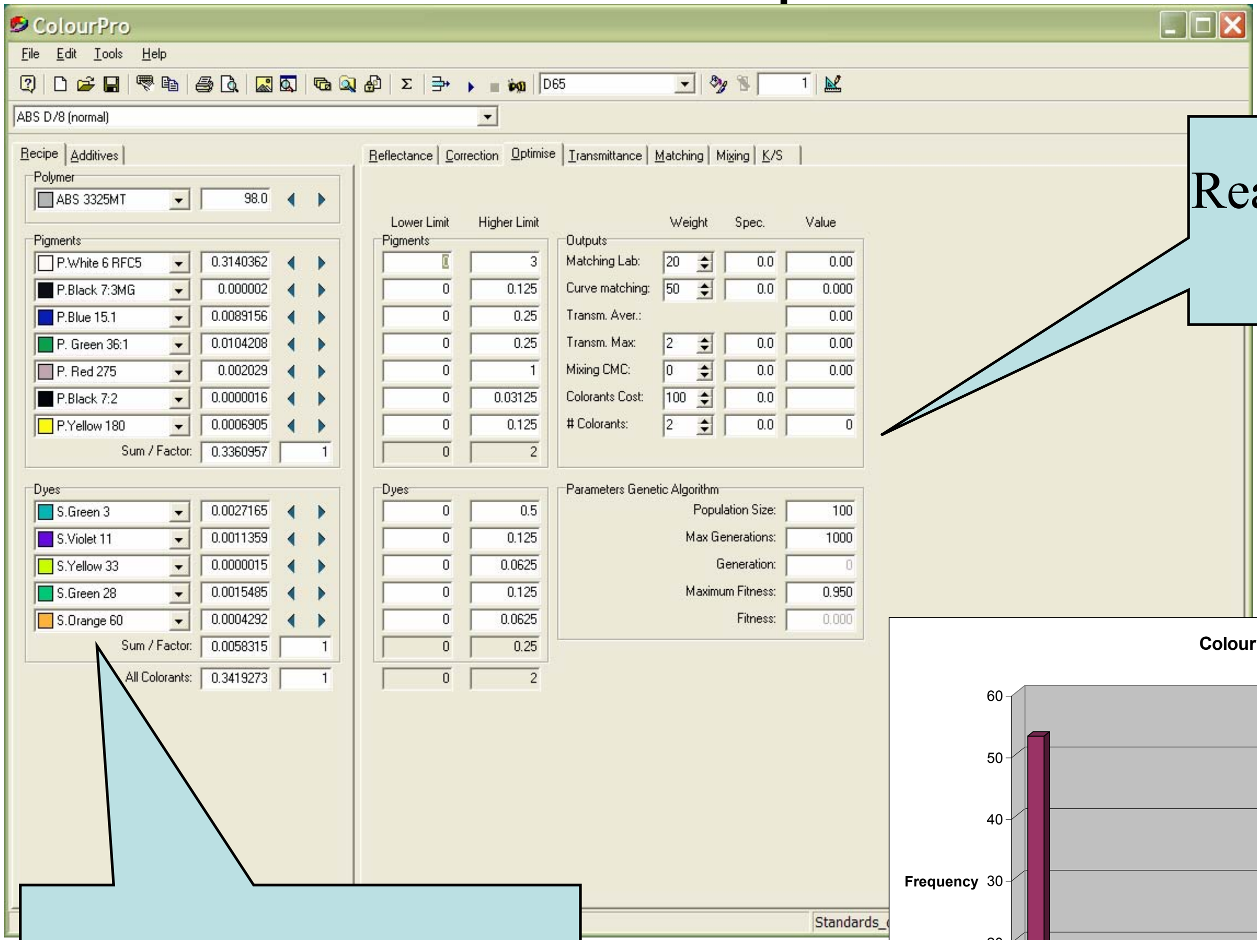
$$S_k = \beta_o + \sum_{i=1}^{4} \beta_i Z_i + \sum\sum_{i<j} \beta_{ij} Z_i Z_j + \sum_{i=1}^{4} \beta_{ii} Z_i^2$$

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	2	0.00049190	0.000246	2.2554
Pure Error	2	0.00021810	0.000109	**Prob > F**
Total Error	4	0.00071000		0.3072
				Max RSq
				0.9999

Note that Lack Of Fit is not significant (p=0.3072)

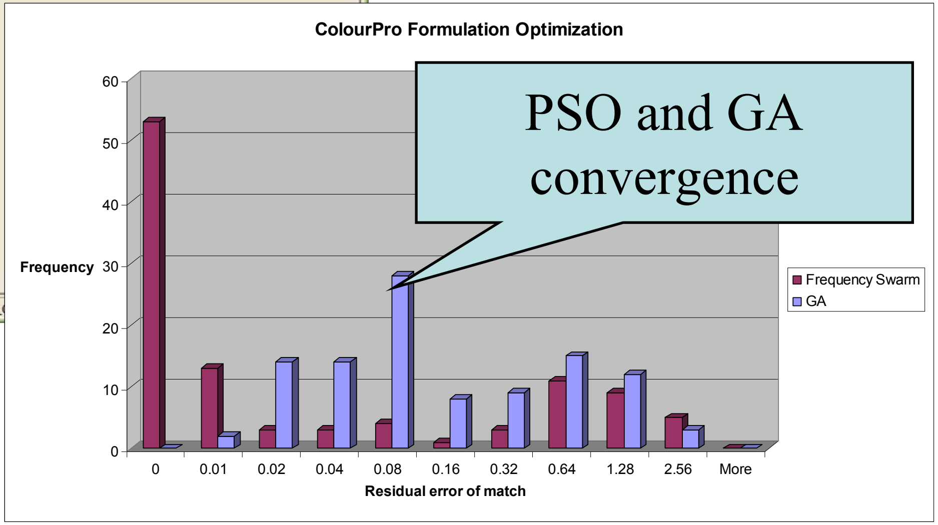
No Lack Of Fit
(p=0.3037)

PSO application: Optimizing color spectrum of plastics



Real-time optimization in 2-3 seconds

Multiple-objective PSO with 15 variables

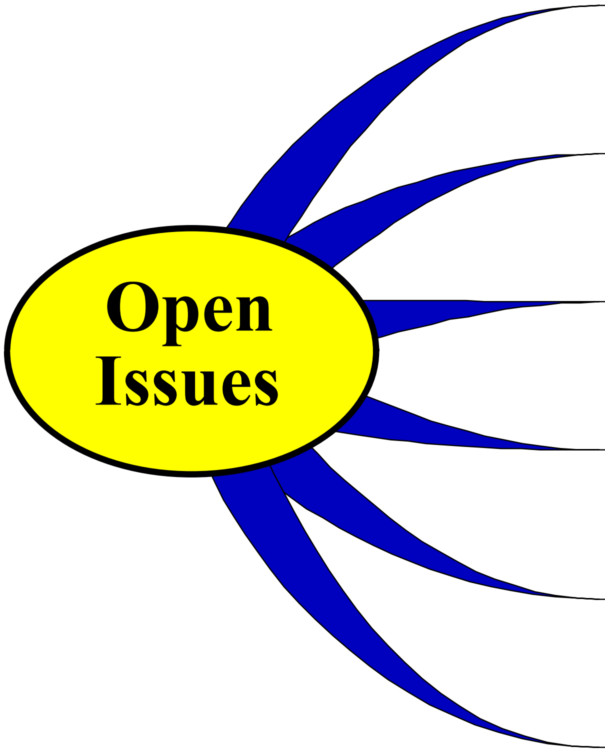


PSO and GA convergence

Other PSO applications

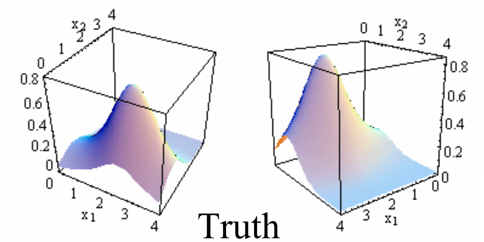
- Drug release predictor
  - 6 parameters
  - population size = 30
  - optimization time: ~ 30 seconds
- Foam acoustics performance predictor
  - 8 parameters
  - population size = 50
  - optimization time: ~ 5 seconds
- Crystallization kinetics predictor
  - 4 parameters
  - population size = 30
  - optimization time: ~ 2 seconds

Open Issues



- **Complexity Control & Smoothness Characterization**
- **Diversity Detection**
- **Identifying Metavariables**
- **Convert Hard GP Problems into Easy Ones**
- **Should we be doing Cultural Programming?**
- **Blending Heuristics & Prior Knowledge**

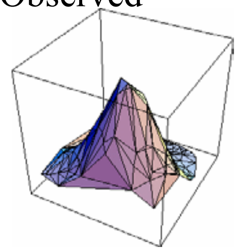
Complexity Control & Smoothness Characterization

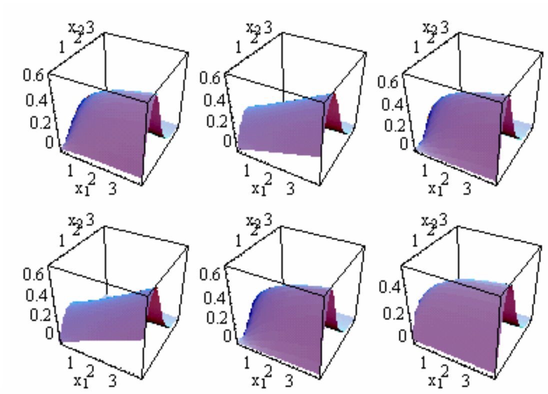


Truth

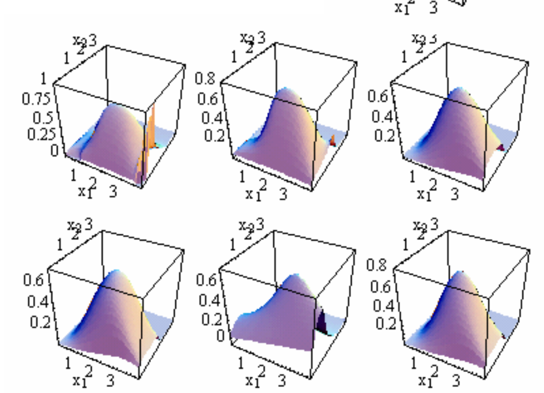
$$\frac{e^{-(-1+b)^2}}{1.2+(-2.5+a)^2}$$

Observed





Early Results



Later Results

1. 0.969 0.975 x1 x2

$$\frac{x_1}{\left|\left|\frac{x_1 \, |x_1|^{\frac{(x_1-3.665)\,x_1}{x_2}} - 2.718^{x_1} + 7}{x_1^2}\right|^{x_1} + \left|\left|x_2\right|^{x_2}\right|^{2} + x_2 + \frac{x_1}{x_2} + 1\right|}$$

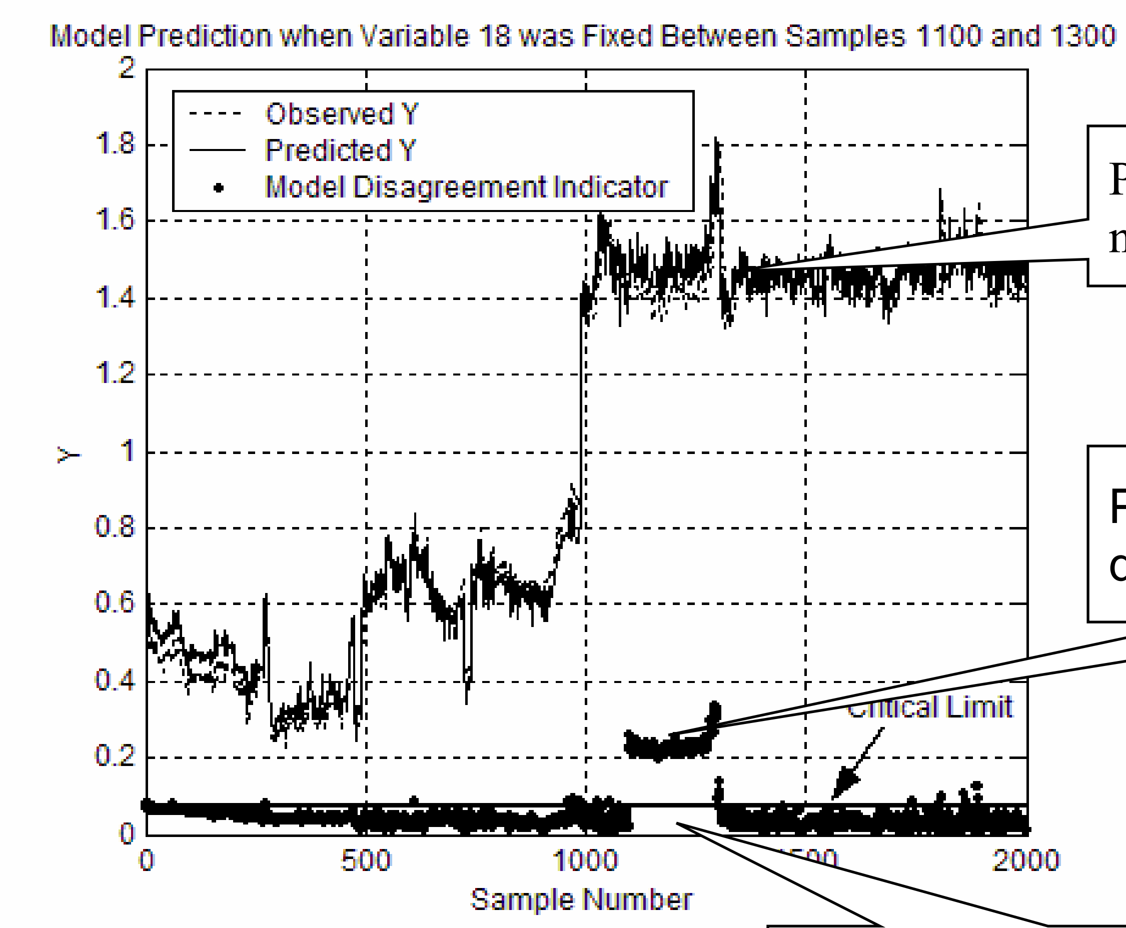
- What is going on between the data points?
- How do we identify and eliminate pathologies?
- How do we recognize and characterize the overall and local smoothness (complexity)?

Cultural Programming?

- Particle swarm optimization has replaced most of GA for our applications
- Does the cultural algorithm metaphor have a similar potential with GP?
  - Coevolution of symbiotic species?
  - Sociological niches?
  - Population size dynamics?
  - Cascaded Evolutionary Programming?

Business potential for ensemble-based predictors

Robustness toward input measurement faults



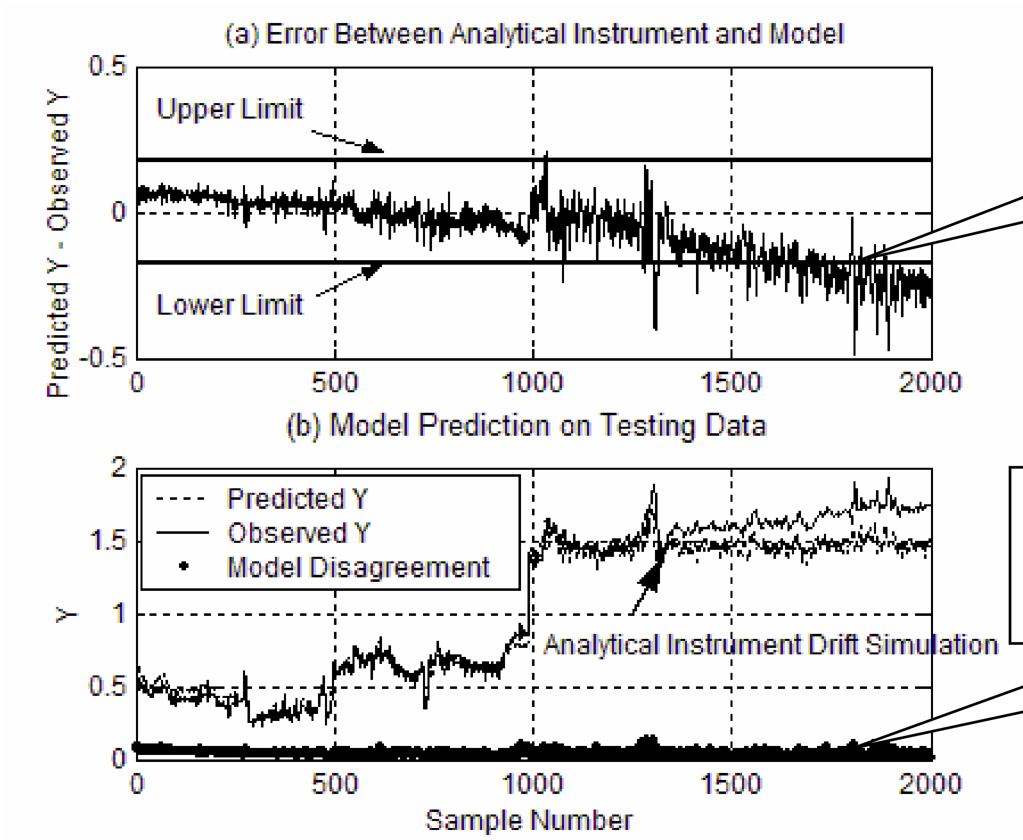
Prediction still available by models not based on fault measurements

Problem detected by model disagreement indicator

Measurement fault between Samples 1100 and 1300

Business potential for ensemble-based predictors

Analytical instrument drift detection



Growing error between analytical instrument and model prediction

Probable root cause: analytical instrument drift

Model disagreement indicator is within critical limit

Summary

- Evolutionary Computing can create significant value to industry by reducing model development time and model exploitation cost
- Integrating EC with Neural Networks, Support Vector Machines, and Statistics is recommended for successful industrial applications
- This strategy works for many real applications in the chemical industry
- The key application areas are:
  - Inferential sensors
  - Improved process monitoring and control
  - Accelerated new product development
  - Effective design of experiments
- And this is only the beginning …

Acknowledgement

We would like to acknowledge the contribution of the following researchers from The Dow Chemical Company:

Alex Kalos

Kip Mercure

Flor Castillo

Elsa Jordaan

Leo Chiang

Irina Graaf

References

1. M. Kotanchek, G. Smits, and A. Kordon, *Industrial Strength Genetic Programming*, In *GP Theory and Practice* (R. Riolo and B. Worzel-Eds), Kluwer, 2003.
2. A. Kordon, G. Smits, A. Kalos, and E. Jordaan, *Robust Soft Sensor Development Using Genetic Programming*, In *Nature-Inspired Methods in Chemometrics*, (R. Leardi-Editor), Elsevier, 2003.
3. Kordon A.K, G.F. Smits, E. Jordaan and E. Rightor, *Robust Soft Sensors Based on Integration of Genetic Programming, Analytical Neural Networks, and Support Vector Machines*, Proceedings of WCCI 2002, Honolulu, pp. 896 – 901, 2002.
4. Kotanchek M., A. Kordon, G. Smits, F. Castillo, R. Pell, M.B. Seasholtz, L. Chiang, P. Margl, P.K. Mercure, A. Kalos, *Evolutionary Computing in Dow Chemical*, Proceedings of GECCO'2002, New York, volume Evolutionary Computation in Industry, pp. 101-110., 2002
5. Kordon A. K., H.T. Pham, C.P. Bosnyak, M.E. Kotanchek, and G. F. Smits, *Accelerating Industrial Fundamental Model Building with Symbolic Regression: A Case Study with Structure – Property Relationships*, Proceedings of GECCO'2002, New York, volume Evolutionary Computation in Industry, pp. 111-116, 2002
6. Castillo F., K. Marshall, J. Greens, and A. Kordon, *Symbolic Regression in Design of Experiments: A Case Study with Linearizing Transformations*, Proceedings of GECCO'2002, New York, pp. 1043-1048.
7. Kordon A., E. Jordaan, L. Chew, G. Smits, T. Bruck, K. Haney, and A. Jenings, Biomass Inferential Sensor Based on Ensemble of Models Generated by Genetic Programming, accepted for GECCO 2004, 2004.
8. Smits G. and M. Kotanchek, *Pareto-Front Exploitation in Symbolic Regression*, In *GP Theory and Practice* (R. Riolo and B. Worzel-Eds), Kluwer, 2004.
9. Kordon A., A. Kalos, and B. Adams, *Empirical Emulators for Process Monitoring and Optimization*, Proceedings of the IEEE 11th Conference on Control and Automation MED'2003, Rhodes, Greece, pp.111, 2003.