Soft Computing in Fault Detection and Isolation

PART III

Evolutionary algorithms in fault diagnosis

Marcin Witczak

8 października 2012

OUTLINE

- ➥ Fundamental definitions, concepts and history of evolutionary algorithms
- ➥ Computational framework of selected algorithms
- ➥ Evolutionary algorithms in control engineering – a brief review
- ➥ Evolutionary algorithms in fault diagnosis:
  - Model design
  - Design of robust observers

☛ FUNDAMENTAL DEFINITIONS, CONCEPTS AND HISTORY EVOLUTIONARY ALGORITHMS

➜ Fundamental definitions

Evolutionary Algorithms (EAs): a broad class of stochastic optimisation algorithms inspired by some biological processes, which allow populations of organisms to adapt to their surrounding environment

Population: a set of individuals being potential solutions of the problem under consideration

Representation of the individual:

- **Genotype**: a genetic code of an individual or a search point in the so-called *genotype space*
- **Phenotype**: the manner of response contained in the behaviour, physiology and morphology of an individual

Fitness function: a measure of the fitness of an individual to a given environment, calculated based on phenotype

➜ Main types of evolutionary algorithms

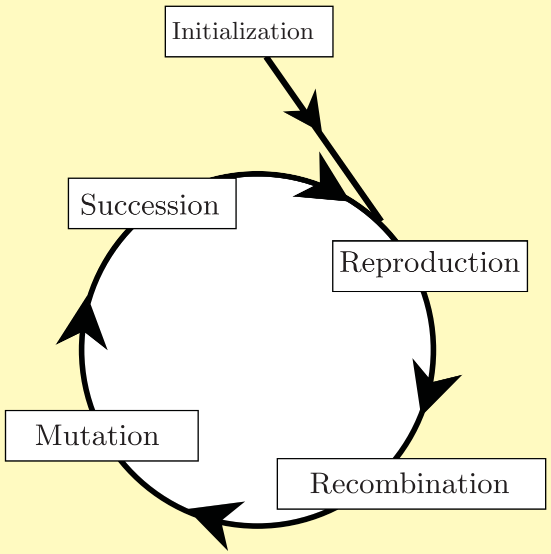
Search in a genotype space:

- Genetic Algorithms (GAs): Holland (1975)
- Genetic Programming (GP): Koza (1992)

Search in a phenotype space:

- evolutionary programming: Fogel (1999)
- evolutionary strategies: Michalewicz (1996)
- evolutionary search with soft selection: Galar (1989)

➜ A general framework



Reproduction (preselection): a randomised process (deterministic in some algorithms) of parent selection from the entire population, i.e. a temporary population of parent individuals is formed

Recombination: allows mixing parental information while passing it onto the descendants

Mutation: introduces an innovation into the current descendants

Succession: applied to choose a new generation of individuals from parents and descendants, based on the fitness of each individual

➜ Brief history of evolutionary algorithms

1950s: Idea of using simulated evolution to solve engineering problems:

- Box (1957)
- Friedberg (1958)
- Bremermann (1962)

1960s: Fundamental works in evolutionary computation:

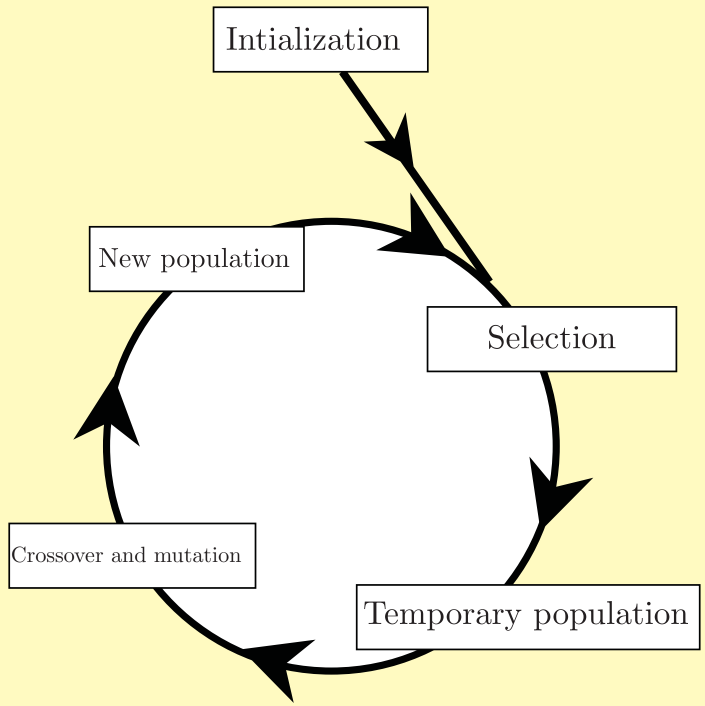
- evolutionary programming: Fogel (1962)
- genetic algorithms: Holland (1962)
- evolutionary strategies: Rechenberg (1962) & Schwefel (1968)

1990s: Genetic programming: Koza (1992)

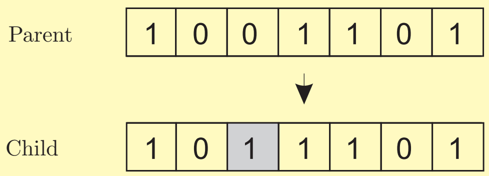
1970-today: Development of evolutionary computation: see Back *et al.* (1997): IEEE Trans. Evol. Comput. for a survey and comprehensive discussion

COMPUTATIONAL FRAMEWORK OF SELECTED ALGORITHMS

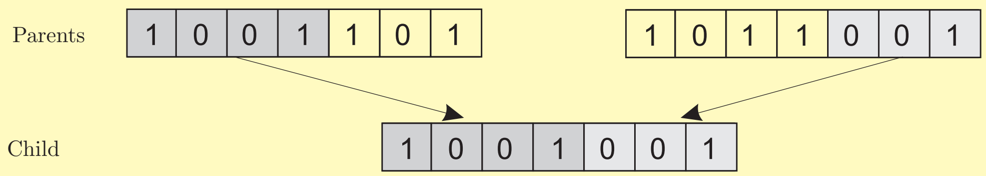
➜ Genetic algorithms



Mutation



Crossover



➜ Exemplary application of genetic algorithms – non-linear parameter estimation

Available information:

- set of input-output measurements $\{(\boldsymbol{u}_k, y_k)\}_{k=1}^{n_t}$
- model structure:

$$y_{m,k} = f(\boldsymbol{p}, \boldsymbol{u}_k)$$

e.g. $y_{m,k} = p_1 \exp(p_2 u_k)$

- fitness function:

$$J = -\sum_{k=1}^{n_t} (y_k - f(\boldsymbol{p}, \boldsymbol{u}_k))^2$$

Binary representation of $\boldsymbol{p} \in \mathbb{R}^{n_p}$:

$$\boldsymbol{p} \quad \Longrightarrow \quad \begin{array}{|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ \hline \end{array}$$

➜ Exemplary application of genetic algorithms – non-linear parameter estimation

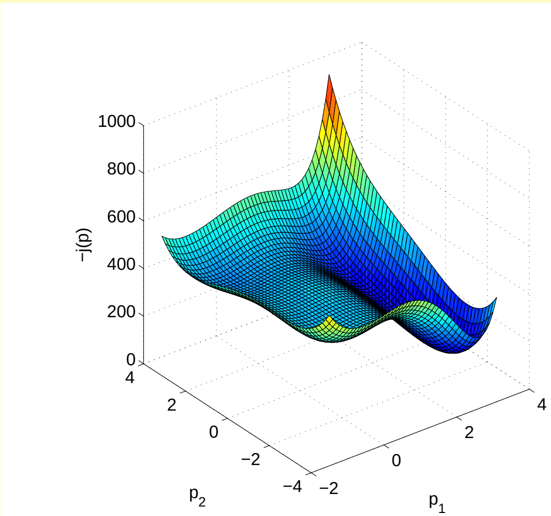
Available information

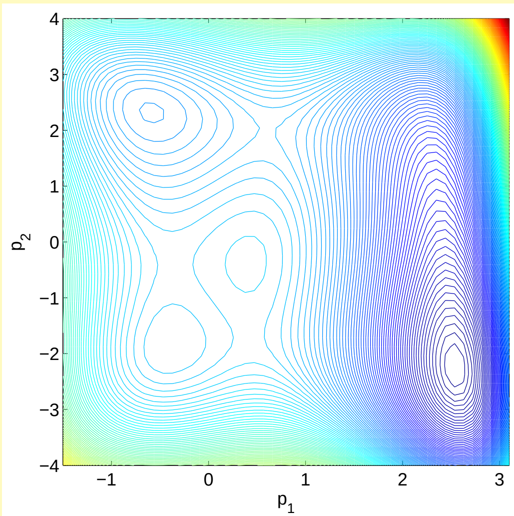
- set of input-output measurements:
  $\{(\boldsymbol{u}_k, y_k)\}_{k=1}^{n_t} = \{([1,0],5),([1,1],-12),([0,1],15)\}$
- model structure:

$$y_{m,k} = f(\boldsymbol{p}, \boldsymbol{u}_k) = p_1 u_{1,k} + p_2 u_{2,k} + p_1^3(1-u_{1,k}) + p_2^2(1-u_{2,k})$$

- fitness function:

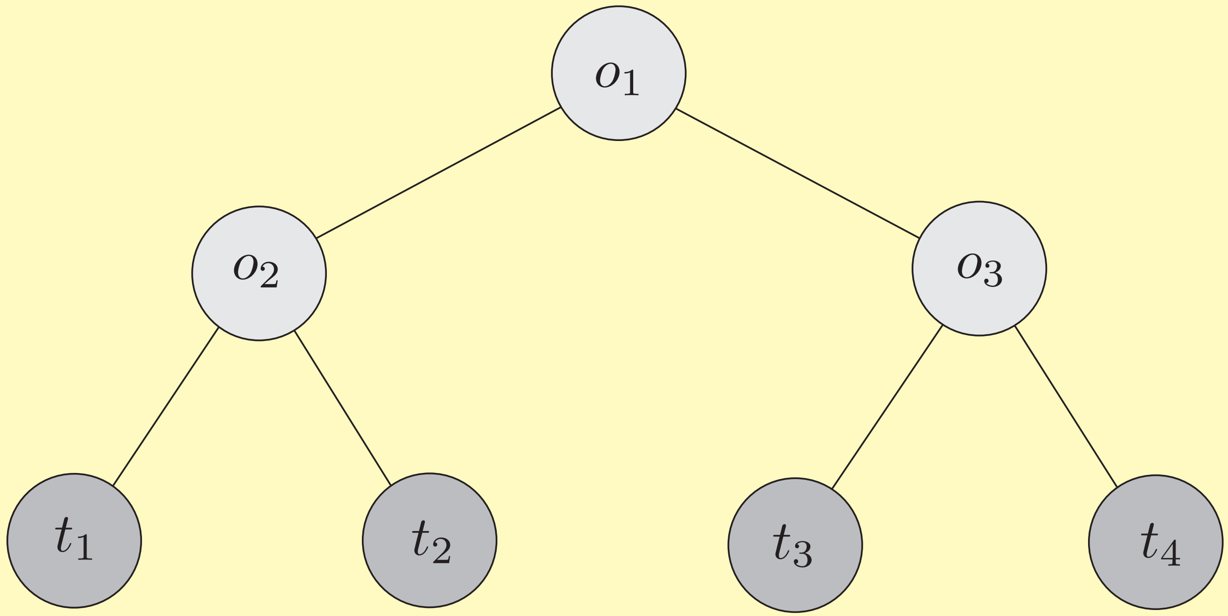
$$J = -\sum_{k=1}^{n_t} (y_k - f(\boldsymbol{p}, \boldsymbol{u}_k))^2$$





➜ Genetic programming

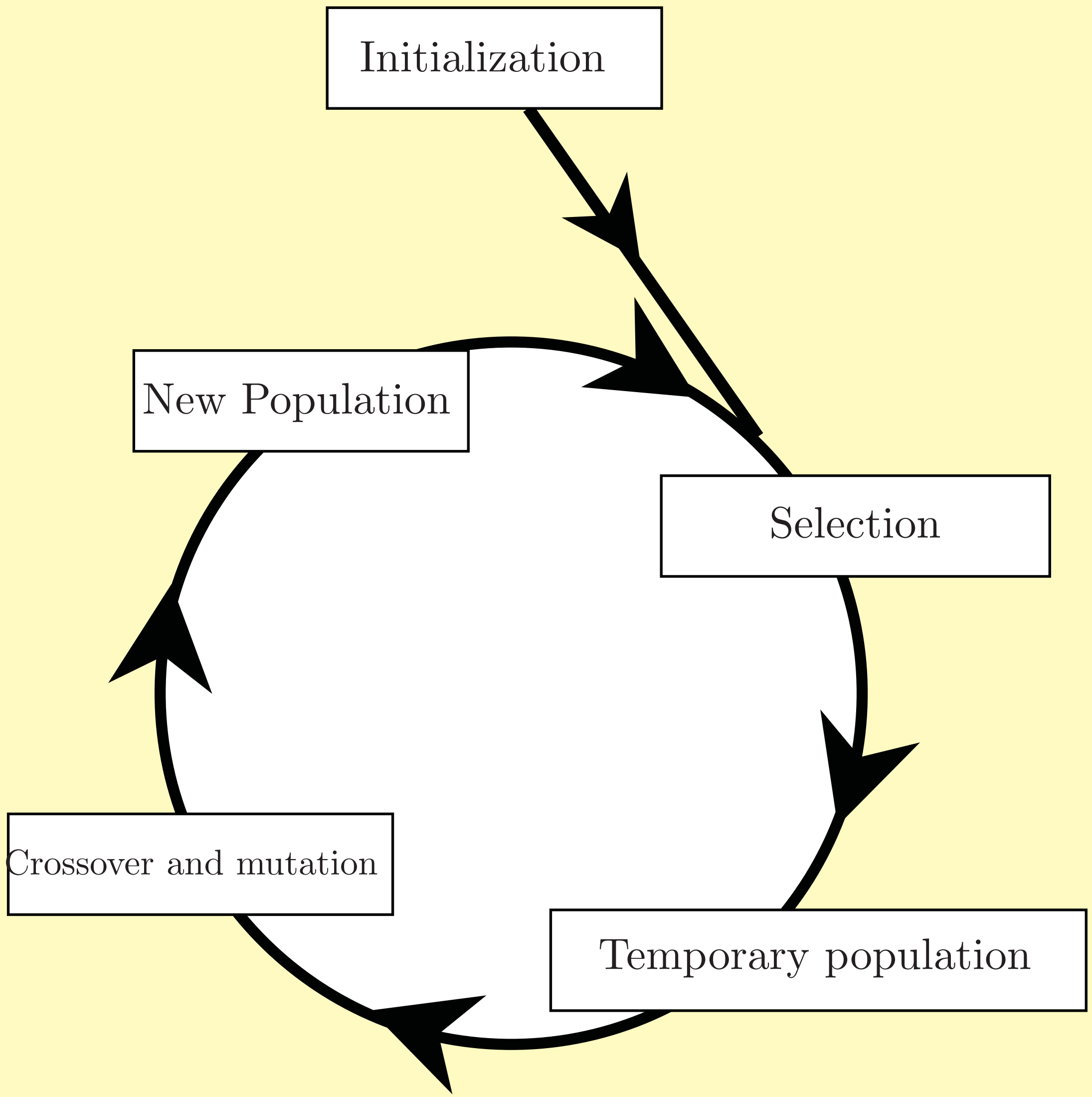
- Tree-based representation of the individual model:



- Terminal and function sets:

$$\mathbb{T} = \{t_i \mid i = 1, \ldots, n_t\} \quad \mathbb{F} = \{o_i \mid i = 1, \ldots, n_o\}$$

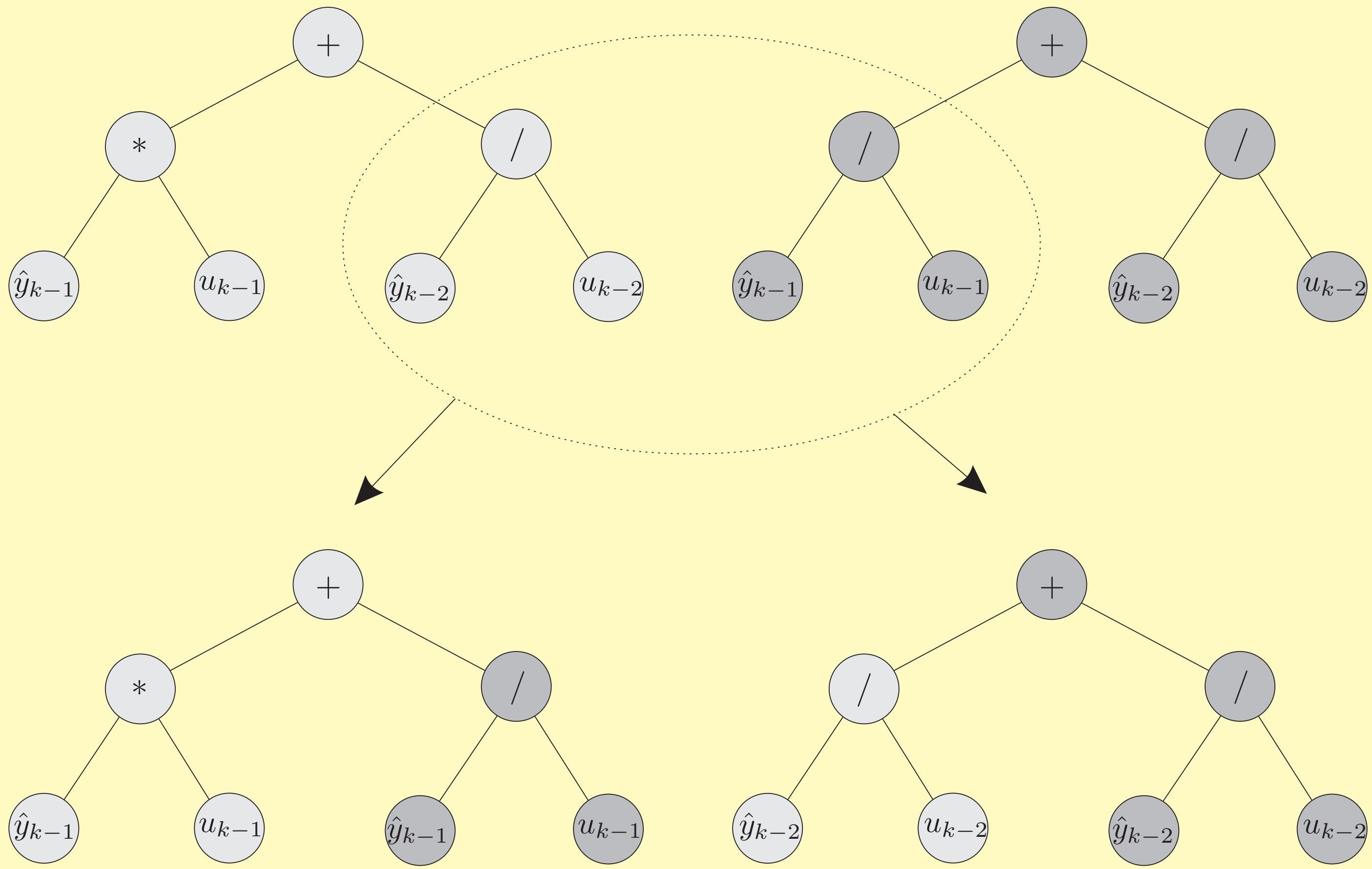
➜ The GP algorithm



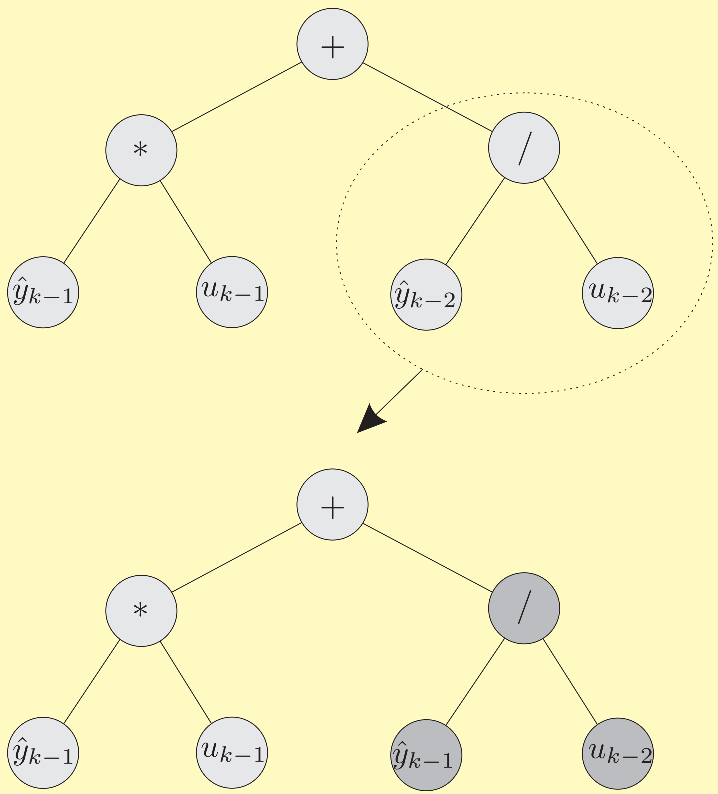
➜ Crossover

Parents

Offspring



➜ Mutation



➜ Exemplary application of genetic programming – system identification

Available information:

- set of input-output measurements $\{(\boldsymbol{u}_k, y_k)\}_{k=1}^{n_t}$
- fitness function

$$J = -\sum_{k=1}^{n_t} (y_k - f(\boldsymbol{p}, \boldsymbol{u}_k))^2 + \text{penalty term dependent on } n_p$$

Determine the structure $f(\cdot)$ and parameter vector $\boldsymbol{p} \in \mathbb{R}^{n_p}$ of

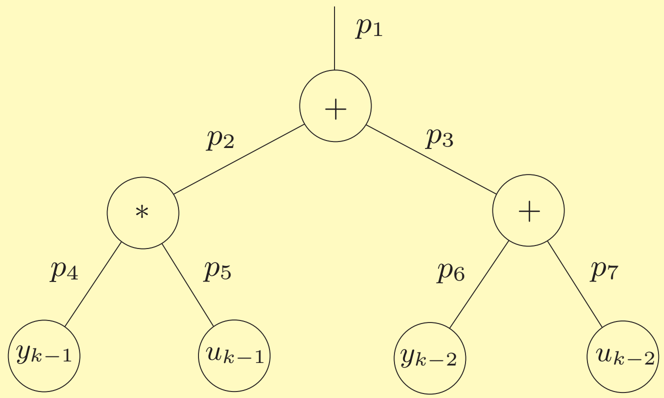
$$y_{m,k} = f(\boldsymbol{p}, \boldsymbol{u}_k)$$

- Problems with parameters

$$\mathbb{T} = \{y_{k-1}, y_{k-2}, u_{k-1}, u_{k-2}, 1\}, \quad \mathbb{F} = \{+, *, -, /\}$$

$$y_k = 3.14 y_{k-1} u_{k-1} + y_{k-2} + u_{k-2}$$

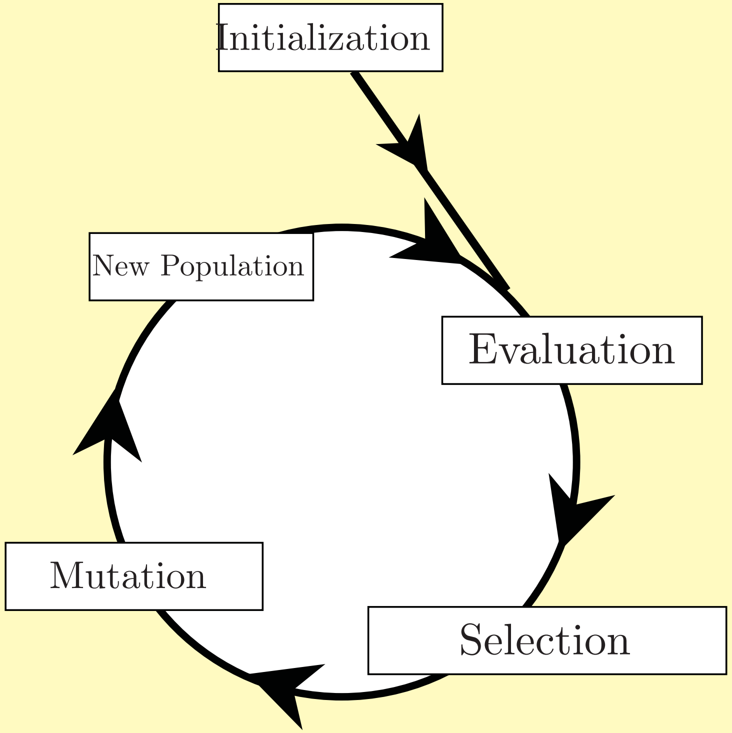
- Parameterized tree



$$y_k = p_1 p_2 p_4 p_5 y_{k-1} u_{k-1} + p_1 p_3 p_6 y_{k-2} + p_1 p_3 p_7 u_{k-2}$$

- Parameter reduction rules

➜ Evolutionary search with soft selection



Main properties:

- phenotypic representation of an individual
- exemplary mutation $x^{t+1} = x^t + \mathcal{N}(0, \sigma)$

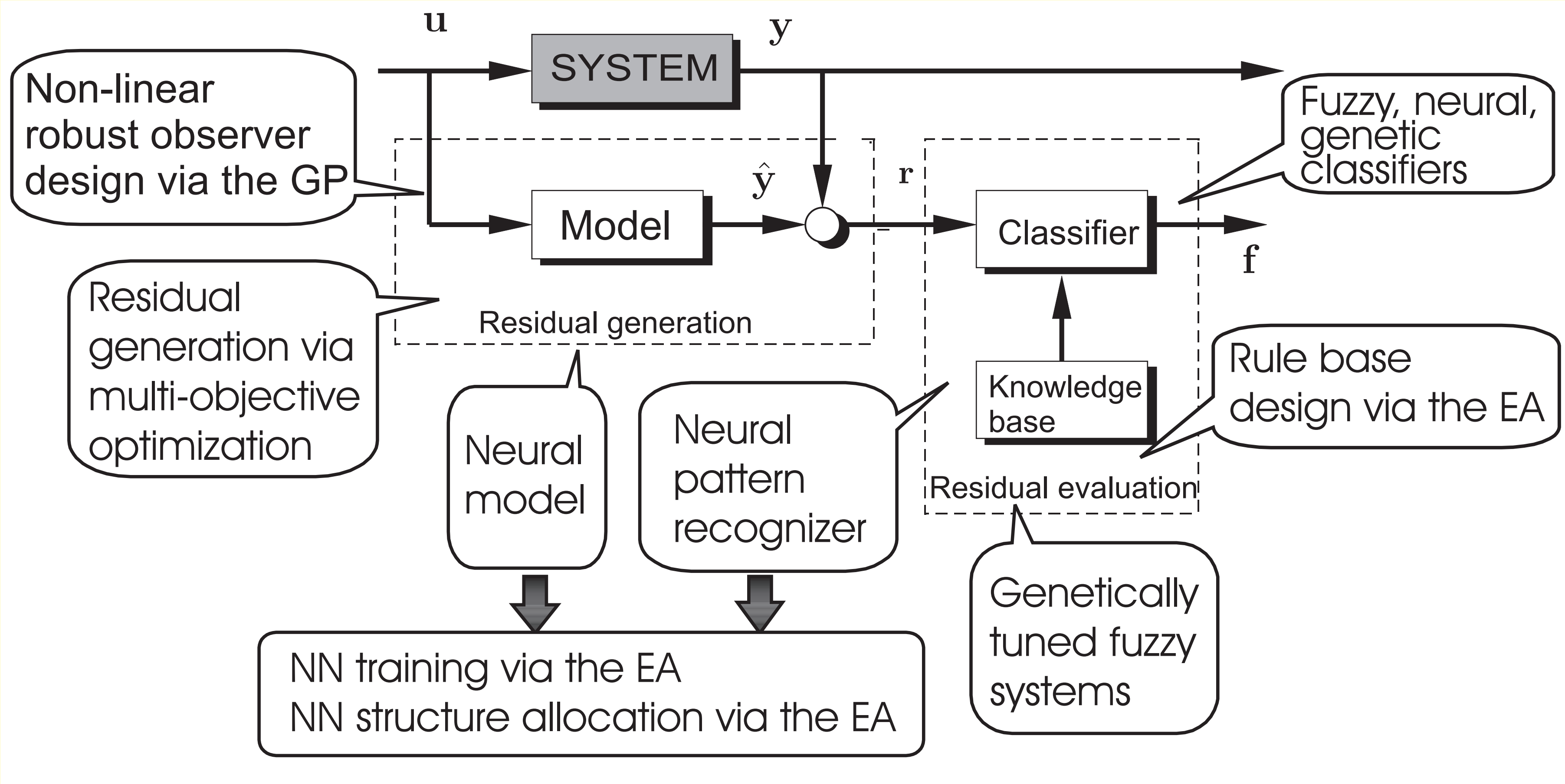
- EVOLUTIONARY ALGORITHMS IN CONTROL ENGINEERING
  - Controller design
    - Parameter setting of PID: Oliveira *et al* (1991): Eng. Syst. with Intelligence. Concepts, Tools and Applications
    - Design of an LQG controller: Mei and Goodal (2000): IEE Proceedings – Control Theory and Applications, Vol. 147 No. 1
    - Design of a robust LQG controller (w Monte Carlo method): Marrison and Stengel (1997): IEEE Trans. Automat Control, Vol. 42, No. 6
    - Design of an optimal control sequence in model-based predictive control: Onnen *et al.* (1997): Control Eng. Practice Vol. 5, No. 10
    - Controller structure and parameter design: Koza *et al.* (2000): Genetic Programming and Evolvable Machines

- Controller structure and parameter design: Chipperfield and Fleming (1996): IEEE Trans. Industrial Electronics, Vol. 43, No. 5
- Parameter determination of neuro-fuzzy controllers: Linkens and Nyongensa (1996): IEE Proc. Control Theory and Applications, Vol. 143, No. 4; Sette *et al.*(1998): Vol. 6, No. 4
- Adaptive control with a population of controllers: Lennon and Passino (1998): Eng. App. Artif. Intelligence, Vol. 12 pp. 185–200
- Iterative Learning Control: Hatzikos *et al.* (2004): Int. J. Control, Vol. 77, No. 2

- Observer design
  - Design of robust observers for linear systems: Kowalczuk and Białaszewski (2004): In Korbicz *et al.*: Fault Diagnosis, Models, Artificial Intelligence, Applications; Chen and Patton (1999): Robust Model-Based Fault Diagnosis for Dynamic Systems
  - Design of adaptive observers for non-linear systems: Moyne *et al.*(1994): Eng. App. Artif. Intell, Vol. 8, No. 3
  - Design of an extended unknown input observer for non-linear systems: Witczak, Obuchowicz and Korbicz (2002): Int. J. Control, Vol. 75, No. 13; Witczak and Korbicz (2004): In Korbicz *et al.*: Fault Diagnosis, Models, Artificial Intelligence, Applications

- Modelling and identification
  - Structure and parameter determination of a neural network: Korbicz *et al.* (2004): Fault Diagnosis, Models, Artificial Intelligence, Applications
  - Experimental design for neural networks: Witczak and Prętki (2005): Computer Assisted Mechanics and Eng. Sciences
  - Model structure and parameter determination: Witczak, Obuchowicz and Korbicz (2002): Int. J. Control, Vol. 75, No. 13; Witczak and Korbicz (2004): In Korbicz *et al.*: Fault Diagnosis, Models, Artificial Intelligence, Applications; Metenidis, Witczak and Korbicz (2004): Eng. App. Artif. Intell, Vol. 8, No. 3
  - Searching for a minimal model structure for non-linear systems: Mao and Billings (1997): Int. J. Contr., Vol. 68, No. 2

☛ EVOLUTIONARY ALGORITHMS IN FAULT DIAGNOSIS



- Fault diagnosis
  - Robust observers designed with EAs
  - Model design for fault diagnosis
  - Classifier design: Chen *et al.* (2003): Eng. App. Artif. Intell, Vol. 16, pp. 31-38;
  - Design of expert and fuzzy systems: Koza (1992): Genetic Programming

➜ Genetic programming in model design for FDI

State-space description of the system:

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}(\hat{\boldsymbol{x}}_k)\hat{\boldsymbol{x}}_k + \boldsymbol{h}(\boldsymbol{u}_k)$$
$$\hat{\boldsymbol{y}}_{k+1} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1}$$

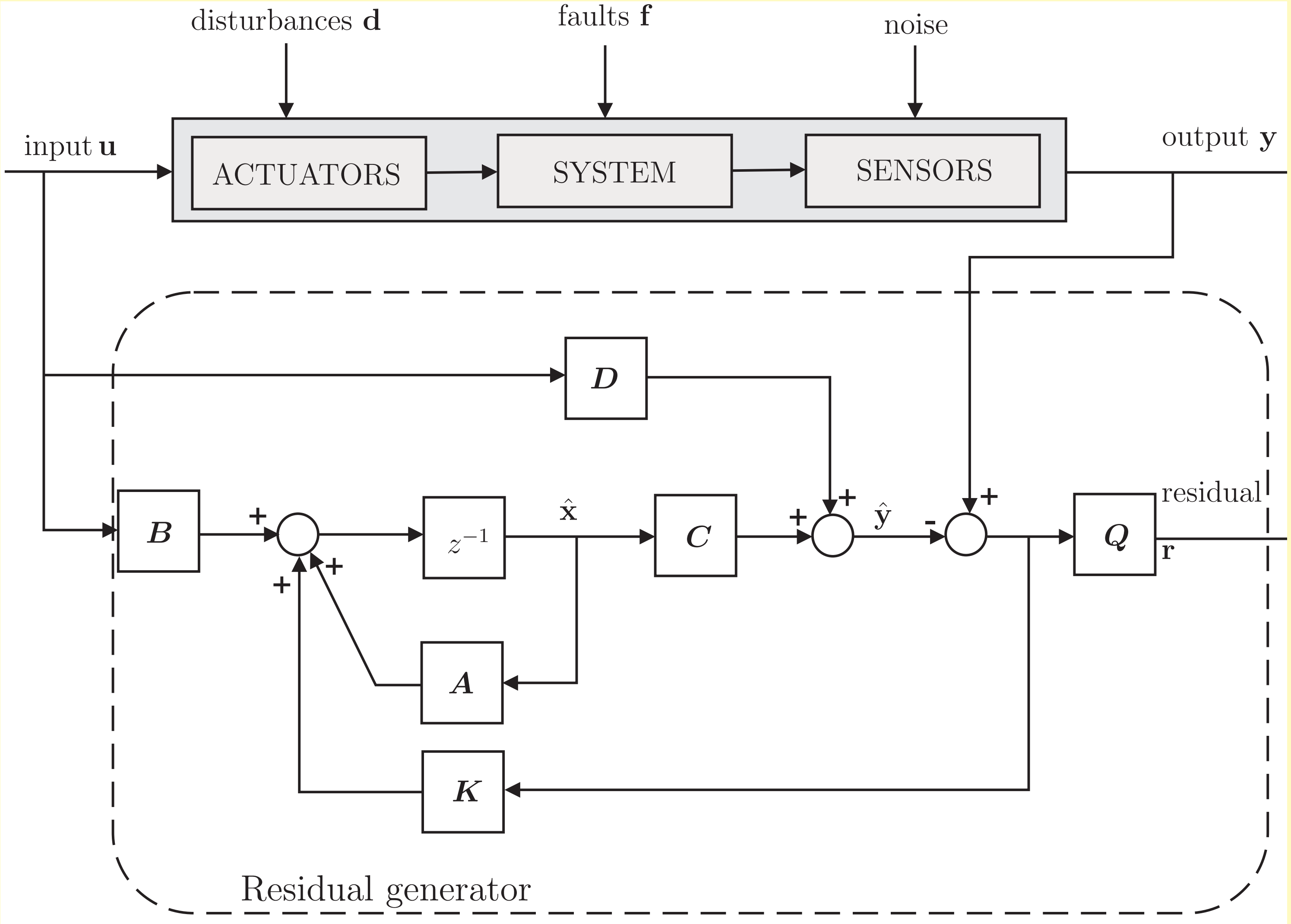
$$\boldsymbol{A}(\hat{\boldsymbol{x}}_k) = \mathrm{diag}[a_{1,1}(\hat{\boldsymbol{x}}_k), a_{2,2}(\hat{\boldsymbol{x}}_k), \ldots, a_{n,n}(\hat{\boldsymbol{x}}_k)]$$

and

$$a_{i,i}(\hat{\boldsymbol{x}}_k) = \mathrm{tgh}(s_{i,i}(\hat{\boldsymbol{x}}_k)), \quad i = 1, \ldots, n.$$

The obtained model can be employed in observer-based fault diagnosis schemes.

➜ Robust observer design for linear systems



➜ Multiobjective optimization in observer design

maximization of the fault effect

minimization of the sensor noise effect

$$\begin{bmatrix} J_1(\mathbf{K},\mathbf{Q}) \\ J_2(\mathbf{K},\mathbf{Q}) \\ J_3(\mathbf{K},\mathbf{Q}) \\ J_4(\mathbf{K},\mathbf{Q}) \end{bmatrix}$$

minimization of the disturbance and initial condition effects

minimization of the input noise effect

➜ Design of an Extended Unknown Input Observer (EUIO) with genetic programming

❒ Class of non-linear systems

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}(\boldsymbol{u}_k) + \boldsymbol{E}_k\boldsymbol{d}_k \\ \boldsymbol{y}_{k+1} &= \boldsymbol{C}_{k+1}\boldsymbol{x}_{k+1}\end{aligned}$$

❒ Linearization around the current state estimate $\hat{\boldsymbol{x}}_k$:

$$\boldsymbol{A}_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k}$$

- Convergence of the EUIO

  - Main objective – *to show the relevance of an appropriate selection of the instrumental matrices* $\boldsymbol{Q}_k$ *and* $\boldsymbol{R}_k$ (Witczak *et al.*, 2002, International Journal of Control, Vol. 75, No. 13):

$$\bar{\sigma}\left(\boldsymbol{\alpha}_k\right) \leqslant \gamma_1 = \frac{\underline{\sigma}\left(\boldsymbol{A}_k\right)}{\bar{\sigma}\left(\boldsymbol{A}_k\right)} \left(\frac{(1-\zeta)\underline{\sigma}\left(\boldsymbol{P}_k\right)}{\bar{\sigma}\left(\boldsymbol{A}_{1,k}\boldsymbol{P}'_k\boldsymbol{A}_{1,k}^T\right)} \right)^{\frac{1}{2}}$$

$$\bar{\sigma}\left(\boldsymbol{\alpha}_k - \boldsymbol{I}\right) \leqslant \gamma_2 = \frac{\underline{\sigma}\left(\boldsymbol{A}_k\right)}{\bar{\sigma}\left(\boldsymbol{A}_k\right)} \left(\frac{\underline{\sigma}\left(\boldsymbol{C}_k^T\right)\underline{\sigma}\left(\boldsymbol{C}_k\right)}{\bar{\sigma}\left(\boldsymbol{C}_k^T\right)\bar{\sigma}\left(\boldsymbol{C}_k\right)} \frac{\underline{\sigma}\left(\boldsymbol{R}_k\right)}{\bar{\sigma}\left(\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T + \boldsymbol{R}_k\right)} \right)^{\frac{1}{2}}$$

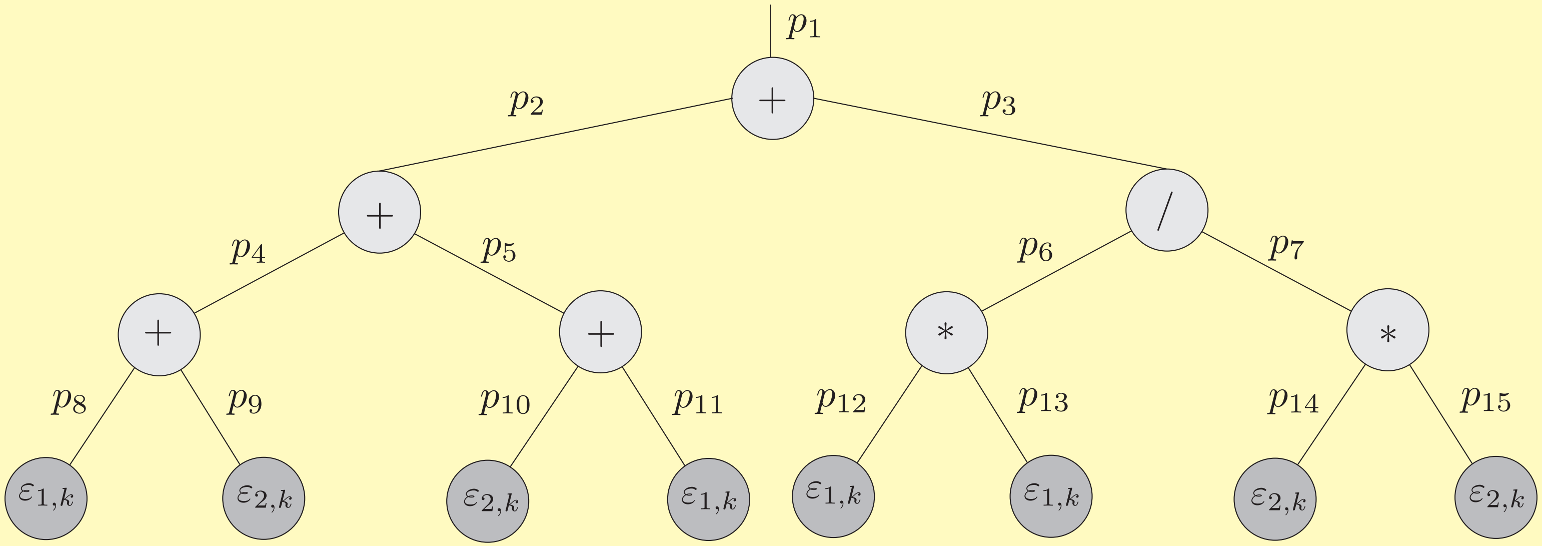
  - Since

$$\boldsymbol{P}_k = \boldsymbol{A}_{1,k}\boldsymbol{P}'_k\boldsymbol{A}_{1,k}^T + \boldsymbol{T}_k\boldsymbol{Q}_{k-1}\boldsymbol{T}_k^T + \boldsymbol{H}_k\boldsymbol{R}_k\boldsymbol{H}_k^T,$$

    it is clear that an appropriate selection of the instrumental matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ may enlarge the bounds γ_1 and γ_2 and, consequently, the domain of attraction.

➠ Structural optimization problem and its genetic-programming-based representation

$$\boldsymbol{Q}_{k-1} = q^2(\boldsymbol{\varepsilon}_{k-1})\boldsymbol{I} + \delta_1\boldsymbol{I} \quad \boldsymbol{R}_k = r^2(\boldsymbol{\varepsilon}_k)\boldsymbol{I} + \delta_2\boldsymbol{I}$$



➠ Definition of the optimization criterion

$$(\boldsymbol{Q}_{k-1}, \boldsymbol{R}_k) = \arg \min_{q(\varepsilon_{k-1}), r(\varepsilon_k)} j_{\text{obs},3}(q(\varepsilon_{k-1}), r(\varepsilon_k))$$

where

$$j_{\text{obs},3}(q(\varepsilon_{k-1}), r(\varepsilon_k)) = \frac{j_{\text{obs},2}(q(\varepsilon_{k-1}), r(\varepsilon_k))}{j_{\text{obs},1}(q(\varepsilon_{k-1}), r(\varepsilon_k))}$$

$$j_{\text{obs},1}(q(\varepsilon_{k-1}), r(\varepsilon_k)) = \sum_{k=0}^{n_t-1} \text{trace}\boldsymbol{P}_k$$

$$j_{\text{obs},2}(q(\varepsilon_{k-1}), r(\varepsilon_k)) = \sum_{k=0}^{n_t-1} \varepsilon_k^T \varepsilon_k.$$

☛ CONCLUDING REMARKS

- ❒ Evolutionary algorithms constitute an attractive optimization tool in designing FDI systems
  - Multimodal cost functions
  - Multiobjective optimization
  - Structural optimization
  - Non-differentiable cost functions
- ❒ They should be applied only when the classical approaches fail to solve a given problem
- ❒ They cannot be applied to on-line optimization problems