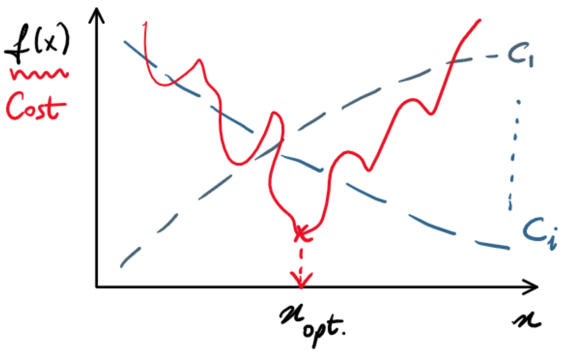
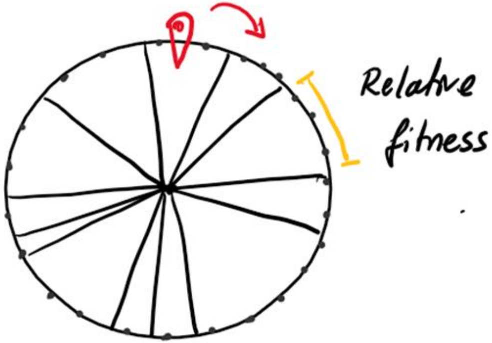
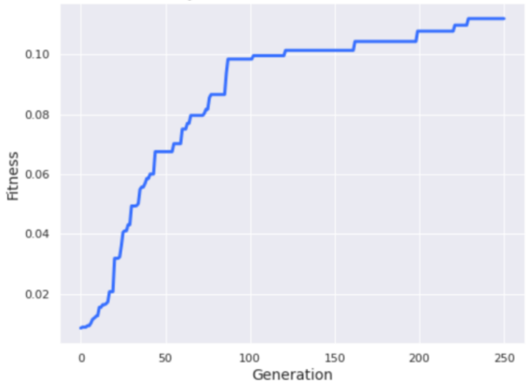
Data Driven Engineering II: Advaced Topics

Genetic algorithms:
towards data driven control

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer







DDE: Dynamical Systems

Discovery // Characterization // Simulation

Karlsruhe Institute of Technology

Understanding how ... works

* "Science" := $\underbrace{\text{interpret. of observations}}_{\text{... in a systematic way}}$

* Prereq. ⇒ organized "book keeping"

* $\left.\begin{array}{l} \text{Scientific Data} \Rightarrow \underbrace{\text{Discovery} \Rightarrow \text{Management Optimization}}_{\text{" Engineering "}} \end{array}\right\}$ interpret. as governing model eqns.

$$f(\underbrace{x}_{\text{state}}, \overset{\text{Transient}}{t}, \underbrace{\alpha, \beta, \gamma \ldots}_{\text{parameters}})$$

DDE: Dynamical Systems

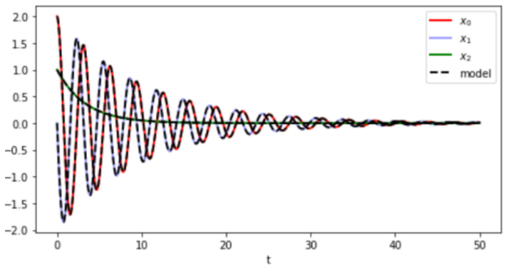
Discovery // Characterization // Simulation

Understanding how ... works

Transient

$f(x, t, \underbrace{\alpha, \beta, \gamma \ldots}_{\text{parameters}})$

state



Data-Driven Model Discovery

Understand how X works

this 'model' would let me...

- ☑ Interpretability : What if ..
- ☑ Design & Optimization
- ☑ Future state prediction
- ☑ Active control with feedback

DDE: Dynamical Systems
Discovery // Characterization // Simulation

* FACT : Process ⇔ Controller
linked &

Tasks:

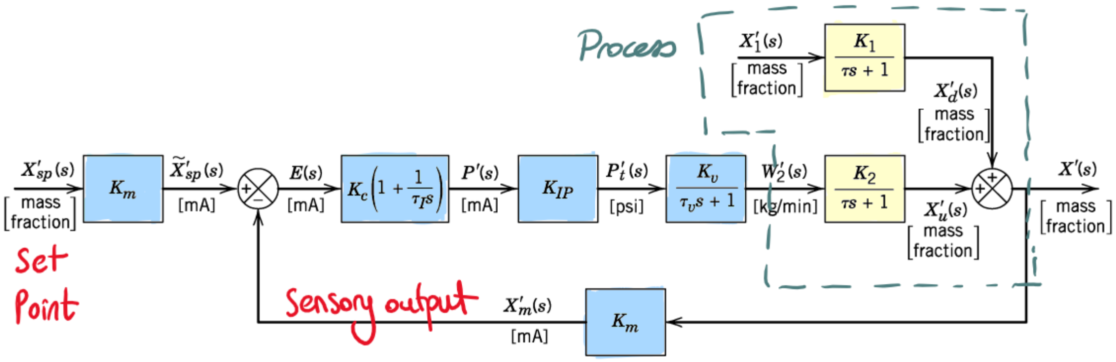
① Create a phy. model I DDE-I

② Create a controller model

③ Coupled optimization

DDE-II

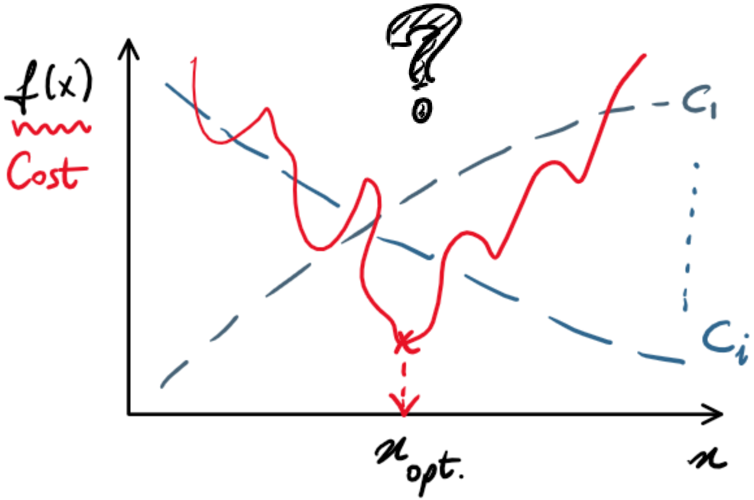
Closed-loop Control System:



DDE ⟺ Optimization

* Optimization landscape
* Evolutionary algorithms
* Genetic algorithms
* Genetic programing

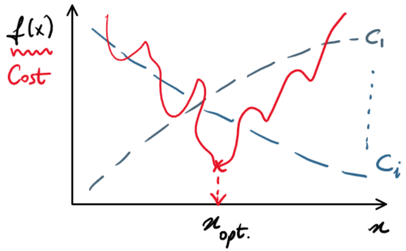
Optimization Problem

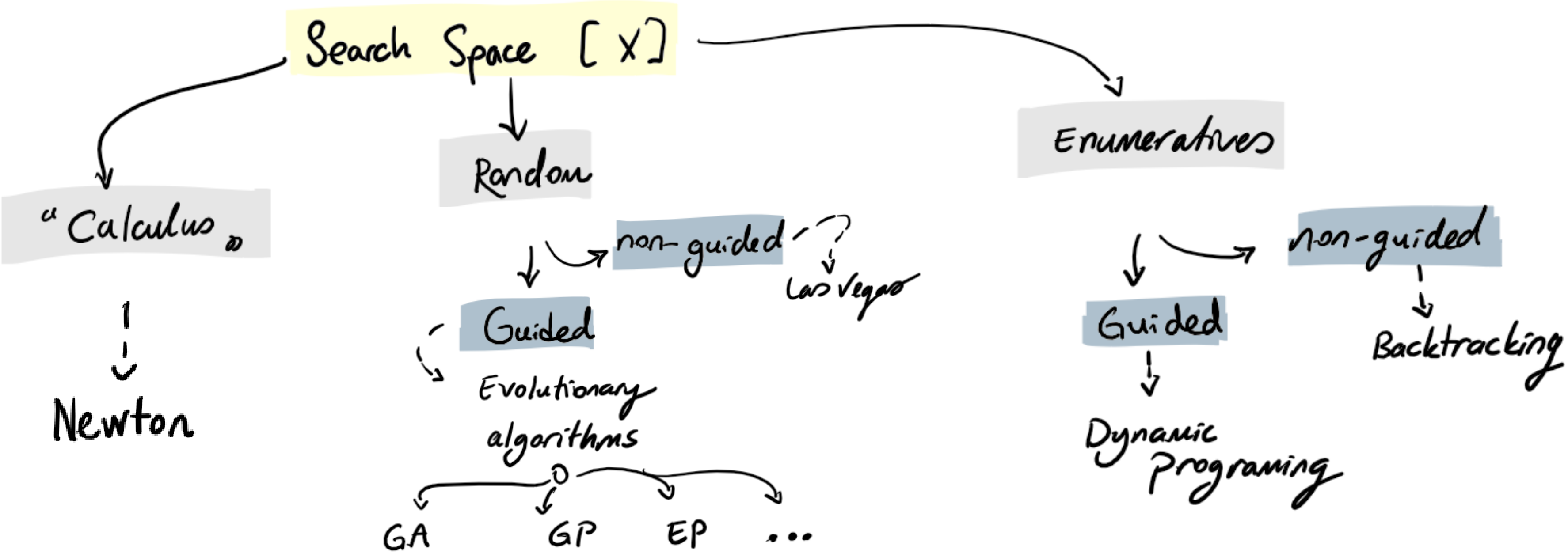


⇨ Competing factors

⇨ Cost function

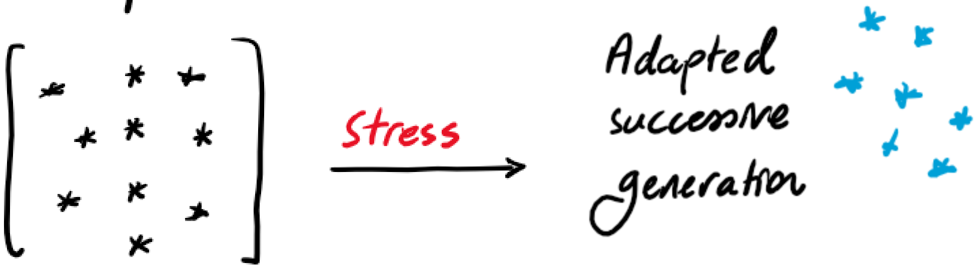
- ☐ Single/multiple variable
- ☐ Static / dynamic
- ☐ Discrete / continuous
- ☐ Contrained / uc
- ☐ Function / Data


f(x)
Cost
c_1
c_i
$x_{opt.}$
x


Search Space [X]
"Calculus"
Newton
Random
non-guided
Las Vegas
Guided
Evolutionary algorithms
GA
GP
EP
...
Enumeratives
Guided
Dynamic Programing
non-guided
Backtracking

Evolutionary Algorithms

* Population-based



Eg Rechenberg 1965, 1973
Airfoil optimization
"Evolutionsstrategie"

* Genetic algorithm: 60s, 70s
  - John Holland "Adaptation in Natural and Artificial Systems"

Genetic Algorithms :

⇨ no rigorous definition on GA

population of chromosomes

selection via a fitness

offsprings → crossover & mutations

The 2006 NASA ST5 spacecraft antenna.
The shape was found by EA

Genetic Algorithms:

Selection: a measure of fitness ⇨ stress over population

↳ not all parents have offsprings (hyperparameter)

↳ individuals not fitting ⇒ eliminated (hp.)

Crossover: allow to exchange info among members (hp.)

Mutations: Change a single element in an individual (hp.)

Algorithm of GA:

1. Initialize population
2. Get current fitness (+filtering)
3. Create offsprings ⟷ crossover
4. Mutations
5. "Survival of the fittest"
   ↳ update the population

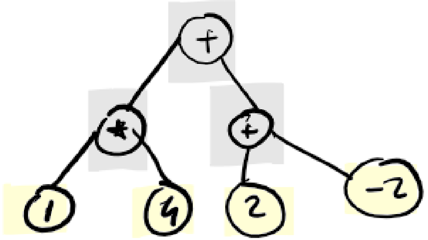
Challanges:

- ☐ Define a fitness function
- ☐ Problem representation
- ☐ Early convergence
- ☐ Calling fitness func. many times
- ☐ Hyperparameter tuning

Gene Encoding

? how to represent individual genes

- ☐ Binary encoding (11010001010)
- ☐ Octal encoding [0,7] → 157124613
- ☐ Hexadecimal enc. [0-9, A-F] → 9CE7
- ☐ Value encoding [1,16 5.24 2.41 2.54]
  [N N S W E]
- ☐ Tree encoding



Breeding

(i) Select parents (ii) Create offsprings (iii) update population

Selection

~ higher the fitness ; higher the chance ~

* affects the conversion rate

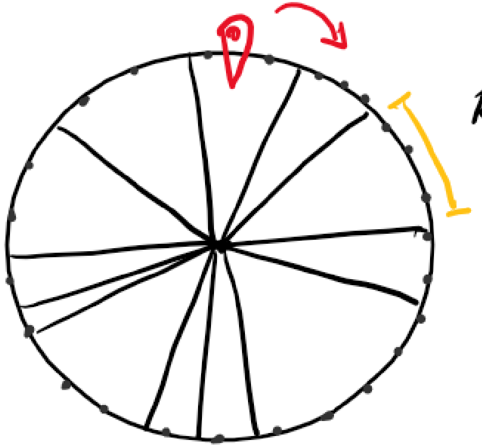
(a) proportionate-based ~ relative fitness

(b) ordinal-based ~ rank within population

Selection

~ higher the fitness; higher the chance ~

Roulette-Wheel



'Rotate' N times
for N parents

« Parent Pool »

* Not a strong selection
* Noisy

Selection

~ higher the fitness ; higher the chance ~

Rank Selection

Roulette ⇒ biased
↳ less chance for potential solutions

* Rank population 1 ⟶ N (best)

* Slow convergence
* Diversity preserved

[Ranked]
⇩
Potential parents

* random selection
↳ pairs
↳ tournament

Selection

~ higher the fitness ; higher the chance ~

Boltzmann Selection

$$P = \exp\left[-(f_{max} - f_i)/T\right]$$

$$T = T_0 (1-\alpha)^k$$

$$k = (1 + 100g/G)$$

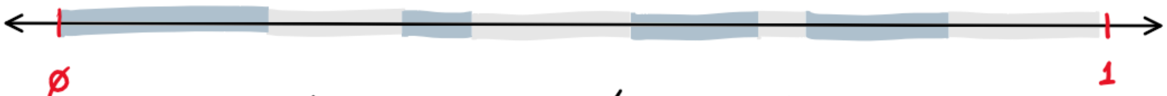
- $T_0 = [5, 100]$
- $\alpha = [0, 1]$
- $g :=$ generation i
- $G :=$ max. gen. allowed
- $f_i =$ fitness

Gradually narrow down search space

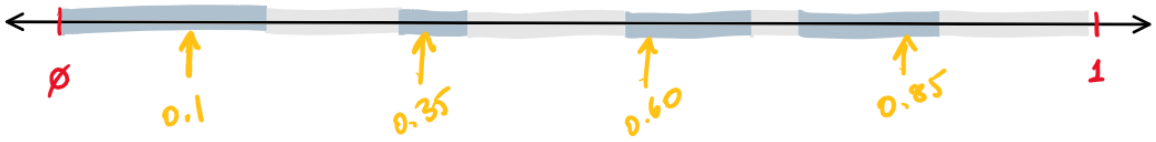
Selection

~ higher the fitness; higher the chance ~

Stochastic Universal Sampling



* 4 pointers $\Rightarrow x_i = 1/4 = 0.25$
* $R_1 = [0, 25] \Rightarrow 0.10$



Selection

~ higher the fitness; higher the chance ~

Random Selection

* Randomly select N parents.
* Much more noisy

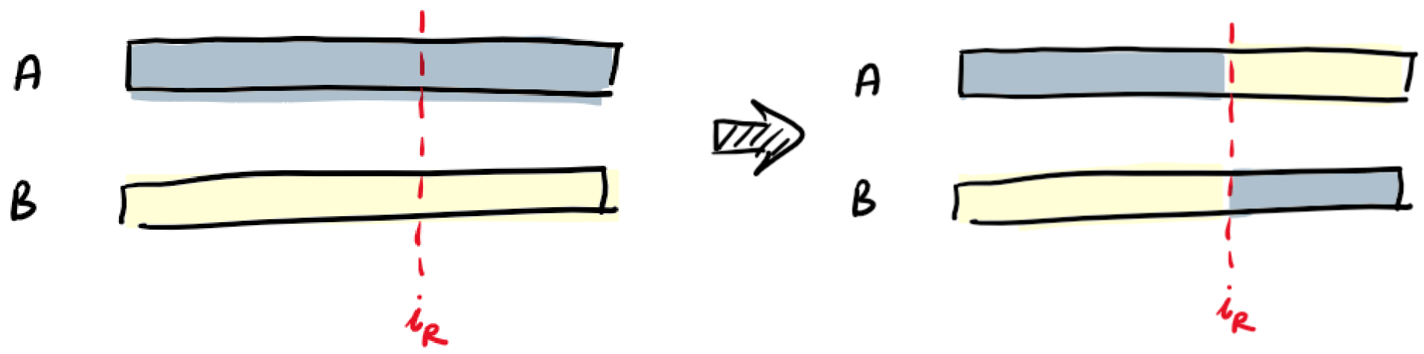
Elitism

* Best E chromosomes are directly passed to parent pool.
* For the rest; other methods are used.

Crossover <=> Recombination

* 2 Parents => 1 offspring
* applied to the parent pool.

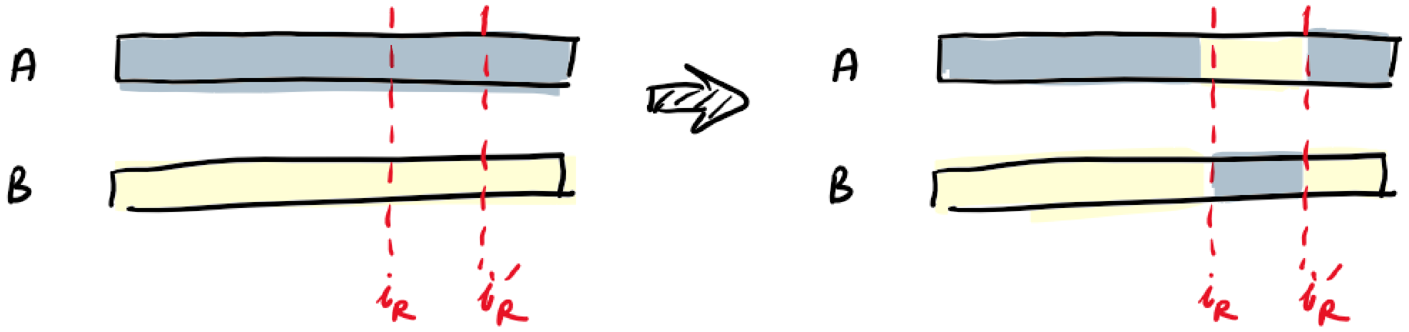
Single point crossover



Crossover <=> Recombination

* 2 Parents ⇒ 1 offspring
* applied to the parent pool.

Two points crossover



Crossover <=> Recombination

- 2 Parents ⇨ 1 offspring
- applied to the parent pool.

Uniform Crossover

- ✓ Parent I 
- ✓ Parent II 
- ✓ Child I $N/2 + N/2 = N$ genes
- ✓ $\text{Child}_{II} := 1 - \text{Child}_{I}$

Others

- ⇒ Three parent crossover
- ⇒ Shuffle Crossover
- ⇒ Custom f.

Mutation :

* Typically after breeding

* Remedy to exit from local minimum

  ↳ extending search space

  ↳ increasing diversity

Mutation:

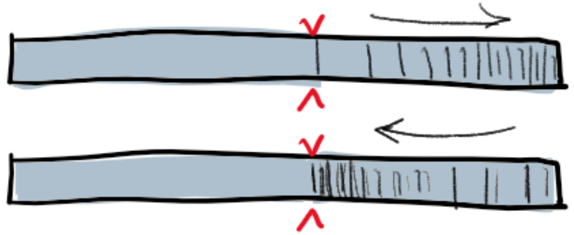
Flipping

* Binary: $0 \Rightarrow 1$, $1 \Rightarrow 0$
* Random numbers ⇒ # events / indices

Interchanging

* Two random genes are exchanged.

Reversing



! Mutation rate ⇒ 100%

GA ⇒ Random search

Population Update ⟺ Replacement

* [Population] → [Parents] → [Offsprings]

  $\underbrace{\text{[Population] → [Parents] → [Offsprings]}}$

  Which ones are kept?

Random Replacement

* Children ⇒ randomly chosen individuals

Both Parents

* Children ⇒ Parents

Weak Parent Replacement

* Child → weaker parent

Convergence

* Max. generation
* Elapsed time
* Track fitness → Best individual
  → worst individual
  → $\sum f_i$ or $\bar{f_i}$

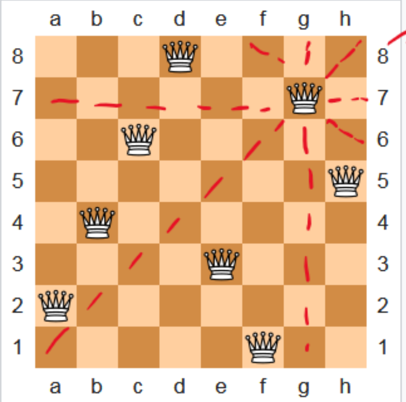
Algorithm of GA:

1. Initialize population
2. Get current fitness (+filtering)
3. Create offsprings ↔ crossover
4. Mutations ϕ
5. "Survival of the fittest"
   ↳ update the population

Examples

1. Password guessing
   - (i) String matching ⇒ Fitness Score
   - (ii) Gene pool ⇒ abc...z

2. "8 Queens"



Solution 1

Fitness score;

4 lines; 32 ⇒ ϕ

Chromosomes;

[: :] ⇒ locations

"N" Queen Problem

Examples

③ Solving system of linear eqs.

$$\left.\begin{aligned} 2x + y - 2z &= 3 \\ x - y - z &= 0 \\ x + y + 3z &= 12 \end{aligned}\right\} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = ?$$

* Ch. $\Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix}$

* Fitness $\Rightarrow \sum f_i \Rightarrow 0$

④ PyGAD $\Rightarrow$ Train a NN via GA

* Regression $\Rightarrow$ Airfoil Noise Problem
* Chromosomes $\Rightarrow$ Weights of NN model
* fitness $\Rightarrow$ MSE / MAE

Karlsruhe Institute of Technology

colab