Linear Drive Optimization by AI

MACHINE LEARNING BASED OPTIMIZATION

Technology Day @ OST

Chafic Abu Antoun
June 11, 2024
Buchs SG

AGENDA

- **Corporate Research & Technology @ Hilti**
- Drive optimization methods
- Machine learning based optimization

HILTI COMPUTATIONAL ENGINEERING & CR&T

Corporate Research & Technology
Andreas Bong

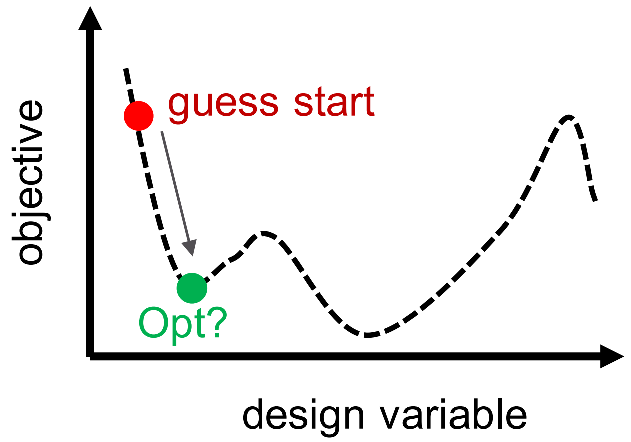
- Mechatronics & Robotics Systems
- Visual Computing & Machine Learning
- Battery Technology
- **Computational Engineering**
  - Team Kaufering
  - Team Schaan
    - Team Zhanjiang*
- Materials Research
- Fastening Applications
- Intellectual Property

AGENDA

- Corporate Research & Technology @ Hilti
- Drive optimization methods
- Machine learning based optimization

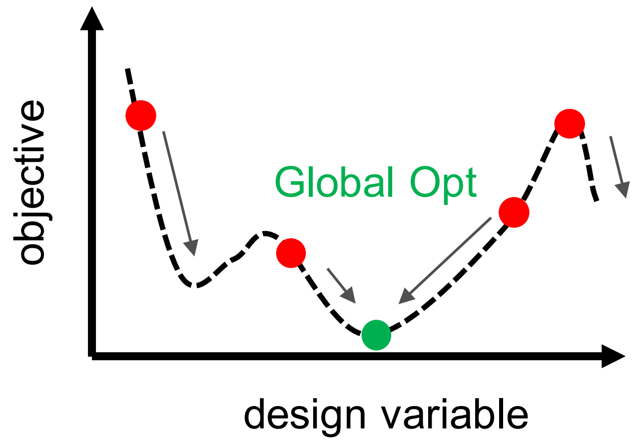
CLASSICAL VERSUS NEW APPROACH IN OPTIMIZATION

Gradient Based Optimization



- Less function calls
- Local optimum
- Typical for few design variables
- Typical for convex systems

Evolution Based Optimization

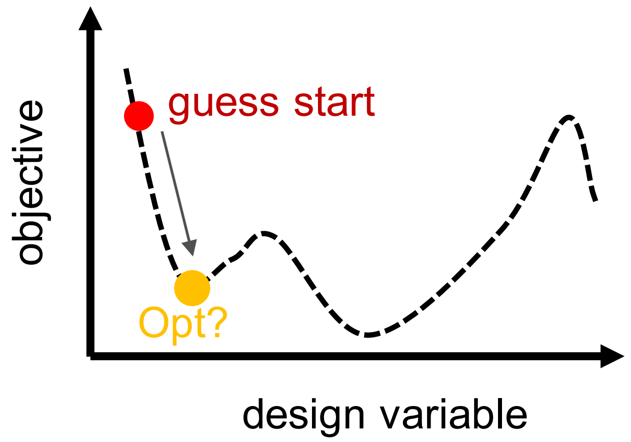


- more function calls (expensive)
- global optimum
- Typical for many design variables
- Typical for convex and concave systems

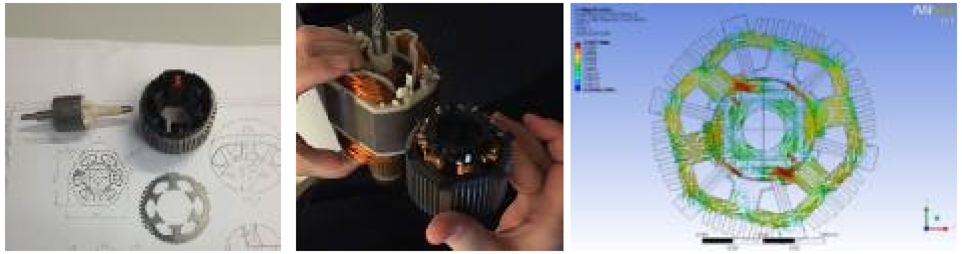
This is solved here by ML

DRIVE OPTIMIZATION TYPES AND EXAMPLES

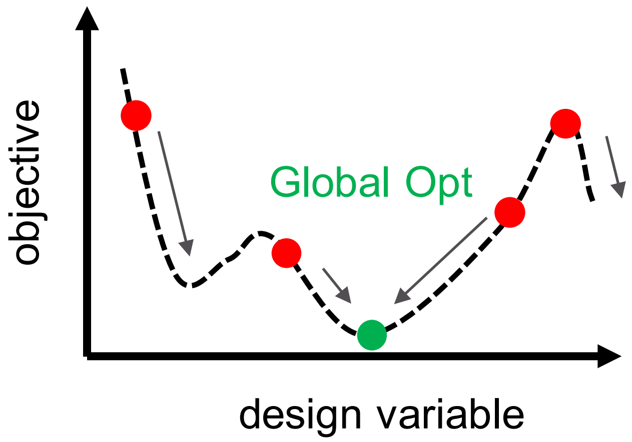
Gradient Based Optimization



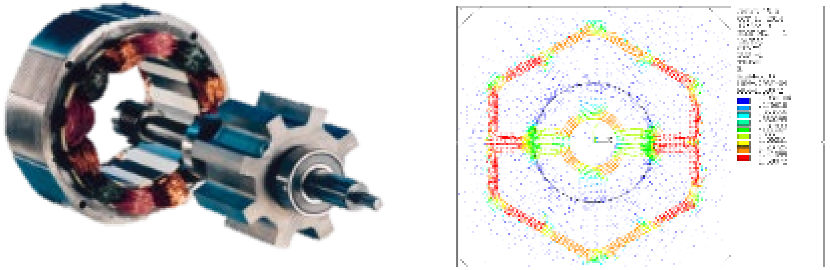
Before 2015 many motors were electro-thermally optimized using gradient based methods like <u>SQP</u> for example: Brushless AC / Switch Reluctance



Evolution Based Optimization proved superiority



Since 2015 many linear and rotary motors were electro-thermally optimized using evolution based methods like <u>Particle Swarm</u> or <u>Genetic Algorithm</u> for example: Switch Reluctance / Inductive / Brushed



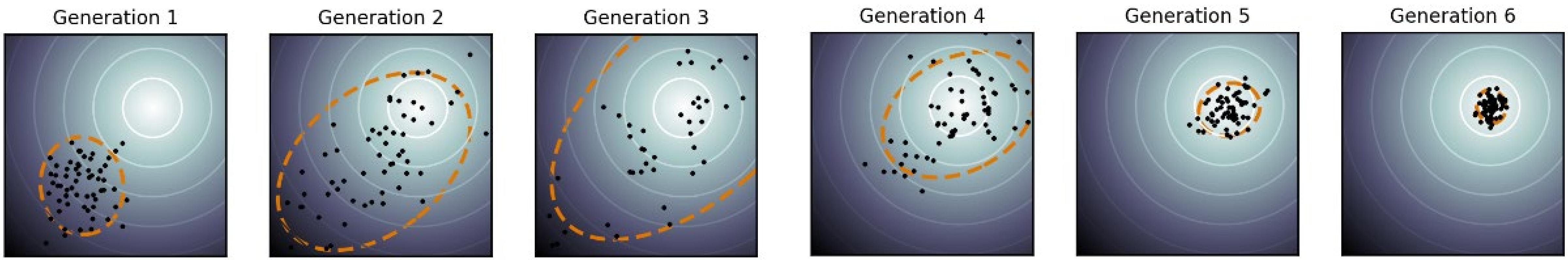
AGENDA

- Corporate Research & Technology @ Hilti
- Drive optimization methods
- Machine learning based optimization

MACHINE LEARNING AND EVOLUTION METHODS

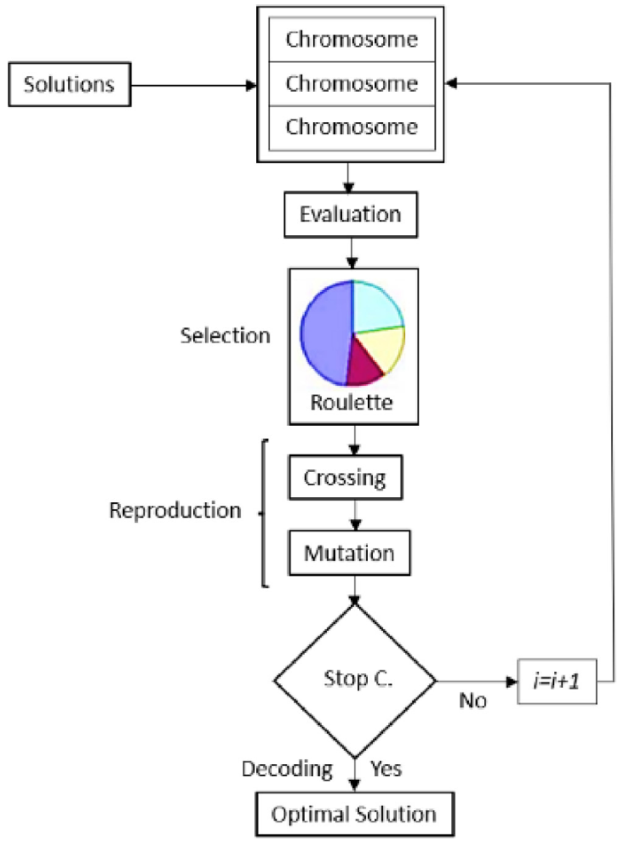
Covariance Matrix Adaptation Evolution Strategy

The covariance matrix of the distribution is updated (incrementally) such that the likelihood of previously successful search steps is increased.



Evolutionary algorithm is broadly based on the principle of biological evolution (survival of the fittest)
Learning through generations to reach an **optimum** generation of **design variables**

OPTIMIZATION USING GENETIC ALGORITHM



Algorithm 1 Evolutionary Algorithm

Input: Population size P, Crossover possibility c_p, Mutation possibility m_p,

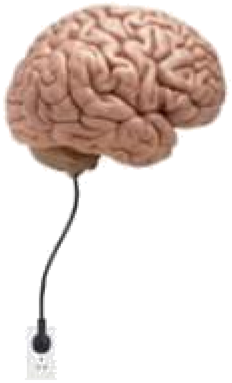
Output: Optimum result r

```
Given population size P and randomly generate the first offspring with a population P
Generate the fitness of each individual and select the best top 10 percent of offspring as the elite
while Not reach iteration limits or get optimum do
    Generate new 90 percent populations
    Randomly mutate the populations with possibility m_p
    Randomly crossover two population with possibility c_p
    Select the best 10 percent of offspring as the elite
end while
return Best Population
```

Classical Algorithm	Genetic Algorithm
Generates a single point at each iteration. The sequence of points approaches an optimal solution.	Generates a population of points at each iteration. The best point in the population approaches an optimal solution.
Selects the next point in the sequence by a deterministic computation.	Selects the next population by computation which uses random number generators.

https://en.wikipedia.org/wiki/Genetic_algorithm

MACHINE LEARNING → A MODEL WITH EXPERIENCE



Nowadays, our **experiences** are saved **data** on drives, clouds, clusters, …

experiences

- Available material
- Manufacturing process
- Geometrical design
- Design settings

Inputs
X

$Y = f(X)$

Outputs
Y

- Weight
- Cost
- Quality
- Handling
- Robustness
- Lifetime

MACHINE LEARNING CATEGORIES

Machine Learning

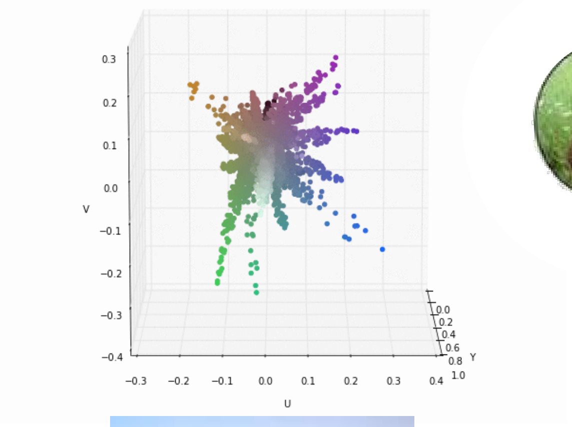
- Unsupervised
  - Clustering
- Supervised
  - Classification
  - Regression
    + evolutionary optimization
- Reinforced

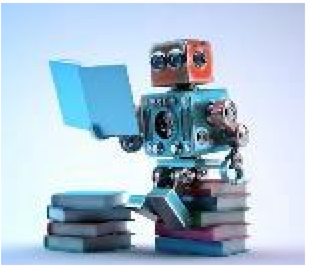
Today's example

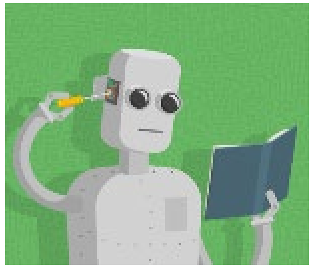
understanding

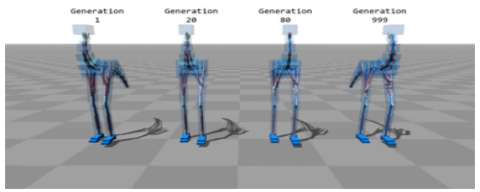
modelling

Prediction & self adaptation

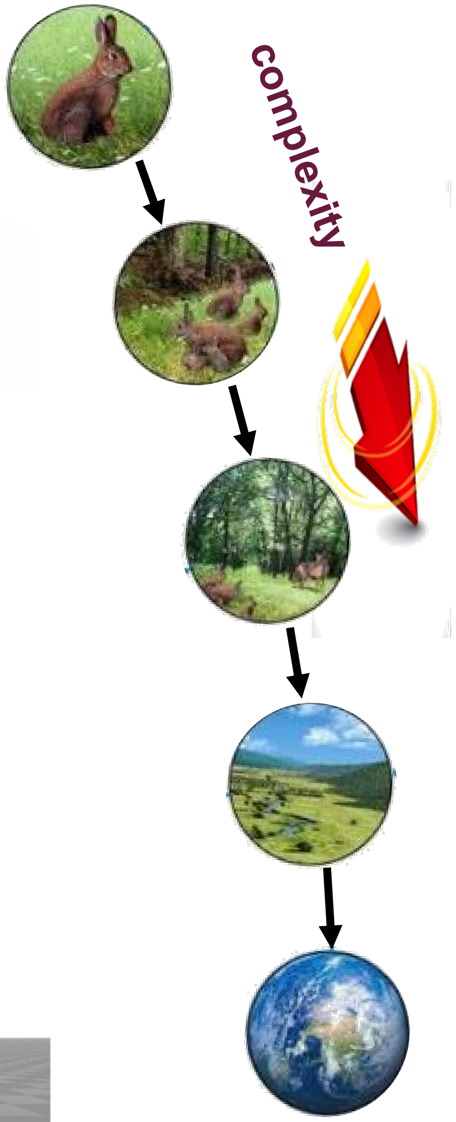






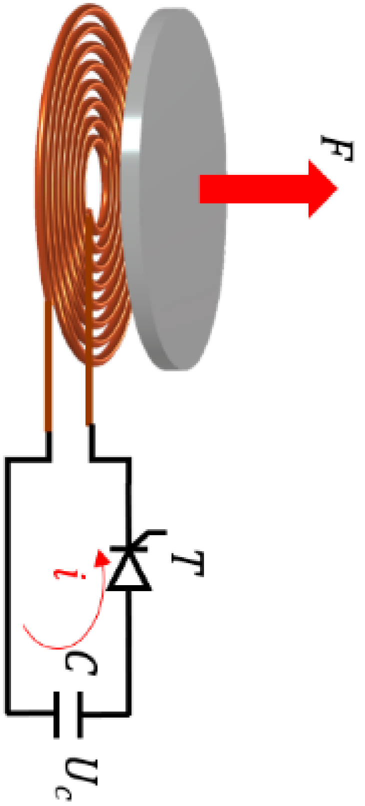


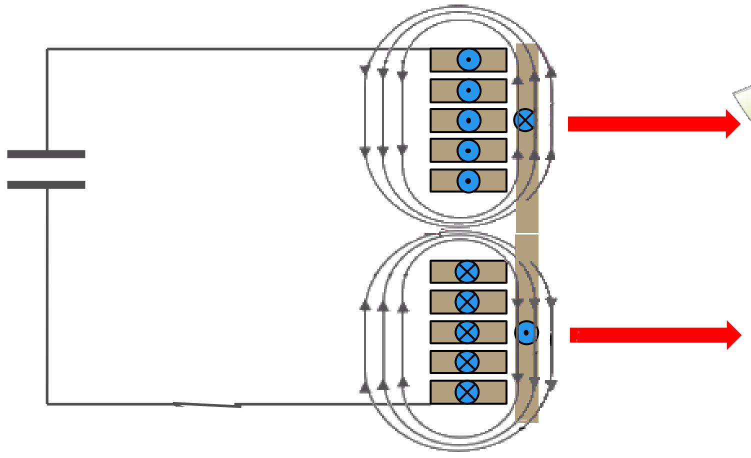
complexity

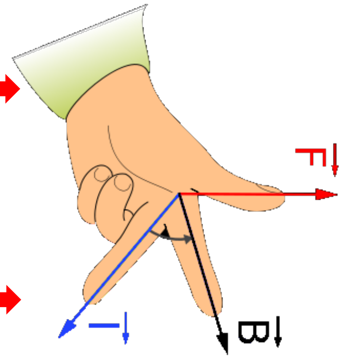


EXAMPLE OF INDUCTIVE THOMSON COIL

A Published Thomson Coil







$$\mathbf{F} = \int_V \mathbf{J} \times \mathbf{B} \, dV$$

Lorentz Force

DRIVE REQUIREMENTS UNDER CHANGING CONSTRAINTS

Inputs, limitations and constraints

Charging power

Charging speed

Capacitance

Voltage

Windings

Current

Geometry

Material

Temperature

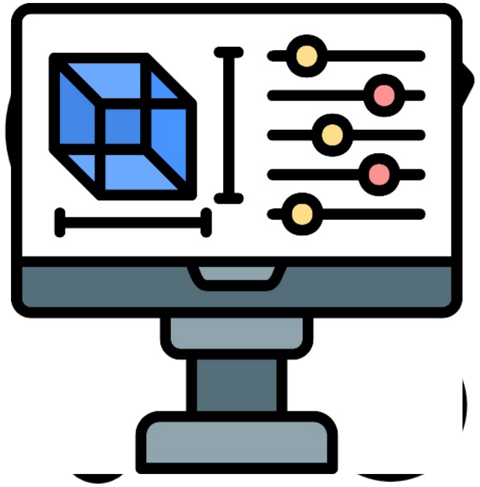
Outputs Target

1. High energy
2. Low weight
3. Customer usage frequency
4. High efficiency

MACHINE LEARNING (REGRESSION)

Discrete
experimental
simulation
data

Train data by machine learning



Continuous function Y of any X

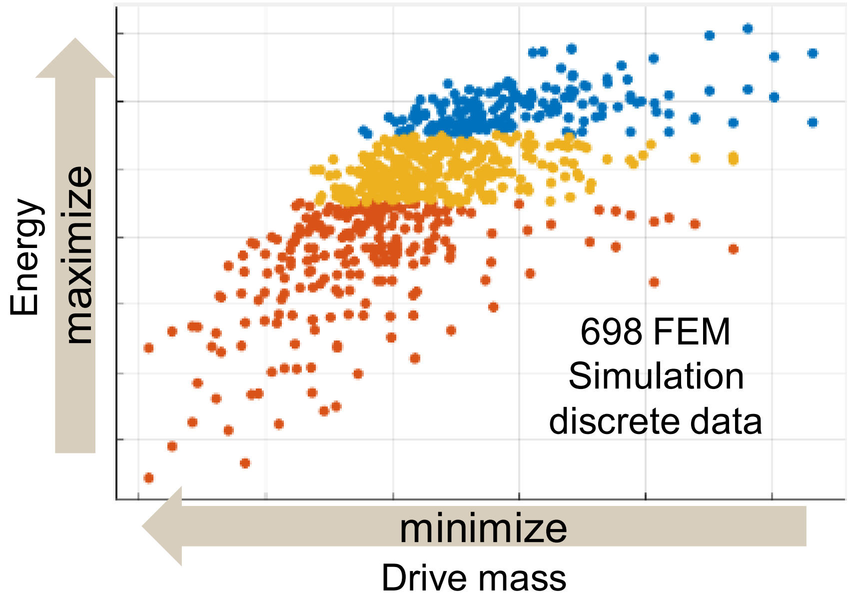
Input X → Output Y

$$Y = f(X)$$

$$X = \begin{bmatrix} Geometry\ variable\ 1 \\ Geometry\ variable\ 2 \\ \vdots \\ Electric\ variable\ 6 \\ \vdots \end{bmatrix} \qquad Y = \begin{bmatrix} Output\ Energy\ 1 \\ Drive\ Mass\ 2 \\ \vdots \\ Peak\ Current\ 5 \\ \vdots \\ Coil\ wire\ thickness\ 12 \\ \vdots \\ Heat\ Coil\ 16 \\ Heat\ Piston\ 17 \\ Heat\ Capacitor\ 18 \end{bmatrix}$$

Within input boundaries
→ a continues function is created by training discrete data

MACHINE LEARNING – REGRESSION – INDUCTION DRIVE



Accurate Emag FEM
simulations

Train discrete data to get
continuous function

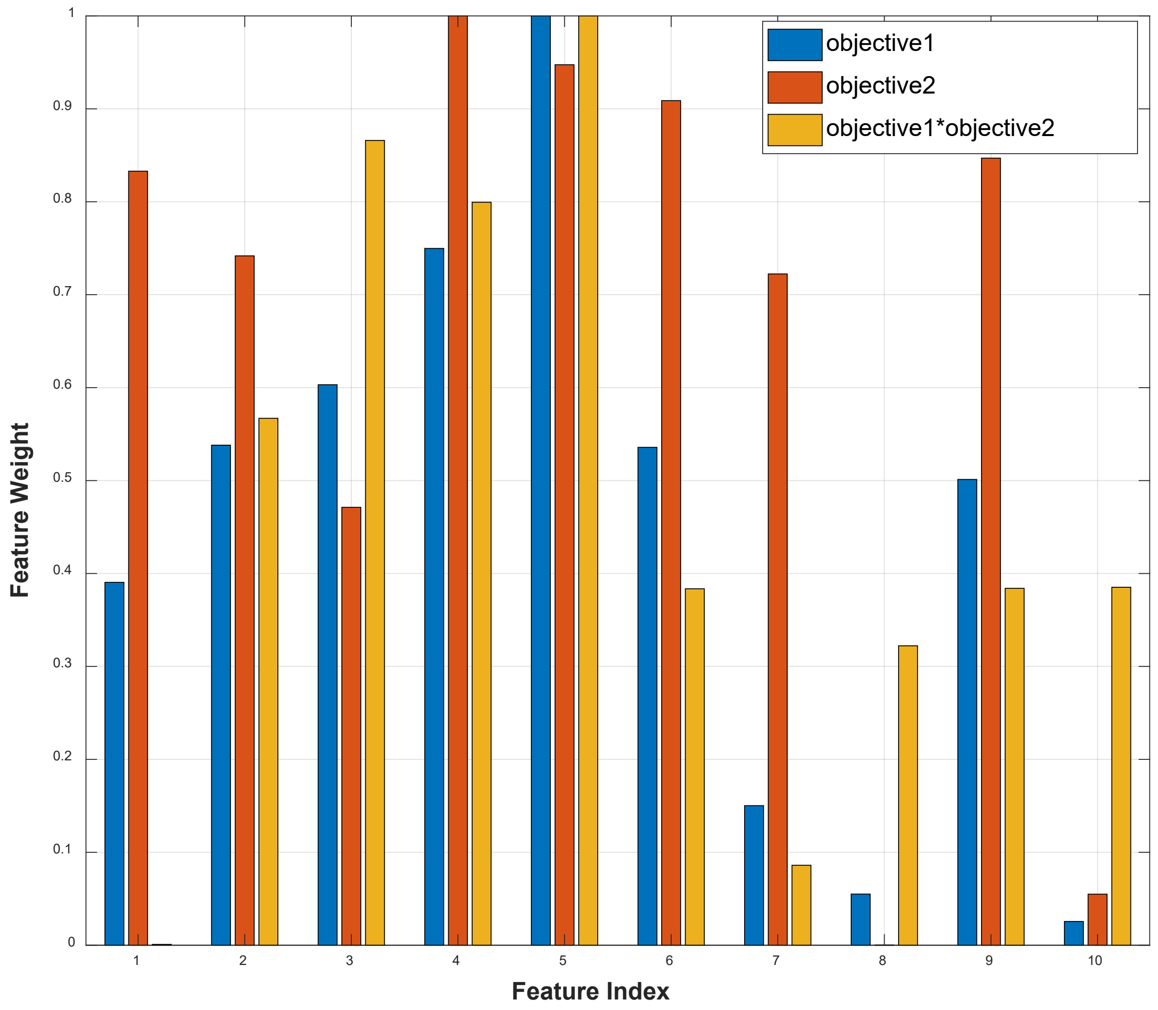
Machine learning by
Gaussian Process Regression

Continuous function

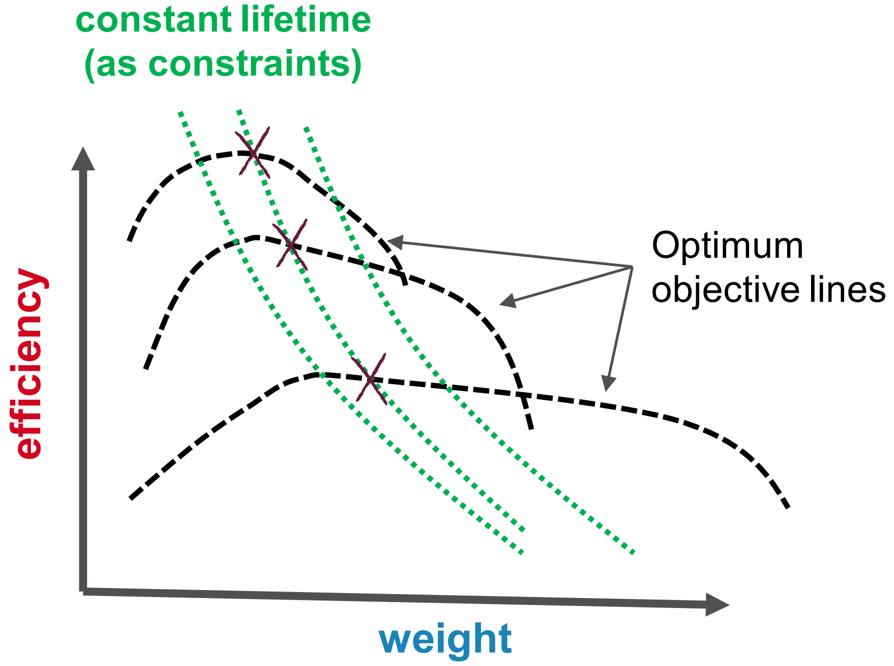
Any Input X → Output Y

$$Y = f(X)$$

OBJECTIVES (IMPORTANT FEATURES)



Feature importance depends on a defined objective
It is so fast and easy to see the effect of each feature on different objectives



One suggestion of how **objectives** can be limited by **constraints** using **genetic algorithm**

GENETIC ALGORITHM FINDS AN OPTIMUM IN 5 MINUTES

Inputs, limitations and constraints

Charging power

Charging speed

Capacitance

Voltage

Windings

Current

Geometry

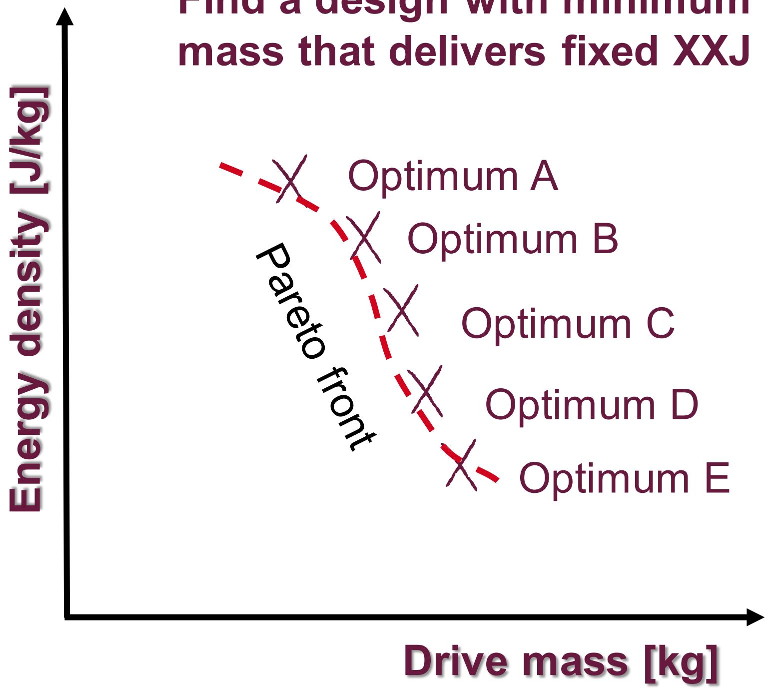
Material

Temperature

Fast generations of optimum pareto front even if we change the constraints or objectives then we only need few minutes to optimize again because the data is already there

Within an accuracy of **0.2%** of expensive FEA simulations

Find a design with minimum mass that delivers fixed XXJ



THANK YOU

Contact
Dr. Chafic Abu Antoun

Hilti Aktiengesellschaft
Feldkircherstrasse 100
9494 Schaan, Liechtenstein

Tel: +423- 234 3408
chafic.abuantoun@hilti.com
www.hilti.group

BACKUP SLIDES

CMA-ES (COVARIANCE MATRIX ADAPTATION)

3.1.2 Covariance Matrix Adaptation Evolution Strategy

After talking about Genetic Algorithm, we will introduce the CMA-ES, which is quite good at generating numeric type solutions compared with GA. What is more, compared with the Simple Gauss evolutionary strategy (SGES), CMA-ES can adjust the step while training. Thus, the optimal value can be found in a short time with high accuracy.
In CMA-ES, the new distribution parameter can be like this:

$$\theta = (\mu, \sigma, C), p_\theta(x) \sim N(\mu, \sigma^2 C) \sim \mu + \sigma N(0, C)$$

where C is the covariance matrix and C has the following good properties:

1. C is always a diagonal matrix
2. C can be eigendecomposed into eigenvectors $B = [b_1, b_2, ..., b_n]$ and eigenvalues $\lambda_1^2, \lambda_2^2, ..., \lambda_n^2, D = diag(\lambda_1^2, ..., \lambda_n^2)$

Sample we can sample the new candidates in each generation by this formula:

$$x_i^{(t+1)} = \mu^{(t)} + \sigma^{(t)} y_i^{(t+1)} \text{ where } y_i \sim N(0, C^{(t)}),\ i = 1, ..., \Lambda$$

where $x_i^{(t+1)}$ means the new candidates of the next generations, $\mu^{(t)}$ means the mean of elites of current generation and $\sigma^{(t)}$ means the step of current generation and y_i is the subject to normal distributed.

Control Step We know that $\sigma^{(t)}$ controls the step of each generation, it is separated from the covariance matrix, so we can change the step size faster than we can change the full covariance. To estimate the step's appropriateness, CMA-ES get the sum of continuous moving sequence $\frac{1}{\lambda}\sum_i^\lambda y_i^{(j)}, j = 1, 2, ..., t$ to get the evolution path p_σ and compare the evolution path with the path generated by random selection. if the evolution path is larger than the random selection path, reduce the $\sigma^{(t)}$, vice versa.

Adaptive covariance matrix The eigendecomposition of Covariance Matrix C obeys:

$$C = BD^2B^T \quad ,$$

We can re-estimate the origin Covariance Martix C using the sampled population.

$$C_\lambda^{(t+1)} = \frac{1}{\lambda}\Sigma_{i=1}^\lambda y_i^{(t+1)} y_i^{(t+1)^T} = \frac{1}{\lambda \sigma^{(t)^2}} \Sigma_{i=1}^\lambda (x_i^{(t+1)} - \mu^{(t)})(x_i^{(t+1)} - \mu^{(t)})^T$$

In this formula, we can see that this estimation is only reliable when the population is larger. However, in each iteration, we want to has rapid iteration with a lower population. CMA-ES has a more reliable but more complicated way to update the C. It contains two kinds of unique ways.
The first method is using the history of C, we can also use the **Polyak Average**:

$$C^{(t+1)} = (1 - \alpha_{c\lambda})C^{(t)} + \alpha_{c\lambda} C_\lambda^{(t+1)} = (1 - \alpha_{c\lambda})C^{(t)} + \alpha_{c\lambda} \frac{1}{\lambda} \Sigma_{i=1}^\lambda y_i^{(t+1)} y_i^{(t+1)^T} \quad ,$$

and we choose the $\alpha_{c\lambda} = min(1, \lambda/n^2)$ normally.
The second way is using a path p_c to log the symbol information, p_c also subject to the normal distribution $N(0, C)$.

$$p_c^{(t+1)} = (1 - \alpha_{cp})p_c^{(t)} + \sqrt{\alpha_{cp}(2 - \alpha_{cp})\lambda} \frac{\mu^{(t+1)} - \mu^{(t)}}{\sigma^{(t)}} \quad ,$$

and we use p_c to update the covariance martix C:

$$C_\lambda^{(t+1)} = (1 - \alpha_{c1})C^{(t)} + 1 - \alpha_{c1} p_c^{(t+1)} p_c^{(t+1)^T} \quad .$$

Finally, we combine these two methods and here we get the final update formula:

$$C^{(t+1)} = (1 - \alpha_{c1} - \alpha_{c\lambda})C^{(t)} + \alpha_{c1} p_c^{(t+1)} p_c^{(t+1)^T} + \alpha_{c\lambda} \frac{1}{\lambda} \sum_{i=1}^\lambda y_i^{(t+1)} y_i^{(t+1)^T}$$