

# *EVOLUTIONARY COMPUTATION IN CONTINUOUS OPTIMIZATION AND MACHINE LEARNING*

*Status and planning seminar*

*Optimization & Evolutionary Computation*

# *INTRODUCTION*

---

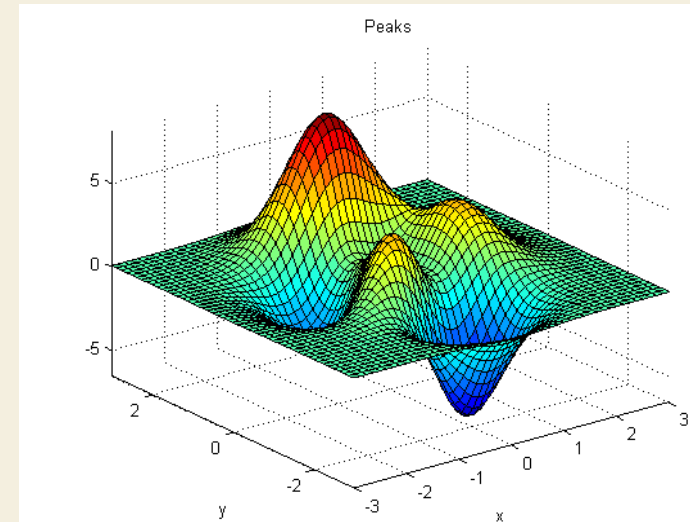
# Optimization

## optimization

/ˌɒptɪmɪˈzeɪʃ(ə)n/


*noun*

the action of making the best or most effective use of a situation or resource.  
"companies interested in the optimization of the business"




# *Evolutionary Computation*

---

- Evolutionary computation focuses on **problem solving algorithms** which draw inspiration from **natural processes**.
  - The basic rationale of the field is to adapt the **mathematical models of biological Darwinian evolution** to optimization problems.
  - The usefulness of this can be illustrated by imagining that an **organism acts as an “input” to the “model” of it’s natural environment** and produces an “output” in the form of offspring. Through multiple iterations biological evolution culls the population of organisms, only keeping the fit specimen, to produce organisms which become continuously better adapted to their environments.
- 

# *Thesis Purpose*

---

- Explore the performance and usefulness of three emerging evolutionary algorithms:
    - **Differential evolution**
    - **Particle swarm optimization**
    - **Estimation of distribution algorithms.**
  - Test them on benchmarks
    - **Mathematical** function optimization
    - **Machine learning** tasks
  - Develop a **new or modified algorithm** which improves upon the aforementioned ones in some aspect
- 

*Evolutionary Algorithms & Machine Learning*

# *BACKGROUND*

---

# *Evolutionary Algorithms*

- Works on the concept of **populations**
- A population is a set of **individuals**
- Individuals contain a vector of **parameters**

---

**Algorithm 1** Basic evolutionary algorithm

---

```
 $t \leftarrow 0$   
initialize  $P(t)$   
evaluate  $P(t)$   
while termination-condition not fulfilled do  
   $t \leftarrow t + 1$   
  select  $P(t)$  from  $P(t - 1)$   
  alter  $P(t)$   
  evaluate  $P(t)$   
end while
```

---

# *Evolutionary Algorithms: Concepts 1*

---

- **Representation**
  - Binary
  - Integer
  - Floating point
  - Tree
- **Evaluation function**
- **Population**
  - Steady state
  - Generational
- **Parent selection**
  - Fitness proportional
  - Ranking
  - Tournament





# *Evolutionary Algorithms: Concepts 2*


---

- **Variation**
  - Mutation
  - Recombination
- **Survivor selection**
- **Initialization**
- **Termination condition**




# *Traditional Evolutionary Algorithms*

---

- Genetic Algorithms
  - Evolution Strategy
  - Evolutionary Programming
  - Genetic Programming
- 

# *Emerging Evolutionary Algorithms*

---

- Differential Evolution
  - Particle Swarm Optimization
  - Estimation of Distribution Algorithm
- 

# Differential Evolution

---

**Algorithm 3** DE algorithm

---

Initialize population within bounds

Evaluate fitness of population

**repeat**

**for all** Individuals  $x$  **do**

        Select three other random individuals  $a$ ,  $b$  and  $c$

        Create mutant vector  $v = a + F(b - c)$

        Create trial vector  $u$  by blending  $x$  with  $v$  (taking at least one gene from  $u$ )

        Replace  $x$  with  $u$  if  $fitness(u) > fitness(x)$

**end for**

**until** End condition

---

- Individuals are mutated using the differences between individual vectors
- New individuals replace old individuals only if they perform better

# Particle Swarm Optimization

---

**Algorithm 4** PSO algorithm

---

Init particles with random positions  $\vec{x}(0)$  and velocities  $\vec{v}(0)$

**repeat**

**for all** Particles  $i$  **do**

        Evaluate fitness  $f(\vec{x}_i)$

        Update  $\vec{p}(t)$  and  $\vec{g}(t)$

        Adapt the velocity of the particle using the above-mentioned equation

        Update the position of the particle

**end for**

**until**  $\vec{g}(t)$  is a suitable solution

---

- Works with swarms of individuals which follow both their own “intuition” and the swarm as a whole
- Uses an update formula to accomplish variation:

$$v_{id}(t+1) = \omega v_{id}(t) + C_1 \phi_1(p_{id}(t) - x_{id}(t)) + C_2 \phi_2(g_{id}(t) - x_{id}(t)) \quad (8)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (9)$$

# *Estimation of Distribution Algorithm*

---

**Algorithm 5** UMDA algorithm

---

Initialize population P

**repeat**

    Evaluate P

    Select the best  $t\%$  of P into S

    Let  $m$  be the mean of S

    Let  $s$  be the standard deviation of S

    Sample S' from normal distribution using  $m$  and  $s$

    Create new generation of P from S' and S

**until** Termination condition

---

- Uses probability distributions to record and sample good solutions
- Different types of distributions provide different results
- The choice of distribution depends on the complexity of the problem

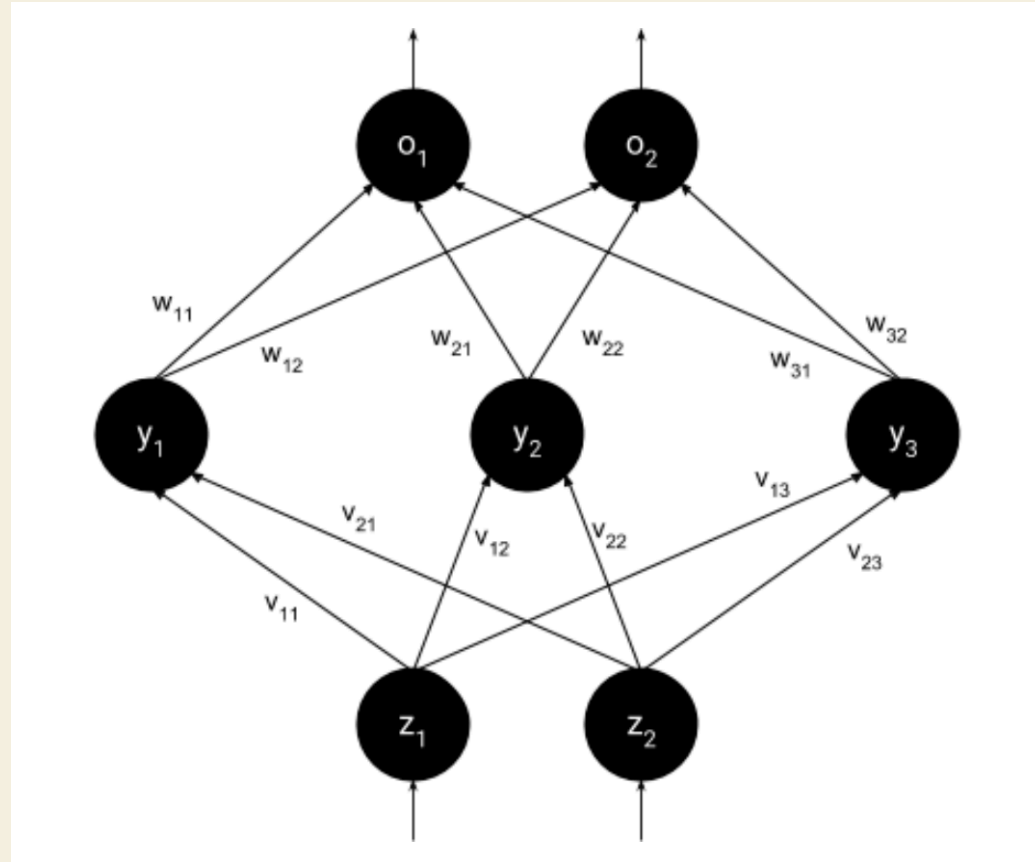
# Machine Learning Concepts

$$x_i = f(\xi_i)$$

$$\xi_i = v_i + \sum_{j=0}^n \omega_{ij} x_j$$

$$f(\xi) = \frac{1}{1 + \exp(-\xi)}$$

- Artificial Neural Networks
  - Feed-Forward Neural Networks
  - Back-propagation



*Recent Research, Emerging Evolutionary Algorithms and Contributions*

# *RELATED WORK*

---




# *Roots*

- Ideas around evolutionary computation began emerging in **1950s**
  - Several researchers, independently from each-other, created algorithms which were inspired by natural Darwinian principles, these include **Holland's Genetic Algorithms, Schwefel's and Rechenberg's Evolution Strategies** and **Fogel's Evolutionary Programming**.
  - Research has shown that **no single algorithm can perform better than all other algorithms** on average. This has been referred to as the 'no-free-lunch' and current solutions instead aim at finding better solutions to specific problems by exploiting inherent biases in the problem.
-

# *Recent Research*

---

- Parallelism
  - Multi-population models
  - Multi-objective optimization
  - Dynamic environments
  - Evolving executable code
- 

# *Research: Differential Evolution*

---

- Conceived in 1995 by **Storn and Price**
- One of **the best algorithms** in continuous optimization
- Hybrids of the algorithm
  - self-adaptive DE
  - opposition-based DE
  - DE with global and local neighborhoods
- DE uses **very few parameters** and the effects of altering these parameters have been well studied
- DE comes in a total of **10 different varieties**
  - DE/best/1/bin generally yields the best results

# *Research: Particle Swarm Optimization*

---

- **Most widely used swarm intelligence algorithm to date**
- Modified versions
  - quantum-behaved PSO
  - bare-bones PSO
  - chaotic PSO
  - fuzzy PSO
  - PSOT-VAC
  - opposition-based PSO
- Hybridized with other evolutionary algorithms

# *Research: Estimation of Distribution Algorithms*

---

- Use probabilistic models to solve complex optimization problems
- **Successful at many engineering problems which at which other algorithms have failed**
  - military antenna design, protein structure prediction, clustering of genes, chemotherapy optimization, portfolio management
- Improvements
  - Parallelization of fitness evaluation and model building
  - Local optimization techniques (deterministic hill climbing)


# *Research: Machine Learning and Evolutionary Computation*

---

- Many attempts have therefore been made to combine the two
- ML has been used to improve EC optimization algorithms
- EC has been used to improve ML techniques
  - **Using DE to optimize feed-forward neural networks**
  - Using PSO to initialize weights and biases in a neural network before training

# *Contributions*

---

- EC is a an interesting alternative to traditional approaches in machine learning and continuous optimization and while algorithms such as DE, PSO, etc. have been compared on mathematical benchmarks before, **the application of EC to machine learning has not been studied with as much detail.**
  - My primary contribution to the field will be to **find what algorithms work best for different ML problems** and based on this propose ML-specific improvements.
- 

*Benchmarks, Measurements and New Algorithms*


# *PROBLEM FORMULATION*

---



# *Research Questions*

---

- How do DE, PSO and EDA perform comparatively when applied to mathematical optimization problems?
  - How do DE, PSO and EDA perform comparatively when applied to machine learning problems such as neural network optimization and artificial intelligence in games
  - How are DE, PSO and EDA suited to these different problems?
  - Can an improved algorithm which draws inspiration from the design of DE, PSO and/or EDA outperform any of them in some/all of the aforementioned benchmarks and problem sets?
- 

# *Outcomes*

- Benchmark DE, PSO, EDA on mathematical optimization problems
  - Benchmark DE, PSO, EDA on machine learning problems
  - Develop a new algorithm inspired by DE, PSO and/or EDA
  - Benchmark the new algorithm
  - Compare the new algorithm with DE, PSO and EDA and draw conclusions from the results
- 



# *TIME PLAN*



# *Time Plan*

- The **first two weeks** will serve as an introduction to the topic and the **introduction** and **background** section of the report will be completed during this period. This will guarantee that they will be available in time for the status and planning seminary.
  - After this, deeper inquiry into the **main algorithms** of the thesis will be conducted and the first **implementation prototypes** will be constructed. During this time a **benchmark methodology** will also be decided. This phase might occupy **1 - 3 weeks**.
  - It will be followed by a **benchmark** and an investigation of how to apply the algorithms to **machine learning problems**. This will probably occupy **one week**.
  - It will be followed by the development of an **improved algorithm** and the implementation of **the machine learning benchmarks**.
  - After this a **complete benchmark** of all four algorithms on both standard functions and machine learning problems will be conducted. This should leave enough time over to fine tune the new algorithm and finalize the report before the end of the course/project period.
-