Mechatronic Engineering program

Basics of AI and Deep Learning:
1: Introduction and optimization

Ziemowit Dworakowski
AGH University of Krakow

1

Before we begin…

Lecture presentations (and all other course materials) will be available on my webpage at least 2 days before a given lecture or laboratory:

http://galaxy.agh.edu.pl/~zdw/students.html

I recommend making notes only for stuff that is **NOT** on the slides. During lectures just focus on understanding relations and thought process. Knowledge will be easier to memorize after you see ***the big picture***

This is the first time that this course is offered, most of it is an on-going experiment. Your feedback is important – if you feel something is not going as it should, notify me as soon as possible, don't wait till the end of semester.

2

What this course is all about?

We will show you how AI methods work

We will show you how AI methods work in engineering

We will teach you about processing of real-life data (including time-domain signals and images)

We will show you what is state-of-the-art in AI

We will teach you concepts that will allow you to understand AI now and in the future

… and how to start designing your own decision systems

… and you will be able to decide if working with information interests you and is worth pursuing in the future

3

Course team

Dr hab. inż. **Ziemowit Dworakowski,** zdw@agh.edu.pl
(course coordinator for MEI)
Office hours: D3, 4.17, Tuesday, 9:00 AM – 11:00 AM
Area of expertise:
AI, decision systems, optimization signal & image processing, vision systems

Dr inż. **Krzysztof Holak,** holak@agh.edu.pl
(course coordinator for IME)
Office hours: D3, 4.04, Thursday, 9:00 AM – 11:00 AM
Area of expertise:
vision systems, image processing, signal processing

Mgr inż. **Adam Machynia,** machynia@agh.edu.pl

Office hours: D3, 4.08, Thursday, 10:00 AM – 11:30 AM
Area of expertise:
AI, image processing, signal processing

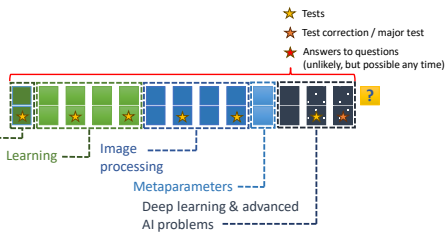
Please notify respective teacher via email at least a day before you plan to use office hours.

4

So what exactly are contents of this course?

We have:

14 lectures
13 laboratories



5

OK, but how will we get a grade?

- Passing all laboratories is required (each results with a grade)
- Passing all the tests is required

Final grade = average of all the grades (simple!)

6

Laboratories will use „flipped classroom" approach:

- You prepare for labs at home
  (I recommend doing tasks marked as „for 3.0")
- The more tasks you do during classes, the better your grade is
- You won't need to prepare report, provided that you manage to show at least all the „3.0" tasks during the laboratory **and** all the tasks marked as necessary (*) during the **next** laboratory.

(Please check the first instruction for more details)

7

What if something goes wrong?

If you don't pass a laboratory (either due to lack of preparation or due to absence) – you prepare a standard report with all the normal tasks plus one additional task selected by the LA. You may do so three times in total (after that passing conditions will be set individually.).

At the end of the semester there will be a chance to correct any three (out of five) tests. More lacks will require writing a major test on the scope of the entire semester.

Note that my webpage contains a document on report writing (including laboratory and project reports)

8

Do we have an elephant in the living room?

a.k.a. what about ChatGPT (and all other genAI models you might use)?

What is mandatory:

- Acknowledgement of form of usage if it is used at home (e.g. *„report was rewritten for clarity using ChatGPT")*
- Taking responsibility for your codes and reports (so if any part of it was prepared using genAI, you still are required to understand it and explain it as if it was your own work)

What is not allowed:

- Usage of genAI in any form in class unless the teacher explicitly allows that
- Usage of genAI for rewriting codes

What you can do:

- Use genAI to correct codes at home (bug fixes only)
- Use it to prepare fragments of reports from prompts or drafts (*note, you should store these prompts or drafts, teacher might ask for them*)
- Use genAI to correct your language

Why?

Because laboratories are already providing you with almost-complete working codes.
The idea behind the whole course is for you to understand what is going on in these codes and use it to your advantage.

9

Real-life problem

- *How to operate an airline to maximize profit?*
- *How to design a car so it is durable, safe and quiet?*
- *How to spend my time at the University so to get as much value as possible?*

Problem description → *Parametrization* → **Parameters**

They say: what can we change in our current setup?

x1
x2
x3
x4
…
xn

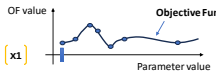
Which parameter values are „the best"?

Optimization means finding a **global optimum** of the **objective function** defined in a **parameter space**

Optimization

10

How to answer such questions?

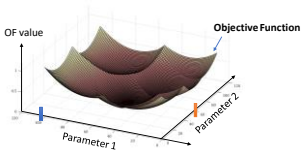


Lets simplify to just one parameter: x1

It is like we'd move a slider trying to find the best setup

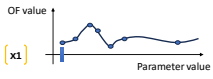
For two parameters, it works the same: x1, x2

We just have two sliders to work with



11

How to answer such questions?



Lets simplify to just one parameter: x1

It is like we'd move a slider trying to find the best setup

For two parameters, it works the same: x1, x2

We just have two sliders to work with

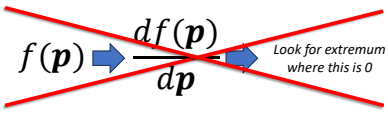
If we have more than two…? x1, x2, x3, x4, …

No differences! We just can't really see the underlying objective function, that's all!

So how would it look for our „real-life problem"?

12

How to answer such questions?



In typical optimization tasks we usually don't know the equation for the objective function.

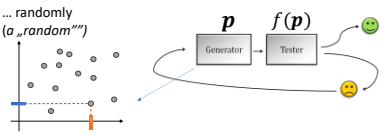
That is why the only way to solve the problem is to do it using a trial-and-error approach

13

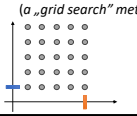
How to answer such questions?

We can generate candidates to check…

… randomly
(a „random"")



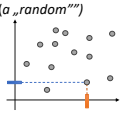
… in an organized way
(a „grid search" method)



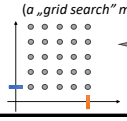
14

How to answer such questions?

… randomly
(a „random"")



… in an organized way
(a „grid search" method)



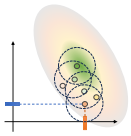
Provided that there are not many dimensions to work with (Not many „sliders")

Does not work **at all** if numer of parameters to setup is higher (roughly speaking: higher than 3?)

Here we have 25 „tries" (5*5)
5 per dimension is rather low, but still…
3D -> 125
4D -> 625
10D -> 9 765 625 tries…
It is impossible to densely fill the parameter space with solution candidates…

15

A 1+1 method



Concept:

Lets start with any point

Lets check its value

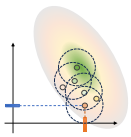
If it is best so far, lets keep it

Lets generate a new point in the neighborhood of the kept point

And this way we are gradually moving towards the best region!

16

A 1+1 method



Concept:

Lets start with any point

Lets check its value

If it is best so far, lets keep it

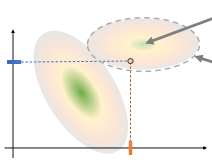
Lets generate a new point in the neighborhood of the kept point

And this way we are gradually moving towards the best region!

or… are we?

17

A 1+1 method



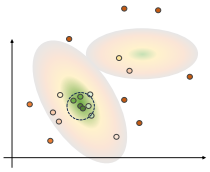
This is called a **local minimum** **(that is: a point that has the best value in its neighborhood)**

But if we start optimization in the **region of attraction** of the local minimum, we won't find the global one…

So… how to solve this problem?

18

A 1+1 method



Algorithm:

Start with randomly picked point candidate $\boldsymbol{p}_c$

Check its objective function value: $f(\boldsymbol{p}_c)$

If it is best so far, keep it: $\boldsymbol{p}_b = \boldsymbol{p}_c$

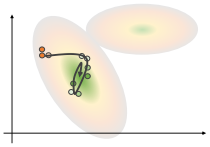
Generate a new point candidate in the neighborhood of the kept point: $\boldsymbol{p}_c = \boldsymbol{p}_b + S \cdot \boldsymbol{r}$
(S stands for size of neighborhood, $\boldsymbol{r}$ is a random vector)

** If no progress for a predefined time, reduce S*
** Stop the method when S is very small and we are not improving any more*

- **Good scalability to mid to high-parameter optimization spaces**
- **Relatively easy to setup**
- **Is working for a non-continuous optimization spaces as well**

19

A gradient method



Concept:

Lets start again with any point

Lets „Feel the ground around" to check the function falls the fastest

Make a big step in the direction of where function is the steepest

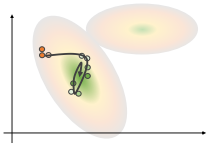
We will find be directed towards the minimum for the whole time – but we might oscillate around it…

So we can again reduce step if no big improvement is made!
We can even make it more inert by adding ***momentum*** *– to allow it gain speed on the slope*

*And since it will be attracted by the local minima, we can repeat it a few times (****multistart****) and get the best result*

20

A gradient method



Algorithm:

Start with randomly picked point candidate $\boldsymbol{p}_c$

Calculate gradient of the objective function around point candidate: $\nabla f(\boldsymbol{p}_c)$

Make a new point by moving towards the steepest gradient ascent (descent) : $\boldsymbol{p}_n = \boldsymbol{p}_c + (-) S \cdot \nabla f(\boldsymbol{p}_c)$
It is our new candidate to consider: $\boldsymbol{p}_c = \boldsymbol{p}_n$

- If no progress for a predefined time, reduce S
- Repeat for a predefined number of iterations or until gradient is non-zero

(you may start the method multiple times from new points and keep the best result as your answer)

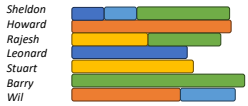
- **Good scalability to very high-parameter optimization spaces**
- **Relatively easy to setup**
- **It requires a continuos optimization space to work**

21

Pro-tip of the day

It may seem wrong to NOT start the gradient multiple times always. But apparently, in many practical optimization problems the more dimensions we have – the more likely it is that we have just one (global) minimum, or if there are more, they are also relatively similar in depth…

Imagine assigning tasks for people that do a major project together:



The more people we have, the more flexible we are in the assignments!

22

***Representation 1**:*
A vector of proportions of ingredients

There's a tradeoff here:
The more flexible and complex the representation is, the better the results can potentially be.
But it is harder to actually converge to them.

x1 – water added [liters]
x2 – flour amount [grams]
x3 – yeast [grams]
x4 – seeds [grams]
…

It is simpler to build a „solver" here and we might reach the optimum quickly. But the optimum in this simplified reality might not reflect the actual optimum

***Representation 2**:*
A process with inputs' description

x1 – Action: add flour
x2 – Volume: k grams
x3 – Action: add yeast
x4 – Volume: k grams
…

There are many parameters to take into account. In optimization we might loop, backtrack, get attracted towards local minima… After all we might not be able to reach any minimum at all

23

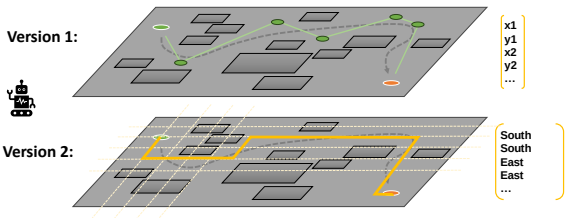
Representation design in practice

You might want to consider:

- ***What can you actually control?***
  (Don't bother with variables you can't control in the first place)
- ***What do your expert context knowledge say?***
  (In typical situations there's someone who can estimate what are reasonable approaches to start with)
- ***How much time and data can you commit to the task?***
  (the less data and time you have, the simpler the representation needs to be)
- ***How difficult is an objective function value test***
  (so: how long it takes to go from parameters' values to estimation of the solution quality)

24

Let's do an example together



25

Objective function can be:

- One criterion or **multi criterion**

This means we have more than one goal of optimization
(e.g. design a car that is as safe as possible while also being as cheap as possible)

1) We can merge all criterions together using weighted average

$$Q = W_1 \cdot \frac{1}{C} + W_2 \cdot \frac{1}{S}$$

2) We can use **Pareto Front**

So we look for a set of diverse solutions and then compare them discarding those that are objectively worse (in terms of all criteria).

26

Objective function can be:

- One criterion or **multi criterion**

- Continuous or **non-continuous**

This means that all parameters can be changed with any arbitrary accuracy, i.e. we can make as small steps as we want
(e.g. you can change car's length with any accuracy, but you can't add a 2/3 of a chair to a car)

Non-continuous parameter space means that we can't use gradient-descent-based approaches

27

Objective function can be:

- **One criterion** or **multi criterion**
- **Continuous** or **non-continuous**
- **With constraints** or **without constraints**

This means that any parameter can be assigned any value in a given range

This means that there are additional conditions or constraints that the parameters need to satisfy – *e.g. If we design a travel trajectory we can't go through highway by foot and we can't go through park if we go by car.*

We can either design a representation that does not allow for „illegal" solutions, or „make them legal" after they are found

28

Things to remember:

1. Give optimization definitione and some practical examples (including multi-criterion, multi-parameter, with or without constraints, in continuous or non-continuous parameter space)
2. Explain why representation choice is important – and what are the risks associated with this step
3. Explain the *random* and *grid search* optimization methods, show their limitations
4. Explain the algorithm, and features of 1+1 algorithm, illustrate it
5. Explain the algorithm, and features of gradient algorithm, illustrate it. Explain multistart and momentum additions.
6. What are local minima and how can we approach them?

29