uc3m | Universidad Carlos III de Madrid

Master in Statistics for Data Science
2024-2025

Master Thesis

"Optimizing Wind Turbine Placement in Wind Parks via Mixed Integer Optimization using Neural Network based Constraint Learning."

Simon Schmetz

Carlos Ruiz Mora
2nd Tutor complete name
Madrid the 29. of January

Acknowledgments

TODO Acknowledgments

ABSTRACT

This thesis combines Linear Optimization with Constraint Learning to optimize wind turbine placement for maximum performance in a predefined area under randomly distributed wind. A Neural Network is trained on simulated data to model the impact of turbine positioning on power output. This model is integrated as a constraint in a linear optimization problem, and the problem evaluated for a current state-of-the-art wind farm configuration.

CONTENTS

1

INTRODUCTION

With the clean energy transition currently taking place in europe with ambitious targets for 2030 and beyond [4] , wind energy is playing a central role in that transition, with wind energy expected rise to 50 % in the EU energy mix. [3] With wind energy thus expected to become the main contributer to the EU's energy production and large potentials identified for both onshore and offshore parks [1] attempts to optimize all parameters of windparks with even minor power efficeny improvement can be expected to yield significant returns in absolute power due to the scale of future wind energy production.

As a contribution to increasing power efficeny on future wind farms, this thesis is dedicated to a new approach for optimizing the placement of a fixed number of wind turbines in a predefined area (typically a square). To solve this optimization problem, a extension to the pyomo python library is used, that allows the introduction of Neural Networks to the optimization problem as constraints. [2] This extension allows for introducing a Neural Neural Network to model the effects of wind turbine placement relative to each other on power production for the respective windturbines. Introducing this model to the optimization problem defined in pyomo then allows for the optimization of overall power productions across all wind turbines in the wind park. The optimization problem in its simplest form can be defined as

$$\max_{\mathbf{x},\mathbf{y}} \sum_{i=1}^{n} f_{Power,\mathrm{NN}}(x_i, y_i; \mathbf{x}, \mathbf{y}) \tag{1}$$

$$\text{s.t.} \quad X_{\min} \leq x_i \leq X_{\max}, \quad \forall i \in \{1, \ldots, n\} \tag{2}$$

$$Y_{\min} \leq y_i \leq Y_{\max}, \quad \forall i \in \{1, \ldots, n\} \tag{3}$$

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq d_{\min}, \quad \forall i \neq j \tag{4}$$

where:

- (x_i, y_i) are the coordinates of turbine i out of n total turbines,
- $f_{\mathrm{NN}}(x_i, y_i; \mathbf{x}, \mathbf{y})$ is a neural network approximating the power output for each turbine i ,
- $X_{\min}, X_{\max}, Y_{\min}, Y_{\max}$ define the boundaries of the rectangular placement
- $d_{\min}$ is the minimum distance between any two turbines.

To create a model optimally fit to the needs of the optimization problem, the model is trained on data specifically generated with the FLORIS wind farm simulation tool for optimal coverage of the parameter space of the optimization.

To simplify the problem, the surface below the turbines is assumed to be perfectly flat and equal wind speed is assumed along the entire hight of the turbines.

Data Generation and Neural Network:

(...)

Optimization Problem:

(...)

This thesis is structued according to the two main steps required to solve the optimization problem as presented above.

2

STATE OF THE ART

3 FARM POWER MODEL

With the goal of training a model tailor made to the requirements of the optimization problem, the data used for training the model has to generated using a open source simulation method, allowing to set the parameter space of the model as required to the given optimization problem. After investigating the two python based wind farm simulation tools FLORIS and PyWake, FLORIS was chosen due to solid documentation, what appears to be very stable releases and broad functionalities regarding wake modelling. FLORIS is a wind farm simulation tool developed by the National Renewable Energy Laboratory (NREL).

Wind turbine power curve modelling under wake conditions using measurements from a spinner-mounted lidar: https://www.sciencedirect.com/science/article/pii/S

Floris: https://github.com/nrel/floris

Main Effects of Wake Turbolence: - Wind Speed (needs time to mix with outside airflow) - Turbolence

3.1 DATA SOURCE

3.2 MODELLING

To model the relationship between the attributes of a incoming airflow to a wind turbine and the output generated by the same wind turbine many surrogate models could be chosen. As the model generated in this for this thesis is created with the goal of introducing it into the pyomo extension referenced in Chapter 1, the model has to be compatible with said extension. That is why the model chosen is a simple neural network with limited number of hidden layers and nodes, with the size of the Neural Network being limited for said extension.

3.2.1 *Neural Networks and Training of Neural Networks*

In the following section, a biref introduction is given into how Neural Networks work and are trained. Fundemanetally, Neural Networks represents Graph Networks consisting out of Function blocks as the nodes, in the case of Neural Networks called perceptron (or Neurons as more general terms) and arcs which correspond to the inputs/outputs of a given perceptron. In simple words, the inputs to the Neurons are summed up and introduced as argument into a function $f()$ which yields a output y, representing the output of the given perceptron. These Neurons are organized in layers, which correspond to the row type structure Neural Networks are usually represented in and each perceptron of one layer is connected to every

other perceptron of the following layer. The following layer in this case means in the direction of flow, in visualizations usually from left to right. The arcs thus in simple terms correspond to a single number and the Neurons to the action of summing up all inputs and then applying the unspecified function $f()$ to generate a output. This process is repeated for all Neurons in the network, with the first layer of the network taking as input the values of the values of the given data features (the input values to the model) and the last layer producing outputs corresponding to the output value(s) of the model. The number of Neurons in this first and last layer correspond to the task the Neural Network is supposed to perform. If the goal is to identify handwritten numbers 1-9 from 50x50 pixel images, the imput layer might have 50x50 Neurons for the value of each pixel, and the output layer 9 layers with the output value of each perceptron representing how much the networks thinks the given image shows the corresponding numbers 1-9. A schematic of this architecture is given in Figure 1.

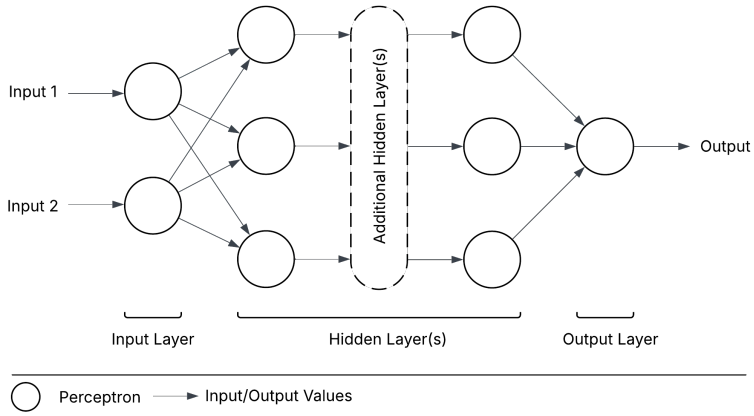

Figure 1: Schematic of Neural Network Architecture

As shown in Figure 2 the output of a Neuron is slighlty more complicated as described befoe, as the output y is generated by summing up the inputs x multiplied by a corresponding weight w together with a bias b and introducing this summation as argument into a activation function $f()$. In this process, the weights w represent a weigh to give importance to the individual inputs and the bias b serves to set a minimum output value that will always be reached, regardless of the inputs.

The activation function $f()$ is called activation function, as in its most simplest form it represents a step function that decides if a neuron activates or not, e.g. takes the binary values $0, 1$ for a given threshhold. Contrary to the human brain where neurons are indeed binary, most Neural Networks resort to a activation function whose outputs are not binary but deliver continious values between 0 and 1 to avoid the boundary issues that occour with binary threshholds. The most common function used instead of a step function is the sigmoid function, which roughly corresponds to a continuous version of the step function, with $\sigma(x) \approx 1$ for $x \to \infty$ and $\sigma(x) \approx 0$ for $x \to -\infty$.

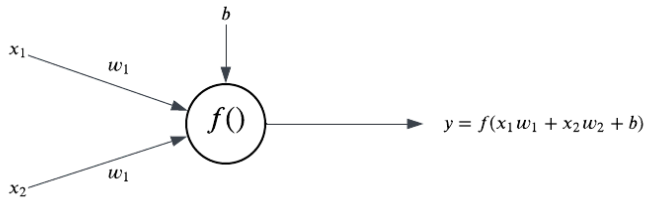

Figure 2: The output of a neuron is generated by applying the activation function to the sum as the weighted inputs $w_i x_i$ and the bias of the neuron b

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

This relationship also becomes aparent when plotting both of those functions over each other, as shown in 3

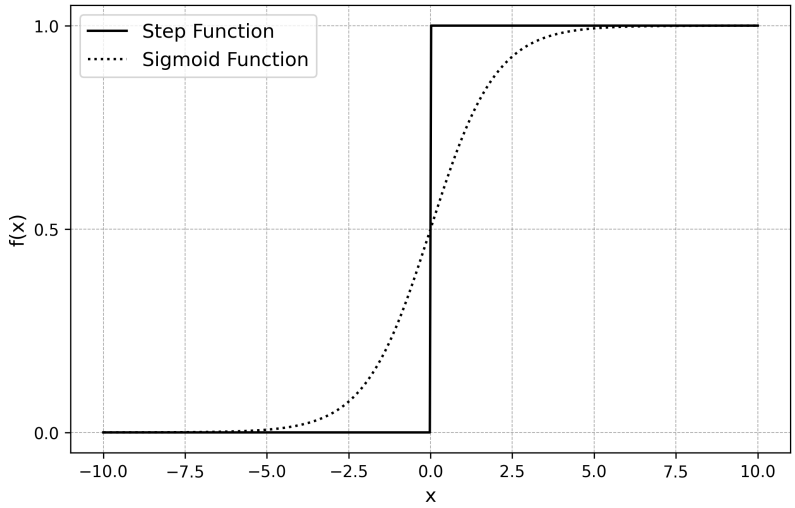

Figure 3: The Sigmoid Function being a close to the Step Function without being as sensetive to slight changes in x due to its continuity

With the general structure set up, and assuming that a layout has been chosen for the neural network (e.g. number of layers and number of their coirresponding neurons), the training of the neural network corresponds to adjusting the weights w and the biases b of each neuron in a way, that allows the model to perform the task it is given well. What is means for the model to perform well is defined by a *Loss Function*, that defines a relationship between the output of the model and the correct output (defined by training data) and gives a Loss as output, which is some sort of delta between model prediction and truth. As what it means for a model to perform well heavily depends on the task (regression, binary classification, multiclass classification etc.), but regardless of the task, the goal is always to minimize the Loss Function. A well known Loss Function in regression is the Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

The training of the Neural Network thus corresponds to adjusting the weights and biases in a way that minimizes the chosen loss function. The algorithm most commonly used for this is called *Backpropagation*.

many Degrees of Freedom, many local maxima

3.2.2 *Generating Neural Network Model for two Windturbines*

3.3 VALIDATION

4

OPTIMIZATION

5

ADDENDA

6 ABBREVIATIONS

MSE Mean Squared Error (MSE)

LIST OF FIGURES

LIST OF TABLES

BIBLIOGRAPHY

[1] European Environment Agency. Europe's onshore and offshore wind energy potential. Technical Report Technical report 6/2009, European Environment Agency, 2009. Accessed: 2025-04-03.

[2] Antonio Alcántara and Carlos Ruiz. A neural network-based distributional constraint learning methodology for mixed-integer stochastic optimization. *Expert Systems with Applications*, 232:120895, 2023.

[3] Analysis and General Secretariat of the Council of the European Union Research Team. Harnessing wind power: Navigating the eu energy transition and its challenges. https://www.consilium.europa.eu/media/1kyk0wjm/2024_685_art_windpower_web.pdf, September 2024. Accessed: 2025-04-03.

[4] European Commission. Renewable energy targets. https://energy.ec.europa.eu/topics/renewable-energy/renewable-energy-directive-targets-and-rules/renewable-energy-targets_en, 2023. Accessed: 2025-04-03.