On Evolutionary Approaches to Wind Turbine Placement with Geo-Constraints

Daniel Lückehe
Department of Geoinformation
Jade University of Applied Sciences
Oldenburg, Germany
daniel.lueckehe@uni-oldenburg.de

Markus Wagner
Optimisation and Logistics
University of Adelaide
Adelaide, Australia
markus.wagner@adelaide.edu.au

Oliver Kramer
Department of Computing Science
University of Oldenburg
Oldenburg, Germany
oliver.kramer@uni-oldenburg.de

ABSTRACT

Wind turbine placement, i.e., the geographical planning of wind turbine locations, is an important first step to an efficient integration of wind energy. The turbine placement problem becomes a difficult optimization problem due to varying wind distributions at different locations and due to the mutual interference in the wind field known as wake effect. Artificial and environmental geological constraints make the optimization problem even more difficult to solve. In our paper, we focus on the evolutionary turbine placement based on an enhanced wake effect model fed with real-world wind distributions. We model geo-constraints with real-world data from OpenStreetMap. Besides the realistic modeling of wakes and geo-constraints, the focus of the paper is on the comparison of various evolutionary optimization approaches. We propose four variants of evolution strategies with turbine-oriented mutation operators and compare to state-of-the-art optimizers like the CMA-ES in a detailed experimental analysis on three benchmark scenarios.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Constrained optimization; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

Keywords

Wind Power, Wind Farm Layout, Evolutionary Optimization, Self-Adaptation, CMA-ES

1. INTRODUCTION

Renewable energy plays an increasing role in the energy supply world-wide. The integration of geo-information into planning processes becomes more and more important, resulting in a complex geo-planning problem with many constraints. In particular, the effective integration of wind energy is significantly influenced by the locations of wind turbines. Wind turbine placement optimization is the process of determining the placement of wind turbines within a specified land or offshore area, such that the output power of the wind farm is maximized. An optimized placement result allows wind farm installations to maximize their cost-effectiveness, thereby increasing their competitiveness in the energy market. However, the optimal placement of wind turbines considering geo-constraints is a complex optimization problem that is hard to solve by exact methods. The solution space is non-linear with respect to how sited turbines interact when considering wake loss and energy capture, in particular when facing constraints. In this paper, we employ evolutionary heuristics to solve the turbine placement problem.

Objective of this paper is to model a realistic wind turbine placement scenario by taking into account realistic wake effects, real-world wind data from a meteorological service, and geo-constraints from a topological map service. On the basis of this model, we define three benchmark scenarios and analyze turbine-oriented heuristics while comparing to state-of-the-art optimization algorithms like the covariance matrix adaptation evolution strategy (CMA-ES).

This paper is structured as follows. In Section 2, we present related work on turbine placement with evolutionary algorithms. Our enhanced wind model based on COSMO-DE data and Kusiak and Song's wake effects [10] is introduced in Section 3. In Section 4 we describe how OpenStreetMap data is modeled as geo-constraints. The evolutionary heuristics we employ for optimization are introduced in Section 5, followed by an experimental analysis on three benchmark scenarios in Section 6. Conclusions are drawn in Section 7.

2. RELATED WORK

For a very comprehensive overview on the single-objective wind farm layout problem, we refer the interested reader to the recent article by Herbert-Acero *et al.* [8]. For example, Wan et al. [19, 20] use cell-based approaches and compare different bio-inspired algorithms, each applied to the same set of wind farm models and parameters. An alternative to cell placement was explored in [10]: each turbine's location is a decision variable pair of real-valued, spatial (x, y) coordinates. In that paper, a simple evolution strategy (ES) [3] is used to optimize very small wind farms. In general, an ES is effective because it is easily parallelized and it self-

GECCO '15, July 11 - 15, 2015, Madrid, Spain

DOI: http://dx.doi.org/10.1145/2739480.2754690

adapts the extent to which it perturbs decision variables when generating a new potential solution. The more powerful CMA-ES has been used for the same problem in [18]. Despite being computationally expensive, it allows for the effective optimization of huge layouts for up to 1000 turbines. In [17] a random local search was presented that combines a problem-specific operator with an asymptotic speed-up of the computation time of the wake effects. Compared to the CMA-ES approach, the problem-specific local search produced higher quality placements in shorter time.

Many of the recent approaches make simplifying assumptions regarding the realism of the wake models. Even in some of the recent work, wakes are either completely ignored, and e.g., assumed to be constant or highly simplified. Nevertheless, we observe a trend towards the use of more realistic models, such as the Jensen [15] wake model. There, the wake effects decrease quadratically with the distance between the turbines. However, to make this model computationally viable, the model is simplified. For example, Kusiak and Song [10] suggest to discretize the wind directions. Not only are the resulting predictions less accurate, the model introduces many local optima which make the optimization an unnecessarily taxing task. To the best of our knowledge, this issue has not been reported in literature before, which is why we highlight and solve this later-on.

3. WIND MODEL

In this section, we introduce the employed wind model. For a given vector

$$\mathbf{x}_i = (x_1, x_2, \cdots, x_N) = (x_1^t, y_1^t, x_2^t, y_2^t, \cdots, x_{N/2}^t, y_{N/2}^t) \quad (1)$$

of $N/2$ turbine locations $\mathbf{t}_i = (x_i^t, y_i^t)$ with the length N, the model yields the power output of the corresponding setting with wind turbines. The model is based on the approach from Kusiak and Song [10] and our objective is to create a more realistic representation. For this purpose, we use power curves from a real wind turbine by Enercon and wind data from the German weather prediction model COSMO-DE by the German Weather Service.

3.1 Wind Power Calculation

In order to compute the power output E of a turbine i at position $\mathbf{t}_i = (x_i, y_i)$ for a given wind direction θ, the wind speed distribution $p_v(v, k(\mathbf{t}_i, \theta), c(\mathbf{t}_i, \theta))$ has to be multiplied with the turbine power curve $\beta_i(v)$, where v is the wind speed. The Weibull parameter $k(\cdot)$ and $c(\cdot)$ are functions depending on position and wind direction. For one wind direction it applies:

$$E_\theta(\mathbf{t}_i, \theta) = \int_0^\infty \beta_i(v) \cdot p_v(v, k(\mathbf{t}_i, \theta), c(\mathbf{t}_i, \theta))dv. \quad (2)$$

The output E for one turbine i considering all wind directions is:

$$E(\mathbf{t}_i) = \int_0^{360} p_\theta(\mathbf{t}_i, \theta) \cdot E_\theta(\mathbf{t}_i, \theta)d\theta \quad (3)$$

with the distribution $p_\theta(\cdot)$ of wind angles depending on position and wind direction. We solve these integrals like Kusiak and Song with a combination of an analytical solution and with the Riemann sum, because a purely analytical form is difficult to obtain [10]. This causes a discretization of the wind speed and the wind direction. Using the existing solution for this integral leads to a linear power curve $\beta_i(v)$ for the wind turbine:

$$\beta(v) = \begin{cases} 0 & v < v_{\text{cut-in}}, v_{\text{cut-out}} \leq v \\ \lambda \cdot v + \eta & v_{\text{cut-in}} \leq v < v_{\text{rated}} \\ P_{\text{rated}} & v_{\text{rated}} \leq v < v_{\text{cut-out}} \end{cases} \quad (4)$$

While $v_{\text{cut-in}}$ is the wind speed the turbine starts to produce power, v_{rated} denotes the minimal wind speed to get the maximum power from the turbine, and $v_{\text{cut-out}}$ is the speed when the turbine switches off for safety reasons. In our paper, we use the parameters of the Enercon E101 wind turbine, which we determined from the manufacturer's datasheet [5]: $\lambda = 277.0\,\text{kW·s/m}$, $\eta = -551.6\,\text{kW}$, $P_{\text{rated}} = 3050\,\text{kW}$, $v_{\text{cut-in}} = 2\,\text{m/s}$, $v_{\text{rated}} = 13\,\text{m/s}$, $v_{\text{cut-out}} = 28\,\text{m/s}$.

3.2 Wind Analysis Data

We use the COSMO-DE analysis data from the German Weather Service [4] as the basis for the wind potential prediction for a position $\mathbf{t}_i$. This data provides a rotated latitude / longitude grid over Germany with $419_{\text{lat}} \cdot 459_{\text{lon}} = 192,321$ points. The distance between every grid point is approximately 2.8 km. Between the points, we will use a bilinear interpolation. For every data point, there are 50 different levels of height. As we use one turbine with the height of 135 m, we linearly interpolate from the model levels 46 and 47 with the height of approximately 180 m and 120 m – the levels in the COSMO-DE model are not at a fixed height.

In this work, we use the data of COSMO-DE 2012, which provides hourly wind vectors. We sort these vectors according to their wind direction θ and fit a Weibull distribution [21] for every wind direction θ. We employ the maximum likelihood estimation for fitting the distribution. The result is a Weibull function $p_v(v, k(\mathbf{t}_i, \theta), c(\mathbf{t}_i, \theta))$ representing the wind distribution for every position $\mathbf{t}_i$ and angle θ.

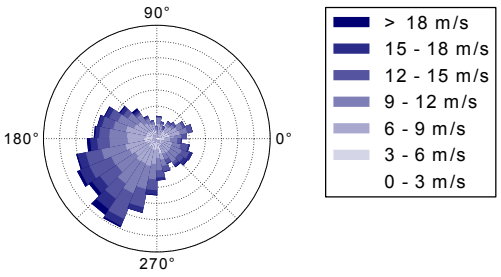


Figure 1: Example of a wind distribution at a location in Lower Saxony, Germany.

Figure 1 shows an exemplary wind rose that illustrates a wind distribution as an example based on the COSMO-DE data in decimal degrees at $53.410°, 8.074°$ and with height level 46 in Lower Saxony, Germany. The wind rose illustrates the distribution of wind directions (from where the wind comes at corresponding angle), and their frequency (represented by the length of the bars), and the corresponding wind speed (represented by color). The detailed wind data is employed for an accurate wake model that will be introduced in the following.

3.3 Wake Model

Like Kusiak and Song [10], we use the Jensen [15] wake model. Figure 2 shows the structure for one turbine.

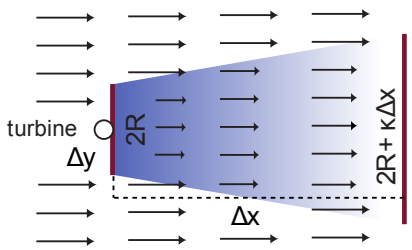


Figure 2: Structure of a single wake effect.

In the following, we describe the method to determine, if a turbine i at position $\mathbf{t}_i$ is affected by the wake effect of a turbine j, according to Kusiak and Song [10]. First, we need to compute the angle β_{ij} between the turbines:

$$\beta_{ij} = \cos^{-1} \frac{(x_i^t - x_j^t) \cdot \cos\theta + (y_i^t - y_j^t) \cdot \sin\theta + R/\kappa}{\sqrt{\left((x_i^t - x_j^t) + \frac{R}{\kappa} \cdot \cos\theta\right)^2 + \left((y_i^t - y_j^t) + \frac{R}{\kappa} \cdot \sin\theta\right)^2}} \quad (5)$$

where θ is the angle of the wind direction, R is the rotor radius of turbine j, κ is the wake spreading factor, and $x_i^t, x_j^t, y_i^t, y_j^t$ are the positions of the turbines i and j. Turbine i is affected by the wake effect of the turbine j when $0 \leq \beta_{ij} \leq \arctan(\kappa)$. A detailed description of this model can be found in [10]. We use a different specification by simply describing the three lines which limit the wake effect. By using the symmetry around the x-axis, we can simplify the description as follows. Turbine i at position $\mathbf{t}_i$ is affected by the wake effect of turbine j if:

$$\Delta x^t > 0 \wedge \left(\left|\Delta y^t\right| - R\right) / \Delta x^t < \kappa \quad (6)$$

with $\Delta x^t = \cos\theta \cdot (x_i^t - x_j^t) + \sin\theta \cdot (y_i^t - y_j^t)$ and $\Delta y^t = \cos\theta \cdot (y_i^t - y_j^t) - \sin\theta \cdot (x_i^t - x_j^t)$. The Equations 5 and 6 characterize the same behavior, but Equation 6 is computationally more efficient and can be modified more easily, what we show in the following. The velocity deficit through a wake effect is described in the Jensen model as follows:

$$\text{Vel\_def}_{ij} = \left(1 - \sqrt{1 - C_T}\right) / \left(1 + \kappa \cdot \Delta x^t / R\right) \quad (7)$$

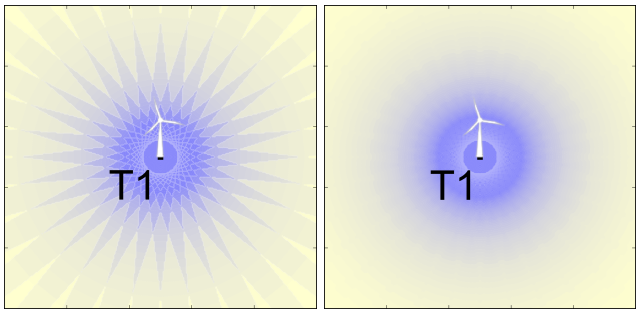
and the summation of multiple wake effects of $N/2$ turbines with:

$$\text{Vel\_def}_i = \sqrt{\sum_{j=1, j\neq i}^{N/2} (\text{Vel\_def}_{ij})^2}. \quad (8)$$

Like in [10], we use $\kappa = 0.075$ and $C_T = 0.8$.

3.3.1 Local Optima Through Wind Discretization

The discretization of the wind direction, see Section 3.1, causes local optima in the solution space, which makes the optimization unnecessarily difficult. We will demonstrate and remove this issue in the following. Figure 3 shows the wake effects for one turbine with 32 wind directions that are equally distributed. In a real-world setting with equally distributed wind, we expect low wind potential next to the turbine and higher potential with increasing distance because the wake effects decrease. Although the wind is equally distributed, see Figure 3(a), there may occur multiple local optima on the potential map. Yellow stands for high potential, while blue represents low potential. This is due the fact that the wake effects are sampled for every direction, there is no smooth transition between the different directions.



(a) before (b) after

Figure 3: Visualization of a wake effect: (a) the 15 degrees apart model as used by Kusiak and Song induces local optima, while (b) our approach leads to a smooth function.

We will add a smooth transition between the different directions into the model by modifing Equation 6 to:

$$\Delta x^t > 0 \wedge \left(\left|\Delta y^t\right| - R\right) / \Delta x^t < \kappa + \theta_{\text{inc}} \quad (9)$$

with $\theta_{\text{inc}} = \arctan \Delta\theta$ and $\Delta\theta$ is the angle increment between the wake effects. And we will change Equation 8 to:

$$\text{Vel\_def}_i = \sqrt{\sum_{j=1, j\neq i}^{N/2} (m \cdot \text{Vel\_def}_{ij})^2}. \quad (10)$$

with

$$m = \begin{cases} 1.0 & \text{if } \Delta x^t > 0 \wedge b \leq \kappa \\ 1.0 - (b - \kappa/\theta_{\text{inc}}) & \text{if } \Delta x^t > 0 \wedge b > \kappa \wedge b < \kappa + \theta_{\text{inc}} \\ 0.0 & \text{otherwise} \end{cases}$$

with $b = \left(\left|\Delta y^t\right| - R\right) / \Delta x^t$. Figure 3(b) shows the wake effects with the modification. Now there are no more local optima.

3.3.2 Rotor Size of the Affected Turbine

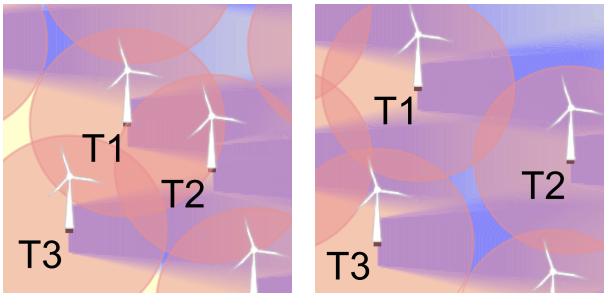
Our model is able to calculate how a point is affected by wake effects. For our optimization, however, we do not want to know the effect on a single point but for an entire turbine with a rotor dimension. We include this fact into the model by adding the rotor size of the affected turbine to the equations. Thus, Equation 9 changes to:

$$\Delta x^t > 0 \wedge \left(\left|\Delta y^t\right| - R_i - R_j\right) / \Delta x^t < \kappa + \theta_{\text{inc}} \quad (11)$$

where R_i is the rotor size of the affected turbine i and R_j is the rotor size of the turbine j, which is causing the wake effect. Also, we change b in Equation 3.3.1 to

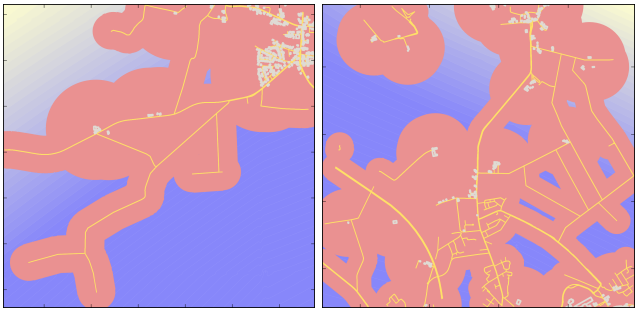
$$b = \left(\left|\Delta y^t\right| - R_i - R_j\right) / \Delta x^t. \quad (12)$$

Figure 4 compares the result of two exemplary optimization runs (a) not considering the turbine rotor size, and (b) considering the rotor size. As we can see, T2's rotor is partially influenced by T1's and T3's wakes in 4(a), while this is not the case in 4(b) where we consider the rotor size.



(a) Rotor size not considered (b) Rotor size considered

Figure 4: Two sectors of a larger map (a) without and (b) with consideration of the affected turbines' rotor size.



(a) Scenario 2 (b) Scenario 3

Figure 5: Illustration of benchmark Scenarios 2 and 3.

4. GEOGRAPHICAL DATA

Besides a detailed wind model, we consider data from OpenStreetMap [6] for a realistic geo-planning scenario. The geo-information adds constraints to our wind turbine placement optimization problem. In this section, we introduce OpenStreetMap and define three scenarios for the experiments.

4.1 OpenStreetMap

OpenStreetMap is a community-driven project, which was started in 2004 with the objective to create a license free world map. It contains geographical like positions of streets and buildings. Neis *et al.* [14] come to the conclusion that "[...] at least in countries, in which the OSM project is well developed, the data is becoming comparable in quality to other geodata from commercial providers [...]". The data is provided in easily processable XML files containing the geo-information. The overall amount of data is more than 400 GByte. For Germany 3.6 GByte geodata are available. In our simulation model, we dynamically load the relevant data for the corresponding sector and use the information to define constrained areas on the target. We model that the minimum distance between a turbine i and a street should be $1.5 \cdot h_i$ and 500 m to a building which is a good practice.

On the map plots in the remainder of this paper, yellow lines stands for streets, grey areas illustrate buildings and the constrainted areas around streets and buildings are visualized in red.

4.2 Scenarios

In this section, we introduce our scenarios to test the optimization algorithms. We focus on onshore wind farms. In Lower Saxony most wind farms consist of fewer than 30 turbines [16], and we define scenarios with this in mind. Even at this scale, a wind farm consisting of 20 Enercon E-101 wind turbines with 3000 kW and 60% of full load hours produces about 315 GWh/a. An output increase by 1% will increase the value of the produced energy on the basis of a price of 0.10 EUR/kWh by 315 000 EUR per year.

In the scenarios we define multiple constraints: (1) the OpenStreetMap data results in areas that are unavailable for the placement of turbines, and (2) we require like Manwell [12] a minimum safety distance depending on the rotor size r between every turbine i and j of $8 \cdot \max(r_i, r_j)$.

Scenario 1. In Scenario 1, the task is to place 25 turbines on an empty map of size $m_{\text{size}} = (3.0, 3.0)$ leading to an area of 9 km^2. The wind distribution is very similar to Figure 1. In addition, the potential in the right upper corner is slightly higher than on the rest of the map. A turbine will produce nearly 700 kW in mean per year at this position if it is not influenced by wake effects.

Scenario 2. In the upper right corner of Scenario 2 in Figure 5(a) Leerhafe, a part of the city Wittmund in Lower Saxony, can be seen. The scenario has the coordinates in decimal degrees $53.5015° - 53.5317°, 7.7346° - 7.7836°$, with size $m_{\text{size}} = (3.3, 3.3)$ leading to an area of 10.9 km^2. It contains 250 buildings and 64 streets consisting of 489 parts. All these elements lead to constraints. As the scenario is next to the position of the wind rose in Figure 1, the wind distribution is very similar to it.

Scenario 3. Scenario 3 on Figure 5(b) has many constrained areas. It lies at the position $53.4023° - 53.4433°, 8.0743° - 8.1412°$. It employs size $m_{\text{size}} = (4.5, 4.5)$ resulting in an area of 20.3 km^2 and contains 185 buildings and 229 streets consisting of 1493 parts. Its wind distribution can be seen in Figure 1.

5. EVOLUTIONARY STRATEGIES

The turbine placement problem is a continuous blackbox optimization problem $f : \mathbb{R}^N \to \mathbb{R}$ with constraints $g : \mathbb{R}^N \to \mathbb{B}$, i.e., a candidate solution $\mathbf{x} \in \mathbb{R}^N$ encodes the geo-positions of $N/2$ turbines that can be feasible ($g(\mathbf{x}) = 0$) w. r. t. geo-constraints or not ($g(\mathbf{x}) = 1$). In our implementation, a fitness function evaluation f first invokes a run of the geo-constraints module, and in case of feasibility a run of the wake model module determines the total power of the wind park. As fitness function f to maximize, we use the sum of the power output E of every turbine:

$$f(\mathbf{x}) = \sum_{i=1}^{N/2} E(\mathbf{t}_i). \tag{13}$$

where $\mathbf{t}_i$ is defined as in Equation 1. For optimization, we employ evolution strategies, as they are strong blackbox optimization heuristics in continuous solution spaces. In this section, we present the optimization variants we employ in the experimental section, and describe how solutions are initialized and how constraints are handled. We use six

strategies that can be categorized into two groups. One group is formed by the strategies that optimize the solution vector $\mathbf{x}$ without any knowledge about which value represents which turbine. For this category we use the state for the art CMA-ES and an adaptive $(1+1)$-ES. In the other group the strategies modify individual turbines. These approaches use the knowledge which values represents which turbine and coordinate. For this category we use a replacing strategy, a deterministic $(1+1)$-ES, an adaptive $(1+1)$-ES and a new self-adaptive approach.

5.1 Constraint Handling

Infeasible solutions from the geo-constraints get a low fitness, i.e., if $g(\mathbf{x}) = 1$, we assign $f(\mathbf{x}) = 0$. This is a variant of death penalty that does not require a modification of the optimization algorithm with a feasibility control loop while generating offspring candidate solutions. This type of death penalty is similar the approach by Morales and Quezada [13].

5.2 Initialization

The initialization of the turbine locations can play an important part in the optimization. We experiment with chessboard and random initialization to create the initial solution $\mathbf{x}_0$. Chessboard initialization places the turbines equidistantly on a grid also taking into account the geo-constraints. If there is an infeasible area, the final chess pattern will look like a chessboard with missing fields. Random initialization places the turbines randomly on the map but only at valid positions taking into account the geo-constraints, if applicable.

5.3 Holistic Approaches

In the following, we introduce the optimization approach we employ in the following experiments. As stated before, our optimization approaches can be classified into holistic ones that treat the turbine placement task as N-dimensional optimization problem, and turbine-oriented approaches that use evolutionary operators oriented to the coordinates $\mathbf{t}_i$ of specific turbines that are randomly chosen.

5.3.1 *Adaptive* $(1+1)^N$*-ES*

The first holistic optimization approach is a simple (1+1)-ES [3] with Gaussian mutation

$$\mathbf{x}' = \mathbf{x} + \sigma \cdot \mathcal{N}(0,1) \tag{14}$$

and Rechenberg's step size control [3], which works as follows. The $(1+1)$-ES runs for a number G_r of generations. During this period, step size σ is kept constant and the number G_s of successful mutations is counted. From G_s, the success probability can be estimated as $P_s = G_s/G_r$. Step size σ is increased according to $\sigma' = \sigma \cdot \tau$, if $P_s > 1/5$, and decreased otherwise, with $\tau > 1$. We use $\tau = 1.1$, which turned out to be the best choice in preliminary experiments. The $(1+1)$-ES accepts the new candidate solution $\mathbf{x}'$, if its fitness is better than or equal to the fitness of $\mathbf{x}$, i.e., if $f(\mathbf{x}') \geq f(\mathbf{x})$. The ES terminates after G generations.

5.3.2 *CMA-ES*

Using a CMA-ES [7] for optimizing wind turbines on an empty map with various kinds of border shapes was already conducted by Wagner *et al.* [18]. Their experiments showed that the optimization results of the CMA-ES outperformed the optimization module of the industry tool OpenWind [1]. The CMA-ES starts with an initial solution $\mathbf{x}_0$. The initial standard deviation σ_0 is set to $\sigma_0 = 10.0$ corresponding to 10 m. A small standard deviation makes constraint violations less likely and prevents the CMA-ES to some extent from problems with infeasible mutations $\mathbf{x}'$ while adjusting its covariance matrix $\mathbf{C}$.

5.4 Turbine-oriented Approaches

The mutation operators of turbine-oriented approaches concentrate on the locations $\mathbf{t}_i$ of subsets of turbines $i \in \mathcal{T}$, in particular they concentrate on only one turbine for each mutation operator execution.

5.4.1 *Adaptive* $(1+1)^1$*-ES*

A special case of our turbine-oriented approaches is the adaptive $(1+1)^1$-ES that randomly selects one turbine i at location $\mathbf{t}_i$ from the set of turbines $\{1, \ldots, N/2\}$. From this turbine, only one coordinate (x_i or y_i) is randomly selected and subject to Gaussian mutation, see Equation 14. Like in case of the holistic adaptive $(1+1)^N$-ES, Rechenberg's step size control is employed.

5.4.2 *Replacing*

The replacing optimization is a simple optimization approach. In every step, the algorithms removes randomly one chosen turbine i at the position $\mathbf{t}_i$ from the solution $\mathbf{x}$ and replaces the turbine i at a randomly chosen new position $\mathbf{t}'_i$ to create a new solution $\mathbf{x}'$. If the fitness function value $f(\mathbf{x}')$ is greater than $f(\mathbf{x})$ the new solution is picked as basis for the next step. So it is a very basic $(1+1)$-ES but without any step size control.

5.4.3 *Deterministic* $(1+1)^\mathbf{t}$*-ES*

Lückehe *et al.* [11] proposed a turbine-oriented deterministic $(1+1)$-ES to place and optimize iteratively turbines on a map with constraints. We use the optimization part of this approach and combine it with the initial solutions $\mathbf{x}_0$. In this approach, the $(1+1)^\mathbf{t}$-ES picks in every generation one turbine i with the position $\mathbf{t}_i$, and mutates it by moving i to a new position $\mathbf{t}'_i$. It applies:

$$\mathbf{t}'_i = \mathbf{t}_i + \sigma(g) \cdot \mathbf{m}_{\text{size}} \cdot \mathcal{N}(0,1) \tag{15}$$

with the step size:

$$\sigma(g) = 1.0 - \left(\left(1.0 - \frac{1}{G} \right) \cdot \frac{g}{G} \right) \tag{16}$$

where g is the actual number of generation and G the total number of generations. After the mutation, the strategy evaluates the new solution and selects the solution with the highest fitness function value.

5.4.4 *Self-Adaptive* $(1+\lambda)$*-ES*

We want to improve the results by using a self-adaptive $(1+\lambda)$-ES. A self-adaptive step size $\sigma \in [0,1]$ should make it possible to react more flexibly to the scenarios. New positions are created like in Equation 15. We also want to extend the options for the strategy, and we do this as follows. First, we add the capability to move multiple turbines at the same time and not consecutively. How many turbines n shall be moved at the same time is controlled self-adaptively. The second new option for the strategy is to remove one turbine $\mathbf{t}_i$ from solution $\mathbf{x}$ and replace it at a new position $\mathbf{t}'_\mathbf{i}$. The

probability $p \in [0, 1]$ that controls how often the strategy move or replace a turbine is also operated self-adaptively.

As step size σ and probability p are bound constrained, we employ the operator by Bäck and Schütz [2] for interval constrained self-adaptation:

$$\sigma' = \left(1 + \frac{1-p}{p} \cdot e^{\tau \cdot \mathcal{N}(0,1)}\right)^{-1}. \quad (17)$$

The parameter τ controls the magnitude of mutations. We choose the setting $\tau = 0.22$ like proposed in [2]. The number n of turbines to change in one step will be unchanged $n' = n$ with the probability 2/3 and increased or decreased by 1 with probability 1/6 with constraint $n \geq 1$. In initial tests, a population size of 50 showed a good performance.

6. EXPERIMENTAL RESULTS

In this section, we experimentally analyze the optimization variants introduced in the previous section on the three scenarios presented in Section 4.2.

6.1 Comparison of Evolutionary Algorithms

Table 1 shows the results of the two initialization variants combined with the six introduced optimization algorithms on benchmark Scenario 1, which does not have geo-constraints. The row *Init.* shows the fitness at initialization. The figures show the mean fitness with corresponding standard deviation and the best fitness achieved in 25 runs. Bold values show the best results achieved for Scenario 1, in which 25 turbines have to be placed ($N = 50$).

Init.	Chess		Random	
Algo.	Mean ± Std	Max	Mean ± Std	Max
Init.	14679.9 ± 0.0	14679.9	14136.4±207.2	14573.6
$(1+1)^N$	15169.1 ± 60.8	15281.6	14892.5±126.6	15083.3
CMA	15356.5 ± 48.6	15418.9	15359.5±110.9	15449.6
$(1+1)^1$	15408.8 ± 22.2	15446.6	15461.4 ± 32.6	15510.6
Rep.	15438.2 ± 25.1	15482.8	15443.8 ± 25.1	15505.5
$(1+1)^{\mathbf{t}}$	**15532.1**±10.2	15552.0	**15527.5**±13.4	15551.0
$(1+\lambda)$	15525.6 ± 17.3	**15557.4**	15524.6 ± 18.9	**15556.4**

Table 1: Experimental results of Scenario 1. Selected corresponding statistical test can be found in Table 4.

We can observe that all optimization heuristics achieve a significant improvement in comparison to the initial solutions. The $(1 + 1)^{\mathbf{t}}$-ES achieves the best results in average with a small standard deviation. The self-adaptive $(1 + \lambda)$-ES achieves the second best mean result, but at the same time the best maximum of all strategies. The class of holistic optimizers is outperformed by all turbine-oriented methods, which is probably due to the fact that it treats the whole optimization problem in a *holistic* fashion, i.e., all N variables at once, although strong dependencies exist between the turbines. The $(1 + 1)^N$-ES achieves the worst results of all strategies and is outperformed by the CMA-ES, which has much stronger capabilities in adapting its Gaussian mutations. The replace strategy achieves higher wind potentials than the $(1 + 1)^1$-ES in the chessboard initialization case, and vice versa in the random intialization case. Both strategies are less specialized than the deterministic $(1+1)^{\mathbf{t}}$-ES and the self-adaptive turbine-oriented $(1+\lambda)$-ES and perform worse, but still clearly outperform the holistic variants in mean and in the maximum achieved values. The statistical significance of the most important observations are confirmed with a Wilcoxon signed rank-sum test [9], see Table 4.

Init.	Chess		Random	
Algo.	Mean ± Std	Max	Mean ± Std	Max
Init.	10944.2 ± 0.0	10944.2	10626.5±145.2	10875.3
$(1+1)^N$	11221.3 ± 33.0	11287.3	11084.6±126.6	11280.2
CMA	11359.7 ± 12.5	11386.6	11348.4 ± 89.8	11475.2
$(1+1)^1$	11399.3 ± 17.5	11444.5	11437.8 ± 47.8	11502.5
Rep.	11484.0 ± 9.7	11505.0	11480.8 ± 12.1	11506.5
$(1+1)^{\mathbf{t}}$	**11524.1** ± 8.4	**11538.2**	**11524.6** ± 6.6	11535.6
$(1+\lambda)$	11516.5 ± 11.7	11537.2	11519.9 ± 9.7	**11536.6**

Table 2: Experimental results of Scenario 2.

Table 2 shows the experimental comparison of all evolutionary approaches on benchmark Scenario 2. This benchmark problem is an $N = 36$-dimensional problem with 18 turbines. We observe a similar behavior of the algorithms in comparison to Scenario 1. Again, the deterministic $(1+1)^{\mathbf{t}}$-ES and the self-adaptive $(1 + \lambda)$-ES outperform the other approaches. Further, we can observe that the replace strategy is better than the $(1 + 1)^1$-ES for both initialization schemes, which is probably due to the highly constrained solution space.

Init.	Chess		Random	
Algo.	Mean ± Std	Max	Mean ± Std	Max
Init.	10537.7 ± 0.0	10537.7	10491.3±200.9	10815.4
$(1+1)^N$	10675.3 ± 32.7	10774.6	10814.4±144.4	11098.4
CMA	10831.0 ± 29.9	10859.1	11008.2±105.4	11187.8
$(1+1)^1$	10889.7 ± 51.5	11083.8	11083.6 ± 88.2	11234.1
Rep.	11261.5 ± 9.9	11286.9	11261.8 ± 11.9	11280.5
$(1+1)^{\mathbf{t}}$	**11288.9** ± 8.0	11301.6	**11286.0** ± 6.9	**11300.7**
$(1+\lambda)$	11286.5 ± 9.8	**11302.7**	11283.8 ± 9.1	11295.8

Table 3: Experimental results of Scenario 3.

Last, we show the experimental results on Scenario 3 in Table 3. This problem employs the most constraints with 17 turbines resulting in an $N = 34$-dimensional problem. Again, $(1 + 1)^{\mathbf{t}}$-ES and $(1 + \lambda)$-ES perform best, i.e., with chessboard initialization, the $(1 + 1)^{\mathbf{t}}$-ES achieves the best mean result, but the $(1 + \lambda)$-ES reaches the highest fitness. With random initialization, the $(1+1)^{\mathbf{t}}$-ES achieves the best mean and best overall fitness. Although the mean results are better, the $(1 + 1)^{\mathbf{t}}$-ES does not perform significantly different than the $(1 + \lambda)$-ES, see the Wilcoxon test in Table 4. Further, we can observe that the quality of the replace strategy is close to the two best turbine-oriented optimizers, probably because of the same argument like in Scenario 2, i.e., because of the highly constrained solution space.

	CMA	$(1+1)^{\mathbf{t}}$	$(1+\lambda)$
CMA		+/+/+	+/+/+
$(1+1)^{\mathbf{t}}$	+/+/+		−/+/−
$(1+\lambda)$	+/+/+	−/−/−	

Table 4: Statistical significance of comparison between selected experiments of Tables 1 to 3.

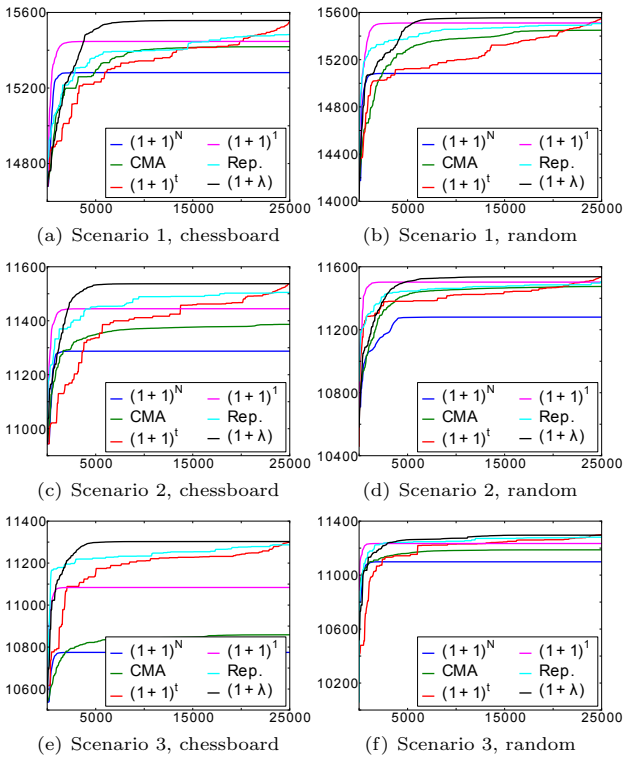
Table 4 shows an analysis of the statistical significance of selected experiments of Tables 1 to 3 employing the Wilcoxon signed rank-sum test. The runs of the CMA-ES,

the $(1+1)^{\mathbf{t}}$-ES and the $(1+\lambda)$-ES are compared to each other on the three scenarios, corresponding to order of scenarios. If the p-value is lower than 0.05, the difference of the runs is statistically significant with a significance level of 5%, indicated by '+', otherwise indicated by '-'. The upper right triangle of the table shows the results for chessboard initialization, while the lower left part shows the corresponding results for random initialization.

In general, when comparing chessboard and random initialization, random tends to result in higher standard deviations. This is probably because of the varying starting conditions. Interestingly, there is no significant difference between chessboard and random initialization for the best optimization strategies $(1+1)^{\mathbf{t}}$-ES and $(1+\lambda)$-ES resulting in a p-value greater than 0.05. For the $(1+1)^{N}$-ES and the $(1+1)^{1}$-ES, the initialization makes a significant difference.

6.2 Evolutionary Runs

Now, we concentrate on the comparison of the evolutionary dynamics by analyzing the best evolutionary optimization runs of all experiments, see Figure 6.



(a) Scenario 1, chessboard (b) Scenario 1, random

(c) Scenario 2, chessboard (d) Scenario 2, random

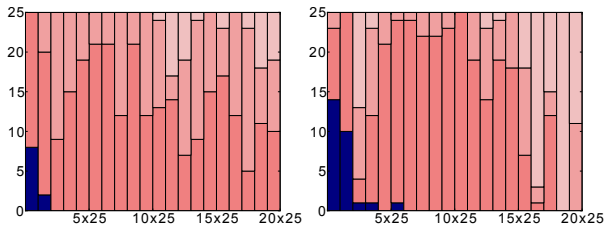
(e) Scenario 3, chessboard (f) Scenario 3, random

Figure 6: Development of best evolutionary runs when optimizing Scenarios 1, 2, and 3 with chessboard and random initialization. On the y-axis we show the energy output of the wind farm after x fitness evaluations.

We can observe that the $(1+1)^{1}$-ES shows the fastest process at the very beginning of the search, except for the chessboard initialization in Scenario 3 where the replacing strategy is the fastest one. After about 2000 to 5000 fitness function evaluations, depending on the initialization and scenario, both strategies are overtaken by the self-adaptive $(1+\lambda)$-ES. In later phases of the search, the $(1+\lambda)$-ES shows an excellent typically evolutionary convergence behavior. Interestingly, the deterministic $(1+1)^{\mathbf{t}}$-ES is optimizing comparatively slowly at the beginning of the search, but shows its strengths in later phases of the search. Although it shows phases of stagnation, the low mutation rates at the end of the search let the optimization progress accelerate in later generations. But at the end of the search, Equation 16 does not yield feasible mutation rates and in most cases the search ends before reaching the level of fitness achieved by $(1+\lambda)$-ES. We will analyze this behavior in the future and modify Equation 16 to allow a continued optimization without infeasible mutation rates.

6.3 Analysis of Operator Ratios

In the following, we analyze the operator frequencies applied during optimization with the $(1+\lambda)$-ES. Figure 7 shows the corresponding results of the best runs in Scenario 2 with (a) chessboard and (b) random initialization. At the beginning, the replacement operation (blue part) is applied in about one third of the generations, while two third are devoted to turbine shifting (red parts). In the following 25 generations, less turbines are replaced. At later phases of the search, the ratio of group shifting operations is increased while converging to the optimum.



(a) Scenario 2 (chessboard) (b) Scenario 2 (random)

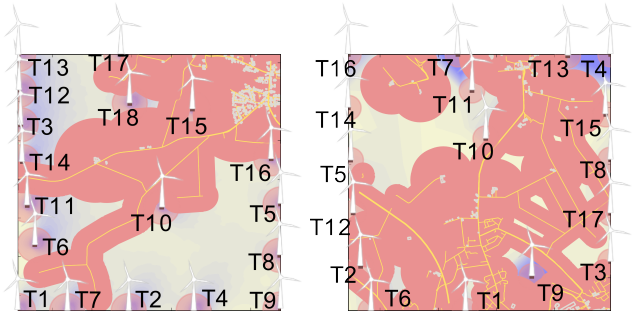
Figure 7: Exemplary development of operator frequencies applied every 25 generations during optimization with the $(1+\lambda)$-ES on Scenario 2 with (a) chessboard and (b) random initialization.

6.4 Optimal Placement Results

Last, we visualize the optimal turbine placements that are evolved by the $(1+1)^{\mathbf{t}}$-ES for Scenario 2 (see Table 2), and by the $(1+\lambda)$-ES for Scenario 3 (see Table 3) in Figure 8. We can observe that the majority of the turbines are placed at the borders of the map. In a few cases, small feasible islands between constrained areas are used in the middle of the map. It is obvious that the EA maximizes the distances between the turbines. Intuitively, this makes sense since the wake effects decrease with increasing distance. The horizontal inter-turbine distances are larger than vertical ones as the horizontal ratio of wind is comparatively large. The large free spaces stay free, as they are mostly influenced by wake effects, whose complex structures are clearly visible.

7. CONCLUSIONS

Turbine placement with a realistic wake effect model, real-world wind data, and also realistic geo-constraints results in a difficult optimization problem. In this work, we proposed four turbine-oriented optimization heuristics and performed a detailed experimental comparison on three benchmark scenarios in Lower Saxony. We compared our approaches to



(a) Scenario 2 (b) Scenario 3

Figure 8: Final turbine placements after 25000 fitness function evaluations on (a) Scenario 2 evolved by the $(1+1)^{\mathbf{t}}$-ES and (b) Scenario 3 optimized by the $(1+\lambda)$-ES.

a simple $(1+1)^N$-ES with adaptive step size control and to the CMA-ES, which belongs to the state-of-the-art approaches in continuous blackbox optimization and showed promising results in turbine placement tasks in the past. All turbine-oriented approaches outperformed the holistic strategies with statistically significance.

As future work, we plan to extend the experimental analysis to further scenarios, in particular concentrating on large-scale scenarios and also off-shore turbine placement with ground and ship route constraints. As constraints have an important part to play in realistic scenarios and the results have shown that the constraints make the optimization problems more difficult to solve, we will concentrate on more advanced constraint handling techniques in the future, ranging from penalty function to meta-modeling of the constraint boundary.

Acknowledgements

We thank the German Weather Service for providing the COSMO-DE data and the Ministry for Science and Culture of Lower Saxony for supporting this work with the PhD program *System Integration of Renewable Energy (SEE)*.

References

[1] AWS Truepower. AWS Openwind, 2008. http://awsopenwind.org/.

[2] T. Bäck and M. Schütz. Intelligent mutation rate control in canonical genetic algorithms. In *Foundations of Intelligent Systems, 9th International Symposium, ISMIS '96, Zakopane, Poland, June 9-13, 1996, Proceedings*, pages 158–167, 1996.

[3] H. Beyer and H. Schwefel. Evolution strategies - A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[4] Deutscher Wetterdienst. COSMO-DE: numerical weather prediction model for Germany, 2012. http://tinyurl.com/dwd-cosmo-de.

[5] ENERCON GmbH. Product Overview, 2015. http://tinyurl.com/enercon101.

[6] M. M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, Oct. 2008.

[7] N. Hansen. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*, pages 75–102. Springer, 2006.

[8] J. F. Herbert-Acero, O. Probst, P.-E. Rethore, G. C. Larsen, and K. K. Castillo-Villar. A review of methodological approaches for the design and optimization of wind farms. *Energies*, 7(11):6930–7016, 2014.

[9] G. Kanji. *100 Statistical Tests*. SAGE Publications, London, 1993.

[10] A. Kusiak and Z. Song. Design of wind farm layout for maximum wind energy capture. *Renewable Energy*, 35(3):685–694, 2010.

[11] D. Lückehe, O. Kramer, and M. Weisensee. An evolutionary approach to geo-planning of renewable energies. In *28th International Conference on Informatics for Environmental Protection: ICT for Energy Effieciency (EnviroInfo)*, pages 501–508, 2014.

[12] J. Manwell, J. McGowan, and A. Rogers. *Wind Energy Explained: Theory, Design and Application*. John Wiley and Sons Ltd, London, 2002.

[13] A. K. Morales and C. V. Quezada. C.v.: A universal eclectic genetic algorithm for constrained optimization. In *In: Proceedings 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT*, pages 518–522. Verlag Mainz, 1998.

[14] P. Neis, D. Zielstra, and A. Zipf. The street network evolution of crowdsourced maps: Openstreetmap in germany 2007–2011. *Future Internet*, 4(1):1–21, 2011.

[15] H. Neustadter. Method for evaluating wind turbine wake effects on wind farm performance. *Journal of Solar Energy Engineering*, pages 107–240, 1985.

[16] The Wind Power. Wind farms in Lower Saxony, Germany, 2015. http://tinyurl.com/parks-lower-saxony.

[17] M. Wagner, J. Day, and F. Neumann. A fast and effective local search algorithm for optimizing the placement of wind turbines. *Renewable Energy*, 51(0):64–70, 2013.

[18] M. Wagner, K. Veeramachaneni, F. Neumann, and U.-M. O'Reilly. Optimizing the layout of 1000 wind turbines. In *European Wind Energy Association Annual Event*, 2011.

[19] C. Wan, J. Wang, G. Yang, X. Li, and X. Zhang. Optimal siting of wind turbines using real coded genetic algorithms. *European Wind Energy Association Conference and Exhibition*, 2009.

[20] C. Wan, J. Wang, G. Yang, and X. Zhang. Optimal micro-siting of wind farms by particle swarm optimization. In *Advances in Swarm Intelligence*, volume 6145 of *Lecture Notes in Computer Science*, pages 198–205. Springer, 2010.

[21] W. Weibull. A statistical distribution function of wide applicability. *Journal Applied Mechanics - Transactions of ASME*, 3(18):293–297, 1951.