Robust wind farm layout optimization

Journal of Physics: Conference Series **2767** (2024) 032036 doi:10.1088/1742-6596/2767/3/032036

Robust wind farm layout optimization

Michael Sinner and Paul Fleming

National Wind Technology Center, National Renewable Energy Laboratory, Colorado, USA [1]

E-mail: michael.sinner@nrel.gov

Abstract. Wake interactions in wind farms cause losses in annual energy production (AEP) on the order of 10%. Wind farm designers optimize the layout of the farm to mitigate wake losses, especially in the dominant site-specific wind directions. As wind turbines and wind farms grow in scale, optimization becomes more complex. Offshore wind farms regularly comprise more than 100 wind turbines and are characterized by complex boundaries due to shipping lanes, neighboring wind farms, and other constraints.

Layout optimization methods are broadly split between gradient-based and gradient-free approaches. Gradient-based approaches can converge quickly and perform well for smaller, academic problems but are often sensitive to initial conditions and tuning parameters and require expert knowledge to use. On the other hand, gradient-free approaches can be more robust to problem complexities. We present a robust layout optimization approach based on a random search algorithm. The algorithm is intended for those who are not optimization experts and has few tuning parameters that need specification to achieve satisfactory results. Unlike off-the-shelf methods, which use generally available, non-domain-specific optimization routines that accept as inputs an optimization function and constraint definitions, this approach takes advantage of the relative computational costs of the different evaluations by evaluating cheaper computations first (boundary and minimum distance constraints) and running expensive AEP evaluations only if all other checks pass. Moreover, an outer genetic algorithm allows multiple solutions to evolve in parallel, enabling rapid solution development on high-performance computers. We discuss the relative ease of selecting necessary tuning parameters and demonstrate the efficacy of the genetic random search on a complex layout problem consisting of placing 70 turbines in a nonconvex and unconnected boundary region.

1. Introduction

As wind farms grow in size and competition for new areas of wind development intensifies, increasing attention has been given to optimizing wind farm layouts to minimize the impacts of turbine-to-turbine wake losses and maximize the annual energy production (AEP) of the farm. Moreover, boundaries for turbine positioning are becoming complex because of the proximity of shipping channels, infrastructure, protected areas, and human activities, which can result in nonconvex wind farm boundaries and even wind farms with separable (or unconnected)

[1] This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Wind Energy Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

regions [1]. Finally, layout optimization is appearing earlier in the wind farm procurement and development cycle, and layouts are reoptimized and fine-tuned as the wind farm approaches construction.

Approaches for wind farm layout optimization are broadly split between gradient-based (or first-order) methods and gradient-free (zeroth-order) methods. Of these, gradient-free approaches [2, 3, 4] are perhaps the more obvious choice due to the presence of many local minima/maxima in the nonconvex layout optimization problem [5]. However, gradient-free optimizers can be slow to converge, prompting growing research into gradient-based methods [6, 7, 8] because of their faster local convergence. The existence of many local minima/maxima remains a challenge for gradient-based approaches, although some heuristic workarounds have been proposed in the literature [5]. Moreover, gradient-based methods cannot generally handle cases where turbines are to be placed in an area consisting of two or more unconnected polygons without substantial modification. Further, in layout optimization problems, the optimization surface can be quite shallow/flat near optimal solutions, and we have observed that while gradient-based methods move quickly when solutions are far from optimal, they become sluggish or unstable nearer to the optimum unless algorithm parameters are carefully tuned for the problem.

We are aware of only one previous study that has specifically addressed layout optimization problems with unconnected regions [1]. Thomas et al. [1] provide a detailed comparison of various optimization algorithms and present a novel (gradient-free) discrete exploration-based optimization (DEBO) algorithm, which consists of an initial phase of greedy turbine placement (initialization) followed by a local maximization by randomly checking a discrete set of candidates in the vicinity of the initialized layout. As the authors point out, this results in finitely many positions to check, which guarantees a finite number of possible layouts and therefore a finite stop time for the algorithm, in contrast to exploration in a continuous space [4].

In this paper, we employ gradient-free layout optimization to provide an algorithm that is relatively robust, has few tuning parameters, and is suitable for optimizing layouts in areas with complex boundaries. The genetic random search (GRS) algorithm chosen is described in Section 2, and its strengths are presented therein. The algorithm is similar to Thomas et al.'s DEBO [1] but searches randomly in a continuous space. While this means that the algorithm does not inherently reach a terminating condition (as there are infinite solution candidates), we argue that in practice, it both simplifies the algorithm (few tuning parameters are required) and may enable an improved solution over discrete searches. The tuning parameters that remain to be chosen, as analyzed in Section 3, can be selected within a broad range and still provide satisfactory results, demonstrating the robustness of the algorithm; moreover, we argue that the parameters are intuitive to select to alter the progress and precision of the candidate solution. We also demonstrate a relative insensitivity to the initial candidate layout, which can hamper gradient-based methods. To demonstrate the full capabilities of the proposed GRS algorithm, we optimize the layout of a 70-turbine wind farm with a complex boundary and heterogeneous inflow conditions in Section 4 before providing brief conclusions in Section 5.

2. Layout optimization using genetic random search

We propose that, for the large-scale, complex wind farm optimization problems that are common in practice, a random search approach is appropriate. Rather than computing gradients and taking a step in a gradient descent direction, the random search takes a step of random length in a random direction. If the new layout is feasible, its AEP is compared to that of the previous layout and, if greater, the solution is saved. Moreover, several random searches can be undertaken in parallel and final solutions compared in a genetic algorithm approach. We see three important benefits to such a GRS as compared to gradient-based optimization methods:

(i) The approach is straightforward to implement and is robust to the layout optimization

problem specifics. In particular, the constraints (such as the boundary constraint and minimum turbine-to-turbine spacing) can be checked, and the candidate discarded if infeasible, before evaluating the cost function (which requires running a flow solver such as FLORIS) to determine the AEP. Because the cost function is much more computationally expensive to evaluate than the constraints, this ordering presents significant speed improvements.

(ii) The random search approach readily handles split regions for optimization (see the figure in Section 4 for an example). Provided that the possible steps are large enough, a random step can move a turbine between regions. Off-the-shelf gradient-based methods require significant modification to deal with such split regions of feasibility [1].

(iii) The outer genetic algorithm is particularly suited to parallelization, making the method easy to scale and use with high-performance computing. Gradient-descent-based algorithms can parallelize gradient and cost function evaluations, but must take gradient descent steps in serial, whereas the proposed genetic random search can be run completely independently for many search steps before selecting a set of top-performing layouts to proceed in the next generation.

2.1. Layout optimization problem

Let $x = \{x_i\}_{i=1,\dots,N}$, $x_i \in \mathbb{R}^2$ represent the set of locations of N turbines in 2-D space, that is, x defines the lateral and longitudinal locations of the turbines within an N-turbine wind farm. Further, let w represent a wind condition bin, for example, wind speed between 8 and 10 m/s and wind direction between 270 and 273 degrees. The actual wind condition at any given time is a random variable W with probability mass function (PMF) (representing the probabilities of wind conditions over a year) $W \sim p_W(w)$. The PMF $p_W(w)$ is commonly referred to as the "wind rose" of the site.

With the operation of each wind turbine fixed according to its power and thrust curves, the instantaneous power P generated by the farm is a function of the current wind condition w and the layout x, i.e.,

$$P = P(w, x)\,. \tag{1}$$

The evaluation of Eq. (1) requires running a wind farm wake model. In the results presented here, we evaluate Eq. (1) using FLORIS's Gauss-curl-hybrid (GCH) model [9, 10]. However, the method we present is agnostic to the form of the model evaluated in Eq. (1).

During layout optimization, wind turbines should be placed inside a closed region $R \subset \mathbb{R}^2$. Often in the literature, R is a (convex) polygon, but in realistic cases R may also be composed of multiple unconnected and/or nonconvex polygons. Further, for structural loading and safety reasons, a minimum distance d that should be maintained between any two turbines is often specified.

Collecting the above information, we can form an optimization problem to maximize the mean (expected) power over a year (equivalently, maximize the AEP) as

$$\underset{x}{\text{maximize}} \sum_{w} P(w, x) p_W(w) \tag{2a}$$

$$\text{subject to } x \subset R \tag{2b}$$

$$||x_i - x_j|| \geq d,\ \forall i = 1, \dots, N\ \ \forall j = 1, \dots, N, j \neq i\,. \tag{2c}$$

Here, $||\cdot||$ denotes the Euclidean norm.

2.2. Solution method

We use a random search approach for layout optimization [4]. Given a feasible candidate solution x, that is, a value x that satisfies the constraints (2b) and (2c), the random search simply chooses

a random turbine to move in a random direction at a random distance, and produces a new candidate $\hat{x}$. If $\hat{x}$ is feasible *and* increases the value of the cost function (2a) compared to x, $\hat{x}$ takes the place of x and the process repeats. If not, x is retained and the process repeats with new random selections. The random search method is formalized in Algorithm 1.

Algorithm 1 Random search layout optimization

Require: Layout x satisfying constraints (2b) & (2c); distance PMF $p_R(r)$; time limit $t_{\max}$
  while $t < t_{\max}$ **do**
    Sample $i \sim \mathcal{U}\{1, N\}$; $\theta \sim \mathcal{U}_{[0,2\pi)}$; $R \sim p_R(r)$
    $\hat{x}_i \leftarrow x_i$ moved distance R in direction θ
    if $\hat{x}_i \notin R$ **then**
      break
    else if $||x_i - x_j|| < d$ for any $j = 1, \ldots, N, j \neq i$ **then**
      break
    else if $\sum_w P(w, \hat{x}) p_W(w) > \sum_w P(w, x) p_W(w)$ **then**
      $x \leftarrow \hat{x}$
    end if
  end while
  return x

In Algorithm 1, t is the computer clock time. Algorithm 1 runs serially, providing an optimized (but not necessarily optimal) layout once the time limit $t_{\max}$ is met. As the algorithm is inherently random, running multiple times with the same initial condition x may not produce the same final layout once the time limit is reached. If multiple processors are available for computation, this property makes Algorithm 1 suitable for wrapping in a genetic algorithm. Specifically, Algorithm 1 may be run in parallel on multiple different processors with the same or differing initial conditions x, and when $t_{\max}$ is reached, the best layout(s) from across the different processors may be selected as new initial layouts before relaunching the next "generation" of the genetic algorithm. This GRS method is formalized in Algorithm 2. Note here that the "selection"

Algorithm 2 Genetic random search layout optimization

Require: Layout x satisfying constraints (2b) & (2c); number of individuals G; relegation number n; total time limit $T_{\max}$
  $x^{(g)} \leftarrow x$, $g = 1, \ldots, G$
  while $t < T_{\max}$ **do**
    for $g = 1, \ldots G$ **do in parallel**
      Run Algorithm 1 with $x \leftarrow x^{(g)}$
      $x^{(g)} \leftarrow x$
    end for
    Order the $x^{(g)}$ by cost function (2a) value
    Replace n lowest value $x^{(g)}$ with n highest value $x^{(g)}$
  end while
  $x \leftarrow \max_g \sum_w P(w, x^{(g)}) p_W(w)$
  return x

step of Algorithm 2 (that is, the selection of a new set of individuals $x^{(g)}$ by replacing the lowest-value candidates with copies of the highest-value candidates) is very simple, and other, more complex selection approaches are possible based on the value ("fitness") of each candidate, which may be more efficient. However, we have found this simple selection satisfactory, at least for relatively small numbers of individuals G ($2 \leq G \leq 10$). See Section 3.1 for more details.

Finally, after the outer computational time limit $T_{\max}$ (which is usually multiple times $t_{\max}$ to allow multiple generations to run, see Section 3.2) is reached, the returned value x is taken as the optimized wind farm layout.

2.3. Convergence and termination

The GRS algorithm presented randomly searches a continuous optimization space [4], which can continue indefinitely unless terminated by the time limits $t_{\max}$ and $T_{\max}$. In contrast, discrete searches can check every candidate solution and "self-terminate" in finite time [1]. However, there is no guarantee that a layout in the discrete candidate set is indeed optimal. In our opinion, simply allowing the GRS to continue to run until the time limit is reached is satisfactory: Because the solution is not updated unless a candidate improves the AEP compared to the previous candidate, the user can be satisfied that even if the best solution is arrived at early, this solution will still be presented at $T_{\max}$. However, if an early termination condition is desired, it would be straightforward to implement a condition of convergence (such as no improvement in the AEP over some chosen number of generations).

2.4. Alternative cost functions

The objective function (2a) specifies the mean power produced by the wind farm over a year as the target for optimization, as this is equivalent to the AEP (when multiplied by the length of a year). AEP has generally been the metric of concern for optimization. However, other objectives may also be of interest, such as maximizing the monetary value of the power produced. Depending on the operating contract that the farm is under, the owner may be paid for electricity at the spot market price. If different wind conditions w are correlated with different electricity prices, the cost function may be exchanged for the expected *value* of power produced annually, for example, $\sum_w v(w)P(w,x)p_W(w)$, where $v(w)$ is the price of electricity (specified in terms of unit power, or converted to a price per unit of energy delivered).

Other terms may be added to the objective function to represent, for example, costs associated with certain layouts such as cabling costs (e.g. $\sum_w P(w,x)p_W(w) - c(x) = \sum_w [P(w,x) - c(x)]\, p_W(w)$ for some cost function $c(\cdot)$. Further, the wind condition bin w can be directly expanded to include other atmospheric parameters such as turbulence intensity, provided that the PMF $p_W(w)$ is updated accordingly. Generally, various modifications can be made to the cost function in a straightforward manner, provided that it is still of the form

$$\sum_w f(w,x)p_W(w)\,. \tag{3}$$

No requirements are made on smoothness, differentiability, or convexity of the cost function. For the purpose of this study, we simply choose to optimize AEP as shown in problem (2).

3. Algorithm tuning

Algorithms 1 and 2 require several parameters that may be considered tuning parameters. Generally, we consider these to be more intuitive than tuning parameters required by gradient descent algorithms, and we provide a description of each here. For the purposes of demonstrating sensitivity to the different parameters, we use the example of placing 35 turbines within the slightly complex region shown in Fig. 1. The initialized gridded positions are shown in black in the left-hand plot, and an example of optimized positions is shown in the right-hand plot.

3.1. Genetic selection process

Aside from the total optimization time limit $T_{\max}$ (discussed in Section 3.2), two parameters particularly affect the outer genetic algorithm: the number of individuals per generation G

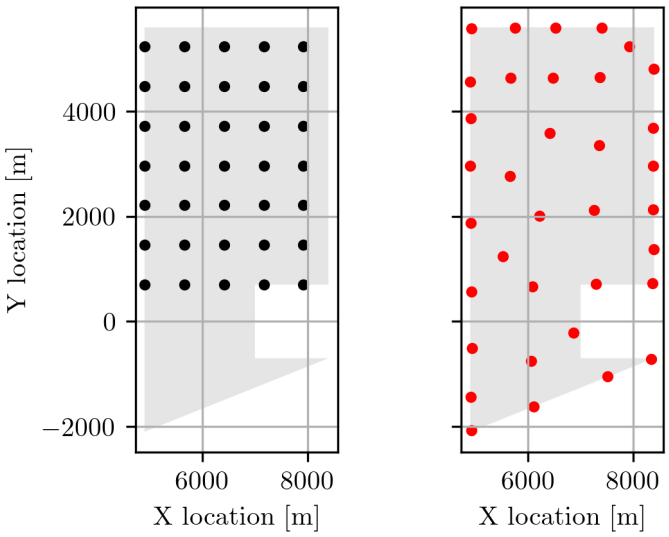

Figure 1. Region for 35-turbine layout problem used for demonstrating tuning parameter sensitivity. Turbines may be placed in the gray region. *Left:* Initial layout used for most studies. *Right:* Example optimized layout.

and the relegation number n. The former of these is simply constrained by the computing hardware: the more processors available, the higher G can be, which allows a wider search in each generation. However, the choice of the relegation number n (which represents the number of individuals who will be replaced by "fitter" individuals) is not so clear. To investigate this, we use the 35-turbine problem shown in Fig. 1. We set $G = 10$, and vary n from 1 to 5. The outer time limit $T_{\max}$ is set as 1 hour, and the inner time limit is set at $t_{\max} = T_{\max}/5$. However, our results did not show a clear dependence of algorithm performance on the relegation number: the best performers were $n = 1$ and $n = 5$, with minimal spread across all tested values of n in either the rate of improvement of the cost function per generation or the final optimized AEP after the five generations were complete. We therefore take the conservative approach of setting $n = 1$ (i.e., only the worst-performing individual is replaced) for all subsequent optimizations.

3.2. Time limits

The time limits $t_{\max}$ and $T_{\max}$ provide the limits on the (real) runtime of the algorithms. To achieve the highest-value solution x, the algorithms should be run as long as possible, and generally speaking, larger wind farms (i.e., larger N) will require more time to converge to a solution than smaller farms. However, there is a trade-off between how long to run the individual (Algorithm 1, limited by $t_{\max}$) compared to the outer genetic algorithm (Algorithm 2, limited by $T_{\max}$). To investigate this, we compare the results from optimizing using a fixed outer time limit $T_{\max} = 3600$ seconds (1 hour) and a varying inner time limit $t_{\max}$ so as to split the genetic algorithm into 2, 4, 8, 16, and 32 generations. We again use $G = 10$ individuals per generation (and set the relegation number to $n = 1$). The progress of the optimization is given in Fig. 2. The left plot shows the progress (in terms of percentage AEP improvement from the initial layout) as a function of the GRS algorithm generation, whereas the right plot shows the same data plotted according to computation time. Considering the right plot, it is evident (if somewhat surprising) that the progress is rather similar in all cases.

The left plot in Fig. 2 provides some intuitive explanation of the process that the GRS algorithm takes. After each generation, exactly one individual is terminated. However, this process can lead to "mutations" that persist for multiple generations before being eventually

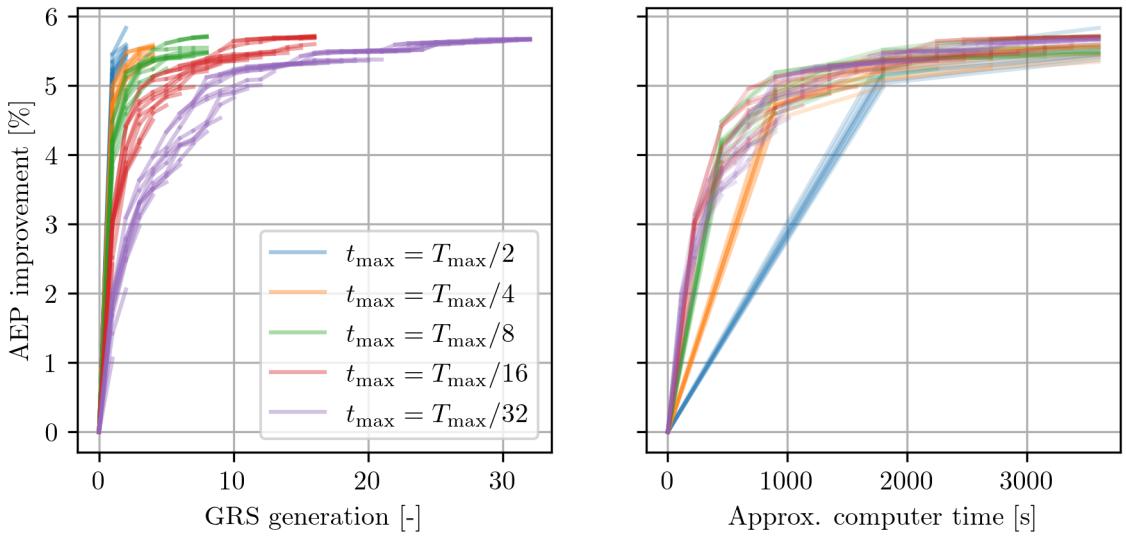

Figure 2. Progress of the GRS algorithm with various values of t_{max} (and fixed T_{max}). *Left:* AEP improvement by GRS generation. *Right:* AEP improvement by computation time.

ended (for example, around generations 15–22 for $t_{max} = T_{max}/32$). Moreover, some mutations survive an initial period of lower performance and recover their fitness, as seen around generations 5–15 for the $t_{max} = T_{max}/32$ case.

3.3. Initial layout

Given the presence of many local minima in layout optimization problems, they are often very sensitive to the initial condition selected. We propose that, although there is still a sensitivity, the random search method presented here may be somewhat *less* sensitive than gradient descent-based optimization approaches. To investigate this, we provide three initial layouts to the optimizer (Fig. 3, top plot): two based on a simple grid and the third based on an algorithm to place turbines far apart within the optimization space (the full approach will not be described here for brevity).

We then compare the results of optimizing using GRS (lower-left plot of Fig. 3) and `scipy`'s `minimize` routine using sequential least squares programming (SLSQP) (lower-right plot) run for an equivalent time period. First, it is clear that the SLSQP method is not well tuned. After an initial period of improvement using all three layouts, the algorithm appears to essentially become unstable. While it is likely that fine-tuning the SLSQP algorithm would improve results, the tuning would be problem-dependent, requiring significant effort for each new wind farm optimization problem. The GRS algorithm, on the other hand, robustly optimizes the layout with the varied initial layouts, converging on a similar (but not identical) optimized AEP value for all three initial layouts.

3.4. Distance probability mass function

Perhaps the most nuanced decision to be made in running the GRS algorithm is in the choice of the distance PMF $p_R(r)$ supplied to Algorithm 1. The distance PMF specifies the distance for random perturbations of the turbine location (the direction of perturbation is chosen from a uniform distribution around the circle). Depending on the complexity of the region for turbine placement, the choice of distance PMF can play a crucial role in the optimization process: if the region is disjoint (see Section 4), the designer can choose whether to allow jumps between the

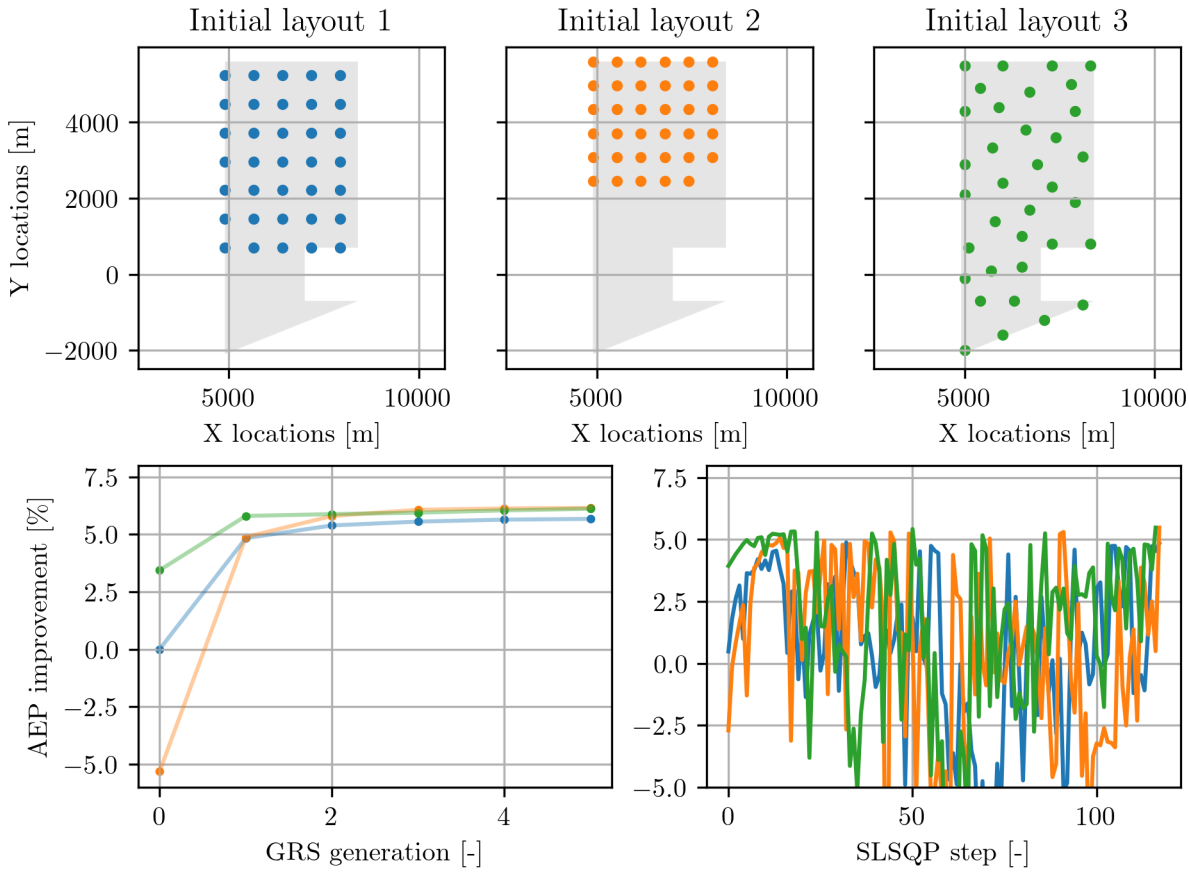

Figure 3. Optimizer sensitivity to initial conditions. Various initial layouts are shown in the upper plots, along with optimizer progress in the lower plots (GRS on the left, SLSQP (from `scipy.minimize`) on the right).

unconnected areas by including or omitting any probability mass above the minimum distance between regions.

Even for a single region, the choice of $p_R(r)$ may affect the optimizer performance, as a larger support will enable more rapid exploration but less fine positioning and vice versa. To investigate the effect, we again run the optimization of 35 turbines within the boundary shown in Fig. 1, using a range of PMFs shown in the upper plots in Fig. 4. The resulting optimization trajectories are shown in the lower plot of Fig. 4 (where only the best-performing individual at the end of each generation is shown). As expected, a PMF with smaller support (in particular, the first PMF shown in blue) proceeds more slowly and conservatively, whereas a PMF with significant mass at larger distances (e.g., the second (orange) and final (purple) PMFs) make faster initial progress. Striking a balance between the two, the fourth (red) PMF is perhaps a good "default" option, where the probability of a large perturbation (say, half of the distance across the optimization region) is 5%.

As indicated by the results in Fig. 4, the choice of distance PMF may depend on the level of fidelity needed in the design: a coarser PMF with larger perturbations may be good for quickly arriving at an approximate solution, whereas final build layouts may use a finer PMF with smaller perturbations. Naturally, these could be run sequentially (i.e., the final layout from a coarser design could be used to initialize a finer layout optimization). We consider the choice of

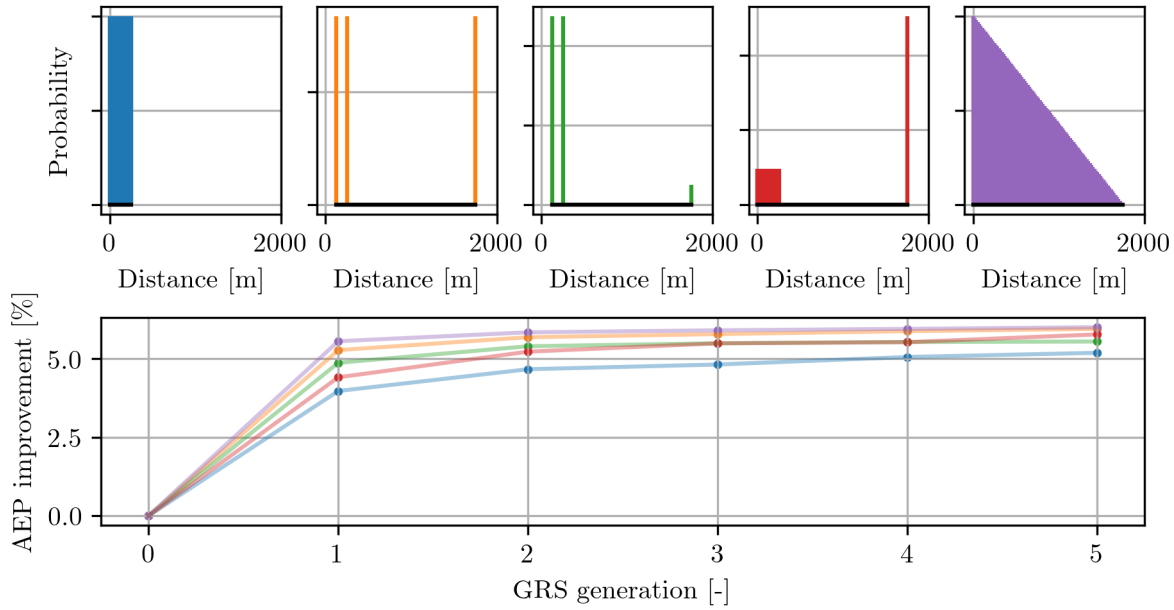

Figure 4. *Top:* Five tested distance PMFs. The first (shown in blue) contains 100 points uniformly distributed between 0 and 2 rotor diameters (D); the second and third contain only 3 points, at 1D, 2D, and 13.9D; the fourth has 99 points uniformly distributed between 0 and 2D and sharing 95% of the probability mass, with a single point of 5% mass at 13.9D; and the fifth (purple) has 100 points of linearly decreasing mass between 0 and 13.9D. *Bottom:* Resulting progress of the best individual in the GRS algorithm (colors match the distance PMFs in the upper plots).

distance PMF to be the most important "tuning parameter" for the GRS algorithm; however, we hope that its impact on the progress and capabilities of the algorithm are clear enough to make it user-friendly to select an appropriate distance PMF for the problem at hand.

4. Complex layout optimization problem

To demonstrate the strengths of the GRS algorithm, we perform layout optimization for a wind farm of 70 turbines in the complex, unconnected region shown in gray in Fig. 5 (left). No special effort is made to select an advantageous initial condition—turbines are simply placed in a grid in the center of the three regions, as shown by the black dots. Further, a heterogeneous wind map is provided to the underlying flow field such that the left side of the domain sees 10% lower wind speeds than the right side (with a linear increase from left to right). The optimization is then run for 40 generations with 15 minutes per generation (10 hours total) using 10 individuals. The resulting layout is shown with red dots. We note that here, a relatively conservative distance PMF was used; the optimizer could be encouraged to explore more quickly with a more aggressive PMF, at the likely expense of attaining a somewhat poorer final solution.

Even with the simple initial layout, the optimizer moves turbines between unconnected regions and achieves a satisfactory final layout. The trend observed by Thomas et al. [1] that optimized layouts often have turbines most densely spread around the boundary with relatively low density internal to the region is also seen here. Moreover, the optimizer packs turbines more densely in the right-hand portion of the domain to best utilize the higher wind speed region.

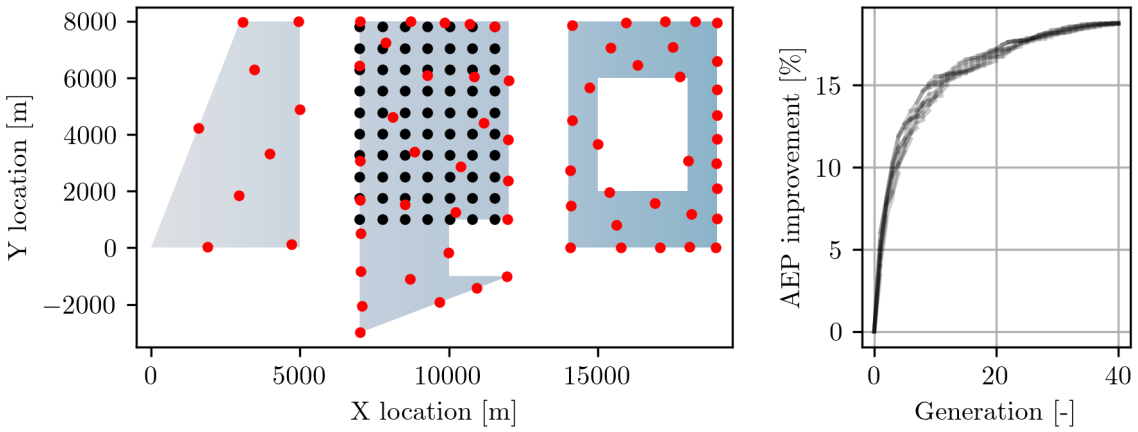

Figure 5. *Left:* Unconnected wind farm optimization problem. Initial wind turbine positions are shown in black; optimized positions are shown in red. The underlying shading indicates wind speed heterogeneity, with winds on the left side of the domain 10% lower than those on the right side across all wind directions. *Right:* Progress of the 10 individuals over the 40 generations.

5. Conclusions

With this paper, we present a robust wind farm layout optimization method based on a genetic random search (GRS). The GRS method has few tuning parameters, with the main sensitivity being the choice of probability mass function used for perturbations of the turbine locations. Moreover, the implementation is such that computationally cheaper constraint checks are carried out before more expensive cost function evaluations, limiting unnecessary computation. As such, we believe that the GRS implementation will be useful to non-optimization experts seeking a simple and robust approach for wind farm layout optimization, especially in complex scenarios such as nonconvex or unconnected boundary regions and wind speed heterogeneity. Our analysis, while heuristic, indicates that the GRS method can indeed be run "out of the box" to achieve satisfactory layout optimization results.

References

[1] J. J. Thomas, N. F. Baker, P. Malisani, E. Quaeghebeur, S. Sanchez Perez-Moreno, J. Jasa, C. Bay, F. Tilli, D. Bieniek, N. Robinson, A. P. J. Stanley, W. Holt, and A. Ning. A comparison of eight optimization methods applied to a wind farm layout optimization problem. *Wind Energy Science*, 8(5):865–891, 2023.
[2] Christopher N. Elkinton, James F. Manwell, and Jon G. McGowan. Algorithms for offshore wind farm layout optimization. *Wind Engineering*, 32(1):67–84, 2008.
[3] Ying Chen, Hua Li, Kai Jin, and Qing Song. Wind farm layout optimization using genetic algorithm with different hub height wind turbines. *Energy Conversion and Management*, 70:56–65.
[4] Ju Feng and Wen Zhong Shen. Solving the wind farm layout optimization problem using random search algorithm. *Renewable Energy*, 78:182–192, 2015.
[5] Beatriz Pérez, Roberto Mínguez, and Raúl Guanche. Offshore wind farm layout optimization using mathematical programming techniques. *Renewable Energy*, 53:389–399, 2013.
[6] David Guirguis, David A. Romero, and Cristina H. Amon. Toward efficient optimization of wind farm layouts: Utilizing exact gradient information. *Applied Energy*, 179:110–123, 2016.
[7] Ryan N. King, Katherine Dykes, Peter Graf, and Peter E. Hamlington. Optimization of wind plant layouts using an adjoint approach. *Wind Energy Science*, 2(1):115–131, 2017.
[8] Pieter Gebraad, Jared J. Thomas, Andrew Ning, Paul Fleming, and Katherine Dykes. Maximization of the annual energy production of wind power plants by optimization of layout and yaw-based wake control. *Wind Energy*, 20(1):97–107, 2017.
[9] NREL. FLORIS version 3.5.0, 2023.
[10] J. King, P. Fleming, R. King, L. A. Martínez-Tossas, C. J. Bay, R. Mudafort, and E. Simley. Control-oriented model for secondary effects of wake steering. *Wind Energy Science*, 6(3):701–714.