https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

# Stochastic Gradient Descent for Wind Farm Optimization

Julian Quick[1], Pierre-Elouan Rethore[1], Mads Mølgaard Pedersen[1], and Rafael Valotta Rodrigues[1]

[1]Technical University of Denmark, Risø National Laboratory for Sustainable Energy, Frederiksborgvej 399, 4000 Roskilde, Denmark

**Correspondence:** Julian Quick (juqu@dtu.dk)

**Abstract.** It is important to optimize wind turbine positions to mitigate potential wake losses. To perform this optimization, atmospheric conditions, such as the inflow speed and direction, are assigned probability distributions according to measured data, and these conditions are propagated through engineering wake models to estimate the annual energy production (AEP). This study presents stochastic gradient descent (SGD) for wind farm optimization, which is an approach that estimates the gradient of the AEP using Monte Carlo simulation, allowing for the consideration of an arbitrarily large number of atmospheric conditions. This method does not require that the atmospheric conditions be discretized, in contrast to the typical rectangular quadrature approximation of AEP. SGD is demonstrated using wind farms with square boundaries, considering cases with 25, 64, and 100 turbines, and the results are compared to a deterministic optimization approach. It is shown that SGD finds a larger optimal AEP in substantially less time than the deterministic counterpart as the number of wind turbines is increased.

## 1 Introduction

Wind farms are groups of wind turbines that harness the power in the atmospheric boundary layer to provide renewable energy. When a wind turbine absorbs energy from the air, the air downstream of the wind turbine has reduced power, which often reduces the power production of downstream turbines. This is known as the wake effect (Sanderse, 2009; Hasager et al., 2013). When a new wind power plant is to be constructed, optimal turbine locations are determined using engineering wind farm models (Annoni et al., 2018). Turbine positions are optimized to exploit the benefits of the local wind resource while avoiding energy losses from turbine wakes. In the wind turbine placement problem, atmospheric conditions, such as the inflow speed and direction, are assigned probability distributions according to measured data. By propagating these probability distributions through the engineering wake model, the annual energy production (AEP) can be estimated. The AEP is often computed using rectangular quadrature, dividing the relevant speeds and directions into equal-sized bins, then computing the expected AEP as the product of the power and probability of each bin, added together, then multiplied by the number of hours per year. It can be costly to compute the power associated with each speed and direction combination, as the number of these combinations can be large. This has given rise to studies seeking convergence of the AEP, proposing methods such as polynomial chaos expansion (Padrón et al., 2019; Murcia et al., 2015) or Bayesian quadrature (King et al., 2020), to avoid discretizing the input distributions into evenly spaced intervals. In this study, we present an approach for wind farm optimization that estimates the gradient of the AEP using Monte Carlo simulation. This does not require that the input be discretized at all, and allows for the consideration of an arbitrarily large number of atmospheric conditions.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

Stochastic gradient decent (SGD) is an optimization algorithm, commonly used in machine learning when selecting neural network weights (Ketkar, 2017). The algorithm approximates the gradient of the objective from a stochastic subset of the training data, following the gradient by a specified distance, then repeating the process.

The SGD algorithm is often enhanced to avoid oscillations caused by large changes in the gradient of the objective (Ruder, 2016). This includes methods to reuse previous gradient information (Qian, 1999), dampen oscillations (Riedmiller and Braun, 1993), or incorporate an estimate of the Hessian matrix (Moritz et al., 2016; Byrd et al., 2016; Liu et al., 2018; Najafabadi et al., 2017). Kingma and Ba (2014) introduced the Adam SGD algorithm, which reuses gradient evaluations and dampens oscillations, and is the SGD method we employ in this study. Interestingly, SGD is not often applied to problems with nonlinear constraints, although it can be fruitful to include nonlinear constraints in the context of training a machine learning algorithm. For example, when recognizing three-dimensional pictures of people, it can be useful to impose a constraint that any person's left arm should be close to the same length as their right arm (Márquez-Neila et al., 2017). Many frameworks have been proposed for constrained SGD, including the log-barrier function (Kervadec et al., 2019), penalty functions (Márquez-Neila et al., 2017), blending barrier and penalty functions (Kervadec et al., 2019), and Riemannian geometry (Roy and Harandi, 2017). In this study, we use a penalty term to transform the constrained problem into an unconstrained optimization. This term introduces sudden steep gradients in the optimization space, necessitating the use of smaller momentum parameters than are typically employed in the Adam SGD algorithm.

This study benchmarks the performance of the proposed SGD approach when compared to conventional gradient-based optimization. As a proxy for conventional gradient-based optimization, we examine the Scipy Sequential Least Squares Programming (SLSQP) optimization algorithm (Virtanen et al., 2020). This algorithm is based on a FORTRAN code written in 1994, and it could easily be argued that it would be more fair to compare SGD with a more modern deterministic optimization algorithm. We plan to explore this question in future work. We chose Scipy SLSQP because it is a widely used approach that is employed in many open-source wind farm optimization codes. The algorithms are compared by examining their performance when optimizing rectangular wind farm layouts of various sizes using the PyWake simulation software (Pedersen et al., 2019; Riva et al., 2020; Rodrigues et al., 2022; Ciavarra et al., 2022). PyWake uses automatic differentiation (Martins and Ning, 2021) to compute gradients using the autograd package (Maclaurin et al., 2015). Wind farms with 25, 64 and 100 turbines are used to benchmark the SGD approach when compared to a deterministic approach, and explore the effects of varying the length of the SGD optimization.

While there are some wind plant optimization studies that resemble our approach, we are not aware of any studies that have applied stochastic gradient descent to the wind farm optimization problem [although stochastic gradient descent has been applied to other problems in engineering optimization, e.g., De et al. (2020); Sivanantham and Gopalakrishnan (2022)]. Several wind farm optimization studies make use of gradient-based optimization techniques (Herbert-Acero et al., 2014). Feng and Shen (2015) present a random search approach, moving the wind turbines one by one using a greedy algorithm. Some studies have employed neural networks to forecast power production (Godinho and Castro, 2021), estimate local atmospheric conditions (Stengel et al., 2020), suggest control strategies (Najd et al., 2020), or optimize engineering wake models (Zhang

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

et al., 2021; Zhang and Zhao, 2022; Hussain et al., 2022), which all use SGD algorithms to the train parameters of the neural networks.

The remainder of the manuscript is the following. Section 2 outlines the SGD and deterministic optimization approaches used in this study. Section 3 details the wind farm optimization application cases examined. Section 4 discusses the results of these optimization comparisons. Section 5 provides conclusions and future research directions.

## 2 Methods

When deciding where to put wind turbines, the typical strategy is to maximize wind farm annual energy production (AEP) while ensuring turbines are within the prospective site and are not spaced too closely together. In this study, the corresponding optimization problem is posed as

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{y}}{\text{minimize}} \quad & \text{AEP}(\boldsymbol{x},\boldsymbol{y}) \\ \text{subject to} \quad & (x_i - x_j)^2 + (y_i - y_j)^2 \geq (N_D D)^2, \quad \forall i \neq j \\ & x_l \leq x_i \leq x_u \\ & y_l \leq y_i \leq y_u \,, \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the turbine horizontal and vertical locations, $x_l$ and $x_u$ are the lower and upper horizontal boundaries, $y_l$ and $y_u$ are the lower and upper vertical boundaries, $D$ is the rotor diameter, and $N_D$ is the minimum allowable spacing between turbines measured in rotor diameters. From this point forward, we will use a single variable to represent the x- and y-locations, $\boldsymbol{s} = \{\boldsymbol{x},\boldsymbol{y}\}$.

The AEP is defined as

$$\text{AEP}(\boldsymbol{s}) = 8760 \int_0^{2\pi} \int_0^{\infty} P(\boldsymbol{s}, u_\infty, \theta) \pi(u_\infty, \theta) du_\infty d\theta \,, \tag{2}$$

where $P$ is power, $\pi$ is probability, $u_\infty$ is the freestream velocity, and $\theta$ is the freestream direction.

The AEP is typically estimated through rectangular quadrature, where the freestream velocity and direction are discretized using evenly spaced intervals,

$$AEP(\boldsymbol{s}) \approx 8760 \sum_{d=1}^{D} \sum_{u=1}^{U} P(\boldsymbol{s}, \mathcal{U}_u, \theta_d) \rho(\mathcal{U}_u, \theta_d) \tag{3}$$

where $\boldsymbol{\mathcal{U}}$ is a vector of evenly spaced wind speeds, $\boldsymbol{\theta}$ is a vector of evenly spaced wind directions, and $\rho(\mathcal{U}_u, \theta_d)$ is a probability mass function.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

The AEP can also be estimated through Monte Carlo integration,

$$AEP(\boldsymbol{s}) \approx 8760 \frac{1}{K} \sum_{k=1}^{K} P(\boldsymbol{s}, u_\infty^{(k)}, \theta^{(k)}), \tag{4}$$

where $u_\infty^{(k)}$ and $\theta^{(k)}$ represent draw k of the probability distribution $\pi(u_\infty, \theta)$.

The associated AEP gradient can also be approximated through Monte Carlo simulation:

$$\frac{d}{d\boldsymbol{s}} AEP \approx 8760 \frac{1}{K} \sum_{k=1}^{K} \frac{d}{d\boldsymbol{s}} P(\boldsymbol{s}, u_\infty^{(k)}, \theta^{(k)}) \tag{5}$$

### 2.1 Stochastic Gradient Descent

The SGD algorithm is shown in Algorithm 1,

**Algorithm 1** Stochastic Gradient Descent

$\boldsymbol{m} \leftarrow \boldsymbol{0}$
$\boldsymbol{v} \leftarrow \boldsymbol{0}$
$\boldsymbol{s} \leftarrow \boldsymbol{s}_0$
for $i$ in $[1, 2, \ldots, T]$
do
$\boldsymbol{j} = -\frac{8760}{K} \sum_{k=1}^{K} \frac{\partial}{\partial \boldsymbol{s}} P(\boldsymbol{s}, u_\infty^{(k)}, \theta^{(k)}) + \alpha_i \frac{\partial \gamma}{\partial \boldsymbol{s}}$
$\boldsymbol{m} = \beta_1 \boldsymbol{m} - (1 - \beta_1)\boldsymbol{j}$
$\boldsymbol{v} = \beta_2 \boldsymbol{v} - (1 - \beta_2)\boldsymbol{j}^2$
$\hat{\boldsymbol{m}} = \frac{\boldsymbol{m}}{1-(\beta_1)^i}$
$\hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{1-(\beta_2)^i}$
$\boldsymbol{s} = \boldsymbol{s} - \eta_i \hat{\boldsymbol{m}} / \sqrt{\hat{\boldsymbol{v}}}$
$\eta_i = \mathcal{S}(\eta_0, \delta, i)$
$\alpha_i = \alpha_0 \frac{\eta_0}{\eta_i}$

where $\boldsymbol{s}_0$ are the initial turbine positions, $i$ is the iteration number, $\alpha_i$ is referred to as the constraint multiplier, $\gamma(\boldsymbol{s})$ is a penalty function, $P(\boldsymbol{s}, u_\infty^{(k)}, \theta^{(k)})$ is the wind farm power associated with inflow speed $u_\infty^{(k)}$ and $\theta^{(k)}$, $K$ is the number of samples employed in each SGD iteration, $\beta_1$ and $\beta_2$ are constants, $T$ is the number of SGD iterations, and $\eta_i$ is the learning rate.

We initially attempted this approach using the widely used default parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The parameters can be thought of as adding momentum to the moving averages of the gradient and squared gradient, $\boldsymbol{m}$ and $\boldsymbol{v}$. We found that these default values gave too much emphasis to gradients from the penalty function, launching the turbines away from the boundaries in a dramatic fashion. Instead, we suggest the parameters $\beta_1 = 0.1$ and $\beta_2 = 0.2$, which encode a shorter memory of the presence of the penalty. With these new default parameters, and the learning rate defined below, we observed successful convergence for a wide variety of test cases.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

The spacing between turbines is enforced using a penalty term,

$$\gamma_s = \sum_{\forall i, j>i} \min\left[(x_i - x_j)^2 + (y_i - y_j)^2 - (N_D D)^2, 0\right]. \tag{6}$$

Similarly, the distance outside of boundaries is enforced using a penalty term,

$$\gamma_b = \sum_{i=1}^{N_t} \left[\max\left(x_i - x_{ub}, 0\right)^2 + \max\left(x_{lb} - x_i, 0\right)^2 + \max\left(y_i - y_{ub}, 0\right)^2 + \max\left(y_{lb} - y_i, 0\right)^2\right], \tag{7}$$

where $N_t$ is the number of wind turbines.

The total penalty, $\gamma$, is defined as the sum of these two penalty terms,

$$\gamma(\boldsymbol{s}) = \gamma_s(\boldsymbol{s}) + \gamma_b(\boldsymbol{s}). \tag{8}$$

The gradient of the penalty term, $\gamma$, is scaled before being added to the negative gradient of the AEP using the scaling factor, $\alpha_i$. The gradients of $\gamma$ are computed using finite differences while the gradient of the AEP is computed using automatic differentiation.

In Algorithm 1, the learning rate ($\eta_i$), constraint multiplier ($\alpha_i$), number of SGD iterations ($T$), and the samples per SGD iteration ($K$) are all free parameters. These parameters can be optimized to perform well for individual wind farm optimization problems. But there is no guarantee that these particular parameters will perform well for other wind farm problems—and this meta-optimization can be expensive. In the machine learning community, these parameters are sometimes optimized using evolutionary, grid search, or Bayesian optimization approaches (Alibrahim and Ludwig, 2021). In addition, it is common to schedule the learning rate to decay as the optimization proceeds (You et al., 2019; Denkowski and Neubig, 2017).

We propose a method for setting free parameters to ensure that all units are consistent. The only free parameters we manually set are the number of optimization iterations and the number of power samples per iteration. The optimization generally becomes more accurate and more expensive as these parameters increase, and users are free to balance this tradeoff as they see fit. Our formulation does not guarantee that all intermediate solutions satisfy the constraints, especially in the beginning of the optimization. The constraint multiplier begins on a comparable scale to the AEP, and is scheduled to increase so that the constraint gradients overwhelm the AEP gradients as the optimization progresses. The number of iterations, $T$, can be based on a prescribed computational budget.

The learning rate, $\eta_i$, can be interpreted as converting $\hat{\boldsymbol{m}}/\sqrt{\hat{\boldsymbol{v}}}$ (with unity units) to distance (units of m). In this study, the learning rate is scheduled to decay according to

$$\begin{matrix} S(t=0) = \eta_0 \\ S(t=T) = \eta_T \end{matrix}, \tag{9}$$

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

where $T$ is the number of optimization iterations, $\eta_0$ is the initial learning rate, and $\eta_T$ is the scheduled final learning rate. This final learning rate can be thought of as a solution tolerance for the design variables. In this study, we set $\eta_T = 0.01\text{m}$.

The initial learning rate, $\eta_0$, is based on a length scale parameter, $L$, which corresponds to a reasonable initial step size for the optimization. By setting the intial learning rate according to

$$\eta_0 = L = D/5\,, \tag{10}$$

where $D$ is the turbine rotor diameter, we encourage the turbines to move at most $L$ every optimization iteration.

The learning rate is scheduled to decay as

$$S(\eta_0, \delta, t) = \eta_0 \prod_{i=1}^{t} \frac{1}{1+i\delta}\,, \tag{11}$$

where $\delta$ is a parameter that controls the learning rate length, such that the final learning rate is $\eta_T$. The parameter $\delta$ is numerically set as

$$\delta(\eta_0, \eta_T, T) = \underset{\delta}{\operatorname{argmin}} |\eta_T - S(\eta_0, \delta, T)|\,, \tag{12}$$

The constraint multiplier, $\alpha_i$, can be interpreted as converting the gradient of constrained square distances (in units of m) to AEP gradients. The initial constraint multiplier, $\alpha_0$, is set as the mean absolute AEP gradient divided by the length scale, $L$, so that the separation constraint has a similar scale to AEP gradients,

$$\alpha_0 = \frac{\text{mean}[|\nabla AEP(\boldsymbol{s}_0)|]}{L}\,, \tag{13}$$

where $\text{mean}[|\nabla AEP(\boldsymbol{s}_0)]$ is the mean of the absolute AEP gradient of the initial guess with respect to each component of the gradient. During each iteration, the constraint multiplier, $\alpha_i$, is scheduled to increase based on the inverse of the learning rate,

$$\alpha_i = \alpha_0 \frac{\eta_0}{\eta_i}\,. \tag{14}$$

The wind rose samples, $(u_\infty^{(i)}, \theta^{(i)}) \sim \pi(u_\infty, \theta)$, are randomly selected based on the direction frequency and direction-specific Weibull shape and scale parameters. Note that the tilde ($\sim$) denotes a shared probability distribution. After a direction is sampled, the wind speed is sampled as a continuous weibull distributed random variable,

$$u_\infty(\theta) \sim \mathcal{W}\left[u_\infty, A(\theta), k(\theta)\right]\,, \tag{15}$$

where the probability density of the Weibull distribution, $\mathcal{W}$, is given by

$$\mathcal{W}(u_\infty, A, k) = \frac{k}{A}\left(\frac{u_\infty}{A}\right)^{k-1} \exp\left[-\left(\frac{u_\infty}{A}\right)^{k}\right]\,. \tag{16}$$

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

### 2.2 Deterministic Approach

A deterministic SLSQP optimization was selected to act as a benchmark to the SGD approach. The upper and lower boundaries of the farm are passed to the optimizer as bounds. The spacing between each set of turbines are passed to the optimizer as individual inequality constraints,

$$C_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2 - (2D)^2 \quad \forall i, j > i\,. \tag{17}$$

We used the Scipy implementation of the algorithm (Virtanen et al., 2020), with 200 maximum iterations and a tolerance of $10^{-3}$, optimizing the AEP in units of kWh subject to the spacing constraint in Equation 17 in units of $m^2$. In each optimization iteration, the AEP, and the corresponding gradient, is computed using rectangular quadrature as described in Equation 3, using 360 wind direction bins and 23 wind speed bins, resulting in 8280 power evaluations.

## 3 Application

We applied the optimization approaches discussed above to optimize wind power plants of various sizes. Each farm consists of turbines with 70-meter hub heights, 80-meter rotor diameters, and 2-megawatt rated powers. Power gradients were computed directly from PyWake using automatic differentiation. The power of each turbine is estimated by a combination of velocity deficits predicted by the Bastankhah Gaussian wake model (Bastankhah and Porté-Agel, 2014) using the default parameters in the PyWake tool and the squared sum superposition (Pedersen et al., 2019; DTU Wind Energy Systems, 2022). We require each turbine to be spaced at minimum two rotor diameters apart ($N_D = 2$). All power plants considered in this study have square boundaries so that, in a grid, the turbines would be spaced five rotor diameters apart ($\Delta = 5$), such that the edge turbines are placed on the boundaries of the wind farm. The boundaries are determined as

$$\begin{aligned} x_l &= 0 \\ y_l &= 0 \\ x_u &= D\sqrt{N_t}\Delta \\ y_u &= D\sqrt{N_t}\Delta \end{aligned}\,. \tag{18}$$

The wind rose, visualized in Figure 1, is based on PyWake's Lillgrund example site. A probability mass function is assigned to different direction bins. Each direction bin is associated with Weibull scale and shape parameters describing the distributions of wind speeds within the sector. Each direction bin is 30 degree wide. The continuous probability density function $\pi(u_\infty, \theta)$ is approximated as $\rho(\theta)\pi(u_\infty|\theta)$, where $\rho$ is the previously mentioned probability mass function, linearly interpolated across one-degree bins, and $\pi(u_\infty|\theta)$ is parameterized by direction-specific Weibull shape and scale parameters that are also linearly interpolated from the provided data. With this formulation, the likelihood of different wind directions is provided as a probability mass function, $\rho(\theta)$. This probability mass is used as weights passed to the Numpy `choice` function (Harris et al., 2020),

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

allowing the wind direction to be sampled as a discrete random variable. We note that this formulation could be extended to a fully continuous formulation by drawing the direction samples from the inverse of an empirical cumulative direction density function.

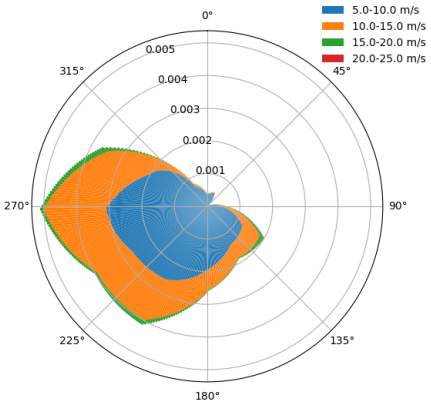


**Figure 1.** Lillgrund wind speed and direction probability mass function with 360 direction bins.

In all wind farm optimization problems considered, the directions are discretized from 0 to 360 degrees, with one-degree increments. In the deterministic formulation, the discretized wind speed ranges from 3-25 m/s and is divided using increments of 1 m/s.

Figure 2 shows the measured computational cost of computing AEP gradients, using the rectangular wind farm described in this study, with different wind farm sizes, using grid layouts with an average spacing of five rotor diameters between each turbine. The minimum measured time was reported from five identical runs on the DTU Sophia supercomputer (Technical University of Denmark, 2019). For small numbers of turbines, evaluating 100 wind rose samples is about as expensive as evaluating 5 samples. This scaling changes as the wind farm grows in size, and it gradually becomes more expensive to sample the wind rose. In our study, we selected 50 samples for every SGD iteration.

The optimization algorithms were timed based on the time elapsed between the first and final optimization gradient evaluations. Each optimization case was run on second generation AMD EPYC 7302 processors. Each optimization case was run sequentially for the sake of simplicity, although we acknowledge that there are several opportunities for parallelization in future work.

## 4 Results and Discussion

The performance of SGD is compared to the deterministic counterpart, considering wind farms with 25, 64 and 100 turbines, using 20 different initial starting conditions to obtain statistically significant results. The AEP and constraint violation of each optimization solution are plotted as a function of time elapsed during the optimization in Figure 3. Note that these plots show the

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


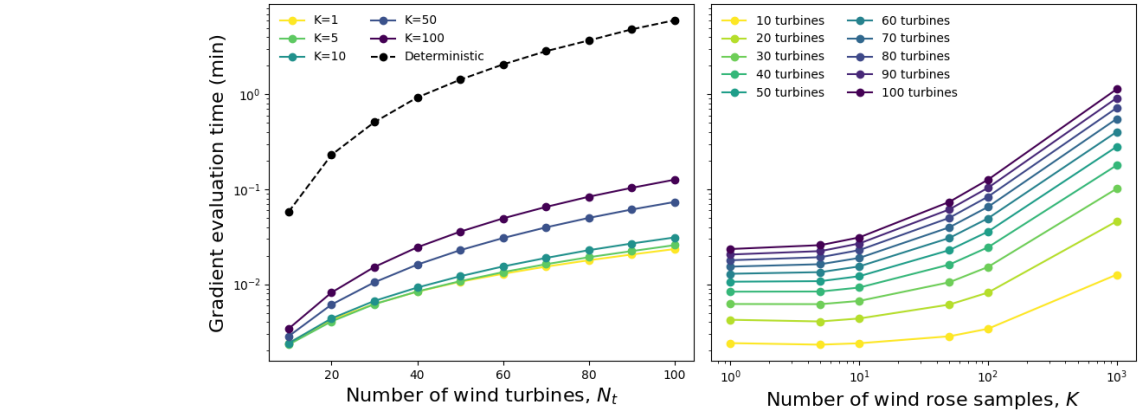


**Figure 2.** Computational time associated with computing the gradient for various wind farm sizes and wind rose sampling strategies. The left panel compares the cost of computing AEP gradients when using different numbers of Monte Carlo samples with the cost of the full factorial wind rose (8,280 samples) for farms with various numbers of wind turbines. The right panel compares the cost of Monte Carlo estimates of the AEP gradient for different numbers of samples of the atmospheric conditions and turbines in the wind farm.

optimization history by plotting the deterministic AEP associated with each gradient evaluation called for by each optimization case. Since the time is measured relative to the first gradient evaluation, the first gradient evaluation is associated with a time of zero, which is not shown in the logarithmic time axis. When considering 25 wind turbines, the solutions found using SGD and deterministic approaches are comparable. When considering larger wind farms (with 64 and 100 turbines), the SGD approach finds optimum AEP in considerably less time than the deterministic counterpart; and the AEPs found using SGD tend to be larger than those found using the deterministic approach. As the proposed SGD formulation does not offer an automatic way to set the $T$ parameter, results are shown for different values of $T$. When $T$ is increased, the optimizer finds solutions with larger AEPs, with a computational cost that is approximately proportional to $T$. As the optimization progresses, $\alpha_i$ becomes large (approaching 100 as the learning rate approaches 0.01), and the gradients of the AEP, which are time-consuming to compute, are overwhelmed by the gradients of the penalty, which take very little time to compute. In production runs of this approach, it could be useful to alter Algorithm 1 to remove the AEP gradients from the formulation for sufficiently large $\alpha_i$.

The final layouts associated with one of the random initial conditions used in the 100 turbine analysis, when $T = 5{,}000$ iterations, is shown in Figure 4. The SGD approach generally identifies solutions with the majority of turbines packed into the side boundaries and a somewhat regular grid in the interior of the farm. The deterministic algorithm also packed turbines into the edges of the farm, although not as many turbines were packed into the East and West boundaries as in the SGD results. The layouts found using the SGD approach tend to have interior turbines that generally appear to be more aligned in the North-South direction than in the deterministic solutions.

The results of the 25-, 64-, and 100-turbine wind farm optimization cases are summarized in Table 1. The mean time, mean constraint violation, and mean and standard deviation of the AEP are reported with respect to the 20 random initial starting

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


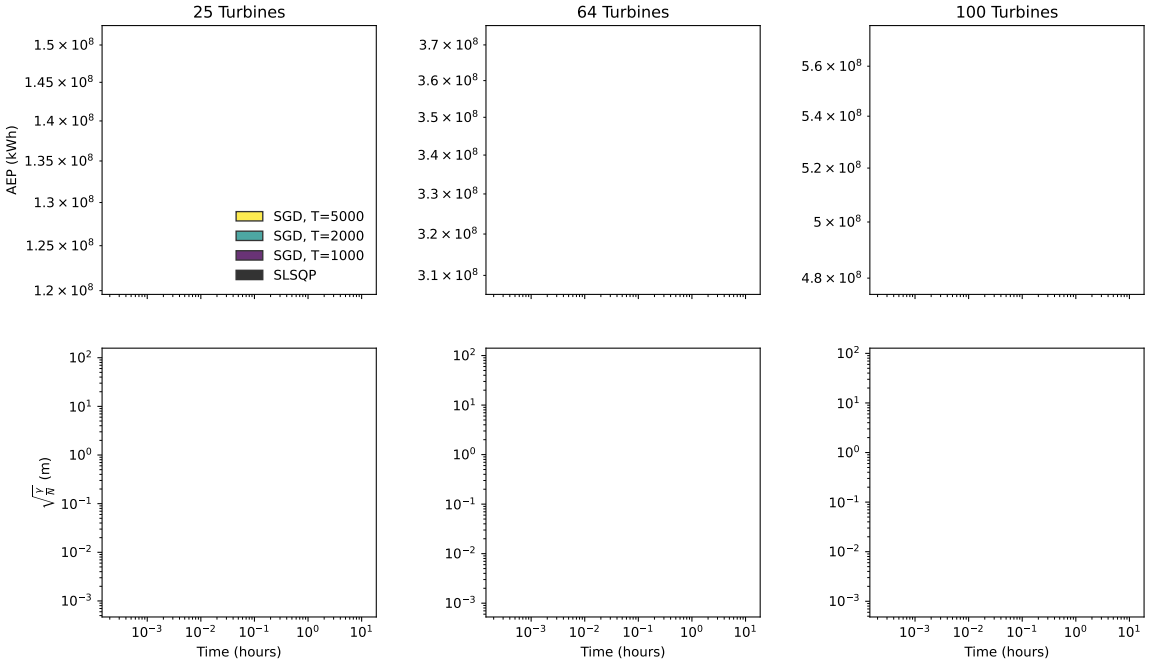


**Figure 3.** Optimization results associated with SGD and SLSQP for a 25-turbine wind farm, using 20 random initial starting conditions. The AEP (top panel) and penalty (bottom panel) are plotted as a function of the time elapsed during the optimization. The SLSQP results are shown in black. The SGD results associated with $T = 1,000$, $2,000$, and $5,000$ iterations are shown in purple, blue, and yellow, respectively.

conditions. The constrain violation is reported as $\sqrt{\gamma/N_t}$ to quantify the mean length of the constraint violations of each turbine. The final constraint violations can be reduced by lowering the $\eta_T$ parameter. The SGD approach takes considerably less time to find a solution than the deterministic approach for the larger wind farms considered. With $T$ =2,000 iterations, the SGD approach yielded an average of 0.2% AEP improvement in the 100-turbine case and the computational time of the optimization is reduced by factors of 7 and 20 in the 64- and 100-turbine cases, respectively, when compared to the deterministic counterparts.

The optimization results presented in this study used 50 power samples per iteration ($K = 50$). We found this to produce high-quality results without incurring unacceptable computational expense. Figure 5 shows the behavior of the SGD approach associated with different different values of $K$, considering 100 turbines with 5,000 scheduled optimization iterations. As $K$ increases, the optimization finds solutions with larger AEPs. There is a small increase in time elapsed and a large increase in the final AEP between the $K = 5$ and $K = 50$ cases, while there is a large increase in time elapsed and a small increase in the final AEP between $K = 50$ and $K = 200$. As $K$ increases, we expect the maximum AEP to reach a plateau and the time

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


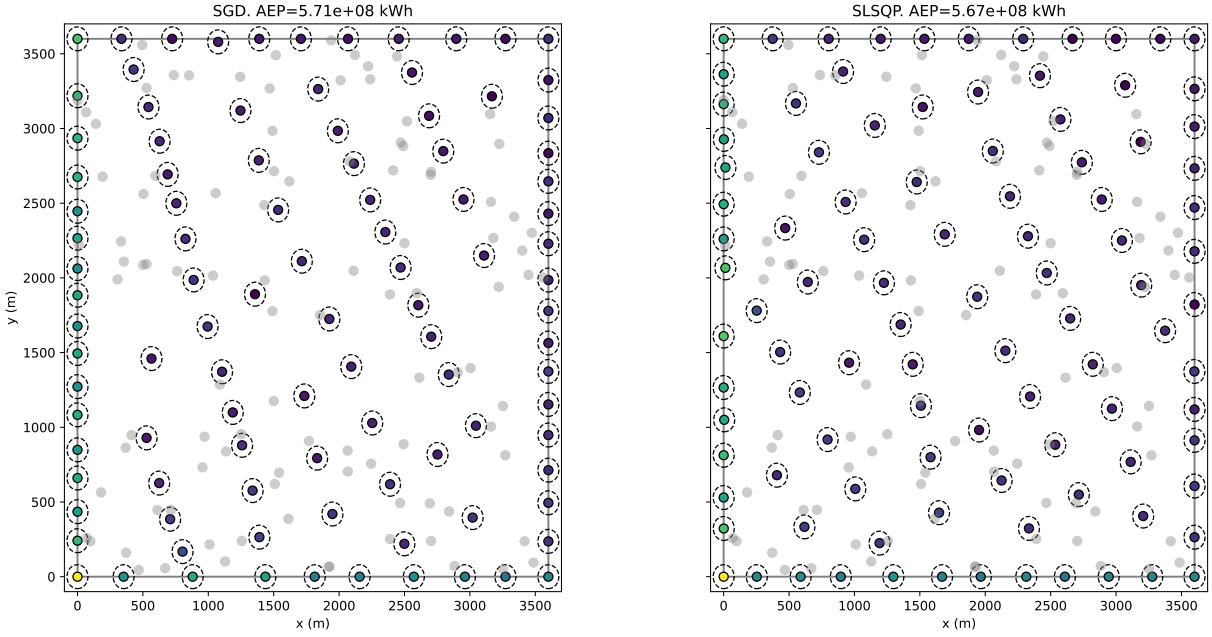


**Figure 4.** The final layouts found using SGD (left) and SLSQP (right) using one of the random initial layouts examined in the 100-turbine wind farm case for $T = 5,000$ iterations. The wind farm boundaries are shown as grey lines. The initial turbine layouts are shown as grey points. The final turbine layouts are shown as filled circles, where brighter fill colors indicate larger annual power production of the individual turbines. The spacing constraint is visualized using hollow circles with dashed lines and a radius of one rotor diameter.

**Table 1.** Results of SGD and deterministic optimizations for various wind farm sizes. Each optimization case is run using 20 random initial starting conditions, and the mean and standard deviation are reported with respect to these 20 initial points. The SGD results are associated with $T = 2,000$ iterations.

| $N_t$ | Case | Mean Time (minutes) | Mean AEP (kWh) | AEP Standard Deviation (kWh) | Mean $\sqrt{\gamma/N_t}$ (m) |
|---|---|---|---|---|---|
| 25 | Deterministic | 2 | 1.496e+08 | 3.069e+05 | 0.000e+00 |
| | SGD | 4 | 1.500e+08 | 4.462e+05 | 3.170e-03 |
| 64 | Deterministic | 110 | 3.691e+08 | 5.633e+05 | 0.000e+00 |
| | SGD | 15 | 3.699e+08 | 4.901e+05 | 2.396e-03 |
| 100 | Deterministic | 608 | 5.674e+08 | 9.975e+05 | 0.000e+00 |
| | SGD | 30 | 5.686e+08 | 7.720e+05 | 2.742e-03 |

and memory required to increase indefinitely. In future work, we plan to explore scheduling $K$ to increase as the optimization progresses.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


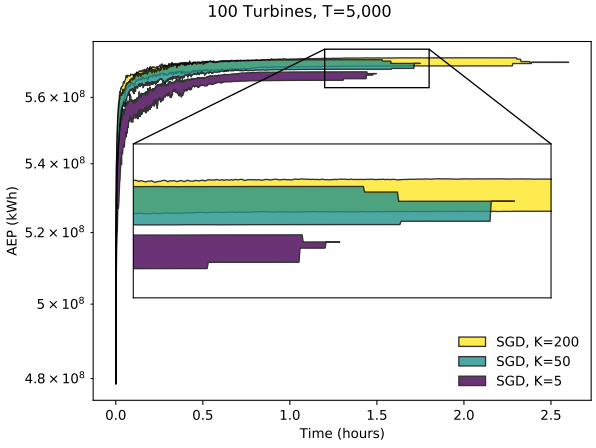


**Figure 5.** Optimization results associated with SGD for a 100-turbine wind farm using 20 random initial starting conditions. The upper and lower bounds of the results are plotted as a function of the optimization iteration number. The SGD results associated with $K = 5$, 50, and 200 iterations are shown in purple, blue, and yellow, respectively.

This study used an exotic learning rate scheduler. We tried several schedulers, and observed this one to be the best at finding sufficiently large AEP solutions that reasonably satisfied the imposed constraints. Figure 6 shows the behavior of the SGD algorithm associated with the presented learning rate scheduler, referred to here as the product scheduler; as well as an exponential and a linear decay scheduler. The exponential scheduler quickly diminishes the learning rate, causing the SGD algorithm to become stuck in local minima. The linear transition from large to lower learning rates prevents the SGD algorithm from having sufficient time to follow enlarged constraint gradients. It is possible that the algorithm could be improved by using separate schedulers for the learning rate and constraint multiplier. For instance, it might be more effective to use a linear scheduler to decrease the learning rate and an exponential scheduler to increase the constraint multiplier. We leave this question for future work.

## 5 Conclusions

SGD is a promising optimization tool for wind farm design. Instead of evaluating all anticipated atmospheric conditions during every optimization iteration, SGD randomly samples the defined distributions of atmospheric conditions, resulting in substantially reduced computational time required for each optimization iteration. The total optimization time can be scheduled according to a prescribed computational budget. The presented formulation allows for continuous resolution of uncertain variables, eliminating the need to choose a discretization resolution of atmospheric conditions, such as the wind speed and direction. This technique does not become exponentially more expensive as a greater number of uncertain parameters is included, allowing for consideration of other atmospheric conditions, such as turbulence intensity, air density, veer, and shear (Saint-Drenan et al., 2020; Duc et al., 2019).

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


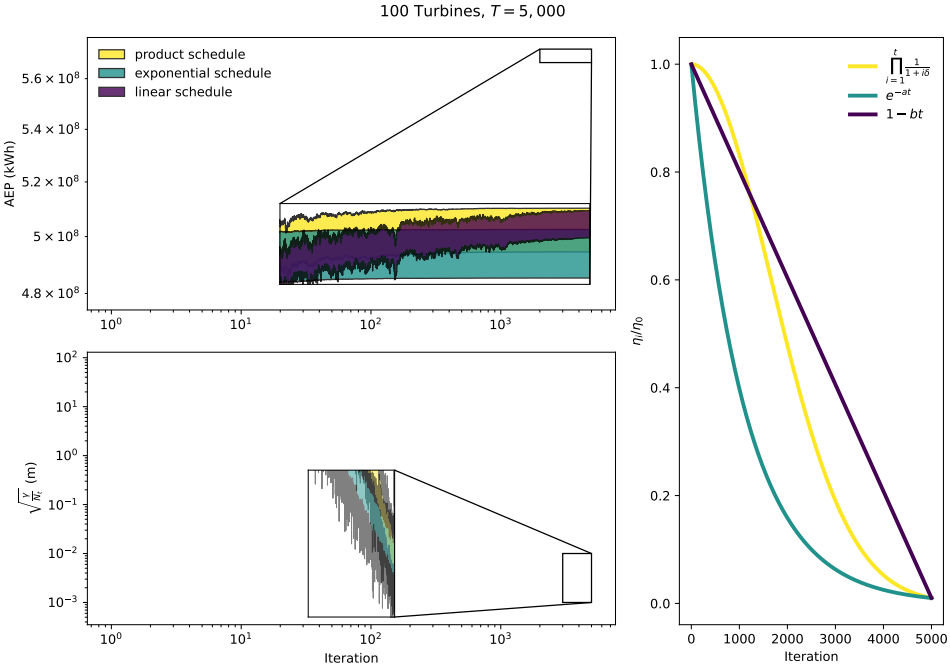


**Figure 6.** Influence of the learning rate scheduler on the SGD results. The product scheduler is shown in yellow. The exponential scheduler is shown in blue. The linear scheduler is shown in purple. The AEP (upper left panel), constraint penalty function (bottom left panel), and the learning rate decay (right panel) are plotted as a function of the number of optimization iterations. The optimization iteration is denoted as $t$ in the legend of the right panel.

The presented SGD approach was shown to become more effective than a deterministic counterpart as the number of wind turbines increased. SGD yielded slightly higher AEPs than the deterministic approach in substantially reduced computational time. The time required to optimize wind farm layouts can be a major bottleneck in corporate workflows, and the time savings associated with the SGD approach allows engineers to access optimization results sooner than a conventional approach. If the inflow conditions were discretized using extremely small bins, or if several atmospheric conditions were to be considered, we expect that the SGD approach would perform the optimization even faster and more effectively than the deterministic approach.

The SGD approach is a simple framework that is well suited large-scale stochastic wind power plant design optimization challenges. Future work includes: exploring separate schedulers for the constraint multiplier and learning rate, scheduling the number of Monte Carlo samples, $K$, to increase as the optimization proceeds, and examining the impacts of discretizing the atmospheric conditions.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

*Author contributions.* JQ, PER, and KD designed the experiments. JQ, PER, MMP, RVP, and KD developed the problem formulation. JQ developed the SGD formulation. JQ and MMP performed the simulations. JQ, MMP, and RVP prepared the manuscript with contributions from all co-authors.

*Acknowledgements.* The authors gratefully acknowledge the computational and data resources provided on the Sophia HPC Cluster at the Technical University of Denmark, DOI: 10.57940/FAFC-6M81

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


eawe european academy of wind energy
Open Access WIND ENERGY SCIENCE DISCUSSIONS

## References

H. Alibrahim and S. A. Ludwig. Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. In *2021 IEEE Congress on Evolutionary Computation (CEC)*, pages 1551–1559, 2021. https://doi.org/10.1109/CEC45853.2021.9504761.

J. Annoni, P. Fleming, A. Scholbrock, J. Roadman, S. Dana, C. Adcock, F. Porte-Agel, S. Raach, F. Haizmann, and D. Schlipf. Analysis of control-oriented wake modeling tools using lidar field results. *Wind Energy Science*, 3(2):819–831, 2018.

M. Bastankhah and F. Porté-Agel. A new analytical model for wind-turbine wakes. *Renewable energy*, 70:116–123, 2014.

R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

A. W. Ciavarra, R. V. Rodrigues, K. Dykes, and P.-E. Réthoré. Wind farm optimization with multiple hub heights using gradient-based methods. In *Journal of Physics: Conference Series*, volume 2265, page 022012. IOP Publishing, 2022.

S. De, J. Hampton, K. Maute, and A. Doostan. Topology optimization under uncertainty using a stochastic gradient-based approach. *Structural and Multidisciplinary Optimization*, 62(5):2255–2278, 2020.

M. Denkowski and G. Neubig. Stronger baselines for trustable results in neural machine translation. *arXiv preprint arXiv:1706.09733*, 2017.

DTU Wind Energy Systems. Pywake. https://gitlab.windenergy.dtu.dk/TOPFARM/PyWake, commit 3a61c8aec34a2505d0b460da2ad831aeaf46802b, 2022.

T. Duc, O. Coupiac, N. Girard, G. Giebel, and T. Göçmen. Local turbulence parameterization improves the jensen wake model and its implementation for power optimization of an operating wind farm. *Wind Energy Science*, 4(2):287–302, 2019.

J. Feng and W. Z. Shen. Solving the wind farm layout optimization problem using random search algorithm. *Renewable Energy*, 78:182–192, 2015.

M. Godinho and R. Castro. Comparative performance of ai methods for wind power forecast in portugal. *Wind Energy*, 24(1):39–53, 2021.

C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del R'ıo, M. Wiebe, P. Peterson, P. G'erard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. https://doi.org/10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

C. B. Hasager, L. Rasmussen, A. Peña, L. E. Jensen, and P.-E. Réthoré. Wind farm wake: The horns rev photo case. *Energies*, 6(2):696–716, 2013.

J. F. Herbert-Acero, O. Probst, P.-E. Réthoré, 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.

M. N. Hussain, N. Shaukat, A. Ahmad, M. Abid, A. Hashmi, Z. Rajabi, and M. A. U. R. Tariq. Micro-siting of wind turbines in an optimal wind farm area using teaching–learning-based optimization technique. *Sustainability*, 14(14):8846, 2022.

H. Kervadec, J. Dolz, J. Yuan, C. Desrosiers, E. Granger, and I. B. Ayed. Constrained deep networks: Lagrangian optimization via log-barrier extensions. *arXiv preprint arXiv:1904.04205*, 2019.

N. Ketkar. Stochastic gradient descent. In *Deep learning with Python*, pages 113–132. Springer, 2017.

R. King, A. Glaws, G. Geraci, and M. S. Eldred. A probabilistic approach to estimating wind farm annual energy production with bayesian quadrature. In *AIAA Scitech 2020 Forum*, page 1951, 2020.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


J. Liu, Y. Rong, M. Takác, and J. Huang. On the acceleration of l-bfgs with second-order information and stochastic batches. *arXiv preprint arXiv:1807.05328*, 2018.

D. Maclaurin, D. Duvenaud, and R. P. Adams. Autograd: Effortless gradients in numpy. In *ICML 2015 AutoML workshop*, volume 238, 2015.

P. Márquez-Neila, M. Salzmann, and P. Fua. Imposing hard constraints on deep networks: Promises and limitations. *arXiv preprint arXiv:1706.02025*, 2017.

J. R. Martins and A. Ning. *Engineering design optimization*. Cambridge University Press, 2021.

P. Moritz, R. Nishihara, and M. Jordan. A linearly-convergent stochastic l-bfgs algorithm. In *Artificial Intelligence and Statistics*, pages 249–258. PMLR, 2016.

J. Murcia, P.-E. Réthoré, A. Natarajan, and J. D. Sørensen. How many model evaluations are required to predict the aep of a wind power plant? In *Journal of Physics: Conference Series*, volume 625, page 012030. IOP Publishing, 2015.

M. M. Najafabadi, T. M. Khoshgoftaar, F. Villanustre, and J. Holt. Large-scale distributed l-bfgs. *Journal of Big Data*, 4(1):1–17, 2017.

A. H. Najd, G. Goksu, and H. F. Hammood. Pitch angle control using neural network in wind turbines. In *IOP Conference Series: Materials Science and Engineering*, volume 928, page 022118. IOP Publishing, 2020.

A. S. Padrón, J. Thomas, A. P. Stanley, J. J. Alonso, and A. Ning. Polynomial chaos to efficiently compute the annual energy production in wind farm layout optimization. *Wind Energy Science*, 4(2):211–231, 2019.

M. M. Pedersen, P. van der Laan, M. Friis-Møller, J. Rinker, and P.-E. Réthoré. Dtuwindenergy/pywake: Pywake. Feb 2019. https://doi.org/10.5281/zenodo.2562662.

N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE international conference on neural networks*, pages 586–591. IEEE, 1993.

R. Riva, J. Liew, M. Friis-Møller, N. Dimitrov, E. Barlas, P.-E. Réthoré, and A. Beržonskis. Wind farm layout optimization with load constraints using surrogate modelling. In *Journal of Physics: Conference Series*, volume 1618, page 042035. IOP Publishing, 2020.

R. V. Rodrigues, M. Friis-Møller, K. Dykes, N. Pollini, and M. Jensen. A surrogate model of offshore wind farm annual energy production to support financial valuation. In *Journal of Physics: Conference Series*, volume 2265, page 022003. IOP Publishing, 2022.

S. K. Roy and M. Harandi. Constrained stochastic gradient descent: The good practice. In *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8, 2017. https://doi.org/10.1109/DICTA.2017.8227420.

S. Ruder. An overview of gradient descent optimization algorithms, 2016. URL https://arxiv.org/abs/1609.04747.

Y.-M. Saint-Drenan, R. Besseau, M. Jansen, I. Staffell, A. Troccoli, L. Dubus, J. Schmidt, K. Gruber, S. G. Simões, and S. Heier. A parametric model for wind turbine power curves incorporating environmental conditions. *Renewable Energy*, 157:754–768, 2020.

B. Sanderse. Aerodynamics of wind turbine wakes-literature review. 2009.

G. Sivanantham and S. Gopalakrishnan. Stochastic gradient descent optimization model for demand response in a connected microgrid. *KSII Transactions on Internet and Information Systems (TIIS)*, 16(1):97–115, 2022.

K. Stengel, A. Glaws, D. Hettinger, and R. N. King. Adversarial super-resolution of climatological wind and solar data. *Proceedings of the National Academy of Sciences*, 117(29):16805–16815, 2020.

Technical University of Denmark. Sophia HPC Cluster, 2019. URL https://dtu-sophia.github.io/docs/.

P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng,

https://doi.org/10.5194/wes-2022-104
Preprint. Discussion started: 9 November 2022


E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. https://doi.org/10.1038/s41592-019-0686-2.

K. You, M. Long, J. Wang, and M. I. Jordan. How does learning rate decay help modern neural networks? *arXiv preprint arXiv:1908.01878*, 2019.

J. Zhang and X. Zhao. Wind farm wake modeling based on deep convolutional conditional generative adversarial network. *Energy*, 238: 121747, 2022.

Z. Zhang, C. Santoni, T. Herges, F. Sotiropoulos, and A. Khosronejad. Time-averaged wind turbine wake flow field prediction using autoencoder convolutional neural networks. *Energies*, 15(1):41, 2021.