Stat
The ISI's Journal for the Rapid
Dissemination of Statistics Research
(wileyonlinelibrary.com) DOI: 10.100X/sta.0000

Adaptively-Tuned Particle Swarm Optimization with Application to Spatial Design

Received ; Accepted

Particle swarm optimization (PSO) algorithms are a class of heuristic optimization algorithms that are especially attractive for complex optimization problems. We propose using PSO in order to solve spatial design problems, e.g. choosing new locations to add to an existing monitoring network. Additioonally we introduce a new class of PSO algorithms, called adaptively-tuned PSO, that perform well in a wide variety of circumstances. In order to illustrate these algorithms, we apply them to a common spatial design problem: choosing new locations to add to an existing monitoring network. Specifically, we consider a network in the Houston, TX area for monitoring ambient ozone levels, which have been linked to out-of-hostpital cardiac arrest rates (Ensor et al., 2013).

Keywords: optimization; particle swarm; geostatistics; kriging; optimal design, spatial prediction

1. Introduction

PSO refers to a large class of heuristic optimization algorithms that that rely on an analogy with animal flocking behavior and are typically more robust than many alternatives (Clerc & Kennedy, 2002; Blum & Li, 2008; Clerc, 2010). This robustness makes them attractive for more difficult optimization problems, especially when near optimal solutions are tolerable. We introduce a new class of PSO algorithms, called adaptively tuned PSO (AT-PSO), which exploit an analogy with a class of adaptive Markov chain Monte Carlo algorithms in order to tune a crucial parameter of the PSO algorithm adaptively based on the state of the particle swarm. We show that the resulting algorithms tend to be superior to many PSO alternatives. Additionally, we illustrate that PSO algorithms are a useful tool for spatial design problems by applying several of them to choosing a set of new monitoring locations for ozone in Harris County, Texas, where Houston is located. We compare to a class of genetic algorithms for the same task.

The remainder of the paper is as follows. Section 2 introduces various PSO and AT-PSO algorithms, and briefly discusses the results of an extended simulation study comparing them. Section 3 introduces a generic spatial design problem and the Houson area ozone problem as an instance of that problem, and compares several PSO algorithms and some alternatives to solving the problem. Finally, Section 4 discusses our results and concludes.

*Email:

Prepared using staauth.cls [Version: 2012/05/12 v1.00]

2. Particle swarm optimization

We briefly describe PSO here; refer to Blum & Li (2008) for a comprehensive introduction, Clerc (2010) for details, and Clerc (2011) for good default versions of the algorithm. Suppose that we wish to minimize some objective function $Q(\boldsymbol{\theta}) : \Re^D \to \Re$. Let $i = 1, 2, \ldots, n$ index a set of particles over time, $k = 1, 2, \ldots, K$, where in every period each particle consists of a location $\boldsymbol{\theta}_i(k) \in \Re^D$, a velocity $\boldsymbol{v}_i(k) \in \Re^D$, a personal best location $\boldsymbol{p}_i(k) \in \Re^D$, and a group best location $\boldsymbol{g}_i(k) \in \Re^D$. Here we mean "best" in the sense of minimizing Q, so $Q(\boldsymbol{p}_i(k)) \leq Q(\boldsymbol{\theta}_i(l))$ for any $k \geq l$. The group best location is defined with respect to some neighborhood $\mathcal{N}_i$ of particle i; that is, $\boldsymbol{g}_i(k) = \arg\min_{\{\boldsymbol{p}_j(k) | j \in \mathcal{N}_i\}} Q(\boldsymbol{p}_j(k))$. In the simplest case where the entire swarm is the neighborhood of each particle, $\boldsymbol{g}_i(k) \equiv \boldsymbol{g}(k) = \arg\min_{\{\boldsymbol{p}_j(k) | j \in 1:n\}} Q(\boldsymbol{p}_j(k))$. The generic PSO algorithm updates each particle i as follows:

$$
\begin{aligned}
\boldsymbol{v}_i(k+1) &= \omega \boldsymbol{v}_i(k) + \phi_1 \boldsymbol{r}_{1i}(k) \circ \{\boldsymbol{p}_i(k) - \boldsymbol{\theta}_i(k)\} + \phi_2 \boldsymbol{r}_{2i}(t) \circ \{\boldsymbol{g}_i(k) - \boldsymbol{\theta}_i(k)\}, \\
\boldsymbol{\theta}_i(k+1) &= \boldsymbol{\theta}_i(k) + \boldsymbol{v}_i(k+1), \\
\boldsymbol{p}_i(k+1) &= \begin{cases} \boldsymbol{p}_i(k) & \text{if } \; Q(\boldsymbol{p}_i(k)) \leq Q(\boldsymbol{\theta}_i(t+1)) \\ \boldsymbol{\theta}_i(k+1) & \text{otherwise,} \end{cases} \\
\boldsymbol{g}_i(k+1) &= \arg \min_{\{\boldsymbol{p}_j(k+1) | j \in \mathcal{N}_i\}} Q(\boldsymbol{p}_j(k+1)),
\end{aligned} \tag{1}
$$

where $\circ$ denotes the Hadamard product (element-wise product), $\boldsymbol{r}_{1i}(k)$ and $\boldsymbol{r}_{2i}(k)$ are each vectors of D random variates independently generated from the $U(0,1)$ distribution, and $\omega > 0$, $\phi_1 > 0$, and $\phi_2 > 0$ are user-defined parameters. The term $\omega \boldsymbol{v}_i(k)$ controls the particle's tendency to keep moving in the direction it is already going, so ω is called the inertia parameter. Similarly $\phi_1 \boldsymbol{r}_{1i}(k) \circ \{\boldsymbol{p}_i(k) - \boldsymbol{\theta}_i(k)\}$ controls the particle's tendency to move towards its personal best location while $\phi_2 \boldsymbol{r}_{2i}(k) \circ \{\boldsymbol{g}_i(k) - \boldsymbol{\theta}_i(k)\}$ controls its tendency to move toward its group best location, so ϕ_1 and ϕ_2 are called the cognitive correction factor and social correction factor, respectively (Blum & Li, 2008). This version of PSO is equivalent to Clerc & Kennedy (2002)'s constriction type I particle swarm, though there are many other variants. One default choice sets $\omega = 0.7298$ and $\phi_1 = \phi_2 = 1.496$ (Clerc & Kennedy, 2002) while another sets $\omega = 1/(2 \ln 2) \approx 0.721$ and $\phi_1 = \phi_2 = 1/2 + \ln 2 \approx 1.193$ (Clerc, 2006).

Any PSO variant can also be combined with various neighborhood topologies that control how the particles communicate to each other. The default global topology allows each particle to see each other particle's previous best location for the social components of their respective velocity updates, but this can cause inadequate exploration and premature convergence. Alternative neighborhood topologies limit how many other particles each particle can communicate with. We use a variant of the stochastic star topology (Miranda et al., 2008). Each particle informs itself and k random particles from the swarm, sampled with replacement during initialization of the algorithm.

Bare bones PSO (BBPSO) is a variant of PSO introduced by Kennedy (2003) that strips away the velocity term. Let $\theta_{ij}(k)$ denote the jth coordinate of the position for the ith particle in period t, and similarly for $p_{ij}(k)$ and $g_{ij}(k)$. Then the BBPSO algorithm obtains a new position coordinate θ_{ij} via

$$
\theta_{ij}(k+1) \sim N\left(\frac{p_{ij}(k) + g_{ij}(k)}{2}, |p_{ij}(k) - g_{ij}(k)|^2 \right). \tag{2}
$$

The updates of $\boldsymbol{p}_i(k)$ and $\boldsymbol{g}_i(k)$ are the same as in (1). Several modifications can be made to PSO and BBPSO to improve performance, and appendix A details which ones we use. In the next two subsections we introduce adaptively-tuned BBPSO and adaptively-tuned PSO respectively.

2.1. Adaptively tuned BBPSO

BBPSO adapts the size effective search space of the swarm over time through the variance term, $|p_{ij}(k) - g_{ij}(k)|^2$. As the personal best locations of the swarm move closer together, these variances decrease and the swarm explores locations which are closer to known high value areas in the space. This behavior is desirable, but the adaptation is forced to occur only through personal and group best locations. In order to allow the algorithm to adapt in a more flexible manner, we modify the BBPSO variance to $\sigma^2(k)|p_{ij}(k) - g_{ij}(k)|^2$ and tune $\sigma^2(k)$ in a manner similar to adaptive random walk Metropolis MCMC algorithms (Andrieu & Thoms, 2008). Define the improvement rate of the swarm in period t as $R(k) = \#\{i : Q(\boldsymbol{p}_i(k)) > Q(\boldsymbol{p}_i(k-1))\}/n$ where $\#A$ is the number of members of the set A. Adaptively tuned BBPSO (AT-BBPSO) compares $R(k)$ to a target improvement rate R^* and adjusts $\sigma^2(k)$ accordingly. Specifically AT-BBPSO updates the swarm's personal best and group best locations as in (1), then updates particle locations as follows:

$$\begin{aligned} \theta_{ij}(k+1) &\sim t_{df}\left(\frac{p_{ij}(k) + g_{ij}(k)}{2}, \sigma^2(k)|p_{ij}(k) - g_{ij}(k)|^2\right), \\ \log\sigma^2(k+1) &= \log\sigma^2(k) + c \times \{R(k+1) - R^*\}, \end{aligned} \tag{3}$$

where df is a user chosen degrees of freedom parameter and c is another user chosen parameter that controls the speed of adaptation. We use a t kernel instead of a Gaussian in order to allow for more flexibility, and we have found that $df = 1$ to works well with adaptively tuning $\sigma^2(k)$. Setting the target rate from $R^* = 0.3$ to $R^* = 0.5$ tends to yield AT-BBPSO algorithms which work well, which is unsurprising given the connection to adaptive random walk Metropolis. The parameter c controls the speed of adaptation so that larger values of c mean the algorithm adapts $\sigma^2(k)$ faster. We find that $c = 0.1$ works well, though anything within an order of maginitude yields similar results and there do no appear to be large gains to optimizing this parameter. We use $\sigma^2(0) = 1$ to initialize the algorithm at the standard BBPSO algorithm.

Both using a t kernel and adding a fixed scale parameter have been discussed in the BBPSO literature, but as Kennedy (2003) notes, something about setting $\sigma^2 = 1$ is special that causes the algorithm to work well. Hsieh & Lee (2010) propose a variant that keeps $\sigma^2 < 1$ for an initial set of iterations and even suggest tuning σ^2 dynamically, but give no suggestion for how to do this.

AT-BBPSO is able to adapt its value on the fly based on local knowledge about the objective function. If too much of the swarm is failing to find new personal best locations, AT-BBPSO proposes new locations closer to known high value areas. If too much of the swarm is improving, AT-BBPSO proposes bolder locations in an effort to make larger improvements. This ability to adapt to local information about the objective function allows AT-BBPSO to more quickly traverse the search space towards the global optimum, though by using local information AT-BBPSO does risk premature convergence to a local optimum. The adaptively tuned component of AT-BBPSO can also be combined with most BBPSO variants, some of which are outlined in Appendix A. In Appendix B we conduct a simulation study on several test functions that compares AT-BBPSO variants to other PSO and BBPSO variants in order to justify the parameter settings discussed above and demonstrate AT-BBPSO variants are attractive PSO algorithms. In particular, AT-BBPSO typically drastically improves on BBPSO, and for the most difficult objective functions AT-BBPSO yields the best algorithms.

2.2. Adaptively-tuned PSO

In AT-BBPSO variants, the parameter $\sigma(k)$ partially controls the effective size of the swarm's search area, and we increase or decrease $\sigma(k)$ and consequently the search area depending on how much of the swarm is finding new

personal best locations. In standard PSO the inertia parameter, denoted by ω in (1), is roughly analogous to σ^2 in BBPSO. It controls the effective size of the swarm's search area by controlling how the magnitude of the velocities evolve over time. We can also use the same mechanism to tune $\omega(k)$. Formally, AT-PSO is the same as PSO in (1), but at the end of an iteration it adds a step to update $\omega(k)$ via

$$\log \omega(k+1) = \log \omega(k) + c \times \{R(k+1) - R^*\} \tag{4}$$

where $R(k)$ is the improvement rate of the swarm in iteration t, R^* is the target improvement rate, and c controls how much $R(k)$ changes on a per iteration basis. Again, we find that target rates from $R^* = 0.3$ to 0.5 work well for AT-PSO. The value of c controls the speed of adaptation. We use $c = 0.1$ as a default value and in simulations (not reported here), we find that the gains from optimizing c appear to be small. This step can be combined with almost any PSO variant, and we combine it with each of the modifications listed in Appendix A.

The idea of time-varying $\omega(k)$ has been in the PSO literature for some time. An early suggestion was to set $\omega(0) = 0.9$ and deterministically decrease it until it reaches $\omega(k) = 0.4$ after the maximum number of iterations allowed (Eberhart & Shi, 2000). In particular, Tuppadung & Kurutach (2011) suggest defining $\omega(k)$ via the parameterized inertia weight function $\omega(k) = 1/\left[1 + (t/\alpha)^{\beta}\right]$ where α and β are user-defined parameters. Tuppadung & Kurutach (2011) suggest setting α to a small fraction of the total amount of iterations in which the algorithm is allowed to run (e.g., 10% or 20%), and setting β between one and four. We call this type of PSO algorithm deterministic inertia PSO (DI-PSO).

DI-PSO tends to improve on standard PSO if $\omega(k)$'s progression is set appropriately, but this can be difficult and often depends on the problem so an automatic method is desirable. A major strength of AT-PSO relative to DI-PSO and standard PSO is that AT-PSO can increase $\omega(k)$ when information from the swarm suggests there is an unexplored high value region of the space. Just like in AT-BBPSO, this mechanism provides a way for the swarm to adapt its behavior on the fly based on local conditions and speed up convergence by allowing the particles that do improve to make larger improvements, but it can also cause premature convergence to a local optimum. While DI-PSO monotonically decreases $\omega(k)$ toward some minimum value, AT-PSO typically oscillates $\omega(k)$ so that the algorithm alternates between periods of exploration and exploitation.

Another similar algorithm comes from Zhang et al. (2003). In their algorithm, ω is constant while ϕ_1 and ϕ_2 not only vary across time, but also across the swarm. Then each particle adapts its values of ϕ_1 and ϕ_2 based on how much it improves on its personal best location. The key difference is in their algorithm each particle adapts differently while in AT-PSO each particle adapts in the same way.

The simulation study in Appendix B includes several AT-PSO algorithms and demonstrates some of the behavior detailed above. Further, it shows that AT-PSO is often an attractive optimization algorithm relative to other PSO options.

3. The Spatial Design Problem

Suppose we are interested predicting some spatially indexed response variable $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subseteq \Re^2$ at a set of target locations $\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_{N_t} \in \mathcal{D}$. Let $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_{N_s} \in \mathcal{D}$ denote a set of N_s fixed sampling locations within the spatial domain. The design problem is to add N_d new sampling locations in order to optimize the amount we learn about $Y(\boldsymbol{u})$ at the target locations. Let $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_{N_d} \in \mathcal{D}$ denote a set of candidate design points and suppose that $Y(\boldsymbol{u})$ is a geostatistical process with mean function $\mu(\boldsymbol{u}) = \boldsymbol{x}(\boldsymbol{u})'\boldsymbol{\beta}$ for some covariate $\boldsymbol{x}(\boldsymbol{u})$ known at every point in $\mathcal{D}$ and some covariance function $C(\boldsymbol{u}, \boldsymbol{v})$ for $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{D}$. Not all covariates can be known a priori at every point in the spatial domain; however, covariates that are known functions of the location satisfy this constraint. Once the design points

are selected, we observe $Z(\boldsymbol{d}_i)$ for $i = 1, 2, \ldots, N_d$ and $Z(\boldsymbol{s}_i)$ for $i = 1, 2, \ldots, N_s$ where $Z(\boldsymbol{u}) = Y(\boldsymbol{u}) + \varepsilon(\boldsymbol{u})$ and $\varepsilon(\boldsymbol{u})$ is mean zero white noise with variance τ^2, representing measurement error. Typically $\boldsymbol{\beta}$, τ^2, and $C(.,.)$ are unknown and must be estimated, though for now we will treat them as known.

To completely specify the problem we need to define an informative design criterion. Intuitively, the larger the mean square prediction error (MSPE), i.e. the kriging variance, is at each of the target locations, the less information we have about $Y(\boldsymbol{u})$ at those locations. A common design criterion is to optimize is some function of these variances, e.g. to minimize the mean kriging variance or the maximum kriging variance over all target locations.

3.1. Universal Kriging

In universal kriging, $C(.,.)$ and τ^2 are treated as known while $\boldsymbol{\beta}$ is treated as a parameter that needs to be estimated. Let $\boldsymbol{Z}$ be the vector of $Z(\boldsymbol{s}_i)$s and $Z(\boldsymbol{d}_i)$s, $\boldsymbol{X}$ denote the corresponding stacked $\boldsymbol{x}(\boldsymbol{s}_i)'$s and $\boldsymbol{x}(\boldsymbol{d}_i)'$s, $\boldsymbol{C}_Z = \text{cov}(\boldsymbol{Z})$ where $\text{cov}[Z(\boldsymbol{u}), Z(\boldsymbol{v})] = C(\boldsymbol{u}, \boldsymbol{v}) + \sigma^2_\xi 1(\boldsymbol{u} = \boldsymbol{v})$, and $\boldsymbol{c}_Y(\boldsymbol{t}_i) = \text{cov}[Y(\boldsymbol{t}_i), \boldsymbol{Z}]$ where $\text{cov}[Y(\boldsymbol{t}_i), Z(\boldsymbol{u})] = C(\boldsymbol{t}_i, \boldsymbol{u})$. The universal kriging predictor of $Y(\boldsymbol{t}_i)$ is $\widehat{Y}_{uk}(\boldsymbol{t}_i; \boldsymbol{d}) = \boldsymbol{x}(\boldsymbol{t}_i)'\widehat{\boldsymbol{\beta}}_{gls} + \boldsymbol{c}_Y(\boldsymbol{t}_i)'\boldsymbol{C}_Z^{-1}(\boldsymbol{Z} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{gls})$ and the MSPE of $\widehat{Y}_{uk}(\boldsymbol{t}_i)$ is

$$\begin{aligned}\sigma^2_{uk}(\boldsymbol{t}_i; \boldsymbol{d}) = C(\boldsymbol{t}_i, \boldsymbol{t}_i) - \boldsymbol{c}_Y(\boldsymbol{t}_i)'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{t}_i) \\ + [\boldsymbol{x}(\boldsymbol{t}_i) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{t}_i)]'[\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{X}]^{-1}[\boldsymbol{x}(\boldsymbol{t}_i) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{t}_i)].\end{aligned}$$

where $\widehat{\boldsymbol{\beta}}_{gls} = [\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{X}]^{-1}\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{Z}$ is the generalized least squares estimate of $\boldsymbol{\beta}$ (Cressie & Wikle, 2011, Section 4.1.2). To avoid clutter we drop the explicit dependence on $\boldsymbol{d}$ in these equations, but $\boldsymbol{c}_Y(\boldsymbol{t}_i)$, $\boldsymbol{C}_Z$, $\boldsymbol{Z}$, $\boldsymbol{X}$, and $\widehat{\boldsymbol{\beta}}_{gls}$ all depend on $\boldsymbol{d}$. The mean universal kriging variance is given by $\overline{Q}_{uk}(\boldsymbol{d}) = \frac{1}{N_t}\sum_i^{N_t}\sigma^2_{uk}(\boldsymbol{t}_i; \boldsymbol{d})$ while the maximum universal kriging variance is given by $Q^*_{uk}(\boldsymbol{d}) = \max_{i=1,2\ldots,N_t}\sigma^2_{uk}(\boldsymbol{t}_i; \boldsymbol{d})$. Zimmerman (2006) finds that the optimal design under both criteria is highly dependent on the class of mean function of the geostatistical process. In practice we are often interested in predicting at the entire spatial domain rather than a finite set of target locations. This changes the mean and maximum kriging variances to $\overline{Q}_{uk}(\boldsymbol{d}) = \frac{1}{|\mathcal{D}|}\int_{\mathcal{D}}\sigma^2_{uk}(\boldsymbol{u}; \boldsymbol{d})d\boldsymbol{u}$ and $Q^*_{uk}(\boldsymbol{d}) = \max_{\boldsymbol{u}\in\mathcal{D}}\sigma^2_{uk}(\boldsymbol{u}; \boldsymbol{d})$ respectively. We can approximate both of these with a large but finite sample of target locations from $\mathcal{D}$ or a large fixed grid in $\mathcal{D}$.

3.2. Parameter Uncertainty Kriging

Assuming that all covariance function parameters are known, the MSPE from kriging at an arbitrary location $\boldsymbol{u}$ is $\sigma^2_{uk}(\boldsymbol{u})$. This underestimates the MSPE when those parameters must be estimated. An approximation of the correct MSPE, which we call the parameter uncertainty kriging variance (PUK variance), is given by (Zimmerman & Cressie, 1992; Abt, 1999)

$$\text{E}[Y(\boldsymbol{u}) - \widehat{Y}_{uk}(\boldsymbol{u})]^2 \approx \sigma^2_{puk}(\boldsymbol{u}; \boldsymbol{d}, \widehat{\boldsymbol{\theta}}) = \sigma^2_{uk}(\boldsymbol{u}; \boldsymbol{d}, \widehat{\boldsymbol{\theta}}) + \text{tr}[\boldsymbol{A}(\boldsymbol{u}; \boldsymbol{d}, \widehat{\boldsymbol{\theta}})\boldsymbol{I}^{-1}(\boldsymbol{d}, \widehat{\boldsymbol{\theta}})]$$

where $\widehat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of $\boldsymbol{\theta}$ from previously observed data, $\boldsymbol{I}^{-1}$ is the inverse Fisher information (FI) matrix, and $\boldsymbol{A} = \text{var}[\partial\widehat{Y}_{uk}/\partial\boldsymbol{\theta}]$. The ijth element of the FI matrix can be derived as $\text{tr}\left(\boldsymbol{C}_Z^{-1}\frac{\partial\boldsymbol{C}_Z}{\partial\theta_i}\boldsymbol{C}_Z^{-1}\frac{\partial\boldsymbol{C}_Z}{\partial\theta_j}\right)$ while $\boldsymbol{A}$ can be derived using elementary matrix calculus. From these we define the mean and maximum PUK variances as $\overline{Q}_{puk}(\boldsymbol{d}) = \frac{1}{N_t}\sum_i^{N_t}\sigma^2_{puk}(\boldsymbol{t}_i; \boldsymbol{d})$ and $Q^*_{puk}(\boldsymbol{d}) = \max_{i=1,2\ldots,N_t}\sigma^2_{puk}(\boldsymbol{t}_i; \boldsymbol{d})$ respectively.

3.3. Houston Ozone Monitoring

A 20 parts per billion (ppb) increase in daily maximum eight-hour ozone concentration (DM8) has been associated with an increased risk of out-of-hospital cardiact arrest, with a relative risk estimate of 1.039 (95% CI (1.005, 1.073);

Ensor et al., 2013). The Texas Commission on Environmental Quality (TCEQ) has monitoring stations throughout Texas recording a variety of environmental indicators, including Ozone. The TCEQ measures ozone at a network of monitoring locations and publishes DM8s in ppb for each monitoring location. DM8s are computed as follows. First, the TCEQ creates publishes an hourly mean in ppb for each monitoring location. Then an eight hour mean is constructed at that location for each contiguous eight-hour period where all eight measurements were present for a given day. The maximum of these eight-hour means for a given day is the published DM8. Days with less than 18 valid eight-hour means have no published DM8.

In August 2016 there were 44 active monitoring locations in the Houston-Galveston-Brazoria area, which we focus on. For each location $\boldsymbol{u}$ we compute the monthly mean DM8, which we denote by $Z(\boldsymbol{u})$. At one location, MRM-3 Haden Road, there are two DM8 observations of 0 ppb in the month of August. We assume that these were data errors and omit them for the purposes of computing $Z(\boldsymbol{u})$ at that location. Of the 44 locations, one has 15 valid DM8 measurements of 31 possible valid measurements, another has 24 valid measurements, and the rest of the locations have at least 27 valid measurements.

The hypothetical design problem we consider is the addition of 100 new ozone monitoring locations to the Houston-Galveston-Brazoria monitoring network in Harris County, where Houston is located, with the goal of predicting ozone concentrations within the county. Harris County contains 33 of the 44 existing locations, though the locations outside of the county are still useful for spatial prediction within the county.

Let $Z(\boldsymbol{u})$ denote the measured mean DM8 at location $\boldsymbol{u}$ and let $Y(\boldsymbol{u})$ denote the true DM8. We assume that $Z(\boldsymbol{u})$ is a noisy Gaussian measurement of $Y(\boldsymbol{u})$; i.e., the data model is $Z(\boldsymbol{u}) \sim N[Y(\boldsymbol{u}), \tau^2]$. The parameter τ^2 represents variability added due to both measurement error from the instruments and potentially sampling error from measuring DM8 on less than the full 31 days in August. At the process level, we assume that $Y(\boldsymbol{u})$ is a geostatistical process with mean function $\mu(\boldsymbol{u}) = \boldsymbol{x}(\boldsymbol{u})'\boldsymbol{\beta}$ and exponential covariance function $C(\boldsymbol{u}, \boldsymbol{v}) = \sigma^2 \exp(-||\boldsymbol{u} - \boldsymbol{v}||/\psi)$. We assume that any fine scale variability is measurement error and captured in the data model. We considered several possible mean functions: constant in $\boldsymbol{u}$, linear in $\boldsymbol{u}$, and quadratic in $\boldsymbol{u}$. We fit each model using maximum likelihood and found that quadratic terms were unnecessary, but linear terms did significantly help explain variation in $Y(\boldsymbol{u})$.

We use both PUK design criteria in order to choose the 100 new monitoring locations in Harris County: $\overline{Q}_{puk}(\boldsymbol{d})$ and $Q^*_{puk}(\boldsymbol{d})$, both defined in Section 3. We assume that the goal is the predict mean DM8 in all of Harris County, so we approximate the continuous versions of $\overline{Q}_{puk}(\boldsymbol{d})$ and $Q^*_{puk}(\boldsymbol{d})$ with the finite sample versions using a grid of 1229 points, obtained by gridding up the smallest rectangle containing Harris County and throwing away all points outside of the county. We try a variety of PSO algorithms in order to select the new locations. Since the design space only allows new monitoring locations within Harris County, we modify each of the PSO algorithms we use so that any particle outside of the design space is forced to moved to the nearest point on the edge of the design space as described in Appendix A. Existing sampling locations outside of Harris County are still used in the estimation of the model and in the construction of the PUK variances.

Table 1 contains the results of applying each optimization algorithm to choosing the 100 new monitoring locations once with each algorithm. In the table, PSO refers to the standard PSO algorithm with the usual velocity update and all of the other modifications listed in Appendix A.1. The CF modifier indicates that the velocity update is coordinate-free – see Appendix A.1 for details. The AT modifier indicates that $\omega(k)$ is adaptively tuned as described in Section 2.2. AT1 uses $R^* = 0.3$ and AT2 uses $R^* = 0.5$, while both use $c = 0.1$ and $\omega(0) = 1.2$. We use two sets of parameter values for all of the PSO algorithms: $\phi_1^{(1)} = \phi_2^{(1)} = 1.496$ (PSO1) and $\phi_1^{(2)} = \phi_2^{(2)} = \ln(2) + 1/2$ (PSO2), and when applicable the corresponding inertias are $\omega^{(1)} = 0.7298$ and $\omega^{(2)} = 1/(2\ln 2)$. We do not list any DI algorithms since they performed extremely poorly on this problem. All of the BBPSO algorithms we consider are AT, with AT1-BBPSO and AT2-BBPSO using the same parameter values of R^*, and c as AT1-PSO and AT2-PSO

respectively. All BBPSO algorithms also use $df = 1$, and each of the modications detailed in Appendix A.2. Further, the CF modifier indicates that the BBPSO algorithm uses the coordinate-free variance update and the xp modifier indicates it has a 0.5 probability of moving any given coordinate of a particle to its personal best location on that coordinate, both described in Appendix A.2. All of the PSO and BBPSO algorithms use either the global neighborhood topology, or the variant of the stochastic star neighborhood topology discussed in Section 2 with three informants. Further, each PSO algorithm has a swarm size of 40 and is run for $K = 2000$ iterations.

Additionally, we employ a class of genetic algorithms (GAs) described in Hamada et al. (2001) to compare to PSO, with either one or two batches, and with two possible mutation rates ($\lambda_1 = 0.01$ or $\lambda_2 = 0.1$) and two possible mutation variances ($\mu_1 = 1$ or $\mu_2 = 2$). The GAs also use a population of 40, though they are only allowed to run for 1000 iterations so that after the initialization, the GAs use the same number of objective function evaluations as each of the PSO algorithms. In order to handle the bounded search space, we use the same method for the GAs as for the PSO algorithms described in Appendix A: any point the GA suggests that is outside of Harris County is moved to the nearest point on the border of the county. Each algorithm was implemented in the R programming language (R Core Team, 2016). In Table 1, GAs are referred to by "GA-*ab*" where *a* denotes which value of λ is used and *b* denotes which value of μ is used. As a point of comparison for both classes of algorithms, we randomly selected the 100 new monitoring locations uniformly from Harris county 10,000 times independently and computed the values of $\overline{Q}_{puk}$ and Q^*_{puk}. The means and standard devations of these values are labeled as "Uniform" in Table 1.

$\overline{Q}_{puk}$

Nbhd:	Global	SS3
PSO1	14.40	15.62
PSO2	14.45	14.87
PSO1-CF	15.53	15.94
PSO2-CF	15.77	15.56
AT1-PSO1	14.38	15.25
AT1-PSO2	14.56	15.88
AT1-PSO1-CF	15.96	15.98
AT1-PSO2-CF	15.60	15.86
AT2-PSO1	14.42	15.70
AT2-PSO2	14.32	15.70
AT2-PSO1-CF	15.85	15.96
AT2-PSO2-CF	15.95	15.87
AT1-BBPSO	14.53	15.72
AT1-BBPSOxp	15.87	15.96
AT1-BBPSO-CF	14.65	14.82
AT1-BBPSOxp-CF	14.84	15.74
AT2-BBPSO	14.65	15.74
AT2-BBPSOxp	15.21	15.32
AT2-BBPSO-CF	14.63	15.32
AT2-BBPSOxp-CF	14.52	15.71

Batches:	One	Two
GA-11	14.40	14.46
GA-21	15.20	15.05
GA-12	14.45	14.39
GA-22	15.26	15.14

	Mean	SD
Uniform	16.40	0.17

Q^*_{puk}

Nbhd:	Global	SS3
PSO1	20.63	22.32
PSO2	21.03	22.63
PSO1-CF	23.54	23.69
PSO2-CF	23.16	23.96
AT1-PSO1	20.57	22.54
AT1-PSO2	23.18	23.14
AT1-PSO1-CF	23.33	23.68
AT1-PSO2-CF	24.02	23.53
AT2-PSO1	21.13	22.63
AT2-PSO2	22.11	23.28
AT2-PSO1-CF	24.00	23.97
AT2-PSO2-CF	23.63	23.58
AT1-BBPSO	22.28	23.31
AT1-BBPSOxp	22.19	23.39
AT1-BBPSO-CF	21.33	23.20
AT1-BBPSOxp-CF	22.34	23.44
AT2-BBPSO	23.49	23.51
AT2-BBPSOxp	23.25	23.15
AT2-BBPSO-CF	21.92	23.33
AT2-BBPSOxp-CF	22.76	23.25

Batches:	One	Two
GA-11	21.19	20.99
GA-21	23.21	22.92
GA-12	20.84	20.66
GA-22	22.61	22.26

	Mean	SD
Uniform	26.80	0.95

Table 1. Simulation results for $\overline{Q}_{puk}$ and Q^*_{puk}. See text for description.

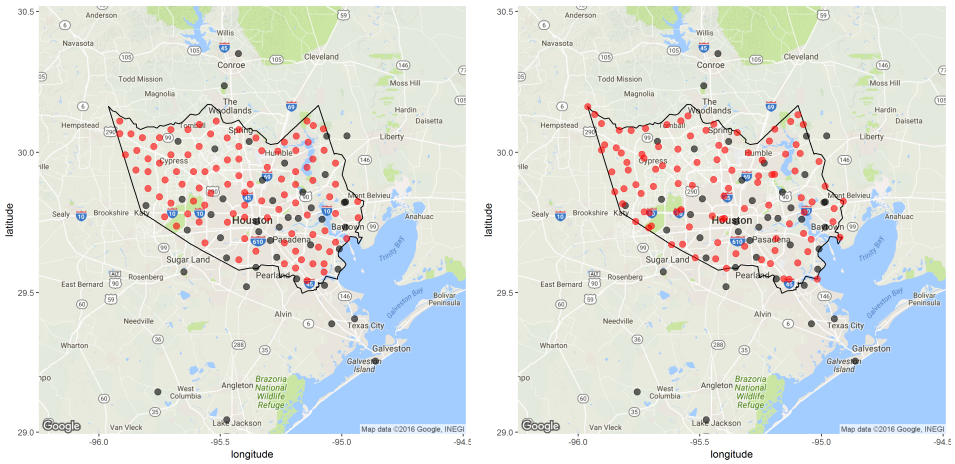

Figure 1. Best designs found according to $\overline{Q}_{puk}$ (left) and Q^*_{puk} (right). Original network points are grey, new points are red. Harris County is outlined in black. Map courtesy of Google Maps via `ggmap` (Kahle & Wickham, 2013).

From Table 1 we immediately see that the best performing PSO algorithms tend to use the global neighborhood topology and tend not to use the coordinate free velocity/variance update. Further the first set of parameter values tends to work the best in the PSO algorithms while in the BBPSO algorithms the non-xp variants tend to outperform the xp variants, though neither of these is universally true. For both objective functions the PSO1 algorithm with the global neighborhood topology performs the best of the non AT variants and is outperformed by few of the AT variants. The best GAs are competitive with the best PSO algorithms, though they appear to be slightly worse. The five best performing algorithms for $\overline{Q}_{puk}$ are, in order, AT2-PSO2-Global, AT1-PSO1-Global, GA-12-Two, GA-11-One, and PSO1-Global. Similarly, the five best performing algorithms for Q^*_{puk} are, in order, AT1-PSO1-Global, PSO1-Global, GA-12-Two, GA-12-One, and GA-11-Two. In Appendix B we found that the AT-BBPSO and AT-BBPSO-CF variants were competitive with the other PSO algorithms and in the case of the hardest problems seemed to be the best. The key seems to be that the AT-BBPSO variants are more robust to complicated objective surfaces with many local minima. When the objective surface is simpler, PSO and AT-PSO variants are more attractive and appear to converge faster. Similarly for problems that are simple enough PSO outperforms AT-PSO, but for hard enough problems AT-PSO becomes advantageous. For example in simulations not reported here, PSO1 outperforms all other algorithms when adding significantly less monitoring locations to the network, e.g. 20. But here with 100 new locations, AT1-PSO1 appears to be superior for both objective functions.

Figure 1 contains the best designs found using both objective functions. When using the mean kriging variance ($\overline{Q}_{puk}$), the best design spreads nodes of the network fairly evenly throughout the spatial domain. With the maximum kriging variance (Q^*_{puk}), the best design is more erratic. Notably, there are many more nodes directly on or very close to the border of the spatial domain. Further, interior areas have a tendency to be more sparsely populated with nodes. This is unsurprising since Q^*_{puk} is worst case kriging variance over the entire county, which incentivizes putting monitoring locations near the edges of the county because the farther away a point is from other monitoring locations, the more the prediction depends on the model's estimate of $\boldsymbol{\beta}$ and the less it depends on the values of nearby observations. So Q^*_{puk} places a premium on a better estimate of $\boldsymbol{\beta}$, and putting more monitoring locations near the edges of the county results in smaller variances for the elements of $\widehat{\boldsymbol{\beta}}_{gls}$.

4. Discussion

In this paper we reviewed PSO algorithms, introduced AT-PSO and AT-BBPSO algorithms, and demonstrated that AT-PSO and PSO algorithms more generally are useful for spatial design problems. In the simulation study in Appendix B we found that AT-BBPSO variants were especially attractive for optimization problems with complicated objective surfaces, potentially with many local optima. However in Section 3.3 we found little difference between the standard PSO algorithm and the best AT algorithms. The upshot is that at least for spatial design problems similar to or easier than the one we considered here, using a standard PSO algorithm with standard parameter settings and the typical modifications we discussed in Appendix A.1 is a good default, especially since this algorithm is already implemented in several widely available packages. Similarly, standard GAs are a good default option, and packages for implementing them are also widely available. Furthermore, in similar spatial design problems which are much lower dimensional than the one discussed in this paper we have found that standard PSO algorithms are often the best.

However, our results in Appendix B suggest that in more complex spatial models, e.g. with a more complex mean function or a very high dimensional search space, there may be significant benefits to using AT-PSO or AT-BBPSO. As long as the user has a reasonable expectation that the objective surface is not too complex, whichever algorithm is easier for them to implement and use is likely the best, but for more complex problems there is much to be gained from trying several different algorithms.

The usefulness of PSO and AT variants for statistics is not limited to spatial design. Other design problems are amenable to PSO, but in general any optimization problem which is difficult to solve with gradient based algorithms is a good candidate, especially when near-optimal solutions are tolerable. In the spatial design literature exchange algorithms are commonly employed instead of PSO or GAs, though these require limiting the search space to a discrete grid (Nychka & Saltzman, 1998; Wikle & Royle, 1999, 2005). PSO (and GA) allow the entire search space to be used without restriction, though in cases where there is a known discrete set of possible monitoring locations the exchange algorithm is more suited to the task.

References

Abt, M (1999), 'Estimating the prediction mean squared error in Gaussian stochastic processes with exponential correlation structure,' *Scandinavian Journal of Statistics*, **26**(4), pp. 563–578.

Andrieu, C & Thoms, J (2008), 'A tutorial on adaptive MCMC,' *Statistics and Computing*, **18**(4), pp. 343–373.

Blum, C & Li, X (2008), 'Swarm intelligence in optimization,' in Blum, C & Merkle, D (eds.), *Swarm Intelligence: Introduction and Applications*, Springer.

Clerc, M (2006), 'Stagnation analysis in particle swarm optimisation or what happens when nothing happens,' Tech. rep.

Clerc, M (2010), *Particle swarm optimization*, John Wiley & Sons.

Clerc, M (2011), 'Standard particle swarm optimisation,' Tech. rep.

Clerc, M & Kennedy, J (2002), 'The particle swarm-explosion, stability, and convergence in a multidimensional complex space,' *Evolutionary Computation, IEEE Transactions on*, **6**(1), pp. 58–73.

Cressie, N & Wikle, CK (2011), *Statistics for Spatio-Temporal Data*, John Wiley & Sons.

Eberhart, RC & Shi, Y (2000), 'Comparing inertia weights and constriction factors in particle swarm optimization,' in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, IEEE, vol. 1, pp. 84–88.

Ensor, KB, Raun, LH & Persse, D (2013), 'A case-crossover analysis of out-of-hospital cardiac arrest and air pollution,' *Circulation*, **127**(11), pp. 1192–1199.

Hamada, M, Martz, H, Reese, C & Wilson, A (2001), 'Finding near-optimal bayesian experimental designs via genetic algorithms,' *The American Statistician*, **55**(3), pp. 175–181.

Hsieh, HI & Lee, TS (2010), 'A modified algorithm of bare bones particle swarm optimization,' *International Journal of Computer Science Issues*, **7**, p. 11.

Kahle, D & Wickham, H (2013), 'ggmap: Spatial visualization with ggplot2,' *The R Journal*, **5**(1), pp. 144–161.

Kennedy, J (2003), 'Bare bones particle swarms,' in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, IEEE, pp. 80–87.

Miranda, V, Keko, H & Duque, AJ (2008), 'Stochastic star communication topology in evolutionary particle swarms (epso),' *International journal of computational intelligence research*, **4**(2), pp. 105–116.

Nychka, D & Saltzman, N (1998), 'Design of air-quality monitoring networks,' in *Case Studies in Environmental Statistics*, Springer, pp. 51–76.

R Core Team (2016), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Tuppadung, Y & Kurutach, W (2011), 'Comparing nonlinear inertia weights and constriction factors in particle swarm optimization,' *International Journal of Knowledge-based and Intelligent Engineering Systems*, **15**(2), pp. 65–70.

Wikle, CK & Royle, JA (1999), 'Space: time dynamic design of environmental monitoring networks,' *Journal of Agricultural, Biological, and Environmental Statistics*, pp. 489–507.

Wikle, CK & Royle, JA (2005), 'Dynamic design of ecological monitoring networks for non-gaussian spatio-temporal data,' *Environmetrics*, **16**(5), pp. 507–522.

Zhang, W, Liu, Y & Clerc, M (2003), 'An adaptive pso algorithm for reactive power optimization,' in *Advances in Power System Control, Operation and Management, 2003. ASDCOM 2003. Sixth International Conference on (Conf. Publ. No. 497)*, IET, vol. 1, pp. 302–307.

Zimmerman, DL (2006), 'Optimal network design for spatial prediction, covariance parameter estimation, and empirical prediction,' *Environmetrics*, **17**(6), pp. 635–652.

Zimmerman, DL & Cressie, N (1992), 'Mean squared prediction error in the spatial linear model with estimated covariance parameters,' *Annals of the Institute of Statistical Mathematics*, **44**(1), pp. 27–43.