

Adaptively-Tuned Particle Swarm Optimization with Application to Spatial Design

Abstract

KEY WORDS:

1 Introduction

Particle swarm optimization (PSO) refers to a large class of heuristic optimization algorithms that use an analogy with animal flocking behavior in order to construct more robust optimization algorithms than many alternatives (Clerc and Kennedy, 2002; Blum and 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 evaluate several PSO algorithms by applying them to choosing a set of new monitoring locations for ozone in Harris County, Texas, where Houston is located. Section 2 introduces various PSO and AT-PSO algorithms, and briefly discusses the results of an extended simulation study comparing them. Section 3 introduces the generic spatial design problem and the Houston 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.

2 Particle swarm optimization

We briefly describe PSO here; refer to Blum and Li (2008) for an excellent introduction and Clerc (2010) for details. Suppose that we wish to maximize some objective function $Q(\boldsymbol{\theta}) : \mathbb{R}^D \rightarrow \mathbb{R}$. Let $i = 1, 2, \dots, n$ index a set of particles over time, $t = 1, 2, \dots, T$, where in every period each particle consists of a location $\boldsymbol{\theta}_i(t) \in \mathbb{R}^D$, a velocity $\mathbf{v}_i(t) \in \mathbb{R}^D$, a personal best location $\mathbf{p}_i(t) \in \mathbb{R}^D$, and a group best location $\mathbf{g}_i(t) \in \mathbb{R}^D$. Here we mean “best” in the sense of maximizing Q , so $Q(\mathbf{p}_i(t)) \geq Q(\boldsymbol{\theta}_i(s))$ for any $s \leq t$. The

group best location is defined with respect to some neighborhood \mathcal{N}_i of particle i ; that is, $\mathbf{g}_i(t) = \arg \max_{\{\mathbf{p}_j(t)|j \in \mathcal{N}_i\}} Q(\mathbf{p}_j(t))$. In the simplest case where the entire swarm is the neighborhood of each particle, $\mathbf{g}_i(t) \equiv \mathbf{g}(t) = \arg \max_{\{\mathbf{p}_j(t)|j \in 1:n\}} Q(\mathbf{p}_j(t))$. The generic PSO algorithm updates as follows for each particle i :

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

where \circ denotes the Hadamard product (element-wise product), $\mathbf{r}_{1i}(t)$ and $\mathbf{r}_{2i}(t)$ 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 $\omega \mathbf{v}_i(t)$ controls the particle's tendency to keep moving in the direction it is already going, so ω is called the inertia parameter. For $\omega < 1$ velocities tend to decrease over time, while for $\omega > 1$ they tend to increase over time. Similarly $\phi_1 \mathbf{r}_{1i}(t) \circ \{\mathbf{p}_i(t) - \boldsymbol{\theta}_i(t)\}$ controls the particle's tendency to move towards its personal best location while $\phi_2 \mathbf{r}_{2i}(t) \circ \{\mathbf{g}_i(t) - \boldsymbol{\theta}_i(t)\}$ 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 and Li, 2008). This version of PSO is equivalent to Clerc and Kennedy (2002)'s constriction type I particle swarm, though there are many other variants. A default version choice sets $\omega = 0.7298$ and $\phi_1 = \phi_2 = 1.496$; see Clerc and Kennedy (2002) for justification. Even when $\omega < 1$, if ϕ_1 and ϕ_2 are set high enough the velocities of the particles can continually increase and cause the swarm to make jumps that are much too large. A heavy handed way to solve this problem is by setting an upper bound on the velocity of any particle in any given direction, called velocity clamping. However, the default parameter values suggested by Clerc and Kennedy (2002)

are also designed to prevent velocity explosion.

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. The ring- k topologies do this by only allowing each particle to see the k particles to its left and the k particles to its right when the particles are arranged in a ring. For example, particle 1 may only look at itself and particles 2, 3, n , and $n - 1$ when determining its group best location. This slows down the transfer of information across the swarm so that each particle spends more time exploring. Several other neighborhood topologies can be used, see [CITATIONS]

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

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

The updates of $\mathbf{p}_i(t)$ and $\mathbf{g}_i(t)$ are the same as in (1). There are several variants of this algorithm, for example Krohling et al. (2009), Hsieh and Lee (2010), Richer and Blackwell (2006), and Campos et al. (2014). Appendix A.1 contains more detail on some of these BBPSO variants.

2.1 Adaptively tuned BBPSO

BBPSO adapts the size effective search space of the swarm over time through the variance term, $|p_{ij}(t) - g_{ij}(t)|^2$. 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 it to adapt in a more flexible manner, we modify the BBPSO variance to $\sigma^2(t)|p_{ij}(t) - g_{ij}(t)|^2$ and tune $\sigma^2(t)$ in a manner similar to adaptive random walk Metropolis MCMC algorithms (Andrieu and Thoms, 2008). Define the improvement rate of the swarm in period t as $R(t) = \#\{i : Q(\mathbf{p}_i(t)) > Q(\mathbf{p}_i(t-1))\}/n$ where $\#A$ is the number of members of the set A , and let R^* denote the target improvement rate. Then adaptively tuned BBPSO (AT-BBPSO) updates the swarm's personal best and group best locations as in (1), then updates particle locations as follows:

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

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 find $df = 1$ appears to combine well with adaptively tuning $\sigma^2(t)$. Setting the target rate from $R^* = 0.3$ to $R^* = 0.5$ tends to yield good AT-BBPSO algorithms, which is unsurprising given the connection to adaptive random walk Metropolis (Gelman et al., 1996). The parameter c controls the speed of adaptation so that larger values of c mean the algorithm adapts $\sigma^2(t)$ faster. We find that $c = 0.1$ to be a good value, though anything within an order of magnitude yields similar results. 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 $\sigma = 1$ is special that causes the algorithm to work well. Another similar BBPSO algorithm in the literature

comes from Hsieh and Lee (2010). They propose a modified version of BBPSO with

$$\theta_{ij}(t+1) \sim N\left(\omega \frac{p_{ij}(t) + g_{ij}(t)}{2}, \sigma^2 |p_{ij}(t) - g_{ij}(t)|^2\right),$$

where $\omega \leq 1$ and $\sigma^2 \leq 1$ are constriction parameters that are eventually both set to one after enough iterations of the algorithm. The authors suggest dynamically adjusting the constriction parameters in the early stage of the algorithm before they are set to one, but give no suggestion for how to do this. AT-BBPSO algorithm 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.1. 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.

2.2 Adaptively tuned PSO

In AT-BBPSO variants, the parameter $\sigma(t)$ partially controls the effective size of the swarm's search area, and we increase or decrease $\sigma(t)$ 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. In AT-BBPSO we use an analogy with tuning a random walk Metropolis-Hastings MCMC algorithm in order to build intuition about how to tune $\sigma(t)$. The analogy is much weaker in this case; nonetheless, the same mechanism works well to tune $\omega(t)$.

The idea of time-varying $\omega(t)$ 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 $\omega(T) = 0.4$ after the maximum number of iterations allowed (Eberhart and Shi, 2000). In particular, Tuppadung and Kurutach (2011) suggest defining $\omega(t)$ via the parameterized inertia weight function

$$\omega(t) = \frac{1}{1 + \left(\frac{t}{\alpha}\right)^\beta} \quad (4)$$

where α and β are user-defined parameters. Roughly, α controls how low $\omega(t)$ can go and β controls how fast it gets there, so α and β can be thought of as intercept and slope parameters respectively. The suggestion in Tuppadung and Kurutach (2011) is to set 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 set 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 $\omega(t)$'s progression is set appropriately, but it invariably makes using PSO more difficult for the average user. Additionally, depending on the problem, it may be more useful to let the swarm explore the space for more or less iterations, necessitating different progressions of $\omega(t)$. A priori it may not be clear exactly which approach is best for any given problem, so an automatic method is desirable. Adaptively tuned PSO (AT-PSO) is just that — it provides an automatic method to adjust the value of $\omega(t)$ depending on local information obtained by the particle swarm. Formally, AT-PSO updates personal and group best locations as in (1) and updates $\omega(t)$ and particle

locations as follows:

$$\begin{aligned}
\mathbf{v}_i(t+1) &= \omega(t+1)\mathbf{v}_i(t) + \phi_1\mathbf{r}_{1i}(t) \circ \{\mathbf{p}_i(t) - \boldsymbol{\theta}_i(t)\} + \phi_2\mathbf{r}_{2i}(t) \circ \{\mathbf{g}_i(t) - \boldsymbol{\theta}_i(t)\}, \\
\boldsymbol{\theta}_i(t+1) &= \boldsymbol{\theta}_i(t) + \mathbf{v}_i(t+1), \\
\log \omega(t+1) &= \log \omega(t) + c \times \{R(t+1) - R^*\},
\end{aligned} \tag{5}$$

where $R(t)$ is the improvement rate of the swarm in iteration t , R^* is the target improvement rate, and c controls how much $R(t)$ 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, and in particular if c is small and $\omega(0)$ is large, AT-PSO can mimic DI-PSO to some extent. This turns out to produce poor AT-PSO algorithms, however. 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. However, very small values like those suggested by an attempt to mimic DI-PSO turn out to cause the algorithm to perform very poorly.

A major strength of AT-PSO relative to DI-PSO and standard PSO is that AT-PSO can increase $\omega(t)$ 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 $\omega(t)$ toward some minimum value, AT-PSO typically oscillates $\omega(t)$ so that the algorithm alternates between exploring and exploiting more, relative to standard PSO. Appendix B contains an extended simulation study comparing a variety of these PSO and BBPSO algorithms on a suite of test functions that demonstrates some of the behavior detailed above and shows that AT-PSO is an attractive PSO algorithm.

3 The Spatial Design Problem

Suppose we are interested predicting some spatially indexed response variable $Y(\mathbf{u})$, $\mathbf{u} \in \mathcal{D} \subseteq \mathbb{R}^2$ at a set of target locations $\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_t} \in \mathcal{D}$. Let $\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{N_s} \in \mathcal{D}$ 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 $Y(\mathbf{u})$ at the target locations. Let $\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_{N_d} \in \mathcal{D}$ denote a set of candidate design points and suppose that $Y(\mathbf{u})$ is a geostatistical process with mean function $\mu(\mathbf{u}) = \mathbf{x}(\mathbf{u})'\boldsymbol{\beta}$ for some covariate $\mathbf{x}(\mathbf{u})$ known at every point in \mathcal{D} and some covariance function $C(\mathbf{u}, \mathbf{v})$ for $\mathbf{u}, \mathbf{v} \in \mathcal{D}$. Not all covariates can be known a priori at every point in the spatial domain, but e.g. covariates that are known functions of the location satisfy this constraint. Once the design points are selected, we observe $Z(\mathbf{d}_i)$ for $i = 1, 2, \dots, N_d$ and $Z(\mathbf{s}_i)$ for $i = 1, 2, \dots, N_s$ where $Z(\mathbf{u}) = Y(\mathbf{u}) + \varepsilon(\mathbf{u})$ and $\varepsilon(\mathbf{u})$ is mean zero white noise with variance τ^2 , representing measurement error. Typically $\boldsymbol{\beta}$, τ^2 , and $C(\cdot, \cdot)$ 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 $Y(\mathbf{u})$ at those locations. A common design criterion is to optimize function of these variances, e.g. to minimize the mean kriging variance or the maximum kriging variance over all target locations. These criteria are somewhat naive since they ignore parameter uncertainty, and taking that into account will often change the optimal design (Zimmerman, 2006). In the Bayesian context, the standard design criterion is the expected entropy gain from observing the data (Ebrahimi et al., 2010), e.g. in Fuentes et al. (2007), and it can be adapted to both the case of known and unknown parameters. Though PSO can be applied to design problems with any combination of known and unknown parameters, for simplicity we consider minimizing both mean and

maximum kriging variance under universal kriging where only $\boldsymbol{\beta}$ and $\{Y(\mathbf{u}) : \mathbf{u} \in \mathcal{D}\}$ are treated as unknown.

3.1 Universal Kriging

In universal kriging, $C(\cdot, \cdot)$ and τ^2 are treated as known while $\boldsymbol{\beta}$ is treated as a parameter that needs to be estimated. Let \mathbf{Z} the vector of $Z(\mathbf{s}_i)$ s and $Z(\mathbf{d}_i)$ s, \mathbf{X} denote the corresponding stacked $\mathbf{x}(\mathbf{s}_i)$'s and $\mathbf{x}(\mathbf{d}_i)$'s, $\mathbf{C}_Z = \text{cov}(\mathbf{Z})$ where $\text{cov}[Z(\mathbf{u}), Z(\mathbf{v})] = C(\mathbf{u}, \mathbf{v}) + \sigma_\varepsilon^2 1(\mathbf{u} = \mathbf{v})$, and $\mathbf{c}_Y(\mathbf{t}_i) = \text{cov}[Y(\mathbf{t}_i), \mathbf{Z}]$ where $\text{cov}[Y(\mathbf{t}_i), Z(\mathbf{u})] = C(\mathbf{t}_i, \mathbf{u})$. The universal kriging predictor of $Y(\mathbf{t}_i)$ is (Cressie and Wikle, 2015, Section 4.1.2)

$$\hat{Y}_{uk}(\mathbf{t}_i; \mathbf{d}) = \mathbf{x}(\mathbf{t}_i)' \hat{\boldsymbol{\beta}}_{gls} + \mathbf{c}_Y(\mathbf{t}_i)' \mathbf{C}_Z^{-1} (\mathbf{Z} - \mathbf{X} \hat{\boldsymbol{\beta}}_{gls})$$

where $\hat{\boldsymbol{\beta}}_{gls} = [\mathbf{X}' \mathbf{C}_Z^{-1} \mathbf{X}]^{-1} \mathbf{X}' \mathbf{C}_Z^{-1} \mathbf{Z}$ is the generalized least squares estimate of $\boldsymbol{\beta}$, and the MSPE of $\hat{Y}_{uk}(\mathbf{t}_i)$ is

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

To avoid clutter we drop the explicit dependence on \mathbf{d} in these equations, but $\mathbf{c}_Y(\mathbf{t}_i)$, \mathbf{C}_Z , \mathbf{Z} , \mathbf{X} , and $\hat{\boldsymbol{\beta}}_{gls}$ all depend on \mathbf{d} . The mean universal kriging variance is given by

$$\overline{Q}_{uk}(\mathbf{d}) = \frac{1}{N_t} \sum_i^{N_t} \sigma_{uk}^2(\mathbf{t}_i; \mathbf{d})$$

while the maximum universal kriging variance is given by

$$Q_{uk}^*(\mathbf{d}) = \max_{i=1,2,\dots,N_t} \sigma_{uk}^2(\mathbf{t}_i; \mathbf{d}).$$

In this context, 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

$$\overline{Q}_{uk}(\mathbf{d}) = \frac{1}{|\mathcal{D}|} \int_{\mathcal{D}} \sigma_{sk}^2(\mathbf{d}; \mathbf{d}) d\mathbf{u}$$

and

$$Q_{uk}^*(\mathbf{d}) = \max_{\mathbf{u} \in \mathcal{D}} \sigma_{sk}^2(\mathbf{u}; \mathbf{d}).$$

respectively. We can approximate both of these with a large but finite sample of target locations from \mathcal{D} , though if the sample is too small some design criteria may favor points directly on top of the sampled locations.

3.2 Houston Ozone Monitoring

The Texas Commission on Environmental Quality (TCEQ) publishes a variety of environmental data for Texas, including many environmental indicators that directly relate to public health. We focus on ozone in the Houston-Galveston-Brazoria area. The TCEQ measures ozone at a variety of monitoring locations in this area and publishes daily maximum eight-hour ozone concentrations (DM8s) in parts per billion (ppb) for each monitoring location. DM8s are computed as follows. First, the TCEQ creates publishes an hourly average (in ppb) for each monitoring location. Then an eight hour average 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 averages for a given day is the published DM8. Days with less than 18 valid eight-hour averages have no published DM8.

In August 2016 there were 44 active monitoring locations in the Houston-Galveston-Brazoria area. For each location \mathbf{u} we compute the monthly average DM8, which we denote by $Z(\mathbf{u})$. 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 throw them out for the

purposes of computing $Z(\mathbf{u})$ 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 five 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 Harris County. Of the 44 existing locations, 33 are already in Harris County, though the locations outside of the county are still useful for spatial prediction within Harris County.

Let $Z(\mathbf{u})$ denote the measured average DM8 at location \mathbf{u} and let $Y(\mathbf{u})$ denote the true DM8. We assume that $Z(\mathbf{u})$ is a noisy Gaussian measurement of $Y(\mathbf{u})$, i.e. the data model is $Z(\mathbf{u}) \sim N[Y(\mathbf{u}), s_Z^2(\mathbf{u}) + \tau^2]$. The measurement error is broken into two pieces which we assume are independent: pure measurement error with variance τ^2 , and sampling error with variance $s_Z^2(\mathbf{u})$. Sampling error occurs from measuring DM8 on less than the full 31 days in August. We assume that the measured days are randomly sampled from the 31 possible days so that the sampling error for $Z(\mathbf{u})$ is $s_Z^2(\mathbf{u}) = s_{DM8}^2(\mathbf{u})[31 - n(\mathbf{u})]/[31n(\mathbf{u})]$ where $s_{DM8}^2(\mathbf{u})$ is the sample variance of the DM8 measurements at location \mathbf{u} and $n(\mathbf{u})$ is the number of measurements at that location. Note that if $n(\mathbf{u}) = 31$, the maximum number of possible DM8 measurements, then $s_Z^2(\mathbf{u}) = 0$. This sampling model is almost certainly incorrect since missing DM8 measurements at a given location tend to be contiguous in time, but it is a useful approximation as long as the probability of a given day being measured is largely unrelated to the true DM8 at that location.

At the process level, we assume that $Y(\mathbf{u})$ is a geostatistical process with mean function $\mu(\mathbf{u}) = \mathbf{x}(\mathbf{u})'\boldsymbol{\beta}$ and exponential covariance function $C(\mathbf{u}, \mathbf{v}) = \sigma^2 \exp(-\|\mathbf{u} - \mathbf{v}\|/\phi)$. We assume that any fine scale variability is measurement error and captured by the data model variance $s_Z^2(\mathbf{u}) + \tau^2$. We considered several possible mean functions: constant in \mathbf{u} , linear in \mathbf{u} , and quadratic in \mathbf{u} . We fit each model using maximum likelihood and found that quadratic

terms were unnecessary, but linear terms did significantly help explain variation in $Y(\mathbf{u})$. Further, both AIC and BIC were essentially indifferent between the linear and constant mean models, so we use the linear model. Note that this model is a slight departure from the class considered in Section 3. We use the same notation in that section as in this section so that, e.g., \mathbf{u} is a generic location inside \mathcal{D} , which can be considered to be the entire Houston-Galveston-Brazoria area, \mathbf{s}_i is an existing monitoring location, \mathbf{t}_i is a target location, and \mathbf{d}_i is a proposed location for a new monitoring station. The only difference is that in Section 3 $\mathbf{C}_Z = \tau^2 \mathbf{I} + \mathbf{C}_Y$, whereas in this section $\mathbf{C}_Z = \mathbf{S}_Z^2 + \tau^2 \mathbf{I} + \mathbf{C}_Y$ where \mathbf{S}_Z^2 is a diagonal matrix with $[s_Z^2(\mathbf{s}_1), s_Z^2(\mathbf{s}_2), \dots, s_Z^2(\mathbf{s}_{N_s}), 0, \dots, 0]$ along the diagonal. This assumes that $s_Z^2(\mathbf{d}_i) = 0$ for $i = 1, 2, \dots, N_d$, in other words that each new monitoring station will record a valid DM8 for every day of a given month so that there is no sampling error. [NOTE: WE SHOULD OBVIOUSLY FIT THE MODEL USING THE $s_Z^2(\mathbf{s}_i)$ s, BUT FOR KRIGING SHOULD WE ASSUME $s_Z^2(\mathbf{s}_i) = 0$? OR MAYBE SOME OTHER VALUE? BASICALLY, SHOULD WE EXPECT SAMPLING ERROR AND TO WHAT DEGREE?]

We use two design criteria in order to choose the five new locations in Harris County: $\bar{Q}_{uk}(\mathbf{d})$ and $Q_{uk}^*(\mathbf{d})$, both defined in Section 3. For both criteria we plug in the MLEs for τ^2 , σ^2 , and ϕ . We assume that the goal is the predict average DM8 in all of Harris County, so we approximate the continuous versions of $\bar{Q}_{ik}(\mathbf{d})$ and $Q_{ik}^*(\mathbf{d})$ with the finite sample versions using a sample of 1,000 locations drawn uniformly from Harris 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 define $\bar{Q}_{uk}(\mathbf{d}) = Q_{uk}^*(\mathbf{d}) = \infty$ when any of the proposed locations are outside of Harris County.

[INSERT PSO RESULTS HERE... CURRENTLY RUNNING/PENDING TO BE RUN ON THE SERVER, THOUGH WE MAY WANT TO TWEAK IT - TAKES 2-3 DAYS TO RUN]

4 Discussion

[TO BE WRITTEN]

References

- Andrieu, C. and Thoms, J. (2008). “A tutorial on adaptive MCMC.” *Statistics and Computing*, 18, 4, 343–373.
- Blum, C. and Li, X. (2008). “Swarm Intelligence in Optimization.” In *Swarm Intelligence: Introduction and Applications*, eds. C. Blum and D. Merkle. Springer.
- Campos, M., Krohling, R. A., and Enriquez, I. (2014). “Bare bones particle swarm optimization with scale matrix adaptation.” *Cybernetics, IEEE Transactions on*, 44, 9, 1567–1578.
- Clerc, M. (2010). *Particle swarm optimization*. John Wiley & Sons.
- Clerc, M. and Kennedy, J. (2002). “The particle swarm-explosion, stability, and convergence in a multidimensional complex space.” *Evolutionary Computation, IEEE Transactions on*, 6, 1, 58–73.
- Cressie, N. and Wikle, C. K. (2015). *Statistics for Spatio-Temporal Data*. John Wiley & Sons.
- Eberhart, R. C. and Shi, Y. (2000). “Comparing inertia weights and constriction factors in particle swarm optimization.” In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1, 84–88. IEEE.
- Ebrahimi, N., Soofi, E. S., and Soyer, R. (2010). “Information Measures in Perspective.” *International Statistical Review*, 78, 3, 383–412.
- Fuentes, M., Chaudhuri, A., and Holland, D. M. (2007). “Bayesian Entropy for Spatial Sampling Design of Environmental Data.” *Environmental and Ecological Statistics*, 14, 3, 323–340.

- Gelman, A., Roberts, G., and Gilks, W. (1996). “Efficient Metropolis jumping rules.” *Bayesian statistics*, 5, 599-608, 42.
- Hsieh, H.-I. and Lee, T.-S. (2010). “A modified algorithm of bare bones particle swarm optimization.” *International Journal of Computer Science Issues*, 7, 11.
- Kennedy, J. (2003). “Bare bones particle swarms.” In *Swarm Intelligence Symposium, 2003. SIS’03. Proceedings of the 2003 IEEE*, 80–87. IEEE.
- Krohling, R., Mendel, E., et al. (2009). “Bare bones particle swarm optimization with Gaussian or Cauchy jumps.” In *Evolutionary Computation, 2009. CEC’09. IEEE Congress on*, 3285–3291. IEEE.
- Richer, T. J. and Blackwell, T. M. (2006). “The Lévy particle swarm.” In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, 808–815. IEEE.
- Tuppadung, Y. and 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, 65–70.
- Zimmerman, D. L. (2006). “Optimal Network Design for Spatial Prediction, Covariance Parameter Estimation, and Empirical Prediction.” *Environmetrics*, 17, 6, 635–652.

Adaptively-Tuned Particle Swarm Optimization with Application to Spatial Design

A PSO and BBPSO details

In Section 2 we introduced PSO, BBPSO, and our adaptively tuned variants of both. Here we describe in detail several BBPSO variants as well as a class of neighborhood topologies called the ring topologies.

A.1 BBPSO

The standard BBPSO algorithm was introduced by Kennedy (2003) and updates from t to $t + 1$ via equation (2). A commonly used variant of BBPSO also introduced by Kennedy (2003) is called BBPSOxp. In this variant, each coordinate of each particle has a 50% chance of updating according to (2) and a 50% chance of moving directly to that particle's personal best location on that coordinate. In other words

$$\theta_{ij}(t+1) = \begin{cases} N\left(\frac{p_{ij}(t)+g_{ij}(t)}{2}, s_{ij}^2(t)\right) & \text{with probability 0.5} \\ p_{ij}(t) & \text{otherwise,} \end{cases} \quad (\text{A.1})$$

where $s_{ij}(t) = |p_{ij}(t) - g_{ij}(t)|$.

A downside of both BBPSO and BBPSOxp is that any particle whose personal best is currently its group best location does not move due to the definition of the standard deviation term. Several methods have been proposed to overcome this; e.g. Hsieh and Lee (2010) and Zhang et al. (2011). Zhang et al. (2011) propose using mutation and crossover operations for the group best particle. To do this, each group best particle randomly selects three other distinct particles from the entire swarm, i_1 , i_2 , and i_3 , and updates according to

$$\theta_{ij}(t+1) = p_{i_1j}(t) + 0.5\{p_{i_2j}(t) - p_{i_3j}(t)\}. \quad (\text{A.2})$$

This combines easily with BBPSOxp to create BBPSOxp-MC by updating group best particles according to (A.2) and all other particles according to (A.1), but it does not completely solve the problem in BBPSOxp-MC. A particle which finds a personal best that is the same

as its group best on some but not all coordinates due to the “xp” component of BBPSOxp-MC will have a standard deviation of zero for those coordinates during the next iteration. This is easily overcome by using the “MC” operation for those coordinates as well. So both BBPSOxp-MC and BBPSO-MC can be encapsulated into the following general BBPSO algorithm. For each particle i , sample three other particles i_1, i_2, i_3 from the swarm, and for each coordinate j let $s_{ij}(t) = |p_{ij}(t) - g_{ij}(t)|$. Then the general updating equation is

$$\theta_{ij}(t+1) = \begin{cases} \begin{cases} N\left(\frac{p_{ij}(t)+g_{ij}(t)}{2}, s_{ij}^2(t)\right) & \text{with probability } \rho \\ p_{ij}(t) & \text{with probability } 1 - \rho, \end{cases} & \text{if } s_{ij}^2(t) > 0 \\ p_{i_1j}(t) + 0.5\{p_{i_2j}(t) - p_{i_3j}(t)\} & \text{otherwise.} \end{cases} \quad (\text{A.3})$$

When $\rho = 0$ we have BBPSO-MC, and when $\rho = 0.5$ we have BBPSOxp-MC. These are the two BBPSO variants that we consider.

Similarly, we can create adaptively tuned versions of BBPSO-MC and BBPSOxp-MC by analogy with (3) and (A.3):

$$\theta_{ij}(t+1) = \begin{cases} \begin{cases} t_{df}\left(\frac{p_{ij}(t)+g_{ij}(t)}{2}, \sigma^2(t)s_{ij}^2(t)\right) & \text{with probability } \rho \\ p_{ij}(t) & \text{with probability } 1 - \rho, \end{cases} & \text{if } s_{ij}^2(t) > 0 \\ p_{i_1j}(t) + 0.5\{p_{i_2j}(t) - p_{i_3j}(t)\} & \text{otherwise.} \end{cases}$$

$$\log \sigma^2(t+1) = \log \sigma^2(t) + c \times \{R(t+1) - R^*\},$$

where $R(t) = \#\{i : Q(\mathbf{p}_i(t)) > Q(\mathbf{p}_i(t-1))\}/n$, and df , R^* , and c are defined in Section 2.1. Then AT-BBPSO-MC sets $\rho = 0$ and AT-BBPSOxp-MC sets $\rho = 0.5$.

B Comparing PSO and BBPSO algorithms

In order to compare AT-BBPSO to other PSO variants, we employ a subset of test functions used in Hsieh and Lee (2010). Each function is listed in Table 1 along with the global

maximum and argmax, and the initialization range for the simulations. Further description of many of these functions can be found in Clerc (2010). For each function, we set $D = 20$ so the domain of each function is \mathfrak{R}^{20} . For each function, the PSO algorithms are initialized in a range that does not contain the true maximum.

Equation	ArgMax	Maximum	Initialization
$Q_1(\boldsymbol{\theta}) = -\sum_{i=1}^D \theta_i^2$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_1(\boldsymbol{\theta}^*) = 0$	$(50, 100)^D$
$Q_2(\boldsymbol{\theta}) = -\sum_{i=1}^D \left(\sum_{j=1}^i \theta_j\right)^2$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_2(\boldsymbol{\theta}^*) = 0$	$(50, 100)^D$
$Q_3(\boldsymbol{\theta}) = -\sum_{i=1}^{D-1} [100\{\theta_{i+1} + 1 - (\theta_i + 1)^2\} + \theta_i^2]$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_3(\boldsymbol{\theta}^*) = 0$	$(15, 30)^D$
$Q_4(\boldsymbol{\theta}) = 9D - \sum_{i=1}^D \{\theta_i^2 - \cos(2\pi\theta_i) + 10\}$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_4(\boldsymbol{\theta}^*) = 0$	$(2.56, 5.12)^D$
$Q_5(\boldsymbol{\theta}) = -\frac{1}{4000}\ \boldsymbol{\theta}\ ^2 + \prod_{i=1}^D \cos\left(\frac{\theta_i}{\sqrt{i}}\right) - 1$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_5(\boldsymbol{\theta}^*) = 0$	$(300, 600)^D$
$Q_6(\boldsymbol{\theta}) = 20 \exp\left(-0.2\sqrt{\frac{1}{D}\ \boldsymbol{\theta}\ }\right) + \exp\left\{\frac{1}{D}\sum_{i=1}^D \cos(2\pi\theta_i)\right\} - 20 - \exp(1)$	$\boldsymbol{\theta}^* = \mathbf{0}$	$Q_6(\boldsymbol{\theta}^*) = 0$	$(16, 32)^D$

Table 1: Test functions for evaluating PSO algorithms. The dimension of $\boldsymbol{\theta}$ is D and $\|\cdot\|$ is the Euclidean norm: $\|\boldsymbol{\theta}\| = \sqrt{\sum_{i=1}^D \theta_i^2}$.

We use several PSO algorithms in the simulation study. The standard PSO algorithm uses the parameter values suggested by Blum and Li (2008) and Clerc and Kennedy (2002). The AT-BBPSO variants are implemented a wide variety of parameter values, but all have the scale parameter initialized at $\sigma(0) = 1$, and both set $c = 0.1$. The AT-PSO variants are initialized at $\omega(0) = 1$ and also set $c = 0.1$. In addition, each algorithm is implemented using each of three neighborhood structures — the global, ring-3, and ring-1 neighborhoods. Each algorithm was used to optimize each objective function for 500 iterations over 50 replications using 20 particles. Initializations were changed across replications but held constant across algorithms. The standard PSO, DI-PSO, and AT-PSO algorithms initialized their velocity terms using the same method as a function of the initial locations of the particles, which we will denote by \mathbf{x}_i for particle i . Let $x_{max,j}$ be the maximum

initial value of coordinate j of \mathbf{x}_i for each particle i , and let $x_{min,j}$ be the corresponding minimum. Then let $d_{max} = \max_j x_{max,j} - x_{min,j}$. Then we initialize the velocities with $v_{ij}(0) \stackrel{iid}{\sim} U(-d_{max}/2, d_{max}/2)$. Tables 2-7 contain the simulation results for objective functions 1-6 respectively (OF1, OF2, etc.). We use several measures to quantify how well each algorithm finds the global maximum. First, each table includes the mean and standard deviation of the absolute difference between the true global maximum and the algorithm's estimated global maximum across all 50 replications, denoted by Mean and SD. Second, each table includes a convergence criterion — the proportion of the replications that came within 0.01 of the true global maximum, denoted by \hat{p} . Finally, \hat{t} denotes the median number of iterations until the algorithm reaches the convergence criterion. When $\hat{p} < 0.5$ then $\hat{t} = \infty$ since greater than 50% of the replications did not converge in the maximum number of iterations allowed. Then Mean, \hat{p} , and \hat{t} can be thought of how close the algorithm gets to the global maximum on average, what proportion of the time it converges, and long it takes to converge respectively.

We highlight only some of the features of these tables. First and foremost, PSO, BBPSO-MC, and BBPSOxp-MC almost always do worse than their AT cousins. Adaptively tuning either the scale or inertia parameter leads to gains in all three of our measures, sometimes large. The comparison is starkest between BBPSO variants and AT-BBPSO variants, partially because BBPSO tends to be pretty bad but also because AT-BBPSO does very well. Second, for most non-AT algorithms the more restrictive neighborhoods appear to yield algorithms which do a better job of finding the global max. For the AT-PSO algorithms, it appears that the target improvement rate (R^*) and the neighborhood interact. When the rate is high a more restrictive neighborhood is preferable, while when the rate is low a less restrictive neighborhood is preferable. On the other hand, for the AT-BBPSO algorithms, the opposite appears to be true — when the target improvement rate is high, a less restrictive neighborhood is desirable and vice versa. In addition, the best AT-BBPSO algorithms

often use the global neighborhood while for the other classes of algorithms their best versions typically use the ring-1 or ring-3 neighborhood.

For the DI-PSO algorithms often there is a parameter-neighborhood combination that does well, typically from setting $\alpha = 200$ (20% of the 500 iterations) and $\beta = 1$ and using either the ring-1 or ring-3 neighborhood. One of these combinations typically does the best of all the DI-PSO algorithms but they can sometimes still do much worse than the best alternatives (e.g., for OF2). In the AT-BBPSO algorithms, it is not always clear what the best parameter settings are, but good default values appear to be $df = 3$ or 5 and $R^* = 0.3$ or 0.5 . When these values are good, they often lead to the best performing PSO algorithms we consider. However, it is not always clear whether to use AT-BBPSO-MC or AT-BBPSOxp-MC. For some objective functions, e.g. OF4, the xp version is consistently better than the non-xp alternative, but the opposite is true for others, e.g. OF2.. AT-PSO is also often very competitive with $R^* = 0.3$ or $R^* = 0.5$, though again sometimes different parameter settings also appear to work well.

The DI-PSO and AT-PSO algorithms are similar conceptually, but often yield very different results. DI-PSO deterministically reduces the inertia parameter over time in the same manner given a fixed set of parameter values (α and β), while AT-PSO dynamically adjusts the inertia parameter to hit a target improvement rate. Figure B.1 plots the inertia over time for the DI-PSO algorithm with $\alpha = 200$ and $\beta = 1$, and observed inertia over time for one replication of the AT-PSO algorithm with target rate $R^* = 0.5$ and ring-1 neighborhood for OF1 and one replication for OF6. All three algorithms have an initial inertia of $\omega(0) = 1$. While DI-PSO smoothly decreases its inertia with a slowly decreasing rate, AT-PSO very quickly drops its inertia for OF1 to about 0.55 then bounces around around near that point. It also jumps up above 1 initially, imploring the particles to cast a wider net in search of higher value areas of the search space. This is pretty typical behavior for the inertia parameter of AT-PSO — it tends to bounce around a level which is approximately

the average over time of the DI-PSO’s inertia, though lower values of R^* will result in higher inertias. In this way, AT-PSO alternates periods of exploration (relatively high inertia) and periods of exploitation (relatively low inertia). The main exception to this pattern is when AT-PSO converges around a local maximum. In this case, inertia plummets to zero as the particles settle down. This is precisely what happens for OF6 in Figure B.1, though in this case the maximum is not global — Table 7 indicates that ring-1 AT-PSO with $R^* = 0.5$ never converged to the global max. In optimization problems with multiple local optima, both AT-PSO and AT-BBPSO variants can exhibit this behavior and prematurely converge to a local optima, so they may not be advantageous for those problems.

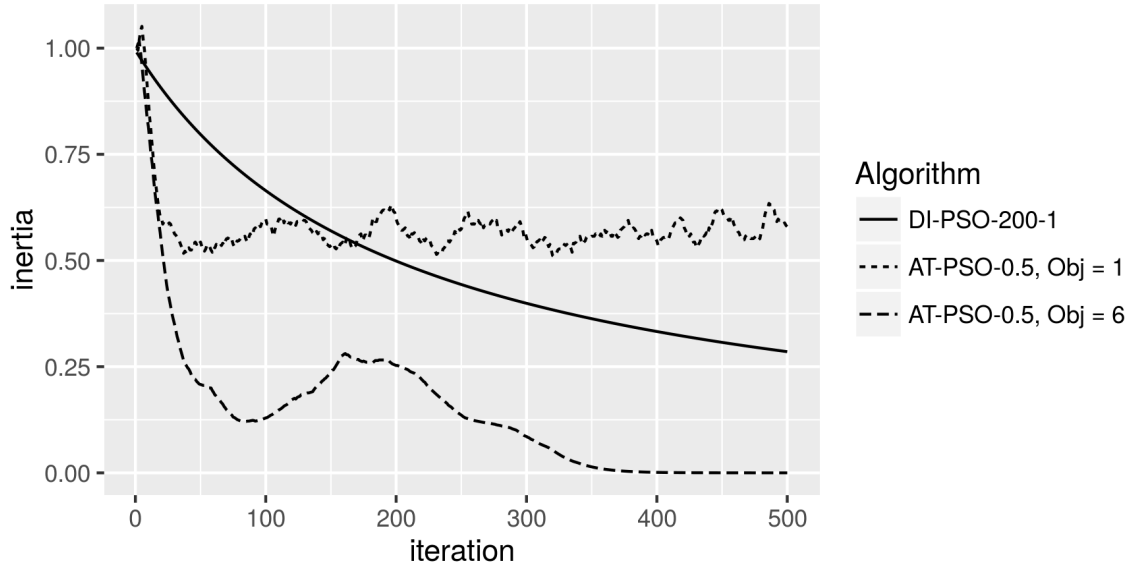


Figure B.1: Inertia over time for the DI-PSO algorithm with $\alpha = 200$ and $\beta = 1$, and for one replication of the AT-PSO-0.5 algorithm for each of OFs 1 and 6.

Based on these simulations, our default recommendation is to use AT-BBPSO-MC or AT-BBPSOxp-MC with $R^* = 0.3$ or 0.5 and $df = 3$ or $df = 5$ along with the global neighborhood. These algorithms will not always be the best of the PSO algorithms, but they will often be very good. AT-PSO with $R^* = 0.3$ or $R^* = 0.5$ with a restrictive neighborhood topology

such as ring-3 also tends to be a very good choice, though perhaps less consistent than the AT-BBPSO variants. Default PSO also performs rather well and is a good baseline algorithm to use for comparisons.

OF1		Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm		Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO		4015.81	3124.33	0.00	∞	0.00	0.01	0.98	362.00	0.01	0.01	0.78	488.50
BBPSO-MC		84517.26	8944.33	0.00	∞	83674.80	9956.73	0.00	∞	84784.51	9451.47	0.00	∞
BBPSOxp-MC		84104.44	8192.44	0.00	∞	82952.29	9655.82	0.00	∞	85886.72	7182.36	0.00	∞
AT-BBPSO-MC													
$df = 1, R^* = 0.1$		1.59	0.47	0.00	∞	2.08	0.73	0.00	∞	2.10	0.89	0.00	∞
$df = 1, R^* = 0.3$		0.00	0.00	1.00	445.00	0.01	0.00	0.96	467.00	0.01	0.01	0.54	499.50
$df = 1, R^* = 0.5$		0.00	0.00	1.00	289.00	0.00	0.00	1.00	309.50	0.00	0.00	1.00	348.00
$df = 1, R^* = 0.7$		0.00	0.00	1.00	223.00	0.00	0.00	1.00	256.00	0.00	0.00	1.00	329.00
$df = 3, R^* = 0.1$		6.92	2.41	0.00	∞	6.92	1.79	0.00	∞	5.10	2.02	0.00	∞
$df = 3, R^* = 0.3$		0.01	0.00	0.22	∞	0.02	0.01	0.06	∞	0.03	0.01	0.00	∞
$df = 3, R^* = 0.5$		0.00	0.00	1.00	331.50	0.00	0.00	1.00	347.50	0.00	0.00	1.00	387.50
$df = 3, R^* = 0.7$		0.00	0.00	1.00	259.50	0.00	0.00	1.00	288.50	0.00	0.00	1.00	400.50
$df = 5, R^* = 0.1$		11.50	2.93	0.00	∞	11.72	2.53	0.00	∞	8.14	2.55	0.00	∞
$df = 5, R^* = 0.3$		0.03	0.01	0.00	∞	0.04	0.01	0.00	∞	0.06	0.02	0.00	∞
$df = 5, R^* = 0.5$		0.00	0.00	1.00	356.50	0.00	0.00	1.00	368.50	0.00	0.00	1.00	415.50
$df = 5, R^* = 0.7$		0.00	0.00	1.00	288.50	0.00	0.00	1.00	326.50	222.69	288.53	0.02	∞
$df = \infty, R^* = 0.1$		42.86	10.31	0.00	∞	38.44	8.12	0.00	∞	20.28	8.59	0.00	∞
$df = \infty, R^* = 0.3$		0.14	0.05	0.00	∞	0.15	0.03	0.00	∞	0.25	0.08	0.00	∞
$df = \infty, R^* = 0.5$		0.00	0.00	1.00	415.00	0.00	0.00	1.00	433.00	0.01	0.00	0.94	489.00
$df = \infty, R^* = 0.7$		0.00	0.00	1.00	352.00	0.00	0.00	0.98	441.50	16949.57	3141.36	0.00	∞
AT-BBPSOxp-MC													
$df = 1, R^* = 0.1$		5.71	1.58	0.00	∞	6.00	1.76	0.00	∞	3.91	1.75	0.00	∞
$df = 1, R^* = 0.3$		0.02	0.01	0.00	∞	0.04	0.01	0.00	∞	0.05	0.02	0.00	∞
$df = 1, R^* = 0.5$		0.00	0.00	1.00	358.00	0.00	0.00	1.00	381.00	0.00	0.00	1.00	419.50
$df = 1, R^* = 0.7$		0.00	0.00	1.00	291.00	0.00	0.00	1.00	326.00	0.00	0.01	0.98	432.50
$df = 3, R^* = 0.1$		17.42	4.87	0.00	∞	16.91	4.41	0.00	∞	8.64	3.12	0.00	∞
$df = 3, R^* = 0.3$		0.10	0.03	0.00	∞	0.12	0.04	0.00	∞	0.15	0.05	0.00	∞
$df = 3, R^* = 0.5$		0.00	0.00	1.00	412.00	0.00	0.00	1.00	431.50	0.01	0.00	0.92	470.50
$df = 3, R^* = 0.7$		0.00	0.00	1.00	350.50	0.00	0.01	0.98	438.00	4193.16	2013.59	0.00	∞
$df = 5, R^* = 0.1$		28.39	6.37	0.00	∞	28.42	7.58	0.00	∞	13.22	5.27	0.00	∞
$df = 5, R^* = 0.3$		0.19	0.05	0.00	∞	0.22	0.07	0.00	∞	0.26	0.10	0.00	∞
$df = 5, R^* = 0.5$		0.00	0.00	1.00	440.00	0.00	0.00	1.00	466.00	0.03	0.02	0.04	∞
$df = 5, R^* = 0.7$		0.00	0.00	1.00	405.50	505.43	755.11	0.02	∞	17674.52	4505.53	0.00	∞
$df = \infty, R^* = 0.1$		65.18	14.92	0.00	∞	60.90	13.82	0.00	∞	26.77	10.11	0.00	∞
$df = \infty, R^* = 0.3$		0.50	0.11	0.00	∞	0.66	0.14	0.00	∞	0.82	0.25	0.00	∞
$df = \infty, R^* = 0.5$		0.01	0.00	0.82	492.00	0.03	0.01	0.02	∞	0.85	0.75	0.00	∞
$df = \infty, R^* = 0.7$		0.19	0.33	0.04	∞	17369.34	3631.71	0.00	∞	35172.80	4377.92	0.00	∞
DI-PSO													
$\alpha = 50, \beta = 1$		6703.90	3993.99	0.00	∞	543.02	683.48	0.00	∞	51.63	95.71	0.00	∞
$\alpha = 50, \beta = 2$		7823.89	4787.27	0.00	∞	1821.42	1508.50	0.00	∞	499.64	441.33	0.00	∞
$\alpha = 50, \beta = 4$		15373.54	6906.92	0.00	∞	5572.40	3137.25	0.00	∞	3705.93	2324.41	0.00	∞
$\alpha = 100, \beta = 1$		3928.51	3855.22	0.00	∞	236.79	733.83	0.00	∞	0.99	4.03	0.20	∞
$\alpha = 100, \beta = 2$		7618.89	4250.92	0.00	∞	1206.09	1393.42	0.00	∞	84.99	137.32	0.00	∞
$\alpha = 100, \beta = 4$		13044.34	6195.43	0.00	∞	4218.13	2881.34	0.00	∞	1745.64	1532.00	0.00	∞
$\alpha = 200, \beta = 1$		2321.68	2088.02	0.00	∞	178.09	618.06	0.02	∞	0.00	0.00	1.00	383.50
$\alpha = 200, \beta = 2$		6427.62	3741.12	0.00	∞	305.62	523.71	0.00	∞	5.45	12.55	0.00	∞
$\alpha = 200, \beta = 4$		12004.75	5708.12	0.00	∞	2236.94	1729.58	0.00	∞	475.93	503.87	0.00	∞
AT-PSO													
$R^* = 0.1$		111.02	237.73	0.00	∞	211.79	889.18	0.00	∞	12223.23	9590.30	0.00	∞
$R^* = 0.3$		54.27	252.08	0.04	∞	0.00	0.00	1.00	347.00	0.00	0.01	0.88	457.00
$R^* = 0.5$		329.77	586.04	0.00	∞	0.00	0.00	1.00	274.50	0.00	0.00	1.00	303.50
$R^* = 0.7$		2131.94	2209.74	0.00	∞	0.04	0.21	0.92	381.50	0.00	0.00	1.00	341.50

Table 2: Simulation results for OF1. See text for description.

OF2		Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm		Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO		10915.05	17201.47	0.00	∞	1246.38	1002.75	0.00	∞	5738.34	3835.37	0.00	∞
BBPSO-MC		9081909.83	1958288.73	0.00	∞	9194595.70	2157840.19	0.00	∞	8830368.63	2234188.42	0.00	∞
BBPSOxp-MC		9138885.89	2265920.11	0.00	∞	8889637.56	2035964.50	0.00	∞	9038770.99	1858659.66	0.00	∞
AT-BBPSO-MC													
$df = 1, R^* = 0.1$		950.30	809.12	0.00	∞	690.95	535.97	0.00	∞	743.12	728.28	0.00	∞
$df = 1, R^* = 0.3$		300.03	258.36	0.00	∞	685.68	458.39	0.00	∞	1091.45	1375.54	0.00	∞
$df = 1, R^* = 0.5$		440.38	436.08	0.00	∞	1419.88	1239.43	0.00	∞	3652.45	4072.28	0.00	∞
$df = 1, R^* = 0.7$		1615.81	1539.22	0.00	∞	4330.31	4863.92	0.00	∞	8858.46	4578.27	0.00	∞
$df = 3, R^* = 0.1$		196.07	88.33	0.00	∞	226.17	163.16	0.00	∞	252.63	187.08	0.00	∞
$df = 3, R^* = 0.3$		23.27	17.11	0.00	∞	54.95	48.43	0.00	∞	189.83	399.03	0.00	∞
$df = 3, R^* = 0.5$		30.94	29.21	0.00	∞	185.87	161.77	0.00	∞	1073.22	1233.74	0.00	∞
$df = 3, R^* = 0.7$		222.12	524.53	0.00	∞	939.33	789.54	0.00	∞	5897.58	4513.86	0.00	∞
$df = 5, R^* = 0.1$		198.27	75.35	0.00	∞	175.31	80.21	0.00	∞	205.86	117.06	0.00	∞
$df = 5, R^* = 0.3$		12.25	9.35	0.00	∞	24.76	18.02	0.00	∞	60.79	63.04	0.00	∞
$df = 5, R^* = 0.5$		11.51	7.41	0.00	∞	63.91	51.53	0.00	∞	458.47	317.21	0.00	∞
$df = 5, R^* = 0.7$		61.67	36.27	0.00	∞	378.27	218.82	0.00	∞	7696.71	5485.99	0.00	∞
$df = \infty, R^* = 0.1$		280.79	97.77	0.00	∞	267.44	87.93	0.00	∞	287.37	128.28	0.00	∞
$df = \infty, R^* = 0.3$		6.75	3.83	0.00	∞	13.91	9.02	0.00	∞	39.37	32.01	0.00	∞
$df = \infty, R^* = 0.5$		4.96	3.73	0.00	∞	30.82	22.83	0.00	∞	269.58	161.10	0.00	∞
$df = \infty, R^* = 0.7$		51.31	38.92	0.00	∞	550.48	301.85	0.00	∞	1100097.72	493290.33	0.00	∞
AT-BBPSOxp-MC													
$df = 1, R^* = 0.1$		1620.04	724.59	0.00	∞	1434.81	637.94	0.00	∞	1004.77	608.51	0.00	∞
$df = 1, R^* = 0.3$		1117.65	614.18	0.00	∞	1541.55	913.10	0.00	∞	1372.47	1108.92	0.00	∞
$df = 1, R^* = 0.5$		2424.76	1458.65	0.00	∞	3639.53	2048.74	0.00	∞	5212.21	3221.11	0.00	∞
$df = 1, R^* = 0.7$		5685.79	3365.32	0.00	∞	6746.82	4254.76	0.00	∞	13170.58	7291.81	0.00	∞
$df = 3, R^* = 0.1$		609.04	289.92	0.00	∞	568.65	326.15	0.00	∞	562.17	301.77	0.00	∞
$df = 3, R^* = 0.3$		231.40	167.47	0.00	∞	320.46	308.01	0.00	∞	366.68	274.58	0.00	∞
$df = 3, R^* = 0.5$		360.34	224.38	0.00	∞	1026.42	709.48	0.00	∞	5089.66	5857.62	0.00	∞
$df = 3, R^* = 0.7$		1426.88	1209.92	0.00	∞	5568.83	7882.29	0.00	∞	38081.89	31747.85	0.00	∞
$df = 5, R^* = 0.1$		476.81	215.75	0.00	∞	497.74	194.10	0.00	∞	540.99	263.49	0.00	∞
$df = 5, R^* = 0.3$		109.01	45.98	0.00	∞	170.67	120.10	0.00	∞	272.26	192.17	0.00	∞
$df = 5, R^* = 0.5$		243.73	154.52	0.00	∞	448.77	241.51	0.00	∞	2147.17	1308.53	0.00	∞
$df = 5, R^* = 0.7$		794.68	388.65	0.00	∞	17971.50	16875.10	0.00	∞	507347.01	409181.08	0.00	∞
$df = \infty, R^* = 0.1$		609.19	197.59	0.00	∞	585.88	163.05	0.00	∞	536.88	263.19	0.00	∞
$df = \infty, R^* = 0.3$		70.82	44.69	0.00	∞	111.39	79.50	0.00	∞	220.30	168.06	0.00	∞
$df = \infty, R^* = 0.5$		114.86	88.86	0.00	∞	329.69	169.04	0.00	∞	2889.16	2088.86	0.00	∞
$df = \infty, R^* = 0.7$		2469.49	2799.80	0.00	∞	1483405.94	407996.51	0.00	∞	2535125.64	778910.46	0.00	∞
DI-PSO													
$\alpha = 50, \beta = 1$		40714.56	24208.86	0.00	∞	16548.67	9312.34	0.00	∞	7839.43	5031.49	0.00	∞
$\alpha = 50, \beta = 2$		47763.21	31361.92	0.00	∞	24627.59	11610.10	0.00	∞	11660.25	6362.56	0.00	∞
$\alpha = 50, \beta = 4$		62225.74	28459.35	0.00	∞	31511.13	15145.07	0.00	∞	16416.52	7831.71	0.00	∞
$\alpha = 100, \beta = 1$		26613.11	14611.63	0.00	∞	10168.37	6029.10	0.00	∞	5378.87	3302.21	0.00	∞
$\alpha = 100, \beta = 2$		46177.71	37974.29	0.00	∞	17795.63	8897.56	0.00	∞	9818.51	4708.72	0.00	∞
$\alpha = 100, \beta = 4$		64833.73	36534.52	0.00	∞	28627.71	14257.04	0.00	∞	14536.17	7921.94	0.00	∞
$\alpha = 200, \beta = 1$		21611.55	12914.90	0.00	∞	5699.97	3916.44	0.00	∞	3499.49	1865.32	0.00	∞
$\alpha = 200, \beta = 2$		36405.44	27997.69	0.00	∞	12492.90	6829.93	0.00	∞	6944.87	4559.84	0.00	∞
$\alpha = 200, \beta = 4$		57723.16	30860.32	0.00	∞	21807.81	11567.18	0.00	∞	12201.41	5365.69	0.00	∞
AT-PSO													
$R^* = 0.1$		8823.66	9727.43	0.00	∞	8852.96	6912.25	0.00	∞	58369.56	38288.17	0.00	∞
$R^* = 0.3$		6658.84	6659.87	0.00	∞	769.74	850.01	0.00	∞	5183.03	4046.47	0.00	∞
$R^* = 0.5$		20994.61	16398.66	0.00	∞	1713.89	2589.20	0.00	∞	1877.08	1465.51	0.00	∞
$R^* = 0.7$		35128.87	17886.90	0.00	∞	7126.76	5756.50	0.00	∞	5771.37	3581.95	0.00	∞

Table 3: Simulation results for OF2. See text for description.

OF3		Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm		Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO		1242764.67	2075592.00	0.00	∞	121.33	104.18	0.00	∞	241.45	255.69	0.00	∞
BBPSO-MC		232830400.95	37473167.09	0.00	∞	247662755.14	43820300.43	0.00	∞	247321508.63	39096259.33	0.00	∞
BBPSOxp-MC		235491326.54	42124202.61	0.00	∞	242679907.63	42872974.55	0.00	∞	253599464.23	43615037.04	0.00	∞
AT-BBPSO-MC													
$df = 1, R^* = 0.1$		382.54	325.35	0.00	∞	486.70	533.46	0.00	∞	669.09	932.66	0.00	∞
$df = 1, R^* = 0.3$		233.17	429.80	0.00	∞	182.89	272.06	0.00	∞	352.64	465.63	0.00	∞
$df = 1, R^* = 0.5$		315.43	497.55	0.00	∞	159.55	216.03	0.00	∞	254.78	669.72	0.00	∞
$df = 1, R^* = 0.7$		235.08	455.38	0.00	∞	96.32	106.25	0.00	∞	189.20	288.45	0.00	∞
$df = 3, R^* = 0.1$		545.05	481.33	0.00	∞	629.79	657.37	0.00	∞	567.75	585.07	0.00	∞
$df = 3, R^* = 0.3$		176.81	186.22	0.00	∞	172.84	204.18	0.00	∞	256.32	426.46	0.00	∞
$df = 3, R^* = 0.5$		159.09	170.77	0.00	∞	128.99	127.25	0.00	∞	137.56	202.68	0.00	∞
$df = 3, R^* = 0.7$		155.74	399.33	0.00	∞	132.16	339.64	0.00	∞	82.12	170.21	0.00	∞
$df = 5, R^* = 0.1$		833.71	627.25	0.00	∞	928.42	982.63	0.00	∞	450.93	444.86	0.00	∞
$df = 5, R^* = 0.3$		283.78	431.37	0.00	∞	135.95	98.01	0.00	∞	229.39	314.25	0.00	∞
$df = 5, R^* = 0.5$		198.53	185.97	0.00	∞	173.66	278.73	0.00	∞	118.79	114.49	0.00	∞
$df = 5, R^* = 0.7$		142.06	190.31	0.00	∞	64.57	98.26	0.00	∞	108.71	538.33	0.00	∞
$df = \infty, R^* = 0.1$		1343.31	757.87	0.00	∞	1056.96	522.89	0.00	∞	958.18	773.33	0.00	∞
$df = \infty, R^* = 0.3$		230.32	320.08	0.00	∞	138.02	120.68	0.00	∞	212.78	331.90	0.00	∞
$df = \infty, R^* = 0.5$		225.28	219.75	0.00	∞	156.78	132.24	0.00	∞	120.23	106.23	0.00	∞
$df = \infty, R^* = 0.7$		131.34	91.53	0.00	∞	56.10	76.80	0.00	∞	179.69	683.04	0.00	∞
AT-BBPSOxp-MC													
$df = 1, R^* = 0.1$		616.80	278.83	0.00	∞	633.00	276.25	0.00	∞	490.75	350.79	0.00	∞
$df = 1, R^* = 0.3$		152.80	179.75	0.00	∞	163.64	131.42	0.00	∞	140.13	158.06	0.00	∞
$df = 1, R^* = 0.5$		139.49	269.65	0.00	∞	99.39	225.41	0.00	∞	202.26	429.55	0.00	∞
$df = 1, R^* = 0.7$		149.01	459.29	0.00	∞	73.54	126.07	0.00	∞	188.28	370.67	0.00	∞
$df = 3, R^* = 0.1$		961.94	370.97	0.00	∞	931.38	294.49	0.00	∞	709.41	452.45	0.00	∞
$df = 3, R^* = 0.3$		168.44	279.24	0.00	∞	144.15	200.78	0.00	∞	120.33	48.68	0.00	∞
$df = 3, R^* = 0.5$		123.39	102.98	0.00	∞	97.56	50.88	0.00	∞	77.47	81.83	0.00	∞
$df = 3, R^* = 0.7$		49.75	70.72	0.00	∞	33.96	52.12	0.00	∞	3586.02	9973.24	0.00	∞
$df = 5, R^* = 0.1$		1229.87	526.97	0.00	∞	1171.93	406.81	0.00	∞	727.95	436.07	0.00	∞
$df = 5, R^* = 0.3$		267.86	435.67	0.00	∞	124.80	75.86	0.00	∞	157.72	217.77	0.00	∞
$df = 5, R^* = 0.5$		137.08	104.13	0.00	∞	91.44	86.38	0.00	∞	75.74	95.04	0.00	∞
$df = 5, R^* = 0.7$		66.87	126.23	0.00	∞	19.28	17.08	0.00	∞	82498.29	102684.12	0.00	∞
$df = \infty, R^* = 0.1$		2002.70	822.46	0.00	∞	1718.67	698.34	0.00	∞	1158.47	903.16	0.00	∞
$df = \infty, R^* = 0.3$		137.09	84.51	0.00	∞	116.46	66.60	0.00	∞	114.47	57.68	0.00	∞
$df = \infty, R^* = 0.5$		121.71	103.55	0.00	∞	104.78	67.31	0.00	∞	41.15	33.00	0.00	∞
$df = \infty, R^* = 0.7$		24.31	24.58	0.00	∞	24.31	20.57	0.00	∞	2088154.00	1258187.91	0.00	∞
DI-PSO													
$\alpha = 50, \beta = 1$		4180196.82	4677639.48	0.00	∞	354071.40	579847.34	0.00	∞	70128.62	165037.08	0.00	∞
$\alpha = 50, \beta = 2$		6964209.15	8322398.93	0.00	∞	2341768.78	4277548.98	0.00	∞	1232022.80	1464906.82	0.00	∞
$\alpha = 50, \beta = 4$		20143670.82	15026478.57	0.00	∞	11740754.91	8374232.25	0.00	∞	6766696.51	4472751.26	0.00	∞
$\alpha = 100, \beta = 1$		1610827.18	3342137.70	0.00	∞	55104.82	171942.01	0.00	∞	1691.16	3160.57	0.00	∞
$\alpha = 100, \beta = 2$		6945649.64	6844592.19	0.00	∞	1001725.17	1774982.73	0.00	∞	150478.59	213009.16	0.00	∞
$\alpha = 100, \beta = 4$		21536033.79	15337048.80	0.00	∞	6691579.83	5932202.88	0.00	∞	4328277.03	4060546.38	0.00	∞
$\alpha = 200, \beta = 1$		1543367.07	2077471.63	0.00	∞	4350.55	16257.10	0.00	∞	266.44	453.75	0.00	∞
$\alpha = 200, \beta = 2$		6626917.67	9093854.31	0.00	∞	299834.83	1040979.02	0.00	∞	5848.88	14329.18	0.00	∞
$\alpha = 200, \beta = 4$		17830796.63	14245007.26	0.00	∞	3367710.63	3194612.95	0.00	∞	776424.69	865443.75	0.00	∞
AT-PSO													
$R^* = 0.1$		5779.95	14928.86	0.00	∞	297272.41	1261108.56	0.00	∞	33785179.04	23020834.78	0.00	∞
$R^* = 0.3$		869.69	2100.98	0.00	∞	131.15	175.33	0.00	∞	184.15	186.17	0.00	∞
$R^* = 0.5$		71363.37	218211.42	0.00	∞	190.13	271.20	0.00	∞	134.32	200.92	0.00	∞
$R^* = 0.7$		1572669.78	3884482.27	0.00	∞	391.10	857.02	0.00	∞	535.09	1277.83	0.00	∞

Table 4: Simulation results for OF3. See text for description.

OF4		Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm		Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO		29.94	15.21	0.00	∞	5.37	3.47	0.02	∞	3.24	1.44	0.00	∞
BBPSO-MC		237.31	18.67	0.00	∞	232.86	24.40	0.00	∞	232.69	21.86	0.00	∞
BBPSOxp-MC		229.80	22.24	0.00	∞	236.60	25.20	0.00	∞	230.84	21.05	0.00	∞
AT-BBPSO-MC													
$df = 1, R^* = 0.1$		4.96	1.58	0.00	∞	3.93	1.29	0.00	∞	4.07	1.88	0.00	∞
$df = 1, R^* = 0.3$		5.14	1.89	0.00	∞	2.76	1.45	0.06	∞	2.48	1.57	0.06	∞
$df = 1, R^* = 0.5$		6.32	2.16	0.00	∞	4.68	1.94	0.00	∞	4.24	2.36	0.02	∞
$df = 1, R^* = 0.7$		8.56	2.84	0.00	∞	5.78	1.67	0.00	∞	6.03	2.01	0.00	∞
$df = 3, R^* = 0.1$		5.44	1.30	0.00	∞	5.40	1.17	0.00	∞	5.39	1.54	0.00	∞
$df = 3, R^* = 0.3$		3.56	1.76	0.00	∞	1.59	1.24	0.20	∞	1.43	1.00	0.18	∞
$df = 3, R^* = 0.5$		4.07	1.85	0.00	∞	2.70	1.57	0.08	∞	2.57	1.44	0.02	∞
$df = 3, R^* = 0.7$		4.70	1.73	0.00	∞	3.83	1.72	0.00	∞	3.43	1.75	0.06	∞
$df = 5, R^* = 0.1$		6.63	1.34	0.00	∞	6.45	1.92	0.00	∞	6.72	1.86	0.00	∞
$df = 5, R^* = 0.3$		2.80	1.51	0.02	∞	1.24	1.12	0.24	∞	1.13	1.21	0.30	∞
$df = 5, R^* = 0.5$		4.11	2.16	0.02	∞	2.26	1.60	0.12	∞	1.90	1.43	0.16	∞
$df = 5, R^* = 0.7$		4.40	2.00	0.00	∞	2.91	1.48	0.00	∞	3.56	1.84	0.00	∞
$df = \infty, R^* = 0.1$		9.12	1.88	0.00	∞	9.17	1.58	0.00	∞	9.42	1.70	0.00	∞
$df = \infty, R^* = 0.3$		2.15	1.31	0.00	∞	0.72	0.92	0.00	∞	0.61	0.74	0.04	∞
$df = \infty, R^* = 0.5$		2.74	1.55	0.04	∞	1.62	1.38	0.18	∞	1.01	0.91	0.26	∞
$df = \infty, R^* = 0.7$		3.22	1.78	0.06	∞	2.11	1.56	0.18	∞	2.13	1.30	0.10	∞
AT-BBPSOxp-MC													
$df = 1, R^* = 0.1$		4.25	1.24	0.00	∞	4.70	1.25	0.00	∞	6.34	1.56	0.00	∞
$df = 1, R^* = 0.3$		0.74	0.87	0.04	∞	0.23	0.44	0.02	∞	0.51	0.74	0.00	∞
$df = 1, R^* = 0.5$		1.75	1.24	0.12	∞	0.95	1.00	0.38	∞	1.41	1.16	0.18	∞
$df = 1, R^* = 0.7$		3.07	1.65	0.02	∞	2.14	1.20	0.06	∞	3.10	1.62	0.02	∞
$df = 3, R^* = 0.1$		5.34	1.01	0.00	∞	6.29	1.31	0.00	∞	7.39	1.59	0.00	∞
$df = 3, R^* = 0.3$		0.25	0.45	0.00	∞	0.08	0.23	0.00	∞	0.19	0.41	0.02	∞
$df = 3, R^* = 0.5$		0.65	0.78	0.52	345.50	0.30	0.52	0.72	342.00	0.56	0.66	0.46	∞
$df = 3, R^* = 0.7$		1.43	1.21	0.22	∞	1.24	1.02	0.24	∞	1.85	1.12	0.08	∞
$df = 5, R^* = 0.1$		6.02	1.08	0.00	∞	6.50	1.08	0.00	∞	7.94	1.31	0.00	∞
$df = 5, R^* = 0.3$		0.10	0.26	0.00	∞	0.08	0.23	0.00	∞	0.10	0.33	0.02	∞
$df = 5, R^* = 0.5$		0.40	0.61	0.66	338.00	0.25	0.42	0.74	338.50	0.46	0.70	0.64	367.00
$df = 5, R^* = 0.7$		1.13	1.01	0.28	∞	0.75	0.85	0.44	∞	1.74	1.10	0.04	∞
$df = \infty, R^* = 0.1$		7.81	1.19	0.00	∞	8.10	1.36	0.00	∞	9.02	1.36	0.00	∞
$df = \infty, R^* = 0.3$		0.09	0.23	0.00	∞	0.03	0.01	0.00	∞	0.05	0.13	0.00	∞
$df = \infty, R^* = 0.5$		0.25	0.57	0.80	346.00	0.13	0.33	0.86	345.00	0.17	0.42	0.84	352.50
$df = \infty, R^* = 0.7$		0.90	0.89	0.36	∞	0.77	0.92	0.46	∞	2.66	1.39	0.00	∞
DI-PSO													
$\alpha = 50, \beta = 1$		32.08	11.51	0.00	∞	13.04	5.65	0.00	∞	6.33	3.37	0.00	∞
$\alpha = 50, \beta = 2$		38.18	17.28	0.00	∞	15.90	5.67	0.00	∞	10.13	4.19	0.00	∞
$\alpha = 50, \beta = 4$		56.49	22.89	0.00	∞	28.28	11.13	0.00	∞	15.83	7.08	0.00	∞
$\alpha = 100, \beta = 1$		28.51	12.71	0.00	∞	11.32	5.72	0.00	∞	4.66	2.48	0.00	∞
$\alpha = 100, \beta = 2$		32.85	13.96	0.00	∞	14.35	7.57	0.00	∞	6.86	2.78	0.00	∞
$\alpha = 100, \beta = 4$		49.67	16.91	0.00	∞	24.47	9.82	0.00	∞	13.74	6.18	0.00	∞
$\alpha = 200, \beta = 1$		22.39	12.14	0.00	∞	7.53	4.31	0.00	∞	3.92	2.11	0.02	∞
$\alpha = 200, \beta = 2$		36.56	14.29	0.00	∞	11.21	5.42	0.00	∞	5.28	2.77	0.00	∞
$\alpha = 200, \beta = 4$		55.67	19.91	0.00	∞	19.67	8.61	0.00	∞	11.18	5.41	0.00	∞
AT-PSO													
$R^* = 0.1$		7.73	5.50	0.00	∞	5.72	1.90	0.00	∞	23.02	10.87	0.00	∞
$R^* = 0.3$		14.10	6.74	0.00	∞	6.67	3.74	0.00	∞	3.77	2.04	0.02	∞
$R^* = 0.5$		23.89	8.43	0.00	∞	11.39	5.91	0.00	∞	6.06	2.92	0.00	∞
$R^* = 0.7$		34.91	12.20	0.00	∞	14.91	5.84	0.00	∞	9.80	3.78	0.00	∞

Table 5: Simulation results for OF4. See text for description.

OF5		Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm		Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO		27.55	20.12	0.00	∞	0.03	0.03	0.28	∞	0.08	0.07	0.04	∞
BBPSO-MC		762.36	82.36	0.00	∞	785.33	69.84	0.00	∞	751.88	80.37	0.00	∞
BBPSOxp-MC		766.30	80.16	0.00	∞	785.43	64.92	0.00	∞	771.17	76.11	0.00	∞
AT-BBPSO-MC													
$df = 1, R^* = 0.1$		0.83	0.10	0.00	∞	0.88	0.09	0.00	∞	0.92	0.06	0.00	∞
$df = 1, R^* = 0.3$		0.02	0.01	0.24	∞	0.02	0.01	0.34	∞	0.05	0.03	0.02	∞
$df = 1, R^* = 0.5$		0.01	0.01	0.52	455.50	0.01	0.01	0.56	455.00	0.01	0.01	0.48	∞
$df = 1, R^* = 0.7$		0.02	0.01	0.36	∞	0.02	0.02	0.46	∞	0.04	0.04	0.16	∞
$df = 3, R^* = 0.1$		1.06	0.02	0.00	∞	1.05	0.02	0.00	∞	1.04	0.02	0.00	∞
$df = 3, R^* = 0.3$		0.08	0.03	0.00	∞	0.12	0.04	0.00	∞	0.32	0.15	0.00	∞
$df = 3, R^* = 0.5$		0.01	0.01	0.54	459.00	0.01	0.01	0.62	481.00	0.09	0.07	0.02	∞
$df = 3, R^* = 0.7$		0.01	0.01	0.46	∞	0.12	0.15	0.04	∞	205.76	57.74	0.00	∞
$df = 5, R^* = 0.1$		1.11	0.03	0.00	∞	1.11	0.03	0.00	∞	1.09	0.03	0.00	∞
$df = 5, R^* = 0.3$		0.18	0.05	0.00	∞	0.29	0.09	0.00	∞	0.66	0.15	0.00	∞
$df = 5, R^* = 0.5$		0.02	0.01	0.32	∞	0.02	0.01	0.26	∞	0.69	0.21	0.00	∞
$df = 5, R^* = 0.7$		0.01	0.01	0.32	∞	39.11	26.34	0.00	∞	480.85	48.89	0.00	∞
$df = \infty, R^* = 0.1$		1.47	0.11	0.00	∞	1.45	0.09	0.00	∞	1.25	0.08	0.00	∞
$df = \infty, R^* = 0.3$		0.69	0.11	0.00	∞	0.85	0.07	0.00	∞	0.97	0.05	0.00	∞
$df = \infty, R^* = 0.5$		0.04	0.02	0.00	∞	0.26	0.09	0.00	∞	9.61	10.88	0.00	∞
$df = \infty, R^* = 0.7$		0.82	0.19	0.00	∞	395.75	42.10	0.00	∞	638.33	59.56	0.00	∞
AT-BBPSOxp-MC													
$df = 1, R^* = 0.1$		1.01	0.02	0.00	∞	1.01	0.02	0.00	∞	1.00	0.03	0.00	∞
$df = 1, R^* = 0.3$		0.07	0.04	0.00	∞	0.13	0.06	0.00	∞	0.25	0.13	0.00	∞
$df = 1, R^* = 0.5$		0.01	0.01	0.54	465.50	0.01	0.01	0.44	∞	0.06	0.08	0.06	∞
$df = 1, R^* = 0.7$		0.02	0.02	0.46	∞	0.02	0.03	0.46	∞	1.20	0.54	0.00	∞
$df = 3, R^* = 0.1$		1.17	0.04	0.00	∞	1.16	0.04	0.00	∞	1.09	0.03	0.00	∞
$df = 3, R^* = 0.3$		0.56	0.12	0.00	∞	0.73	0.08	0.00	∞	0.87	0.09	0.00	∞
$df = 3, R^* = 0.5$		0.04	0.02	0.00	∞	0.33	0.14	0.00	∞	5.84	7.01	0.00	∞
$df = 3, R^* = 0.7$		7.20	9.68	0.00	∞	295.56	44.31	0.00	∞	439.15	53.40	0.00	∞
$df = 5, R^* = 0.1$		1.26	0.06	0.00	∞	1.30	0.06	0.00	∞	1.16	0.06	0.00	∞
$df = 5, R^* = 0.3$		0.80	0.08	0.00	∞	0.92	0.05	0.00	∞	1.00	0.02	0.00	∞
$df = 5, R^* = 0.5$		0.22	0.10	0.00	∞	0.93	0.09	0.00	∞	72.53	27.81	0.00	∞
$df = 5, R^* = 0.7$		172.86	53.14	0.00	∞	491.44	39.13	0.00	∞	527.93	56.57	0.00	∞
$df = \infty, R^* = 0.1$		1.71	0.13	0.00	∞	1.70	0.17	0.00	∞	1.34	0.14	0.00	∞
$df = \infty, R^* = 0.3$		0.99	0.04	0.00	∞	1.03	0.01	0.00	∞	1.06	0.03	0.00	∞
$df = \infty, R^* = 0.5$		0.82	0.12	0.00	∞	2.65	1.75	0.00	∞	255.87	50.83	0.00	∞
$df = \infty, R^* = 0.7$		451.87	38.28	0.00	∞	622.22	35.60	0.00	∞	586.09	47.80	0.00	∞
DI-PSO													
$\alpha = 50, \beta = 1$		55.37	25.87	0.00	∞	7.99	9.84	0.00	∞	1.07	0.35	0.00	∞
$\alpha = 50, \beta = 2$		73.04	43.14	0.00	∞	21.12	16.40	0.00	∞	6.20	4.72	0.00	∞
$\alpha = 50, \beta = 4$		142.00	66.28	0.00	∞	61.48	36.04	0.00	∞	27.71	18.29	0.00	∞
$\alpha = 100, \beta = 1$		32.00	20.76	0.00	∞	2.52	2.93	0.00	∞	0.22	0.19	0.00	∞
$\alpha = 100, \beta = 2$		78.45	51.91	0.00	∞	12.10	18.31	0.00	∞	1.76	1.50	0.00	∞
$\alpha = 100, \beta = 4$		139.13	62.71	0.00	∞	42.32	21.23	0.00	∞	15.40	12.26	0.00	∞
$\alpha = 200, \beta = 1$		28.74	20.48	0.00	∞	0.56	0.70	0.00	∞	0.06	0.07	0.18	∞
$\alpha = 200, \beta = 2$		76.70	56.65	0.00	∞	4.61	6.06	0.00	∞	0.52	0.34	0.00	∞
$\alpha = 200, \beta = 4$		130.62	67.27	0.00	∞	21.48	17.97	0.00	∞	5.24	5.82	0.00	∞
AT-PSO													
$R^* = 0.1$		3.28	6.03	0.00	∞	2.22	2.69	0.00	∞	111.55	65.99	0.00	∞
$R^* = 0.3$		0.74	1.00	0.02	∞	0.02	0.02	0.34	∞	0.06	0.06	0.16	∞
$R^* = 0.5$		4.06	8.44	0.00	∞	0.07	0.13	0.22	∞	0.01	0.02	0.56	414.50
$R^* = 0.7$		28.82	33.05	0.00	∞	0.43	1.25	0.04	∞	0.07	0.16	0.22	∞

Table 6: Simulation results for OF5. See text for description.

OF6	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}	Mean	SD	\hat{p}	\hat{t}
PSO	19.66	0.56	0.00	∞	14.77	8.35	0.02	∞	19.48	2.63	0.00	∞
BBPSO-MC	20.72	0.23	0.00	∞	20.74	0.22	0.00	∞	20.74	0.21	0.00	∞
BBPSOxp-MC	20.75	0.18	0.00	∞	20.73	0.18	0.00	∞	20.73	0.18	0.00	∞
AT-BBPSO-MC												
$df = 1, R^* = 0.1$	18.35	5.39	0.00	∞	9.73	9.19	0.00	∞	6.24	7.88	0.00	∞
$df = 1, R^* = 0.3$	18.65	4.66	0.00	∞	14.66	8.70	0.00	∞	11.40	9.71	0.00	∞
$df = 1, R^* = 0.5$	19.93	0.37	0.00	∞	18.90	4.71	0.04	∞	16.91	6.85	0.06	∞
$df = 1, R^* = 0.7$	19.69	2.00	0.00	∞	19.52	2.28	0.00	∞	18.05	5.13	0.00	∞
$df = 3, R^* = 0.1$	19.98	0.09	0.00	∞	19.48	3.55	0.00	∞	14.60	8.47	0.00	∞
$df = 3, R^* = 0.3$	19.82	0.05	0.00	∞	19.48	2.82	0.00	∞	19.94	0.54	0.00	∞
$df = 3, R^* = 0.5$	19.83	0.06	0.00	∞	19.95	0.30	0.00	∞	20.24	0.38	0.00	∞
$df = 3, R^* = 0.7$	19.85	0.10	0.00	∞	20.05	0.37	0.00	∞	20.34	0.39	0.00	∞
$df = 5, R^* = 0.1$	20.00	0.07	0.00	∞	20.12	0.25	0.00	∞	17.83	6.21	0.00	∞
$df = 5, R^* = 0.3$	19.83	0.04	0.00	∞	19.87	0.21	0.00	∞	20.12	0.45	0.00	∞
$df = 5, R^* = 0.5$	19.83	0.04	0.00	∞	19.90	0.26	0.00	∞	20.32	0.45	0.00	∞
$df = 5, R^* = 0.7$	19.85	0.12	0.00	∞	20.06	0.35	0.00	∞	20.42	0.39	0.00	∞
$df = \infty, R^* = 0.1$	20.09	0.08	0.00	∞	20.16	0.14	0.00	∞	20.32	0.27	0.00	∞
$df = \infty, R^* = 0.3$	19.83	0.03	0.00	∞	19.85	0.17	0.00	∞	19.98	0.33	0.00	∞
$df = \infty, R^* = 0.5$	19.83	0.03	0.00	∞	19.89	0.20	0.00	∞	20.14	0.38	0.00	∞
$df = \infty, R^* = 0.7$	19.83	0.06	0.00	∞	20.06	0.36	0.00	∞	20.36	0.37	0.00	∞
AT-BBPSOxp-MC												
$df = 1, R^* = 0.1$	15.26	7.90	0.00	∞	11.33	8.97	0.00	∞	9.05	8.54	0.00	∞
$df = 1, R^* = 0.3$	18.42	5.50	0.00	∞	17.06	6.87	0.00	∞	14.76	8.83	0.00	∞
$df = 1, R^* = 0.5$	20.19	0.20	0.00	∞	19.48	3.44	0.02	∞	17.05	6.20	0.00	∞
$df = 1, R^* = 0.7$	20.22	0.26	0.00	∞	20.08	0.67	0.00	∞	19.74	2.08	0.00	∞
$df = 3, R^* = 0.1$	20.19	0.16	0.00	∞	20.26	0.13	0.00	∞	19.58	3.36	0.00	∞
$df = 3, R^* = 0.3$	20.08	0.17	0.00	∞	20.23	0.18	0.00	∞	20.27	0.18	0.00	∞
$df = 3, R^* = 0.5$	20.19	0.13	0.00	∞	20.27	0.13	0.00	∞	20.33	0.25	0.00	∞
$df = 3, R^* = 0.7$	20.21	0.13	0.00	∞	20.32	0.11	0.00	∞	20.39	0.11	0.00	∞
$df = 5, R^* = 0.1$	20.25	0.08	0.00	∞	20.27	0.11	0.00	∞	20.32	0.12	0.00	∞
$df = 5, R^* = 0.3$	20.07	0.16	0.00	∞	20.20	0.17	0.00	∞	20.21	0.18	0.00	∞
$df = 5, R^* = 0.5$	20.14	0.15	0.00	∞	20.23	0.12	0.00	∞	20.33	0.14	0.00	∞
$df = 5, R^* = 0.7$	20.20	0.11	0.00	∞	20.31	0.13	0.00	∞	20.37	0.11	0.00	∞
$df = \infty, R^* = 0.1$	20.18	0.08	0.00	∞	20.27	0.08	0.00	∞	20.29	0.12	0.00	∞
$df = \infty, R^* = 0.3$	20.01	0.17	0.00	∞	20.14	0.18	0.00	∞	20.20	0.19	0.00	∞
$df = \infty, R^* = 0.5$	20.06	0.15	0.00	∞	20.22	0.14	0.00	∞	20.27	0.15	0.00	∞
$df = \infty, R^* = 0.7$	20.14	0.15	0.00	∞	20.30	0.13	0.00	∞	20.36	0.14	0.00	∞
DI-PSO												
$\alpha = 50, \beta = 1$	19.42	1.48	0.00	∞	18.52	3.08	0.00	∞	19.02	3.09	0.00	∞
$\alpha = 50, \beta = 2$	19.50	1.04	0.00	∞	17.94	3.01	0.00	∞	19.33	1.89	0.00	∞
$\alpha = 50, \beta = 4$	19.83	0.64	0.00	∞	19.66	1.00	0.00	∞	19.76	1.07	0.00	∞
$\alpha = 100, \beta = 1$	18.71	2.25	0.00	∞	16.04	5.48	0.00	∞	17.63	5.10	0.00	∞
$\alpha = 100, \beta = 2$	19.75	1.02	0.00	∞	19.27	1.95	0.00	∞	18.96	3.30	0.00	∞
$\alpha = 100, \beta = 4$	19.99	1.04	0.00	∞	19.80	1.11	0.00	∞	20.13	0.75	0.00	∞
$\alpha = 200, \beta = 1$	19.13	2.10	0.00	∞	14.83	6.47	0.00	∞	16.77	5.92	0.00	∞
$\alpha = 200, \beta = 2$	20.00	1.29	0.00	∞	19.63	2.37	0.00	∞	20.27	0.32	0.00	∞
$\alpha = 200, \beta = 4$	20.38	0.65	0.00	∞	20.19	0.65	0.00	∞	20.47	0.31	0.00	∞
AT-PSO												
$R^* = 0.1$	18.67	4.43	0.00	∞	18.18	5.06	0.00	∞	20.18	0.66	0.00	∞
$R^* = 0.3$	19.44	1.34	0.00	∞	16.84	6.30	0.04	∞	15.62	7.72	0.00	∞
$R^* = 0.5$	19.69	0.92	0.00	∞	18.88	3.16	0.00	∞	19.66	2.29	0.00	∞
$R^* = 0.7$	19.97	0.53	0.00	∞	19.76	0.69	0.00	∞	20.12	0.32	0.00	∞

Table 7: Simulation results for OF6. See text for description.

References

- Blum, C. and Li, X. (2008). “Swarm Intelligence in Optimization.” In *Swarm Intelligence: Introduction and Applications*, eds. C. Blum and D. Merkle. Springer.
- Clerc, M. (2010). *Particle swarm optimization*. John Wiley & Sons.
- Clerc, M. and Kennedy, J. (2002). “The particle swarm-explosion, stability, and convergence in a multidimensional complex space.” *Evolutionary Computation, IEEE Transactions on*, 6, 1, 58–73.
- Hsieh, H.-I. and Lee, T.-S. (2010). “A modified algorithm of bare bones particle swarm optimization.” *International Journal of Computer Science Issues*, 7, 11.
- Kennedy, J. (2003). “Bare bones particle swarms.” In *Swarm Intelligence Symposium, 2003. SIS’03. Proceedings of the 2003 IEEE*, 80–87. IEEE.
- Zhang, H., Kennedy, D. D., Rangaiah, G. P., and Bonilla-Petriciolet, A. (2011). “Novel bare-bones particle swarm optimization and its performance for modeling vapor–liquid equilibrium data.” *Fluid Phase Equilibria*, 301, 1, 33–45.