Particle Swarm Optimization Assisted Metropolis Hastings Algorithms

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

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

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

$$\theta_{ij}(t+1) = p_{i_1 j}(t) + 0.5\{p_{i_2 j}(t) - p_{i_3 j}(t)\}. \tag{A.2}$$

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

$$\theta_{ij}(t+1) = \begin{cases} \begin{cases} N\left(\frac{p_{ij}(t)+g_{ij}(t)}{2}, s_{ij}^2(t)\right) & \text{with probability } \rho \\ p_{ij}(t) & \text{with probability } 1-\rho, \end{cases} & \text{if } s_{ij}^2(t) > 0 \\ p_{i_1 j}(t) + 0.5\{p_{i_2 j}(t) - p_{i_3 j}(t)\} & \text{otherwise.} \end{cases} \tag{A.3}$$

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):

$$\theta_{ij}(t+1) = \begin{cases} \begin{cases} t_{df}\left(\frac{p_{ij}(t)+g_{ij}(t)}{2}, \sigma^2(t) s_{ij}^2(t)\right) & \text{with probability } \rho \\ p_{ij}(t) & \text{with probability } 1-\rho, \end{cases} & \text{if } s_{ij}^2(t) > 0 \\ p_{i_1 j}(t) + 0.5\{p_{i_2 j}(t) - p_{i_3 j}(t)\} & \text{otherwise.} \end{cases}$$

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

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

A.2 Ring-k neighborhood topologies

For all PSO and BBPSO algorithms, the group best location for a given particle is defined relative to the particle's group. To wit $\boldsymbol{g}_i(t+1) = \arg\max_{\{\boldsymbol{p}_j(t+1) | j \in \mathcal{N}_i\}} Q(\boldsymbol{p}_j(t+1))$ where $\mathcal{N}_i$ is the set of particles that part of particle i's group, or are "neighbors" of particle i. The

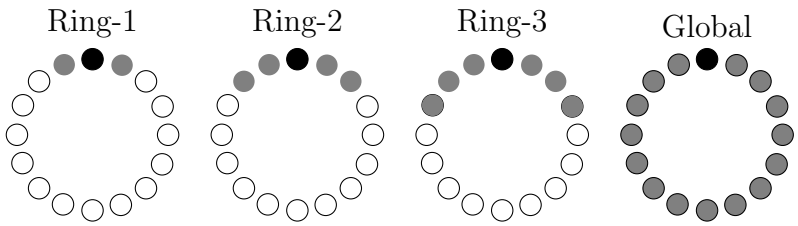

Figure A.1: Ring-k and global neighborhood topologies. Filled in gray particles are neighbors of the filled in black particle.

default neighborhood topology is $\mathcal{N}_i = \{1, 2, \ldots, n\}$ where n is the number of particles, so each particle is in each other particle's neighborhood. PSO and BBPSO algorithms using this topology are often call gbest swarms or gbest algorithms.

The main problem with gbest PSO and BBPSO algorithms is that by allowing each particle to see the best location of the entire swarm, they curtail early exploration of the search space. In order to restrict the flow of information across the swarm and allow for more exploration, it is often useful to only allow each particle to see a subset of the other particles using a more restrictive neighborhood topology. There are several commonly used neighborhood topologies [CITATION], but we will use the ring topologies. The ring-k neighborhood topology for $k = 1, 2, \ldots, \lceil n/2 \rceil$ is defined as follows. Arrange the particles in a circle, as in Figure A.1. Then each particle is neighbors with the k particles to its left and the k particles to its right. That is, $\mathcal{N}_i = \{i-k, i-k+1, \ldots, i-1, i+1, \ldots, i+k-1, i+k\}$ where addition and subtraction are modulo n so that e.g. $n+1 = 1$. The value of k controls how quickly information moves around the swarm — the larger k, the faster it moves. In the limit of $k = \lceil n/2 \rceil$, we have the global neighborhood topology.

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 $\Re^{20}$. 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 $||.||$ 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 $\boldsymbol{x}_i$ for particle i. Let $x_{max,j}$ be the maximum initial value of coordinate j of $\boldsymbol{x}_i$ 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 $v_{ij}(0) \overset{iid}{\sim} U(-d_{max}/2, d_{max}/2)$. 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 $\widehat{p}$. Finally, $\widehat{t}$ 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 in 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. Second, for most non-AT algorithms the more restrictive neighborhood appears to be result in 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. 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.2 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 0.5 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.2, 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.

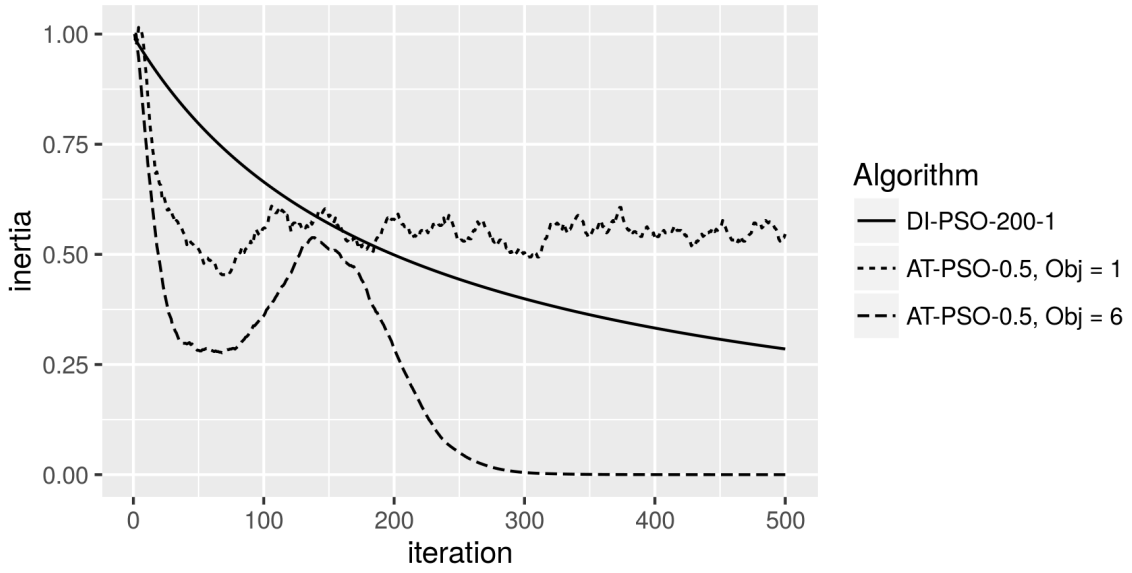

Figure B.2: Inertia over time for the DI-PSO algorithm with $\alpha = 200$ and $\beta = 1$, and for one replication of the AT-PSO-0.5 algorithm for each of OFs 1 and 6.

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.

C County Population Model Details

We consider two possible data models, Poisson and lognormal, and two possible random effect distributions, iid normal and normal with a fully correlated covariance matrix. The posterior distributions for the models with iid random effects can be written as

$$\begin{aligned}
p(\boldsymbol{\beta},\sigma^2,\boldsymbol{\delta},\phi^2|\boldsymbol{z},\boldsymbol{X},\boldsymbol{S}) &\propto (\phi^2)^{-n/2-a_\phi-1}\exp\left[-\frac{1}{\phi^2}\left\{\frac{(\log\boldsymbol{z}-\boldsymbol{y})'(\log\boldsymbol{z}-\boldsymbol{y})}{2}+b_\phi\right\}\right] \\
&\quad\times(\sigma^2)^{-\frac{r}{2}-a_\sigma-1}\exp\left\{-\frac{1}{\sigma^2}\left(\frac{\boldsymbol{\delta}'\boldsymbol{\delta}}{2}+b_\sigma\right)\right\}\exp\left\{-\frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{2v}\right\} \qquad \text{(iid lognormal)}, \\
p(\boldsymbol{\beta},\sigma^2,\boldsymbol{\delta}|\boldsymbol{z},\boldsymbol{X},\boldsymbol{S}) &\propto \prod_{i=1}^{n}\frac{\exp\{-\exp(y_i)\}\exp(y_i z_i)}{z_i!} \\
&\quad\times(\sigma^2)^{-\frac{r}{2}-a_\sigma-1}\exp\left\{-\frac{1}{\sigma^2}\left(\frac{\boldsymbol{\delta}'\boldsymbol{\delta}}{2}+b_\sigma\right)\right\}\exp\left\{-\frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{2v}\right\} \qquad \text{(iid Poisson)}.
\end{aligned}$$

For fully parameterized $\boldsymbol{\Sigma}$ we write the posteriors in terms of the precision matrix $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$, yielding

$$\begin{aligned}
p(\boldsymbol{\beta},\boldsymbol{\Omega},\boldsymbol{\delta},\phi^2|\boldsymbol{z},\boldsymbol{X},\boldsymbol{S}) &\propto (\phi^2)^{-n/2-a_\phi-1}\exp\left\{-\frac{1}{\phi^2}\left(\frac{(\log\boldsymbol{z}-\boldsymbol{y})'(\log\boldsymbol{z}-\boldsymbol{y})}{2}+b_\phi\right)\right\} \\
&\quad\times|\boldsymbol{\Omega}|^{(d-r)/2}\exp\left\{-\frac{1}{2}\left(\boldsymbol{\delta}'\boldsymbol{\Omega}\boldsymbol{\delta}+\frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{v}+\mathrm{tr}(\boldsymbol{E}\boldsymbol{\Omega})\right)\right\} \qquad \text{(full lognormal)}, \\
p(\boldsymbol{\beta},\boldsymbol{\Omega},\boldsymbol{\delta}|\boldsymbol{z},\boldsymbol{X},\boldsymbol{S}) &\propto \prod_{i=1}^{n}\frac{\exp\{-\exp(y_i)\}\exp(y_i z_i)}{z_i!} \\
&\quad\times|\boldsymbol{\Omega}|^{(d-r)/2}\exp\left\{-\frac{1}{2}\left(\boldsymbol{\delta}'\boldsymbol{\Omega}\boldsymbol{\delta}+\frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{v}+\mathrm{tr}(\boldsymbol{E}\boldsymbol{\Omega})\right)\right\} \qquad \text{(full Poisson)}
\end{aligned}$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator. For the PSO algorithm and some MCMC algorithms it will be easier to work with the Cholesky decomposition of $\boldsymbol{\Omega}$ given by $\boldsymbol{L}\boldsymbol{L}' = \boldsymbol{\Omega}$ where $\boldsymbol{L}$ is

lower triangular. It is often more convenient to put the prior distribution directly on $\boldsymbol{L}$ rather than on $\boldsymbol{\Omega}$ or $\boldsymbol{\Omega}^{-1}$ and solving for the Jacobian, but this depends in part on which MCMC algorithm is used to fit the model. So while we use the prior distribution on $\boldsymbol{L}$ implied by a Wishart prior on $\boldsymbol{\Omega}$, in practice it is advantageous to use one of the priors suggested by Chen and Dunson (2003) or Frühwirth-Schnatter and Tüchler (2008). We allow the diagonal entries of $\boldsymbol{L}$ to be negative in the PSO and IMH algorithms, so $\boldsymbol{L}$ is not strictly speaking a Cholesky decomposition. The determinant of the Jacobian is the same in both cases up to a proportionality constant. The signs of the elements of $\boldsymbol{L}$ are not identified, therefore care needs to be taken when interpreting the results of MCMC, but transforming back to the precision matrix in a post processing step is sufficient. Let ℓ_{ij} denote the (i,j)th element of $\boldsymbol{L}$. Then the Jacobian of $\boldsymbol{\Omega} \to \boldsymbol{L}$ is given by

$$|J(\boldsymbol{\Omega} \to \boldsymbol{L})| \propto \prod_{k=1}^{r} |\ell_{kk}|^{r+1-k}$$

where $\boldsymbol{\Omega}$ is $r \times r$. Under this parameterization the full posteriors can be written as

$$\begin{aligned}
p(\boldsymbol{\beta}, \boldsymbol{L}, \boldsymbol{\delta}, \phi^2 | \boldsymbol{z}, \boldsymbol{X}, \boldsymbol{S}) \propto{}& (\phi^2)^{-n/2-a_\phi-1} \exp\left[-\frac{1}{\phi^2}\left\{\frac{(\log \boldsymbol{z} - \boldsymbol{y})'(\log \boldsymbol{z} - \boldsymbol{y})}{2} + b_\phi\right\}\right] \\
&\times \prod_{k=1}^{r} (\ell_{kk}^2)^{(d-k+1)/2} \\
&\times \exp\left[-\frac{1}{2}\left\{\boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}'\boldsymbol{\delta} + \frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{v} + \mathrm{tr}(\boldsymbol{E}\boldsymbol{L}\boldsymbol{L}')\right\}\right] \qquad \text{(full lognormal)}, \\
p(\boldsymbol{\beta}, \boldsymbol{L}, \boldsymbol{\delta} | \boldsymbol{z}, \boldsymbol{X}, \boldsymbol{S}) \propto{}& \prod_{i=1}^{n} \frac{\exp\{-\exp(y_i)\}\exp(y_i z_i)}{z_i!} \times \prod_{k=1}^{r} (\ell_{kk}^2)^{(d-k+1)/2} \\
&\times \exp\left[-\frac{1}{2}\left\{\boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}'\boldsymbol{\delta} + \frac{(\boldsymbol{\beta}-\boldsymbol{b})'(\boldsymbol{\beta}-\boldsymbol{b})}{v} + \mathrm{tr}(\boldsymbol{E}\boldsymbol{L}\boldsymbol{L}')\right\}\right] \qquad \text{(full Poisson)}.
\end{aligned}$$

In Appendix C.2 we derive the Hessian for the fully parameterized Poisson model. The other models are analogous, though the variances in the iid models and in the lognormal models should be transformed to the log scale first.

C.1 Full conditionals for MCMC

We consider sevaeral alternative Gibbs sampling algorithms, detailed in Appendix D, which draw the covariance matrix from its full conditional distribution. When the covariance matrix is fully parameterized the full conditional distribution of the precision matrix $\boldsymbol{\Omega}$ is Wishart, that is $\boldsymbol{\Omega} \sim W(\widetilde{d}, \widetilde{\boldsymbol{E}})$. This draw is usually accomplished via the Bartlett decomposition (Smith and Hocking, 1972). Let $\widetilde{\boldsymbol{C}}$ be the lower triangular Cholesky decomposition of $\widetilde{\boldsymbol{E}}$. Then let $\boldsymbol{A}$ be an $r \times r$ random lower triangular matrix with independent elements $\{a_{ij} : 0 < i \leq j \leq r\}$ where $a_{ii} \sim \sqrt{\chi^2_{\widetilde{d}-i+1}}$ for $i = 1, 2, \ldots, r$ and $a_{ij} \sim N(0,1)$ for $0 < i \leq j \leq r$. Then $\boldsymbol{\Omega} = \boldsymbol{L}\boldsymbol{L}' \sim W(\widetilde{d}, \widetilde{\boldsymbol{E}})$ where $\boldsymbol{L} = \widetilde{\boldsymbol{C}}\boldsymbol{A}$. In the process of drawing $\boldsymbol{\Omega}$ we must first draw $\boldsymbol{L}$, so we construct our Gibbs samplers in terms of $\boldsymbol{L}$ instead of $\boldsymbol{\Omega}$.

IMH applies straightforwardly to each of these models, though see Appendix C.2 for a detailed derivation of the Hessian for the county population models with a fully parameterized covariance matrix associated. To apply IMHwG in the county population models, we draw $(\boldsymbol{\beta}, \boldsymbol{\delta})$ in the Metropolis step and σ^2, $\boldsymbol{L}$, and/or ϕ^2 in the conditionally conjugate step, depending on the model. The full conditional distribution of σ^2 in both iid models is $IG(a_\sigma + r/2, b_\sigma + \boldsymbol{\delta}'\boldsymbol{\delta}/2)$. The full conditional distribution of $\boldsymbol{\Omega}$ in both fully correlated models is $W\{r+1, (\boldsymbol{E} + \boldsymbol{\delta}\boldsymbol{\delta}')^{-1}\}$. Then a draw from the full conditional distribution of $\boldsymbol{L}$ can be obtained via the Bartlett decomposition as described in the previous paragraph. In the lognormal models the full conditional distribution of ϕ^2 is $IG\{a_\phi + n/2, b_\phi + (\log \boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{S}\boldsymbol{\delta})'(\log \boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{S}\boldsymbol{\delta})/2\}$.

C.2 Deriving the Hessian

The log posterior in the fully parameterized Poisson case can be written as

$$\begin{aligned}\log p(\boldsymbol{\beta}, \boldsymbol{L}, \boldsymbol{\delta} | \boldsymbol{z}, \boldsymbol{X}, \boldsymbol{S}) &= \text{constant} + \sum_{i=1}^{n} (y_i z_i - \exp(y_i)) + \sum_{k=1}^{r} \frac{d-k+1}{2} \log \ell_{kk}^2 \\ &\quad - \frac{1}{2} \left\{ \boldsymbol{\delta}' \boldsymbol{L} \boldsymbol{L}' \boldsymbol{\delta} + \frac{(\boldsymbol{\beta} - \boldsymbol{b})'(\boldsymbol{\beta} - \boldsymbol{b})}{v^2} + \operatorname{tr}(\boldsymbol{E} \boldsymbol{L} \boldsymbol{L}') \right\} \\ &= \text{constant} + \sum_{i=1}^{n} \{ z_i y_i - \exp(y_i) \} + \frac{1}{2} \boldsymbol{R}_r \operatorname{vech}(\log \boldsymbol{L}^2) \\ &\quad - \frac{1}{2} \left[\operatorname{vech}(\boldsymbol{L})' \boldsymbol{M}_r \left\{ \boldsymbol{K}_r' (\boldsymbol{\delta}\boldsymbol{\delta}' \otimes \boldsymbol{I}_r) \boldsymbol{K}_r + (\boldsymbol{I}_r \otimes \boldsymbol{E}) \right\} \boldsymbol{M}_r' \operatorname{vech}(\boldsymbol{L}) + \frac{(\boldsymbol{\beta} - \boldsymbol{b})'(\boldsymbol{\beta} - \boldsymbol{b})}{v^2} \right]\end{aligned}$$

where $y_i = \boldsymbol{x}_i' \boldsymbol{\beta} + \boldsymbol{s}_i' \boldsymbol{\delta}$, $\otimes$ is the Kronecker product, $\boldsymbol{I}_r$ is the $r \times r$ identity matrix, and $\boldsymbol{R}_r$ is an $r(r+1)/2 \times r(r+1)/2$ matrix of zeroes except the $(r+1)k - k(k+1)/2 + k - r$st diagonal element is equal to $d - k + 1$ for $k = 1, 2, \ldots, r$, corresponding to locations in $\operatorname{vech}(\boldsymbol{L})$ that store the diagonal elements of $\boldsymbol{L}$. In $\operatorname{vech}(\log \boldsymbol{L}^2)$ the log and power are applied element-wise. The expansions of $\boldsymbol{\delta}' \boldsymbol{L} \boldsymbol{L}' \boldsymbol{\delta}$ and $\operatorname{tr}(\boldsymbol{E} \boldsymbol{L} \boldsymbol{L}')$ can be derived from the properties of the trace operator, Kronecker products, and the following identities.

$$\begin{aligned}\boldsymbol{\delta}' \boldsymbol{L} \boldsymbol{L}' \boldsymbol{\delta} &= \operatorname{vec}(\boldsymbol{L}' \boldsymbol{\delta})' \operatorname{vec}(\boldsymbol{L}' \boldsymbol{\delta}), \\ \operatorname{vec}(\boldsymbol{L}' \boldsymbol{\delta}) &= (\boldsymbol{\delta}' \otimes \boldsymbol{I}_r) \operatorname{vec}(\boldsymbol{L}'), \\ \operatorname{vec}(\boldsymbol{L}') &= \boldsymbol{K}_r \operatorname{vec}(\boldsymbol{L}) = \boldsymbol{K}_r \boldsymbol{M}_r' \operatorname{vech}(\boldsymbol{L}),\end{aligned}$$

and assuming $\boldsymbol{E} = \boldsymbol{C}\boldsymbol{C}'$, then $\operatorname{tr}(\boldsymbol{L}'\boldsymbol{C}\boldsymbol{C}'\boldsymbol{L}) = \operatorname{vec}(\boldsymbol{C}'\boldsymbol{L})' \operatorname{vec}(\boldsymbol{C}'\boldsymbol{L})$ where $\operatorname{vec}(\cdot)$ is the vectorization operation which stacks each column of its argument on top of each other into a single column vector, $\operatorname{vech}(\cdot)$ is the half-vectorization operator which is similar but omits elements above the diagonal, tr is the trace operator, $\boldsymbol{K}_r$ is the $r^2 \times r^2$ commutation matrix such that for any $r \times r$ matrix $\boldsymbol{L}$, $\operatorname{vec}(\boldsymbol{L}') = \boldsymbol{K}_r \operatorname{vec}(\boldsymbol{L})$, $\boldsymbol{M}_r$ is the $r^2 \times r(r+1)/2$ elimination matrix such that for $\operatorname{vech}(\boldsymbol{L}) = \boldsymbol{M}_r \operatorname{vec}(\boldsymbol{L})$ and if additionally $\boldsymbol{L}$ is lower triangular, $\operatorname{vec}(\boldsymbol{L}) = \boldsymbol{M}_r' \operatorname{vech}(\boldsymbol{L})$ (Magnus and Neudecker, 1980; Magnus, 1988). Let $\boldsymbol{E}_{ij}$ be an $r \times r$

matrix of zeroes with a one in only its (i,j)th position, $\boldsymbol{u}_{ij} = \text{vech}(\boldsymbol{E}_{ij})$, and $\boldsymbol{e}_i$ be a r-dimensional column vector of zeroes with a one in the ith row. Then $\boldsymbol{K}_r$ and $\boldsymbol{M}_r$ can be written as

$$\boldsymbol{K}_r = \sum_{i=1}^{r}\sum_{j=1}^{r} \boldsymbol{E}_{ij} \otimes \boldsymbol{E}'_{ij} \qquad \text{and} \qquad \boldsymbol{M}_r = \sum_{i\geq j}^{r} \boldsymbol{u}_{ij} \otimes \boldsymbol{e}'_j \otimes \boldsymbol{e}'_i.$$

Now the first derivatives of the log posterior are given by

$$\begin{aligned}
\frac{\partial \log p}{\partial \boldsymbol{\beta}} &= \sum_{i=1}^{n}(z_i - e^{y_i})\boldsymbol{x}'_i - \frac{(\boldsymbol{\beta} - \boldsymbol{b})'}{v^2}, \\
\frac{\partial \log p}{\partial \boldsymbol{\delta}} &= \sum_{i=1}^{n}(z_i - e^{y_i})\boldsymbol{s}'_i - \boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}', \\
\frac{\partial \log p}{\partial \,\text{vech}(\boldsymbol{L})} &= -\,\text{vech}(\boldsymbol{L})'\boldsymbol{M}_r\left[\boldsymbol{K}'_r(\boldsymbol{\delta}\boldsymbol{\delta}' \otimes \boldsymbol{I}_r)\boldsymbol{K}_r + (\boldsymbol{I}_r \otimes \boldsymbol{E})\right]\boldsymbol{M}'_r + \boldsymbol{R}_r\,\text{vech}(1/\boldsymbol{L}),
\end{aligned}$$

where in $\text{vech}(1/\boldsymbol{L})$ the division is applied element-wise. Next the second derivatives are

$$\begin{aligned}
\frac{\partial^2 \log p}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}'} &= -\sum_{i=1}^{n} e^{y_i}\boldsymbol{x}_i\boldsymbol{x}'_i - \frac{1}{v^2}\boldsymbol{I}_p \\
\frac{\partial^2 \log p}{\partial \boldsymbol{\delta}\partial \boldsymbol{\delta}'} &= -\sum_{i=1}^{n} e^{y_i}\boldsymbol{s}_i\boldsymbol{s}'_i - \boldsymbol{L}\boldsymbol{L}' \\
\frac{\partial^2 \log p}{\partial\,\text{vech}(\boldsymbol{L})\partial\,\text{vech}(\boldsymbol{L})'} &= -\,\boldsymbol{M}_r\left[\boldsymbol{K}'_r(\boldsymbol{\delta}\boldsymbol{\delta}' \otimes \boldsymbol{I}_r)\boldsymbol{K}_r + (\boldsymbol{I}_r \otimes \boldsymbol{E})\right]\boldsymbol{M}'_r - \boldsymbol{R}_r\,\text{diag}\left(\text{vech}(\boldsymbol{L})^{-2}\right) \\
\frac{\partial^2 \log p}{\partial \boldsymbol{\beta}\partial \boldsymbol{\delta}'} &= -\sum_{i=1}^{n} e^{y_i}\boldsymbol{s}_i\boldsymbol{x}'_i \\
\frac{\partial^2 \log p}{\partial \boldsymbol{\beta}\partial\,\text{vech}(\boldsymbol{L})'} &= \boldsymbol{0}_{p\times r(r+1)/2} \\
\frac{\partial^2 \log p}{\partial \boldsymbol{\delta}\partial\,\text{vech}(\boldsymbol{L})'} &= -\,\frac{\partial \boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}'}{\partial\,\text{vech}(\boldsymbol{L})'} = -(\boldsymbol{\delta}' \otimes \boldsymbol{I}_r)(\boldsymbol{I}_{r^2} + \boldsymbol{K}_r)(\boldsymbol{L} \otimes \boldsymbol{I}_r)\boldsymbol{M}'_r,
\end{aligned}$$

where $\text{diag}(\text{vech}(\boldsymbol{L})^{-2})$ is a diagonal matrix with the elements of $\text{vech}(\boldsymbol{L})$ raised to the power -2 along the diagonal. The last second derivative matrix can be derived using repeated application of the chain rule and from the following facts for any $r \times r$ matrix $\boldsymbol{A}$ (Magnus

and Neudecker, 2005):

$$\frac{\partial \operatorname{vec}(\boldsymbol{A}\boldsymbol{A}')}{\partial \operatorname{vec}(\boldsymbol{A})} = (\boldsymbol{I}_{r^2} + \boldsymbol{K}_r)(\boldsymbol{A} \otimes \boldsymbol{I}_r)$$
$$\frac{\partial \boldsymbol{A}\boldsymbol{\delta}}{\partial \operatorname{vec}(\boldsymbol{A})} = \boldsymbol{\delta}' \otimes \boldsymbol{I}_r.$$

Then, using the chain rule for lower triangular $\boldsymbol{L}$ we have

$$\begin{aligned}\frac{\partial \boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}'}{\partial \operatorname{vech}(\boldsymbol{L})'} &= \frac{\partial \boldsymbol{\delta}'\boldsymbol{L}\boldsymbol{L}'}{\partial \boldsymbol{L}\boldsymbol{L}'\boldsymbol{\delta}} \frac{\partial \boldsymbol{L}\boldsymbol{L}'\boldsymbol{\delta}}{\partial \operatorname{vec}(\boldsymbol{L}\boldsymbol{L}')} \frac{\partial \operatorname{vec}(\boldsymbol{L}\boldsymbol{L}')}{\partial \operatorname{vec}(\boldsymbol{L})} \frac{\partial \operatorname{vec}(\boldsymbol{L})}{\partial \operatorname{vech}(\boldsymbol{L})} \frac{\partial \operatorname{vech}(\boldsymbol{L})}{\partial \operatorname{vech}(\boldsymbol{L})'} \\ &= \boldsymbol{I}_r \frac{\partial \boldsymbol{L}\boldsymbol{L}'\boldsymbol{\delta}}{\partial \operatorname{vec}(\boldsymbol{L}\boldsymbol{L}')} \frac{\partial \operatorname{vec}(\boldsymbol{L}\boldsymbol{L}')}{\partial \operatorname{vec}(\boldsymbol{L})} \boldsymbol{M}_r' \boldsymbol{I}_{r(r+1)/2} \\ &= (\boldsymbol{\delta}' \otimes \boldsymbol{I}_r)(\boldsymbol{I}_{r^2} + \boldsymbol{K}_r)(\boldsymbol{L} \otimes \boldsymbol{I}_r)\boldsymbol{M}_r'.\end{aligned}$$

Finally, the Hessian is

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 \log p}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}'} & \frac{\partial^2 \log p}{\partial \boldsymbol{\beta} \partial \boldsymbol{\delta}'} & \frac{\partial^2 \log p}{\partial \boldsymbol{\beta} \partial \operatorname{vech}(\boldsymbol{L})'} \\ \frac{\partial^2 \log p}{\partial \boldsymbol{\delta} \partial \boldsymbol{\beta}'} & \frac{\partial^2 \log p}{\partial \boldsymbol{\delta} \partial \boldsymbol{\delta}'} & \frac{\partial^2 \log p}{\partial \boldsymbol{\delta} \partial \operatorname{vech}(\boldsymbol{L})'} \\ \frac{\partial^2 \log p}{\partial \operatorname{vech}(\boldsymbol{L}) \partial \boldsymbol{\beta}'} & \frac{\partial^2 \log p}{\partial \operatorname{vech}(\boldsymbol{L}) \partial \boldsymbol{\delta}'} & \frac{\partial^2 \log p}{\partial \operatorname{vech}(\boldsymbol{L}) \partial \operatorname{vech}(\boldsymbol{L})'} \end{bmatrix}.$$

D Alternative MCMC algorithms

[CURRENTLY THIS SECTION ONLY HAS GENERIC VERSIONS OF THE ALGORITHMS. DETAILS FOR OUR CASES?] This section describes the alternative MCMC algorithms used for comparisons with PSO assisted Metropolis-Hastings algorithms. In particular, we use two types of adaptive random walk Metropolis within Gibbs algorithms. In the generic problem, suppose we wish to sample from the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$ using MCMC methods. Suppose that $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ and that the full conditional $p(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2, \boldsymbol{y})$ can be sampled from easily while the full conditional for $\boldsymbol{\theta}_2$ may be intractable. Then a single move random walk Metropolis within Gibbs (RWwG) algorithm for this problem is as follows.

Algorithm D.1 (Single move random walk Metropolis within Gibbs) *Given target posterior $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2|\boldsymbol{y})$ where $\boldsymbol{\theta}_2 = (\theta_{21}, \theta_{22}, \ldots, \theta_{2n})$, it is easy to sample from $p(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2, \boldsymbol{y})$, and given that the support of $p(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1, \boldsymbol{y})$ is $\Re^n$, iteration $t+1$ is obtained from iteration t via*

1. *Draw* $\boldsymbol{\theta}_1^{(t+1)} \sim p(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2^{(t)}, \boldsymbol{y})$

2. *For* $k = 1, 2 \ldots, n$

   Draw $\theta_{2k}^{(prop)} \sim N(\theta_{2k}^{(t)}, \eta_k)$ *and form the Metropolis acceptance ratio*

$$a_k^{(t)} = \frac{p(\theta_{2k}^{(prop)}|\boldsymbol{\theta}_1^{(t+1)}, \theta_{21}^{(t+1)}, , \ldots, \theta_{2k-1}^{(t+1)}, \theta_{2k+1}^{(t)}, \ldots, \theta_{2n}^{(t)}, \boldsymbol{y})}{p(\theta_{2k}^{(t)}|\boldsymbol{\theta}_1^{(t+1)}, \theta_{21}^{(t+1)}, , \ldots, \theta_{2k-1}^{(t+1)}, \theta_{2k+1}^{(t)}, \ldots, \theta_{2n}^{(t)}, \boldsymbol{y})}.$$

   Then set $\theta_{2k}^{(t+1)} = \theta_{2k}^{(prop)}$ *with probability* $r_k^{(t)} = \min(a_k, 1)$ *and otherwise set* $\theta_{2k}^{(t+1)} = \theta_{2k}^{(t)}$

A major problem with this algorithm is that the η_ks must be selected so that the algorithm Metropolis steps accept a reasonable amount of time. According to Gelman et al. (1996) the optimal acceptance rate in a narrow set of problems is 0.44 for a single dimensional random walk, though this is often used as a guideline for more complex problems. We adaptively tune η_k during the burn-in period of the chain in a manner discussed in Andrieu and Thoms (2008). The computed Metropolis acceptance probability for θ_{2k} is denoted by $r_k^{(t)} = \min(a_k^{(t)}, 1)$; let r^* denote the target acceptance probability. Then we evolve η_k over time via

$$\log\ \eta_k^{(t+1)} = \log\ \eta_k^{(t)} + \gamma^{(t+1)}(r_k^{(t)} - r^*)$$

where $\gamma^{(t)}$ is a fixed sequence decreasing to zero fast enough to enough that the algorithm converges. The intuition here is that if the acceptance rate is too high, we can increase the effective search space by increasing the random walk standard deviation, while if it is too low, we can decrease the effective search space by decreasing the random walk standard deviation. We use $r^* = 0.44$. A tuning method such as this implies that $(\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots)$ is no longer a Markov chain, and additional conditions are required to ensure convergence to the target posterior distribution. So in practice our RWwG algorithms run Algorithm D.1 with the tuning method described above until convergence and until the η_ks settle into a rough

equilibrium, then we use Algorithm D.1 without tuning but using the last known values of the η_ks. In practice this is equivalent setting $\gamma^{(t)} = 0$ after the burn-in period. During the burn-in we set $\gamma^{(t)} = 1$ and we initialize at $\eta_k^{(0)} = 1$ for all k.

An alternative to RWwG algorithms are block random walk Metropolis within Gibbs algorithms (B-RWwG). Suppose we have an estimate of the covariance structure between the elements of $\boldsymbol{\theta}_2$ in the posterior. Then

Algorithm D.2 (Block random walk Metropolis within Gibbs) *Given target posterior $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2|\boldsymbol{y})$ where it is easy to sample from $p(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2, \boldsymbol{y})$, the support of $p(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1, \boldsymbol{y})$ is $\Re^n$, and given an estimate $\boldsymbol{\Sigma}$ of $\mathrm{cov}(\boldsymbol{\theta}_2|\boldsymbol{y})$, iteration $t+1$ is obtained from iteration t via*

1. *Draw $\boldsymbol{\theta}_1^{(t+1)} \sim p(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2^{(t)}, \boldsymbol{y})$*

2. *Draw $\boldsymbol{\theta}_2^{(prop)} \sim N(\boldsymbol{\theta}_2^{(t)}, \eta\boldsymbol{\Sigma})$ and form the Metropolis acceptance ratio*

$$a = \frac{p(\boldsymbol{\theta}_2^{(prop)}|\boldsymbol{\theta}_1^{(t+1)}, \boldsymbol{y})}{p(\boldsymbol{\theta}_2^{(t)}|\boldsymbol{\theta}_1^{(t+1)}, \boldsymbol{y})}.$$

Then set $\boldsymbol{\theta}_2^{(t+1)} = \boldsymbol{\theta}_2^{(prop)}$ with probability $\min(a, 1)$ and otherwise set $\boldsymbol{\theta}_2^{(t+1)} = \boldsymbol{\theta}_2^{(t)}$.

We make this algorithm adaptive as well by tuning η in the same fashion as the η_ks above. The main difference is that this algorithm must be initialized with a crude estimate of $\boldsymbol{\Sigma}$ by running another algorithm, e.g. Algorithm D.1, or Algorithm D.2 with $\boldsymbol{\Sigma}$ chosen to be diagonal and manually tuned.

OF1	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	3147.42	2726.79	0.00	∞	0.00	0.00	1.00	360.50	0.01	0.01	0.58	482.50
BBPSO-MC	88237.79	4013.36	0.00	∞	89130.57	4684.99	0.00	∞	88274.59	4528.45	0.00	∞
BBPSOxp-MC	81476.62	3295.56	0.00	∞	75471.96	4929.58	0.00	∞	59817.42	5244.39	0.00	∞
AT-BBPSO-MC												
$df = 1, \quad R^* = 0.1$	1.71	0.55	0.00	∞	2.01	0.57	0.00	∞	2.04	0.66	0.00	∞
$df = 1, \quad R^* = 0.3$	0.00	0.00	1.00	449.00	0.00	0.00	1.00	463.00	0.01	0.00	0.46	∞
$df = 1, \quad R^* = 0.5$	0.00	0.00	1.00	286.00	0.00	0.00	1.00	308.00	0.00	0.00	1.00	353.50
$df = 1, \quad R^* = 0.7$	0.00	0.00	1.00	223.00	0.00	0.00	1.00	254.50	0.00	0.00	1.00	328.00
$df = 3, \quad R^* = 0.1$	7.17	2.25	0.00	∞	6.91	1.65	0.00	∞	5.11	1.45	0.00	∞
$df = 3, \quad R^* = 0.3$	0.02	0.01	0.10	∞	0.02	0.01	0.02	∞	0.03	0.01	0.00	∞
$df = 3, \quad R^* = 0.5$	0.00	0.00	1.00	330.00	0.00	0.00	1.00	348.00	0.00	0.00	1.00	390.50
$df = 3, \quad R^* = 0.7$	0.00	0.00	1.00	263.00	0.00	0.00	1.00	288.50	0.00	0.00	1.00	388.00
$df = 5, \quad R^* = 0.1$	12.49	3.78	0.00	∞	12.42	3.39	0.00	∞	7.43	3.03	0.00	∞
$df = 5, \quad R^* = 0.3$	0.03	0.01	0.00	∞	0.03	0.01	0.00	∞	0.07	0.03	0.00	∞
$df = 5, \quad R^* = 0.5$	0.00	0.00	1.00	357.00	0.00	0.00	1.00	373.00	0.00	0.00	1.00	420.50
$df = 5, \quad R^* = 0.7$	0.00	0.00	1.00	290.00	0.00	0.00	1.00	325.50	195.94	336.65	0.06	∞
$df = \infty, \; R^* = 0.1$	43.64	10.08	0.00	∞	37.75	8.20	0.00	∞	19.58	6.50	0.00	∞
$df = \infty, \; R^* = 0.3$	0.14	0.03	0.00	∞	0.17	0.05	0.00	∞	0.23	0.06	0.00	∞
$df = \infty, \; R^* = 0.5$	0.00	0.00	1.00	413.00	0.00	0.00	1.00	430.50	0.01	0.00	0.82	491.00
$df = \infty, \; R^* = 0.7$	0.00	0.00	1.00	357.00	0.00	0.00	1.00	443.00	16625.74	3842.72	0.00	∞
AT-BBPSOxp-MC												
$df = 1, \quad R^* = 0.1$	3.77	0.91	0.00	∞	2.22	0.76	0.00	∞	0.43	0.20	0.00	∞
$df = 1, \quad R^* = 0.3$	0.02	0.01	0.00	∞	0.01	0.01	0.22	∞	0.00	0.00	1.00	448.50
$df = 1, \quad R^* = 0.5$	0.00	0.00	1.00	349.50	0.00	0.00	1.00	349.50	0.00	0.00	1.00	323.00
$df = 1, \quad R^* = 0.7$	0.00	0.00	1.00	283.50	0.00	0.00	1.00	295.00	0.00	0.00	1.00	285.00
$df = 3, \quad R^* = 0.1$	11.64	2.95	0.00	∞	6.57	2.30	0.00	∞	1.52	0.91	0.00	∞
$df = 3, \quad R^* = 0.3$	0.08	0.03	0.00	∞	0.05	0.02	0.00	∞	0.01	0.01	0.20	∞
$df = 3, \quad R^* = 0.5$	0.00	0.00	1.00	399.50	0.00	0.00	1.00	407.50	0.00	0.00	1.00	374.00
$df = 3, \quad R^* = 0.7$	0.00	0.00	1.00	340.50	0.00	0.00	1.00	365.50	0.00	0.00	1.00	365.00
$df = 5, \quad R^* = 0.1$	18.53	4.90	0.00	∞	9.29	2.96	0.00	∞	1.86	0.75	0.00	∞
$df = 5, \quad R^* = 0.3$	0.14	0.04	0.00	∞	0.10	0.03	0.00	∞	0.02	0.01	0.04	∞
$df = 5, \quad R^* = 0.5$	0.00	0.00	1.00	430.50	0.00	0.00	1.00	436.00	0.00	0.00	1.00	404.00
$df = 5, \quad R^* = 0.7$	0.00	0.00	1.00	389.50	0.01	0.02	0.92	451.50	0.04	0.13	0.74	454.50
$df = \infty, \; R^* = 0.1$	43.01	9.65	0.00	∞	23.58	7.63	0.00	∞	4.82	1.83	0.00	∞
$df = \infty, \; R^* = 0.3$	0.42	0.12	0.00	∞	0.32	0.09	0.00	∞	0.07	0.04	0.00	∞
$df = \infty, \; R^* = 0.5$	0.01	0.00	0.96	482.50	0.01	0.00	0.84	489.00	0.00	0.00	1.00	454.00
$df = \infty, \; R^* = 0.7$	0.06	0.15	0.24	∞	1766.73	1214.77	0.00	∞	2237.55	1691.20	0.00	∞
DI-PSO												
$\alpha = 50, \quad \beta = 1$	5800.59	3155.97	0.00	∞	698.71	1183.33	0.00	∞	35.24	79.82	0.00	∞
$\alpha = 50, \quad \beta = 2$	8130.93	5431.61	0.00	∞	2254.74	1775.35	0.00	∞	707.41	570.80	0.00	∞
$\alpha = 50, \quad \beta = 4$	14881.90	8467.41	0.00	∞	6133.60	3281.73	0.00	∞	3051.71	2314.76	0.00	∞
$\alpha = 100, \; \beta = 1$	3574.49	3170.07	0.00	∞	146.17	331.79	0.00	∞	0.34	0.91	0.22	∞
$\alpha = 100, \; \beta = 2$	7903.23	4250.18	0.00	∞	1248.39	1326.94	0.00	∞	83.35	140.17	0.00	∞
$\alpha = 100, \; \beta = 4$	13455.03	6484.96	0.00	∞	4378.86	3004.31	0.00	∞	1591.49	1194.36	0.00	∞
$\alpha = 200, \; \beta = 1$	2746.62	2890.48	0.00	∞	27.67	126.87	0.00	∞	0.00	0.00	0.96	384.50
$\alpha = 200, \; \beta = 2$	8261.84	4537.27	0.00	∞	180.57	334.02	0.00	∞	3.38	7.17	0.02	∞
$\alpha = 200, \; \beta = 4$	12100.60	6066.27	0.00	∞	2947.83	2171.07	0.00	∞	581.14	1413.97	0.00	∞
AT-PSO												
$R^* = 0.1$	118.60	228.55	0.00	∞	188.52	718.43	0.00	∞	10169.59	7405.93	0.00	∞
$R^* = 0.3$	70.10	326.06	0.04	∞	0.00	0.00	1.00	339.00	0.01	0.01	0.86	464.50
$R^* = 0.5$	373.93	894.14	0.00	∞	0.00	0.00	1.00	275.00	0.00	0.00	1.00	302.50
$R^* = 0.7$	1932.49	1906.13	0.00	∞	0.15	0.99	0.90	395.50	0.00	0.00	0.98	343.00

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

OF2	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	9058.41	9685.19	0.00	∞	1264.42	799.32	0.00	∞	5339.50	3143.93	0.00	∞
BBPSO-MC	11883975.00	628634.30	0.00	∞	12276069.13	699375.46	0.00	∞	11493578.31	1172502.58	0.00	∞
BBPSOxp-MC	11010472.51	544947.26	0.00	∞	10140651.29	619325.29	0.00	∞	7004850.88	1225368.46	0.00	∞
AT-BBPSO-MC												
$df = 1, \quad R^* = 0.1$	945.06	802.75	0.00	∞	710.80	469.14	0.00	∞	774.49	747.78	0.00	∞
$df = 1, \quad R^* = 0.3$	327.20	240.51	0.00	∞	926.29	1132.84	0.00	∞	1008.87	1034.00	0.00	∞
$df = 1, \quad R^* = 0.5$	469.09	503.21	0.00	∞	1468.95	1561.26	0.00	∞	3253.60	2878.65	0.00	∞
$df = 1, \quad R^* = 0.7$	1406.45	1261.92	0.00	∞	3977.07	3875.19	0.00	∞	10836.34	7853.42	0.00	∞
$df = 3, \quad R^* = 0.1$	219.80	111.62	0.00	∞	202.99	142.50	0.00	∞	255.86	157.95	0.00	∞
$df = 3, \quad R^* = 0.3$	19.11	13.28	0.00	∞	63.30	60.89	0.00	∞	145.93	153.16	0.00	∞
$df = 3, \quad R^* = 0.5$	29.80	35.95	0.00	∞	214.82	214.48	0.00	∞	846.76	697.18	0.00	∞
$df = 3, \quad R^* = 0.7$	144.09	103.71	0.00	∞	1033.32	796.45	0.00	∞	8079.02	20596.37	0.00	∞
$df = 5, \quad R^* = 0.1$	185.82	78.05	0.00	∞	187.84	85.91	0.00	∞	237.84	122.56	0.00	∞
$df = 5, \quad R^* = 0.3$	10.85	8.18	0.00	∞	24.21	18.38	0.00	∞	55.58	39.08	0.00	∞
$df = 5, \quad R^* = 0.5$	11.60	10.52	0.00	∞	54.75	40.96	0.00	∞	468.53	369.98	0.00	∞
$df = 5, \quad R^* = 0.7$	72.38	51.38	0.00	∞	422.19	277.83	0.00	∞	8889.40	7833.07	0.00	∞
$df = \infty, \; R^* = 0.1$	282.38	102.62	0.00	∞	262.49	79.80	0.00	∞	281.59	115.66	0.00	∞
$df = \infty, \; R^* = 0.3$	7.83	6.04	0.00	∞	16.43	10.41	0.00	∞	37.22	23.14	0.00	∞
$df = \infty, \; R^* = 0.5$	6.84	5.16	0.00	∞	35.82	32.97	0.00	∞	261.12	152.46	0.00	∞
$df = \infty, \; R^* = 0.7$	50.86	36.80	0.00	∞	484.76	270.15	0.00	∞	1167787.27	401166.29	0.00	∞
AT-BBPSOxp-MC												
$df = 1, \quad R^* = 0.1$	1872.52	1038.69	0.00	∞	1799.53	862.91	0.00	∞	942.70	590.06	0.00	∞
$df = 1, \quad R^* = 0.3$	1204.75	814.82	0.00	∞	1864.65	1185.93	0.00	∞	961.45	928.89	0.00	∞
$df = 1, \quad R^* = 0.5$	1881.72	1041.49	0.00	∞	3108.80	1482.50	0.00	∞	2519.47	1307.36	0.00	∞
$df = 1, \quad R^* = 0.7$	5161.81	3212.03	0.00	∞	7854.74	4468.87	0.00	∞	7775.57	4525.87	0.00	∞
$df = 3, \quad R^* = 0.1$	565.47	256.64	0.00	∞	665.17	307.78	0.00	∞	508.77	268.54	0.00	∞
$df = 3, \quad R^* = 0.3$	191.75	130.61	0.00	∞	363.46	260.03	0.00	∞	267.88	190.70	0.00	∞
$df = 3, \quad R^* = 0.5$	438.28	273.27	0.00	∞	654.47	404.48	0.00	∞	1110.91	1424.90	0.00	∞
$df = 3, \quad R^* = 0.7$	1384.56	909.83	0.00	∞	3503.63	3161.16	0.00	∞	4256.67	3547.43	0.00	∞
$df = 5, \quad R^* = 0.1$	480.63	187.88	0.00	∞	504.39	220.77	0.00	∞	400.66	214.64	0.00	∞
$df = 5, \quad R^* = 0.3$	115.80	68.47	0.00	∞	197.49	138.10	0.00	∞	225.22	142.07	0.00	∞
$df = 5, \quad R^* = 0.5$	181.62	128.13	0.00	∞	333.87	159.91	0.00	∞	472.13	276.68	0.00	∞
$df = 5, \quad R^* = 0.7$	660.28	313.88	0.00	∞	1817.45	834.00	0.00	∞	4632.54	4302.74	0.00	∞
$df = \infty, \; R^* = 0.1$	586.91	176.77	0.00	∞	595.51	179.25	0.00	∞	541.43	255.32	0.00	∞
$df = \infty, \; R^* = 0.3$	81.20	47.02	0.00	∞	128.97	94.06	0.00	∞	176.75	109.79	0.00	∞
$df = \infty, \; R^* = 0.5$	119.58	66.07	0.00	∞	232.02	163.01	0.00	∞	559.93	315.23	0.00	∞
$df = \infty, \; R^* = 0.7$	971.24	600.68	0.00	∞	118742.43	156281.96	0.00	∞	34066.02	28609.67	0.00	∞
DI-PSO												
$\alpha = 50, \quad \beta = 1$	40389.25	24692.95	0.00	∞	18720.02	11239.63	0.00	∞	8622.85	4980.17	0.00	∞
$\alpha = 50, \quad \beta = 2$	44518.25	22261.16	0.00	∞	21028.28	10707.64	0.00	∞	12726.73	5322.96	0.00	∞
$\alpha = 50, \quad \beta = 4$	51851.57	25120.63	0.00	∞	29532.84	13000.01	0.00	∞	16540.60	8846.75	0.00	∞
$\alpha = 100, \; \beta = 1$	24843.60	15674.35	0.00	∞	10917.91	7294.55	0.00	∞	5011.53	2384.94	0.00	∞
$\alpha = 100, \; \beta = 2$	37890.78	24040.40	0.00	∞	17226.82	8980.99	0.00	∞	10305.47	5737.74	0.00	∞
$\alpha = 100, \; \beta = 4$	52019.62	25824.86	0.00	∞	30086.99	20743.14	0.00	∞	12073.62	5451.03	0.00	∞
$\alpha = 200, \; \beta = 1$	22007.15	13078.38	0.00	∞	5785.75	4635.98	0.00	∞	3694.82	2434.96	0.00	∞
$\alpha = 200, \; \beta = 2$	32862.70	16618.65	0.00	∞	13173.95	6095.13	0.00	∞	8033.58	4532.79	0.00	∞
$\alpha = 200, \; \beta = 4$	50850.73	21907.04	0.00	∞	21202.31	11345.72	0.00	∞	14321.18	7476.13	0.00	∞
AT-PSO												
$R^* = 0.1$	7359.75	6803.21	0.00	∞	8340.43	7908.49	0.00	∞	56057.72	26008.24	0.00	∞
$R^* = 0.3$	7582.23	12259.21	0.00	∞	593.85	754.83	0.00	∞	5252.44	4221.37	0.00	∞
$R^* = 0.5$	17909.32	16070.66	0.00	∞	1397.43	1598.40	0.00	∞	1846.47	1312.13	0.00	∞
$R^* = 0.7$	43178.25	24887.81	0.00	∞	6301.86	5349.85	0.00	∞	5760.60	3355.07	0.00	∞

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

OF3	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	970637.16	1719993.18	0.00	∞	144.21	197.18	0.00	∞	207.96	306.95	0.00	∞
BBPSO-MC	121023363.91	13812838.21	0.00	∞	154011414.01	16232288.40	0.00	∞	174968862.42	23613990.88	0.00	∞
BBPSOxp-MC	138645962.86	15657816.44	0.00	∞	132653861.04	13853873.26	0.00	∞	89975682.91	15709701.24	0.00	∞
AT-BBPSO-MC												
$df = 1, \quad R^* = 0.1$	502.81	549.75	0.00	∞	489.99	659.01	0.00	∞	417.60	453.85	0.00	∞
$df = 1, \quad R^* = 0.3$	238.76	403.40	0.00	∞	241.03	308.07	0.00	∞	221.48	324.13	0.00	∞
$df = 1, \quad R^* = 0.5$	195.84	235.99	0.00	∞	122.20	115.62	0.00	∞	230.69	508.80	0.00	∞
$df = 1, \quad R^* = 0.7$	226.04	526.49	0.00	∞	274.61	575.37	0.00	∞	225.18	415.80	0.00	∞
$df = 3, \quad R^* = 0.1$	872.47	954.91	0.00	∞	603.26	714.65	0.00	∞	657.78	863.02	0.00	∞
$df = 3, \quad R^* = 0.3$	187.54	155.35	0.00	∞	140.43	115.93	0.00	∞	267.09	432.14	0.00	∞
$df = 3, \quad R^* = 0.5$	179.88	235.11	0.00	∞	156.27	214.31	0.00	∞	172.14	335.49	0.00	∞
$df = 3, \quad R^* = 0.7$	217.20	455.21	0.00	∞	123.77	165.76	0.00	∞	207.58	550.05	0.00	∞
$df = 5, \quad R^* = 0.1$	781.19	711.43	0.00	∞	679.51	634.98	0.00	∞	541.45	530.74	0.00	∞
$df = 5, \quad R^* = 0.3$	221.15	284.28	0.00	∞	214.66	361.21	0.00	∞	221.80	250.50	0.00	∞
$df = 5, \quad R^* = 0.5$	206.31	228.63	0.00	∞	144.48	112.27	0.00	∞	165.02	319.30	0.00	∞
$df = 5, \quad R^* = 0.7$	182.94	257.84	0.00	∞	48.60	73.43	0.00	∞	108.13	360.85	0.00	∞
$df = \infty, \; R^* = 0.1$	1382.42	1071.14	0.00	∞	1089.55	708.37	0.00	∞	704.85	580.05	0.00	∞
$df = \infty, \; R^* = 0.3$	269.15	320.43	0.00	∞	141.57	144.43	0.00	∞	160.88	290.82	0.00	∞
$df = \infty, \; R^* = 0.5$	191.11	147.95	0.00	∞	194.51	297.34	0.00	∞	165.00	212.70	0.00	∞
$df = \infty, \; R^* = 0.7$	146.08	127.14	0.00	∞	61.55	73.27	0.00	∞	317.42	999.55	0.00	∞
AT-BBPSOxp-MC												
$df = 1, \quad R^* = 0.1$	611.07	434.02	0.00	∞	402.89	264.11	0.00	∞	216.75	152.59	0.00	∞
$df = 1, \quad R^* = 0.3$	221.23	231.63	0.00	∞	129.90	92.90	0.00	∞	92.61	60.58	0.00	∞
$df = 1, \quad R^* = 0.5$	170.81	394.57	0.00	∞	107.35	136.60	0.00	∞	104.84	110.07	0.00	∞
$df = 1, \quad R^* = 0.7$	136.87	248.84	0.00	∞	131.50	432.59	0.00	∞	98.33	135.63	0.00	∞
$df = 3, \quad R^* = 0.1$	720.92	215.53	0.00	∞	560.46	249.45	0.00	∞	275.19	192.65	0.00	∞
$df = 3, \quad R^* = 0.3$	174.45	244.90	0.00	∞	127.44	58.88	0.00	∞	150.36	127.50	0.00	∞
$df = 3, \quad R^* = 0.5$	151.95	244.27	0.00	∞	92.78	78.27	0.00	∞	109.43	148.82	0.00	∞
$df = 3, \quad R^* = 0.7$	104.03	181.86	0.00	∞	168.15	436.88	0.00	∞	36.17	52.63	0.00	∞
$df = 5, \quad R^* = 0.1$	946.03	410.19	0.00	∞	555.51	250.59	0.00	∞	278.16	196.26	0.00	∞
$df = 5, \quad R^* = 0.3$	121.66	80.43	0.00	∞	120.35	74.89	0.00	∞	109.47	91.42	0.00	∞
$df = 5, \quad R^* = 0.5$	106.68	49.70	0.00	∞	129.63	124.28	0.00	∞	76.17	45.36	0.00	∞
$df = 5, \quad R^* = 0.7$	54.18	64.54	0.00	∞	39.95	84.35	0.00	∞	37.37	59.62	0.00	∞
$df = \infty, \; R^* = 0.1$	1429.56	774.39	0.00	∞	896.51	333.33	0.00	∞	390.78	225.22	0.00	∞
$df = \infty, \; R^* = 0.3$	109.28	76.19	0.00	∞	112.02	95.78	0.00	∞	115.01	89.85	0.00	∞
$df = \infty, \; R^* = 0.5$	103.74	35.01	0.00	∞	118.78	171.39	0.00	∞	90.54	74.14	0.00	∞
$df = \infty, \; R^* = 0.7$	43.35	75.92	0.00	∞	18.13	12.93	0.00	∞	19.99	20.18	0.00	∞
DI-PSO												
$\alpha = 50, \quad \beta = 1$	3854178.52	3563712.76	0.00	∞	476166.48	1068595.33	0.00	∞	49808.17	90745.61	0.00	∞
$\alpha = 50, \quad \beta = 2$	6562468.86	8285814.85	0.00	∞	1595754.13	1624066.71	0.00	∞	1327935.46	1951123.16	0.00	∞
$\alpha = 50, \quad \beta = 4$	19701402.98	12961045.13	0.00	∞	10884269.41	8299038.15	0.00	∞	5985429.81	3917709.91	0.00	∞
$\alpha = 100, \; \beta = 1$	1870977.66	2324247.39	0.00	∞	57070.98	164749.98	0.00	∞	1760.17	3897.70	0.00	∞
$\alpha = 100, \; \beta = 2$	7530967.66	8856235.46	0.00	∞	866151.25	1270221.03	0.00	∞	207921.17	463007.58	0.00	∞
$\alpha = 100, \; \beta = 4$	22584490.35	22402688.80	0.00	∞	7448049.14	6263068.66	0.00	∞	4916024.35	4602692.81	0.00	∞
$\alpha = 200, \; \beta = 1$	1217548.01	1884033.63	0.00	∞	3621.58	8954.06	0.00	∞	483.36	1129.34	0.00	∞
$\alpha = 200, \; \beta = 2$	4461093.42	4947722.54	0.00	∞	120433.51	307696.11	0.00	∞	3403.48	4242.88	0.00	∞
$\alpha = 200, \; \beta = 4$	17327523.00	13868599.90	0.00	∞	3061892.74	3273826.33	0.00	∞	1189001.99	1583478.70	0.00	∞
AT-PSO												
$R^* = 0.1$	5135.53	14617.98	0.00	∞	446469.66	2260240.37	0.00	∞	34766227.37	32913538.16	0.00	∞
$R^* = 0.3$	1577.07	4053.03	0.00	∞	176.34	324.07	0.00	∞	168.65	155.33	0.00	∞
$R^* = 0.5$	19154.96	47380.82	0.00	∞	315.71	711.01	0.00	∞	110.71	98.15	0.00	∞
$R^* = 0.7$	1297015.56	3307403.64	0.00	∞	395.90	933.41	0.00	∞	239.90	394.68	0.00	∞

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

OF4	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	26.31	11.97	0.00	∞	4.45	2.63	0.02	∞	3.17	1.81	0.00	∞
BBPSO-MC	151.12	10.61	0.00	∞	144.75	15.12	0.00	∞	124.80	14.36	0.00	∞
BBPSOxp-MC	165.73	5.44	0.00	∞	159.62	8.67	0.00	∞	132.89	13.23	0.00	∞
AT-BBPSO-MC												
$df = 1, \quad R^* = 0.1$	5.11	2.09	0.00	∞	4.13	1.43	0.00	∞	3.82	1.43	0.00	∞
$df = 1, \quad R^* = 0.3$	4.87	1.98	0.02	∞	3.05	1.54	0.02	∞	2.88	1.66	0.06	∞
$df = 1, \quad R^* = 0.5$	6.74	1.77	0.00	∞	4.62	2.09	0.00	∞	3.96	1.84	0.00	∞
$df = 1, \quad R^* = 0.7$	8.20	2.12	0.00	∞	7.04	2.17	0.00	∞	5.75	1.94	0.00	∞
$df = 3, \quad R^* = 0.1$	5.58	1.31	0.00	∞	5.60	1.33	0.00	∞	5.77	1.80	0.00	∞
$df = 3, \quad R^* = 0.3$	3.24	1.93	0.08	∞	1.41	0.97	0.16	∞	1.15	0.94	0.24	∞
$df = 3, \quad R^* = 0.5$	4.30	1.80	0.00	∞	2.49	1.56	0.06	∞	2.38	1.23	0.06	∞
$df = 3, \quad R^* = 0.7$	5.18	2.00	0.00	∞	3.41	1.44	0.00	∞	3.84	1.82	0.02	∞
$df = 5, \quad R^* = 0.1$	6.30	1.42	0.00	∞	6.44	1.59	0.00	∞	6.60	1.63	0.00	∞
$df = 5, \quad R^* = 0.3$	2.63	1.46	0.04	∞	1.51	1.14	0.18	∞	1.07	1.05	0.20	∞
$df = 5, \quad R^* = 0.5$	3.60	2.01	0.04	∞	2.13	1.28	0.08	∞	1.77	1.22	0.14	∞
$df = 5, \quad R^* = 0.7$	4.36	1.40	0.00	∞	3.01	1.58	0.02	∞	3.10	1.47	0.02	∞
$df = \infty, \; R^* = 0.1$	9.32	1.28	0.00	∞	9.44	1.44	0.00	∞	9.11	1.73	0.00	∞
$df = \infty, \; R^* = 0.3$	1.54	1.02	0.00	∞	0.86	1.08	0.00	∞	0.68	0.86	0.02	∞
$df = \infty, \; R^* = 0.5$	2.68	1.60	0.10	∞	1.41	1.07	0.16	∞	1.39	1.11	0.20	∞
$df = \infty, \; R^* = 0.7$	3.23	1.87	0.04	∞	1.88	1.51	0.18	∞	1.69	1.31	0.24	∞
AT-BBPSOxp-MC												
$df = 1, \quad R^* = 0.1$	3.25	1.09	0.00	∞	2.05	0.79	0.00	∞	0.57	0.41	0.00	∞
$df = 1, \quad R^* = 0.3$	0.66	0.76	0.26	∞	0.26	0.46	0.62	490.50	0.59	0.72	0.54	445.00
$df = 1, \quad R^* = 0.5$	1.43	0.98	0.20	∞	1.20	1.10	0.32	∞	1.47	1.25	0.22	∞
$df = 1, \quad R^* = 0.7$	2.78	1.75	0.08	∞	2.45	1.45	0.06	∞	2.58	1.56	0.06	∞
$df = 3, \quad R^* = 0.1$	4.33	1.11	0.00	∞	3.33	1.04	0.00	∞	0.80	0.53	0.00	∞
$df = 3, \quad R^* = 0.3$	0.22	0.44	0.02	∞	0.12	0.31	0.38	∞	0.31	0.49	0.70	458.50
$df = 3, \quad R^* = 0.5$	0.51	0.72	0.60	335.50	0.51	0.93	0.64	327.50	0.84	0.99	0.48	∞
$df = 3, \quad R^* = 0.7$	1.24	0.95	0.22	∞	1.18	0.99	0.24	∞	1.58	1.21	0.20	∞
$df = 5, \quad R^* = 0.1$	5.12	1.04	0.00	∞	4.14	1.26	0.00	∞	0.91	0.62	0.00	∞
$df = 5, \quad R^* = 0.3$	0.17	0.35	0.00	∞	0.05	0.19	0.24	∞	0.27	0.43	0.72	458.00
$df = 5, \quad R^* = 0.5$	0.34	0.57	0.70	334.00	0.30	0.49	0.70	327.50	0.74	0.82	0.46	∞
$df = 5, \quad R^* = 0.7$	1.10	0.99	0.34	∞	0.89	0.95	0.40	∞	1.94	1.57	0.14	∞
$df = \infty, \; R^* = 0.1$	6.79	1.32	0.00	∞	5.75	1.36	0.00	∞	1.29	0.60	0.00	∞
$df = \infty, \; R^* = 0.3$	0.03	0.01	0.00	∞	0.08	0.23	0.00	∞	0.20	0.38	0.78	470.50
$df = \infty, \; R^* = 0.5$	0.25	0.46	0.76	340.50	0.21	0.40	0.78	334.00	0.63	0.76	0.52	323.00
$df = \infty, \; R^* = 0.7$	0.71	0.83	0.46	∞	0.95	1.02	0.42	∞	1.50	1.11	0.22	∞
DI-PSO												
$\alpha = 50, \quad \beta = 1$	32.00	14.44	0.00	∞	12.58	5.43	0.00	∞	6.46	3.78	0.00	∞
$\alpha = 50, \quad \beta = 2$	35.82	12.86	0.00	∞	18.69	7.41	0.00	∞	9.51	4.88	0.00	∞
$\alpha = 50, \quad \beta = 4$	54.12	18.93	0.00	∞	28.25	10.10	0.00	∞	15.24	5.62	0.00	∞
$\alpha = 100, \; \beta = 1$	23.66	12.32	0.00	∞	11.01	6.22	0.00	∞	4.88	2.75	0.00	∞
$\alpha = 100, \; \beta = 2$	32.56	13.79	0.00	∞	13.56	5.63	0.00	∞	7.34	4.10	0.00	∞
$\alpha = 100, \; \beta = 4$	52.63	21.66	0.00	∞	24.35	9.50	0.00	∞	13.91	5.97	0.00	∞
$\alpha = 200, \; \beta = 1$	19.10	6.98	0.00	∞	8.20	4.44	0.00	∞	4.53	2.38	0.00	∞
$\alpha = 200, \; \beta = 2$	36.31	12.85	0.00	∞	11.45	5.21	0.00	∞	5.46	2.87	0.00	∞
$\alpha = 200, \; \beta = 4$	51.61	20.50	0.00	∞	20.01	7.50	0.00	∞	10.15	3.55	0.00	∞
AT-PSO												
$R^* = 0.1$	7.78	3.44	0.00	∞	5.20	2.00	0.00	∞	23.42	13.89	0.00	∞
$R^* = 0.3$	15.00	6.19	0.00	∞	7.59	3.13	0.00	∞	4.11	2.31	0.00	∞
$R^* = 0.5$	22.59	8.87	0.00	∞	11.60	5.14	0.00	∞	6.01	2.99	0.00	∞
$R^* = 0.7$	35.35	14.56	0.00	∞	15.32	4.51	0.00	∞	9.60	4.15	0.00	∞

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

OF5	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	28.03	26.04	0.00	∞	0.04	0.05	0.24	∞	0.10	0.09	0.00	∞
BBPSO-MC	874.83	38.75	0.00	∞	871.43	40.28	0.00	∞	841.29	53.32	0.00	∞
BBPSOxp-MC	777.25	29.42	0.00	∞	723.32	36.24	0.00	∞	580.79	51.18	0.00	∞
AT-BBPSO-MC												
$df = 1,\ \ R^* = 0.1$	0.86	0.09	0.00	∞	0.90	0.07	0.00	∞	0.91	0.06	0.00	∞
$df = 1,\ \ R^* = 0.3$	0.02	0.01	0.34	∞	0.02	0.01	0.38	∞	0.03	0.02	0.00	∞
$df = 1,\ \ R^* = 0.5$	0.01	0.01	0.54	432.50	0.01	0.01	0.56	457.00	0.01	0.01	0.54	464.50
$df = 1,\ \ R^* = 0.7$	0.02	0.02	0.44	∞	0.01	0.02	0.60	413.50	0.04	0.05	0.10	∞
$df = 3,\ \ R^* = 0.1$	1.06	0.02	0.00	∞	1.06	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	∞	0.31	0.13	0.00	∞
$df = 3,\ \ R^* = 0.5$	0.01	0.01	0.52	461.00	0.01	0.01	0.72	467.50	0.10	0.09	0.02	∞
$df = 3,\ \ R^* = 0.7$	0.02	0.01	0.44	∞	0.09	0.12	0.08	∞	225.75	51.35	0.00	∞
$df = 5,\ \ R^* = 0.1$	1.11	0.03	0.00	∞	1.11	0.03	0.00	∞	1.08	0.03	0.00	∞
$df = 5,\ \ R^* = 0.3$	0.17	0.05	0.00	∞	0.30	0.11	0.00	∞	0.64	0.14	0.00	∞
$df = 5,\ \ R^* = 0.5$	0.02	0.01	0.34	∞	0.02	0.01	0.22	∞	0.73	0.20	0.00	∞
$df = 5,\ \ R^* = 0.7$	0.02	0.01	0.30	∞	31.72	19.26	0.00	∞	476.26	46.12	0.00	∞
$df = \infty,\ R^* = 0.1$	1.46	0.10	0.00	∞	1.47	0.13	0.00	∞	1.22	0.07	0.00	∞
$df = \infty,\ R^* = 0.3$	0.68	0.13	0.00	∞	0.84	0.09	0.00	∞	0.97	0.05	0.00	∞
$df = \infty,\ R^* = 0.5$	0.04	0.01	0.00	∞	0.25	0.11	0.00	∞	6.49	6.73	0.00	∞
$df = \infty,\ R^* = 0.7$	0.88	0.18	0.00	∞	377.73	43.53	0.00	∞	655.88	49.74	0.00	∞
AT-BBPSOxp-MC												
$df = 1,\ \ R^* = 0.1$	0.98	0.04	0.00	∞	0.89	0.10	0.00	∞	0.48	0.18	0.00	∞
$df = 1,\ \ R^* = 0.3$	0.05	0.02	0.00	∞	0.06	0.03	0.00	∞	0.02	0.02	0.22	∞
$df = 1,\ \ R^* = 0.5$	0.01	0.01	0.74	419.50	0.01	0.01	0.66	438.50	0.01	0.01	0.52	482.50
$df = 1,\ \ R^* = 0.7$	0.02	0.03	0.58	439.50	0.02	0.02	0.60	437.50	0.02	0.02	0.48	∞
$df = 3,\ \ R^* = 0.1$	1.10	0.03	0.00	∞	1.06	0.02	0.00	∞	0.93	0.09	0.00	∞
$df = 3,\ \ R^* = 0.3$	0.47	0.10	0.00	∞	0.47	0.12	0.00	∞	0.17	0.07	0.00	∞
$df = 3,\ \ R^* = 0.5$	0.04	0.03	0.00	∞	0.07	0.05	0.02	∞	0.04	0.03	0.02	∞
$df = 3,\ \ R^* = 0.7$	1.57	1.96	0.00	∞	74.80	39.86	0.00	∞	95.06	41.38	0.00	∞
$df = 5,\ \ R^* = 0.1$	1.19	0.06	0.00	∞	1.10	0.03	0.00	∞	1.00	0.05	0.00	∞
$df = 5,\ \ R^* = 0.3$	0.70	0.10	0.00	∞	0.71	0.11	0.00	∞	0.28	0.12	0.00	∞
$df = 5,\ \ R^* = 0.5$	0.14	0.07	0.00	∞	0.34	0.15	0.00	∞	0.22	0.13	0.00	∞
$df = 5,\ \ R^* = 0.7$	116.60	42.47	0.00	∞	306.40	53.78	0.00	∞	261.33	43.65	0.00	∞
$df = \infty,\ R^* = 0.1$	1.49	0.12	0.00	∞	1.26	0.09	0.00	∞	1.05	0.03	0.00	∞
$df = \infty,\ R^* = 0.3$	0.96	0.06	0.00	∞	0.93	0.07	0.00	∞	0.64	0.11	0.00	∞
$df = \infty,\ R^* = 0.5$	0.73	0.12	0.00	∞	0.95	0.07	0.00	∞	0.89	0.15	0.00	∞
$df = \infty,\ R^* = 0.7$	395.68	33.71	0.00	∞	504.84	39.71	0.00	∞	389.26	40.73	0.00	∞
DI-PSO												
$\alpha = 50,\ \ \beta = 1$	60.30	38.12	0.00	∞	7.26	10.66	0.00	∞	1.14	0.61	0.00	∞
$\alpha = 50,\ \ \beta = 2$	66.12	33.41	0.00	∞	21.06	15.97	0.00	∞	6.18	4.13	0.00	∞
$\alpha = 50,\ \ \beta = 4$	127.46	57.77	0.00	∞	56.24	25.74	0.00	∞	24.65	13.98	0.00	∞
$\alpha = 100,\ \beta = 1$	28.60	21.21	0.00	∞	1.62	1.71	0.00	∞	0.27	0.24	0.00	∞
$\alpha = 100,\ \beta = 2$	73.41	67.24	0.00	∞	7.98	8.39	0.00	∞	1.90	1.34	0.00	∞
$\alpha = 100,\ \beta = 4$	129.76	51.11	0.00	∞	36.80	25.83	0.00	∞	14.49	11.96	0.00	∞
$\alpha = 200,\ \beta = 1$	25.75	23.08	0.00	∞	0.91	2.29	0.00	∞	0.06	0.07	0.28	∞
$\alpha = 200,\ \beta = 2$	72.48	45.02	0.00	∞	4.38	6.49	0.00	∞	0.66	0.34	0.00	∞
$\alpha = 200,\ \beta = 4$	126.67	76.77	0.00	∞	23.59	19.07	0.00	∞	4.98	4.04	0.00	∞
AT-PSO												
$R^* = 0.1$	2.25	1.98	0.00	∞	1.91	1.98	0.00	∞	98.02	57.26	0.00	∞
$R^* = 0.3$	1.09	1.89	0.00	∞	0.02	0.03	0.44	∞	0.05	0.05	0.16	∞
$R^* = 0.5$	5.65	11.07	0.00	∞	0.06	0.08	0.28	∞	0.02	0.02	0.46	∞
$R^* = 0.7$	32.51	39.20	0.00	∞	0.13	0.18	0.14	∞	0.07	0.17	0.26	∞

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

OF6	Global nbhd				Ring-3 nbhd				Ring-1 nbhd			
Algorithm	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$	Mean	SD	$\widehat{p}$	$\widehat{t}$
PSO	19.59	1.20	0.00	∞	15.41	8.13	0.04	∞	17.89	5.35	0.00	∞
BBPSO-MC	20.16	0.47	0.00	∞	20.24	0.50	0.00	∞	20.49	0.45	0.00	∞
BBPSOxp-MC	20.11	0.19	0.00	∞	20.23	0.16	0.00	∞	20.06	0.11	0.00	∞
AT-BBPSO-MC												
$df = 1, \quad R^* = 0.1$	17.10	6.89	0.00	∞	9.21	9.18	0.00	∞	6.23	8.00	0.00	∞
$df = 1, \quad R^* = 0.3$	18.66	4.77	0.00	∞	15.62	8.27	0.00	∞	13.05	9.54	0.00	∞
$df = 1, \quad R^* = 0.5$	19.05	3.82	0.02	∞	17.29	6.59	0.06	∞	16.41	7.57	0.04	∞
$df = 1, \quad R^* = 0.7$	19.31	3.54	0.00	∞	18.57	4.73	0.00	∞	19.05	3.24	0.00	∞
$df = 3, \quad R^* = 0.1$	19.61	2.52	0.00	∞	18.65	4.81	0.00	∞	17.12	7.06	0.00	∞
$df = 3, \quad R^* = 0.3$	19.82	0.05	0.00	∞	19.87	0.28	0.00	∞	19.08	3.97	0.00	∞
$df = 3, \quad R^* = 0.5$	19.85	0.17	0.00	∞	19.60	2.85	0.02	∞	20.08	0.55	0.00	∞
$df = 3, \quad R^* = 0.7$	19.89	0.22	0.00	∞	20.13	0.40	0.00	∞	20.35	0.42	0.00	∞
$df = 5, \quad R^* = 0.1$	19.99	0.06	0.00	∞	19.75	2.52	0.00	∞	17.90	6.25	0.00	∞
$df = 5, \quad R^* = 0.3$	19.83	0.03	0.00	∞	19.87	0.24	0.00	∞	19.66	2.85	0.00	∞
$df = 5, \quad R^* = 0.5$	19.84	0.09	0.00	∞	19.90	0.25	0.00	∞	20.12	0.62	0.00	∞
$df = 5, \quad R^* = 0.7$	19.83	0.05	0.00	∞	20.00	0.33	0.00	∞	20.33	0.39	0.00	∞
$df = \infty, \; R^* = 0.1$	20.10	0.07	0.00	∞	20.10	0.09	0.00	∞	20.47	0.29	0.00	∞
$df = \infty, \; R^* = 0.3$	19.83	0.02	0.00	∞	19.84	0.15	0.00	∞	19.68	2.83	0.00	∞
$df = \infty, \; R^* = 0.5$	19.83	0.03	0.00	∞	19.92	0.26	0.00	∞	20.10	0.39	0.00	∞
$df = \infty, \; R^* = 0.7$	19.83	0.03	0.00	∞	19.99	0.33	0.00	∞	20.36	0.38	0.00	∞
AT-BBPSOxp-MC												
$df = 1, \quad R^* = 0.1$	14.13	8.56	0.00	∞	11.18	9.07	0.00	∞	12.71	9.05	0.00	∞
$df = 1, \quad R^* = 0.3$	18.04	6.04	0.00	∞	16.06	7.84	0.00	∞	16.02	7.64	0.00	∞
$df = 1, \quad R^* = 0.5$	18.96	4.84	0.06	∞	18.16	5.26	0.00	∞	17.59	5.83	0.06	∞
$df = 1, \quad R^* = 0.7$	20.17	0.17	0.00	∞	20.04	0.87	0.00	∞	19.60	2.01	0.00	∞
$df = 3, \quad R^* = 0.1$	20.22	0.11	0.00	∞	20.17	0.11	0.00	∞	19.70	2.21	0.00	∞
$df = 3, \quad R^* = 0.3$	19.70	2.82	0.00	∞	20.12	0.15	0.00	∞	19.99	0.16	0.00	∞
$df = 3, \quad R^* = 0.5$	20.17	0.13	0.00	∞	20.20	0.12	0.00	∞	20.02	0.12	0.00	∞
$df = 3, \quad R^* = 0.7$	20.20	0.12	0.00	∞	20.21	0.10	0.00	∞	20.06	0.12	0.00	∞
$df = 5, \quad R^* = 0.1$	20.22	0.10	0.00	∞	20.19	0.11	0.00	∞	20.03	0.12	0.00	∞
$df = 5, \quad R^* = 0.3$	20.08	0.16	0.00	∞	20.15	0.13	0.00	∞	20.03	0.12	0.00	∞
$df = 5, \quad R^* = 0.5$	20.12	0.13	0.00	∞	20.21	0.12	0.00	∞	20.03	0.12	0.00	∞
$df = 5, \quad R^* = 0.7$	20.19	0.12	0.00	∞	20.22	0.10	0.00	∞	20.05	0.11	0.00	∞
$df = \infty, \; R^* = 0.1$	20.18	0.09	0.00	∞	20.19	0.09	0.00	∞	20.03	0.11	0.00	∞
$df = \infty, \; R^* = 0.3$	20.03	0.14	0.00	∞	20.09	0.16	0.00	∞	20.02	0.13	0.00	∞
$df = \infty, \; R^* = 0.5$	20.08	0.14	0.00	∞	20.15	0.13	0.00	∞	20.02	0.09	0.00	∞
$df = \infty, \; R^* = 0.7$	20.12	0.15	0.00	∞	20.19	0.10	0.00	∞	20.04	0.10	0.00	∞
DI-PSO												
$\alpha = 50, \quad \beta = 1$	19.81	0.52	0.00	∞	18.11	3.34	0.00	∞	18.80	3.52	0.00	∞
$\alpha = 50, \quad \beta = 2$	19.72	1.24	0.00	∞	18.36	2.48	0.00	∞	19.59	1.65	0.00	∞
$\alpha = 50, \quad \beta = 4$	19.85	0.60	0.00	∞	19.54	0.81	0.00	∞	20.03	0.42	0.00	∞
$\alpha = 100, \; \beta = 1$	19.09	2.07	0.00	∞	15.60	6.01	0.00	∞	18.20	4.53	0.00	∞
$\alpha = 100, \; \beta = 2$	19.39	1.51	0.00	∞	18.33	3.97	0.00	∞	19.48	2.42	0.00	∞
$\alpha = 100, \; \beta = 4$	20.18	0.62	0.00	∞	19.71	1.32	0.00	∞	20.22	0.42	0.00	∞
$\alpha = 200, \; \beta = 1$	18.96	2.05	0.00	∞	14.66	6.67	0.00	∞	17.95	4.74	0.00	∞
$\alpha = 200, \; \beta = 2$	20.15	0.64	0.00	∞	19.78	1.46	0.00	∞	20.29	0.35	0.00	∞
$\alpha = 200, \; \beta = 4$	20.32	0.69	0.00	∞	20.17	0.81	0.00	∞	20.42	0.41	0.00	∞
AT-PSO												
$R^* = 0.1$	19.31	2.85	0.00	∞	16.16	6.93	0.00	∞	20.25	0.32	0.00	∞
$R^* = 0.3$	19.58	1.24	0.00	∞	14.64	7.42	0.04	∞	16.28	7.41	0.04	∞
$R^* = 0.5$	19.84	0.67	0.00	∞	19.00	2.47	0.00	∞	19.84	2.12	0.00	∞
$R^* = 0.7$	20.00	0.23	0.00	∞	19.61	1.19	0.00	∞	20.11	0.29	0.00	∞

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

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.

Chen, Z. and Dunson, D. B. (2003). "Random effects selection in linear mixed models." *Biometrics*, 59, 4, 762–769.

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.

Frühwirth-Schnatter, S. and Tüchler, R. (2008). "Bayesian parsimonious covariance estimation for hierarchical linear mixed models." *Statistics and Computing*, 18, 1, 1–13.

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.

Magnus, J. R. (1988). *Linear structures*. No. 42 in Griffin's Statistical Monographs and Courses. New York: Oxford University Press.

Magnus, J. R. and Neudecker, H. (1980). "The elimination matrix: some lemmas and applications." *SIAM Journal on Algebraic Discrete Methods*, 1, 4, 422–449.

— (2005). *Matrix differential calculus with applications in statistics and econometrics*. 3rd ed. New York: John Wiley & Sons.

Smith, W. and Hocking, R. (1972). "Algorithm AS 53: Wishart variate generator." *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 21, 3, 341–345.

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.