Adaptively Tuned Particle Swarm Optimization for Spatial Design

Matthew Simpson

Department of Statistics, University of Missouri
SAS Institute, Inc.

August 3, 2016

Joint work with Christopher Wikle and Scott H. Holan

Research supported by the NSF-Census Research Network

Overview of the Talk

1. What is particle swarm optimization (PSO)? (Blum and Li, 2008; Clerc, 2010, 2012)
2. New adaptively-tuned PSO algorithms.
3. Using (adaptively-tuned) PSO for spatial design.
4. Example adding to an existing monitoring network.

Particle Swarm Optimization — Intuition

Click!

Put a "swarm" of particles in the search space:

> Don't search alone, pay attention to what your neighbors are doing!

Particle Swarm Optimization

Goal: minimize some objective function $Q(\boldsymbol{\theta}) : \mathbb{R}^D \to \mathbb{R}$.

Populate Θ with n particles. Define particle i in period k by:

- a location $\boldsymbol{\theta}_i(k) \in \mathbb{R}^D$;
- a velocity $\boldsymbol{v}_i(k) \in \mathbb{R}^D$;
- a personal best location $\boldsymbol{p}_i(k) \in \mathbb{R}^D$;
- a neighborhood (group) best location $\boldsymbol{g}_i(k) \in \mathbb{R}^D$.

Particle Swarm Optimization

Goal: minimize some objective function $Q(\boldsymbol{\theta}) : \mathbb{R}^D \to \mathbb{R}$.

Populate Θ with n particles. Define particle i in period k by:

- a location $\boldsymbol{\theta}_i(k) \in \mathbb{R}^D$;
- a velocity $\boldsymbol{v}_i(k) \in \mathbb{R}^D$;
- a personal best location $\boldsymbol{p}_i(k) \in \mathbb{R}^D$;
- a neighborhood (group) best location $\boldsymbol{g}_i(k) \in \mathbb{R}^D$.

Basic PSO: update particle i from k to $k+1$ via:

- For $j = 1, 2, \ldots, D$:

$$\begin{aligned} v_{ij}(k+1) &= \omega v_{ij}(k) + \mathrm{U}(0, \phi_1) \times \{p_{ij}(k) - \theta_{ij}(k)\} \\ &\qquad + \mathrm{U}(0, \phi_2) \times \{g_{ij}(k) - \theta_{ij}(k)\} \\ &= \text{inertia} + \text{cognitive} + \text{social}, \\ \theta_{ij}(k+1) &= \theta_{ij}(k) + v_{ij}(k+1), \end{aligned}$$

- Then update personal and group best locations.

PSO — Parameters

Inertia parameter: ω.

- Controls the particle's tendency to keep moving in the same direction.

Cognitive correction factor: ϕ_1.

- Controls the particle's tendency to move toward its personal best.

Social correction factor: ϕ_2.

- Controls the particle's tendency to move toward its neighborhood best.

Default choices:

- $\omega = 0.7298$, $\phi_1 = \phi_2 = 1.496$ (Clerc and Kennedy, 2002).
- $\omega = 1/(2\ln 2) \approx 0.721$, $\phi_1 = \phi_2 = 1/2 + \ln 2 \approx 1.193$ (Clerc, 2006).

PSO — Neighborhood Topologies

Sometimes it is useful to restrict the flow of information across the swarm — e.g. complicated objective functions with many local optima.

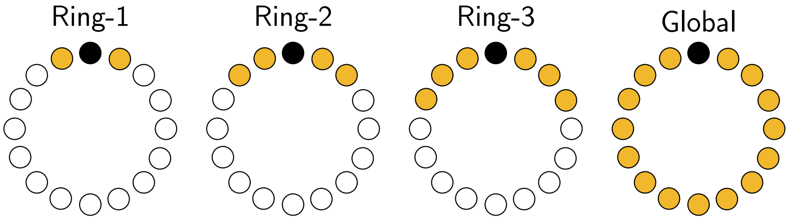
Particles are only informed by their *neighbors*.

PSO — Neighborhood Topologies

Sometimes it is useful to restrict the flow of information across the swarm — e.g. complicated objective functions with many local optima.

Particles are only informed by their *neighbors*.

Easy to visualize example: Ring-k neighborhood topology.



Each particle is informed by k neighbors to the left and k to the right, *no matter where they are in the search space*.

Stochastic Star Topology, and Other Bells and Whistles

We use the stochastic star neighborhood topology (Miranda et al., 2008).

- Each particle informs itself and m random particles.
  $\rightarrow$ sampled with replacement once during initialization.
- On average each particle is informed by m particles.
- A small number of particles will be informed by many particles.

Stochastic Star Topology, and Other Bells and Whistles

We use the stochastic star neighborhood topology (Miranda et al., 2008).

- Each particle informs itself and m random particles.
  $\rightarrow$ sampled with replacement once during initialization.
- On average each particle is informed by m particles.
- A small number of particles will be informed by many particles.

Many variants available (Clerc, 2012), (Simpson et al., 2017, appendix).

- Handling search space constraints.
- Coordinate free velocity updates.
- Parallelization.
- Asynchronous updates.
- Redraw neighborhoods.

Bare Bones PSO (BBPSO)

Developed by Kennedy (2003).

Strips out the velocity term:

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

Mimics the behavior of standard PSO.

Easier to analyze, but tends to perform worse.

Adaptively Tuned BBPSO

Add flexibility to the scale parameter:

$$\theta_{ij}(k+1) \sim \mathrm{T}_{df}\left(\frac{p_{ij}(k)+g_{ij}(k)}{2}, \sigma^2(k)|p_{ij}(k)-g_{ij}(k)|^2\right).$$

with e.g. $df = 1$ by default.

- Larger $\sigma^2(k)$: more exploration.
- Smaller $\sigma^2(k)$: more exploitation.

How to choose $\sigma^2(k)$'s progression?

Adaptively Tuned BBPSO

Add flexibility to the scale parameter:

$$\theta_{ij}(k+1) \sim \mathrm{T}_{df}\left(\frac{p_{ij}(k)+g_{ij}(k)}{2}, \sigma^2(k)|p_{ij}(k)-g_{ij}(k)|^2\right).$$

with e.g. $df = 1$ by default.

- Larger $\sigma^2(k)$: more exploration.
- Smaller $\sigma^2(k)$: more exploitation.

How to choose $\sigma^2(k)$'s progression?

Analogy with adaptively tuned random walk Metropolis.
(Andrieu and Thoms, 2008)

Adaptively Tuned BBPSO — $\sigma^2(k)$'s progression

Define the improvement rate of the swarm in period k:

$$R(k) = \text{proportion of particles that improved on their personal best last period.}$$

Adaptively Tuned BBPSO — $\sigma^2(k)$'s progression

Define the improvement rate of the swarm in period k:

$$R(k) = \text{proportion of particles that improved on their personal best last period.}$$

Let R^* denote a target improvement rate, and
c denote an adjustment factor.

Update $\sigma^2(k)$ via:

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

Adaptively Tuned BBPSO — $\sigma^2(k)$'s progression

Define the improvement rate of the swarm in period k:

$$R(k) = \text{proportion of particles that improved on their personal best last period.}$$

Let R^* denote a target improvement rate, and
c denote an adjustment factor.

Update $\sigma^2(k)$ via:

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

Larger c, faster changes to $\sigma^2(k)$.

Adaptively Tuned BBPSO — $\sigma^2(k)$'s progression

Define the improvement rate of the swarm in period k:

$$R(k) = \text{proportion of particles that improved on their personal best last period.}$$

Let R^* denote a target improvement rate, and
c denote an adjustment factor.

Update $\sigma^2(k)$ via:

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

Larger c, faster changes to $\sigma^2(k)$.

Defaults: $R^* \in [0.3, 0.5]$, $c = 0.1$.

Adaptively Tuning PSO

In PSO larger $\omega \implies$ more exploration, smaller $\omega \implies$ more exploitation.

Adaptively Tuning PSO

In PSO larger $\omega \implies$ more exploration, smaller $\omega \implies$ more exploitation.

Idea: deterministic inertia PSO (DI-PSO)
$\rightarrow$ slowly decrease $\omega(k)$ over time (Eberhart and Shi, 2000).

- Hard to set appropriately for any given problem.

Adaptively Tuning PSO

In PSO larger $\omega \implies$ more exploration, smaller $\omega \implies$ more exploitation.

Idea: deterministic inertia PSO (DI-PSO)
$\rightarrow$ slowly decrease $\omega(k)$ over time (Eberhart and Shi, 2000).

- Hard to set appropriately for any given problem.

AT-PSO: tune $\omega(k)$ just like $\sigma^2(k)$ in AT-BBPSO.

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

Adaptively Tuning PSO

In PSO larger $\omega \implies$ more exploration, smaller $\omega \implies$ more exploitation.

Idea: deterministic inertia PSO (DI-PSO)
$\rightarrow$ slowly decrease $\omega(k)$ over time (Eberhart and Shi, 2000).

- Hard to set appropriately for any given problem.

AT-PSO: tune $\omega(k)$ just like $\sigma^2(k)$ in AT-BBPSO.

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

Same defaults: $R^* \in [0.3, 0.5]$, $c = 0.1$.

Adaptively Tuning PSO

In PSO larger $\omega \implies$ more exploration, smaller $\omega \implies$ more exploitation.

Idea: deterministic inertia PSO (DI-PSO)
$\rightarrow$ slowly decrease $\omega(k)$ over time (Eberhart and Shi, 2000).

- Hard to set appropriately for any given problem.

AT-PSO: tune $\omega(k)$ just like $\sigma^2(k)$ in AT-BBPSO.

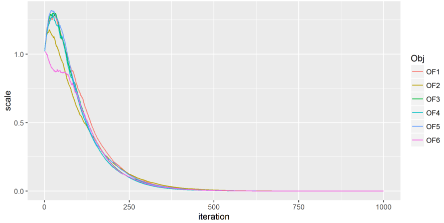
$$\log \omega(k+1) = \log \omega(k) + c\{R(k+1) - R^*\}.$$

Same defaults: $R^* \in [0.3, 0.5]$, $c = 0.1$.

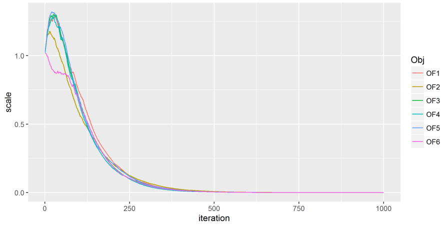
Similar PSO algorithm in spirit: Zhang et al. (2003).

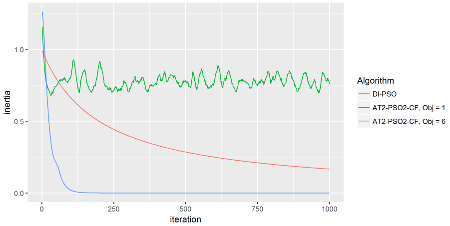
- ω is constant while ϕ_1 and ϕ_2 vary across time *and particle*.
- Can't use the same method to adapt ϕ_1 and ϕ_2.

Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):



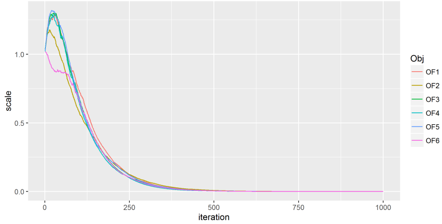
Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):





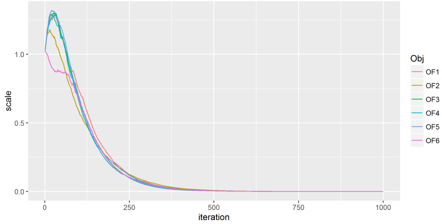
- AT-BBPSO's scale progressions are all remarkably similar.

Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):



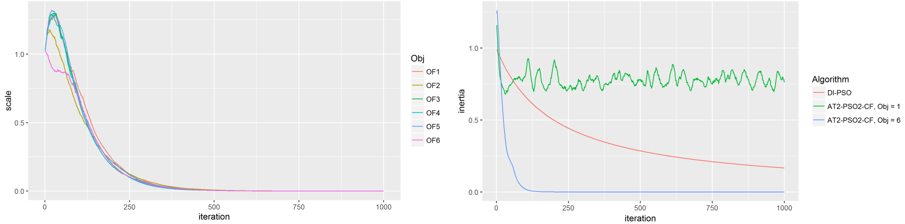
- AT-BBPSO's scale progressions are all remarkably similar.
- Often AT-BBPSO initially increases $\sigma^2(k)$ above 1, then starts decaying it to zero — initial burst of high-exploration... sometimes.

Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):



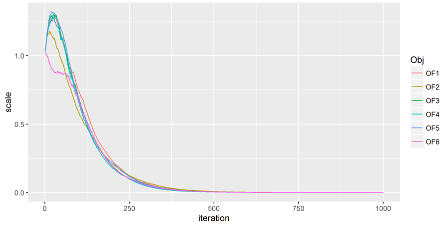
- AT-BBPSO's scale progressions are all remarkably similar.
- Often AT-BBPSO initially increases $\sigma^2(k)$ above 1, then starts decaying it to zero — initial burst of high-exploration... sometimes.
- DI-PSO's inertia has a nice smooth progression towards zero.

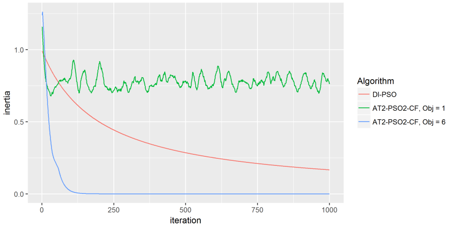
Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):



- AT-BBPSO's scale progressions are all remarkably similar.
- Often AT-BBPSO initially increases $\sigma^2(k)$ above 1, then starts decaying it to zero — initial burst of high-exploration... sometimes.
- DI-PSO's inertia has a nice smooth progression towards zero.
- AT-PSO's inertia typically bounces around a flat trendline.
  $\rightarrow$ Alternating between *relative* exploration and exploitation.

Example progressions of $\sigma^2(k)$ (left) and $\omega(k)$ (right):





- AT-BBPSO's scale progressions are all remarkably similar.
- Often AT-BBPSO initially increases $\sigma^2(k)$ above 1, then starts decaying it to zero — initial burst of high-exploration... sometimes.
- DI-PSO's inertia has a nice smooth progression towards zero.
- AT-PSO's inertia typically bounces around a flat trendline.
  $\rightarrow$ Alternating between *relative* exploration and exploitation.
- AT-PSO's inertia crashes to zero when it converges.
  $\rightarrow$ May be premature local convergence.

Comparing AT-PSO/BBPSO to PSO/BBPSO

Tuning $\omega(k)/\sigma^2(k)$ allows the swarm to adjust the exploration / exploitation tradeoff based on local conditions.

- This has a tendency to speed up convergence.
- ...but convergence may be premature in multi-modal problems.

Comparing AT-PSO/BBPSO to PSO/BBPSO

Tuning $\omega(k)/\sigma^2(k)$ allows the swarm to adjust the exploration / exploitation tradeoff based on local conditions.

- This has a tendency to speed up convergence.
- ...but convergence may be premature in multi-modal problems.

Overview of results from a simulation study with a variety of objective functions:

- BBPSO tends to perform poorly, but AT-BBPSO performs quite well.
- AT-BBPSO often the best for complex objective functions.
  $\rightarrow$ The T_{df} makes it fairly robust to many local optima.

Comparing AT-PSO/BBPSO to PSO/BBPSO

Tuning $\omega(k)/\sigma^2(k)$ allows the swarm to adjust the exploration / exploitation tradeoff based on local conditions.

- This has a tendency to speed up convergence.
- ...but convergence may be premature in multi-modal problems.

Overview of results from a simulation study with a variety of objective functions:

- BBPSO tends to perform poorly, but AT-BBPSO performs quite well.
- AT-BBPSO often the best for complex objective functions.
  $\rightarrow$ The T_{df} makes it fairly robust to many local optima.
- AT-PSO performs better than PSO on "hard enough" problems...

Comparing AT-PSO/BBPSO to PSO/BBPSO

Tuning $\omega(k)/\sigma^2(k)$ allows the swarm to adjust the exploration / exploitation tradeoff based on local conditions.

- This has a tendency to speed up convergence.
- ...but convergence may be premature in multi-modal problems.

Overview of results from a simulation study with a variety of objective functions:

- BBPSO tends to perform poorly, but AT-BBPSO performs quite well.
- AT-BBPSO often the best for complex objective functions.
  $\rightarrow$ The T_{df} makes it fairly robust to many local optima.
- AT-PSO performs better than PSO on "hard enough" problems...
- ...but has trouble with many local optima.

Spatial Design — Problem Setup

Goal: want to learn about the spatial field $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subset \mathbb{R}^2$.

Spatial Design — Problem Setup

Goal: want to learn about the spatial field $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subset \mathbb{R}^2$.

We observe a noisy signal of $Y(\boldsymbol{u})$ at N_s fixed monitoring locations, $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N_s}$, and assume:

$$Z(\boldsymbol{u}) = Y(\boldsymbol{u}) + \varepsilon(\boldsymbol{u})$$

for all $\boldsymbol{u} \in \mathcal{D}$, and $\varepsilon(\boldsymbol{u}) \overset{iid}{\sim} \mathrm{N}(0, \tau^2)$.

Spatial Design — Problem Setup

Goal: want to learn about the spatial field $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subset \mathbb{R}^2$.

We observe a noisy signal of $Y(\boldsymbol{u})$ at N_s fixed monitoring locations, $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N_s}$, and assume:

$$Z(\boldsymbol{u}) = Y(\boldsymbol{u}) + \varepsilon(\boldsymbol{u})$$

for all $\boldsymbol{u} \in \mathcal{D}$, and $\varepsilon(\boldsymbol{u}) \overset{iid}{\sim} \mathrm{N}(0, \tau^2)$.

Assume

$$Y(\boldsymbol{u}) \sim \mathrm{GP}(\boldsymbol{x}(\boldsymbol{u})'\beta, C(\cdot, \cdot))$$

where $\boldsymbol{x}(\boldsymbol{u})$ is a vector of covariates known at all locations $\boldsymbol{u} \in \mathcal{D}$.

Spatial Design — Problem Setup

Goal: want to learn about the spatial field $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subset \mathbb{R}^2$.

We observe a noisy signal of $Y(\boldsymbol{u})$ at N_s fixed monitoring locations, $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N_s}$, and assume:

$$Z(\boldsymbol{u}) = Y(\boldsymbol{u}) + \varepsilon(\boldsymbol{u})$$

for all $\boldsymbol{u} \in \mathcal{D}$, and $\varepsilon(\boldsymbol{u}) \overset{iid}{\sim} \mathrm{N}(0, \tau^2)$.

Assume

$$Y(\boldsymbol{u}) \sim \mathrm{GP}(\boldsymbol{x}(\boldsymbol{u})'\boldsymbol{\beta}, C(\cdot,\cdot))$$

where $\boldsymbol{x}(\boldsymbol{u})$ is a vector of covariates known at all locations $\boldsymbol{u} \in \mathcal{D}$.

Where should we put an additional N_d monitoring locations, $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{N_d}$?

Spatial Design — Problem Setup

Goal: want to learn about the spatial field $Y(\boldsymbol{u})$, $\boldsymbol{u} \in \mathcal{D} \subset \mathbb{R}^2$.

We observe a noisy signal of $Y(\boldsymbol{u})$ at N_s fixed monitoring locations, $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N_s}$, and assume:

$$Z(\boldsymbol{u}) = Y(\boldsymbol{u}) + \varepsilon(\boldsymbol{u})$$

for all $\boldsymbol{u} \in \mathcal{D}$, and $\varepsilon(\boldsymbol{u}) \overset{iid}{\sim} \mathrm{N}(0, \tau^2)$.

Assume

$$Y(\boldsymbol{u}) \sim \mathrm{GP}(\boldsymbol{x}(\boldsymbol{u})'\boldsymbol{\beta}, C(\cdot, \cdot))$$

where $\boldsymbol{x}(\boldsymbol{u})$ is a vector of covariates known at all locations $\boldsymbol{u} \in \mathcal{D}$.

Where should we put an additional N_d monitoring locations, $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_{N_d}$?

What if τ^2, $\boldsymbol{\beta}$, and $C(\cdot, \cdot)$ are unknown?

Spatial Design — MSPE and Kriging

Sensible goal: minimize MSPE in some sense.

When τ^2 and $C(\cdot,\cdot)$ are known, the universal kriging predictor is:

$$\widehat{Y}_{uk}(\boldsymbol{u};\boldsymbol{d}) = \boldsymbol{x}(\boldsymbol{u})'\widehat{\beta}_{gls} + \boldsymbol{c}_Y(\boldsymbol{u})'\boldsymbol{C}_Z^{-1}(\boldsymbol{Z} - \boldsymbol{X}\widehat{\beta}_{gls})$$

(Cressie and Wikle, 2011) where

$$\begin{aligned}
\boldsymbol{X} &= (\boldsymbol{x}(\boldsymbol{s}_1),\dots,\boldsymbol{x}(\boldsymbol{s}_{N_s}),\boldsymbol{x}(\boldsymbol{d}_1,\dots,\boldsymbol{x}(\boldsymbol{d}_{N_d}))',\\
\boldsymbol{Y} &= (Y(\boldsymbol{s}_1),\dots,Y(\boldsymbol{s}_{N_s}),Y(\boldsymbol{d}_1),\dots,Y(\boldsymbol{d}_{N_d}))',\\
\boldsymbol{Z} &= (Z(\boldsymbol{s}_1),\dots,Z(\boldsymbol{s}_{N_s}),Z(\boldsymbol{d}_1),\dots,Z(\boldsymbol{d}_{N_d}))',\\
\boldsymbol{C}_Z &= \mathrm{Cov}(\boldsymbol{Z}) = \tau^2\mathbf{I} + \mathrm{Cov}(\boldsymbol{Y}),\\
\boldsymbol{c}_Y &= \mathrm{Cov}(Y(\boldsymbol{u}),\boldsymbol{Y}),\\
\widehat{\beta}_{gls} &= (\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{Z}.
\end{aligned}$$

Spatial Design — Kriging Variances

Kriging MSPE: $\mathrm{E}\left\{Y(\boldsymbol{u}) - \widehat{Y}_{uk}(\boldsymbol{u})\right\}^2 = \sigma^2_{uk}(\boldsymbol{u};\boldsymbol{d}) =$

$$C(\boldsymbol{u},\boldsymbol{u}) - \boldsymbol{c}_Y(\boldsymbol{u})'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})$$
$$+\{\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})\}'(\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{X})^{-1}\{\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})\}$$

Spatial Design — Kriging Variances

Kriging MSPE: $\mathrm{E}\left\{Y(\boldsymbol{u}) - \widehat{Y}_{uk}(\boldsymbol{u})\right\}^2 = \sigma^2_{uk}(\boldsymbol{u};\boldsymbol{d}) =$

$$C(\boldsymbol{u},\boldsymbol{u}) - \boldsymbol{c}_Y(\boldsymbol{u})'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})$$
$$+ \{\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})\}'(\boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{X})^{-1}\{\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{X}'\boldsymbol{C}_Z^{-1}\boldsymbol{c}_Y(\boldsymbol{u})\}$$

What about when τ^2 and/or $C(\cdot,\cdot)$ are estimated?
$\rightarrow$ Parameter uncertainty universal kriging MSPE:

$$\mathrm{E}\left\{Y(\boldsymbol{u}) - \widehat{Y}_{uk}(\boldsymbol{u})\right\}^2 \approx \sigma^2_{puk}(\boldsymbol{u};\boldsymbol{d},\widehat{\boldsymbol{\theta}}) = \sigma^2_{uk}(\boldsymbol{u};\boldsymbol{d},\widehat{\boldsymbol{\theta}}) + \text{ stuff},$$

depending on the Fisher information matrix and the MLE of all parameters, denoted by $\widehat{\boldsymbol{\theta}}$ (Zimmerman and Cressie, 1992; Abt, 1999).

Spatial Design — Design Criteria

Ideal design criteria: choose design points to minimize...

- Mean/total MSPE: $\int_{\mathcal{D}} \sigma^2(\boldsymbol{u}) d\boldsymbol{u}$
- Maximum MSPE: $\max_{\boldsymbol{u} \in \mathcal{D}} \sigma^2(\boldsymbol{u})$

Spatial Design — Design Criteria

Ideal design criteria: choose design points to minimize...

- Mean/total MSPE: $\int_{\mathcal{D}} \sigma^2(\boldsymbol{u}) d\boldsymbol{u}$
- Maximum MSPE: $\max_{\boldsymbol{u} \in \mathcal{D}} \sigma^2(\boldsymbol{u})$

This is computationally infeasible.

Spatial Design — Design Criteria

Ideal design criteria: choose design points to minimize...

- Mean/total MSPE: $\int_{\mathcal{D}} \sigma^2(\boldsymbol{u}) d\boldsymbol{u}$
- Maximum MSPE: $\max_{\boldsymbol{u} \in \mathcal{D}} \sigma^2(\boldsymbol{u})$

This is computationally infeasible.

Realistic criteria: approximate with a grid of target points $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_t}$:

- Minimize $\sum_{i=1}^{N_t} \sigma^2(\boldsymbol{r}_i)$
- Minimize $\max_{i=1,2,\ldots,N_t} \sigma^2(\boldsymbol{r}_i)$

Thank you!

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

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 Blum, C. and Merkle, D., editors, *Swarm Intelligence: Introduction and Applications*, pages 43–85. Springer-Verlag, Berlin.

Clerc, M. (2006). Stagnation analysis in particle swarm optimisation or what happens when nothing happens. 17 pages. https://hal.archives-ouvertes.fr/hal-00122031.

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

Clerc, M. (2012). Standard particle swarm optimisation. 15 pages. https://hal.archives-ouvertes.fr/hal-00764996.

References II

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. (2011). *Statistics for Spatio-Temporal Data*. John Wiley & Sons, Hoboken, NJ.

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*, volume 1, pages 84–88. IEEE.

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

References III

Miranda, V., Keko, H., and Duque, A. J. (2008). Stochastic star communication topology in evolutionary particle swarms (EPSO). *International journal of computational intelligence research*, 4(2):105–116.

Simpson, M., Wikle, C. K., and Holan, S. H. (2017). Adaptively tuned particle swarm optimization with application to spatial design. *Stat*, 6(1):145–159.

Zhang, W., Liu, Y., and Clerc, M. (2003). An adaptive PSO algorithm for reactive power optimization. In *IET Conference Proceedings*, pages 302–307. Institution of Engineering and Technology.

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