SWARM INTELLIGENCE METHODS FOR STATISTICAL REGRESSION

Lecture 3

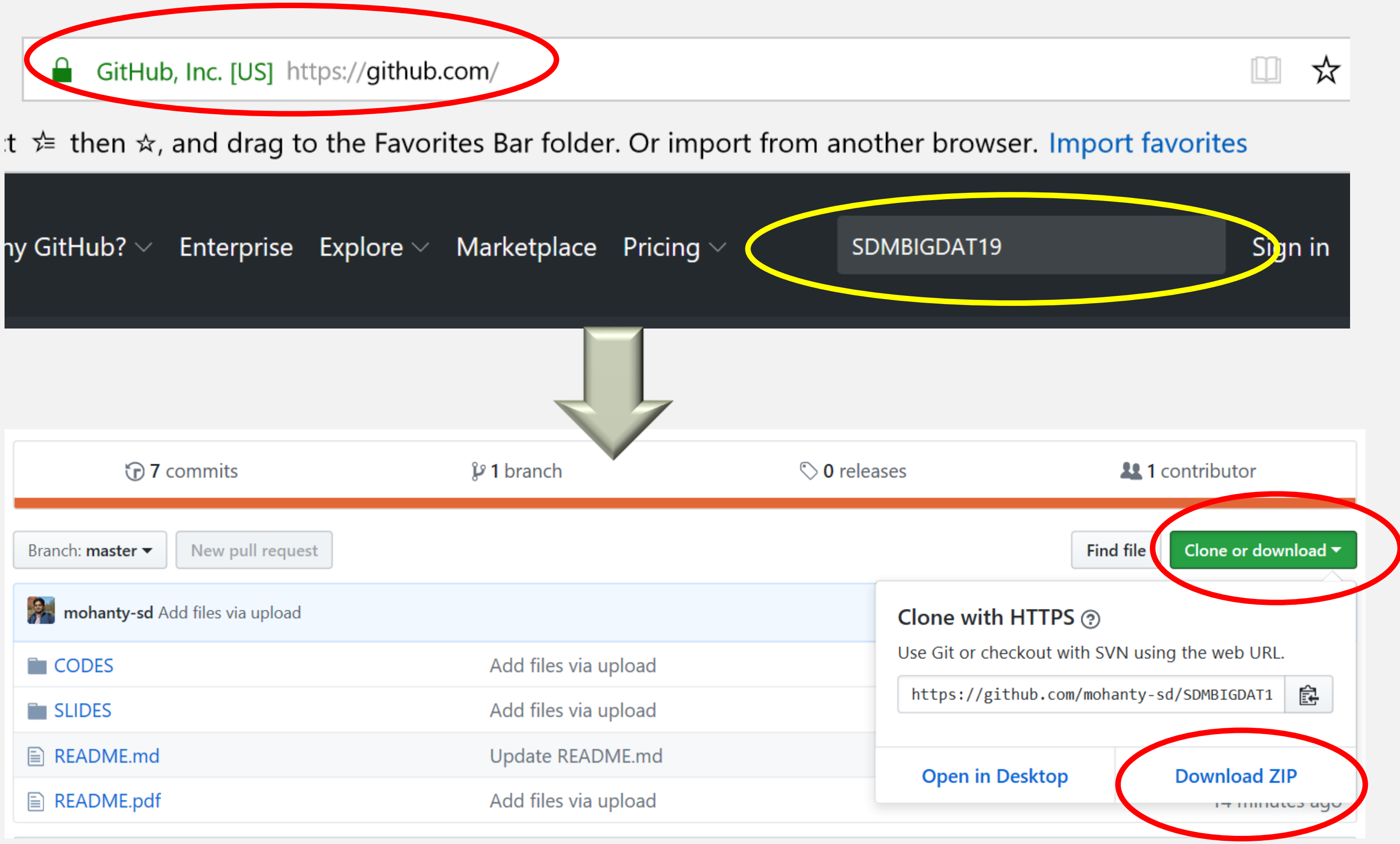
Soumya D. Mohanty

University of Texas Rio Grande Valley

ANNOUNCEMENT

- GitHub repository: SDMBIGDAT19
- Latest drafts of lecture slides
- Codes
  - Description and user manual to be added soon
- Repository will be deleted after end of course!

❖ Google drive shared folder remains primary outlet but slower update cycle

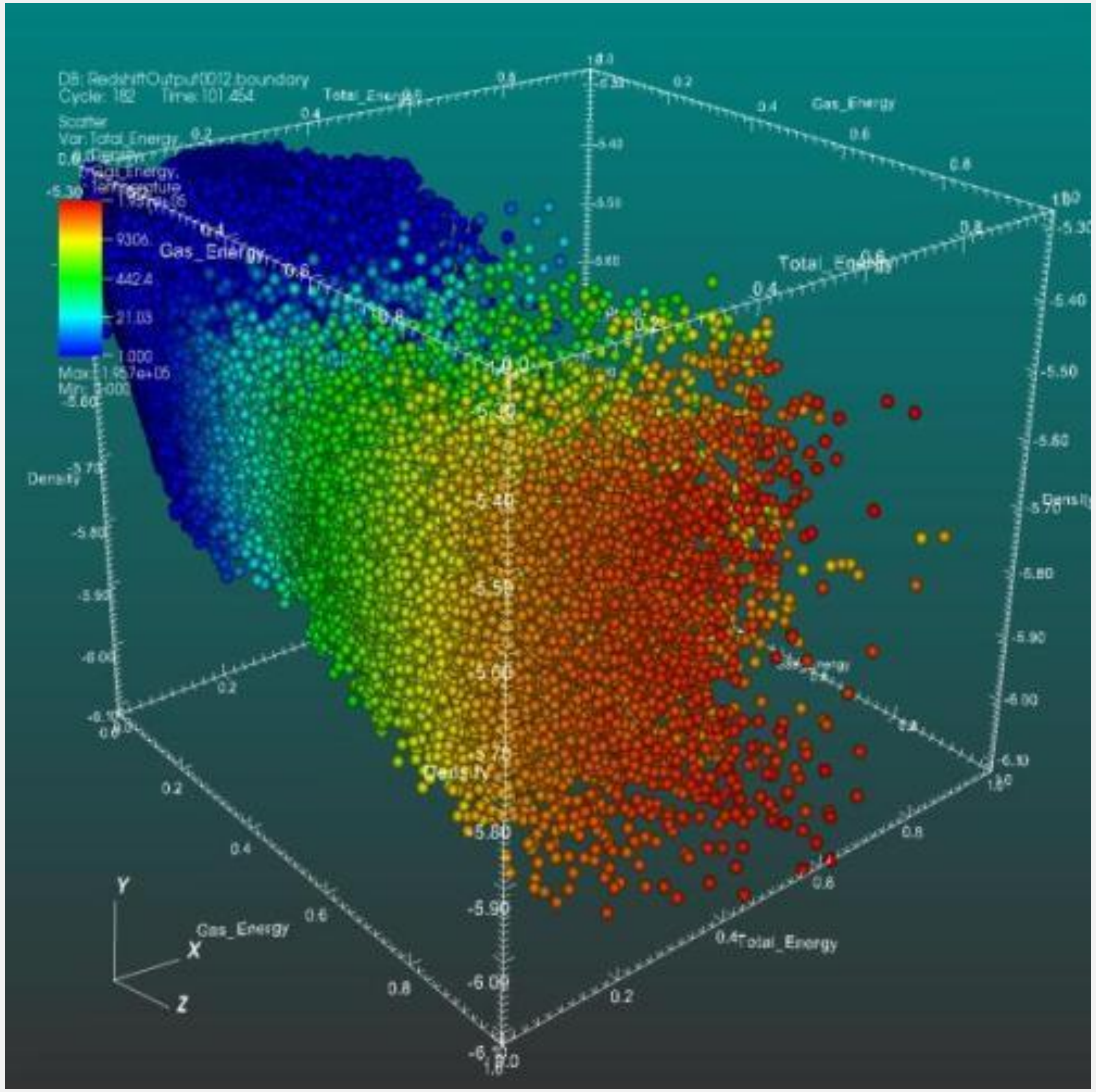


MOTIVATION

For large and complex data sets in the big data era, we need more flexible models

Flexibility may require models to be

- High-dimensional and/or
- Non-linear



Wikipedia: Data visualization

MOTIVATION

Global optimization of high-dimensional, non-linear statistical models is a challenging task

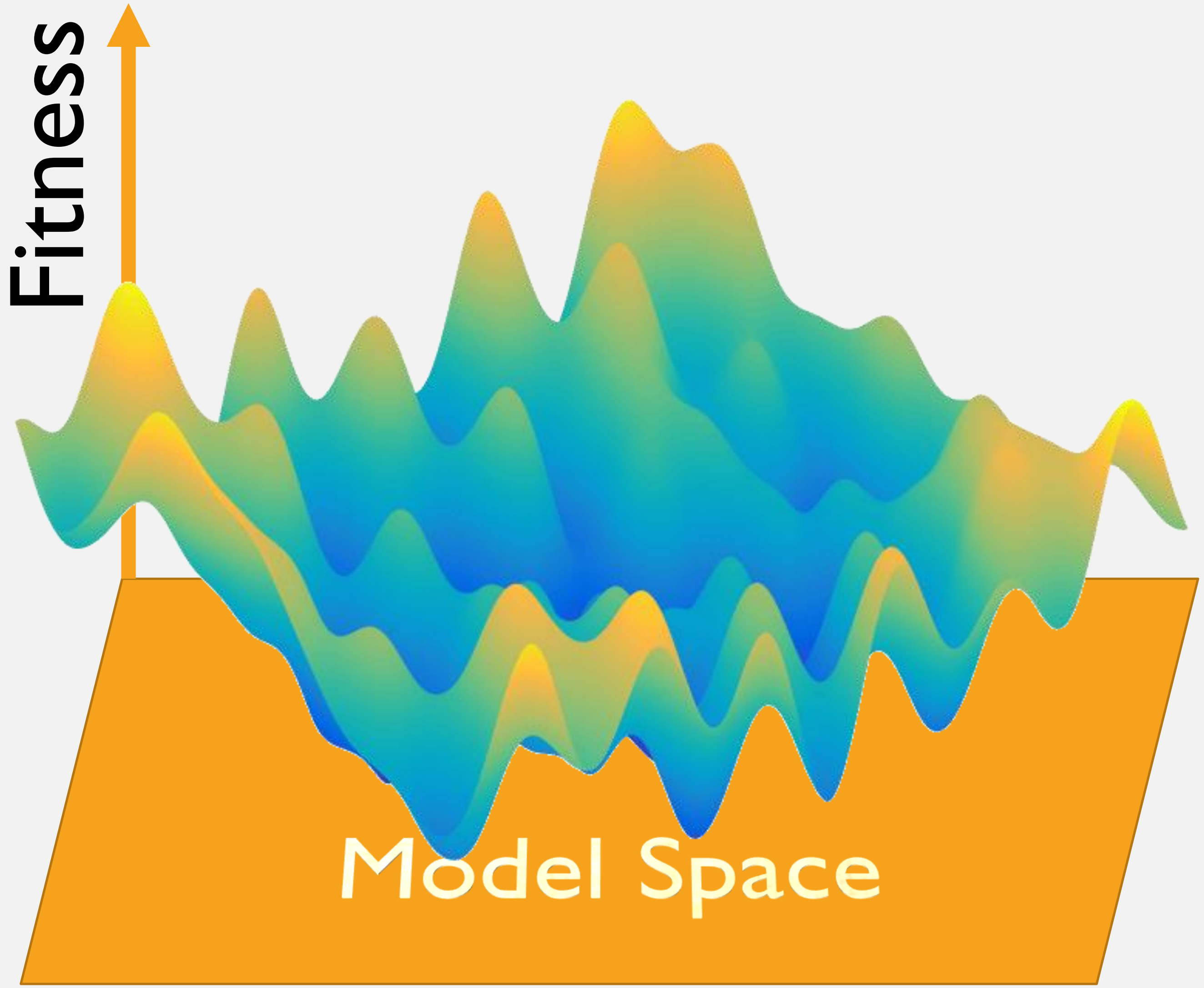


Image: https://users.ece.cmu.edu/~yuejiec/research.html

MOTIVATION

Success in breaking through the optimization barrier allows better modeling of data

Swarm intelligence (SI) methods can prove effective for optimization in statistical analysis

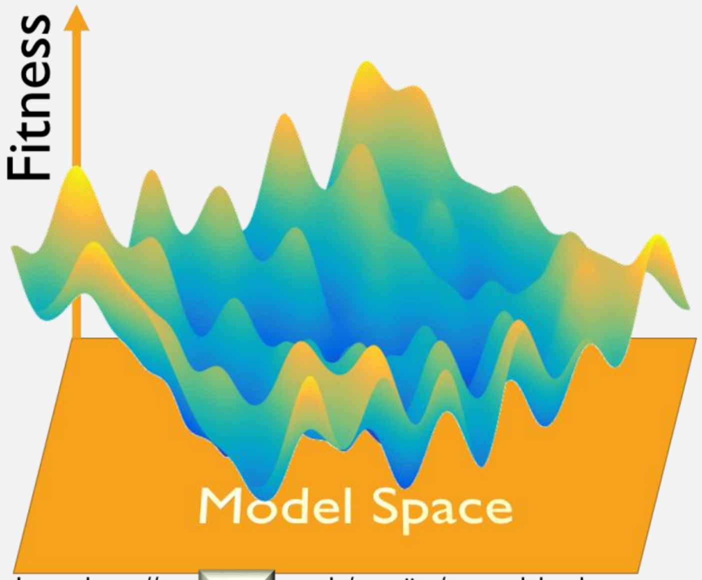
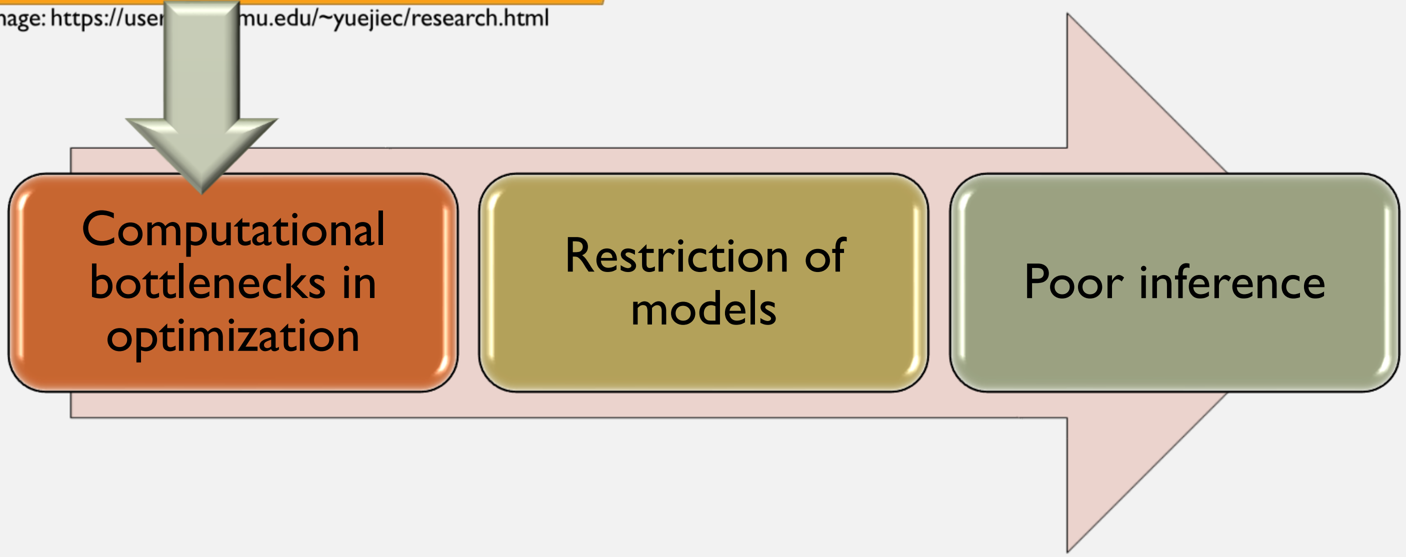


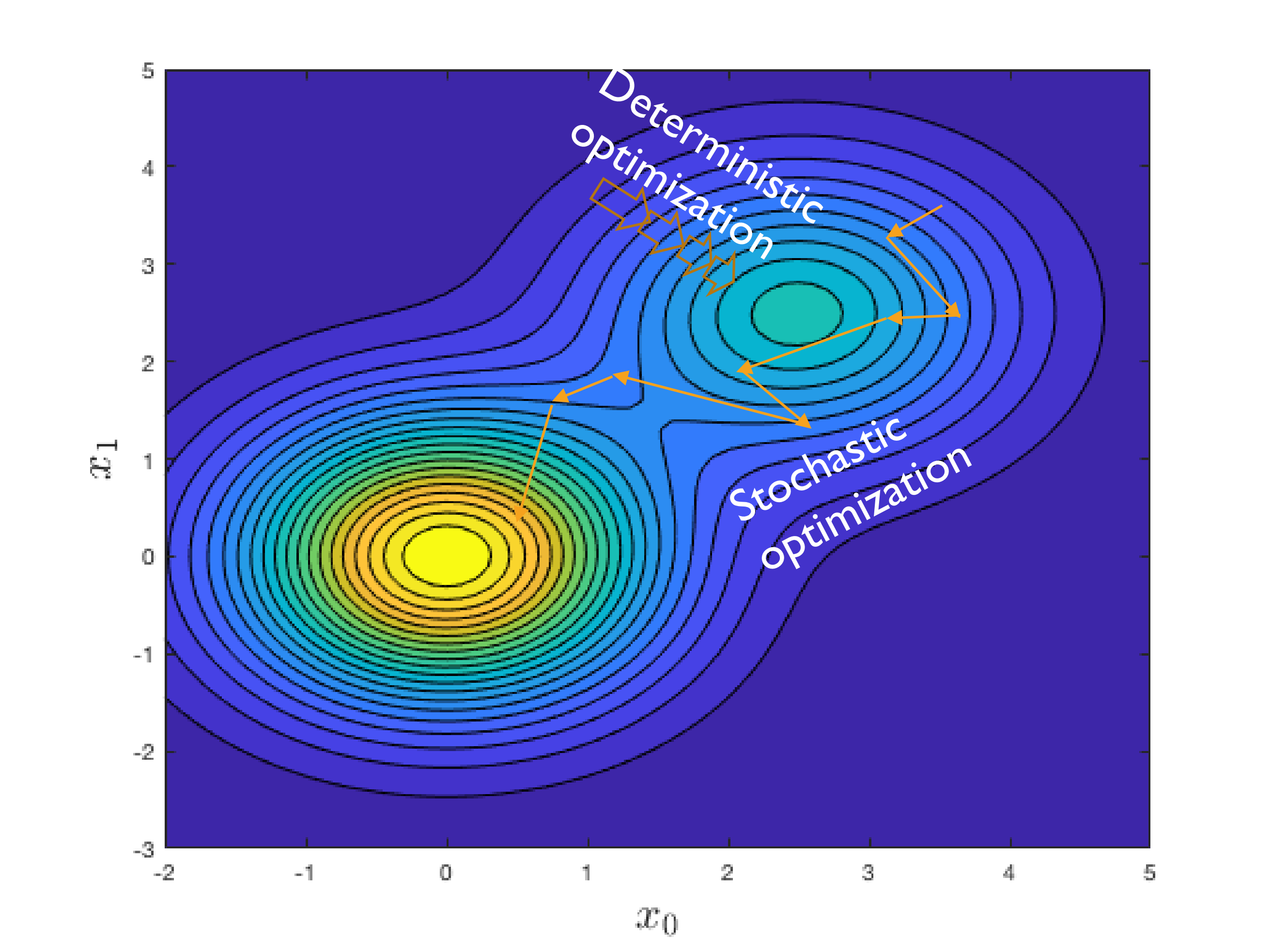
Image: https://user mu.edu/~yuejiec/research.html



REVIEW

STOCHASTIC OPTIMIZATION

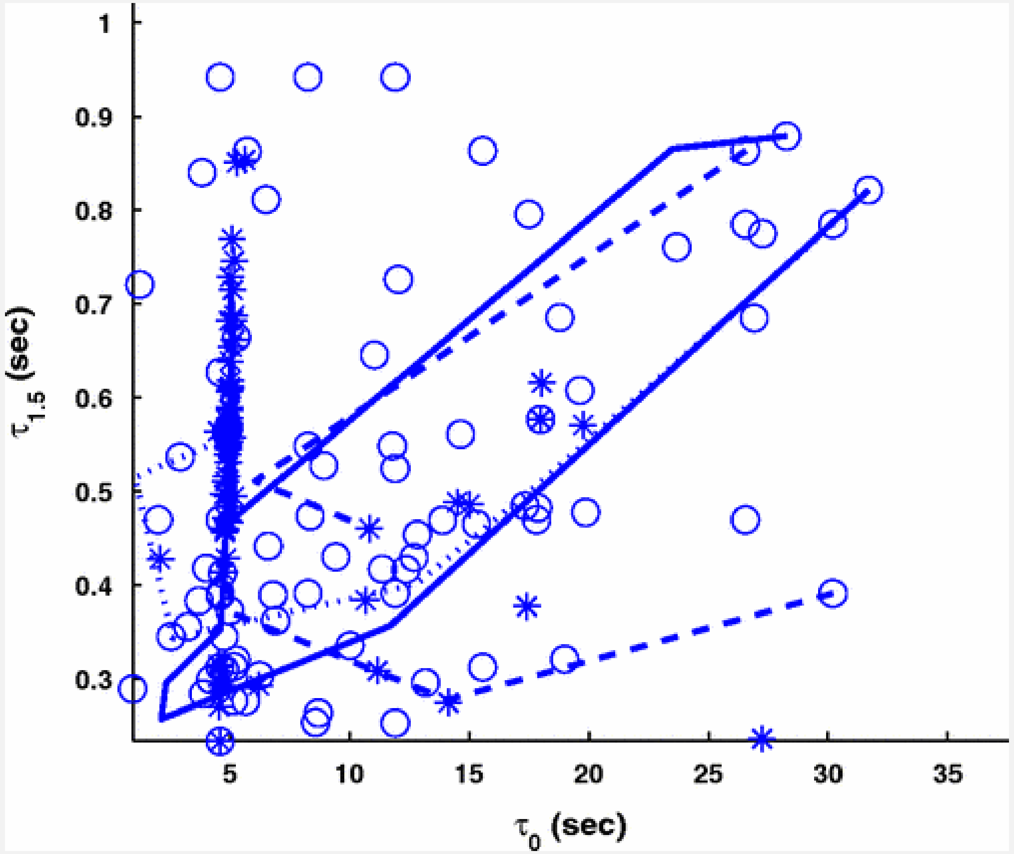
- Escape local minimum by random moves in search space
- Avoid completely random motion by imposing some goal or guidance



EXPLORATION AND EXPLOITATION

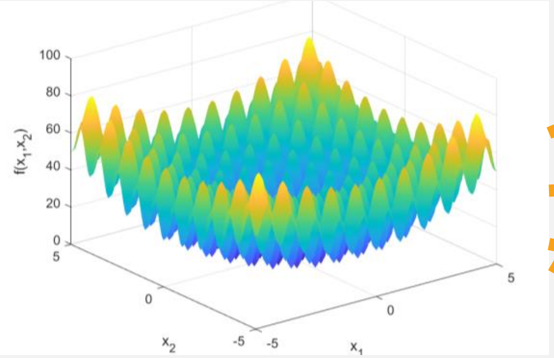
- The opposite pulls of the sufficiency conditions ⇒ practical stochastic optimization methods have two phases
- Exploration: explore the search space
- Exploitation: Identify a promising region and focus on improving the fitness

❖ Fitness must improve in both phases

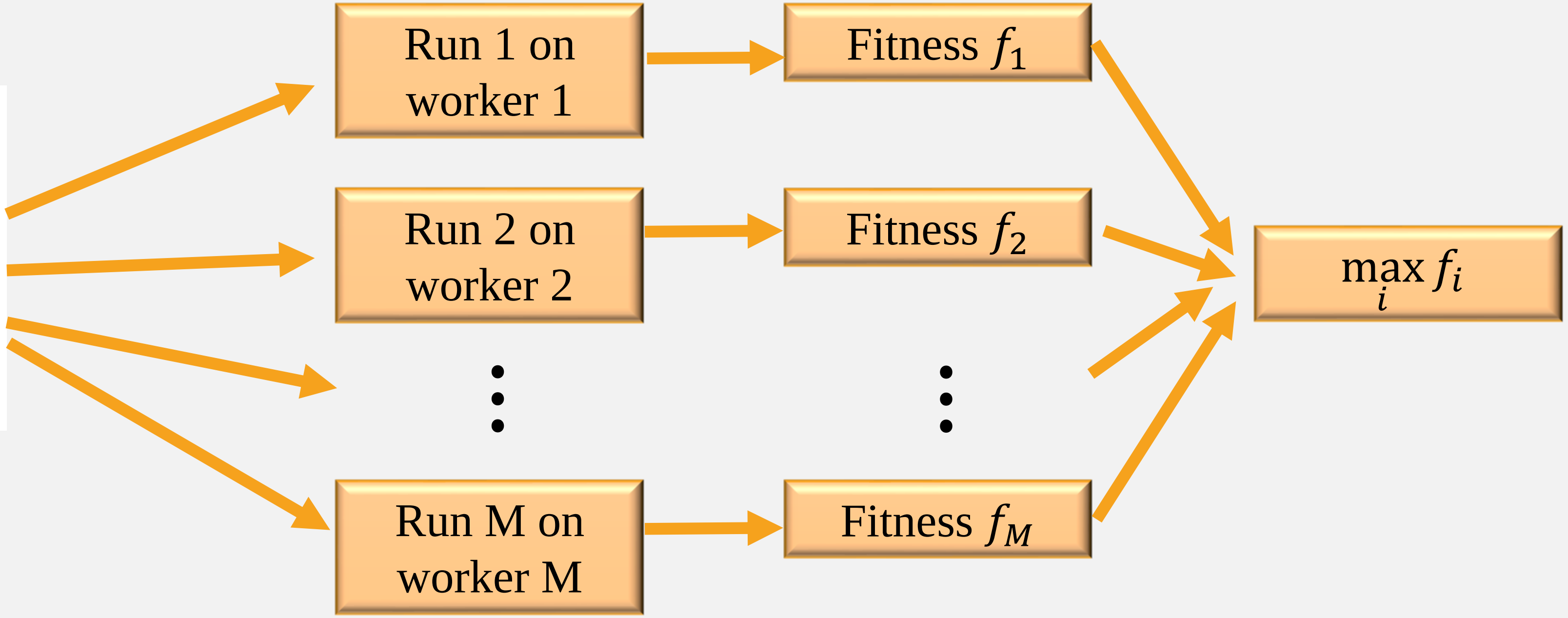


Lecture 1: From Wang, Mohanty, Physical Review D (2010)

TUNING: BMR STRATEGY



Fitness function



PSO DYNAMICAL EQUATIONS

Velocity update

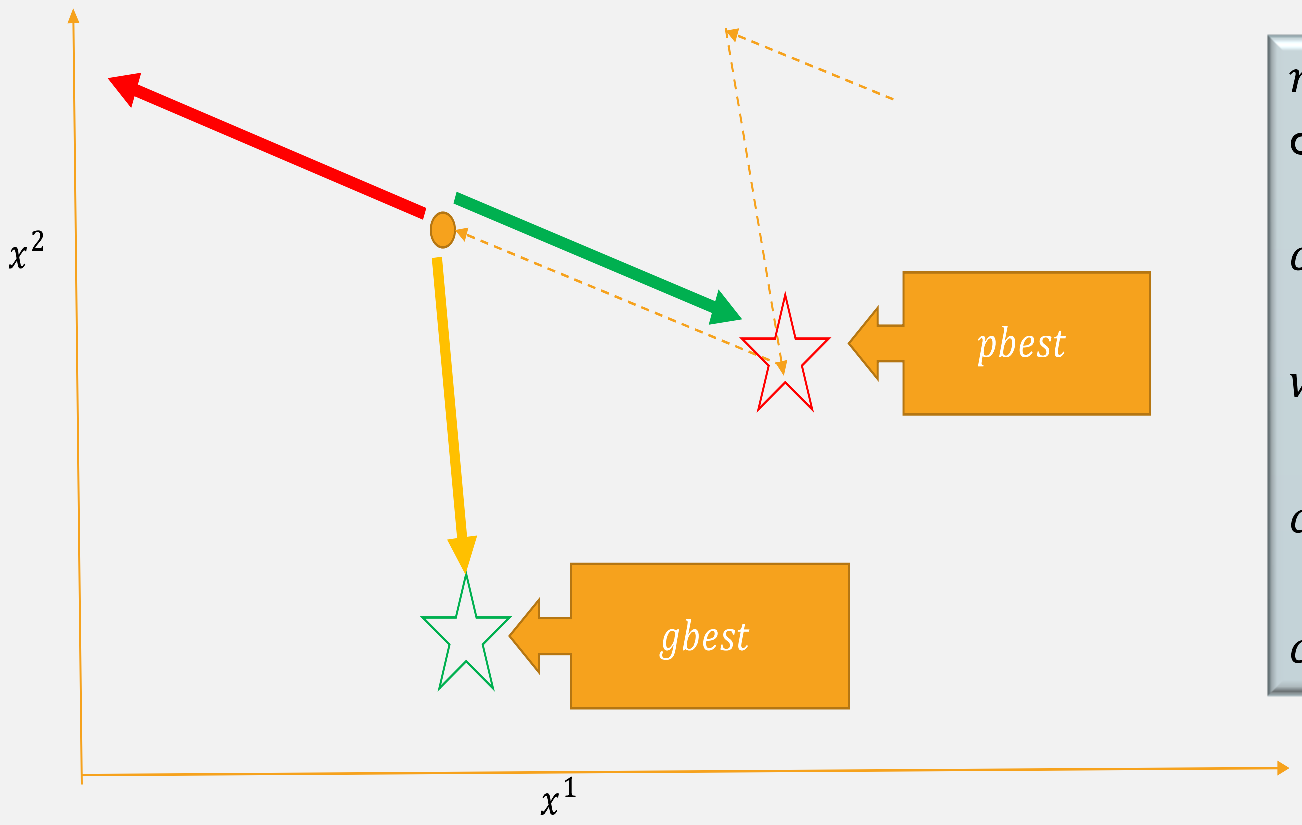
$$v_j^{(i)}[k+1] = w\, v_j^{(i)}[k] + c_1 r_{1,j}(p_j^{(i)}[k] - x_j^{(i)}[k]) + c_2 r_{2,j}(g_j\ [k] - x_j^{(i)}[k])$$

Position update

$$x_j^{(i)}[k+1] = x_j^{(i)}[k] + v_j^{(i)}[k+1]$$

VELOCITY UPDATE

$$v_j^{(i)}[k+1] = w\, v_j^{(i)}[k] + c_1 r_{1,j}(p_j^{(i)}[k] - x_j^{(i)}[k]) + c_2 r_{2,j}(g_j\,[k] - x_j^{(i)}[k])$$



$r_{m,j}$: random variable with uniform distribution in $[0,1]$

c_1, c_2 :"**acceleration constants**"

w : "inertia" $\rightarrow w\, v_j^{(i)}[k]$:"**Inertia Term**"

$c_1 r_{1,j}(p_i^j[k] - x_i^j[k])$:"**Cognitive term**"

$c_2 r_{2,j}(g[k] - x_i^j[k])$:"**Social term**"

LECTURE 3 OUTLINE

Particle Swarm Optimization

- Important variations under the PSO metaheuristic
- Local-best PSO
- Recommended settings

Applications of PSO

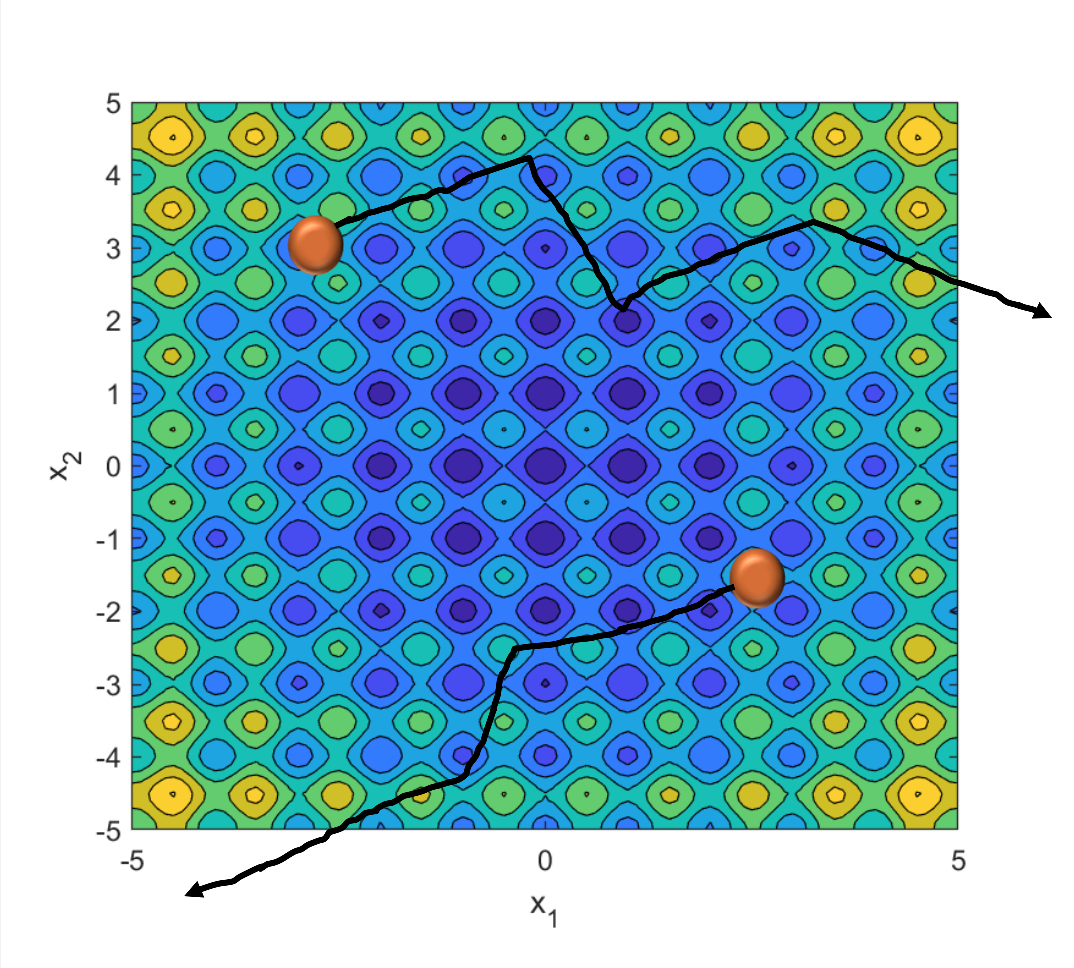
- Parametric regression (quadratic chirp)
- Non-parametric regression (regression spline)
- PSO tuning

PSO VARIANTS

Not a comprehensive review!

PARTICLE EXPLOSION

- Early PSO algorithms did not restrict the particle velocity (e.g., clamping)
- These versions suffered from "particle explosion": The swarm would quickly move out of the search space
- Particles moving outside the search space are useless to the search



PARTICLE EXPLOSION

- Velocity clamping was introduced to suppress particle explosion

$$v_j^{(i)}[k] \in [-v_{max}, v_{max}]$$

- Inertia was introduced as a replacement for clamping

$$w < 1 \Rightarrow v_j^{(i)}[k+1] = w\, v_j^{(i)}[k] < v_j^{(i)}[k]$$

- Generous velocity clamping still recommended to prevent too many particles leaking out of the search space

VELOCITY CONSTRICTION

- Velocity constriction is another way to contain a particle explosion

$$v_j^{(i)}[k+1] = K\left[v_j^{(i)}[k] + c_1 r_{1,j}\left(p_j^{(i)}[k] - x_j^{(i)}[k]\right) + c_2 r_{2,j}\left(g_j[k] - x_j^{(i)}[k]\right)\right]$$

- K is called the constriction factor

$$K = \frac{2}{|2 - c - \sqrt{c^2 - 4c}|};$$
$$c = c_1 + c_2 > 4$$

- Standard choice for K is 0.729 corresponding to $c = 4.01$
- Normally, $c_1 = c_2 \Rightarrow 2.05$
- Even without velocity constriction, $c_1 = c_2 \simeq 2$ is widely adopted in the literature

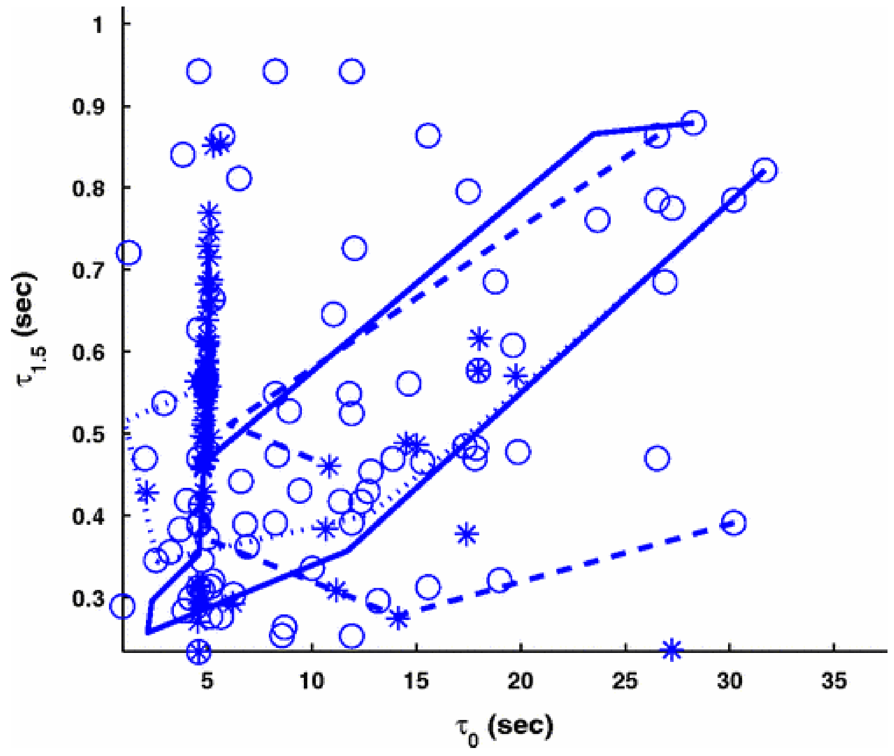
PSO: EXPLORATION AND EXPLOITATION

$$v_j^{(i)}[k+1] = w\, v_j^{(i)}[k] + \text{ (Inertia)}$$

$$c_1 r_{1,j}\left(p_j^{(i)}[k] - x_j^{(i)}[k]\right) + \text{(Cognitive)}$$

$$c_2 r_{2,j}(g_j\,[k] - x_j^{(i)}[k]) \text{ (Social)}$$

Exploration	Exploitation
• Inertia term: particle flies past local minima • Randomization: acceleration terms are not deterministic	• Cognitive force: attractive • Social force: attractive



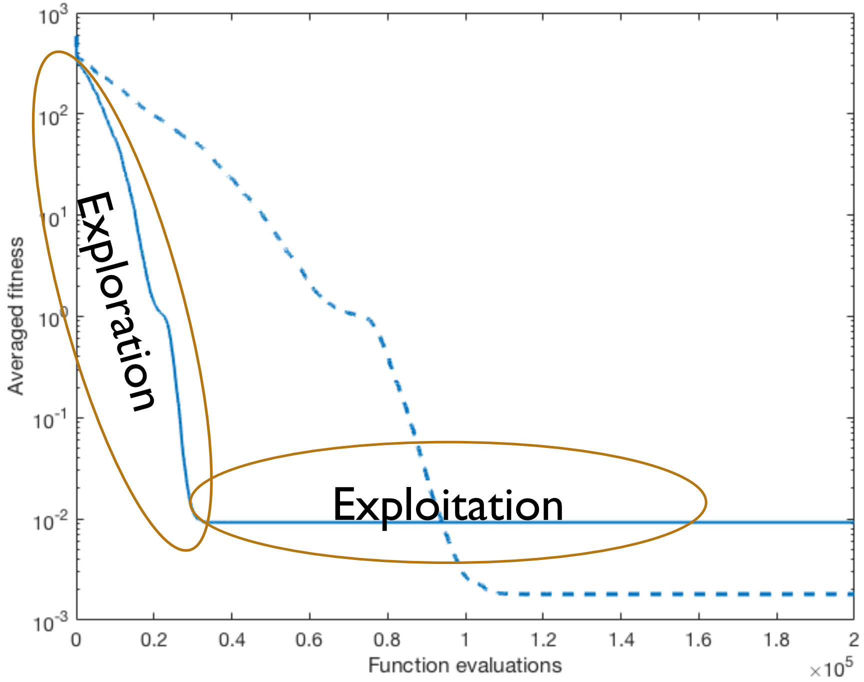
INERTIA DECAY

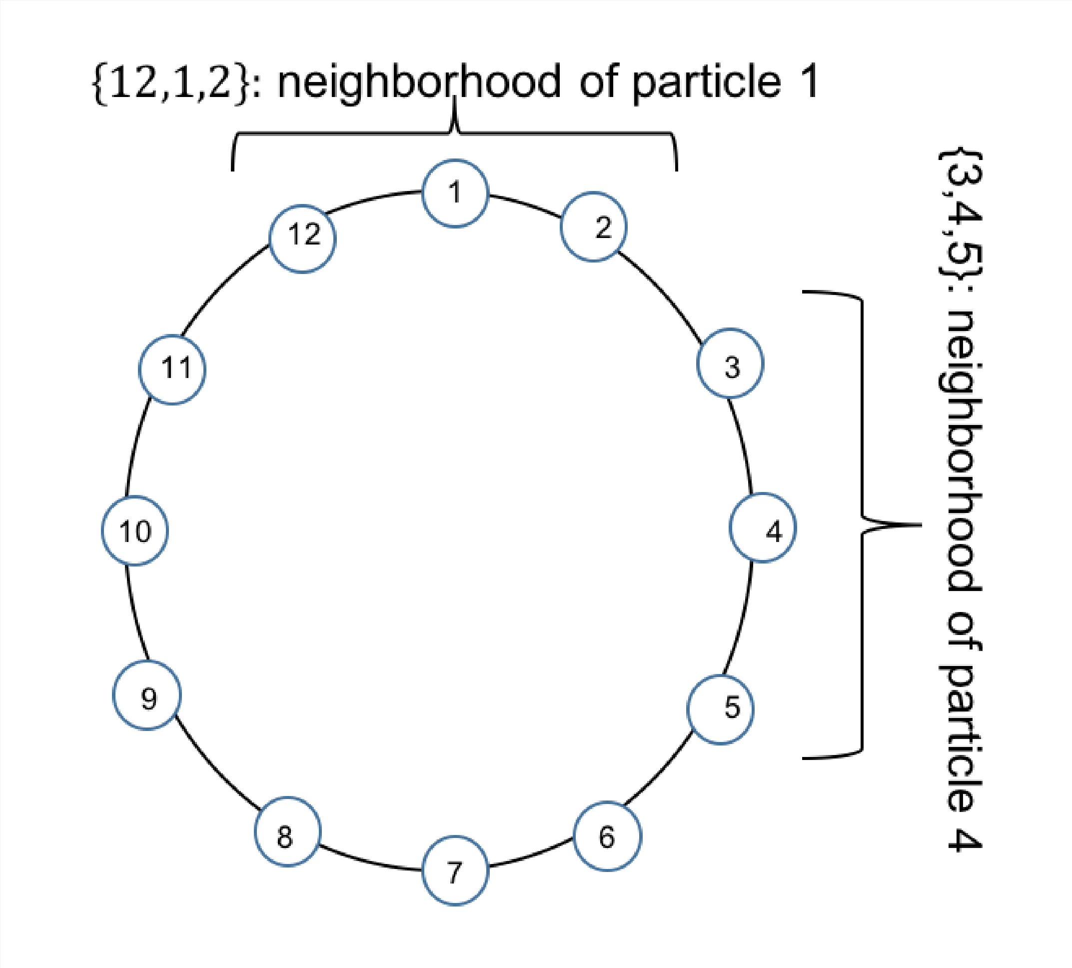
- For termination after a fixed number, N_{iter}, of iterations

<u>Linear decay</u>

$$w \rightarrow w[k] = 0.9 - 0.5\frac{k-1}{N_{iter} - 1}$$

- Transition from exploration to exploitation behavior
- Other laws of inertia decay have been proposed





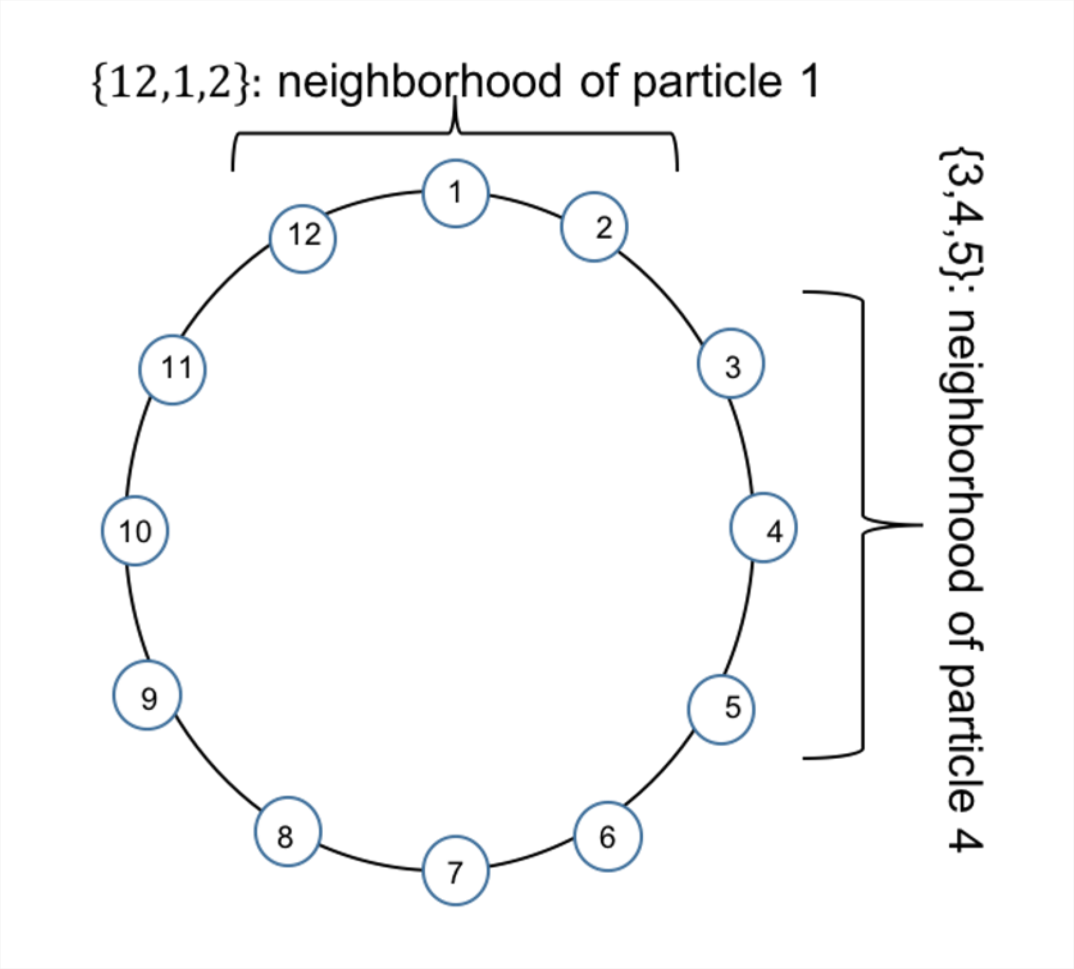
COMMUNICATION TOPOLOGY

$$v_j^{(i)}[k+1] = w[k]v_j^{(i)}[k] + c_1 r_{1,j}\left(p_j^{(i)}[k] - x_j^{(i)}[k]\right) + c_2 r_{2,j}(g_j[k] - x_j^{(i)}[k])$$

<u>Local best PSO</u>

$\bar{g}[k] \rightarrow \bar{l}^{(i)}[k]$: best value in a neighborhood of the i^{th} particle

$lbest: \bar{l}^{(i)}[k]$



$lbest$ PSO

Local best PSO

$$\bar{g}[k] \to \bar{l}^{(i)}[k]$$

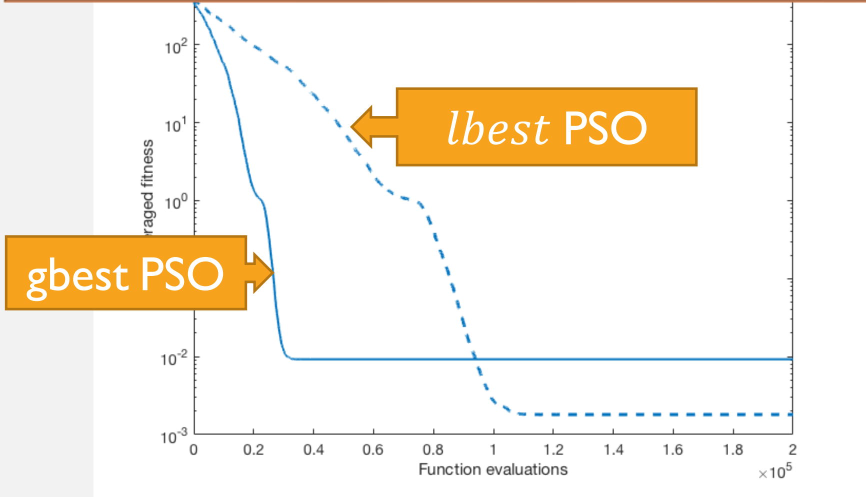
- Information about global best (e.g., particle #5) shared through common particles

$$\ldots, (1, 2, 3), (2, 3, 4)$$

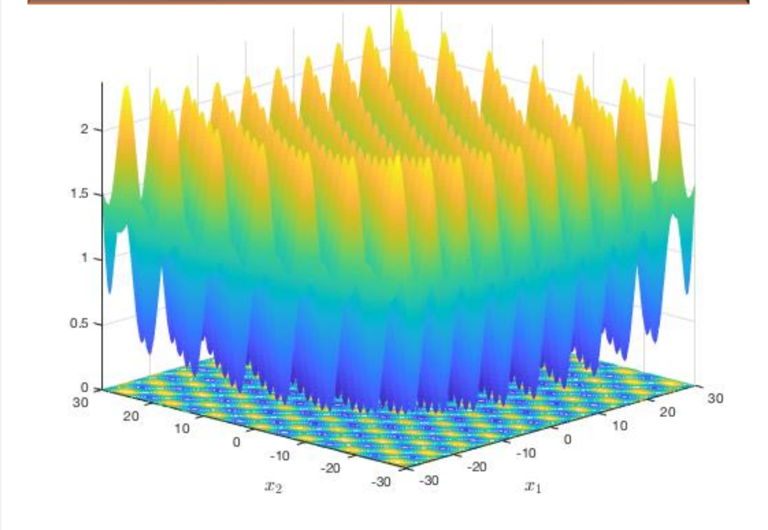
$$(2, 3, 4), (3, 4, 5), \ldots$$

- Information about global best propagates more slowly through the swarm
- Less social attraction: extended exploration

30D Generalized Griewank; $x_i \in [-600,600]$



2D Generalized Griewank



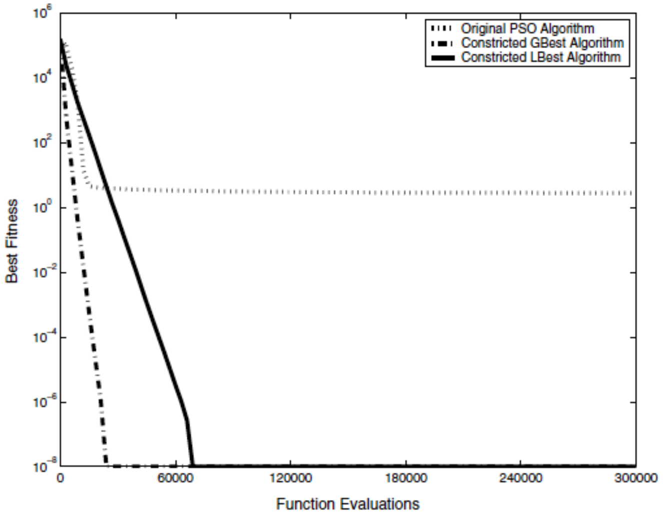
EXPLORATION AND EXPLOITATION IN *lbest* PSO

- The better performance of *lbest* PSO shows the importance of controlling exploration vs exploitation
- Trade off between convergence and number of iterations: *lbest* PSO is computationally more expensive than *gbest* PSO

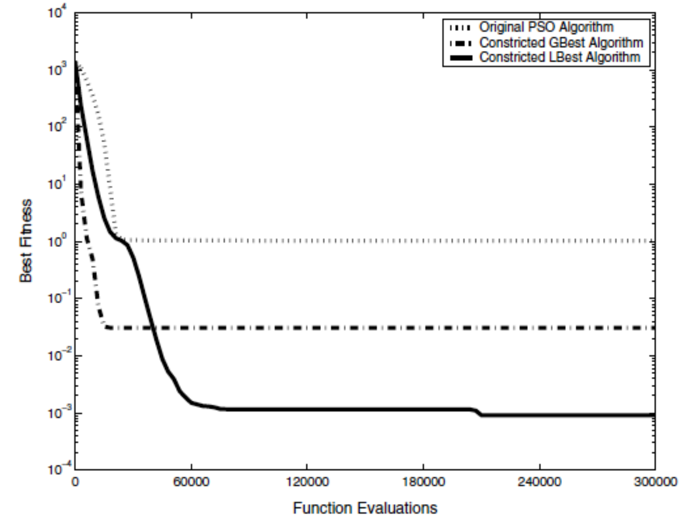
LBEST VS. GBEST PSO

- Tends to become better than gbest PSO as:
  - the dimensionality increases and/or
  - fitness function becomes more rugged
- Penalty: More fitness function evaluations

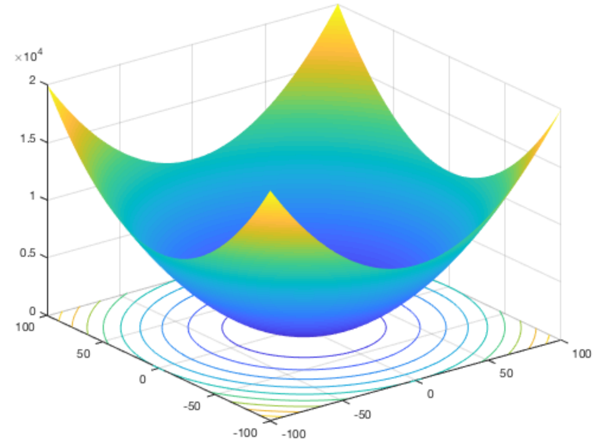
From Bratton & Kennedy, 2007

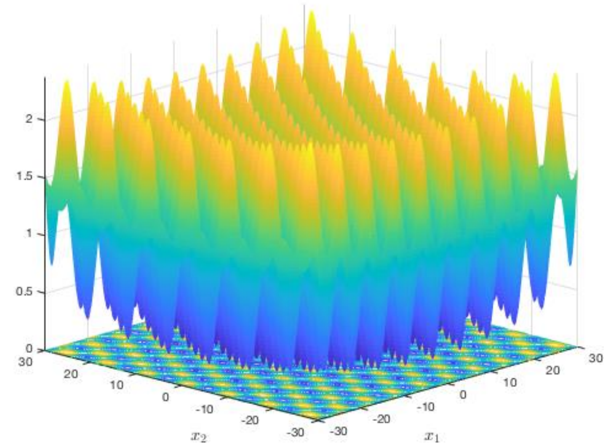


(a) f01 (Sphere/Parabola)



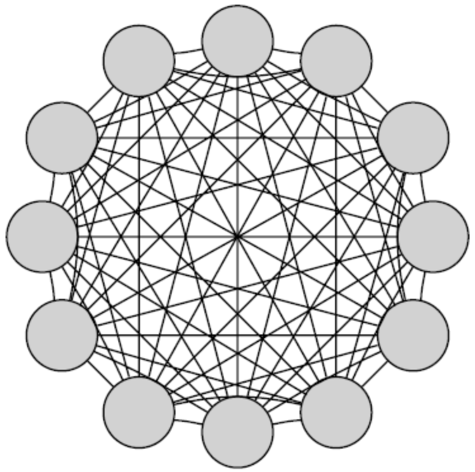
(d) f07 (Generalized Griewank)



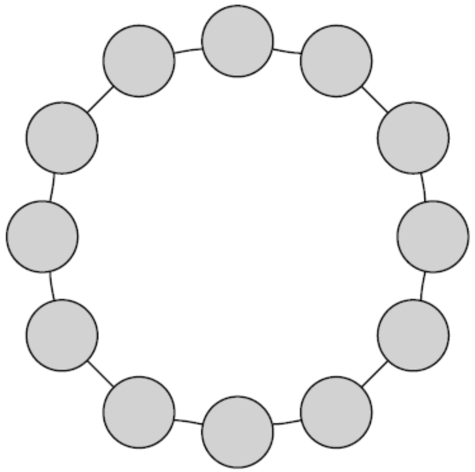


TOPOLOGIES

- Gbest
- Ring
- Star
- Wheel
- Pyramid
- Four Clusters
- Von-Neumann
- …



(a) The gbest topology



(b) The lbest ring topology

TERMINATION (VARIATIONS)

Stop when maximum number of iterations reached

Stop when an acceptable solution has been found

- If x^* is known (e.g., benchmark functions), stop when $|f(x_k) - f(x^*)| < \epsilon$
- Stop if the average change in particle positions is small
- Stop if the average velocity over a number of iterations is close to zero
- Stop if there is no significant improvement in the fitness value over a number of iterations

Stop when the normalized swarm radius is close to zero

- $R_{norm} = \frac{R_{max}}{\text{initial } R_{max}}$; $R_{max} = \max_i \left\| \bar{x}^{(i)}[k] - \bar{g}[k] \right\|$

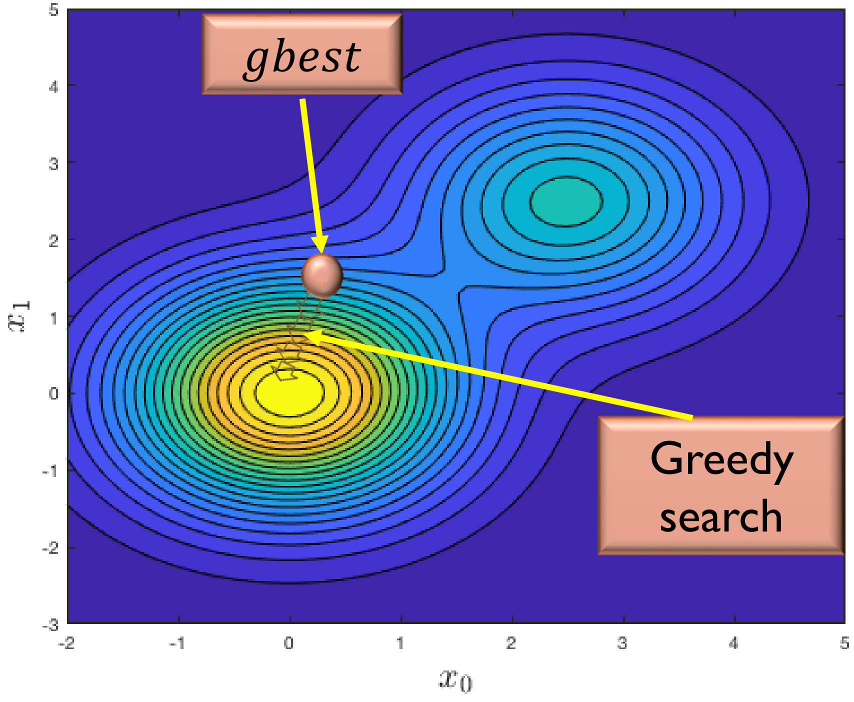
Wang, Mohanty, Physical Review D, 2010

- Stop when the best particle does not move out of a small box over a specified number of iterations

MEMETIC SEARCH

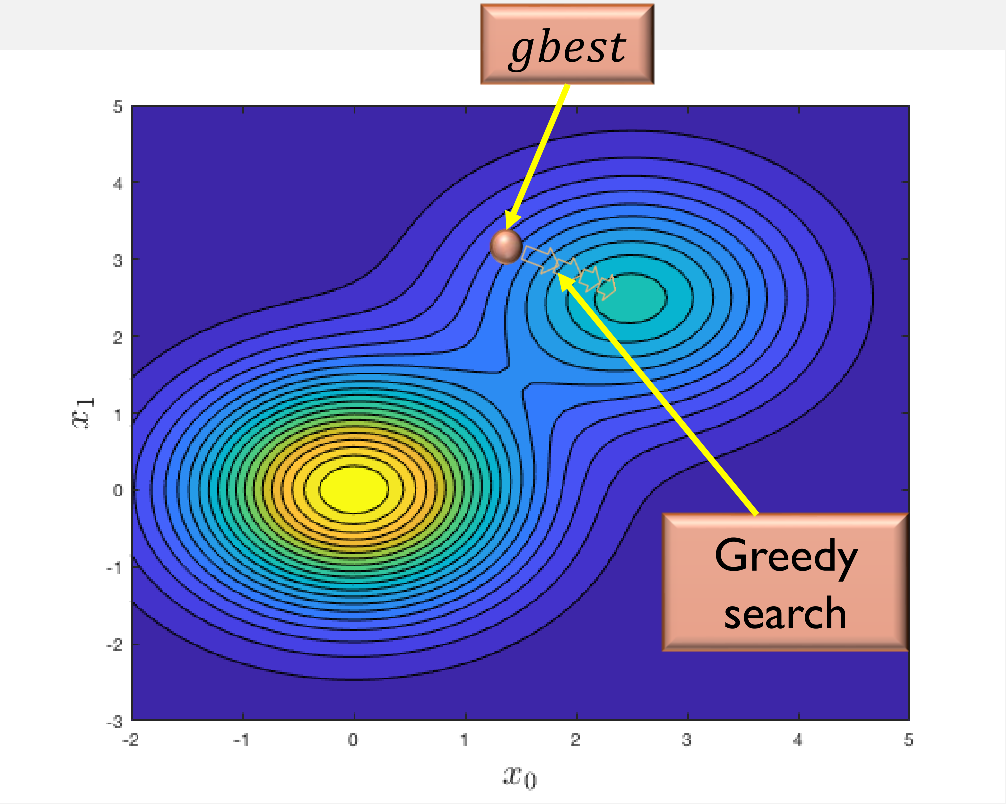
- PSO converges slowly at late stages
- Local optimizers, e.g., steepest descent, can converge to a local minimum much faster

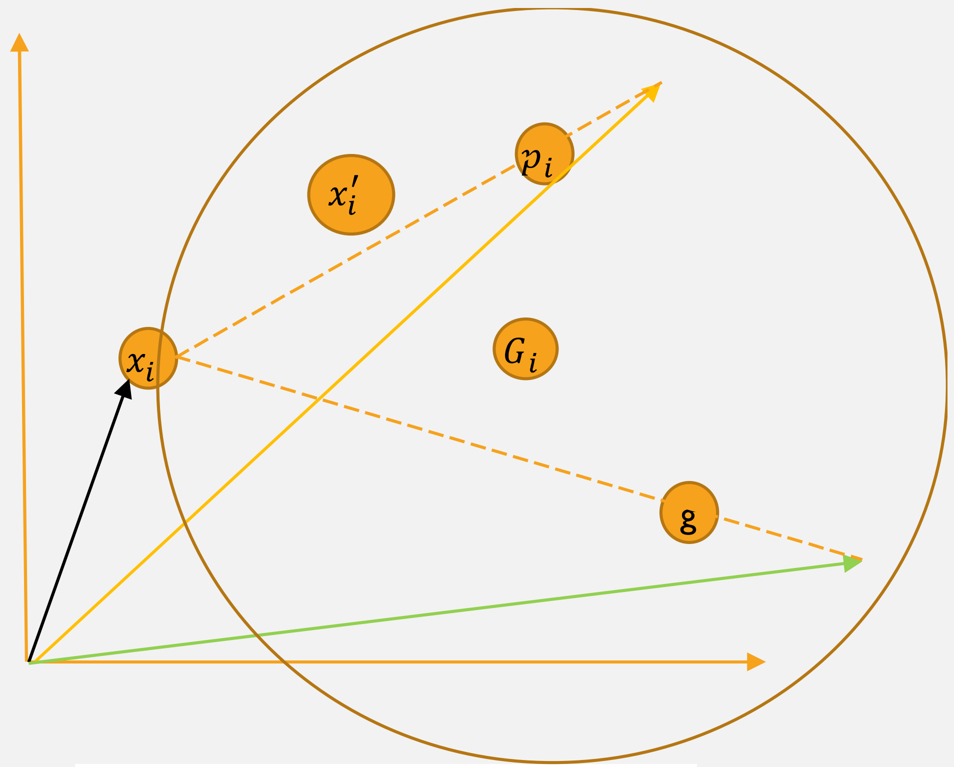
1. Use local search in each iteration to refine $gbest$ or $lbest$, or …
2. Use stochastic local search in a neighborhood of *gbest* or *lbest*

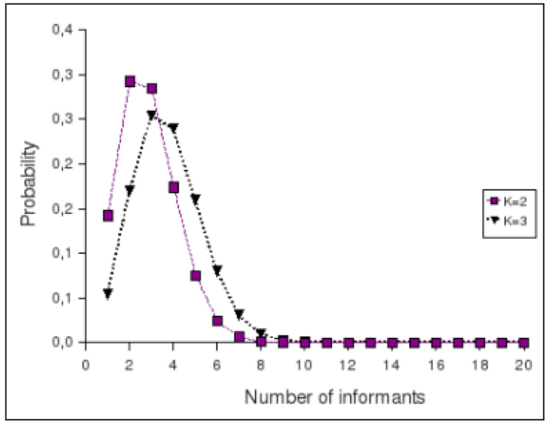


MEMETIC SEARCH

- A danger is finding a good local minimum too early
- $gbest$ stays locked to this local minimum and attracts all the particles
- This shortens the exploration phase and increases the chances of missing the global minimum







3.2: Adaptive random topology. Distribution of the number of informants of a particle.

STANDARD PSO (SPSO)

- SPSO ‘06, ‘07, ‘11: http://clerc.maurice.free.fr/pso/SPSO_descriptions.pdf
- Velocity update ($\bar{x}^{(i)} \rightarrow x_i$ here):

$$G_i = \frac{1}{3}\left(x_i + \left(x_i + c(p_i - x_i)\right) + \left(x_i + c(g - x_i)\right)\right)$$

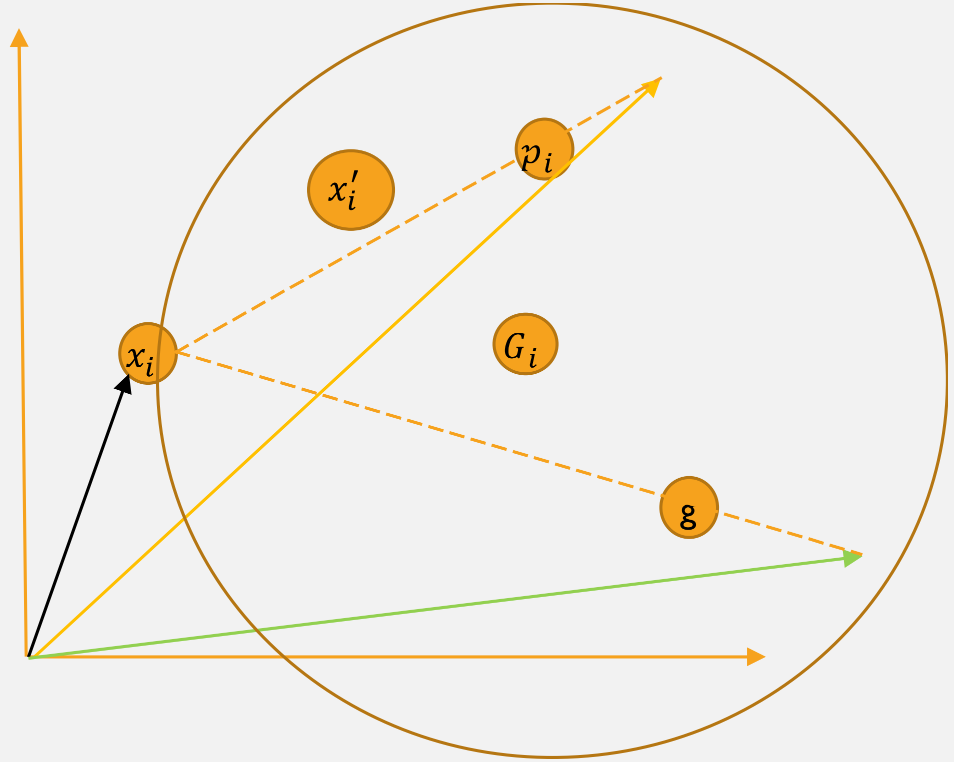
  - x_i': Point picked **randomly** in the sphere centered on G_i with radius $\|G_i - x_i\|$

$$v_i[k+1] = w\, v_i[k] + (x_i' - x_i)$$

- Reflecting inelastic walls:

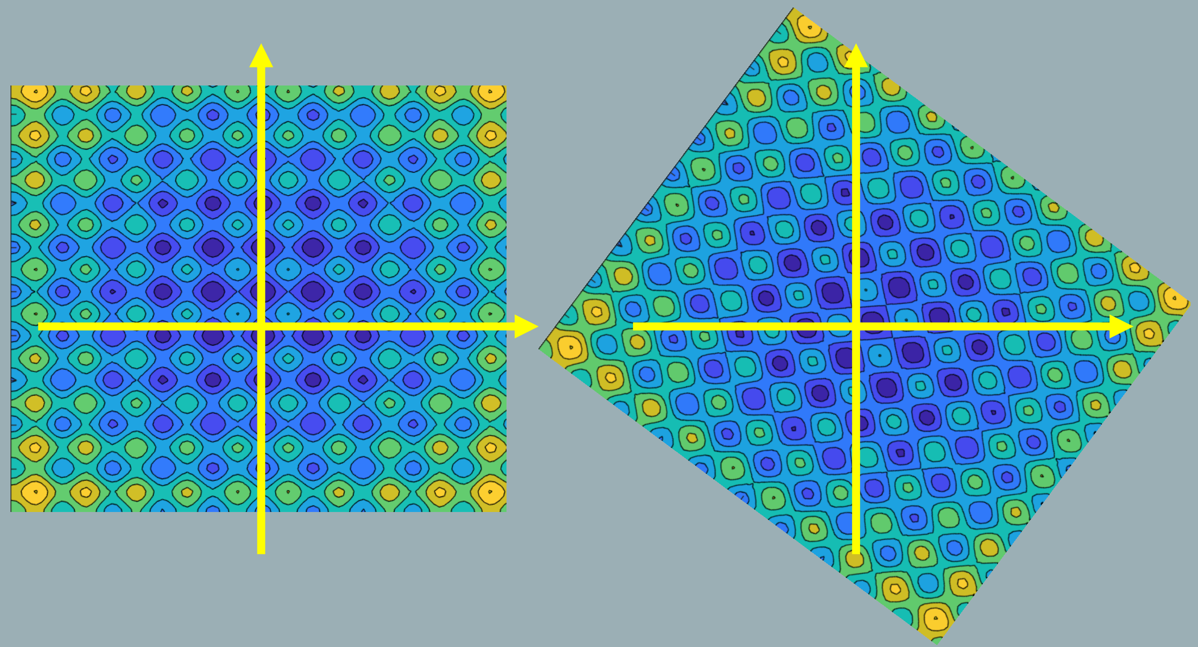
$$v_i[\mathrm{k}+1] = -0.5\, v_i[k+1]$$

- Neighborhood sizes picked randomly at each iteration



STANDARD PSO (SPSO)

The goal is SPSO dynamical equations is to maintain PSO performance under rotation of the fitness function



RECOMMENDED PSO PARAMETER SETTINGS

- Follows Bratton and Kennedy, 2007
- Optimum particle number (N_{part}) ?
  - Too few $\Rightarrow$ Less exploration
  - Too many $\Rightarrow$ Premature convergence
- *lbest* PSO with ring topology (2 nearest neighbors)
  - Increases exploration
  - Slower convergence but often better probability of success

Setting Name	Setting Value
Position initialization	$x_j^{(i)}[0]$ drawn from $U(x;0,1)$
Velocity initialization	$v_j^{(i)}[0]$ drawn from $U(x;0,1) - x_j^{(i)}[0]$
$v_{\max}$	0.5
N_{part}	40
$c_1 = c_2$	2.0
$w[k]$	Linear decay from 0.9 to 0.4
Boundary condition	Let them fly
Termination condition	Fixed number of iterations
lbest PSO	Ring topology; Neighborhood size = 3

PSO APPLICATIONS

Parametric and non-parametric regression

PARAMETRTIC REGRESSION

- Least-squares fit:

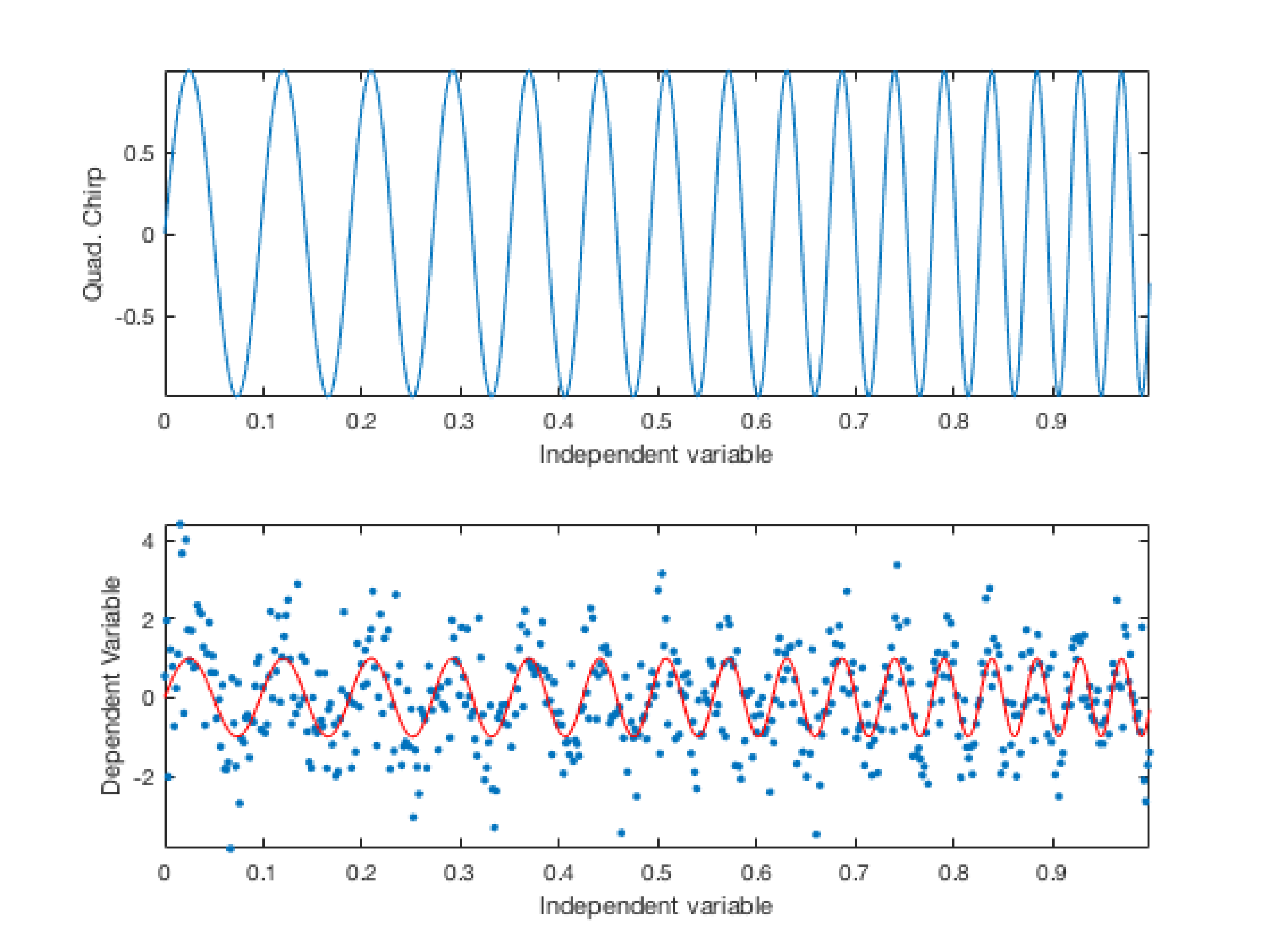
$$\min_{\bar{\theta}} \sum_{i=0}^{N-1} \left(y_i - f(x_i;\bar{\theta})\right)^2$$

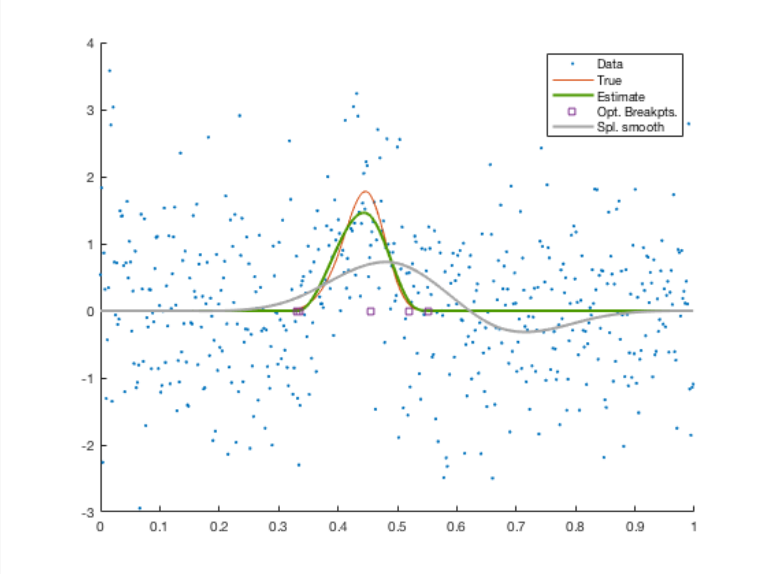
- Non-linear model:

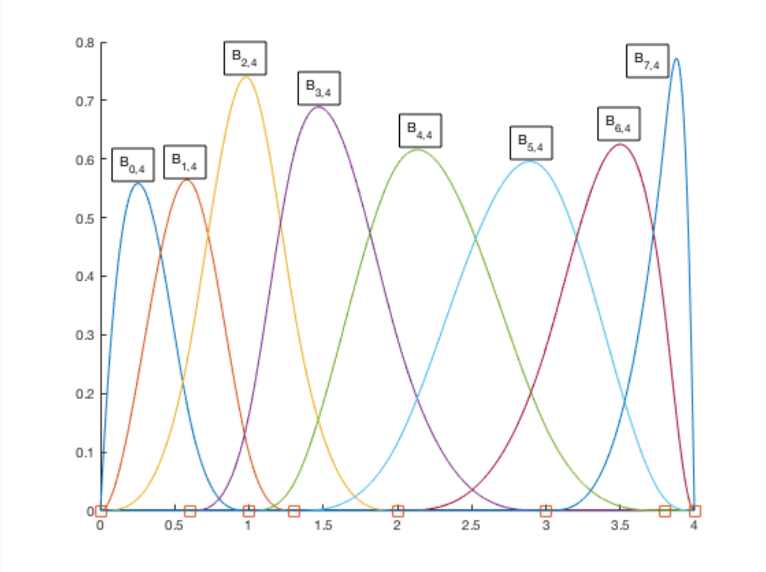
  Quadratic chirp (*Lecture 1)

$$f\left(x;\bar{\theta}\right) = A\sin(2\pi\Phi(x))$$

$$\Phi(x) = a_1 x + a_2 x^2 + a_3 x^3$$







NON-PARAMETRIC REGRESSION

- Regression spline (*Lecture 1)

$$f(x;\ \bar{\alpha},\bar{b}) = \sum_{j=0}^{M-1} \alpha_j B_{j,4}(x;\bar{b})$$

- Least-squares:

$$\min_{\bar{\alpha},\bar{b}} \sum_{i=0}^{N-1} \left(y_i - f(x_i;\ \bar{\alpha})\right)^2$$

- Fixed number (M) but not fixed locations of breakpoints $(\bar{b})$

STEP 1: ANALYTIC MINIMIZATION

$$\min_{\bar{\theta}} \sum_{i=0}^{N-1} \left(y_i - f(x_i; \bar{\theta})\right)^2$$

Parametric

$$f(x; \bar{\theta}) = A \sin(2\pi\Phi(x))$$

$$\Phi(x) = a_1 x + a_2 x^2 + a_3 x^3$$

- $\min_{(a_1,a_2,a_3)} \left(\min_A \sum_{i=0}^{N-1} (y_i - A\sin(2\pi\Phi(x)))^2\right)$
- A can be minimized analytically
- PSO handles the optimization over phase parameters $a_i, 1 \le i \le 3$

Non-parametric

$$f(x;\ \bar{\alpha}, \bar{b}) = \sum_{j=0}^{M-1} \alpha_j B_{j,4}(x; \bar{b})$$

- $\min_{\bar{b}} \left(\min_{\bar{\alpha}} \sum_{i=0}^{N-1} \left(y_i - \sum_{j=0}^{M-1} \alpha_j B_{j,4}(x; \bar{b})\right)^2\right)$
- Inner minimization can be done analytically
- PSO handles the optimization over the breakpoints $\bar{b}$

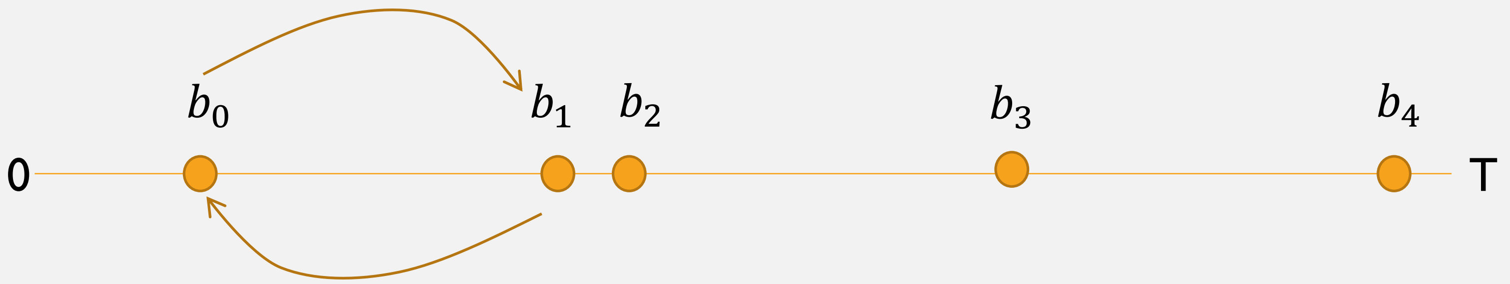
STEP 2: DEGENERACY CONTROL

Example: **Unconstrained** optimization over breakpoints

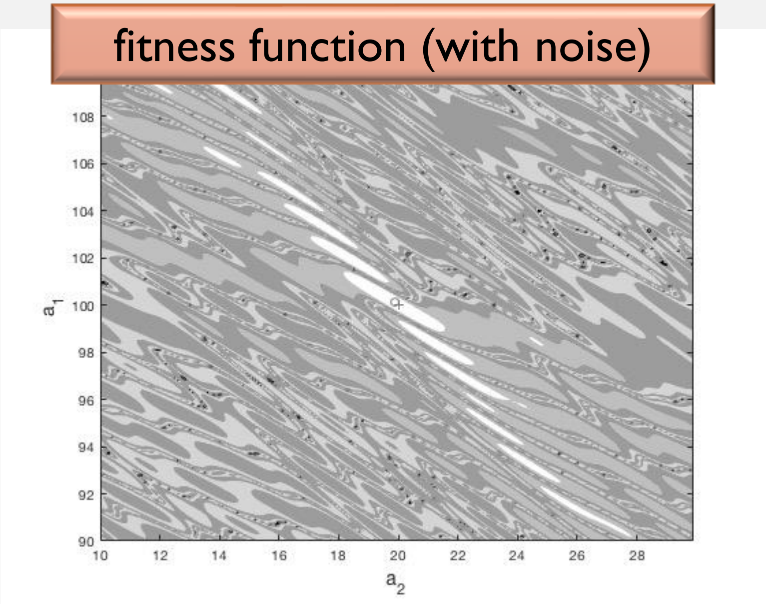
$$\bar{b} = (b_0, b_1, \ldots, b_{M-1})$$

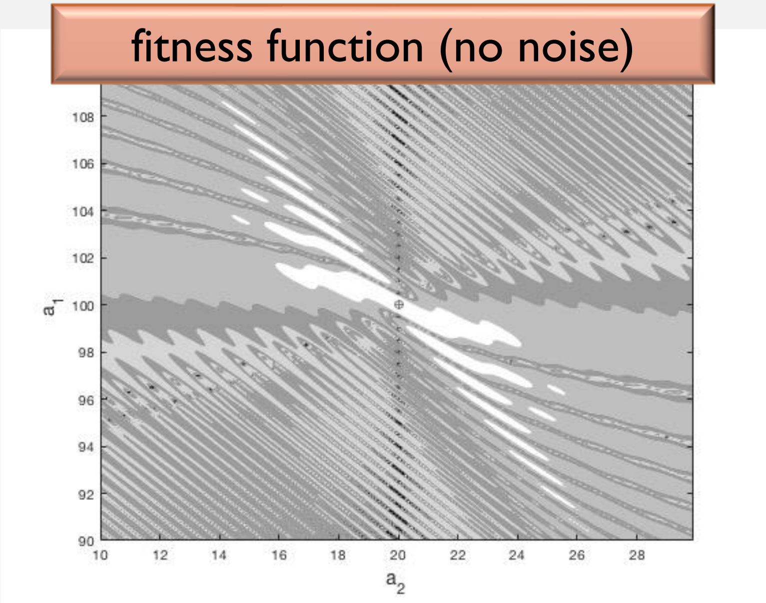
Search space: $b_i \in (0, T), \forall i$

- Permutations of $\bar{b}$ correspond to the same spline since the sequence must be ordered before the corresponding spline can be generated
- Degeneracy: Multiple widely-separated points have same fitness values



DEGENERACY IN PARAMETRIC REGRESSION

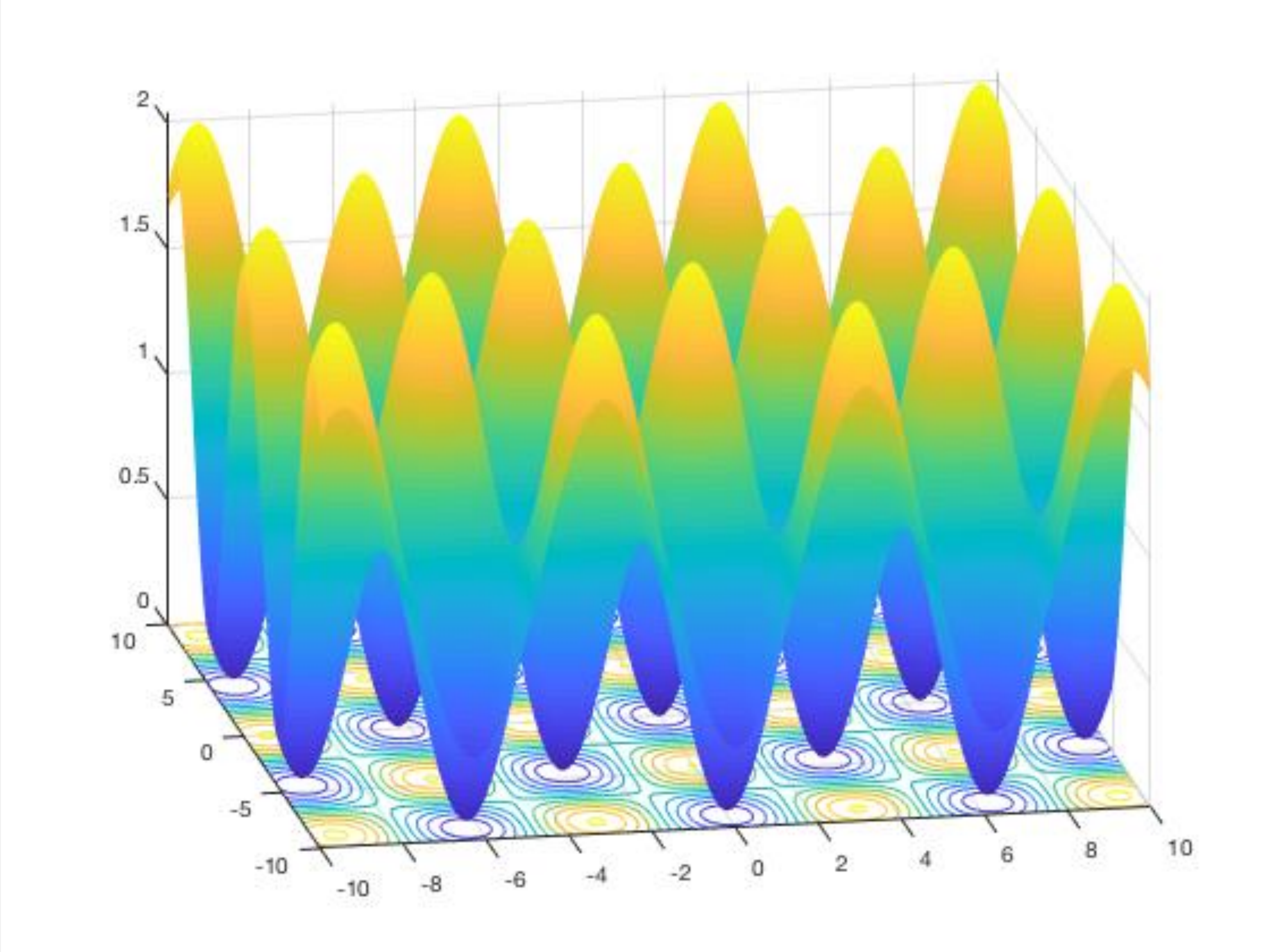




- Quadratic chirp: Multiple local minima in fitness function even in the absence of noise
- Multiple local minima are a hallmark of non-linear regression
- Note: Degeneracy is not restricted to equally deep minima

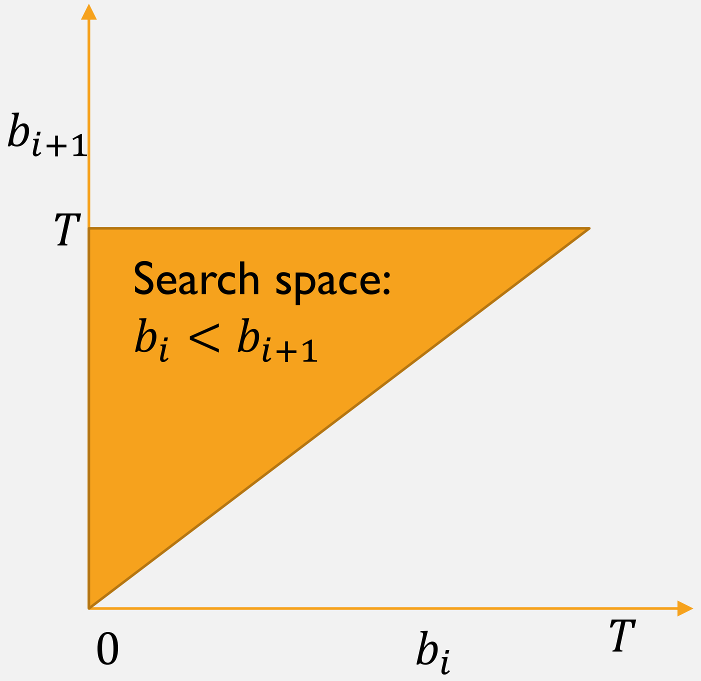
FITNESS FUNCTION DEGENERACY

- A stochastic optimization method must escape from local minima
- ⇒ Multiple local minima make the search for the global minimum harder
- Degeneracy of a fitness function (absence of noise) leads to multiple local minima



CONSTRAINED SEARCH

- 1st solution to degeneracy: Constrained search
$$b_i < b_{i+1}$$
- $\Rightarrow$ Search space shape is a simplex in M dimensions
- PSO does not perform well when the search space is not a hypercube
  - $\Rightarrow$ Excessive leakage of particles from the search space

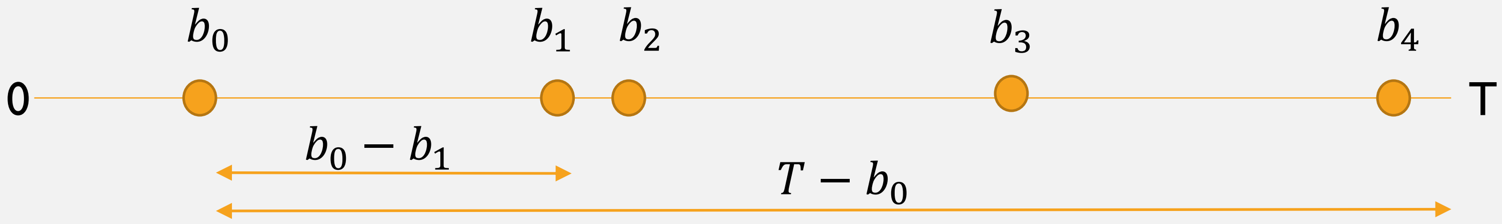


REPARAMETRIZATION

- 2nd solution: [Reparametrization]

$$\alpha_i = \frac{b_i - b_{i-1}}{T - b_{i-1}} \; ; \; \alpha_0 = \frac{b_0}{T}$$

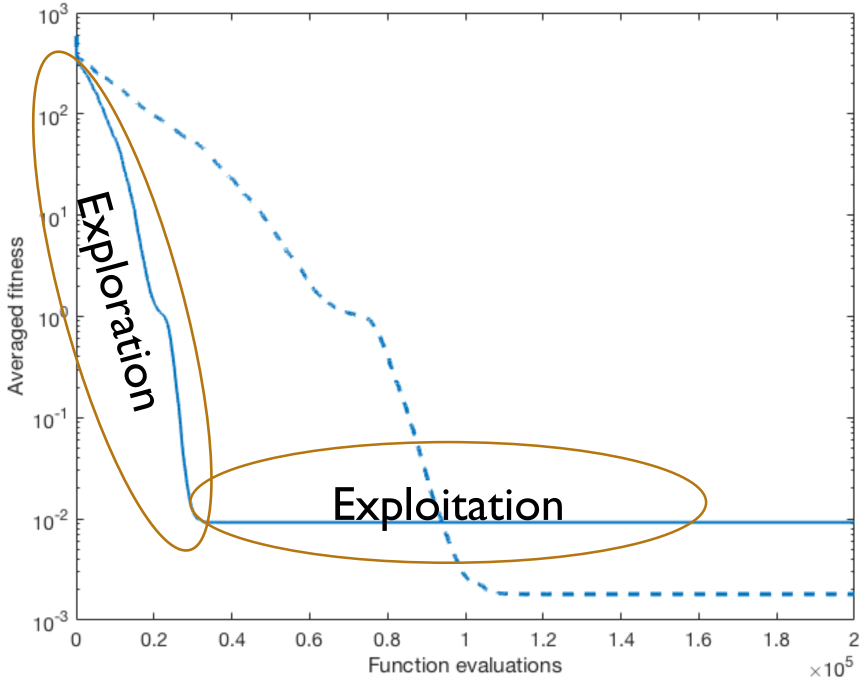
- Guarantees ordered breakpoint sequence while keeping the search space hypercubical: $\alpha_i \in (0,1); \forall i$

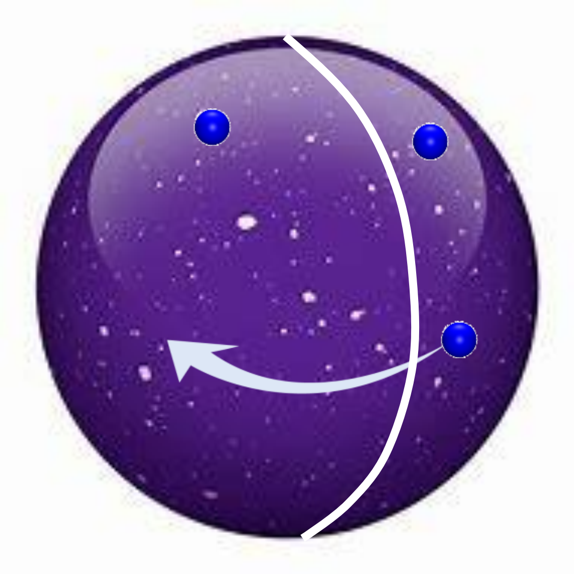


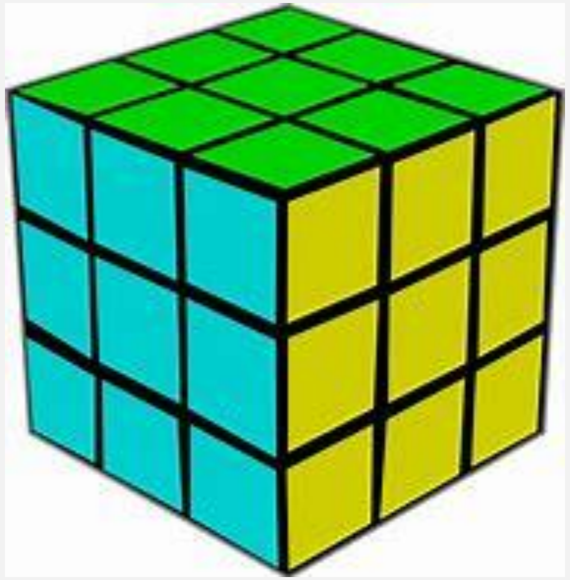
*Further reparametrization (see book): center the uniformly spaced breakpoint sequence

STEP 3: PSO VARIANT

- Exploration-exploitation trade-off
- Examine the extent of degeneracy to get an idea of which PSO variant to use
- Greater ruggedness of fitness function ⇒ Use longer exploration phase
  - Example: Use *lbest* PSO to extend exploration phase
  - *Other options such as slower inertia decay

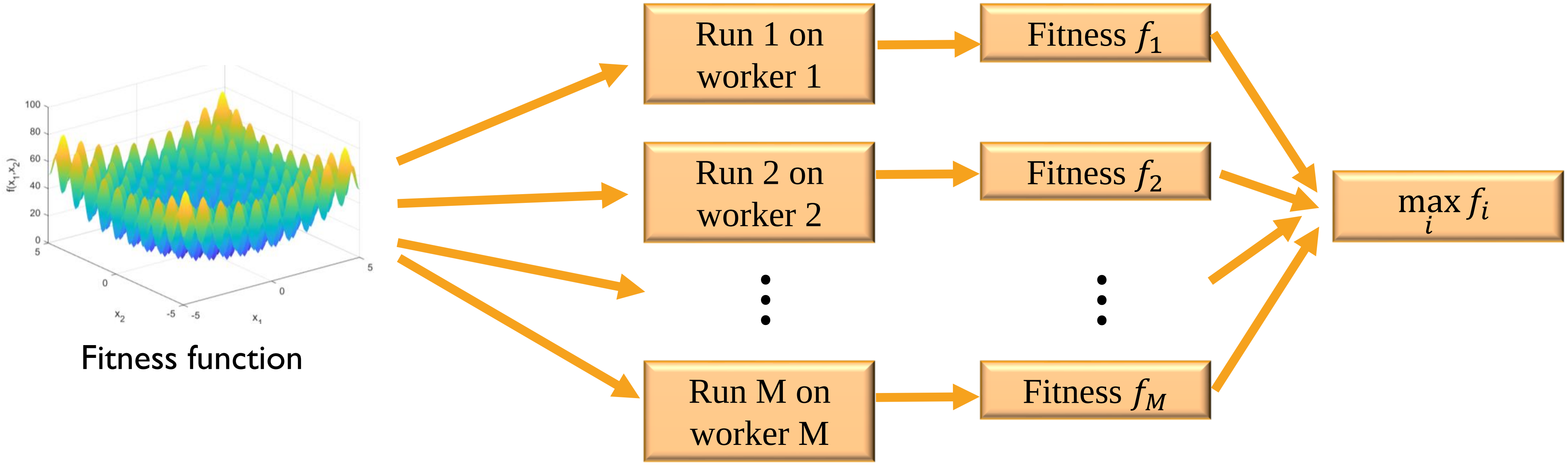






PSO VARIANTS

- If the fitness function has a periodic dependence on a variable, the corresponding boundary condition should be periodic
  - Very helpful in the case of gravitational wave searches because two of the parameters are sky angles
- Consider splitting the search space into smaller domains



STEP4: TUNING

Recommended: Best-of-M-runs (BMR)

STEP4: TUNING

- PSO parameters have robust values and do not need to be changed in most cases
- Two main parameters to tune
  - Number of iterations: N_{iter}
  - Number of independent PSO runs in BMR: N_{runs}

Setting Name	Setting Value
Position initialization	$x_j^{(i)}[0]$ drawn from $U(x;0,1)$
Velocity initialization	$v_j^{(i)}[0]$ drawn from $U(x;0,1) - x_j^{(i)}[0]$
$v_{\max}$	0.5
N_{part}	40
$c_1 = c_2$	2.0
$w[k]$	Linear decay from 0.9 to 0.4
Boundary condition	Let them fly
Termination condition	Fixed number of iterations
lbest PSO	Ring topology; Neighborhood size = 3

TUNING PSO FOR REGRESSION

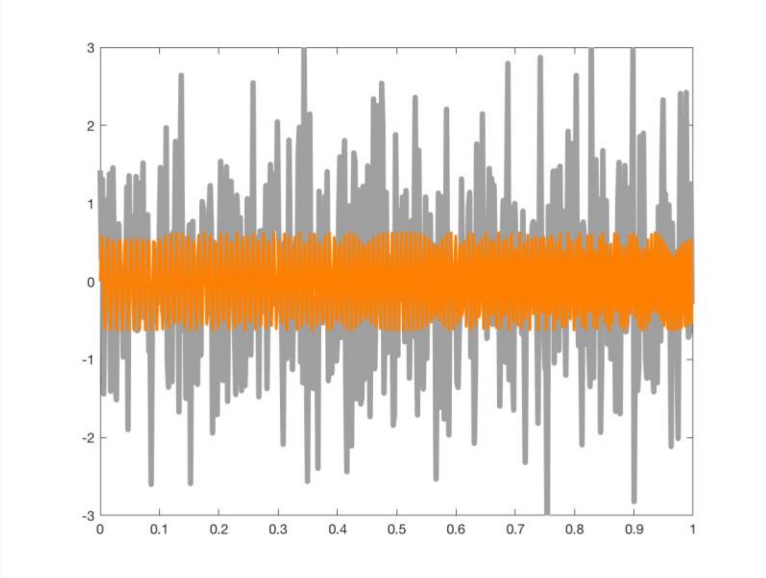
TUNING FOR REGRESSION PROBLEMS

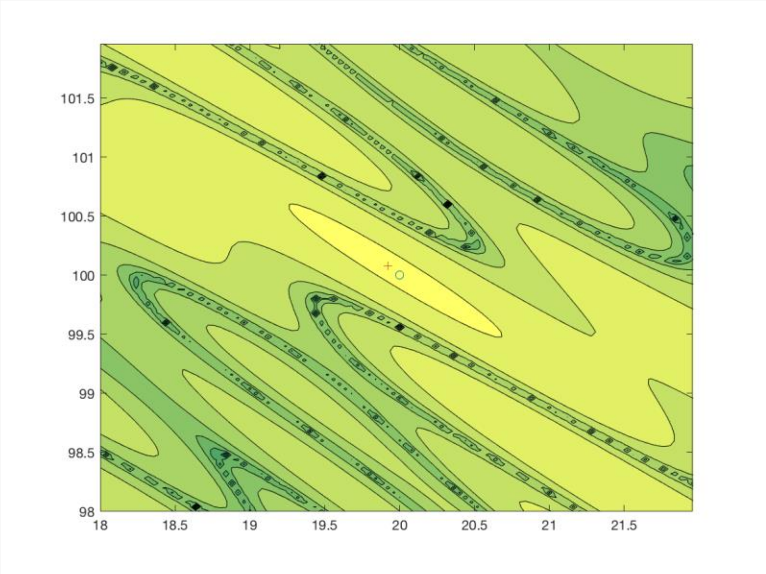
NFL ⇒ Over-tuning on one data realization ⇒
Worse performance on other realizations

Simulate data realizations based on assumed models

Cost function for each data realization as an independent fitness function

Use statistical metrics ⇒ Robustness across data realizations

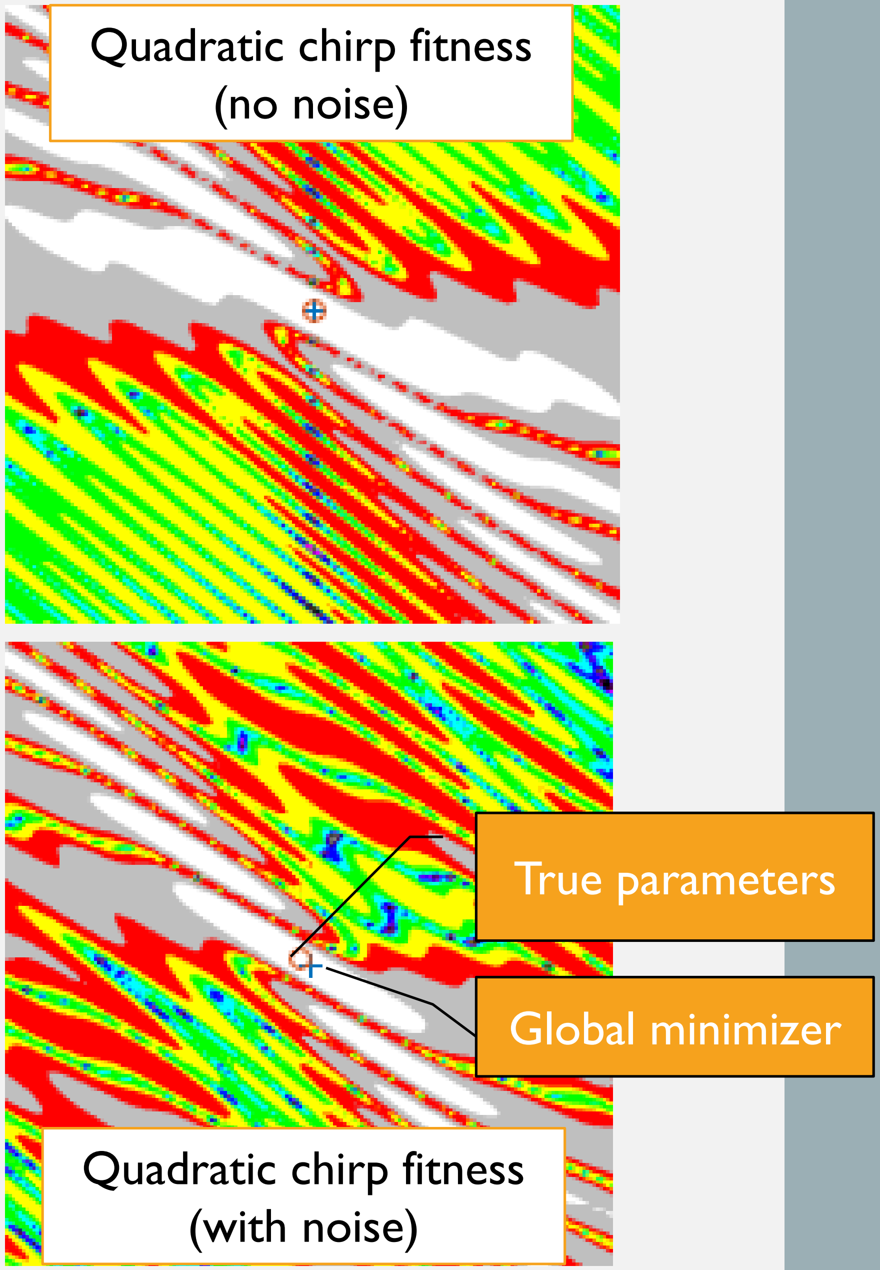




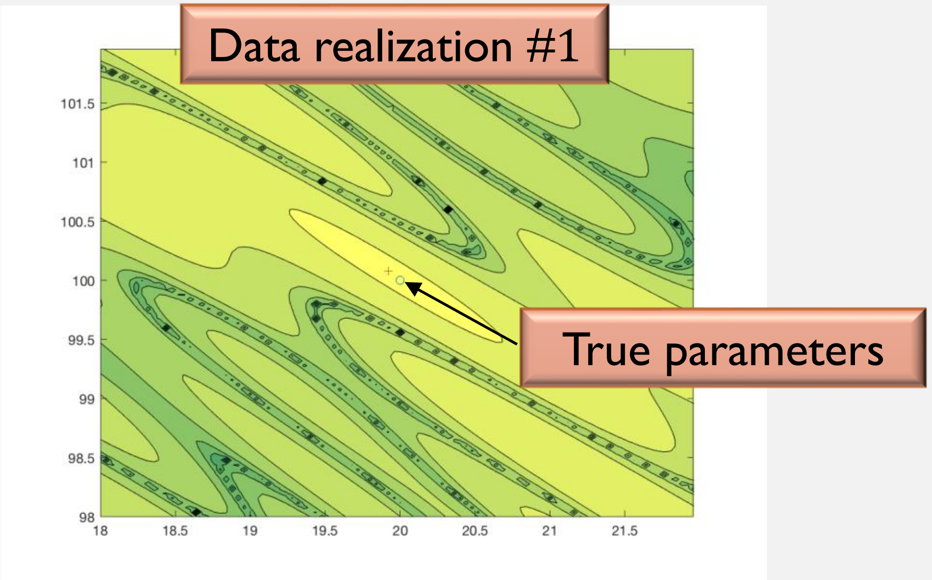
DATA SIMULATION

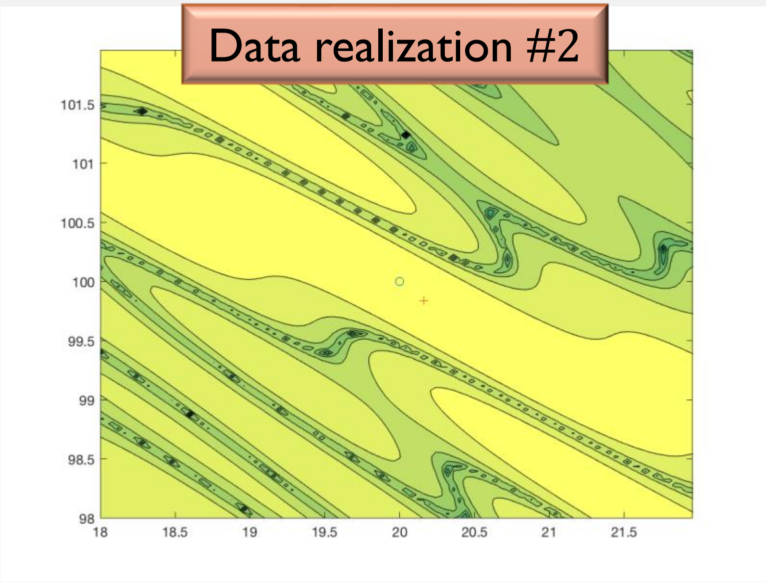
- In the case of the regression examples:
  - Keep the parameters of the true signal (quadratic chirp or spline) fixed
  - Add different noise realizations (pseudo-random numbers)
- Each data realization ⇒ one fitness function realization

STATISTICAL TUNING APPROACH



- For the same true signal parameters, the global minimizer will be different for different data realizations
- The best fitness value will always occur away from the true parameters
  - This is why we get error in parameter estimates in the presence of noise
- This fact can be used to develop a tuning procedure that is well-suited to regression



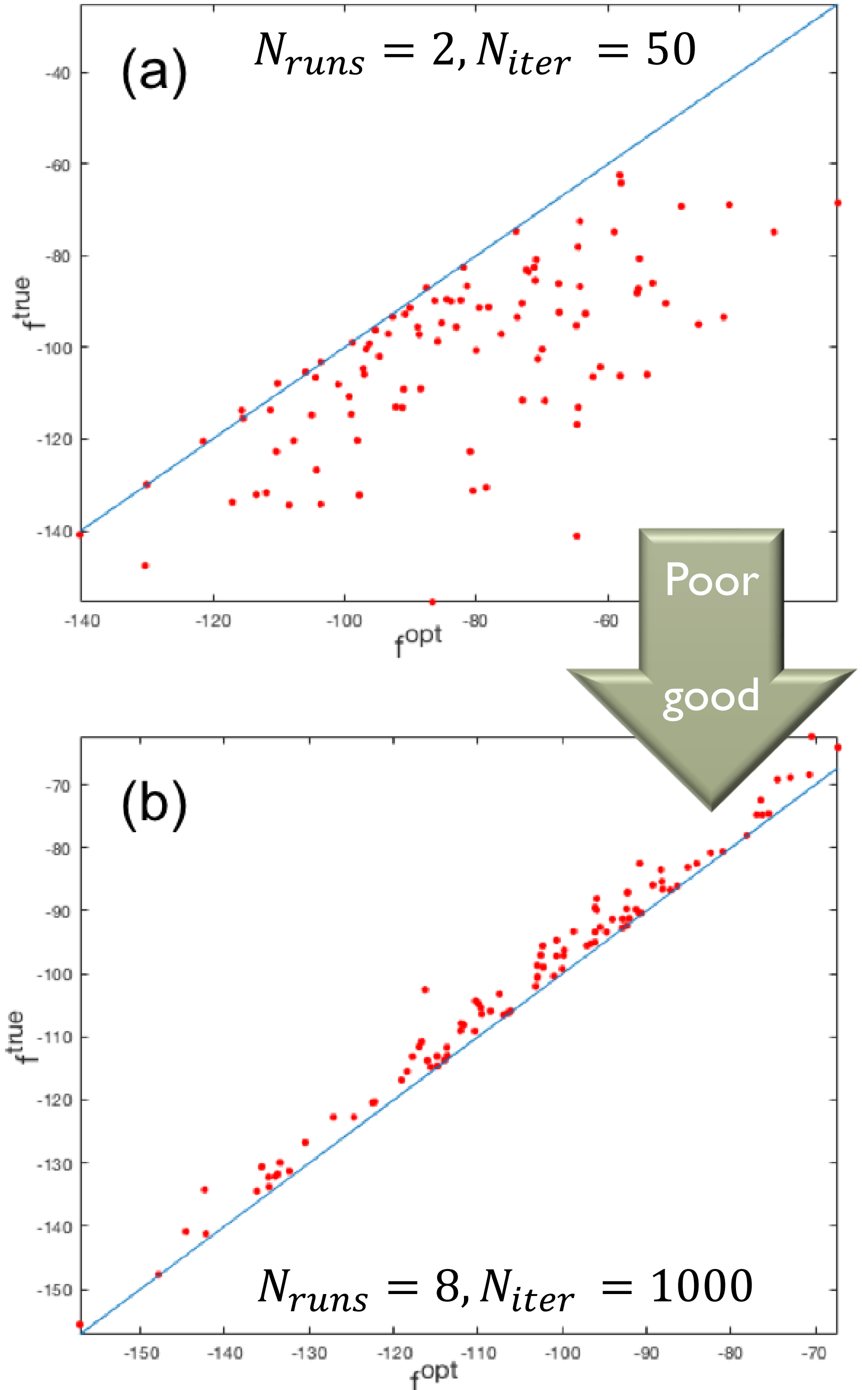


PSO TUNING FOR REGRESSION PROBLEMS

- Key idea: The global minimum must be lower than the fitness at the true parameters

$$f^{opt} < f^{true}$$

- PSO is working well if this condition is satisfied for a sufficiently high fraction of data realizations
- Proposed in:
  - Wang, Mohanty, Physical Review D, 2010
  - Normandin, Mohanty, Weerathunga, Physical Review D, 2018



PARAMETRIC REGRESSION

- The true parameters are known for simulated data
- ⇒Possible to check $f^{opt} < f^{true}$ for each data realization
- Set up a grid of values in
  - N_{iter}: Number of iterations
  - N_{runs}: Number of runs in BMR strategy
- For each combo (N_{iter}, N_{runs}): Get fraction X of N data realizations where this condition is satisfied
- Get all (N_{iter}, N_{runs}) for which X is below some preset value
- Pick the combo in this set with the lowest computational cost

NON-PARAMETRIC REGRESSION

- No explicit parametrization of models ⇒ $f^{opt} < f^{true}$ not easy to check
- The non-parametric fit may never capture every feature of the true signal ⇒ Fitness will typically be worse than fitness for true signal
- Further development of the basic idea needed: $f^{opt} \in f^{true} + [-\epsilon, \epsilon]$?
- ⇒ Seat-of-the-pants approach but not very difficult for PSO due to only two tuning parameters: N_{iter} and N_{runs}

RESULTS

Parametric and non-parametric regression

PARAMETRIC REGRESSION

- Quadratic chirp:

$$f\left(x;\bar{\theta}\right) = A\sin(2\pi\Phi(x))\,;\ \bar{\theta} = (A, a_1, a_2, a_3)$$
$$\Phi(x) = a_1 x + a_2 x^2 + a_3 x^3$$

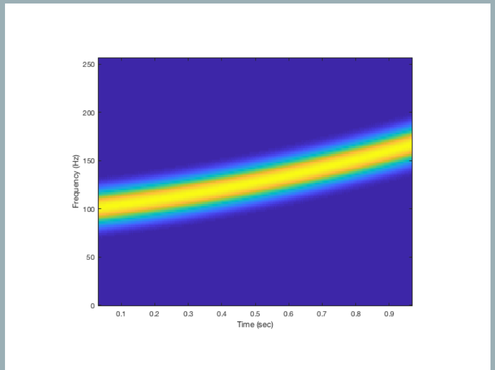
- True parameters

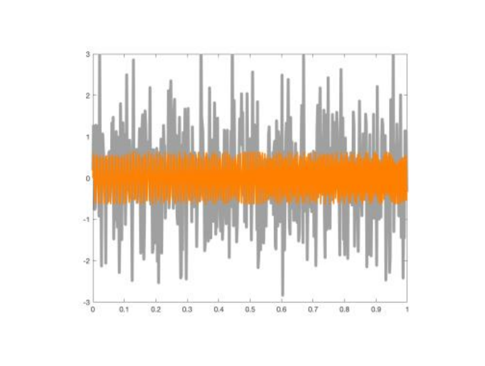
$$A = 0.625, a_1 = 100, a_2 = 20, a_3 = 10$$

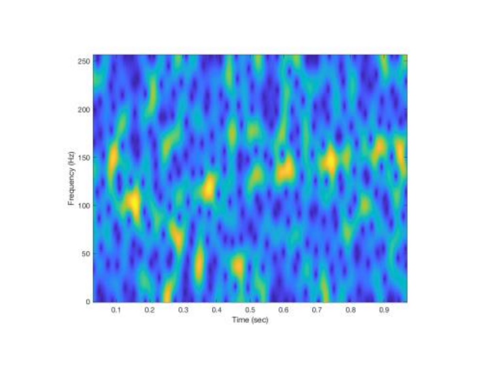
- White Gaussian Noise (WGN): iid Normal with mean =0 and variance =1
- 100 data realizations
- PSO Search space:

$$a_1 \in [10,150], a_2 \in [1,30], a_3 \in [1,15]$$

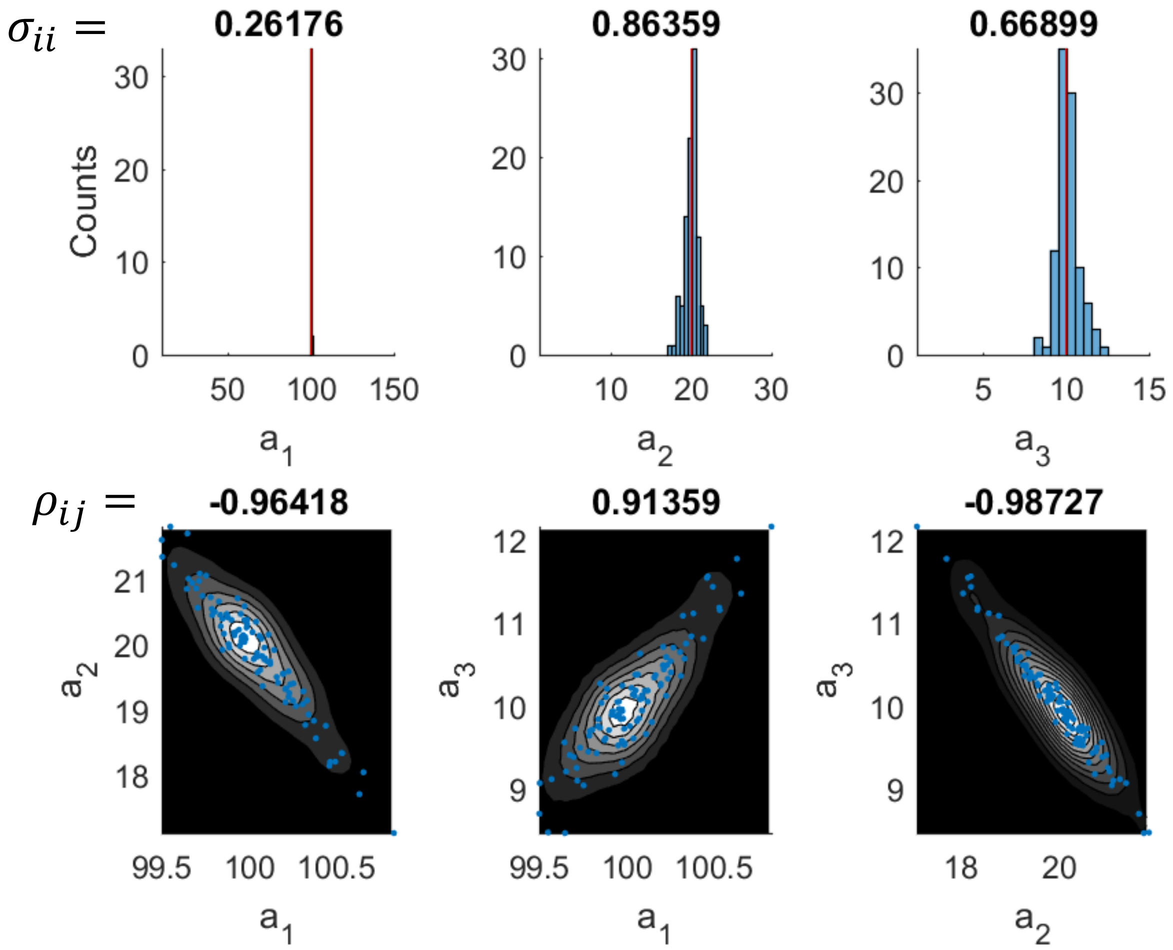
*True parameters not centered in search space







PARAMETER ESTIMATION



NON-PARAMETRIC REGRESSION

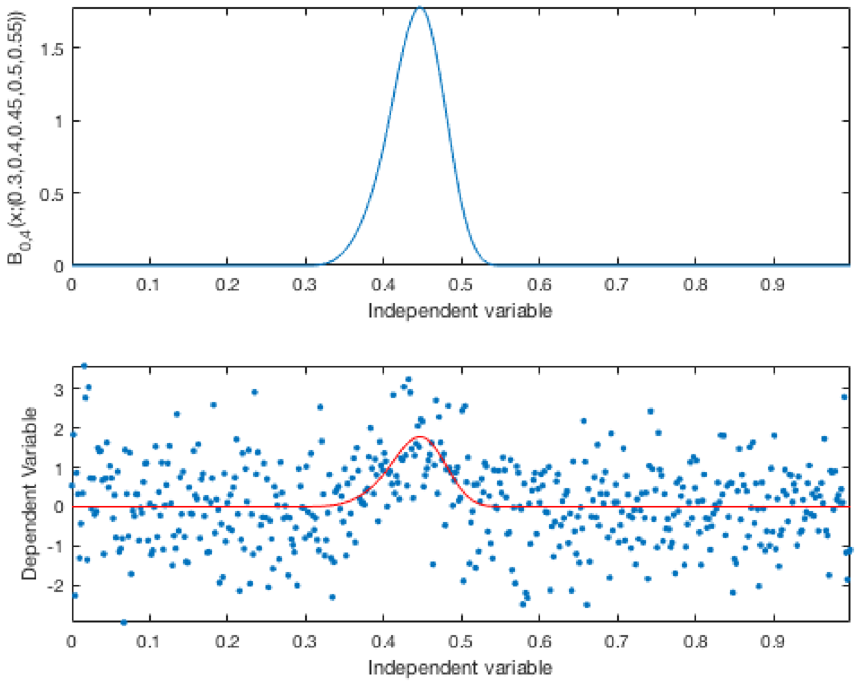
- True signal:

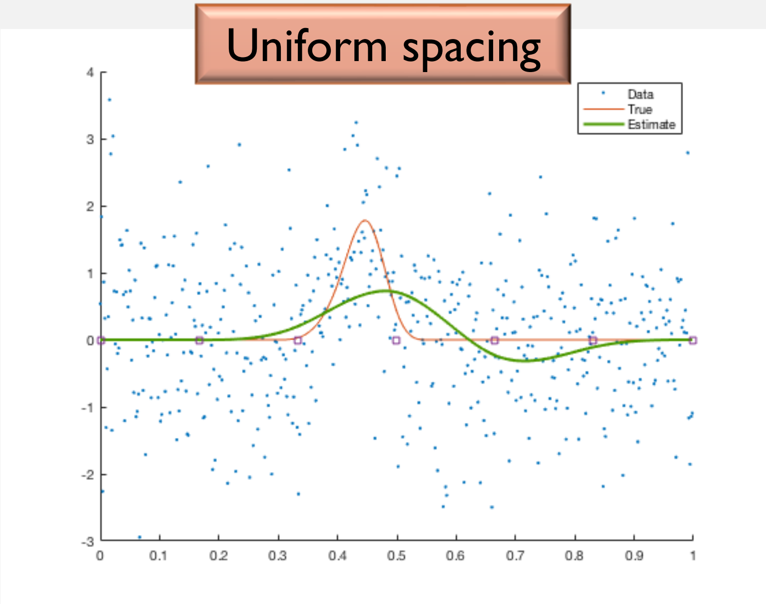
$$f\left(x;\bar{\theta}\right) = 10 \times B_{0,4}(x;\bar{c});$$
$$\bar{c} : \text{Breakpoint locations}$$
$$\bar{c} = (0.3, 0.4, 0.45, 0.5, 0.55)$$

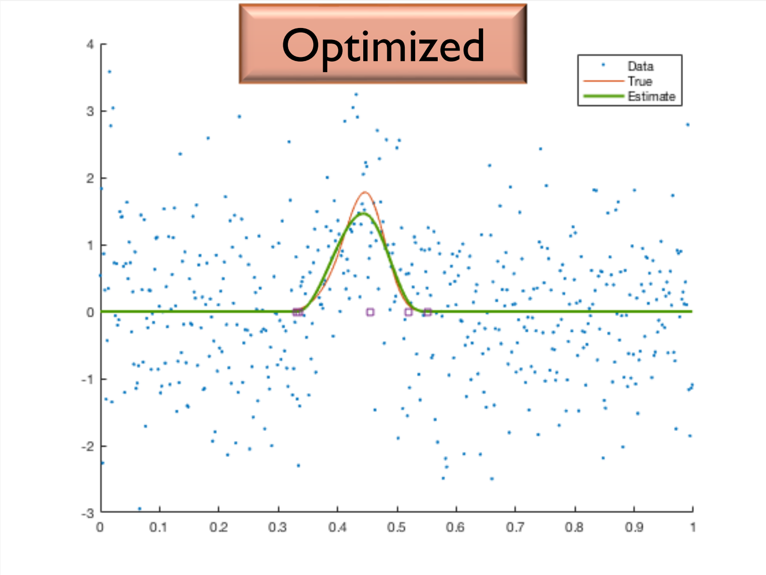
- White Gaussian Noise (WGN): iid Normal with mean $=$ 0 and variance $=$ 1
- 100 data realizations
- PSO Search space (after reparametrization of breakpoints):

$$\bar{b} \to \bar{\alpha}; \alpha_i \in (0,1)$$

*True breakpoints not uniformly spaced ⇒ not centered in search space







FIXED NUMBER OF BREAKPOINTS

- The true signal has 5 breakpoints
- We keep the same number of breakpoints for the spline to be fitted
- PSO tuning:
  - $N_{runs} = 4$
  - $N_{iter} = 200$
- PSO optimized breakpoints show much better performance than uniformly spaced breakpoints

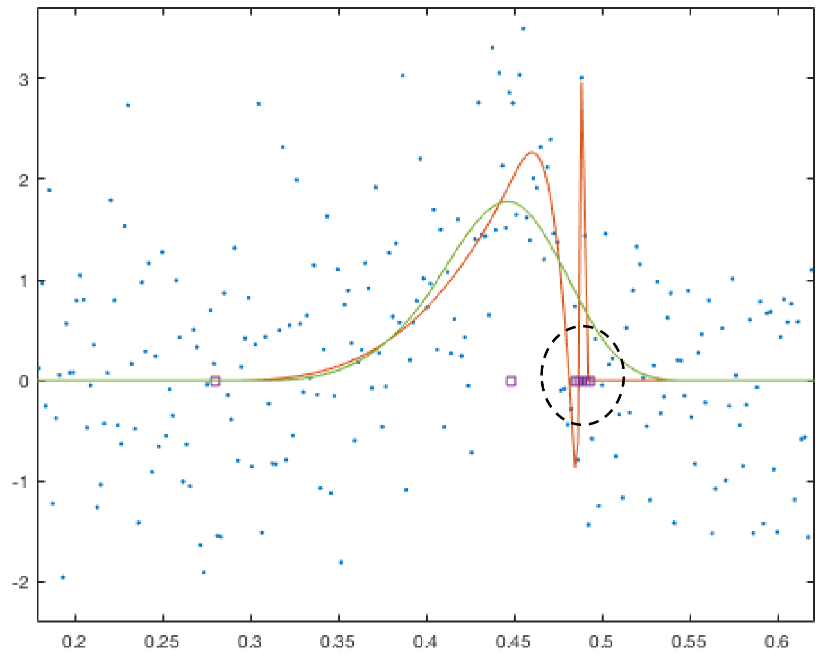
VARIABLE NUMBER OF BREAKPOINTS

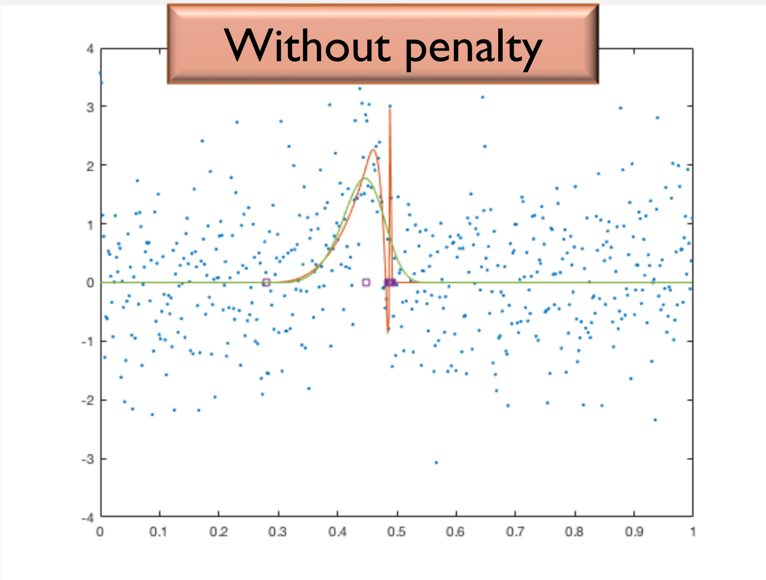
- In a realistic application, we do not have prior knowledge of the number of breakpoints to use
- $\Rightarrow$ Use **model selection**:
  - Fit the data with different breakpoint numbers ($\equiv$ different models)
  - Select the best number of breakpoints using the Akaike Information Criterion (AIC)
- *Model selection is a vast topic and AIC is not the only approach

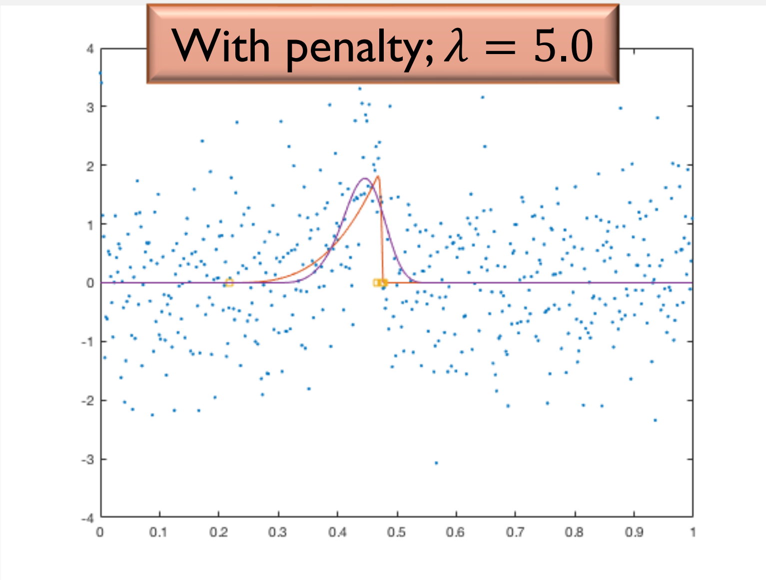
MODEL SELECTION

<u>Example</u>

- Best model found has 7 breakpoints
  - 5 breakpoints in true signal
- Variable number of breakpoints ⇒ Excessive freedom in the model
- Conspiracy: Excess knots cluster and coefficients $\bar{\alpha}$ increase to fit outliers
- Smoothness constraint on solution is violated







REGULARIZATION

- Penalized spline fit: Add a penalty term (regulator) to the cost function

$$\min_{\bar{b}}\left(\min_{\bar{\alpha}}\left(\sum_{i=0}^{N-1}\left(y_i-\sum_{j=0}^{M-1}\alpha_j B_{j,4}(x_i;\bar{b})\right)^2+\lambda\sum_{j=0}^{M-1}\alpha_j^2\right)\right)$$

- λ: Regulator gain
- Penalize solutions that have large values of α_i

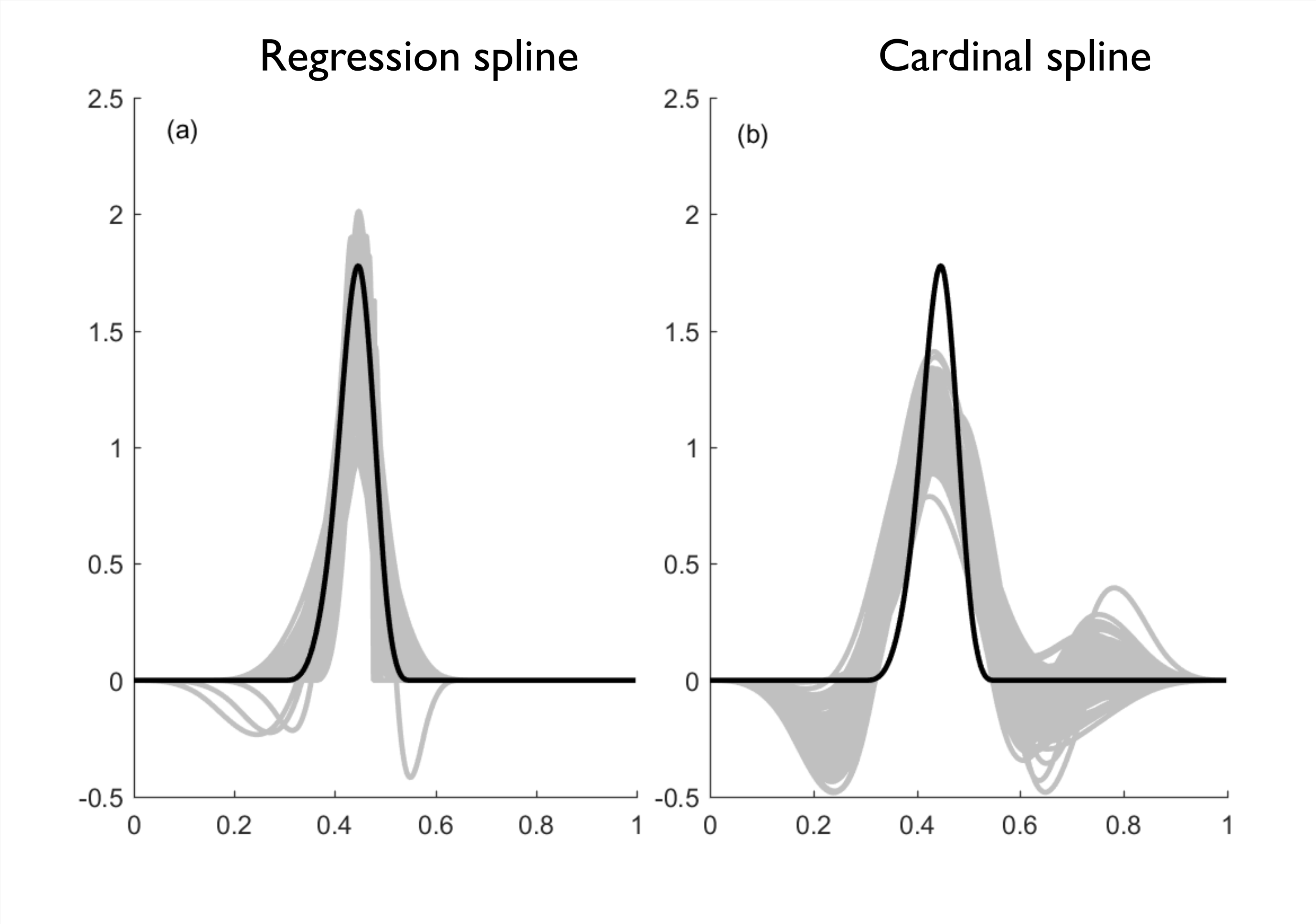
REGULARIZATION AND MODEL SELECTION

Cardinal spline fit

- Breakpoints spaced uniformly
- Model selection used

Regression spline fit

- Breakpoints optimized by PSO
- Model selection used
- Penalized spline used



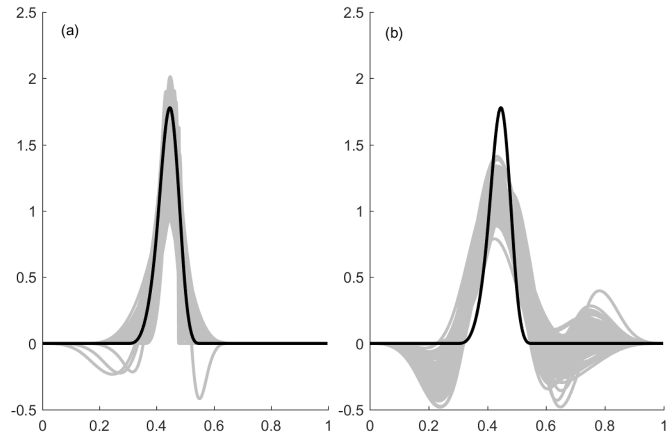
- 100 data realizations
- Breakpoint numbers: $\{5,6,7,8,9\}$

SUMMARY

BREAKING THE OPTIMIZATION BARRIER

Non-parametric regression

- Linear model
  - Optimization: simple
- Non-linear model
  - Optimization: Challenging
    - Swarm intelligence



Non-linear model fit by PSO

linear model fit

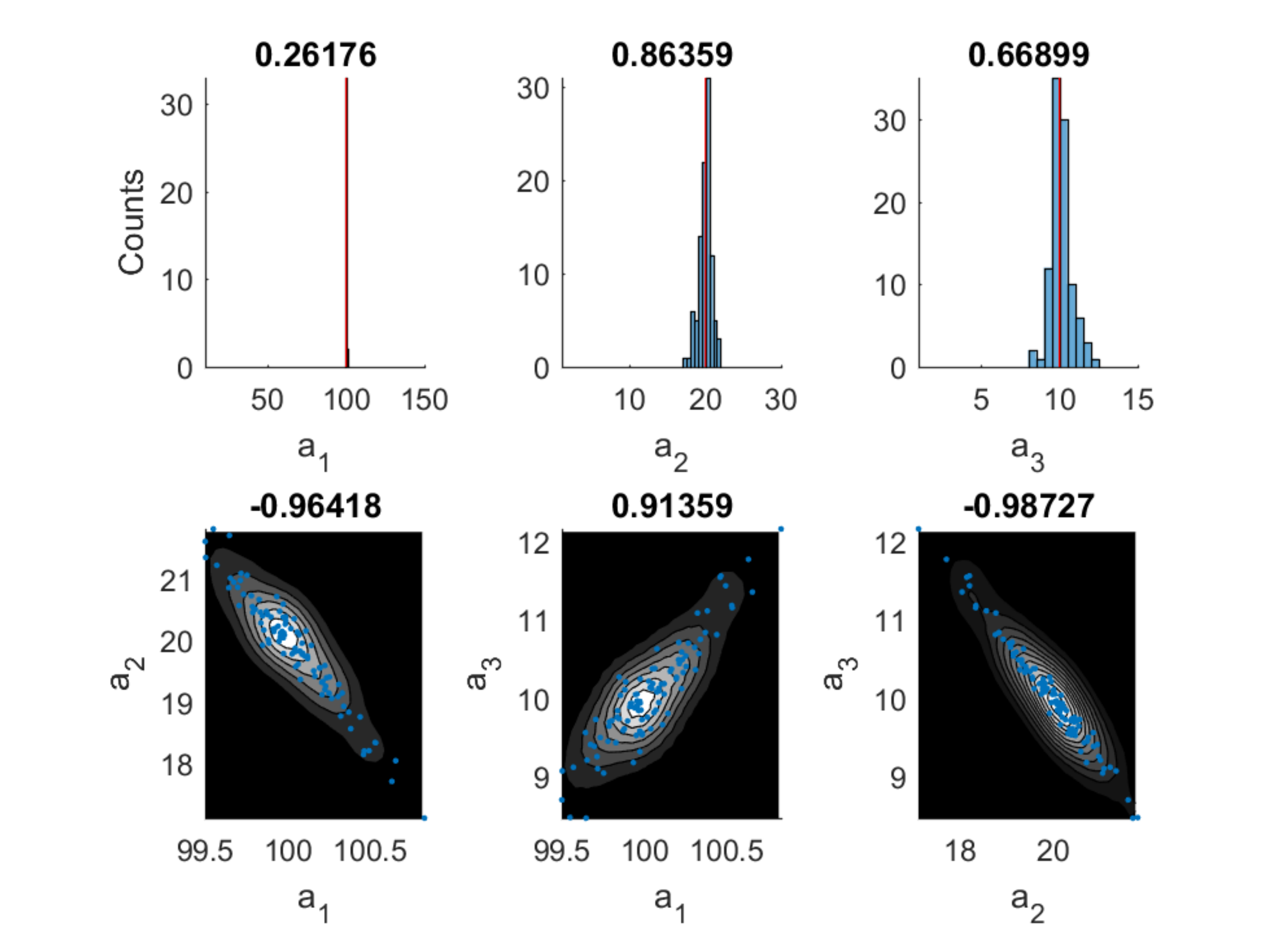
BREAKING THE OPTIMIZATION BARRIER

Parametric regression

Non-linear model

Optimization: Challenging

Swarm intelligence



SWARM INTELLIGENCE AND BIG DATA

Big data era: datasets and inference problems have become more complex

Flexible modeling ⇒ Non-parametric regression methods ⇒ large number of parameters ⇒ Optimization bottleck

- Forced to use linear models but non-linear models may be better

Swarm intelligence methods like PSO should be in the toolbox of every big data analyst

- Not a magic pill! Try SI methods on simpler problems first