Dynamic Anisotropic Smoothing for Noisy Derivative-Free Optimization

Sam Reifenstein*, Timothée Leleu* and Yoshihisa Yamamoto

ICML 2024 (poster session)

ICML
International Conference
On Machine Learning

NTT Research

Problem Setup (noisy derivative-free optimization)

Given: $f : \mathbb{R}^d \to \mathbb{R}$, and $\hat{f}(x, \zeta) = f(x) + \zeta$, find: $\text{argmax}_x f(x)$

(Unknown gradient of objective function) (Noisy Oracle) (Approximate Optimum of f)

Existing Approaches		
Gradient Estimator Methods: Use noisy samples to estimate gradient - Nesterov et al. 2017 - Gasnikov et al. 2023 - SPSA (Spall 1998)	**Traditional Zeroth-Order Methods** - - Nelder-Mead Methods - Trust Region Methods -	**Bayesian Optimization** - BOHB (Falkner et al 2018) - Optuna
Limitations		
- - Fixed sampling window causes noisy estimation of gradient - Do not adapt to heterogeneous curvature of objective	- - Typically, not designed to handle noisy functions	- - Not great scaling with dimension (number of parameters) - Large dependance on choice of fixed sampling window

Motivation: Heuristic Combinatorial Optimization Algorithms

Typical use-case of our algorithm:

➢ Parameter tuning for <u>non-convex combinatorial optimization</u> (CO) known to be NP-Hard (e.g. MAXCUT, k-SAT, TSP, etc.)

➢ Other noisy derivative-free optimization with the following key properties (common in machine learning)

Key properties for when to use our algorithm:

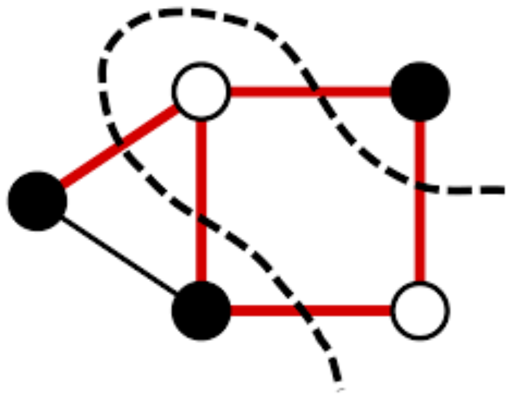
1) Evaluation the objective function is noisy*

* e.g. due to random initialization of CO solver and problem instance

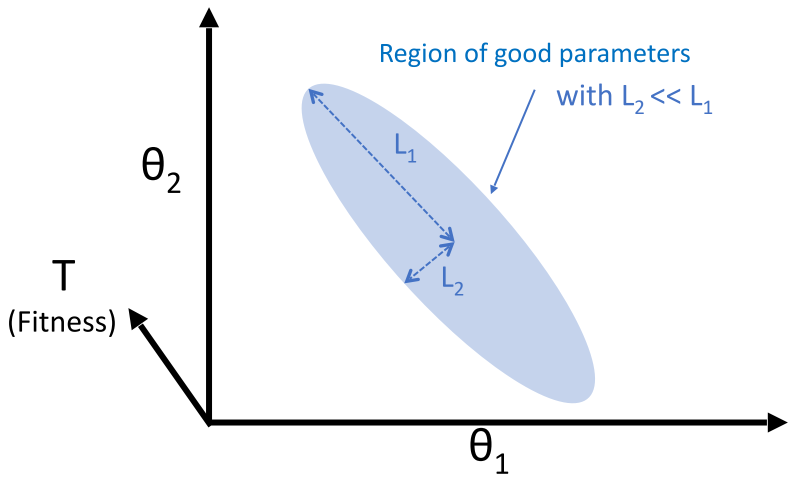
2) Heterogeneous curvature of objective function in the parameter space

3) Intermediate to large number of parameters

<u>NP-Hard combinatorial optimization (e.g. MAXCUT):</u>



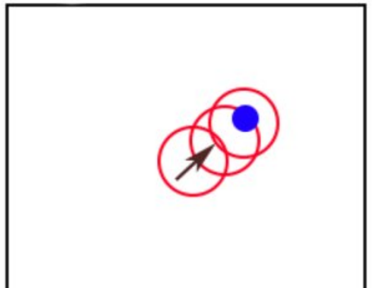
<u>Heterogeneous curvature of the parameter space:</u>



Extend Gaussian ball smoothing to include an adaptive sampling window

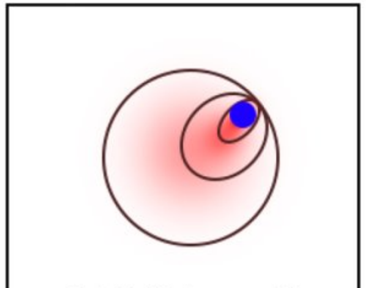
Nesterov et al 2017*		This Work
$f_\mu(x) = E_u(f(x+\mu u)).$	<- Smoothed objective -> function	$h(L,x) = \int \kappa(v) f(Lv+x)dv$
$\nabla f_\mu(x) = \frac{1}{\mu} E_u(f(x+\mu u)u)$	<- gradient estimators ->	$\frac{\partial h(L,x)}{\partial L} = (L^{-1})^\top \int (vv^\top - I)\kappa(v) f(Lv+x)dv,$ $\frac{\partial h(L,x)}{\partial x} = (L^{-1})^\top \int v\kappa(v) f(Lv+x)dv.$

Gradient estimation of smoothed objective function allows for gradient descent of x for a fixed window (μ).



Ball Smoothing

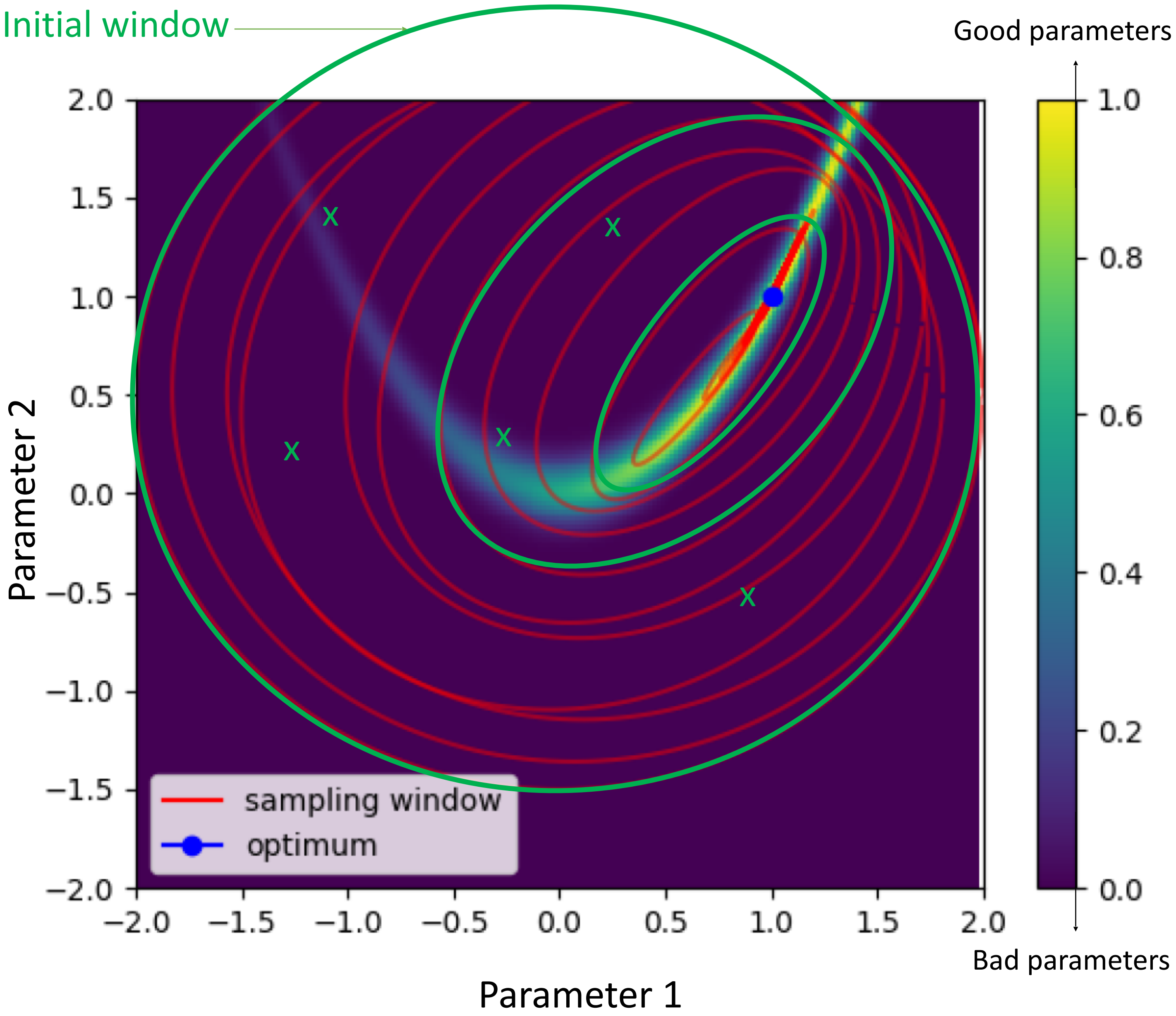
Gradient estimation of smoothed objective function allows for gradient descent of both window position (x) and window size and shape (L)

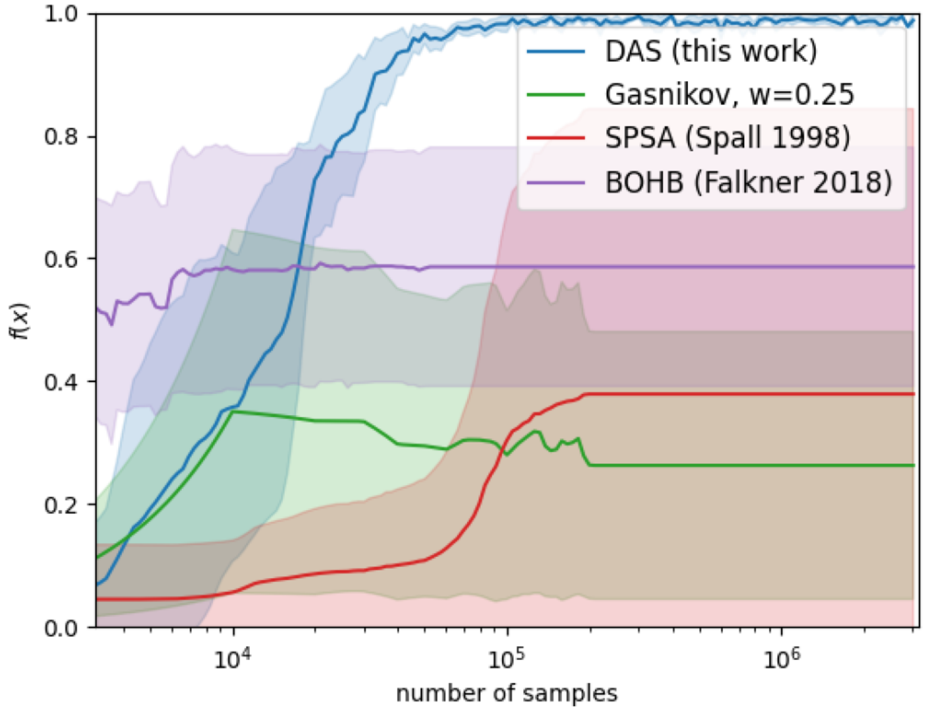


DAS (this work)

*Nesterov et al., Foundations of Computational Mathematics, 17(2), 527-566., 2017

Dynamic Anisotropic Smoothing on Rosenbrock function





	mean	worst	best
DAS (this work)	**0.981**	**0.962**	**0.994**
Gasnikov, w=0.25	0.280	0.000	0.564
SPSA (Spall 1998)	0.304	0.000	0.762
BOHB (Falkner 2018)	0.586	0.243	0.796

Table 1. Table shows best, worst, and mean fitness achieved by four algorithms for $n_s = 10^5$. The toy function is the modified Rosenbrock function in 4 dimensions with $\beta = 0.5$.

Dynamics of DAS in Differential Form

Shape of smoothing window

Noise in position estimation

can show is minimized when shape of smoothing window and fitness function are aligned

SDE's of DAS:

$$\begin{cases} \dfrac{dL}{dt} = \alpha_L \left(LL^\top \dfrac{\partial h(L,x)}{\partial L} + \lambda L + \eta_L \right), \\ \dfrac{dx}{dt} = \alpha_x \left(LL^\top \dfrac{\partial h(L,x)}{\partial x} + \eta_x \right), \end{cases}$$

Position of smoothing window

Smoothed objective function

Gaussian kernel

True objective (unknown)

Kernel smoothing:

$$\begin{aligned} h(L,x) &= \int \kappa(v) f(Lv + x) dv \\ &= \det(L) \int \kappa(L^{-1}(x-u)) f(u) du \end{aligned}$$

Fixed Points of SDE Dynamics

At the fixed points:

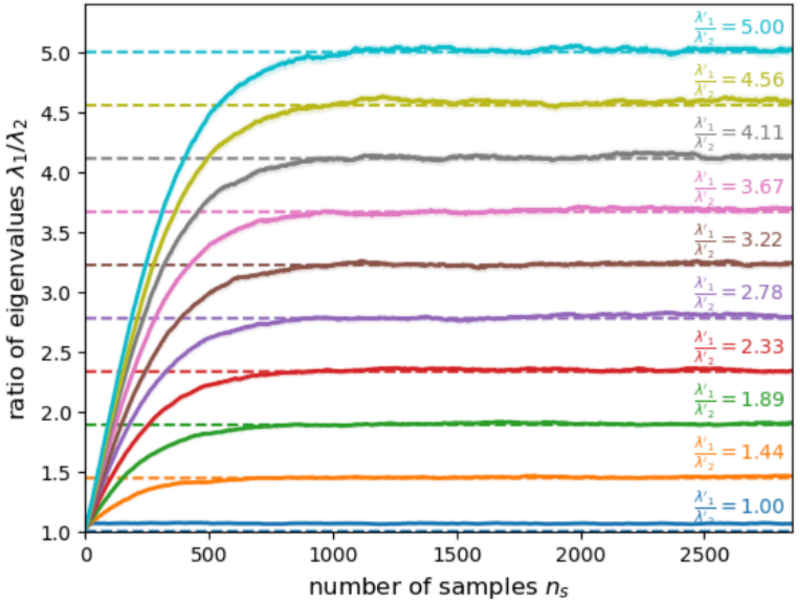
1) Sampling window position x converges to local optimum of smoothed objective:

$$\nabla_x h(x, L) = 0.$$

2) Sampling window shape L converges to match Hessian:

$$-\lambda(LL^\top)^{-1}_{ij} = \frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j}h(x, L),$$

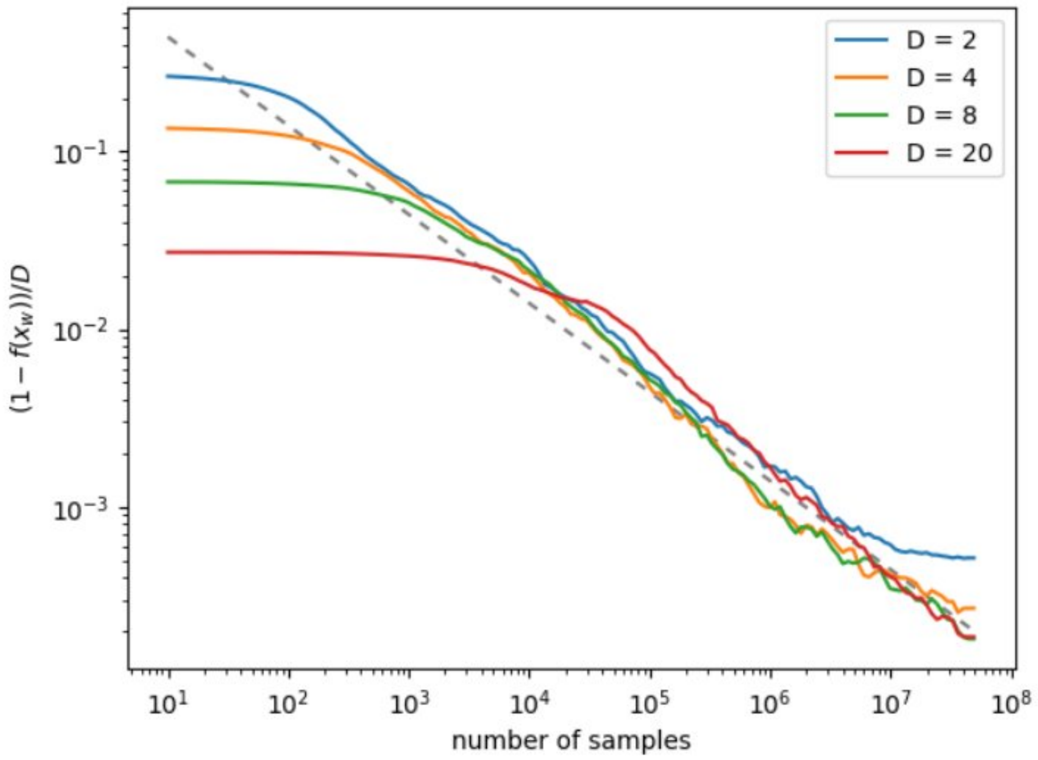
Numerical simulation of the convergence of window shape to the objective function's Hessian:



Notes:

- The matrix L, which is used as a preconditioner for the gradient, converges to approximate the Hessian of the objective function.
- Preconditioning is important when the curvature will be very different in different directions (this property is known to be common in machine learning)

Asymptotic Convergence



- Scaling with number of samples:

$$\epsilon = O(n_s^{-1/2})$$

- Scaling with problem dimension*:

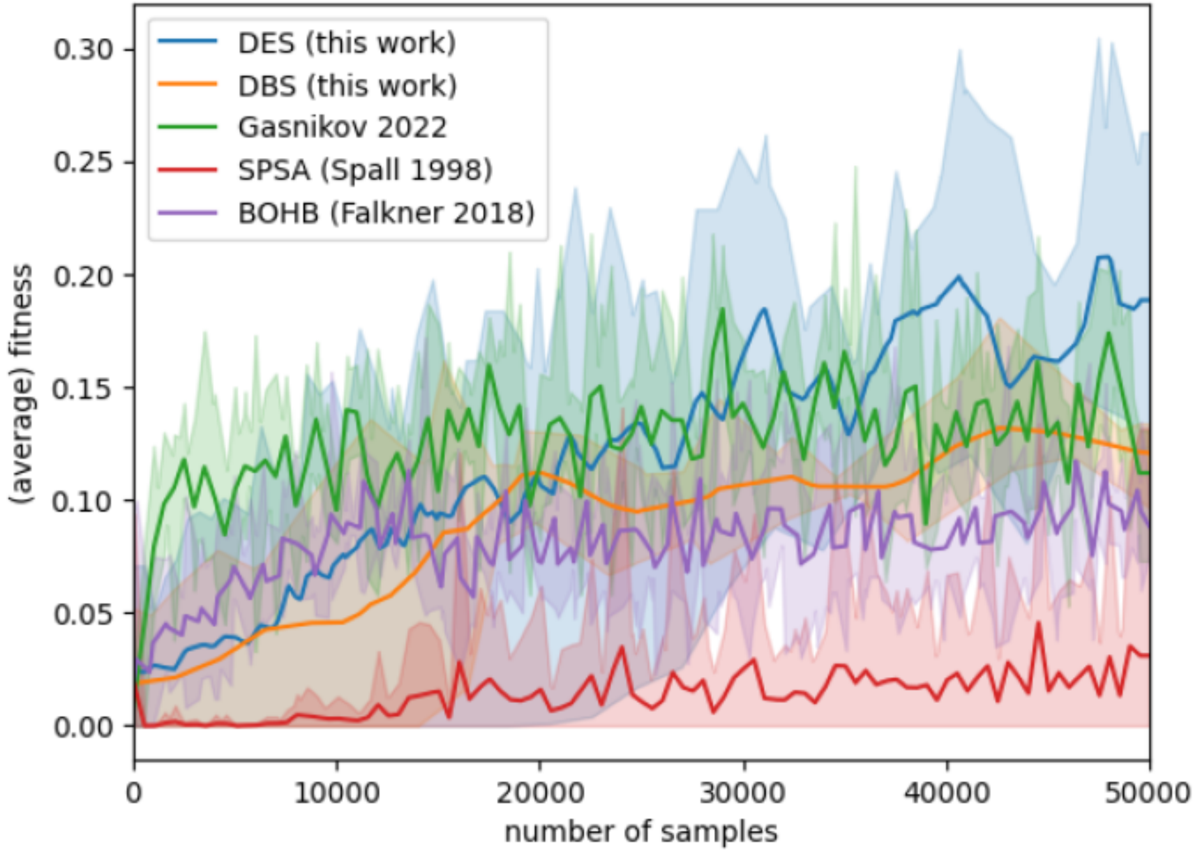
$$\epsilon = O(d)$$

➢ Empirically, DAS exhibits the theoretical optimal convergence with number of samples.

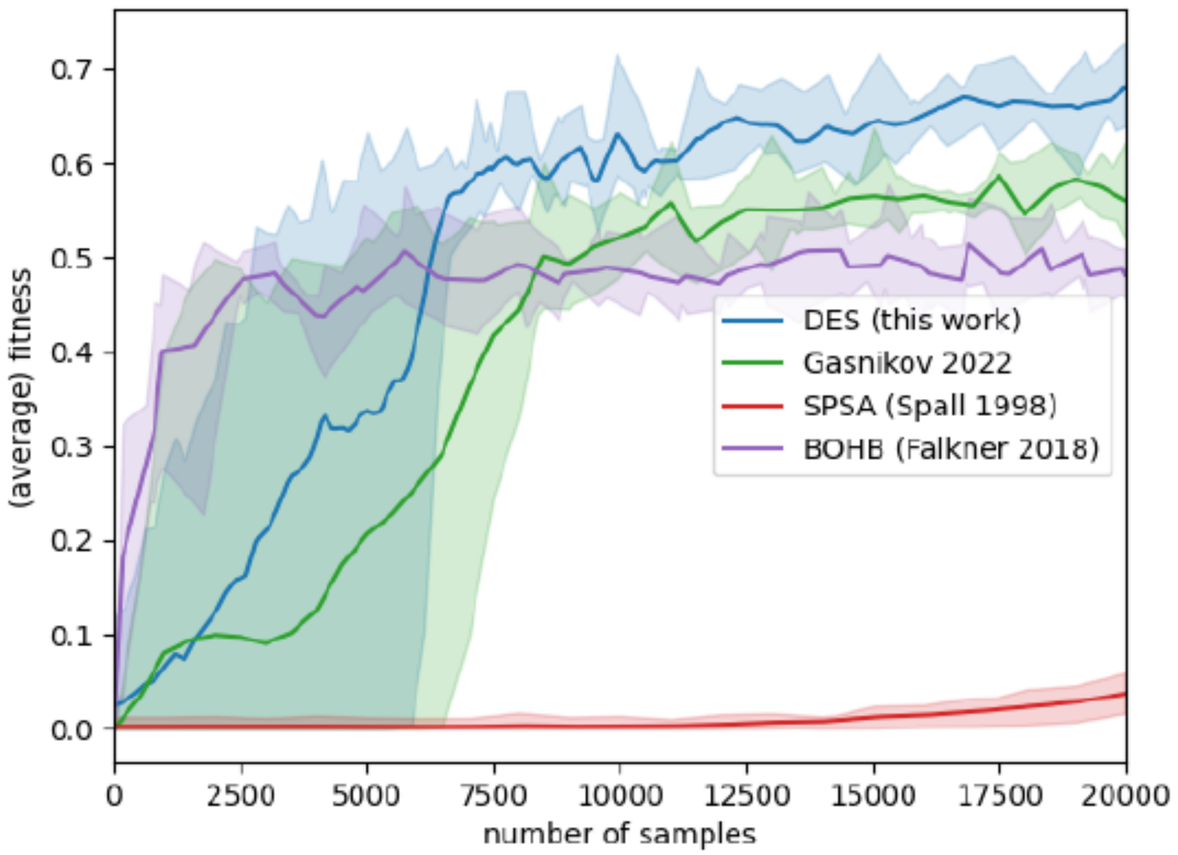
➢ Scaling of residual error with problem dimension depend on the objective function structure

Numerical Results:
Application to tuning combinatorial optimization solvers

SAT-CAC (Random 3-SAT N=150, α=4)



ISING-CAC (SK N=300)



SAT-CAC: Reifenstein et al. AOP, 15(2), 385-441, 2023

Ising-CAC: Leleu et al. PRL 122(4), 040607.2019, 2019

Outlook and Future Directions

- Apply DAS to tune "**differential solvers**"
  (=recently developed nonlinear coupled ODEs solving CO)

Coherent Ising Machine (CIM): Yamamoto et al. APL, 117(16), 2020
Chaotic Amplitude Control (CAC): Leleu et al. PRL 122(4), 040607.2019, 2019
Coherent SAT machines (SAT-CAC): Reifenstein et al. AOP, 15(2), 385-441, 2023

<u>Chaotic amplitude control:</u>

Recurrent synaptic updates
Neural network Parameters
Graph Convolution

$$x_i(t+1) = x_i(t) + \mathrm{dt}\left((1-p)x_i(t) + -x_i(t)^3 + e_i(t)\sum_j J_{ij}\right)$$

$$e_i(t+1) = e_i(t) + \mathrm{dt}\beta(1 - x_i(t)^2)$$

- Generalize to parameter tuning in machine learning when the parameter landscape has **heterogeneous curvature**.

Paper

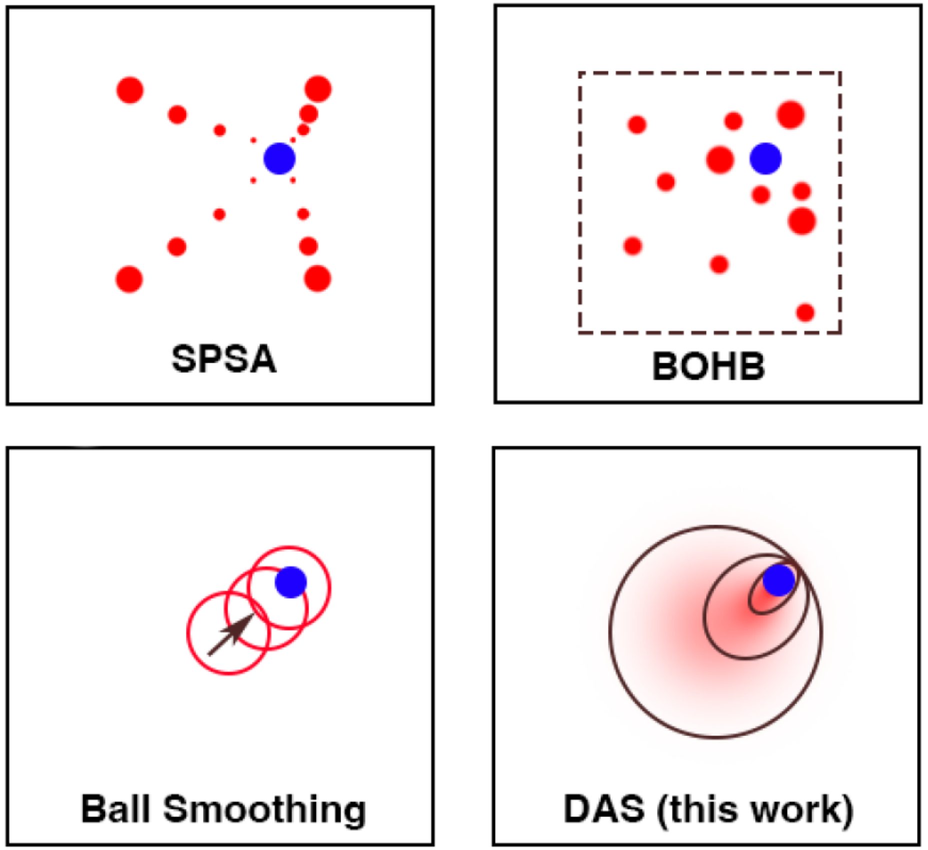
Code

(End here)

Derivative-free Optimization Methods

Finite difference methods:
→ Not robust to noise

Bayesian optimization:
→ Scales poorly with the number of parameters



Smoothing techniques:
→ Struggles with heterogeneous curvature

Our approach:
→ Good scaling
→ Robust to noise
→ Adapts to heterogeneous curvature

[Reifenstein*, Leleu*, Yamamoto, accepted ICML 2024]

Algorithm Description (Dynamic Anisotropic Smoothing)

Initialize x, L
(Start with x as some rough guess for the parameters and L large.)
for $t \leftarrow 0$ to T **do**
    Choose B random values for the random vector v
    Sample $y_i = \hat{f}(x + Lv_i)$ for each random vector
    Using y_i, compute the estimates of $\frac{\partial h(L,x)}{\partial L} \approx \hat{h}_L$ and $\frac{\partial h(L,x)}{\partial x} \approx \hat{h}_x$
    Compute $\Delta L = LL^\top \hat{h}_L$, $\Delta x = LL^\top \hat{h}_x$
    Set $L \leftarrow L + \Delta t \Delta L$, $x \leftarrow x + \Delta t \Delta x$ (update window)
end for
return x (Return the putative best parameters)

We evolve two variables: a d-dimensional vector **x** corresponding to our current guess of where the optimum is, and a matrix **L** representing the size and shape of the window in which the objective function is sampled. **L** evolves to compensate for the curvature of the objective function in the vicinity of **x**.

When is it best to use DAS?

1. Noisy Objective Function
2. Heterogeneous Curvature of Objective Function
3. Medium Size Dimension d = 4-20*

(*NOTE: Modifications of DAS could be useful for larger dimensions relevant to training of NNs)

Paper

Code