Self-Adaptive Surrogate-Assisted Covariance Matrix Adaptation Evolution Strategy

Ilya Loshchilov, Marc Schoenauer, Michèle Sebag

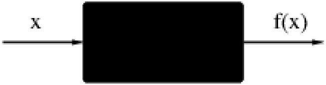
TAO
INRIA – CNRS – Univ. Paris-Sud

Motivations

Find Argmin $\{\mathcal{F} : X \mapsto \mathbb{R}\}$

Context: ill-posed optimization problems continuous

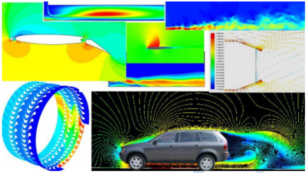
- Function $\mathcal{F}$ (fitness function) on $X \subset \mathbb{R}^d$
- Gradient not available or not useful
- $\mathcal{F}$ available as an oracle (black box)



Build $\{\mathbf{x_1}, \mathbf{x_2}, \ldots\} \rightarrow$ Argmin$(\mathcal{F})$

Black-Box approaches

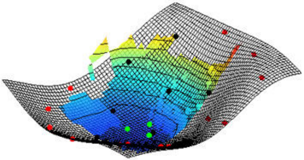
+ Robust
– High computational costs: number of expensive function evaluations (e.g. CFD)



Surrogate-Assisted Optimization

Principle

- Gather $\mathcal{E} = \{(x_i, \mathcal{F}(x_i))\}$ *training set*
- Build $\hat{\mathcal{F}}$ from $\mathcal{E}$ *learn surrogate model*
- Use surrogate model $\hat{\mathcal{F}}$ for some time:
  - Optimization: use $\hat{\mathcal{F}}$ instead of true $\mathcal{F}$ in std algo
  - Filtering: select promising $\mathbf{x_i}$ based on $\hat{\mathcal{F}}$ in population-based algo.
- Compute $\mathcal{F}(\mathbf{x_i})$ for some $\mathbf{x_i}$
- Update $\hat{\mathcal{F}}$
- Iterate



Surrogate model based optimisation
with noisy and missing data

Surrogate-Assisted Optimization, cont

Issues

- Learning
  - Hypothesis space (polynoms, neural nets, Gaussian processes,...)
  - Selection of training set (prune, update, ...)
  - What is the learning quality indicator ?
- Interaction of Learning & Optimization modules
  - Schedule (when to relearn)
  - $\ast$ How to use $\hat{\mathcal{F}}$ to support optimization search
  - $\ast\ast$ How to use search results to support learning $\hat{\mathcal{F}}$

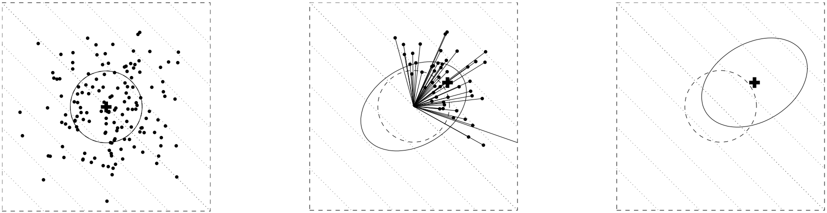
This talk

- $\ast$ Using Covariance-Matrix Estimation within Support Vector Machines
- $\ast\ast$ Self-adaptation of surrogate model hyper-parameters

Content

(μ,λ)-Covariance Matrix Adaptation Evolution Strategy

Rank-μ Update



$\boldsymbol{y}_i \sim \mathcal{N}(\mathbf{0},\mathbf{C})\,,\ \mathbf{C}=\mathbf{I}$
$\boldsymbol{x}_i = \boldsymbol{m} + \sigma\,\boldsymbol{y}_i\,,\ \sigma = 1$

$\mathbf{C}_\mu = \frac{1}{\mu}\sum \boldsymbol{y}_{i:\lambda}\boldsymbol{y}_{i:\lambda}^{\mathrm{T}}$
$\mathbf{C} \leftarrow (1-1)\times\mathbf{C} + 1\times\mathbf{C}_\mu$

$\boldsymbol{m}_{\text{new}} \leftarrow \boldsymbol{m} + \frac{1}{\mu}\sum \boldsymbol{y}_{i:\lambda}$

sampling of λ solutions

calculating $\mathbf{C}$ from best μ out of λ

new distribution

Ruling principles:
i). the adaptation increases the probability of successful steps to appear again;
ii). $\mathbf{C} \approx H^{-1}$

Invariance: Guarantee for Generalization

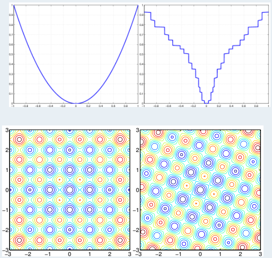
Invariance properties of CMA-ES

- Invariance to order preserving transformations in function space

  true for all comparison-based algorithms

- Translation and rotation invariance

  thanks to $\mathbf{C}$



CMA-ES is almost **parameterless**

- Tuning on a small set of functions — Hansen & Ostermeier 2001
- Default values generalize to whole classes
- Exception: population size for multi-modal functions

  try restarts (see our PPSN'2012 paper)

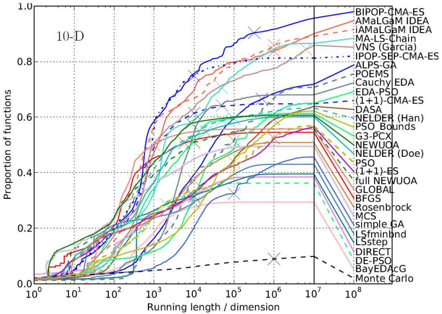
State-of-the-art Results

BBOB – Black-Box Optimization Benchmarking

- ACM-GECCO workshop, in 2009, 2010 and 2012
- Set of 24 benchmark functions, dimensions 2 to 40
- With known difficulties (ill-conditioning, non-separability, . . .)
- Noisy and non-noisy versions

Competitors include

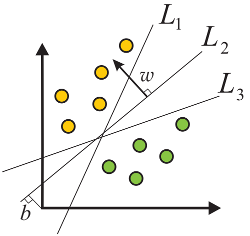
- BFGS (Matlab version),
- DFO (Derivative-Free Optimization, Powell 04)
- Differential Evolution
- Particle Swarm Optimization
- and many more

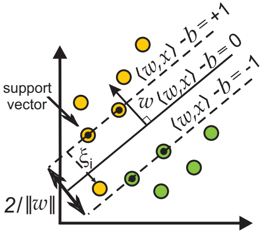


Contents

Support Vector Machine for Classification

Linear Classifier





Main Idea

Training Data:

$$D = \left\{(x_i, y_i) | x_i \in \mathbb{R}^d, y_i \in \{-1, +1\}\right\}_{i=1}^{n}$$

$$\langle w, x_i \rangle - b \geq +1 \Rightarrow y_i = +1;$$

$$\langle w, x_i \rangle - b \leq -1 \Rightarrow y_i = -1;$$

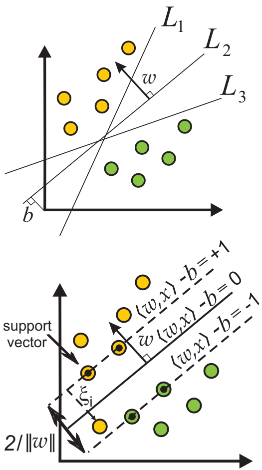
Optimization Problem: Primal Form

Minimize$_{\{w,\ \xi\}}$ $\frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i$

subject to: $y_i(\langle w, x_i \rangle - b) \geq 1 - \xi_i,\ \xi_i \geq 0$

Support Vector Machine for Classification

Linear Classifier



Optimization Problem: Dual Form

Quadratic in Lagrangian multipliers:

Maximize$_{\{\alpha\}}$ $\sum_i^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle$

subject to: $0 \leq \alpha_i \leq C$, $\sum_i^n \alpha_i y_i = 0$

Properties

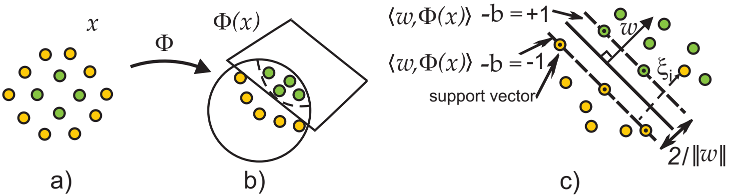
Decision Function:

$\hat{F}(x) = sign(\sum_i^n \alpha_i y_i \langle x_i, x\rangle - b)$

The Dual form may be solved using standard quadratic programming solver.

Support Vector Machine for Classification

Non-Linear Classifier



Non-linear classification with the "Kernel trick"

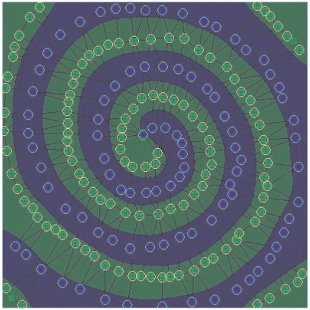
Maximize$_{\{\alpha\}} \sum_i^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j)$
subject to: $0 \leq \alpha_i \leq C$, $\sum_i^n \alpha_i y_i = 0$,
where $K(x, x') =_{def} < \Phi(x), \Phi(x') >$ **is the Kernel function** [a]
Decision Function: $\hat{F}(x) = sign(\sum_i^n \alpha_i y_i K(x_i, x) - b)$

[a] Φ must be chosen such that K is positive semi-definite

Support Vector Machine for Classification

Non-Linear Classifier: Kernels

Gaussian or Radial Basis Function (RBF): $K(x_i, x_j) = exp(\frac{\|x_i - x_j\|^2}{2\sigma^2})$



Regression To Value or Regression To Rank?

Why Comparison-Based Surrogates?

Example

- $F(x) = x_1^2 + (x_1 + x_2)^2$.
- An efficient Evolutionary Algorithm (EA) **with surrogate models** may be 4.3 faster on $F(x)$.
- But the same EA is only 2.4 faster on $G(x) = F(x)^{1/4}$! [a]

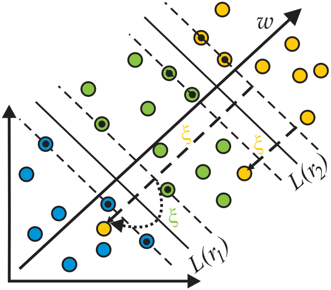
[a] CMA-ES with quadratic meta-model (lmm-CMA-ES) on fSchwefel 2-D

✓ Comparison-based surrogate models → **invariance** to rank-preserving transformations of $F(x)$!

Rank-based Support Vector Machine

Find $\hat{F}(x)$ which preserves the ordering of training/test points

- On training set $\mathcal{E} = \{\mathbf{x_i}, i = 1 \ldots n\}$
- expert gives preferences: $(\mathbf{x_{i_k}} \succ \mathbf{x_{j_k}})$, $k = 1 \ldots K$
- *underconstrained regression*



Order constraints

Primal form

$$\begin{cases} \text{Minimize} & \frac{1}{2}\,||\mathbf{w}||^2 + C\sum_{k=1}^{K} \xi_k \\ \text{subject to} & \forall\, k,\ \langle \mathbf{w}, \mathbf{x_{i_k}} \rangle - \langle \mathbf{w}, \mathbf{x_{j_k}} \rangle \geq 1 - \xi_k \end{cases}$$

Surrogate Models for CMA-ES

Using Rank-SVM

- Builds a global model using Rank-SVM

$$\mathbf{x_i} \succ \mathbf{x_j} \text{ iff } \mathcal{F}(\mathbf{x_i}) < \mathcal{F}(\mathbf{x_j})$$

- Kernel and parameters highly problem-dependent

ACM Algorithm

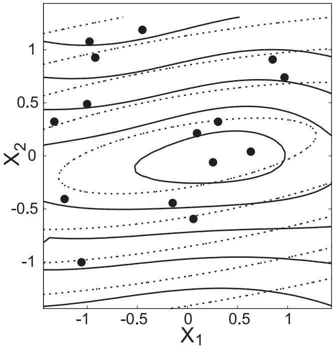
- Use C from CMA-ES as Gaussian kernel

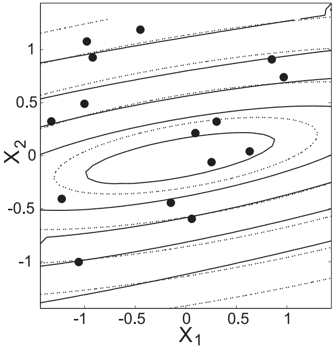
I. Loshchilov, M. Schoenauer, M. Sebag (2010). "Comparison-based optimizers need comparison-based surrogates”

Model Learning

Non-Separable Ellipsoid Problem

$$K(x_i, x_j) = e^{-\frac{(x_i - x_j)^T (x_i - x_j)}{2\sigma^2}}; \quad K_C(x_i, x_j) = e^{-\frac{(x_i - x_j)^T C^{-1} (x_i - x_j)}{2\sigma^2}}$$





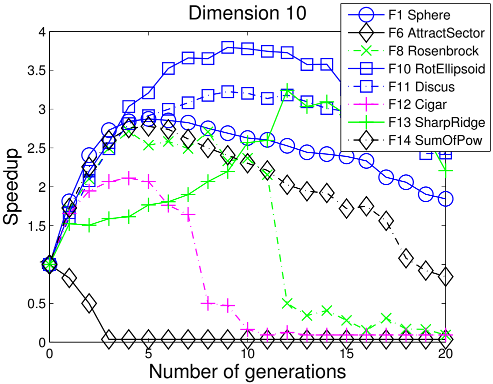
Invariance to rotation of the search space thanks to $C \approx H^{-1}$!

Contents

Using the Surrogate Model

Direct Optimization of Surrogate Model

Simple optimization loop:
optimize $\mathcal{F}$ for 1 generation, then optimize $\hat{\mathcal{F}}$ for $\hat{n}$ generations.

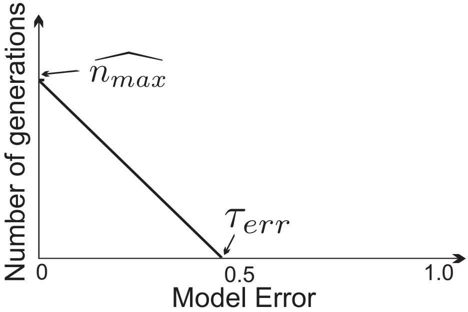


How to choose $\hat{n}$?

Adaptation of Surrogate's Life-Length

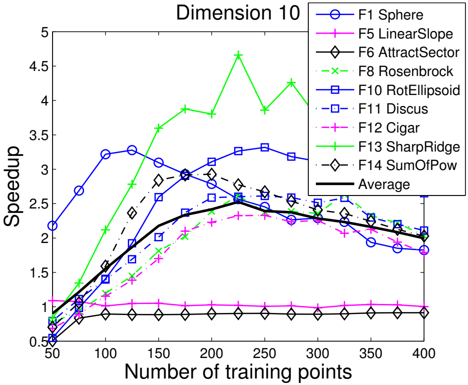
- Test set: λ recently evaluated points
- Model error: fraction of incorrectly ordered points

Number of generation is **inversely** proportion to the model error:

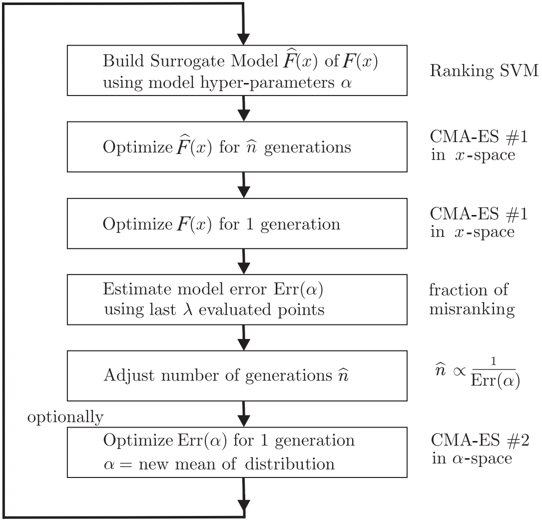


Model Hyper-Parameter Sensitive Results

The speed-up of surrogate-assisted CMA-ES is very sensitive to the number of training points!

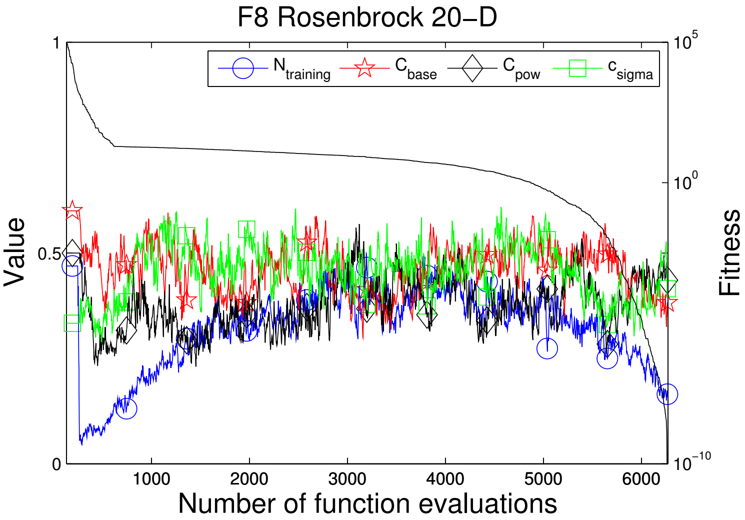


The Proposed ^{s*}aACM Algorithm



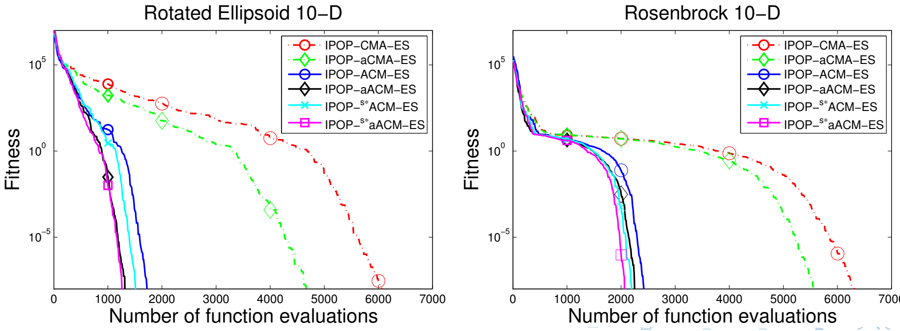
Surrogate-assisted CMA-ES with online adaptation of model hyper-parameters.

Online Adaptation of Model Hyper-Parameters



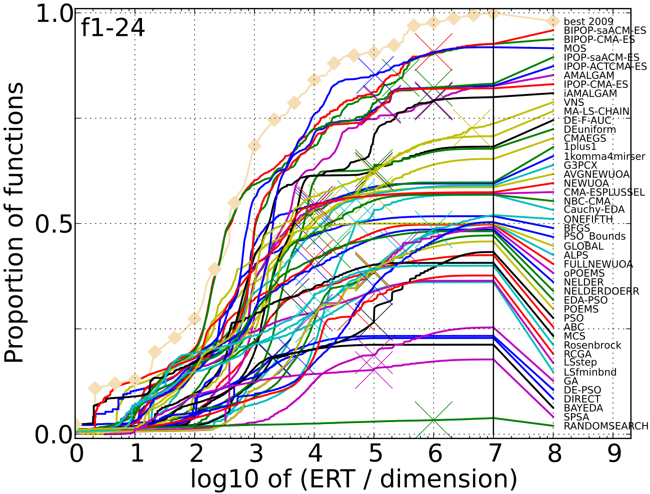
Results

- Self-adaptation of model hyper-parameters is better than the best offline settings! (IPOP-^{s*}aACM vs IPOP-aACM)
- Improvements of original CMA lead to improvements of its surrogate-assisted version (IPOP-^{s*}aACM vs IPOP-^{s*}ACM)



Results on Black-Box Optimization Competition

BIPOP-^{s*}aACM and IPOP-^{s*}aACM (with restarts) on 24 noiseless 20 dimensional functions



Machine Learning for Optimization: Discussion

- ^{s*}aACM is from 2 to 4 times faster on **Uni-Modal Problems**.
- Invariant to rank-preserving and orthogonal transformations of the search space : Yes
- The computation complexity (the cost of speed-up) is $O(d^3)$
- The source code is available online: `https://sites.google.com/site/acmesgecco/`

Open Questions

- How to improve the search on **multi-modal** functions ?
- **What else** from Machine Learning can improve the search ?

Thank you for your attention!

Questions?

Related Publications:

Joint work with Marc Schoenauer and Michèle Sebag:

"Alternative Restart Strategies for CMA-ES". *Parallel Problem Solving from Nature (PPSN XII), September 2012.*

"Black-box Optimization Benchmarking of IPOP-saACM-ES and BIPOP-saACM-ES on the BBOB-2012 Noiseless Testbed". *GECCO'2012 BBOB Workshop, July 2012.*

"Dominance-Based Pareto-Surrogate for Multi-Objective Optimization". *Simulated Evolution and Learning (SEAL 2010), December 2010.*

"Comparison-Based Optimizers Need Comparison-Based Surrogates". *Parallel Problem Solving from Nature (PPSN XI), September 2010.*

"A Pareto-Compliant Surrogate Approach for Multiobjective Optimization". *GECCO'2010 , July 2010.*

"A Mono Surrogate for Multiobjective Optimization". *GECCO' 2010 EMO workshop, July 2010.*