Surrogate models for Single and Multi-Objective Stochastic Optimization: Integrating Support Vector Machines and Covariance-Matrix Adaptation-ES

Ilya Loshchilov, Marc Schoenauer, Michèle Sebag

TAO
CNRS – INRIA – Univ. Paris-Sud

May 23rd, 2011

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

+ Applicable
+ Robust comparison-based approaches are invariant
– High computational costs: number of function evaluations

Surrogate 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 optimization, cont

Issues

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

This talk

∗ Using Covariance-Matrix Estimation within Support Vector Machines

∗∗ Using SVM for multi-objective optimization

Content

Stochastic Search

A black box search template to minimize $f : \mathbb{R}^n \rightarrow \mathbb{R}$

Initialize distribution parameters $\boldsymbol{\theta}$, set sample size $\lambda \in \mathbb{N}$
While not terminate

1. Sample distribution $P(\boldsymbol{x}|\boldsymbol{\theta}) \rightarrow \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda \in \mathbb{R}^n$
2. Evaluate $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda$ on f
3. Update parameters $\boldsymbol{\theta} \leftarrow F_\theta(\boldsymbol{\theta}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda, f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_\lambda))$

Covers

- Deterministic algorithms,
- Evolutionary Algorithms, PSO, DE P implicitly defined by the variation operators
- Estimation of Distribution Algorithms

The $(\mu, \lambda)-$Evolution Strategy

Gaussian Mutations

$$\boldsymbol{x}_i \sim \boldsymbol{m} + \sigma \mathcal{N}_i(\mathbf{0}, \mathbf{C}) \qquad \text{for } i = 1, \ldots, \lambda$$

as perturbations of $\boldsymbol{m}$ where $\boldsymbol{x}_i, \boldsymbol{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, and $\mathbf{C} \in \mathbb{R}^{n \times n}$

where

- the mean vector $\boldsymbol{m} \in \mathbb{R}^n$ represents the favorite solution
- the so-called step-size $\sigma \in \mathbb{R}_+$ controls the *step length*
- the covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ determines the shape of the distribution ellipsoid

How to update $\boldsymbol{m}$, σ, and $\mathbf{C}$?

History

The one-fifth rule — Rechenberg, 73

- One single parameter σ for the whole population
- Measure empirical success rate
- Increase σ if too large, decrease σ if too small

Often wrong in non-smooth landscapes

Self-adaptive mutations — Schwefel, 81

- Each individual carries its own mutation parameter

  from 1 to $\frac{n^2-n}{2}$
- Log-normal mutation of mutation parameters
- (Normal) mutation of individual

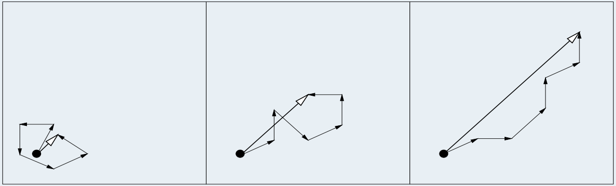
Adaptation is slow for full covariance case

Cumulative Step-Size Adaptation (CSA)

$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{m} + \sigma\, \boldsymbol{y}_i \\ \boldsymbol{m} &\leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w \end{aligned}$$

Measure the length of the *evolution path*

the pathway of the mean vector $\boldsymbol{m}$ in the generation sequence



↓ decrease σ ↓ increase σ

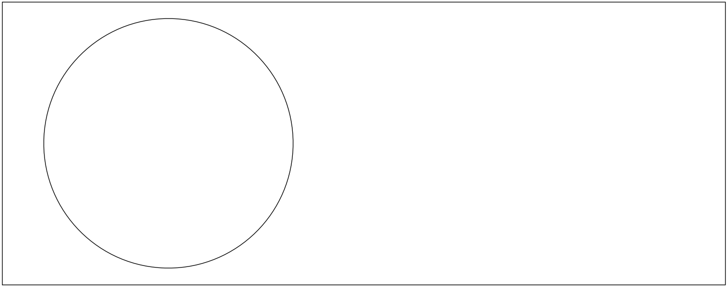
loosely speaking steps are

- perpendicular under random selection (in expectation)
- perpendicular in the desired situation (to be most efficient)

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



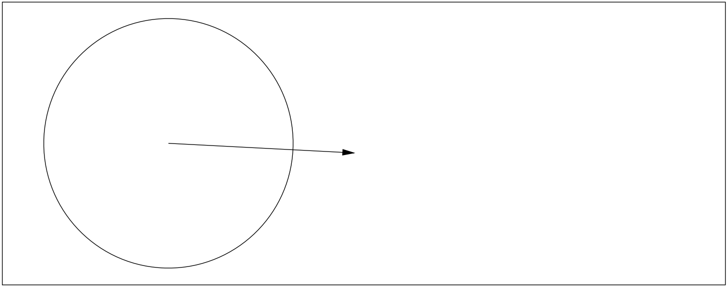
initial distribution, $\mathbf{C} = \mathbf{I}$

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



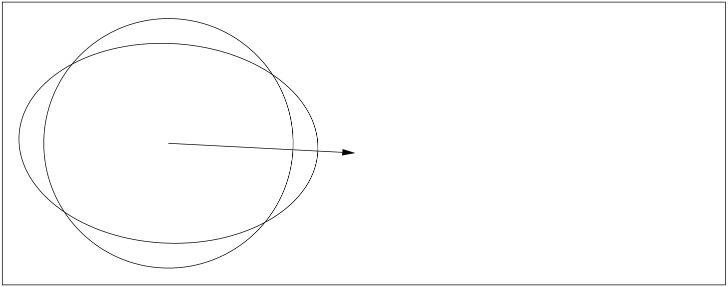
$\boldsymbol{y}_w$, movement of the population mean $\boldsymbol{m}$ (disregarding σ)

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



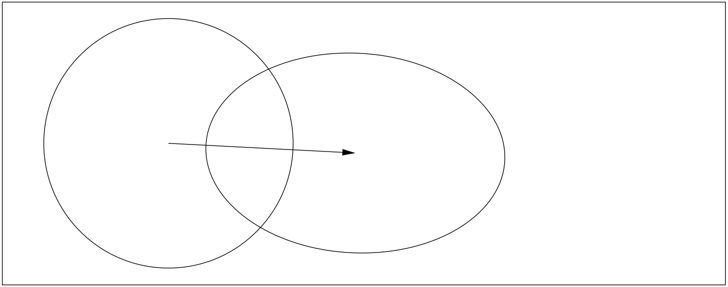
mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,
$\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



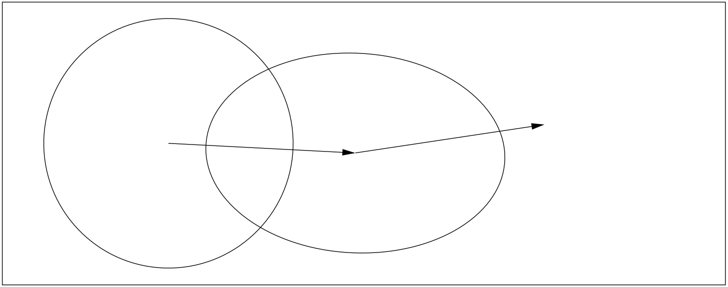
new distribution (disregarding σ)

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



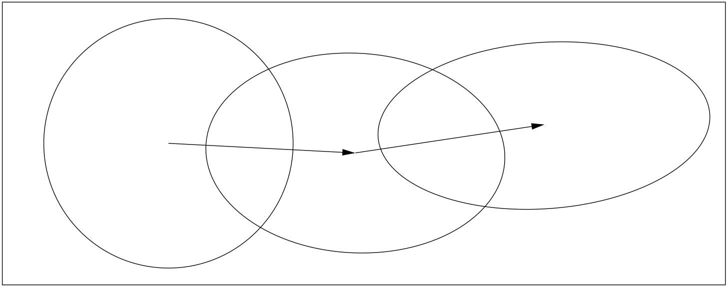
movement of the population mean $\boldsymbol{m}$

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,

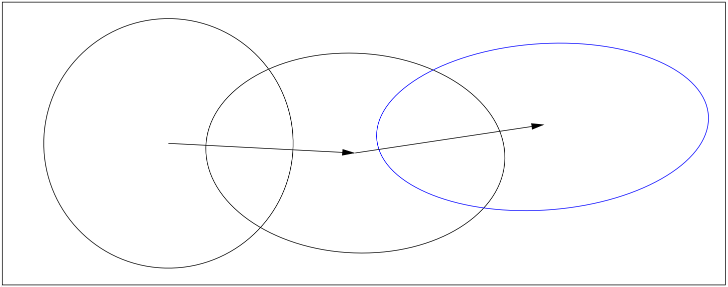
$\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$

- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Covariance Matrix Adaptation

Rank-One Update

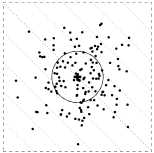
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



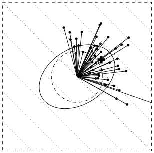
- new distribution: $\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$
- ruling principle: the adaptation increases the probability of successful steps, $\boldsymbol{y}_w$, to appear again

Rank-μ Update

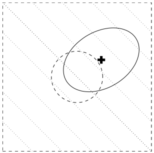
$$\boldsymbol{x}_i = \boldsymbol{m} + \sigma\,\boldsymbol{y}_i, \qquad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}),$$
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma\,\boldsymbol{y}_w \qquad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\,\boldsymbol{y}_{i:\lambda}$$



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



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



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

sampling of $\lambda = 150$ solutions where $\mathbf{C} = \mathbf{I}$ and $\sigma = 1$

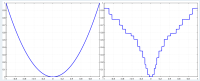
calculating $\mathbf{C}$ from $\mu = 50$ points, $w_1 = \cdots = w_\mu = \frac{1}{\mu}$

new distribution

Remark: the old (sample) distribution shape has a great influence on the new distribution $\longrightarrow$ iterations needed

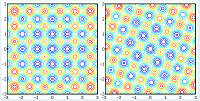
Invariance: Guarantee for Generalization

Invariance properties of CMA-ES



- Invariance to order preserving transformations in function space

  like all comparison-based algorithms



- Translation and rotation invariance

  to *rigid transformations* of the search space

CMA-ES is almost parameterless

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

  but try the *Restart-CMA-ES* Auger & Hansen, 2005

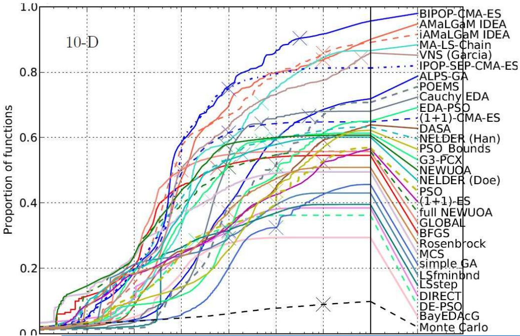
State-of-the-art Results

BBOB – Black-Box Optimization Benchmarking

- ACM-GECCO workshop, in 2009 and 2010
- Set of 25 benchmark functions, dimensions 2 to 40
- With known difficulties (ill-conditioning, non-separability, . . .)
- Noisy and non-noisy versions

Competitors include

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



Contents

Supervised Machine Learning

Context

$$\text{Universe} \rightarrow \text{instance } \mathbf{x}_i \rightarrow \begin{array}{c}\text{Oracle}\\ \downarrow \\ y_i\end{array}$$

Input: Training set $\mathcal{E} = \{(\mathbf{x}_i, y_i), i = 1 \ldots n, \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$

Output: Hypothesis $h : \mathcal{X} \mapsto \mathcal{Y}$

Criterion: Quality of h

Supervised Machine Learning, 2

Definitions

- $\mathcal{E} = \{(\mathbf{x}_i, y_i), \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1 \ldots n\}$
  - Classification : $\mathcal{Y}$ finite *failure/ok*
  - Regression : $\mathcal{Y} \subseteq \mathbb{R}$ *time to failure*
- Hypothesis space $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$

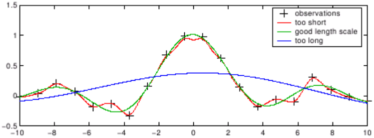
Tasks

- Select $\mathcal{H}$ model selection
- Assess $h \in \mathcal{H}$ expected accuracy $\mathbb{E}[h(\mathbf{x}) \neq y]$
- Find h^* in $\mathcal{H}$ minimizing the error cost in expectation

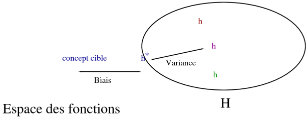
$$h^* = \text{Arg min} \{\mathbb{E}[\ell(h(\mathbf{x}) \neq y), h \in \mathcal{H}\}$$

Dilemma

Fitting the data



Bias variance tradeoff



Statistical Machine Learning

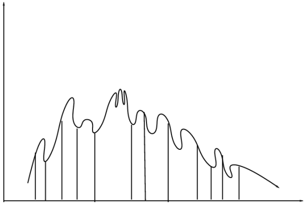
Minimize expected loss

$$\text{Minimize } \mathbb{E}[\ell(h(\mathbf{x}), y)]$$

Principle

If h is well-behaved on the training set, if the training set is "representative" and if h is "regular", then h is well-behaved in expectation.

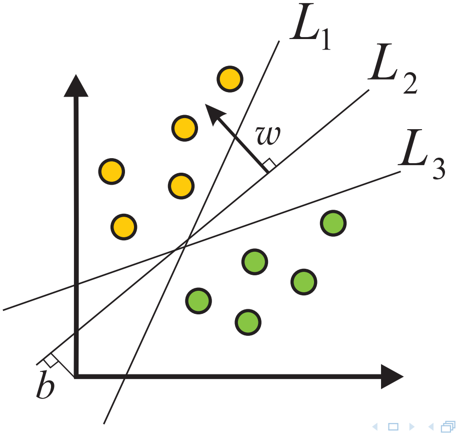
$$E[F] \leq \frac{\sum_{i=1}^{n} F(x_i)}{n} + c(F, n)$$



Linear classification; the noiseless case

$H : X \subset \mathbb{R}^d \mapsto \mathbb{R}$ prediction = $sgn(h(\mathbf{x}))$

$$h(\mathbf{x}) = <\mathbf{w}, \mathbf{x}> + b$$



Linear classification; the noiseless case, 2

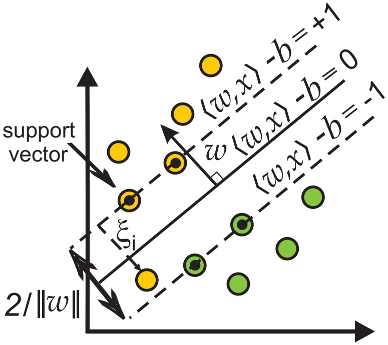
Example $\rightarrow$ Constraint

$$y_i(<\mathbf{w}, \mathbf{x}_i> + b) \geq \textit{ margin } \geq 0$$

Maximize minimum margin $2/||w||$

Formalisation

$$\left\{ \begin{array}{ll} \text{Minimize} & \frac{1}{2}\ ||\mathbf{w}||^2 \\ \text{subject to} & \forall\ i,\ y_i(<\mathbf{w}, \mathbf{x}_i> + b) \geq 1 \end{array} \right.$$



Linear classification; the noiseless case, 3

Primal form

$$\begin{cases} \text{Minimize} & \frac{1}{2}\,||\mathbf{w}||^2 \\ \text{subject to} & \forall\, i,\ y_i(<\mathbf{w}, \mathbf{x}_i> + b) \geq 1 \end{cases}$$

Using Lagrange multipliers:

$$\text{Minimize}_{\mathbf{w},b}\text{max}_\alpha \left\{ \frac{1}{2}\,||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i [y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1] \right\}$$

Dual form

$$\text{Maximize}_\alpha \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \right\}$$

subject to $\alpha_i \geq 0, i = 1 \ldots n$

Optimization: quadratic programming

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x_i}$$

Linear classification; the noisy case

Allow constraint violation; consider slack variables

Primal form

$$\left\{ \begin{array}{ll} \text{Minimize} & \frac{1}{2}\,||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i \\ \text{subject to} & \forall\, i,\ y_i(<\mathbf{w}, \mathbf{x}_i> + b) \geq 1 - \xi_i, \qquad 0 \leq \xi_i \end{array} \right.$$

Lagrange multipliers: Minimize $_{\mathbf{w},b,\xi}$ max$_{\alpha,\beta}$

$$\left\{ \frac{1}{2}\,||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i [y_i(\mathbf{w} \cdot \mathbf{x_i} - b) - 1 + \xi_i] - \sum_{i=1}^{n} \beta_i \xi_i \right\}$$

Dual form

$$\text{Maximize}_\alpha \left\{ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \right\}$$

subject to $0 \leq \alpha_i \leq C, i = 1 \ldots n, \sum \alpha_i y_i = 0$

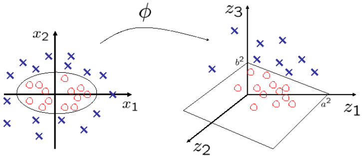
Solution support vectors

$$h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = \sum \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle$$

The kernel trick

Intuition

$$
\begin{array}{lcl}
X & \mapsto & \Omega \\
\Phi : \mathbf{x} = (x_1, x_2) & \mapsto & (x_1^2, \sqrt{2} x_1 . x_2, x_2^2)
\end{array}
$$



$$\phi : (x_1, x_2) \longrightarrow (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$$

$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 = 1 \longrightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1$$

Principle: choose Φ, K such that

$$\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}')$$

The kernel trick, 2

SVM only considers the scalar product

$$h(\mathbf{x}) = \sum_i \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle \qquad \text{linear case}$$
$$h(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) \qquad mboxkerneltrick$$

PROS

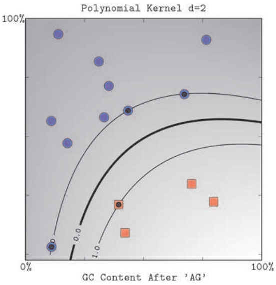
- A rich hypothesis space
- No computational overhead: no explicit mapping on the feature space
- Open problem: kernel design

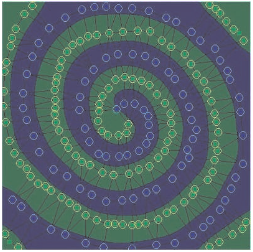
The kernel trick, 3

Kernels

- Polynomial: $k(x_i, x_j) = (\langle x_i, x_j \rangle + 1)^d$
- Gaussian or Radial Basis Function: $k(x_i, x_j) = exp(\frac{\|x_i - x_j\|^2}{2\sigma^2})$
- Hyperbolic tangent: $k(x_i, x_j) = tanh(k \langle x_i, x_j \rangle + c)$

Examples for Polynomial (left) and Gaussian (right) Kernels:





Rank-based SVM

Learning to order things

- 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

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

Contents

Surrogate Models for CMA-ES

Imm-CMA-ES

- Build a full quadratic meta-model around current point
- Weighted by Mahalanobis distance from covariance matric
- Speed-up: a factor of 2-3 for $n \geq 4$
- Complexity: from $O(n^4)$ to $O(n^6)$ (intractable for n>16)
- Rank-invariance is lost

S. Kern et al. (2006). "Local Meta-Models for Optimization Using Evolution Strategies"

Z. Bouzarkouna et al. (2010). “Investigating the Imm-CMA-ES for Large Population Sizes“

Surrogate Models for CMA-ES, cont

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
- Note: no use of information from current state of CMA

T. Runarsson (2006). "Ordinal Regression in Evolutionary Computation"

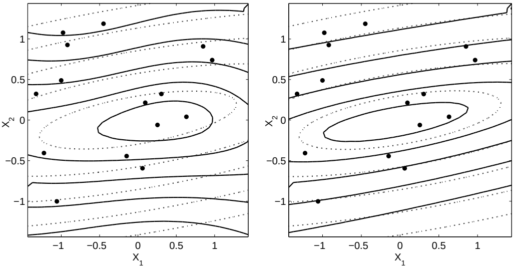
ACM Algorithm

- Use C from CMA-ES as Gaussian kernel

I. Loschilov et al. (2010). "Comparison-based optimizers need comparison-based surrogates”

Model Learning

Non-separable Ellipsoid problem



Rank-SVM regression in original coordinate system

Rank-SVM regression in transformed coordinate system given by current covariance matrix C and mean m:

$$x' = C^{-\frac{1}{2}}(x - m)$$

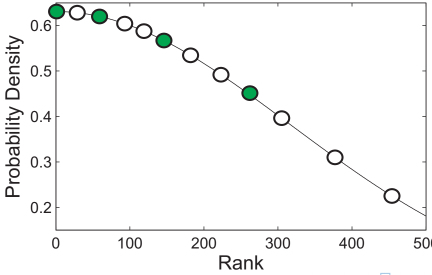
Using the Surrogate Model

- Optimization: Significant Speed-Up . . . if global and accurate model
- Filtering: "Guaranteed" Speed-Up

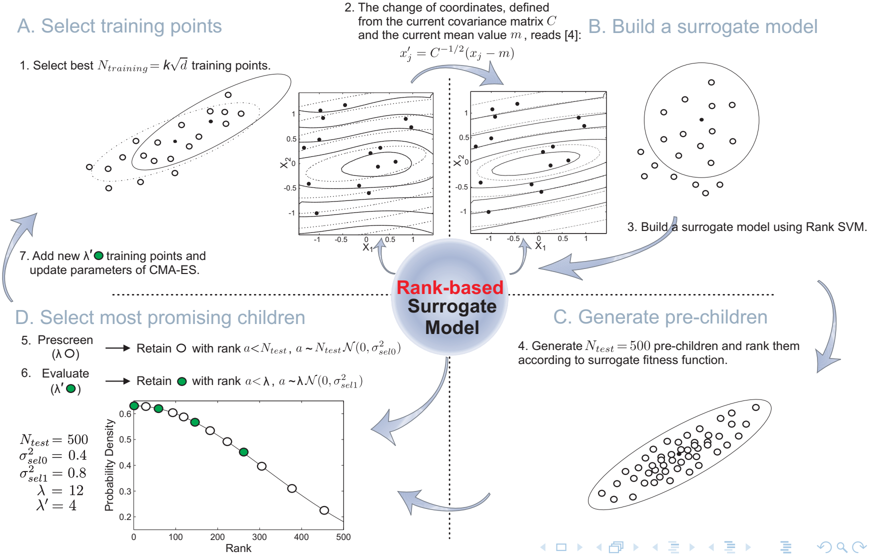
Prescreen (λ ○) → Retain ○ with rank $a < N_{test}$, $a \sim N_{test}\mathcal{N}(0, \sigma^2_{sel0})$

Evaluate (λ' ●) → Retain ● with rank $a < \lambda$, $a \sim \lambda\mathcal{N}(0, \sigma^2_{sel1})$

$$N_{test} = 500$$
$$\sigma^2_{sel0} = 0.4$$
$$\sigma^2_{sel1} = 0.8$$
$$\lambda = 12$$
$$\lambda' = 4$$



ACM-ES Optimization Loop



Parameters

SVM Learning

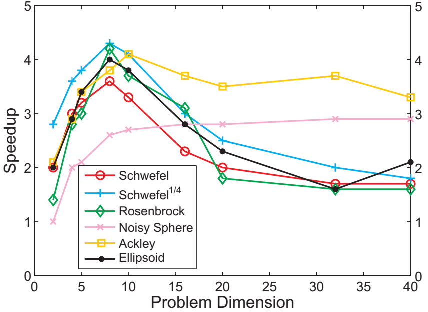
- Number of training points: $N_{training} = 30\sqrt{d}$ for all problems, except Rosenbrock and Rastrigin, where $N_{training} = 70\sqrt{(d)}$
- Number of iterations: $N_{iter} = 50000\sqrt{d}$
- Kernel function: RBF function with σ equal to the average distance of the training points
- The cost of constraint violation: $C_i = 10^6(N_{training} - i)^{2.0}$

Offspring Selection

- Number of test points: $N_{test} = 500$
- Number of evaluated offsprings: $\lambda' = \frac{\lambda}{3}$
- Offspring selection pressure parameters: $\sigma^2_{sel0} = 2\sigma^2_{sel1} = 0.8$

Results

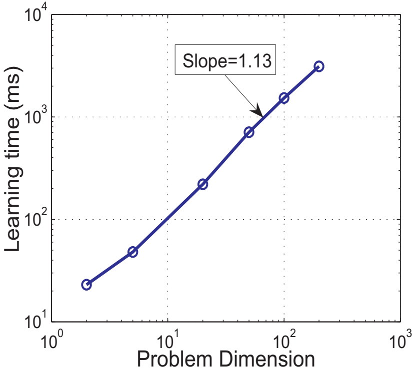
Speed-up



Function	n	λ	λ′	e	ACM-ES	spu	CMA-ES
Schwefel	10	10	3		801 ± 36	3.3	2667 ± 87
	20	12	4		3531 ± 179	2.0	7042 ± 172
	40	15	5		13440 ± 281	1.7	22400 ± 289
Schwefel$^{1/4}$	10	10	3		1774 ± 37	4.1	7220 ± 206
	20	12	4		6138 ± 82	2.5	15600 ± 294
	40	15	5		22658 ± 390	1.8	41534 ± 466
Rosenbrock	10	10	3		2059 ± 143 (0.95)	3.7	7669 ± 691 (0.90)
	20	12	4		11793 ± 574 (0.75)	1.8	21794 ± 1529
	40	15	5		49750 ± 2412 (0.9)	1.6	82043 ± 3991
NoisySphere	10	10	3	0.15	766 ± 90 (0.95)	2.7	2058 ± 148
	20	12	4	0.11	1361 ± 212	2.8	3777 ± 127
	40	15	5	0.08	2409 ± 120	2.9	7023 ± 173
Ackley	10	10	3		892 ± 28	4.1	3641 ± 154
	20	12	4		1884 ± 50	3.5	6641 ± 108
	40	15	5		3690 ± 80	3.3	12084 ± 247
Ellipsoid	10	10	3		1628 ± 95	3.8	6211 ± 264
	20	12	4		8250 ± 393	2.3	19060 ± 501
	40	15	5		33602 ± 548	2.1	69642 ± 644
Rastrigin	5	140	70		23293 ± 1374 (0.3)	0.5	12310 ± 1098 (0.75)

Results
Learning Time

Cost of model learning/testing increases quasi-linearly with *d* on Sphere function:



ACM-ES: conclusion

ACM-ES

- From 2 to 4 times faster on Uni-Modal Problems
- Invariance to rank-preserving transformations preserved
- Computation complexity is $O(n)$
- Available online at
  `http://www.lri.fr/~ilya/publications/ACMESppsn2010.zip`

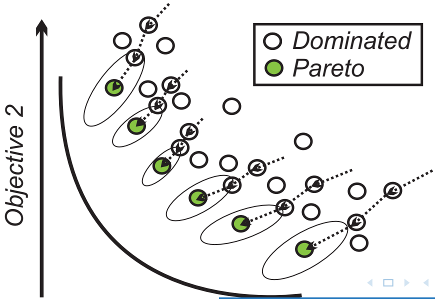
Open Issues

- Extention to multi-modal optimization
- On-line adaptation of selection pressure and surrogate model complexity

Contents

Multi-objective CMA-ES (MO-CMA-ES)

- MO-CMA-ES = μ_{mo} independent (1+1)-CMA-ES.
- Each (1+1)-CMA samples new offspring. The size of the temporary population is $2\mu_{mo}$.
- Only μ_{mo} best solutions should be chosen for new population after the hypervolume-based non-dominated sorting.
- Update of CMA individuals takes place.

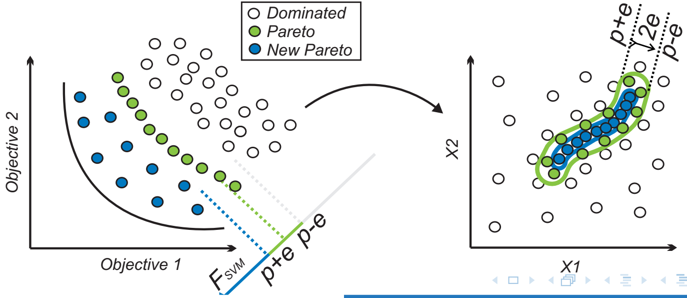


A Multi-Objective Surrogate Model

Rationale

- Rationale: find a unique function $F(x)$ that defines the aggregated quality of the solution x in multi-objective case.
- Idea originally proposed using a mixture of One-Class SVM and regression-SVM[a]

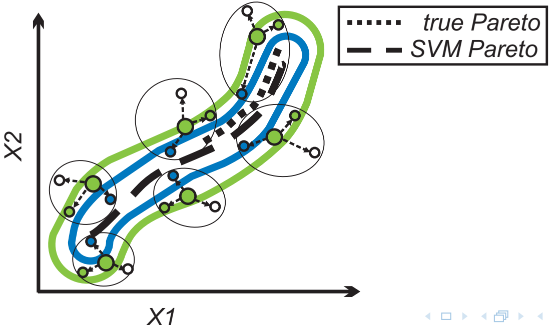
[a] I. Loshchilov, M. Schoenauer, M. Sebag (GECCO 2010). "A Mono Surrogate for Multiobjective Optimization"



Unsing the Surrogate Model

Filtering

- Generate N_{inform} pre-children
- For each pre-children A and the nearest parent B calculate $Gain(A, B) = F_{svm}(A) - F_{svm}(B)$
- New children is the point with the maximum value of $Gain$

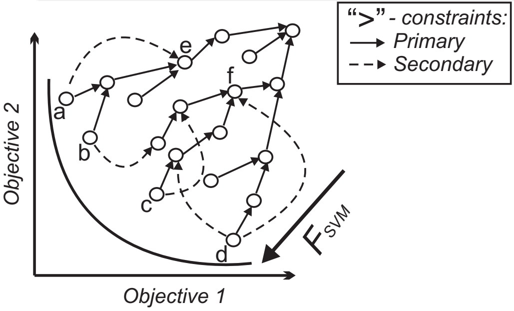


Dominance-Based Surrogate

Using Rank-SVM

Which ordered pairs?

- Considering **all** possible $\succ$ relations may be too expensive.
- Primary constraints: x and its nearest dominated point
- Secondary constraints: any 2 points not belonging to the same front (according to non-dominated sorting)



All primary constraints, and a limited number of secondary constraints

Dominance-Based Surrogate (2)

Construction of the surrogate model

- Initialize archive Ω_{active} as the set of **Primary constraints**, and $\Omega_{passive}$ as the set of **Secondary constraints**.
- Learn the model for $1000\,|\Omega_{active}|$ iterations.
- Add the most violated passive contraint from $\Omega_{passive}$ to Ω_{active} and optimize the model for $10\,|\Omega_{active}|$ iterations.
- Repeat the last step 0.1$|\Omega_{active}|$ times.

Experimental Validation

Parameters

Surrogate Models

- ASM - aggregated surrogate model based on One-Class SVM and Regression SVM
- RASM - proposed Rank-based SVM

SVM Learning

- Number of training points: at most $N_{training} = 1000$ points
- Number of iterations: $1000\,|\Omega_{active}|$ + $|\Omega_{active}|^2 \approx 2N^2_{training}$
- Kernel function: RBF function with σ equal to the average distance of the training points
- The cost of constraint violation: $C = 1000$

Offspring Selection

- Number of pre-children: $p = 2$ and $p = 10$

Experimental Validation

Comparative Results

ΔHtarget	1	0.1	0.01	1e-3	1e-4	1	0.1	0.01	1e-3	1e-4
	ZDT1					ZDT2				
Best	1100	3000	5300	7800	38800	1400	4200	6600	8500	32700
S-NSGA-II	1.6	2	2	2.3	1.1	1.8	1.7	1.8	2.3	1.2
ASM-NSGA p=2	1.2	1.5	1.4	1.5	1.5	1.2	1.2	1.2	1.4	**1**
ASM-NSGA p=10	**1**	**1**	**1**	**1**	.	**1**	**1**	**1**	**1**	.
RASM-NSGA p=2	1.2	1.4	1.4	1.6	**1**	1.3	1.2	1.2	1.5	**1**
RASM-NSGA p=10	**1**	1.1	1.1	1.5	.	1.1	**1**	**1**	1.2	.
MO-CMA-ES	16.5	14.4	12.3	11.3	.	14.7	10.7	10	10.1	.
ASM-MO-CMA p=2	6.8	8.5	8.3	8	.	5.9	8.2	7.7	7.5	.
ASM-MO-CMA p=10	6.9	10.1	10.4	12.1	.	5	.	.	.	.
RASM-MO-CMA p=2	5.1	7.7	7.6	7.4	.	5.2	.	.	.	.
RASM-MO-CMA p=10	3.6	4.3	4.9	7.2	.	3.2	.	.	.	.
	IHR1					IHR2				
Best	500	2000	35300	41200	50300	1700	7000	12900	52900	.
S-NSGA-II	1.6	1.5	.	.	.	1.1	3.2	6.2	.	.
ASM-NSGA p=2	1.2	1.3	.	.	.	**1**	3.9	4.9	.	.
ASM-NSGA p=10	**1**	1.5	.	.	.	1.4	6.4	4.6	.	.
RASM-NSGA p=2	1.2	1.2	.	.	.	1.5	.	.	.	.
RASM-NSGA p=10	**1**	**1**	.	.	.	1.2	5.1	4.8	.	.
MO-CMA-ES	8.2	6.5	1.1	1.2	1.2	5.8	2.7	2.1	**1**	.
ASM-MO-CMA p=2	4.6	2.9	**1**	**1**	**1**	3.1	1.6	1.4	1.1	.
ASM-MO-CMA p=10	9.2	6.1	1.3	1.2	.	5.9	2.6	2.4	.	.
RASM-MO-CMA p=2	2.6	2.3	2.4	2.1	.	2.2	**1**	**1**	.	.
RASM-MO-CMA p=10	1.8	1.9	.	.	.	.	.	.	.	.

ASM and Rank-based ASM applied on top of NSGA-II (with hypervolume secondary criterion) and MO-CMA-ES, on ZDT and IHR functions.

N = How many more true evaluations than best performer

Discussion

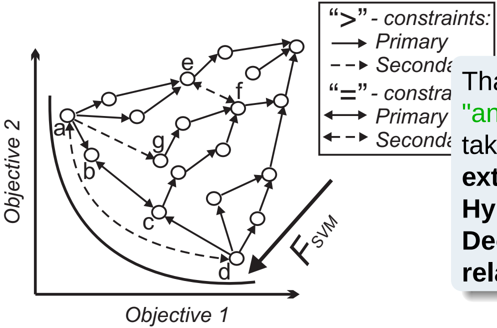
Results on **ZDT** and **IHR** problems

- Rank-SVM versions are
  - 1.5 times faster for $p = 2$
  - 2-5 for $p = 10$

  before the algorithm reaches nearly-optimal Pareto points
- premature convergence of approximation of optimal μ-distribution:
  the **surrogate** model only enforces **convergence** toward Pareto front, but does **not** care about **diversity**.

Dominance-based Surrogate: Conclusion

- The proposed aggregated surrogate model is invariant to $\succ$ preserving transformation of the objective functions.
- The speed-up is significant, but limited to the convergence to the optimal Pareto front.



Thanks to the flexibility of SVM, "any" kind of preference could be taken into account:
extreme points, "=" relations, Hypervolume Contribution, Decision Maker - defined $\succ$ relations.

Machine Learning for Optimization: Discussion

Learning about the landscape — easy

- Using available samples
- Using prior knowledge / constraints

...using it to speed-up the search — very doable; more difficult

- Dilemma Exploration vs Exploitation
- Multi-modal optimization