Gaussian processes in evolutionary black-box search

Lukáš Bajer\textsuperscript{1,2}, Andrej Kudinov\textsuperscript{3}

\textsuperscript{1}Faculty of Mathematics and Physics, Charles University,
\textsuperscript{2}Institute of Computer Science, Czech Academy of Sciences, and
\textsuperscript{3}Faculty of Information Technologies, Czech Technical University

Prague, Czech Republic

21. 5. 2015
Contents

1. Optimization of expensive black-box functions
   - Model-based methods
   - Gaussian Processes

2. MGSO
   - Introduction to MGSO
   - Experimental results on Niching functions
   - Experimental results on BBOB functions

3. Surrogate CMA-ES
   - CMA-ES
   - Surrogate CMA-ES
   - Experimental results
optimization (minimization) is finding such \( x^* \in \mathbb{R}^n \) that

\[
f(x^*) = \min_{\forall x \in \mathbb{R}^n} f(x)
\]

“near-optimal” solution is usually sufficient
Optimization of black-box functions

- black-box functions

- only evaluation of the function value, no derivatives or gradients $\rightarrow$ no gradient methods available

- we consider continuous domain: $x \in \mathbb{R}^n$
empirical function:

- assessing the function-value via an experiment (measuring, intensive calculation, evaluating a prototype)
- evaluating such functions are expensive (time and/or money)
- search cost $\sim$ the number of function evaluations
evolutionary algorithms (EA’s) used in this study
EA’s often manage to escape from local optima
but usually use many function evaluations
(at least in comparison with gradient methods like BFGS)
EA’s for empirical black-box optimization

what can help with decreasing the number of function evaluations:

- utilize **already measured values**
  (at least prevent measuring the same thing twice)
- learn **the shape** of the function landscape
  or learn the (global) **gradient** or step direction & size
several methods are used in order to **decrease**
the number of objective function **evaluations** needed by EA’s

1. Surrogate modelling
2. Estimation of Distribution Algorithms (EDA’s)
3. Efficient Global Optimization (EGO)
Surrogate modelling

- technique which builds an **approximating model** of the fitness function landscape
- the model provides a **cheap and fast**, but also **inaccurate** replacement of the fitness function for **part of the population**
- **inaccurate** approximating model can **deceive** the optimizer

from the EUMC presentation “Viscous optimization of bulkers and tankers” (Mattia Brenner, June 17, 2010)
Estimation of Distribution Algorithms (EDA)

- the model represents a **distribution** of solutions in the **input space**
- new candidate solutions are generated via **sampling** the model
- **better solutions** are selected for being represented by the model in the next generation
Stochastic search of Evolutionary algorithms

Stochastic black box search

1. Initialize distribution parameters $\theta$
2. Set population size $\lambda \in \mathbb{N}$
3. While not terminate
   1. Sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
   2. Evaluate $x_1, \ldots, x_\lambda$ on $f$
   3. Update parameters $\theta$

(A. Auger, Tutorial CMA-ES, GECCO 2013)

- Schema of most of the evolutionary algorithms and EDA algorithms
Gaussian Process

GP is a stochastic approximation method based on Gaussian distributions.

GP can express uncertainty of the prediction in a new point $x$: it gives a probability distribution of the output value.
given a set of \( N \) training points \( \mathbf{X}_N = (x_1 \ldots x_N)^T \), \( x_i \in \mathbb{R}^d \), and measured values \( \mathbf{y}_N = (y_1, \ldots , y_N)^T \) of a function \( f \) being approximated

\[ y_i = f(x_i), \quad i = 1, \ldots , N \]

GP considers vector of these function values as a sample from \( N \)-variate Gaussian distribution

\[ \mathbf{y}_N \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_N) \]
Gaussian Process

\[ \mathbf{X} = (\mathbf{x}_1 \ldots \mathbf{x}_N)^\top, \quad \mathbf{x}_i \in \mathbb{R}^d \]
\[ \mathbf{y} = (y_1, \ldots, y_N)^\top, \quad y_i = f(\mathbf{x}_i) \]

\( \mathbf{y} \in \mathbb{R}^N \) considered to be a realisation of a \( N \)-dimensional Gaussian distribution with a covariance matrix \( \mathbf{C}_N \) and zero mean

\[ \mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_N) \]

Covariance \( \mathbf{C}_N \) is determined by

- covariance function \( \text{cov}(\mathbf{x}_i, \mathbf{x}_j) \) and its hyperparameters
- training data points \( \mathbf{X}_N \)

forming the density of the Gaussian

\[ p(\mathbf{y}|\mathbf{X}_N) \]
Gaussian Process covariance

covariance $C_N$ is given by

$$C_N = K + \sigma^2 I$$

where $K$ is a matrix of covariance function values and $\sigma^2$ is the signal noise.

Covariance functions are defined on pairs from the input space

$$(K)_{ij} = \text{cov}(x_i, x_j), \quad x_{i,j} \in \mathbb{R}^d$$

expressing the degree of correlations between two points’ values; typically decreasing functions on two points distance
Gaussian Process covariance

The most frequent covariance function is *squared-exponential*

\[(K)_{ij} = \text{cov}^{SE}(x_i, x_j) = \theta \exp \left( \frac{-1}{2\ell^2} (x_i - x_j)^\top (x_i - x_j) \right)\]

with the parameters (usually fitted by MLE)
- \(\theta\) – signal variance (scales the correlation)
- \(\ell\) – characteristic length scale
Another usual option is *Matérn covariance*, which is for
\[ r = (x_i - x_j) \]

\[
(K)_{ij} = \text{cov}^{\text{Matern}}_{\nu=5/2}(r) = \theta \left( 1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2} \right) \exp \left( -\frac{\sqrt{5}r}{\ell} \right).
\]

with the parameters (same as for squared exponential)
- \( \theta \) – signal variance
- \( \ell \) – characteristic length scale
Gaussian Process covariance

from Michael Osborne: An Introduction to Fitting Gaussian Processes to Data (presentation)
Gaussian Process prediction

Making predictions
Prediction $y^*$ in a new point $x^*$ is made by adding this new point to the matrix $X_N$ and vector $y_N$.

This gives an $(N+1)$-dimensional Gaussian with density

$$p(y_{N+1} \mid X_{N+1}) = \frac{1}{\sqrt{(2\pi)^{N+1} \det(C_{N+1})}} \exp\left(-\frac{1}{2} y_{N+1}^\top C_{N+1}^{-1} y_{N+1}\right)$$

where $C_{N+1}$ is the covariance matrix

$$C_{N+1} = \begin{pmatrix} C_N & k \\ k^\top & \kappa + \sigma \end{pmatrix}$$

which is $C_N$ extended with
- $k$ – covariances between $x^*$ and $X_N$
- $\kappa + \sigma$ – variance of the new point itself (with added signal noise)
Making predictions
Because $y_N$ is known and the inverse $C_{N+1}^{-1}$ can be expressed using inverse of the training covariance $C_N^{-1}$, the density in a new point marginalize to 1D Gaussian density

$$p(y^* | X_{N+1}, y_N) \propto \exp \left( -\frac{1}{2} \frac{(y^* - \hat{y}_{N+1})^2}{s_{y_{N+1}}^2} \right)$$

with the mean and variance given by

$$\hat{y}_{N+1} = k^\top C_N^{-1} y_N,$$
$$s_{y_{N+1}}^2 = \kappa - k^\top C_N^{-1} k.$$
Efficient Global Optimization (EGO)

**EGO**

- **Needed**: specific kind of surrogate model which can express **uncertainty** of the prediction in a new point.
- EGO uses Kriging / **Gaussian processes** (GP).
- For any given input $x$, it gives a **probability distribution** of the output value.
Efficient Global Optimization (EGO)

- The resulting output for a specified input $x$ is a 1-D Gaussian with
  - mean at the predicted value
  - standard deviation expressing uncertainty of this prediction

- Probability of improvement (PoI) is the probability that the function value will be lower than a specified target $T$

$$
\text{PoI}_T(x) = \Phi\left(\frac{T - \hat{f}(x)}{\hat{\sigma}(x)}\right) = P(\hat{f}(x) \leq T)
$$
Efficient Global Optimization (EGO)

EGO

1. dataset $D \leftarrow$ generate a random initial sample and evaluate it
2. build a GP/Kriging model $\hat{f}$ using $D$
3. $x_{\text{max}} \leftarrow$ maximize $\text{PoI}(x)$ (from the GP model)
4. evaluate $x$
5. $D = D \cup \{x\}$
6. repeat from step 2

maximizing the PoI is a way of balancing between exploration of the input space and exploitation of the local optima.
main idea of MGSO:

- consider the PoI(x) to be a function proportional to a probability density
- sample this density to get a population of candidate solutions (like EDAs)

motivation: not getting trapped in local minima while exploring the search space
**Model Guided Sampling Optimization**

**EGO**  **MGSO**

1. dataset $D \leftarrow$ generate a random initial sample and evaluate it
2. build a GP model $\hat{f}$ using $D$
3. $x_{\text{max}} \leftarrow$ maximize $\text{PoI}(x)$ (from the GP model)
4. :
5. sample the $\text{PoI}(x)$ resulting in a new population $P$
6. find the minimum $x_{\text{min}} = \arg \min \hat{f}(x)$ of the GP and add to $P$
7. evaluate $x$
8. $D = D \cup \{x\}$
9. evaluate all $x \in P$
10. $D = D \cup P$
11. repeat from step 2
Probability of improvement

PoI of 2D Rastrigin, $N = 40$

\[ \text{PoI}_T(x) = \Phi \left( \frac{T - \hat{f}(x)}{\hat{\sigma}(x)} \right) = P(\hat{f}(x) \leq T) \]

- no explicit formula for \( \text{PoI}(x, T, \hat{f}) \)
- improvement in a new point \( x \) is probable when
  - not many samples around, i.e. large \( \sigma \) at \( x \)
  - promising area is searched, i.e. low \( f(x) \)
- different way of generating new points
  - EGO: find the maximum of \( \text{PoI} \) \( \rightarrow \) one solution
  - MGSO: sample the \( \text{PoI} \) \( \rightarrow \) multiple solutions
Numerical instability

- adding a new sampled point \((\mathbf{x}', f(\mathbf{x}'))\) can cause the Gaussian process’ covariance matrix to be close to semi-positive indefinite
- such points are rejected already during sampling and new points are sampled instead
Degeneration of PoI

- PoI degenerates close to a discrete distribution in later phases of the optimization
- happens when the input space is well explored
- partially solved by cropping the input space to a region around the so-far optimum and rescaling the input space to get better numerical resolution for sampling
[ ... to be continued ... ]
MGSO was also tested on three benchmark functions from the BBOB testbed

- sphere
- Rosenbrock
- Rastrigin

three dimensionalities: 2D, 5D, 10D

compared to CMA-ES and Tomlab’s EGO implementation

(C) Wikimedia Commons
Results in 2-D

- Sphere, 2D
- Rosenbrock, 2D
- Rastrigin, 2D

medians and quartiles of the best-found fitness from 15 runs
MGSO, CMA-ES, EGO, older MGSO without ARD
Results in 5-D

medians and quartiles of the best-found fitness from 15 runs

MGSO, CMA-ES, EGO, older MGSO without ARD
Results in 10-D

medians and quartiles of the best-found fitness from 15 runs

MGSO, CMA-ES, EGO, older MGSO without ARD
Stochastic search of Evolutionary algorithms

Stochastic black box search

- initialize distribution parameters $\theta$
- set population size $\lambda \in \mathbb{N}$
- while not terminate
  1. sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
  2. evaluate $x_1, \ldots, x_\lambda$ on $f$
  3. update parameters $\theta$

(A. Auger, Tutorial CMA-ES, GECCO 2013)

- schema of most of the evolutionary strategies (and EDA algorithms)
- as well as CMA-ES (Covariance Matrix Adaptation ES) – current state of the art in continuous optimization
The CMA-ES

**Input:** \( m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, \lambda \in \mathbb{N} \)

**Initialize:** \( C = I \) (and several other parameters)

**Set** the weights \( w_1, \ldots, w_\lambda \) appropriately

**while not terminate**

1. \( x_i = m + \sigma y_i, \quad y_i \sim N(0, C), \quad \text{for } i = 1, \ldots, \lambda \) sampling

2. \( m \leftarrow \sum_{i=1}^{\mu} w_i x_{i:}\lambda = m + \sigma y_w \quad \text{where } y_w = \sum_{i=1}^{\mu} w_i y_{i:}\lambda \) update mean

3. update \( C \)

4. update step-size \( \sigma \)
Covariance matrix adaptation

- **eigenvectors** of the covariance matrix $C$ are the principle components – the principle axes of the mutation ellipsoid
- CMA-ES learns and updates a new **Mahalanobis metric**
- successively approximates the **inverse Hessian** on quadratic functions
  - transforms ellipsoid function into sphere function
  - it somehow holds for other functions, too (up to some degree)
Is CMA-ES the best for everything?

- CMA-ES is state-of-the-art optimization algorithm, especially for rugged and ill-conditioned objective functions
- however, **not the fastest** if we can afford only **very few** objective function evaluations
- what we have already seen: use a **surrogate model**!
- however, original evaluated solutions are available only along the **search path**
- solution: construct **local surrogate models**
The Surrogate CMA-ES

Input: \( m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, \lambda \in \mathbb{N} \)
Initialize: \( C = I \) (and several other parameters)
Set the weights \( w_1, \ldots, w_\lambda \) appropriately

while not terminate

1. \( x_i = m + \sigma y_i, \quad y_i \sim N(0, C), \quad \text{for } i = 1, \ldots, \lambda \) \( \quad \text{sampling} \)
2. evaluate with the original fitness & build a model
3. evaluate with the model fitness
4. update \( m, C, \sigma \)
The Surrogate CMA-ES

Input: $g$ (generation), $f_M$ (model), $A$ (archive), $n_{REQ}$, $\sigma$, $\lambda$, $m$, $C$

1: $x_k \sim \mathcal{N}(m, \sigma^2C)$ \quad $k = 1, \ldots, \lambda$ \quad \{CMA-ES sampling\}
2: if $g$ is original-evaluated then
3: $y_k \leftarrow f(x_k)$ \quad $k = 1, \ldots, \lambda$ \quad \{fitness evaluation\}
4: $A = A \cup \{(x_k, y_k)\}^\lambda_{k=1}$
5: if $|X| \geq n_{REQ}$ then
6: $X \leftarrow \text{TransformToTheEigenvectorBasis}(X, \sigma, C)$
7: $f_M \leftarrow \text{trainModel}(X, y)$
8: end if
9: else
10: $X \leftarrow \text{TransformToTheEigenvectorBasis}(X, \sigma, C)$
11: $y_k \leftarrow f_M(x_k)$ \quad $k = 1, \ldots, \lambda$ \quad \{model evaluation\}
12: end if

Lukáš Bajer, Andrej Kudinov
Optimization of expensive black-box functions

**MGSO**

**Surrogate CMA-ES**

CMA-ES

Surrogate CMA-ES

Experimental results

[ ... to be continued ... ]
Experimental results on BBOB (5 D)
Experimental results on BBOB (10 D)
Experimental results on BBOB (20 D)

- **f1-24,20-D**
- **log10 of (# f-evals / dimension)**
- Proportion of function+target pairs
- **GP5-CMAES**
- **GP1-CMAES**
- **CMA-ES**
- **saACMES**
- **best 2009**

Lukáš Bajer, Andrej Kudinov
The best results on subset of BBOB (5 D)
The best results on subset of BBOB (20 D)

![Graph showing optimization results for different methods: CMA-ES, Surrogate CMA-ES, GP5-CMAES, GP1-CMAES, and saACMES. The x-axis represents the log10 of the number of function evaluations per dimension, and the y-axis represents the proportion of function-target pairs. The best 2009 results are also indicated.]
Results on #3 Rastrigin separable
Results on #8 Rosenbrock

![Graph showing results on #8 Rosenbrock](image_url)
Results on #10 Ellipsoid function
Thank you!

bajer@cs.cas.cz