Stochastic Optimization

Marc Schoenauer

DataScience @ LHC Workshop 2015

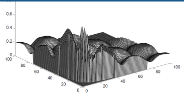
Content

Content

Stochastic Optimization

Hypotheses

- Search Space Ω — with some topological structure
- Objective function $\mathcal{F}$ — assume some weak regularity



Hill-Climbing

- Randomly draw $\boldsymbol{x}_0 \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0)$ — Initialisation
- Until(happy)
  - $\boldsymbol{y} =$ Best neighbor$(\boldsymbol{x}_t)$ — neighbor structure on Ω
  - Compute $\mathcal{F}(\boldsymbol{y})$
  - If $\mathcal{F}(\boldsymbol{y}) \succ F(\boldsymbol{x}_t)$ then $\boldsymbol{x}_{t+1} = \boldsymbol{y}$ — accept if improvement
    else $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$

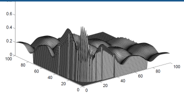
Comments

- Find *closest* local optimum — defined by neighborhood structure
- Iterate from different $\boldsymbol{x}_0$'s — Until(very happy)

Stochastic Optimization

Hypotheses

- Search Space Ω with some topological structure
- Objective function $\mathcal{F}$ assume some weak regularity



Stochastic (Local) Search

- Randomly draw $\boldsymbol{x}_0 \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0)$ Initialisation
- Until(happy)
  - $\boldsymbol{y} =$ Random neighbor$(\boldsymbol{x}_t)$ neighbor structure on Ω
  - Compute $\mathcal{F}(\boldsymbol{y})$
  - If $\mathcal{F}(\boldsymbol{y}) \succ F(\boldsymbol{x}_t)$ then $\boldsymbol{x}_{t+1} = \boldsymbol{y}$ accept if improvement
    else $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$

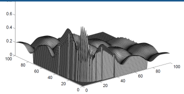
Comments

- Find one *close* local optimum defined by neighborhood structure
- Iterate, leaving current optimum Iterated Local Search

Stochastic Optimization

Hypotheses

- Search Space Ω — with some topological structure
- Objective function $\mathcal{F}$ — assume some weak regularity



Stochastic (Local) Search – alternative viewpoint

- Randomly draw $\boldsymbol{x}_0 \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0)$ — Initialisation
- Until(happy)
  - $\boldsymbol{y} = \text{Move}(\boldsymbol{x}_t)$ — stochastic variation on Ω
  - Compute $\mathcal{F}(\boldsymbol{y})$
  - If $\mathcal{F}(\boldsymbol{y}) \succ F(\boldsymbol{x}_t)$ then $\boldsymbol{x}_{t+1} = \boldsymbol{y}$ — accept if improvement

    else $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$

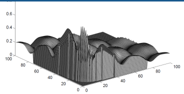
Comments

- Find one *close* local optimum — defined by the Move operator
- Iterate, leaving current optimum — Iterated Local Search

Stochastic Optimization

Hypotheses

- Search Space Ω — with some topological structure
- Objective function $\mathcal{F}$ — assume some weak regularity



Stochastic Search — aka Metaheuristics

- Randomly draw $\boldsymbol{x}_0 \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0)$ — Initialisation
- Until(happy)
  - $\boldsymbol{y} = \text{Move}(\boldsymbol{x}_t)$ — stochastic variation on Ω
  - Compute $\mathcal{F}(\boldsymbol{y})$
  - $\boldsymbol{x}_{t+1} = \text{Select}(\boldsymbol{x}_t, \boldsymbol{y})$ — stochastic selection, biased toward better points

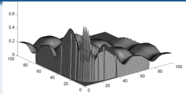
Comments

- Can escape local optima — e.g., Select = Boltzman selection
- Iterate ?

Stochastic Optimization

Hypotheses

- Search Space Ω — with some topological structure
- Objective function $\mathcal{F}$ — assume some weak regularity



Population-based Metaheuristics

- Randomly draw $\boldsymbol{x}_0^i \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0^i)$, $i = 1, \ldots, \mu$ — Initialisation
- Until(happy)
  - $\boldsymbol{y}^i = \text{Move}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu)$, $i = 1, \ldots, \lambda$ — stochastic variations on Ω
  - Compute $\mathcal{F}(\boldsymbol{y}^i)$, $i = 1, \ldots, \lambda$
  - $(\boldsymbol{x}_{t+1}^1, \ldots, \boldsymbol{x}_{t+1}^\mu) = \text{Select}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ — $(\mu + \lambda)$ selection

    $= \text{Select}(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ — (μ, λ) selection

Evolutionary Metaphor: 'natural' selection and 'blind' variations

- $\boldsymbol{y}^i = \text{Move}_m(\boldsymbol{x}_t^i)$ for some i: mutation
- $\boldsymbol{y}^i = \text{Move}_c(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j)$ for some (i, j): crossover

Metaheuristics: an Alternative Viewpoint

- Population and variation operators define a distribution on Ω
- $\longrightarrow$ directly evolve a parameterized distribution $P(\boldsymbol{\theta})$

Distribution-based Metaheuristics

- Initialize distribution parameters $\boldsymbol{\theta_0}$
- Until(happy)
  - Sample distribution $P(\boldsymbol{\theta_t}) \to \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda \in \Omega$
  - Evaluate $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda$ on $\mathcal{F}$
  - Update parameters $\boldsymbol{\theta_{t+1}} \leftarrow F_\theta(\boldsymbol{\theta_t}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda, \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\lambda))$

includes selection

Covers

- Estimation of Distribution Algorithms
- Population-based Metaheuristics — Evolutionary Algorithms
  Particle Swarm Optimization, Differential Evolution, ...
- Deterministic algorithms

This is (mostly) Experimental Science

Theory

- lagging behind practice
- though rapidly catching up in recent years
- not discussed here

Results need to be experimentally validated

Experiments

Validation relies on

- grounded statistical validation — CPU costly
- informed (meta)parameter setting/tuning — CPU costly ++
- reproducibility — Open Science

From Solution Space to Search Space

Choices

- Objective function
- Search space — representation
- and move operators / parameterized distribution

Approximation bias vs optimization error

This talk

- Non-convex, non-smooth objective — parametric continuous optimization
  $\rightarrow$ Rank-based ML — seeded identification of influencers
- Search spaces and crossover operators — the TSP
- Non-parametric representations — exploration vs optimization

Content

Content

Stochastic Optimization Templates

Population-based Metaheuristics

- Randomly draw $\boldsymbol{x}_0^i \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0^i)$, $i = 1, \ldots, \mu$ Initialisation
- Until(happy)
  - $\boldsymbol{y}^i = \text{Move}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu)$, $i = 1, \ldots, \lambda$ stochastic variations on Ω
  - Compute $\mathcal{F}(\boldsymbol{y}^i)$, $i = 1, \ldots, \lambda$
  - $(\boldsymbol{x}_{t+1}^1, \ldots, \boldsymbol{x}_{t+1}^\mu) = \text{Select}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ $(\mu + \lambda)$ selection
    $= \text{Select}(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ (μ, λ) selection

Distribution-based Metaheuristics

- Initialize distribution parameters $\boldsymbol{\theta_0}$
- Until(happy)
  - Sample distribution $P(\boldsymbol{\theta_t}) \rightarrow \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda \in \Omega$
  - Evaluate $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda$ on $\mathcal{F}$
  - Update parameters $\boldsymbol{\theta_{t+1}} \leftarrow F_\theta(\boldsymbol{\theta_t}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda, \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\lambda))$

Stochastic Optimization Templates

Distribution-based Metaheuristics

- Initialize distribution parameters $\boldsymbol{\theta_0}$
- Until(happy)
  - Sample distribution $P\left(\boldsymbol{\theta_t}\right) \rightarrow \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda \in \Omega$
  - Evaluate $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda$ on $\mathcal{F}$
  - Update parameters $\boldsymbol{\theta_{t+1}} \leftarrow F_\theta(\boldsymbol{\theta_t}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda, \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\lambda))$

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

$\Omega = \mathbb{R}^n$ $\qquad$ $P(\boldsymbol{\theta})$: normal distributions

Initialize distribution parameters $\boldsymbol{m}, \sigma, \mathbf{C}$, set population size $\lambda \in \mathbb{N}$

While not terminate

1. Sample distribution $\mathcal{N}(\boldsymbol{m}, \sigma^2\mathbf{C}) \to \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda \in \mathbb{R}^n$

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

2. Evaluate $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda$ on $\mathcal{F}$

$$\text{Compute } \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\lambda)$$

3. Select μ best samples

$$\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\mu$$

4. Update distribution parameters

$$\boldsymbol{m}, \sigma, \mathbf{C} \leftarrow F(\boldsymbol{m}, \sigma, \mathbf{C}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\mu, \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\mu))$$

The $(\mu, \lambda)-$Evolution Strategy (2)

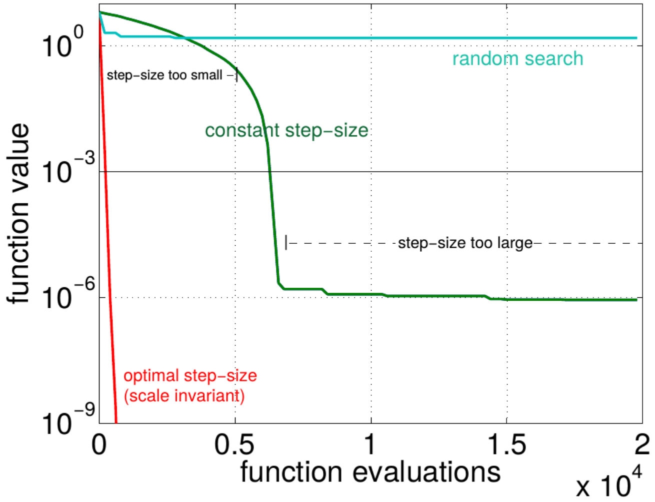
Gaussian Mutations

- mean vector $\boldsymbol{m} \in \mathbb{R}^n$ is the current 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 $\mathbf{C}$?

- $m = \frac{1}{\mu} \sum_{i=1}^{i=\mu} x_{i:\lambda}$ "Crossover" of best μ individuals
- Adaptive σ and $\mathbf{C}$

Need for adaptation



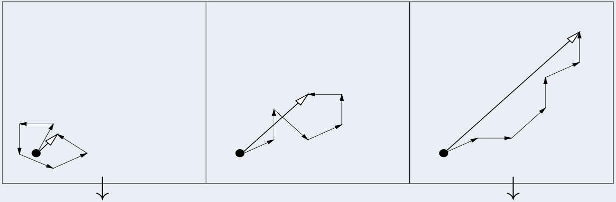
Sphere function $f(x) = \sum x_i^2, n = 10$

Content

Cumulative Step-Size Adaptation (CSA)

- Measure the length of the *evolution path*

path of the mean vector m in the generation sequence

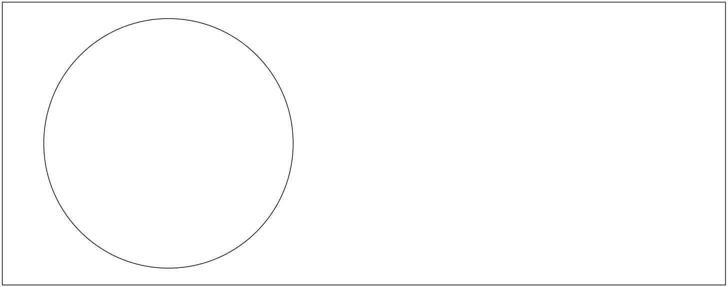


- Compare to random walk without selection

Covariance Matrix Adaptation

Rank-One Update

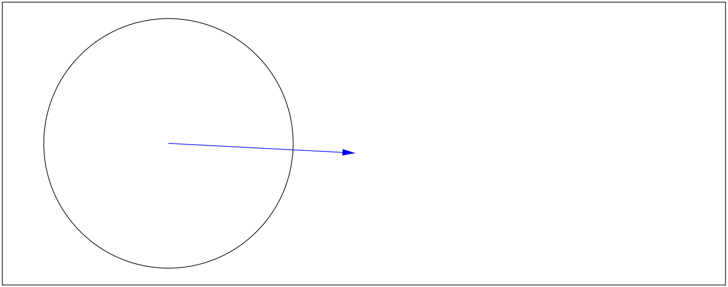
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



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

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



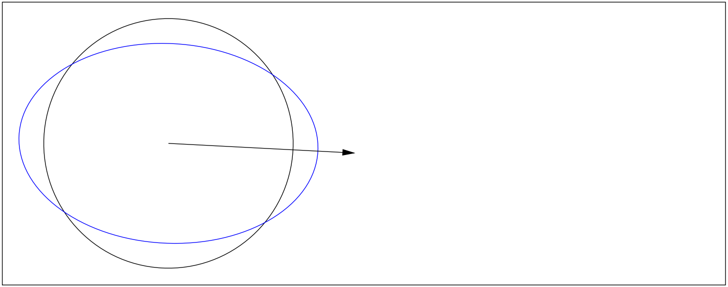
$\boldsymbol{y}_w$, movement of the population mean $\boldsymbol{m}$ (disregarding σ)

- 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

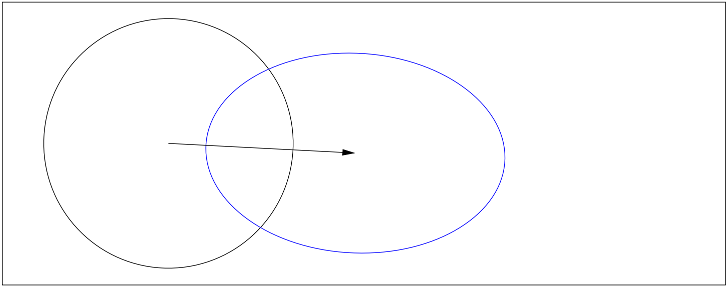
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



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

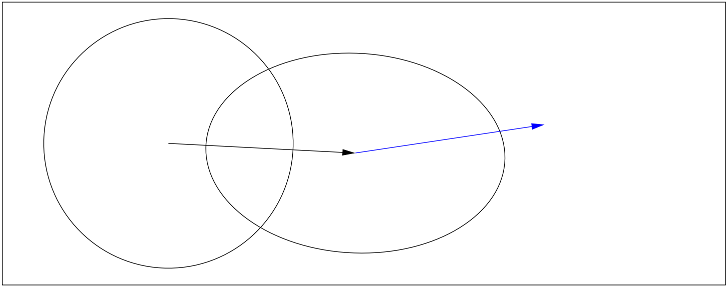
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



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

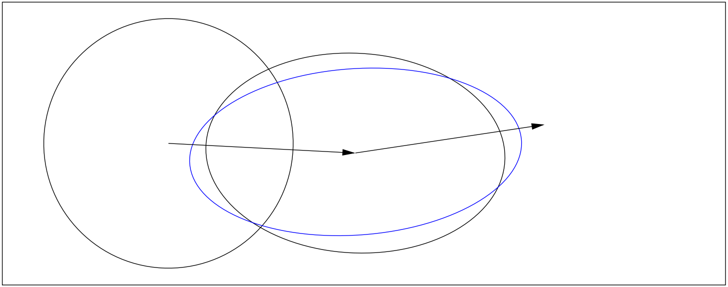
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i \, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



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

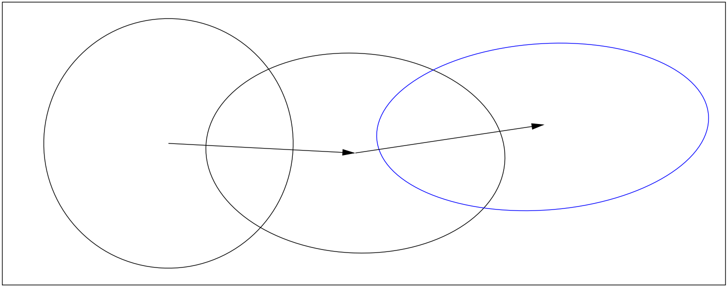
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w, \quad \boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$



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

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



- 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

CMA-ES in one slide

Input: $m \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, λ
Initialize: $\mathbf{C} = \mathbf{I}$, and $p_\mathbf{c} = \mathbf{0}$, $p_\sigma = \mathbf{0}$,
Set: $c_\mathbf{c} \approx 4/n$, $c_\sigma \approx 4/n$, $c_1 \approx 2/n^2$, $c_\mu \approx \mu_w/n^2$, $c_1 + c_\mu \leq 1$,
$d_\sigma \approx 1 + \sqrt{\frac{\mu_w}{n}}$, and $w_{i=1\ldots\lambda}$ such that $\mu_w = \frac{1}{\sum_{i=1}^{\mu} {w_i}^2} \approx 0.3\,\lambda$

While not terminate

$\boldsymbol{x}_i = \boldsymbol{m} + \sigma\, \boldsymbol{y}_i, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})\,,$ for $i = 1, \ldots, \lambda$	sampling
$\boldsymbol{m} \leftarrow \sum_{i=1}^{\mu} w_i\, \boldsymbol{x}_{i:\lambda} = \boldsymbol{m} + \sigma \boldsymbol{y}_w$ where $\boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}$	update mean
$\boldsymbol{p}_\mathbf{c} \leftarrow (1 - c_\mathbf{c})\, \boldsymbol{p}_\mathbf{c} + \mathbb{1}_{\{\|\boldsymbol{p}_\sigma\| < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_\mathbf{c})^2} \sqrt{\mu_w}\, \boldsymbol{y}_w$	cumulation for $\mathbf{C}$
$\boldsymbol{p}_\sigma \leftarrow (1 - c_\sigma)\, \boldsymbol{p}_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w}\, \mathbf{C}^{-\frac{1}{2}}\, \boldsymbol{y}_w$	cumulation for σ
$\mathbf{C} \leftarrow (1 - c_1 - c_\mu)\, \mathbf{C} + c_1\, \boldsymbol{p}_\mathbf{c} {\boldsymbol{p}_\mathbf{c}}^{\mathrm{T}} + c_\mu \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda} \boldsymbol{y}_{i:\lambda}^{\mathrm{T}}$	update $\mathbf{C}$
$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right)$	update of σ

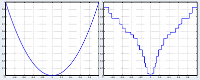
Not covered on this slide: termination, restarts, active or mirrored sampling, outputs, and boundaries

Invariances: Guarantee for Generalization

Invariance properties of CMA-ES

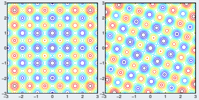
- Invariance to order preserving transformations in function space

  like all comparison-based algorithms



- Translation and rotation invariance

  to *rigid transformations* of the search space



→ Natural Gradient in distribution space

NES, Schmidhuber et al., 08

IGO, Olliver et al., 11

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 see IPOP-CMA-ES Auger & Hansen, 05

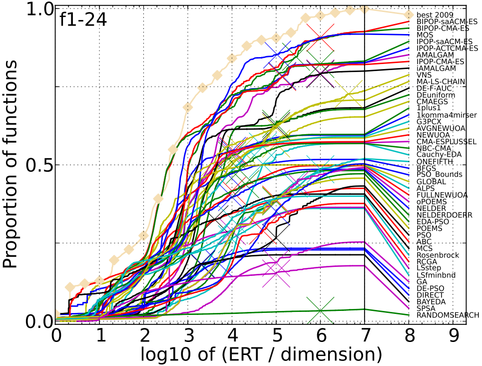
and BIPOP-CMA-ES Hansen, 09

BBOB – Black-Box Optimization Benchmarking

- ACM-GECCO workshops, 2009-2015 http://coco.gforge.inria.fr/
- 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

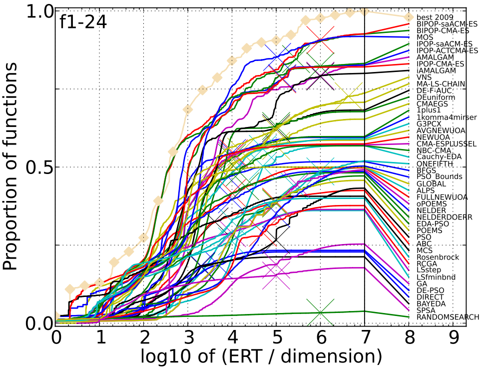


BBOB – Black-Box Optimization Benchmarking

- ACM-GECCO workshops, 2009-2015 http://coco.gforge.inria.fr/
- 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



(should be) available in ROOT6

Benazera, Hansen, Kégl, 15

Content

Goal

- Find the influencers in a given social network
- from public data sources — tweets, blogs, articles, ...

- Commercial goal: so as to bribe them :-)
- Scientific goal: with little user input

What is an Influencer?

No clear definition

- # followers?
- Highly retweeted? but # retweets disagrees with # followers
- Sources of information cascades?
- # invitations to join?
- Topic-specific PageRanking?
- Presence on Wikipedia?

Need for

- Topic-specific features
- **and** graph features
- **and** user input no generic ground truth

Seeded Influencer Ranking

Input

- A social network
- (Big) Data of user interactions — Tweets, blogs, messages, . . .
- Some identified influencers

The global picture

- Derive features representing the users — using content and traces
- Optimize scoring function — giving highest scores to known influencers
- return best-k scoring users — the most (known + new) influencers

Search Space and Learning Criterion

The scoring function

- Each individual is described by d features $x_i \in \mathbb{R}^d$
- Non-linear score h defined by pair (v, a) in $\mathbb{R}^d \times \mathbb{R}^d$

$$h_{v,a}(x) = \sum_{i=1}^{d} v_i |x_i - a_i|$$

A rank-based criterion

- Each score $h_{v,a}$ induces a ranking $R_{v,a}$ on the dataset
- Goal: Known influencers (KIs) ranked highest Best has rank n

Non-convex optimization

$$ArgMax_{(v,a)} \ \{ \sum_{x_i \in \text{KI}} R_{v,a}(x_i) \}$$

$\longrightarrow$ Stochastic Optimizer e.g., sep-CMA-ES

Data

Domains

- **Fashion**: 18/23 influencers with twitter account
- **SmartPhones**: 69/206 influencers with twitter account
- **Social Influence**: 39/39 influencers with twitter account
- **Wine**: 28/235 influencers with twitter account, coming from public sources (e.g., *Time Magazine*)
- **Human Resources**: 99 influencers from the *100 most influencial people in HR and recruiting on twitter*

Sources

- random 10% of all retweets of November 2014
- 100M tweets, 45M unique tweets,
- with origin-destination pairs $\rightarrow$ 43M nodes graph
- only words in at least 0.001% and at most 10% tweets considered $\rightarrow$ 10.5M candidates

Features

100 Content-based Features

foreach medium eventually

- Foreach user x, identify $\mathcal{W}(x)$, the N words with max tf-idf

  term frequency - inverse document frequency

- 50 words that appear most often in all $\mathcal{W}(influencer)$
- 50 words with max sum of tf-idf over $\bigcup \mathcal{W}(influencer)$
- each selected word is a feature for x: 0 if not present in $\mathcal{W}(x)$, ti-idf otherwise

  Many are null for most candidates

6 Network Features

- from the weighted graph of retweets

  centrality, PageRank, ...

5 Social Features

- from tweeter user profiles

  # tweets, # followers, ...

Seeded Influencer Identification: Discussion

Results

- Better than using only social features
- Found unexpected influencers
- Confirmed by experts

Forthcoming application

- Unveil some influencial xxx-sceptics Global warming, genocides, ...

Content

Content

Stochastic Optimization Templates (reminder)

Population-based Metaheuristics

- Randomly draw $\boldsymbol{x}_0^i \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0^i)$, $i = 1, \ldots, \mu$ Initialisation
- Until(happy)
  - $\boldsymbol{y}^i = \text{Move}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu)$, $i = 1, \ldots, \lambda$ stochastic variations on Ω
  - Compute $\mathcal{F}(\boldsymbol{y}^i)$, $i = 1, \ldots, \lambda$
  - $(\boldsymbol{x}_{t+1}^1, \ldots, \boldsymbol{x}_{t+1}^\mu) = \text{Select}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ $(\mu + \lambda)$ selection
    $= \text{Select}(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ (μ, λ) selection

Distribution-based Metaheuristics

- Initialize distribution parameters $\boldsymbol{\theta_0}$
- Until(happy)
  - Sample distribution $P(\boldsymbol{\theta_t}) \rightarrow \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda \in \Omega$
  - Evaluate $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda$ on $\mathcal{F}$
  - Update parameters $\boldsymbol{\theta_{t+1}} \leftarrow F_\theta(\boldsymbol{\theta_t}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_\lambda, \mathcal{F}(\boldsymbol{y}_1), \ldots, \mathcal{F}(\boldsymbol{y}_\lambda))$

Population-based Metaheuristics

- Randomly draw $\boldsymbol{x}_0^i \in \Omega$ and compute $\mathcal{F}(\boldsymbol{x}_0^i)$, $i = 1, \ldots, \mu$ Initialisation
- Until(happy)
  - $\boldsymbol{y}^i = \text{Move}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu)$, $i = 1, \ldots, \lambda$ stochastic variations on Ω
  - Compute $\mathcal{F}(\boldsymbol{y}^i)$, $i = 1, \ldots, \lambda$
  - $(\boldsymbol{x}_{t+1}^1, \ldots, \boldsymbol{x}_{t+1}^\mu) = \text{Select}(\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^\mu, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ $(\mu + \lambda)$ selection
    $= \text{Select}(\boldsymbol{y}^1, \ldots, \boldsymbol{y}^\lambda)$ (μ, λ) selection

Crossover or not Crossover

Issues with crossover

- Crossover ubiquitous in nature

  though not in all bacteria

- Showed to mainly work on dynamical problems

  Compared to ILS, SA, Ch. Papadimitriou et al., 15

Crossover or not Crossover

Issues with crossover

- Crossover ubiquitous in nature

  though not in all bacteria

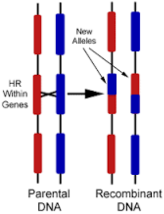
- Showed to mainly work on dynamical problems

  Compared to ILS, SA, Ch. Papadimitriou et al., 15

Yes, but

- Most artificial crossover operators exchange chunks of solutions
- Nature does not exchange "organs"
- but chunks of programs that builds the solution

$\neq$



Crossover or not Crossover

Issues with crossover

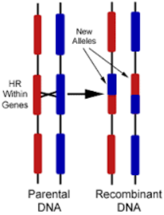
- Crossover ubiquitous in nature

  though not in all bacteria

- Showed to mainly work on dynamical problems

  Compared to ILS, SA, Ch. Papadimitriou et al., 15

$\neq$



Yes, but

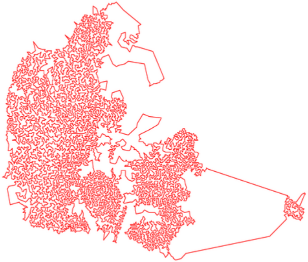
- Most artificial crossover operators exchange chunks of solutions
- Nature does not exchange "organs"
- but chunks of programs that builds the solution

Better no crossover than a poor crossover

Content

The Traveling Salesman Problem

Find shortest Hamiltonian circuit on n cities



dk11455.tsp

Permutation Representation

- Ordered list of cities:
  no semantic link with the objective
- Blind exchange of chunks meaningless, and leads to unfeasible circuits

Edge Representation

- List of all edges of the tour: unordered
  clearly related to objective
- but blind exchange of edges doesn't work either

Challenge or opportunity?

- A strong exact solver Concorde
- A very strong heuristic solver LKH-2

Lin-Kernighan-Helsgaun heuristic

Helsgaun, 06

LKH-2

- Local Search
- Deep k-opt moves — efficient implementation
- Many ad hoc heuristics

Multi-trial LKH-2

- Iterated Local Search
- Using . . . a crossover operator between best-so-far and new local optima
- Iterative Partial Transcription, a particular case of GPX — Mobius et al., 99-08

next slide

Most best known results on non-exactly solved instances, up to 10M cities

Respectful and Transmitting crossover

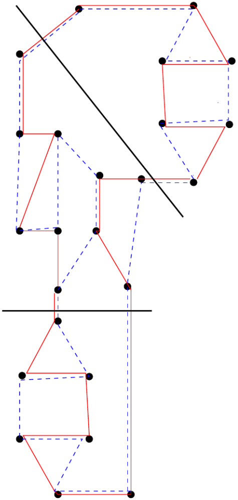
- Find k-partitions of cost 2 if exist $O(n)$
- Chooses best of $2^k - 2$ solutions $O(k)$
  - All common edges are kept
  - No new edge is created

Two possible hybridizations

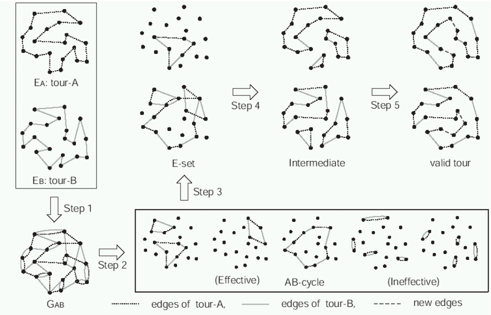
- Evolutionary Algorithm using GPX and iterated 2-opt as mutation operator

  Some best results on small instance $< 10k$ cities
- Replacing IPT in LKH-2 heuristic

  Improved some state-of-the-art results



Edge Assembly Crossover



The EAX crossover

1. Merge A and B
2. Alternate A/B edges
3. Local, then global choice strategy
4. Use sub-tours to remove parent edges
5. Local search for minimal new edges

The EAX algorithm

- Foreach randomly chosen parent pair — typically 300 iterations
- apply EAX N times — typically 30
- keep best of parents $+ N$ offspring — diversity-preserving selection

Best known results on several instances $\ldots \leq 200k$ cities

Evolutionary TSP: Discussion

Meta or not Meta?

- GPX can be applied to other graph problems
- EAX is definitely a specialized TSP operator

Take Home Message

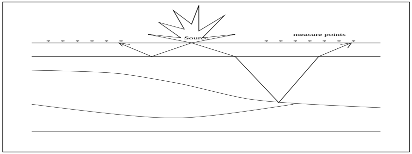
- Efficient crossovers might exist
- for semantically relevant representations
- but require domain knowledge . . . and fine-tuning

Content

Content

Identification of Spatial Geological Models

MS, Ehinger, Braunschweig, 98



The direct problem

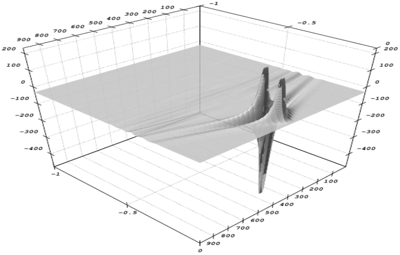
- Function ϕ
- applied to model $\mathcal{M}$
- Find result $\mathcal{R} = \phi(\mathcal{M})$

  $\rightarrow$ numerical simulation

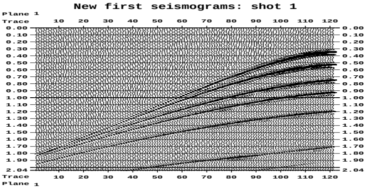
The inverse problem

- Given experimental results $\mathcal{R}^*$
- find model $\mathcal{M}^*$
- such that $\phi(\mathcal{M}^*) = \mathcal{R}^*$

$\rightarrow$ Arg $\text{Min}_{\mathcal{M}}\{||\phi(\mathcal{M}) - \mathcal{R}^*|^2\}$



(simulated) 3D seismogram



$120 \times 2D$ seismogram

Voronoi Representation

The solution space

- Velocities in some 3D domain
- Blocky model — piecewise constant

Values on a grid/mesh

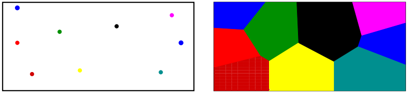
- Thousands of variables
- Uncorrelated

Voronoi representation

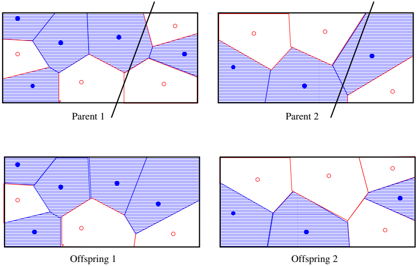
- List of labelled Voronoi sites
- Velocity = site label inside each cell

Variation Operators

- Geometric exchange — crossover
- Add, remove, move sites — mutations



Variable unordered list of Voronoi sites, and corresponding 'colored' Voronoi diagram



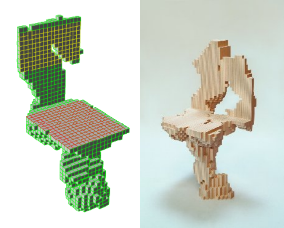
The crossover operator

Other Voronoi successes

Evolutionary chairs

- Minimize weight
- Constraint on stiffness

Permanent collection of
Beaubourg Modern Art Museum



The Seroussi architectural competition

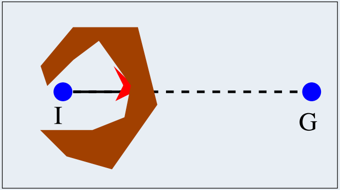
- A building for private exhibitions
- Optimize natural lightning of paintings all year round
- Room seeds for adaptive room desing

1st price (out of 8 submissions)

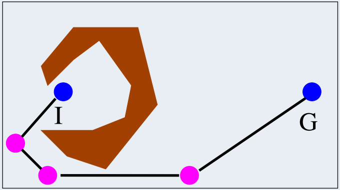
Content

The TGV paradigm

Improve a dummy algorithm



Cannot directly go from I to G



Can go sequentially from I to the intermediate stations, and to G

→ evolve the sequence of intermediate 'stations'

Representation

- Variable-length ordered lists of (●$_1$,●$_2$,●$_3$)

Variation operators

- Chromosome-like swap — variable-length crossover
- Move, add or remove intermediate stations — mutations

Evolutionary AI Planning

MS, P. Savéant, V. Vidal, 06

AI Planning problem $(\mathcal{S}, \mathcal{A}, I, G)$

- Given a state space $\mathcal{S}$ and a set of actions $\mathcal{A}$ — conditions, effects
- An initial state I and a goal state G — possibly partially defined
- Find optimal sequence of actions from I to G.

Many complete / satisficing planners — biennal IPC since 1998

Evolving intermediate states

- Search space: Variable-length ordered lists of states $(S_1, \ldots, S_l)$
- Solve $(\mathcal{S}, \mathcal{A}, S_i, S_{i+1})$ using a given planner — "dummy"
- All sub-problems solved $\rightarrow$ solution of initial problem

Highlights

J. Bibaï et al., 09-11

- Solves instances where 'dummy' planner fails
- Silver Medal, Humies Awards, GECCO 2010
- Winner, satisficing track, IPC 2011
- First multi-objective planner, IJCAI 2013 — M. Khouadjia et al.

Non-Parametric Representations: Discussion

Further examples

- Spatial geological models: horizontal layers + geological parameters

  patented
- Neural Networks HyerNeat, Stanley 07; Compressed NNs, Schmidhuber 10

Non-parametric representations

- Explore larger search spaces, but with some constraints
  $\rightarrow$ explore a sub-variety of the whole space
- Modifies both the approximation bias

  how far from the true optimum is the best solution under the streetlight?
- and the optimization error our algorithm is not perfect

Optimization as an exploratory tool

Content

The Black-Box Hypothesis

Meta or not Meta?

- A continuum of embedded classes of problems
  Where does Black Box start? e.g., GPX and EAX
- Continuous optimization: at least the dimension is known
- The more information the better e.g. multimodality for restarts
- Learn about $\mathcal{F}$ during the search
  $\rightarrow$ online tuning beyond distribution parameters

Not Covered Today

General

- Statistical tests — Tutorials available
- From parameter setting to hyper-heuristics — Programming by Optimization, Hoos, 2014; Burke et al., 08-15
- Multi-objective — Full track with own conferences

Continuous

- Surrogates — ML helping Optimization
- Constraints — from Lagrangian to specific approaches
- Noise handling — e.g., using Höffding or Bernstein inequalities

Other Representations/Paradigms

- Genetic Programming — Search a space of programs
- Generative Developmental Systems — ......that build the solution

Flexibility