Introduction to Black-Box Optimization in Continuous Search Spaces

Definitions, Examples, Difficulties
I am happy to answer questions at any time!
Problem Statement

Continuous Domain Search/Optimization

Task: minimize an objective function (*fitness* function, *loss* function) in continuous domain

\[ f : \mathcal{X} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}, \quad x \mapsto f(x) \]

- **Black Box** scenario (direct search scenario)
  - gradients are not available or not useful
  - problem domain specific knowledge is used only within the black box, e.g. within an appropriate encoding

- **Search costs:** number of function evaluations
Typical Applications

- model/system calibration
  - biological/chemical/physical $\Rightarrow$ universal constants
  - production process
- optimization of control parameters
  - movements of a robot (e.g. for the RoboCup)
  - trajectory of a rocket
  - stability of a gas flame
- shape optimization
  - curve fitting
  - aero- or fluid dynamics design (airfoil, airship)
Optimization of walking gaits

CMA-ES, Covariance Matrix Adaptation Evolution Strategy [Hansen et al 2003]
IDEA, Iterated Density Estimation Evolutionary Algorithm [Bosman 2003]
Fminsearch, downhill simplex method [Nelder & Mead 1965]
CMA-ES, Covariance Matrix Adaptation Evolution Strategy [Hansen et al 2003]
IDEA, Iterated Density Estimation Evolutionary Algorithm [Bosman 2003]
Fminsearch, downhill simplex method [Nelder & Mead 1965]
We present a control system based on 3D muscle actuation

Flexible Muscle-Based Locomotion for Bipedal Creatures
http://vimeo.com/79098420
Problem Statement

Continuous Domain Search/Optimization

- **Goal**
  - fast convergence to the global optimum
  - solution $x$ with small function value $f(x)$ with least search cost
  
  ... or to a robust solution $x$
  
  there are two conflicting objectives

- **Typical Examples**
  - shape optimization (e.g. using CFD)
  - model calibration
  - parameter calibration
  
  curve fitting, airfoils
  
  biological, physical
  
  controller, plants, images

- **Problems**
  - exhaustive search is infeasible
  - naive random search takes too long
  - deterministic search is not successful / takes too long
Problem Statement
Continuous Domain Search/Optimization

- **Goal**
  - fast convergence to the global optimum
  - solution $x$ with small function value $f(x)$ with least search cost
  - ... or to a robust solution $x$

- **Typical Examples**
  - shape optimization (e.g. using CFD)
  - model calibration
  - parameter calibration
  - curve fitting, airfoils
  - biological, physical
  - controller, plants, images

- **Problems**
  - exhaustive search is infeasible
  - naive random search takes too long
  - deterministic search is not successful / takes too long
Objective Function Properties

The objective function $f : \mathcal{X} \subset \mathbb{R}^n \rightarrow \mathbb{R}$ has typically moderate dimensionality, say $n \ll 10$, and can be

- non-linear
- non-separable
- non-convex
- multimodal
- non-smooth
- discontinuous, plateaus
- ill-conditioned
- noisy
- ...

there are possibly many local optima
derivatives do not exist

Goal: cope with any of these function properties
they are related to real-world problems
Objective Function Properties

The objective function \( f : \mathcal{X} \subset \mathbb{R}^n \rightarrow \mathbb{R} \) has typically moderate dimensionality, say \( n \ll 10 \), and can be

- non-linear
- non-separable
- non-convex
- multimodal
- non-smooth
- discontinuous, plateaus
- ill-conditioned
- noisy
- ... 

Goal: cope with any of these function properties

they are related to real-world problems
What Makes a Function Difficult to Solve?

Why stochastic search?

- **non-linear, non-quadratic, non-convex**
  - on linear and quadratic functions much better search policies are available

- **ruggedness**
  - non-smooth, discontinuous, multimodal, and/or noisy function

- **dimensionality (size of search space)**
  - (considerably) larger than three

- **non-separability**
  - dependencies between the objective variables

- **ill-conditioning**
Ruggedness
non-smooth, discontinuous, multimodal, and/or noisy

cut from a 5-D example, (easily) solvable with evolution strategies
Ruggedness
non-smooth, discontinuous, multimodal, and/or noisy

multi-funnel example
Curse of Dimensionality

The term *Curse of dimensionality* (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 20 points equally spaced onto the interval \([0, 1]\). Now consider the 10-dimensional space \([0, 1]^{10}\). To get similar coverage in terms of distance between adjacent points requires \(20^{10} \approx 10^{13}\) points. 20 points appear now as isolated points in a vast empty space.

Remark: distance measures break down in higher dimensionalities (the central limit theorem kicks in)

Consequence: a search policy that is valuable in small dimensions might be useless in moderate or large dimensional search spaces. Example: exhaustive search.
Curse of Dimensionality

The term *Curse of dimensionality* (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 20 points equally spaced onto the interval \([0, 1]\). Now consider the 10-dimensional space \([0, 1]^{10}\). To get similar coverage in terms of distance between adjacent points requires \(20^{10} \approx 10^{13}\) points. 20 points appear now as isolated points in a vast empty space.

Remark: distance measures break down in higher dimensionalities (the central limit theorem kicks in)

Consequence: a search policy that is valuable in small dimensions might be useless in moderate or large dimensional search spaces. Example: exhaustive search.
Curse of Dimensionality

The term *Curse of dimensionality* (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 20 points equally spaced onto the interval \([0, 1]\). Now consider the 10-dimensional space \([0, 1]^{10}\). To get similar coverage in terms of distance between adjacent points requires \(20^{10} \approx 10^{13}\) points. 20 points appear now as isolated points in a vast empty space.

Remark: *distance measures* break down in higher dimensionalities (the central limit theorem kicks in)

Consequence: a *search policy* that is valuable in small dimensions might be useless in moderate or large dimensional search spaces. Example: exhaustive search.
Curse of Dimensionality

The term *Curse of dimensionality* (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 20 points equally spaced onto the interval $[0, 1]$. Now consider the 10-dimensional space $[0, 1]^{10}$. To get similar coverage in terms of distance between adjacent points requires $20^{10} \approx 10^{13}$ points. 20 points appear now as isolated points in a vast empty space.

Remark: distance measures break down in higher dimensionalities (the central limit theorem kicks in)

Consequence: a search policy that is valuable in small dimensions might be useless in moderate or large dimensional search spaces. Example: exhaustive search.
Separable Problems

Definition (Separable Problem)

A function $f$ is separable if

$$\arg\min_{(x_1, \ldots, x_n)} f(x_1, \ldots, x_n) = \left( \arg\min_{x_1} f(x_1, \ldots), \ldots, \arg\min_{x_n} f(\ldots, x_n) \right)$$

$\Rightarrow$ it follows that $f$ can be optimized in a sequence of $n$ independent 1-D optimization processes

Example: Additively decomposable functions

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} f_i(x_i)$$

eample: Rastrigin function, where $f_i = f_j \forall i, j$
Non-Separable Problems

Building a non-separable problem from a separable one \(^{(1,2)}\)

Rotating the coordinate system

- \( f : \mathbf{x} \mapsto f(\mathbf{x}) \) separable
- \( f : \mathbf{x} \mapsto f(\mathbf{R}\mathbf{x}) \) non-separable

\( \mathbf{R} \) rotation matrix

---

Ill-Conditioned Problems

Curvature of level sets

Consider the convex-quadratic function
\[ f(x) = \frac{1}{2} (x - x^*)^T H (x - x^*) = \frac{1}{2} \sum_i h_{i,i} (x_i - x_i^*)^2 + \frac{1}{2} \sum_{i \neq j} h_{i,j} (x_i - x_i^*) (x_j - x_j^*) \]

\( H \) is Hessian matrix of \( f \) and symmetric positive definite

gradient direction \(-f'(x)^T\)

Newton direction \(-H^{-1}f'(x)^T\)

Ill-conditioning means squeezed level sets (high curvature).
Condition number equals nine here. Condition numbers up to \(10^{10}\) are not unusual in real world problems.

If \( H \approx I \) (small condition number of \( H \)) first order information (e.g. the gradient) is sufficient. Otherwise second order information (estimation of \( H^{-1} \)) is necessary.
Landscape of Continuous Search Methods

*Gradient-based (Taylor, local)*
- Conjugate gradient methods [Fletcher & Reeves 1964]
- Quasi-Newton methods (BFGS) [Broyden et al 1970]

*Derivative-free optimization (DFO)*
- Trust-region methods (NEWUOA, BOBYQA) [Powell 2006, 2009]
- Simplex downhill [Nelder & Mead 1965]
- Pattern search [Hooke & Jeeves 1961, Audet & Dennis 2006]

*Stochastic (randomized) search methods*
- Evolutionary algorithms (broader sense, continuous domain)
  - Differential Evolution [Storn & Price 1997]
  - Particle Swarm Optimization [Kennedy & Eberhart 1995]
  - Evolution Strategies [Rechenberg 1965, Hansen & Ostermeier 2001]
- Simulated annealing [Kirkpatrick et al 1983]
- Simultaneous perturbation stochastic approximation (SPSA) [Spall 2000]
What Makes a Function Difficult to Solve?  
... and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dimensionality</strong></td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td></td>
<td><strong>separability, locality/neighborhood, encoding</strong></td>
</tr>
<tr>
<td><strong>Ill-conditioning</strong></td>
<td>second order approach</td>
</tr>
<tr>
<td></td>
<td>changes the neighborhood metric</td>
</tr>
<tr>
<td><strong>Ruggedness</strong></td>
<td><strong>non-local</strong> policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td><strong>population-based</strong> method, stochastic, non-elitistic</td>
</tr>
<tr>
<td></td>
<td>recombination operator</td>
</tr>
<tr>
<td></td>
<td>serves as repair mechanism</td>
</tr>
<tr>
<td></td>
<td>restarts</td>
</tr>
<tr>
<td></td>
<td>... metaphors</td>
</tr>
</tbody>
</table>
### What Makes a Function Difficult to Solve?

...and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensionality</td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td></td>
<td>separability, locality/neighborhood, encoding</td>
</tr>
<tr>
<td>Ill-conditioning</td>
<td>second order approach</td>
</tr>
<tr>
<td></td>
<td>changes the neighborhood metric</td>
</tr>
<tr>
<td>Ruggedness</td>
<td>non-local policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>population-based method, stochastic, non-elitistic</td>
</tr>
<tr>
<td></td>
<td>recombination operator</td>
</tr>
<tr>
<td></td>
<td>serves as repair mechanism</td>
</tr>
<tr>
<td></td>
<td>restarts</td>
</tr>
</tbody>
</table>

...metaphors
## Problem Statement

### Ill-Conditioned Problems

**What Makes a Function Difficult to Solve?**

...and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dimensionality</strong></td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td></td>
<td>separability, locality/neighborhood, encoding</td>
</tr>
<tr>
<td><strong>Ill-conditioning</strong></td>
<td>second order approach</td>
</tr>
<tr>
<td></td>
<td>changes the neighborhood metric</td>
</tr>
<tr>
<td><strong>Ruggedness</strong></td>
<td>non-local policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>population-based method, stochastic, non-elitistic</td>
</tr>
<tr>
<td></td>
<td>recombination operator</td>
</tr>
<tr>
<td></td>
<td>serves as repair mechanism</td>
</tr>
<tr>
<td></td>
<td>restarts</td>
</tr>
</tbody>
</table>

...metaphors

Anne Auger & Nikolaus Hansen

CMA-ES

July, 2014 12 / 81
Questions?