Lecture 29/30: Stochastic Search Algorithms

BT 3051 – Data Structures and Algorithms for Biology

Karthik Raman
Department of Biotechnology
Indian Institute of Technology Madras

Direct Search Methods

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Why Direct Search?

Courtesy: Prof. Alonso, Stanford AA222

Many real-life applications involve major challenges:

- non-differentiable objective functions (also constraints?)
- non-convex search spaces
- discrete search spaces
- mixed variables (discrete, continuous)
- very high dimensionality
- many local minima

Methods for Optimisation

- Therefore, methods cannot compute/use gradient information
- Such methods broadly categorised as *Direct Search Methods*
- Central aspect of direct search methods
  - Strategy to vary parameter vector
  - Strategy to accept/reject a new parameter vector

Methods for Optimisation

- Therefore, methods cannot compute/use gradient information
- Such methods broadly categorised as *Direct Search Methods*
- Central aspect of direct search methods
  - Strategy to vary *parameter vector*
  - Strategy to accept/reject a *new parameter vector*

Methods for Optimisation

- Therefore, methods cannot compute/use gradient information
- Such methods broadly categorised as *Direct Search Methods*
- Central aspect of direct search methods
  - Strategy to vary *parameter vector*
  - Strategy to accept/reject a *new parameter vector*

Methods for Optimisation

- Therefore, methods cannot compute/use gradient information
- Such methods broadly categorised as *Direct Search Methods*
- Central aspect of direct search methods
  - Strategy to vary *parameter vector*
  - Strategy to accept/reject a *new parameter vector*

Methods for Optimisation

- Therefore, methods cannot compute/use gradient information
- Such methods broadly categorised as *Direct Search Methods*
- Central aspect of direct search methods
  - Strategy to vary *parameter vector*
  - Strategy to accept/reject a *new parameter vector*

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods of Optimisation

Acceptance/Rejection

- Once a variation is generated, a decision must then be made whether or not to accept the newly derived parameters
- Most standard direct search methods use the greedy criterion to make this decision
  - a new parameter vector is accepted iff it reduces the value of the cost function
- Greedy decision process converges fairly fast — runs the risk of becoming trapped in a local minimum
- Inherently parallel search techniques like GAs and ESs have some built-in safeguards to forestall misconvergence
- By running several vectors simultaneously, superior parameter configurations can help other vectors escape local minima

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods for Optimisation

Desiderata

- Ability to handle non-differentiable, non-linear and multi-modal cost functions
- Parallelisability to cope with computation intensive cost functions
- Ease of use, i.e. few control variables to steer the minimisation
- These variables should also be robust and easy to choose
- Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods for Optimisation

Desiderata

- Ability to handle non-differentiable, non-linear and multi-modal cost functions
- Parallelisability to cope with computation intensive cost functions
- Ease of use, i.e. few control variables to steer the minimisation
- These variables should also be robust and easy to choose
- Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods for Optimisation

Desiderata

- Ability to handle non-differentiable, non-linear and multi-modal cost functions
- Parallelisability to cope with computation intensive cost functions
- Ease of use, i.e. few control variables to steer the minimisation
- These variables should also be robust and easy to choose
- Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods for Optimisation

Desiderata

- Ability to handle non-differentiable, non-linear and multi-modal cost functions
- Parallelisability to cope with computation intensive cost functions
- Ease of use, i.e. few control variables to steer the minimisation
- These variables should also be robust and easy to choose
- Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Methods for Optimisation

Desiderata

- Ability to handle non-differentiable, non-linear and multi-modal cost functions
- Parallelisability to cope with computation intensive cost functions
- Ease of use, i.e. few control variables to steer the minimisation
- These variables should also be robust and easy to choose
- Good convergence properties, i.e. consistent convergence to the global minimum in consecutive independent trials

Also see **Storn R & Price K** (1997) *J. of Global Optimization* **11**:341–359

Direct Search Methods:
Classic Methods

Classic Methods of Direct Search

- Hooke–Jeeves Pattern Search
- Nelder–Mead Simplex / Downhill Simplex Method (`fminsearch`)
- Grid Search
- Random Search
- Hill-climbing

Classic Methods of Direct Search

- Hooke–Jeeves Pattern Search
- Nelder–Mead Simplex / Downhill Simplex Method (`fminsearch`)
- Grid Search
- Random Search
- Hill-climbing

Classic Methods of Direct Search

- Hooke–Jeeves Pattern Search
- Nelder–Mead Simplex / Downhill Simplex Method (`fminsearch`)
- Grid Search
- Random Search
- Hill-climbing

Classic Methods of Direct Search

- Hooke–Jeeves Pattern Search
- Nelder–Mead Simplex / Downhill Simplex Method (`fminsearch`)
- Grid Search
- Random Search
- Hill-climbing

Classic Methods of Direct Search

- Hooke–Jeeves Pattern Search
- Nelder–Mead Simplex / Downhill Simplex Method (`fminsearch`)
- Grid Search
- Random Search
- Hill-climbing

Overview

Direct/Stochastic Search Algorithms

- Simulated Annealing
- Evolutionary Algorithms
  - Genetic Algorithms
  - Evolutionary Strategies
- Swarm Algorithms

Direct/Stochastic Search Algorithms

- Simulated Annealing
- Evolutionary Algorithms
  - Genetic Algorithms
  - Evolutionary Strategies
- Swarm Algorithms

Direct/Stochastic Search Algorithms

- Simulated Annealing
- Evolutionary Algorithms
  - Genetic Algorithms
  - Evolutionary Strategies
- Swarm Algorithms

Direct/Stochastic Search Algorithms

- Simulated Annealing
- Evolutionary Algorithms
  - Genetic Algorithms
  - Evolutionary Strategies
- Swarm Algorithms

Direct/Stochastic Search Algorithms

- Simulated Annealing
- Evolutionary Algorithms
  - Genetic Algorithms
  - Evolutionary Strategies
- Swarm Algorithms

Simulated Annealing

Simulated Annealing

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- Borrowed idea from condensed matter physics – annealing of metals
- Perturb configuration of system
- Accept all moves that reduce cost
- Accept those that increase cost with low probability (*Metropolis Criterion*)

Simulated Annealing

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- Borrowed idea from condensed matter physics – annealing of metals
- Perturb configuration of system
- Accept all moves that reduce cost
- Accept those that increase cost with low probability (*Metropolis Criterion*)

Simulated Annealing

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- Borrowed idea from condensed matter physics – annealing of metals
- Perturb configuration of system
- Accept all moves that reduce cost
- Accept those that increase cost with low probability (*Metropolis Criterion*)

Simulated Annealing

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- Borrowed idea from condensed matter physics – annealing of metals
- Perturb configuration of system
- Accept all moves that reduce cost
- Accept those that increase cost with low probability (*Metropolis Criterion*)

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Simulated Annealing: Metropolis Criterion

Metropolis N et al. (1953) *The Journal of Chemical Physics* **21**:1087–1092

- In particular, the transition probability $P(e_i, e_j, T)$ from energy e_i to e_j at temperature T is given by

$$P(e_i, e_j, T) = e^{\frac{e_i - e_j}{k_B T}}$$

Key Annealing Parameters

- Initial temperature
- Annealing schedule
- Length of run
- Stopping conditions
- Often decided by trial-and-error

Evolutionary Algorithms

Evolutionary Algorithms

Definition

Tom Mitchell:

"Computational procedures patterned on biological evolution"

"Search procedure that probabilistically applies search operators to a set of points in the search space"

Evolutionary Algorithms

Definition

Tom Mitchell:

“Computational procedures patterned on biological evolution”

“Search procedure that probabilistically applies search operators to a set of points in the search space”

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

Genetic Algorithms

- Evolves populations of solutions!
- Binary string representation of solutions
- Key terms
  - Population
  - Chromosome
  - Mutation
  - Crossover
  - Fitness function
  - Selection

How do we choose/tune parameters such as mutation probability, crossover probability, population size etc.?

Evolution Strategies

- Popular for solving complex optimisation problems
- Individuals: Real numbers, rather than data structures
- Strategy parameters — genes that affect the evolutionary process for a particular individual (probability distribution/random process rate)
- All parameters evolve
- Self-adaptation — genotype adapts to alter the evolutionary process

Evolution Strategies

- Popular for solving complex optimisation problems
- Individuals: Real numbers, rather than data structures
- Strategy parameters — genes that affect the evolutionary process for a particular individual (probability distribution/random process rate)
- All parameters evolve
- Self-adaptation — genotype adapts to alter the evolutionary process

Evolution Strategies

- Popular for solving complex optimisation problems
- Individuals: Real numbers, rather than data structures
- Strategy parameters — genes that affect the evolutionary process for a particular individual (probability distribution/random process rate)
- All parameters evolve
- Self-adaptation — genotype adapts to alter the evolutionary process

Evolution Strategies

- Popular for solving complex optimisation problems
- Individuals: Real numbers, rather than data structures
- Strategy parameters — genes that affect the evolutionary process for a particular individual (probability distribution/random process rate)
- All parameters evolve
- Self-adaptation — genotype adapts to alter the evolutionary process

Evolution Strategies

- Popular for solving complex optimisation problems
- Individuals: Real numbers, rather than data structures
- Strategy parameters — genes that affect the evolutionary process for a particular individual (probability distribution/random process rate)
- All parameters evolve
- Self-adaptation — genotype adapts to alter the evolutionary process

Evolvable Hardware

- Evolving digital/analogue circuits
- Adders and other circuits have been evolved
- Fitness evaluations are often done by simulations, except for FPGAs
- Evolved circuits are often more robust!

Evolvable Hardware

- Evolving digital/analogue circuits
- Adders and other circuits have been evolved
- Fitness evaluations are often done by simulations, except for FPGAs
- Evolved circuits are often more robust!

Evolvable Hardware

- Evolving digital/analogue circuits
- Adders and other circuits have been evolved
- Fitness evaluations are often done by simulations, except for FPGAs
- Evolved circuits are often more robust!

Evolvable Hardware

- Evolving digital/analogue circuits
- Adders and other circuits have been evolved
- Fitness evaluations are often done by simulations, except for FPGAs
- Evolved circuits are often more robust!

Representation Paradigms

- Simple binary chromosomes
- Trees and complex data structures
- Cartesian Genetic Programming (for Evolvable Hardware)
- ...

Operators

- Macro-mutation: Large change in alleles without recombination
- Hybrid operators (not typical of evolution at all), e.g. Hill climbing
- Operators for permutations (e.g. Travelling Salesman)
- Learning — individuals alter their chromosomes before replication
- Evolving operators (e.g. by encoding probabilities into the chromosomal representations)
- ...

Selection

- Steady state (replace few)
  - replace worst
  - replace random
  - ...
- Tournament selection
- Fitness proportionate selection (Roulette Wheel)
- Elitism
- ...

Applications

- Complex multi-objective optimisation problems
- Exam/course timetables!
- Computational biology
  - Phylogenetic trees
  - Multiple sequence alignment
  - Protein folding
  - Identifying coding regions
  - Clustering microarray data
  - Parameter optimisation (kinetic models)
- Electrical circuit design
- . . .

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations

Conclusions

When to use Evolutionary Computation?

- Often quite useful
- Careful choice of representation, operators etc. is crucial
- Especially useful, when structure of search space is poorly understood
- *Might help understand the problem better*
- No reason to believe that a GA approach would be any better than any other optimisation technique!

Challenges

- Choice of representation
- $\sim$ Black box behaviour
- Computationally expensive, esp. fitness calculations