Stochastic methods for global optimization

Computational Methods in Optimization
CS 520, Purdue

David F. Gleich
Purdue University
Spring 2025

IMPORTANT

University class survey.
Please fill these out.
These are used to evaluate faculty members.

My class survey.
I use this to improve the class.
Do quizzes help you?
Does it help to discuss with peers for in-class questions?
Did you expect to cover additional material? (If so, what?)

Optimization and Sampling

Construct a probability distribution where the "most likely" point is a global max.

$$P(x) = 1/Z \exp(-f(x))\, Ind[c(x) = 0]\, Ind[x \geq \ell]$$

Optimization and Sampling

Construct a probability distribution where the "most likely" point is a global max.

$$P(x) = 1/Z \exp(-f(x))\, Ind[c(x) = 0]\, Ind[x \geq \ell]$$

Metropolis-Hastings Markov-Chain Monte Carlo

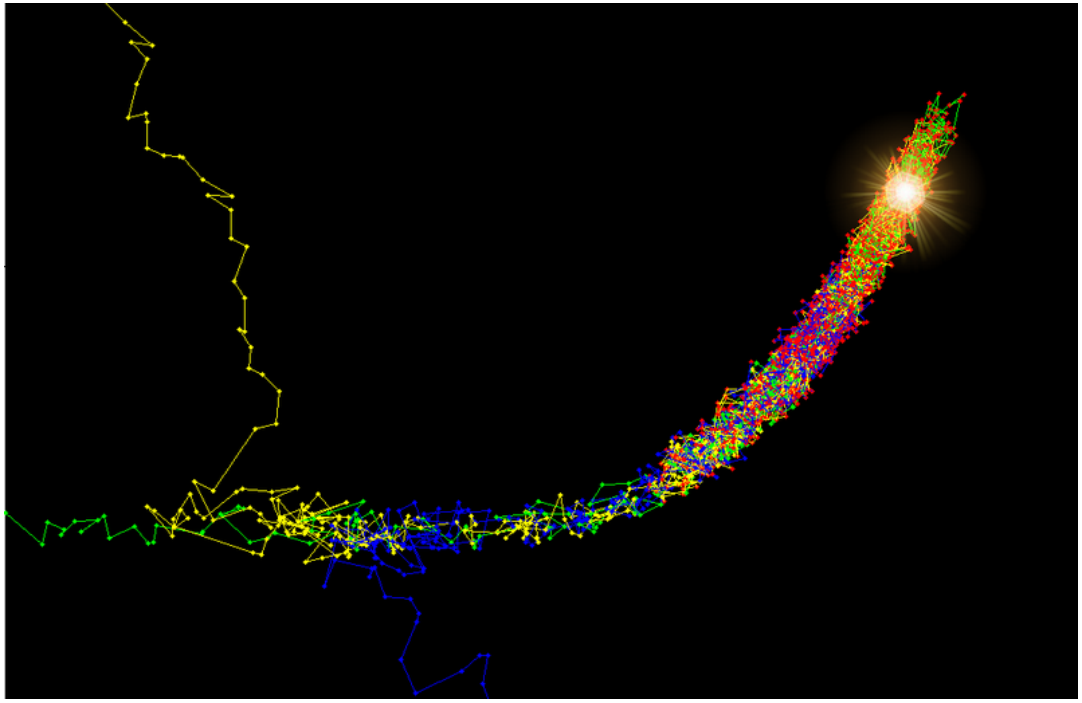
Given x_k, propose $x_{k+1} = x_k + N(0,\tau)$

Then compute $a = P(x_{k+1}) / P(x_k)$.
Accept if $a > 1$. Else accept with prob a.

Markov-Chain Monte Carlo

Sampling from the Rosenbrock function with a Markov-chain Monte Carlo sampler. The chain spends most of its time in regions where the objective function is large.

From wikipedia.



Show movie!

http://vimeo.com/22616409

Simulated Annealing (like M-H)

Let P(a,T) be the probability of accepting a move with a such that P(a,T) → Ind[a > 1] as T → 0. "T" is called the temperature. It's like a step-size parameter.

T going to zero called "cooling"

Improving these samplers

Abraham Flaxman has a great series of blog posts about improving these samplers.

http://healthyalgorithms.com/2011/01/28/mcmc-in-python-pymc-step-methods-and-their-pitfalls/

Goodman and Weare suggest multiple walks:

http://astrobites.com/2012/02/20/code-you-can-use-the-mcmc-hammer/

These improvements

All of these improvements seem to have the flavor of randomizing a deterministic derivative free method.

This can often be a very good idea.

Simulated annealing

Genetic algorithms

Ant colony optimization

“meta-heuristics”

The elephant in the room

What is the big issue with these methods?

MCMC for combinatorial problems

Instead of $x_{k+1} = x_k + N(0,\tau)$, consider a combinatorial structure:

- paths
- permutations
- graphs

And let x_{k+1} = random neighbor of x_k

Now… same procedure.

Discussion

These are all “randomized greedy” procedures in a combinatorial search space.

Difficult to trust the output for anything but improvements.

I’m not an expert on these things.

Avoid them if possible. Find a way to state your problem as an optimization problem and relax.

Look at Wikipedia for more details if you need to use them.

Computational Optimization

CS 520 – Purdue – David F. Gleich

INTEGER OPTIMIZATION – Branch and Bound

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \boldsymbol{A}\mathbf{x} = \mathbf{b} \\ & x_i \in \{0, 1\} \end{aligned}$$

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \boldsymbol{A}\mathbf{x} = \mathbf{b} \\ & x_i \in \{0, 1\} \end{aligned}$$

Relaxation

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \boldsymbol{A}\mathbf{x} = \mathbf{b} \\ & x_i \geq 0, x_i \leq 1 \end{aligned}$$

Root relaxation

$$\mathbf{c}^T\mathbf{x}^*_{\text{relax}} \leq \mathbf{c}^T\mathbf{x}^*_{\text{binary}}$$

We increase the feasible points, so the objective function can only get "better" (lower)

minimize $\mathbf{c}^T\mathbf{x}$

subject to $\boldsymbol{A}\mathbf{x} = \mathbf{b}$

$x_i \geq 0, x_i \leq 1$

Rounding

$$\mathbf{c}^T\mathbf{x}^*_{\text{relax}} \leq \mathbf{c}^T\mathbf{x}^*_{\text{binary}} \leq \mathbf{c}^T\mathbf{x}_{\text{round}}$$

BRANCH

For any fractional value x_i in the root relaxation, we split on that value to be an integer (in this case, either 0 or 1)

minimize $\mathbf{c}^T\mathbf{x}$

subject to $\boldsymbol{A}\mathbf{x} = \mathbf{b}$

$x_i \geq 0, x_i \leq 1$

$x_1 = 0$

minimize $\mathbf{c}^T\mathbf{x}$

subject to $\boldsymbol{A}\mathbf{x} = \mathbf{b}$

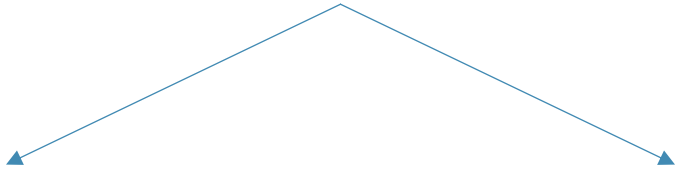
$x_i \geq 0, x_i \leq 1$

$x_1 = 1$

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \boldsymbol{A}\mathbf{x} = \mathbf{b} \\ & x_i \geq 0, x_i \leq 1 \end{aligned}$$

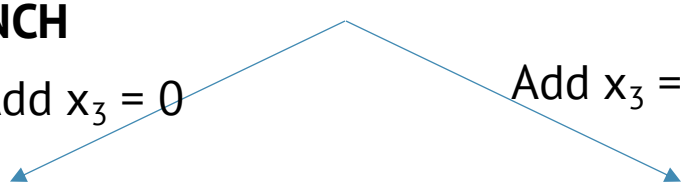
Rounding

$$\mathbf{c}^T\mathbf{x}^*_{\text{relax}} \leq \mathbf{c}^T\mathbf{x}^*_{\text{binary}} \leq \mathbf{c}^T\mathbf{x}_{\text{round}}$$

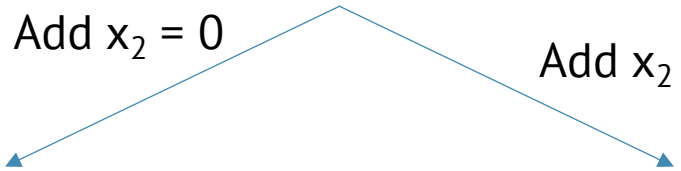


BRANCH
For any fractional value x_i in the root relaxation, we split on that value to be an integer (in this case, either 0 or 1)

BRANCH



BRANCH



For each branch we get a lower-bound & upper-bound by solving the LP and rounding.
At some point, we can stop branching because it'll be worse than some other branch.

One other idea.

Cutting planes are new linear inequality constraints that cut off many non-integer solutions in one step.