Black Box Policy Optimization

Let's not lose sight of the goal in RL: **find an optimal policy**

Previously:

- Indirect policy search by estimating value:
  - With models: VI, PI, LQR, MPC
  - Without models: Fitted-Q, TD, Monte Carlo, SARSA, Q-Learning

Black Box Policy Optimization

Let's not lose sight of the goal in RL: **find an optimal policy**

Previously:

- Indirect policy search by estimating value:
  - With models: VI, PI, LQR, MPC
  - Without models: Fitted-Q, TD, Monte Carlo, SARSA, Q-Learning

Today:

Don't be "blinded by the beauty of the Bellman Equation"
-Andrew Moore:

Black Box Policy Optimization

Idea: parameterize policy directly $\pi_\theta : x \mapsto a$

Black Box Policy Optimization

Idea: parameterize policy directly $\pi_\theta : x \mapsto a$

roll out trajectory under current policy: $\xi = (x_0, a_0, \ldots, x_{T-1}, a_{T-1})$

calculate reward $R(\xi) = \sum_{t=0}^{T-1} r(x_t, a_t)$

objective $J(\theta) = E_{p(\xi|\theta)}[R(\xi)] = E_{p(\xi|\theta)}\left[\sum_{t=0}^{T-1} r(x_t, a_t)\right]$

Goal: find policy parameters that maximize expected total reward

Black Box Policy Optimization

Pros:

- Policy does not directly depend on the size of the state space
- Policy can be very simple compared to MDP
  (small action space / manifold)
- Domain knowledge can be inserted into the policy directly
- Only need access to the reset model

Cons:

- Policy parameterization matters a lot!

$$\pi_\theta(x) = \underset{a \in \mathbb{A}}{\operatorname{argmin}}\ \theta^T f(x, a)$$

Black Box Policy Optimization

How should we optimize a policy directly?

Black Box Policy Optimization

How should we optimize a policy directly?

Gradient Ascent / Descent :

- Analytic gradient
- Autodiff
- Finite differencing (be careful if gradient estimate is noisy!)
- Policy gradient methods (stay tuned!)

What if gradient is hard to estimate / doesn't exist?

Black Box Policy Optimization

How should we optimize a policy directly?

- **Coordinate Descent**
  (line search along one coordinate direction at current point
  at each iteration)

Black Box Policy Optimization

How should we optimize a policy directly?

- **Coordinate Descent**
  (line search along one coordinate direction at current point
  at each iteration)
- **Simulated Annealing**
  small random perturbations $\theta + \Delta$. If $J(\theta + \Delta) > J(\theta)$, update parameter $\theta \leftarrow \theta + \Delta$. Otherwise update parameter randomly anyway according to "temperature." Lower temperature over time.

Black Box Policy Optimization

How should we optimize a policy directly?

- **Coordinate Descent**
  (line search along one coordinate direction at current point at each iteration)
- **Simulated Annealing**
  small random perturbations $\theta + \Delta$. If $J(\theta + \Delta) > J(\theta)$, update parameter $\theta \leftarrow \theta + \Delta$. Otherwise update parameter randomly anyway according to "temperature." Lower temperature over time.
- **Genetic Algorithms / Evolutionary Algorithms**
  Evolution inspired: randomly generate parameters. Best parameters "survive" and "reproduce." Parameter space explored via "crossover" and "mutation."

Black Box Policy Optimization

How should we optimize a policy directly?

- **Coordinate Descent**
  (line search along one coordinate direction at current point
  at each iteration)
- **Simulated Annealing**
  small random perturbations $\theta + \Delta$. If $J(\theta + \Delta) > J(\theta)$, update parameter $\theta \leftarrow \theta + \Delta$. Otherwise update parameter randomly anyway according to "temperature." Lower temperature over time.
- **Genetic Algorithms / Evolutionary Algorithms**
  Evolution inspired: randomly generate parameters. Best parameters "survive" and "reproduce." Parameter space explored via "crossover" and "mutation."
- **Cat Swarm Optimization**
  Inspired by the behavior of cats. This is real. Look it up.

Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead (simplex search)

"There are occasions where it has been spectacularly good. . . Mathematicians hate it because you can't prove convergence; engineers seem to love it because it often works." - John Nelder

Tries to find the optimum of a function by iteratively modifying a simplex to surround it. What `fminsearch` implements

Framed as minimization. Builds an *n+1*-dimensional simplex to find parameters in an *n*-dimensional space.

Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead

Sample n+1 points.

One iteration consists of three steps:

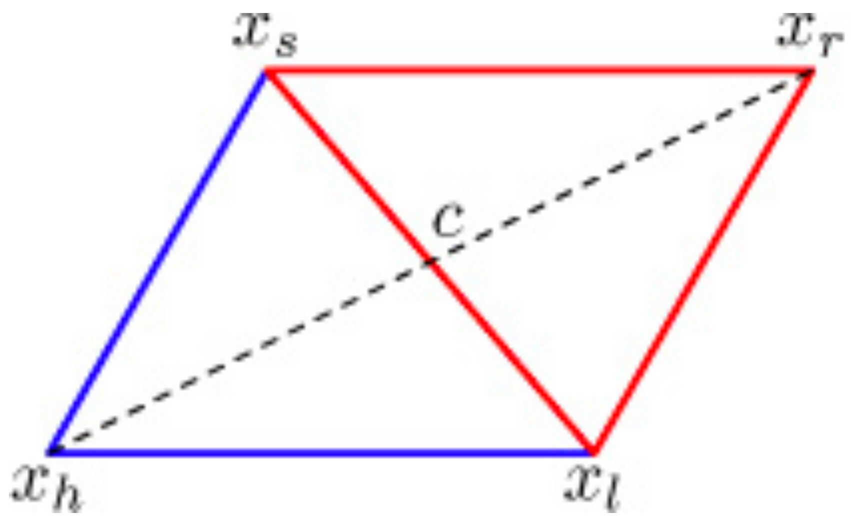
- Ordering: find **worst**, **second-worst**, and **best vertex**.
- Centroid: find centroid of **best side** (opposite worst vertex)
- Transformation: sequential rules for transforming simplex

Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: Transformation

Reflect: compute x_r, if better than second worse x_s but not as good as best x_l, replace worst x_h with x_r.

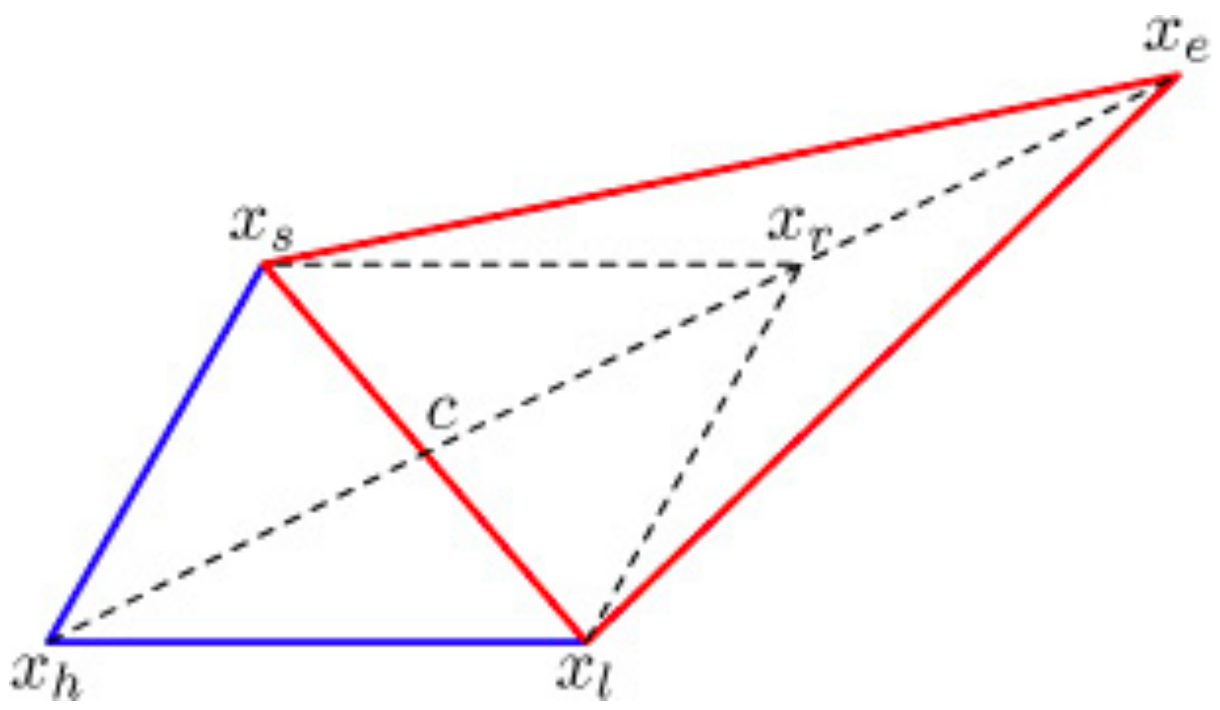


Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: Transformation

Expand: compute x_r, if best so far, expand to x_e. If better than x_r replace worst x_h, with x_e. Otherwise replace x_h with x_r.

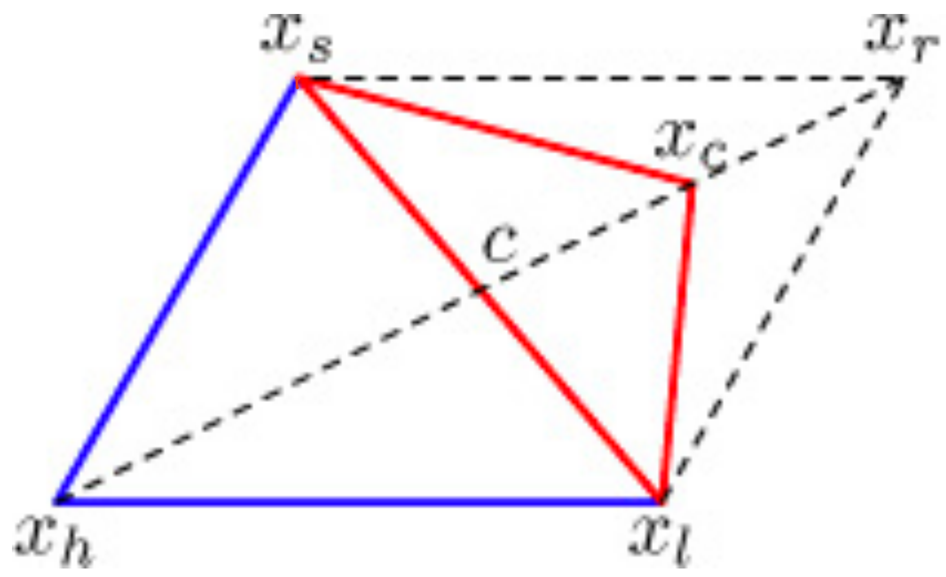


Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: Transformation

Contract (outside): compute x_r, if worse than x_s, but better than x_h, compute x_c. If x_c is better than x_r, replace x_h with x_c.

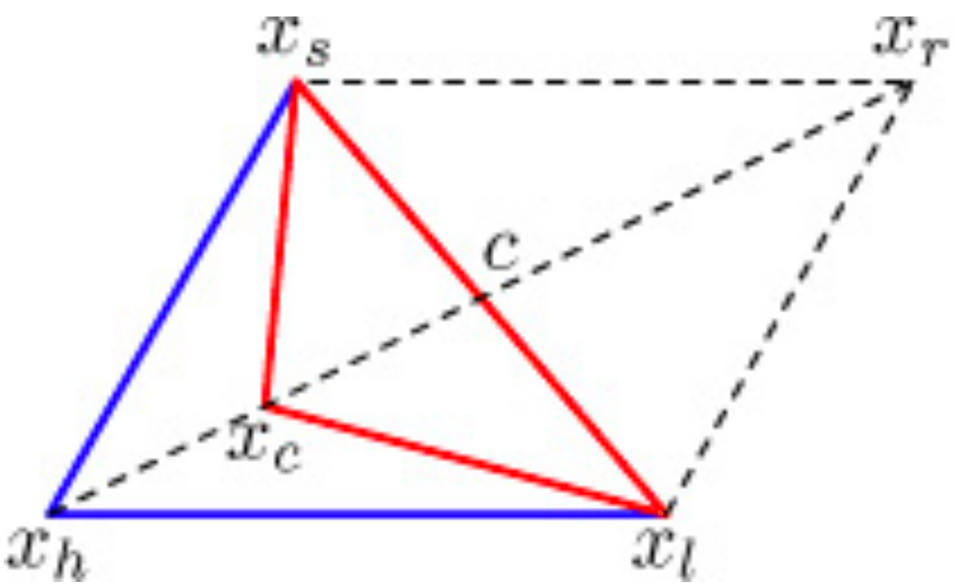


Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: Transformation

Contract (inside): compute x_r, if worse than x_h, compute x_c.
If x_c is better than x_h, replace.

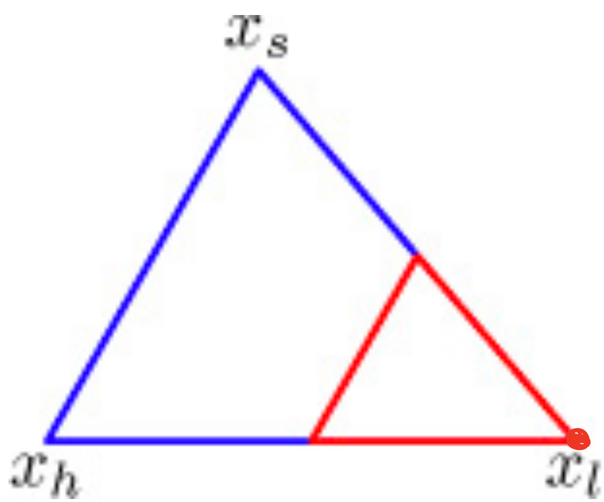


Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: Transformation

Shrink: if none of the above work, shrink the simplex toward the best point.

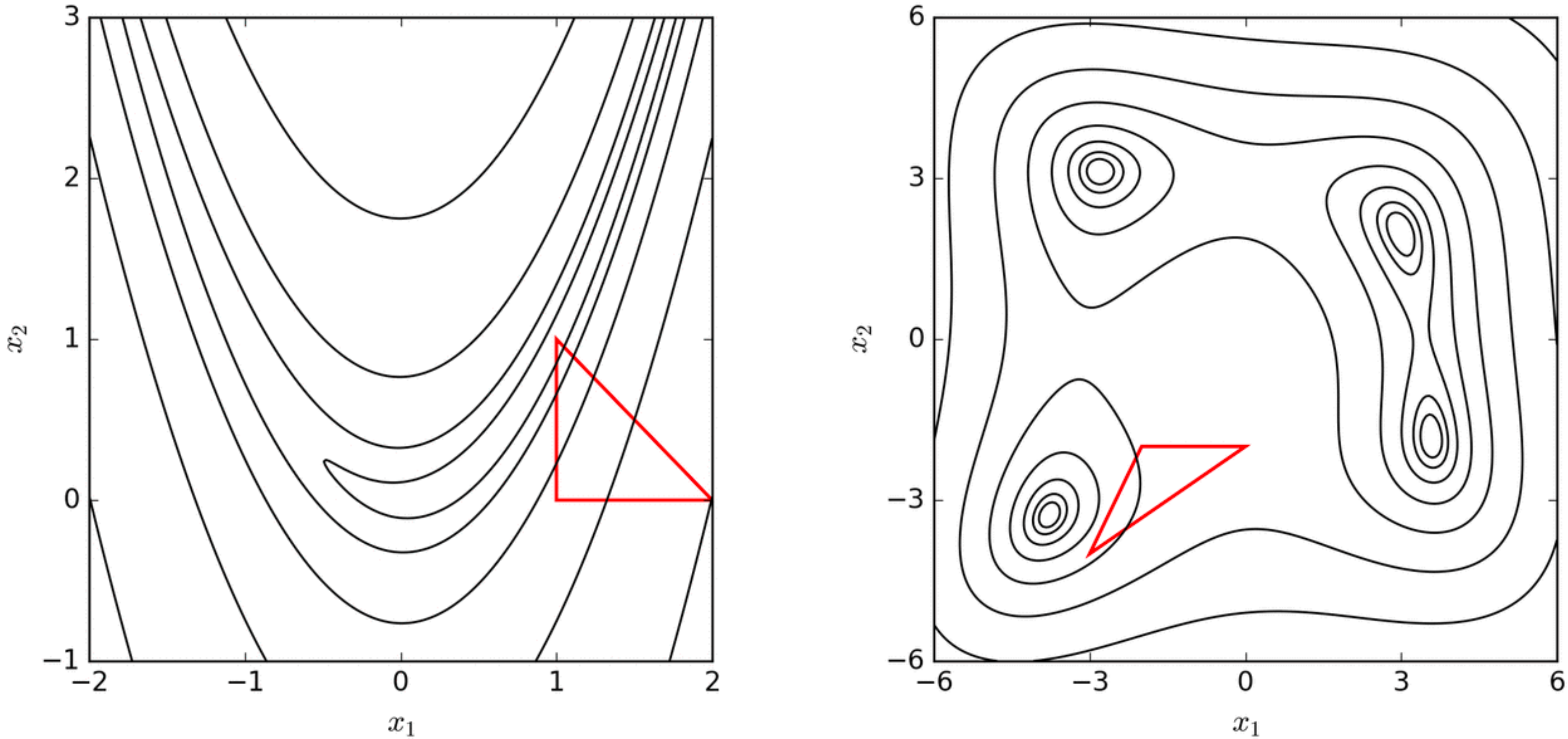


Termination: when simplex is sufficiently small, or values are sufficiently close, or two many iterations, terminate.

Black Box Policy Optimization

How should we optimize a policy directly?

Nelder-Mead: animations of wikipedia



Black Box Policy Optimization

How should we optimize a policy directly?

Cross Entropy Method

1. Start with distribution of samples over parameter space (typically Gaussian)

2. Evaluate each sample.

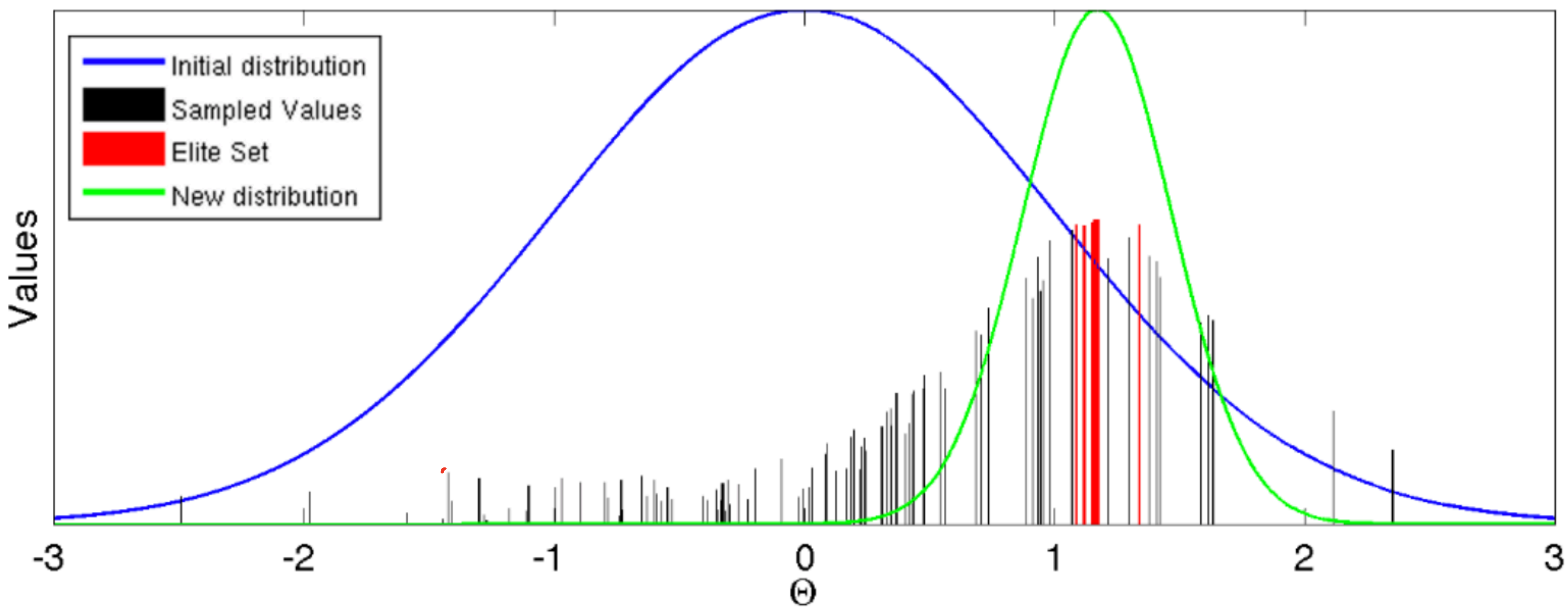
3. Keep top 1-5% of samples ("elite set").

4. Estimate parameters of new distribution from elite set (e.g. mean, covariance). Can interpolate with previous distribution.

5. Resample

How should we optimize a policy directly?

Cross Entropy Method



Black Box Policy Optimization

How should we optimize a policy directly?

Cross Entropy Method

Issues:

- does not handle multi-modal distributions well
- covariance can become singular, need to regularize
- may not converge if policy is stochastic

Black Box Policy Optimization

Issues with Black Box Optimization:

- It is hard! May not converge, typically finds local optima.
- Must evaluate $J(\theta)$ many times. This can be **really** expensive.