An introduction to Numerical Optimization

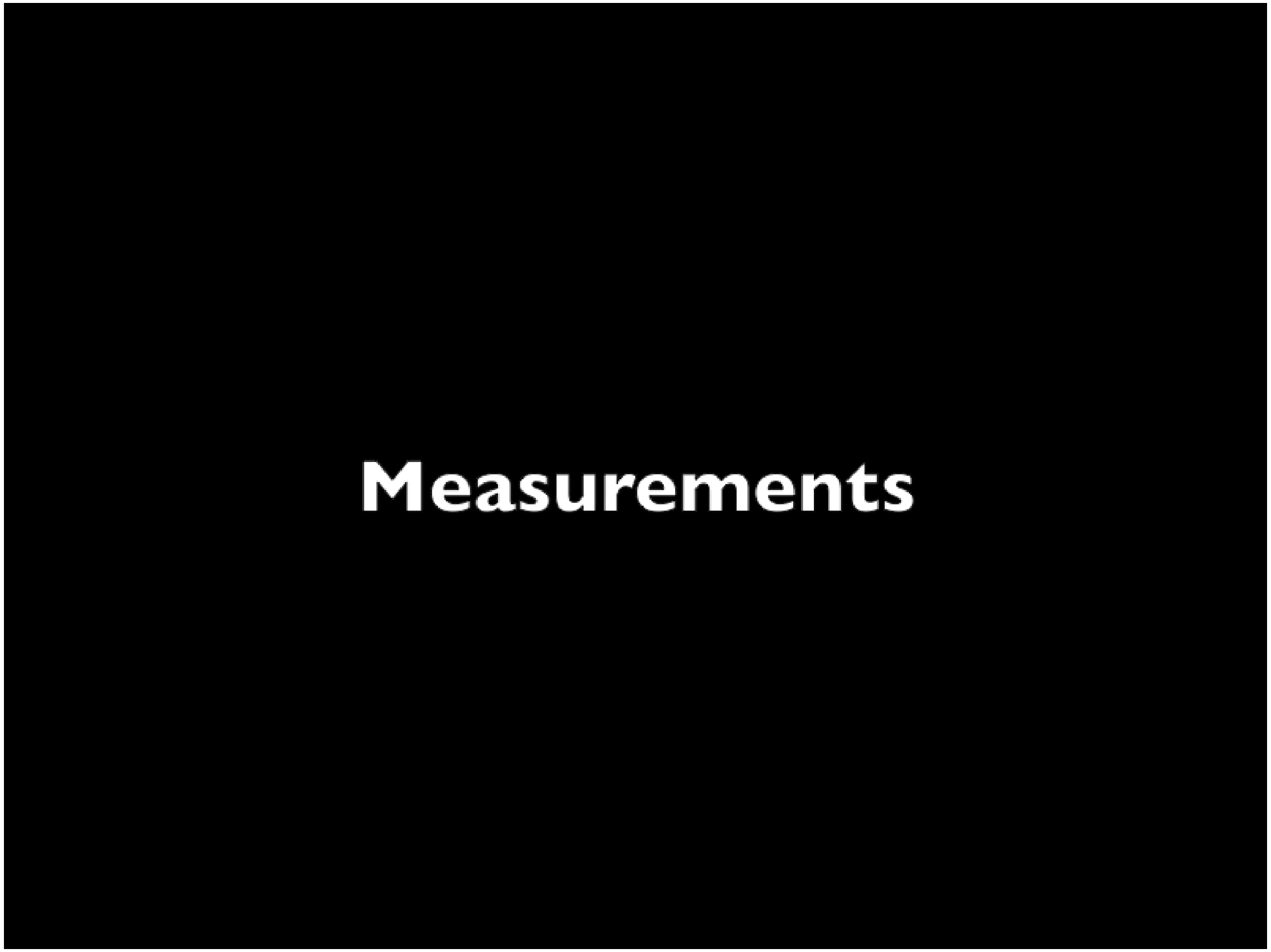
Stelian Coros

Plan for Today

- A fast and furious tour through numerical optimization
  - Unconstrained Optimization
    - Gradient Descent
    - Newton’s Method
  - Constrained Optimization
    - Newton’s Method
    - Quadratic Programming
  - Stochastic Optimization
  - Discrete Optimization

Optimization in Graphics

- Simulation and Material Parameter Estimation



Introduction to Optimization

- Optimization involves finding an "optimal value"

- i.e. Maximizing a profit, minimizing forces, etc…

$$\underbrace{\boxed{\min}}_{\text{minimize}}\ \overbrace{\boxed{f\left(x\right)}}^{\text{Cost Function}}$$

Introduction to Optimization

- Optimization involves finding an "optimal value"

- i.e. Maximizing a profit, minimizing an area etc…

$$\boxed{x^*} = \arg\min f(x)$$

Optimal Solution

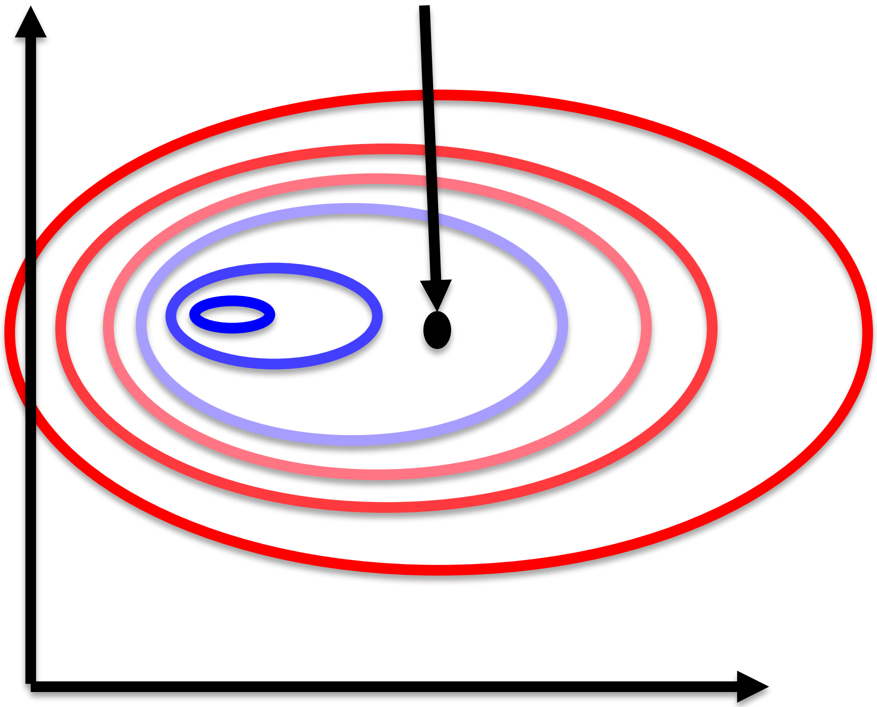
Types of Optimization

- Continuous vs. Discrete
- Constrained vs. Unconstrained

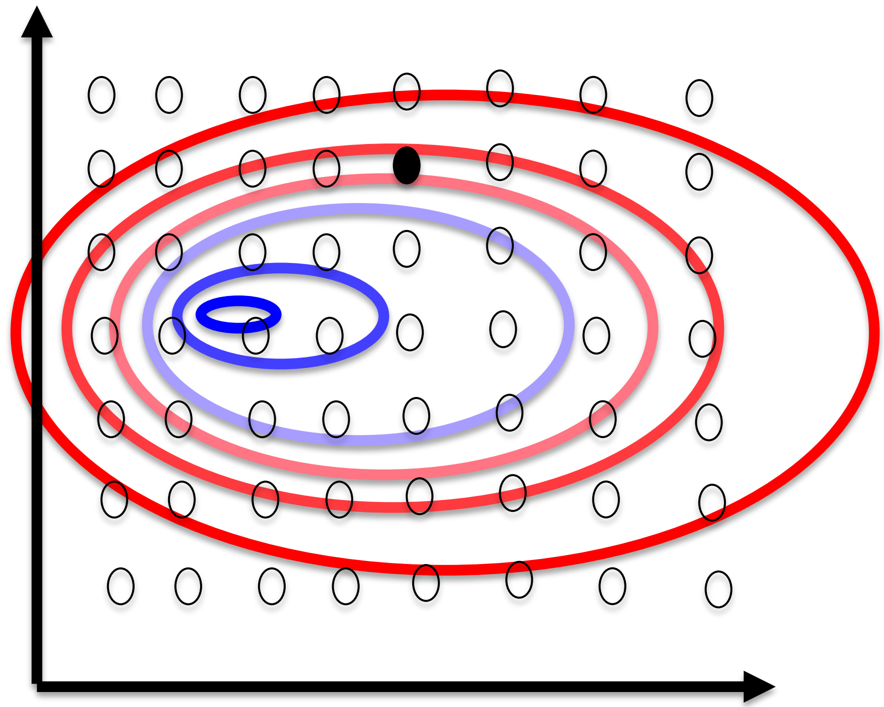
Continuous

Discrete

This point can move smoothly

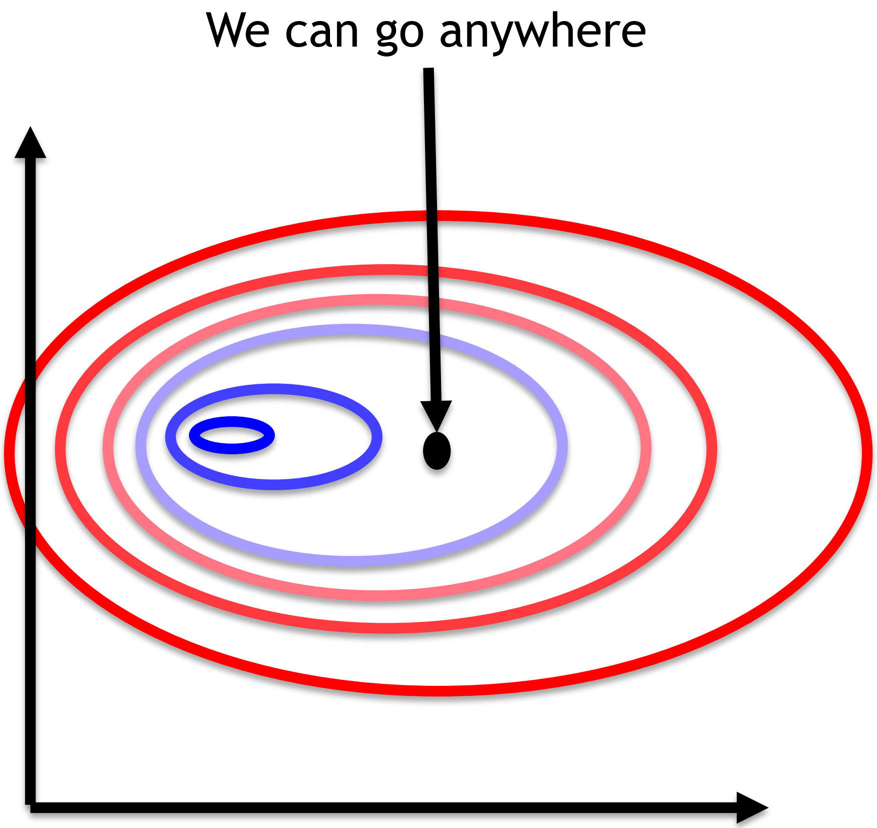


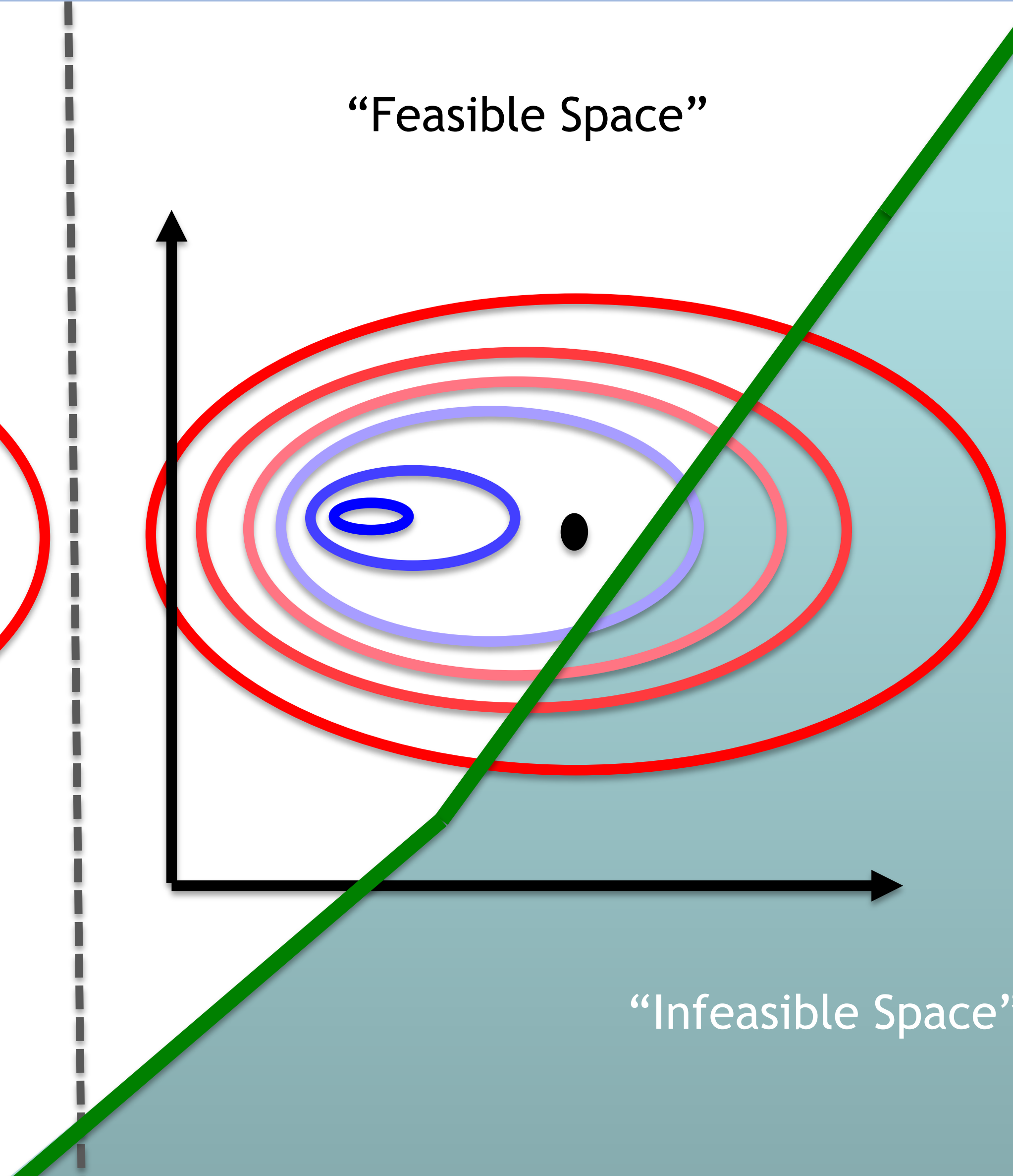
Choose from discrete points in parameter space



Unconstrained

Constrained





Types of Optimization

- $\boxed{\text{Continuous}}$ vs. Discrete
- Constrained vs. $\boxed{\text{Unconstrained}}$

Continuous Optimization

- We're solving

$$x^* = \arg\min f(x)$$

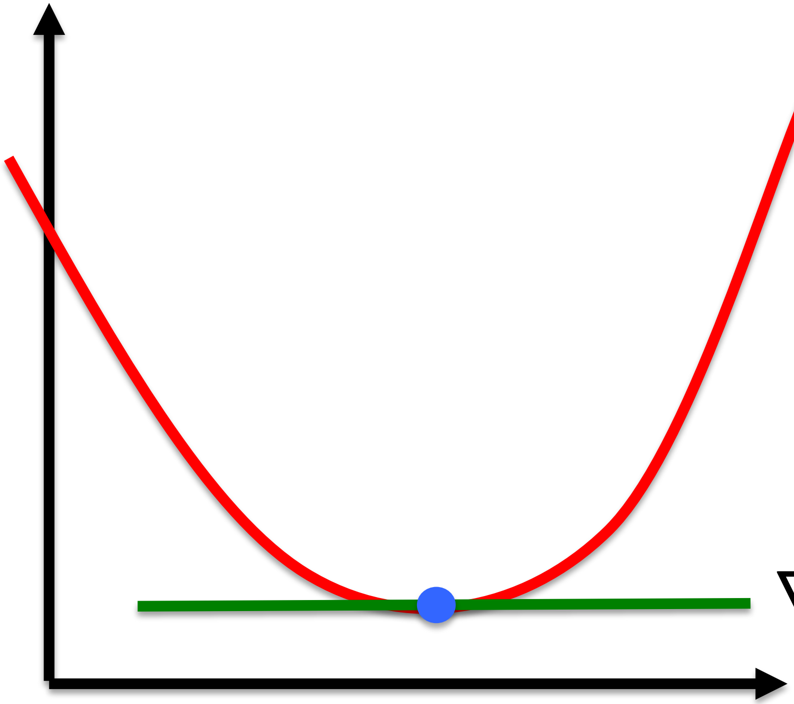
- How do we know we've found a potential solution?

IMPORTANT!!!!!

$$\nabla \mathbf{f}(\mathbf{x}^*) = \mathbf{0}$$

Potential Solutions

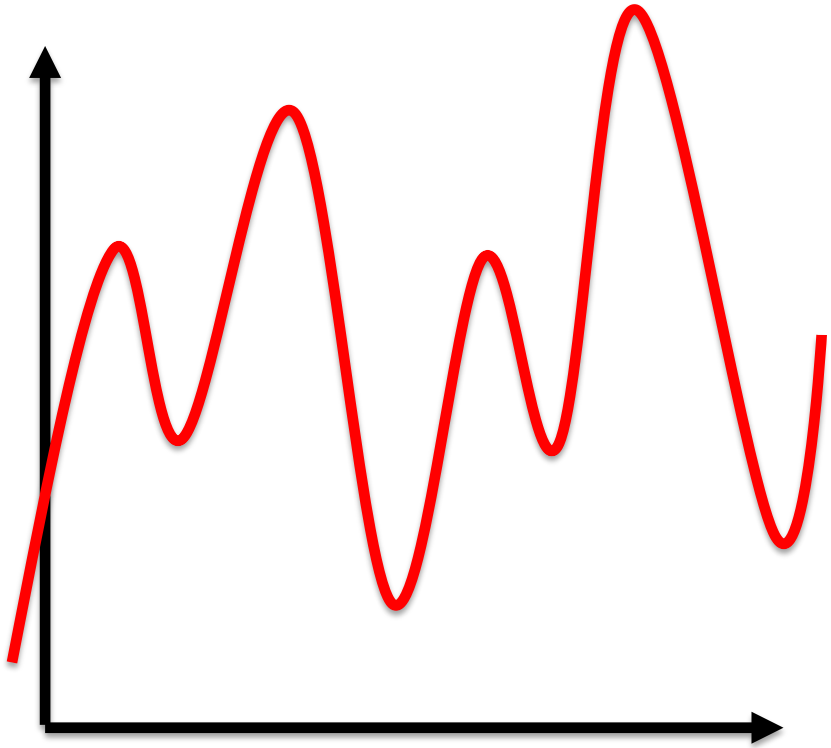
- Intuitively we look for a flat point on the cost function



$$\nabla \mathbf{f}\left(\mathbf{x}^*\right) = \mathbf{0}$$

Potential Solutions

- Sometimes that’s easier said than done



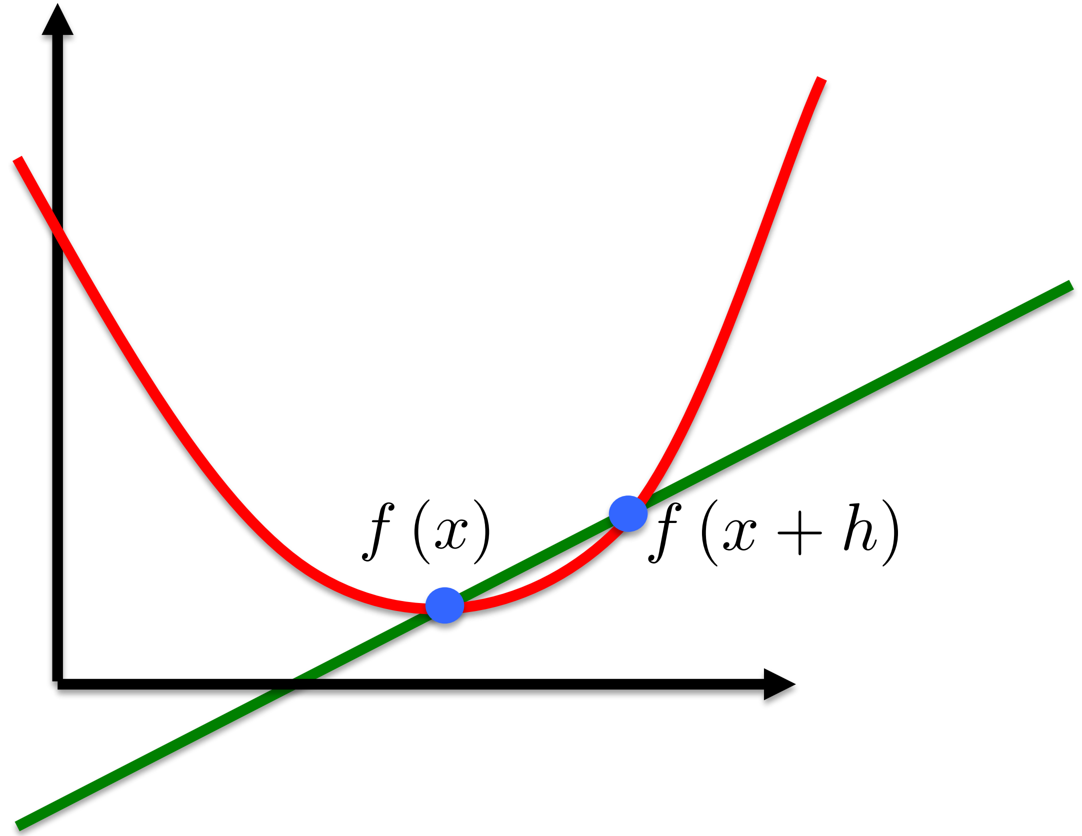
Computing Derivatives

- Analytically (pen and paper, mathematica)
- Finite Differences (via Taylor series)

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f^{(3)}(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \ldots.$$

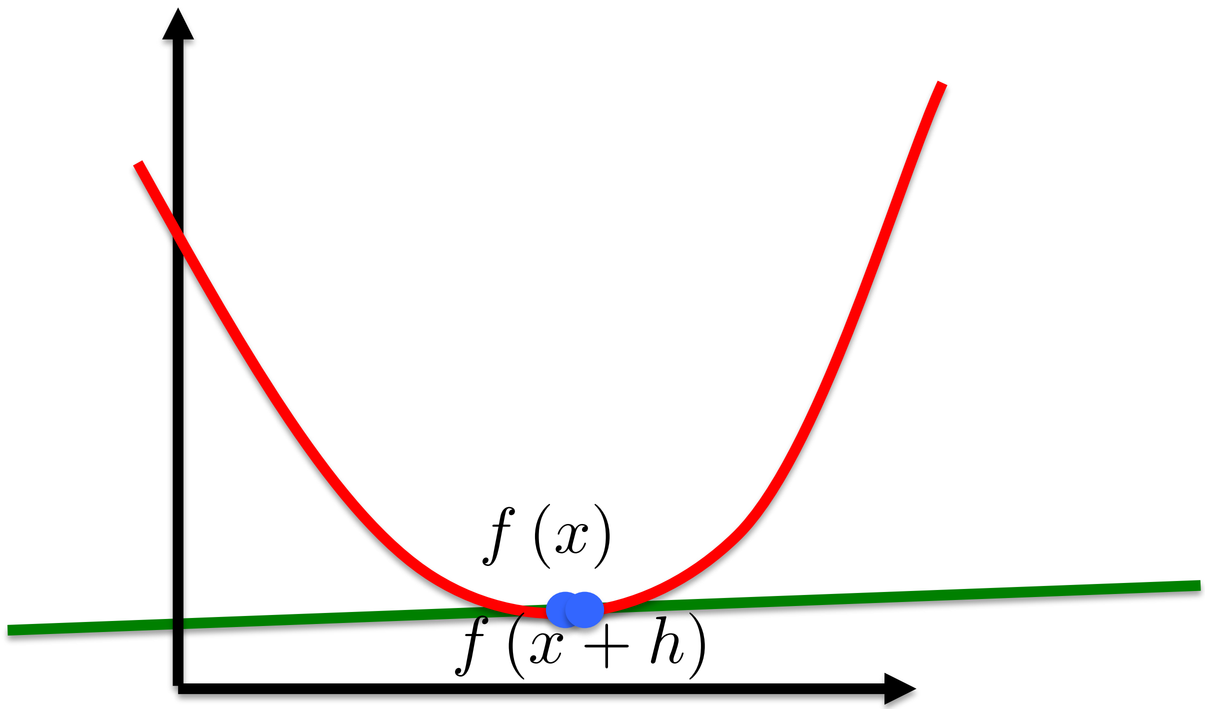
Potential Solutions

- Computing the gradient requires a limit



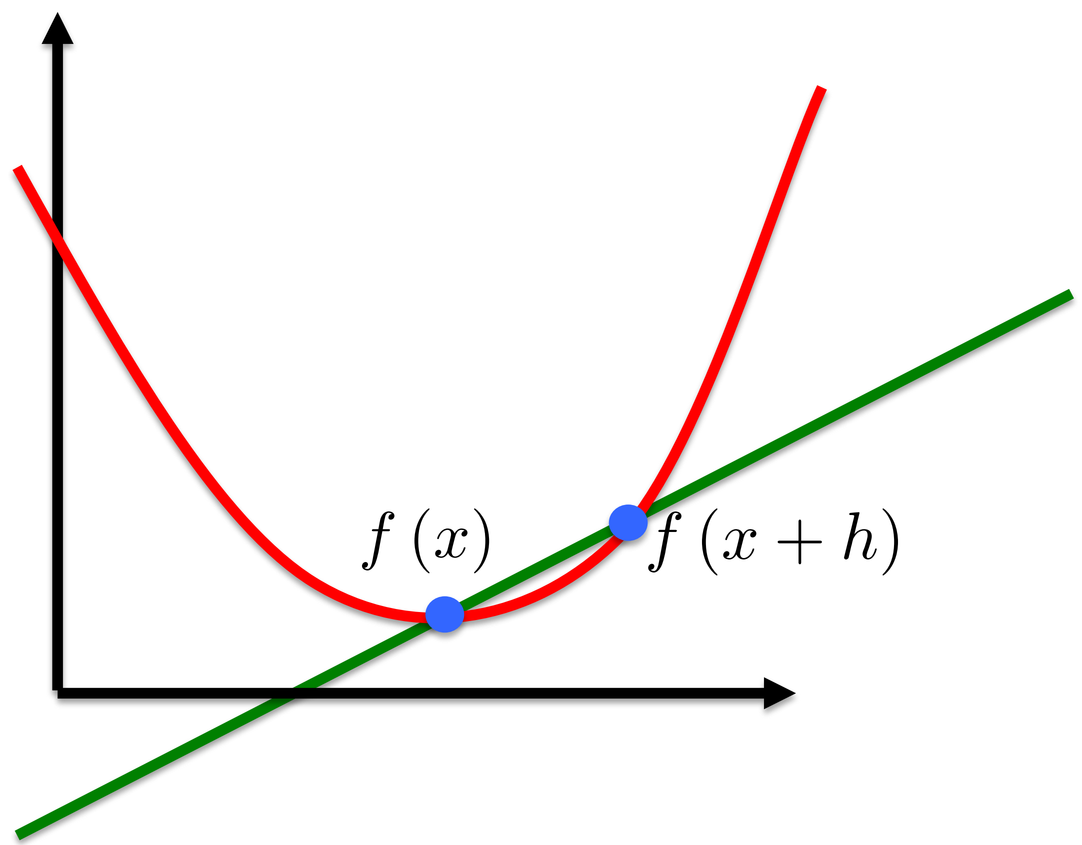
Potential Solutions

- Computing the gradient requires a limit



Potential Solutions

- In Finite Differencing we choose *h* and estimate the derivative numerically



Continuous Optimization

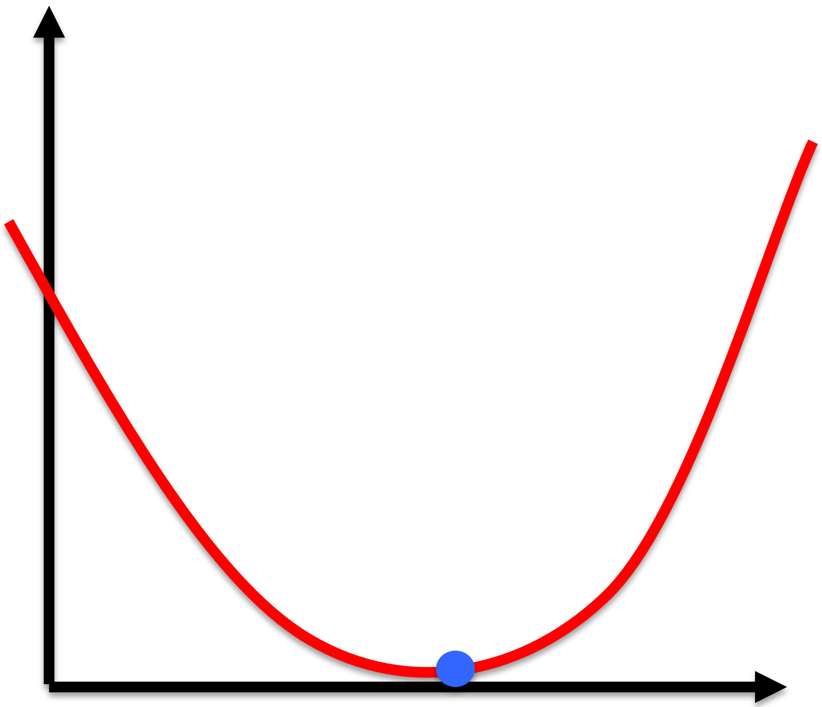
- General, continuous optimizations are difficult to solve – but we keep trying anyway
- We focus on certain classes of problems that are solvable

Convex Optimization

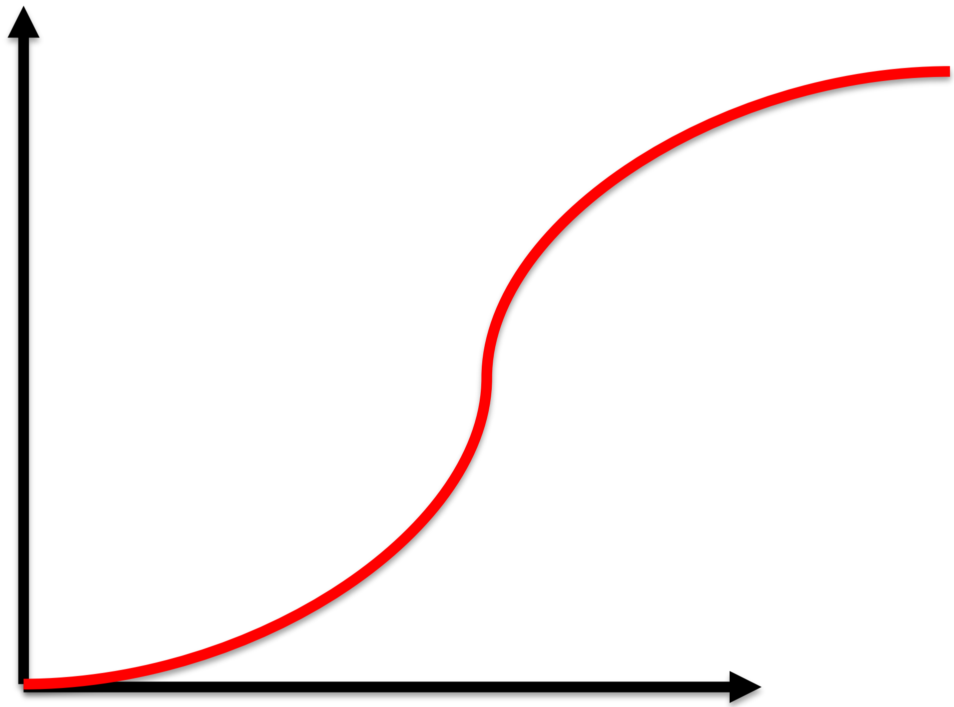
Convex Optimization

- Convex optimizations are ones that have a single minimum
- Let’s look at some examples of convex cost functions

Convex Optimization



Is This Convex ?



Descent Algorithms

- Idea: Follow search directions that reduce the cost!
- Two Types
  - Gradient Descent
  - Newton's Method

Gradient Descent

- Recall that the gradient of a function is given by

$$\nabla f\left(\mathbf{x}\right) = \begin{pmatrix} \frac{\partial f}{\partial \mathbf{x}_1} & \frac{\partial f}{\partial \mathbf{x}_2} & \cdots & \frac{\partial f}{\partial \mathbf{x}_n} \end{pmatrix}$$

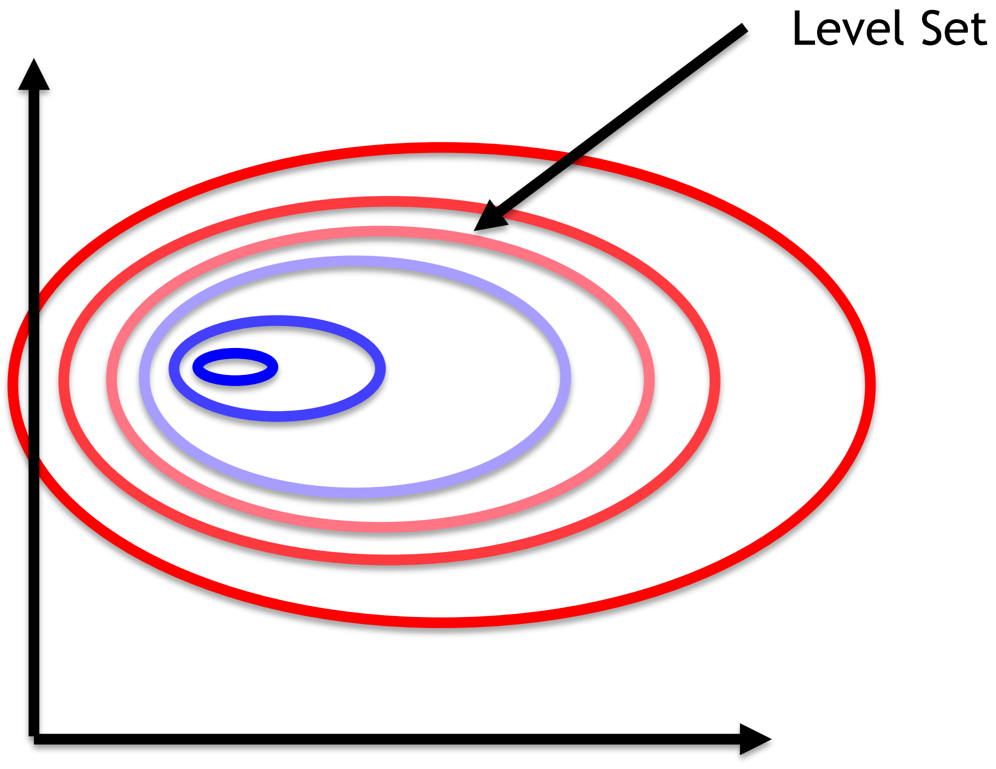
Gradient Descent

- Recall that the gradient of a function is given by

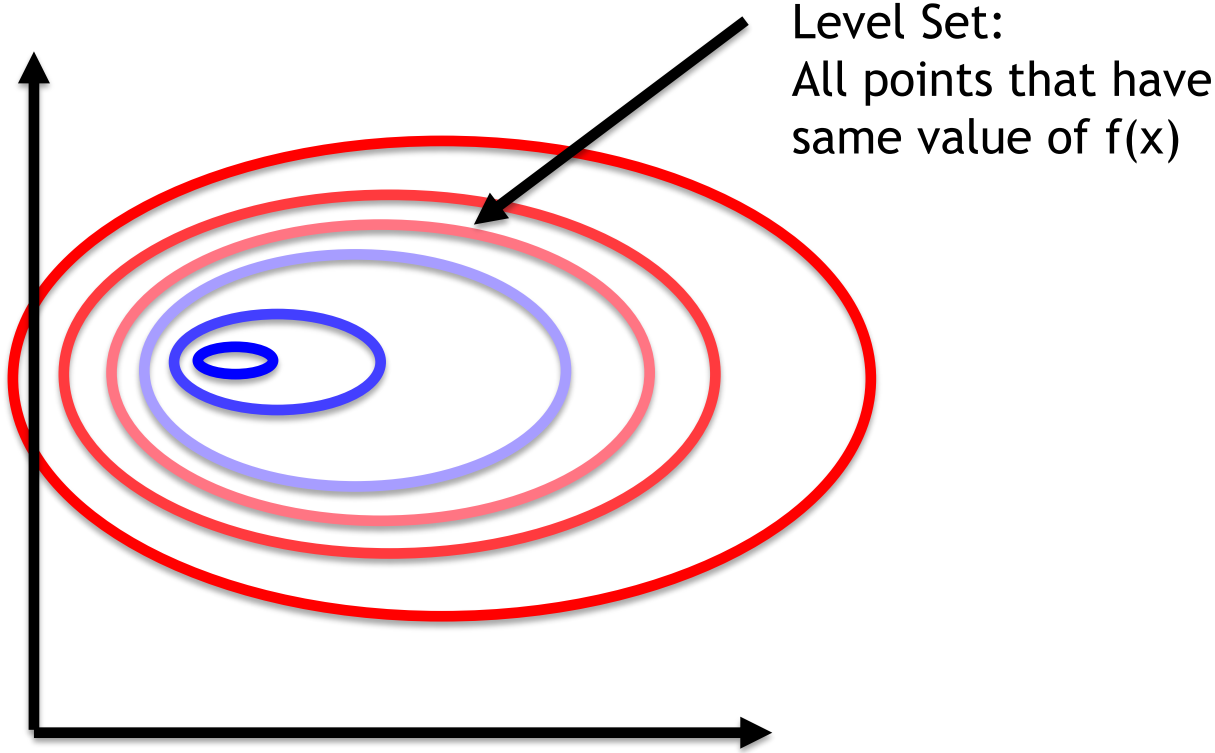
$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial \mathbf{x}_1} & \frac{\partial f}{\partial \mathbf{x}_2} & \cdots & \frac{\partial f}{\partial \mathbf{x}_n} \end{pmatrix}$$

- Points in direction of maximum ascent

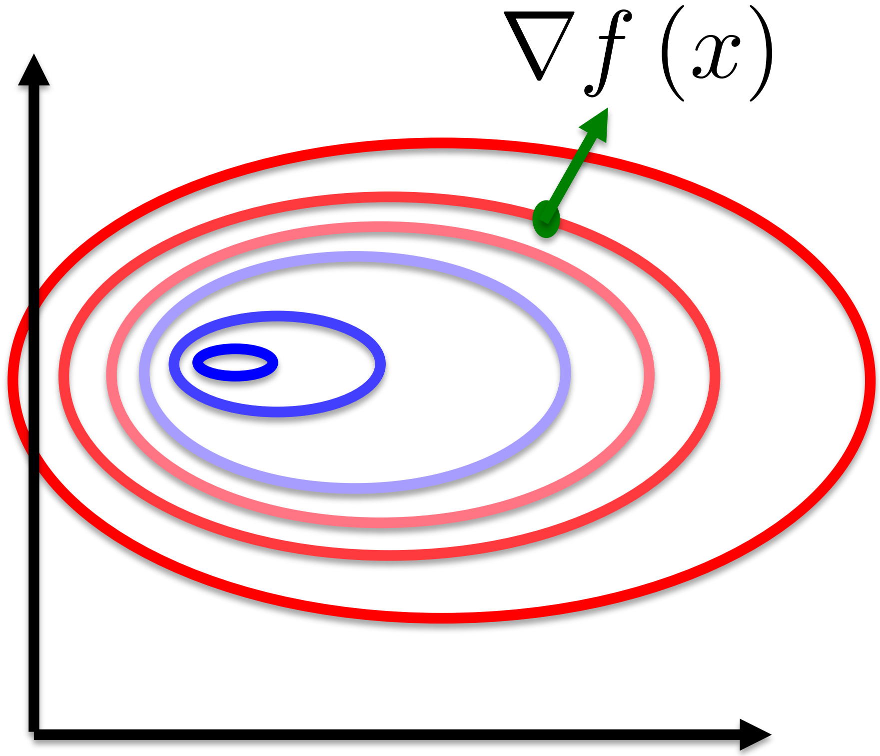
An Aside: Level Sets



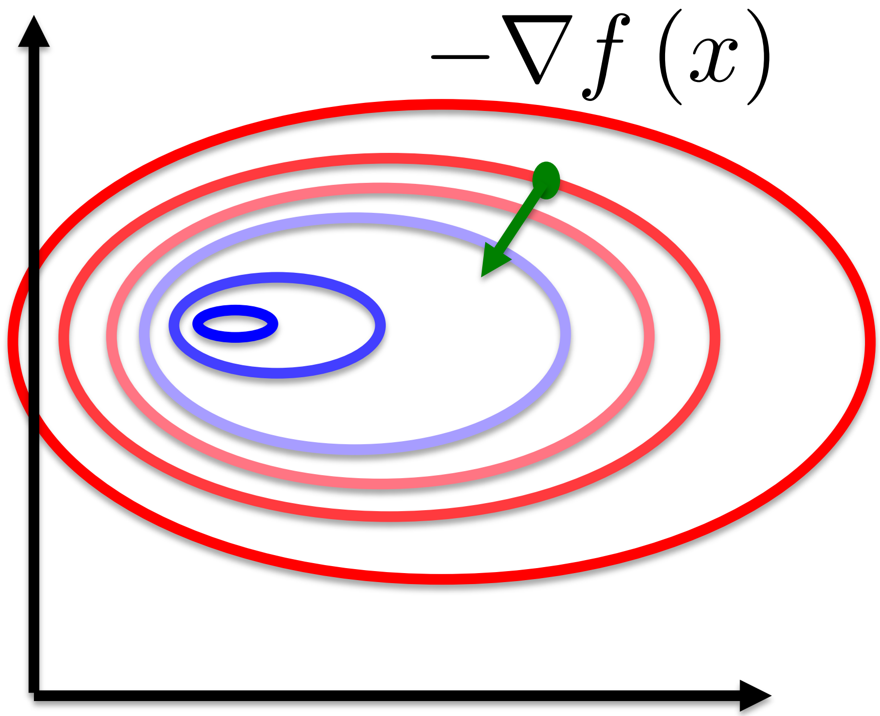
An Aside: Level Sets



Gradient Descent



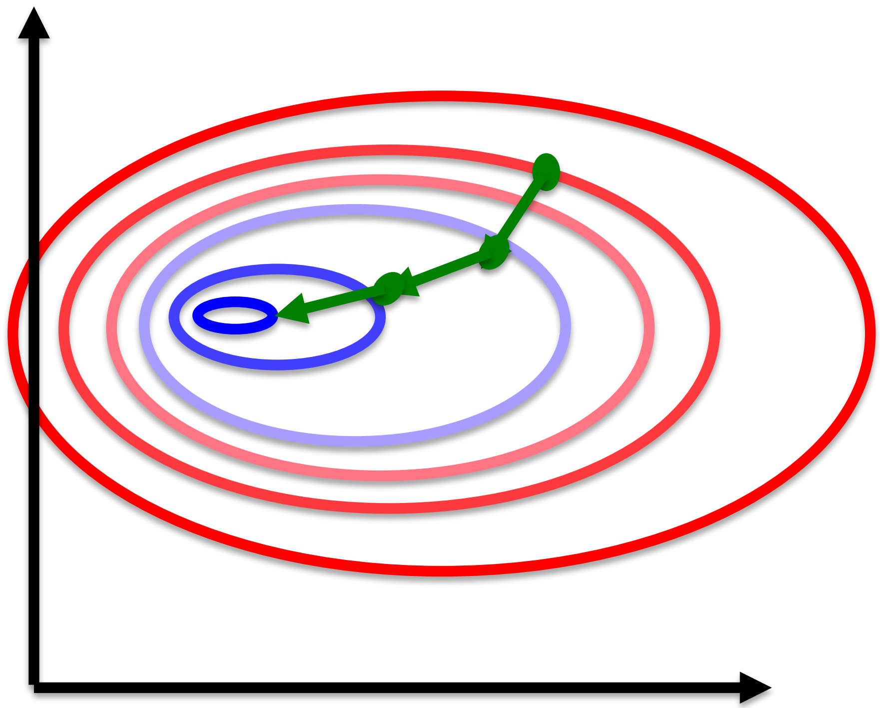
Gradient Descent



Simple Gradient Descent Algorithm

- While not at an optimal point
  - Compute the gradient at current point (x)
  - Move to new point $x = x - \boxed{h} \nabla f(x)$
  - Step size h: typically need line search

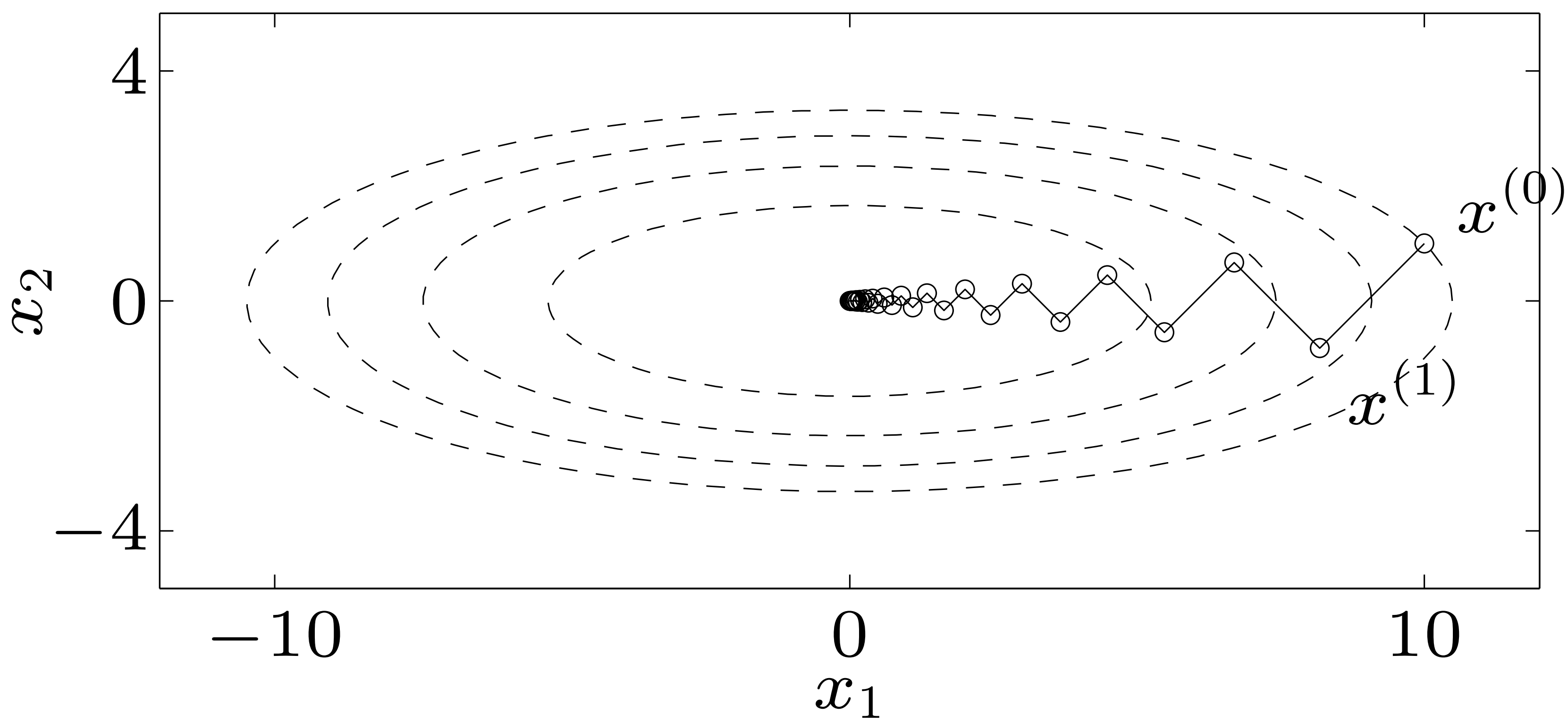
Gradient Descent



Gradient Descent

- Good:
  - Simple to implement
- Bad:
  - Sometimes converges badly

Gradient Descent

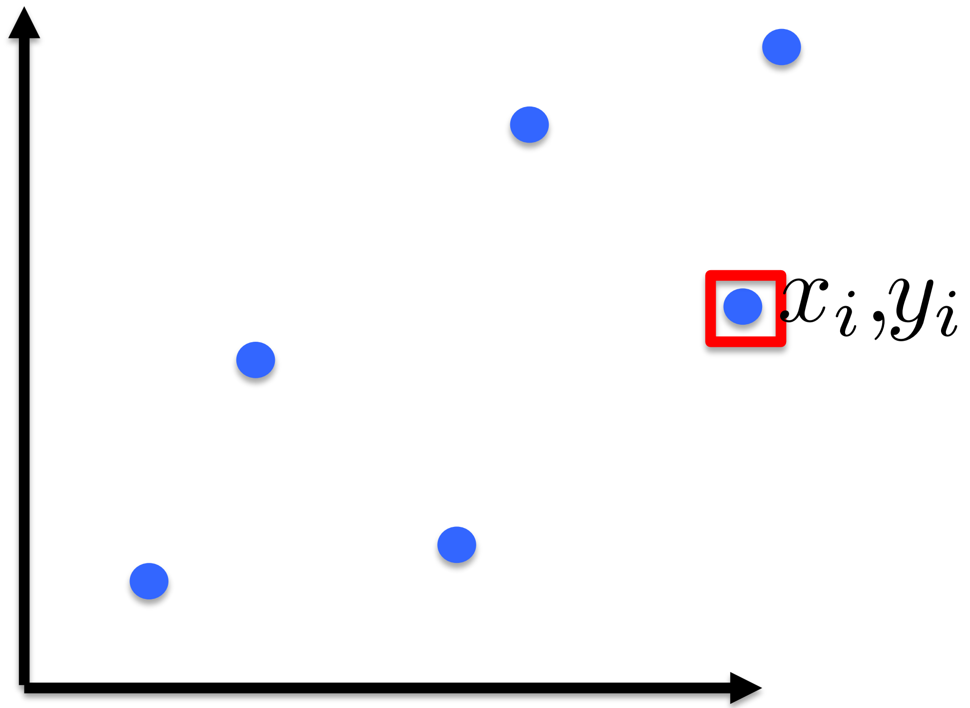


Newton’s Method

- Can we choose better search directions?
- Hint: quadratic functions are very nice!

A Simple Example: Least Squares

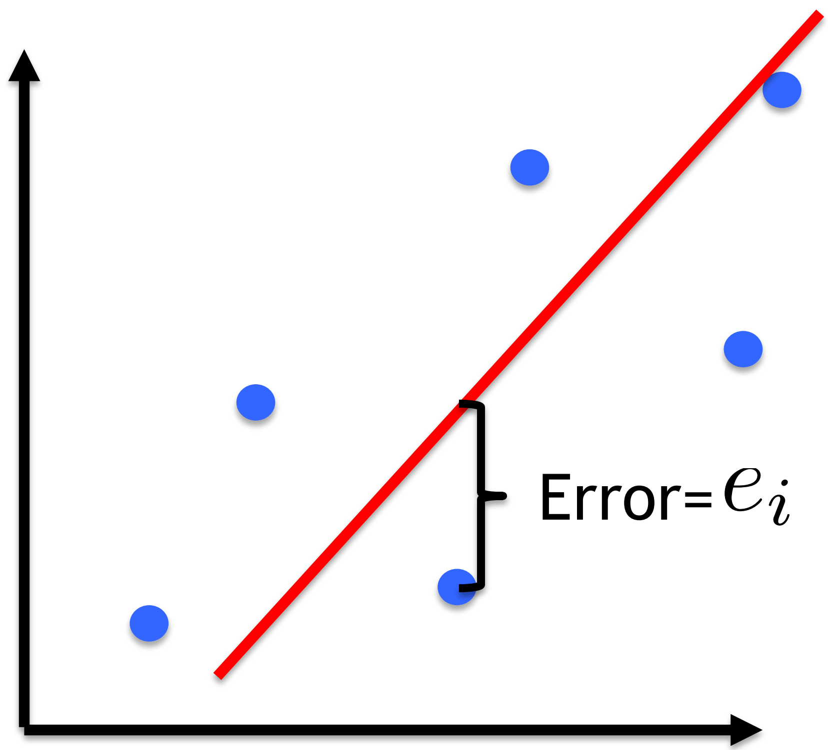
- Least Squares Fitting of a Curve



- Want to find a line, mx + c, that is a “best fit”

A Simple Example: Least Squares

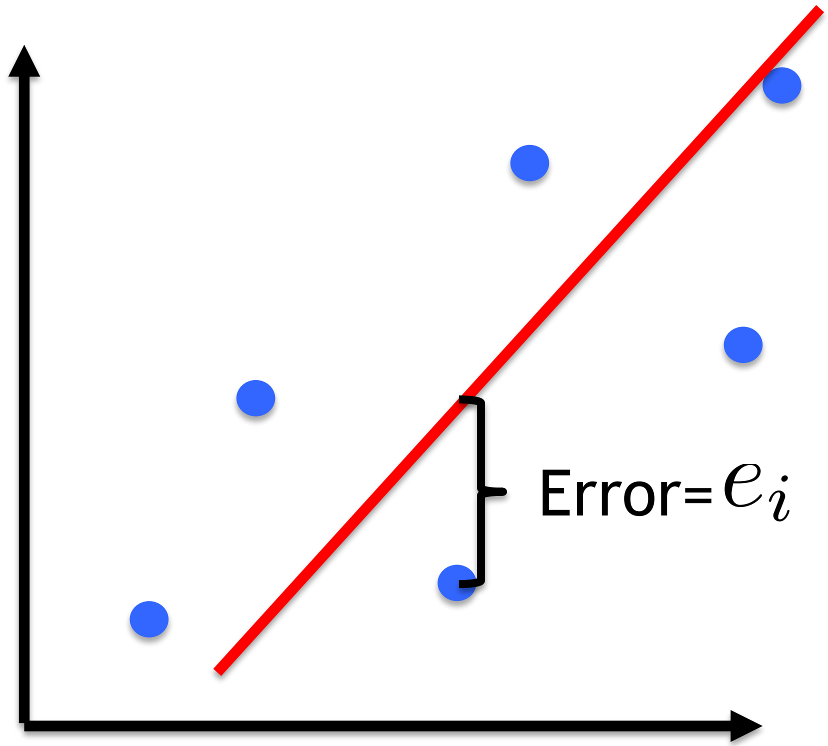
- Least Squares Fitting of a Curve



- Want to find a line, mx + c, that is a “best fit”

A Simple Example: Least Squares

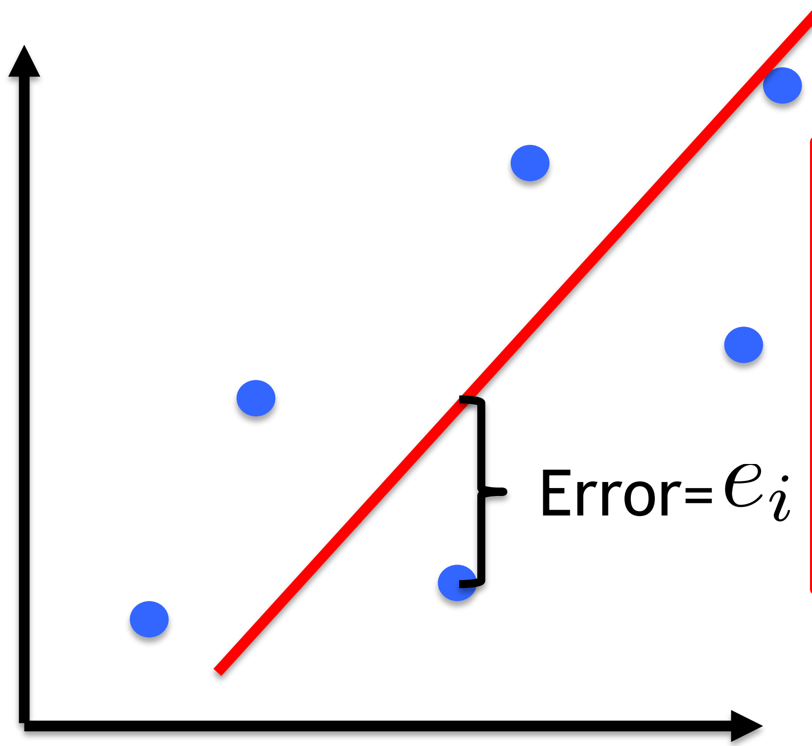
- Least Squares Fitting of a Curve



$$e_i = y_i - mx_i - c$$

A Simple Example: Least Squares

- Minimize sum of squared errors



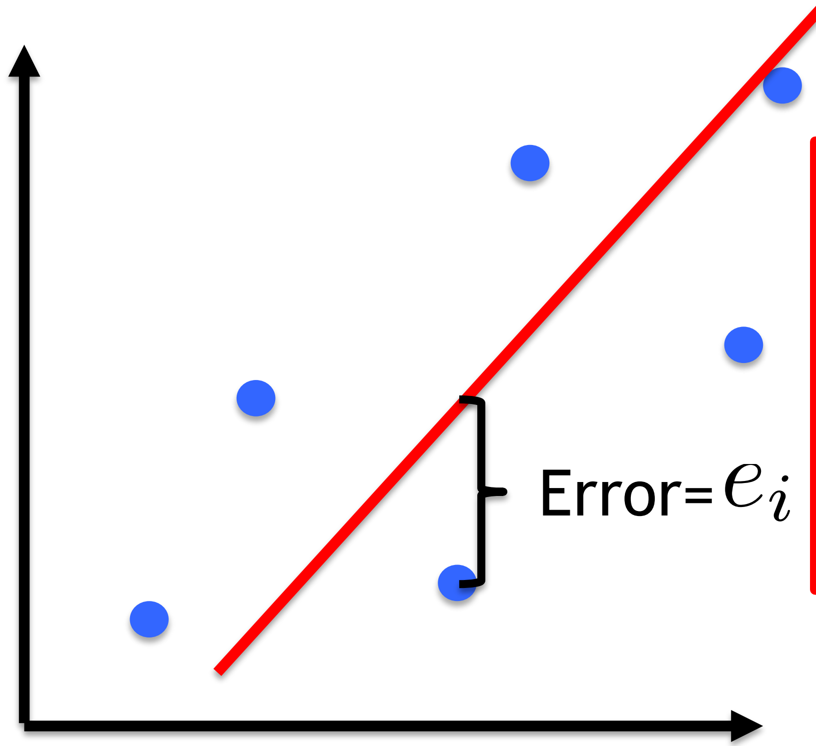
$$\mathbf{A} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_n & 1 \end{pmatrix}$$

Total Error = $\|\mathbf{A}\theta - \mathbf{y}\|^2$

A Simple Example: Least Squares

- Minimize sum of squared errors

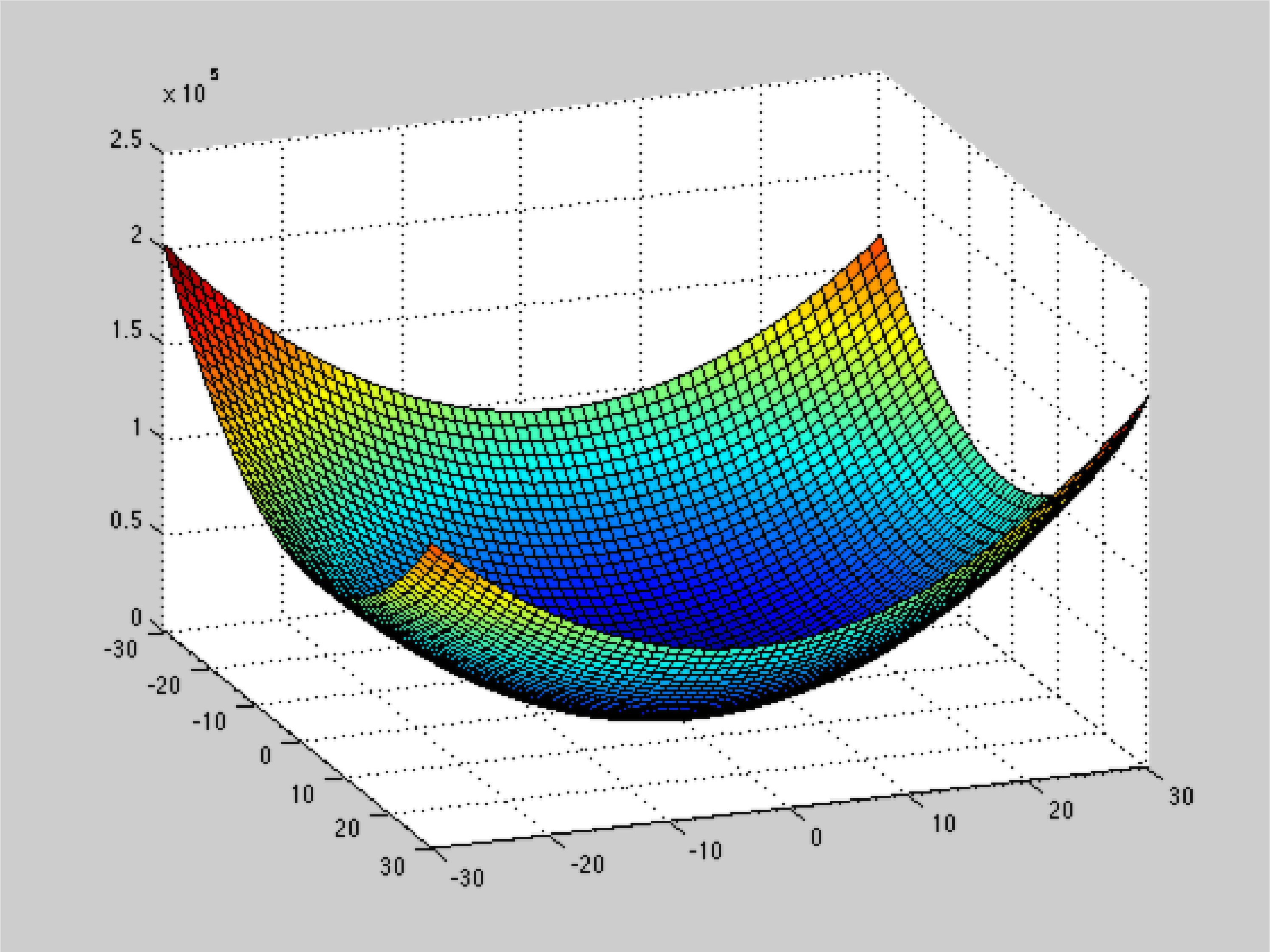
$$\boxed{\theta = \begin{pmatrix} m \\ c \end{pmatrix}}$$



$$\boxed{\mathbf{A} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_n & 1 \end{pmatrix}}$$

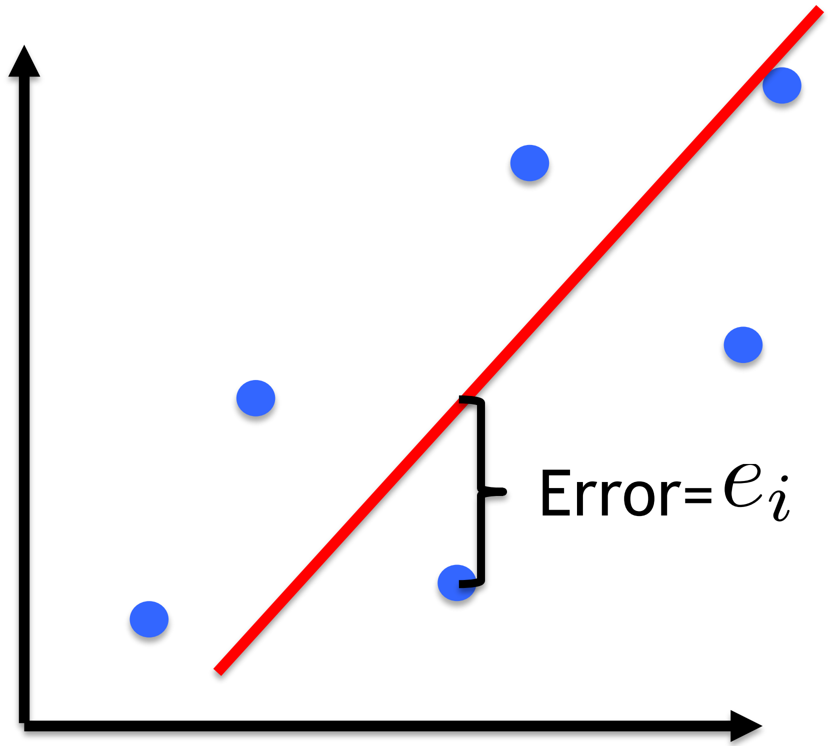
Sum of Squared Error = f(x) = $||\mathbf{A}\theta - \mathbf{y}||^2$

Simple Example: Least Squares



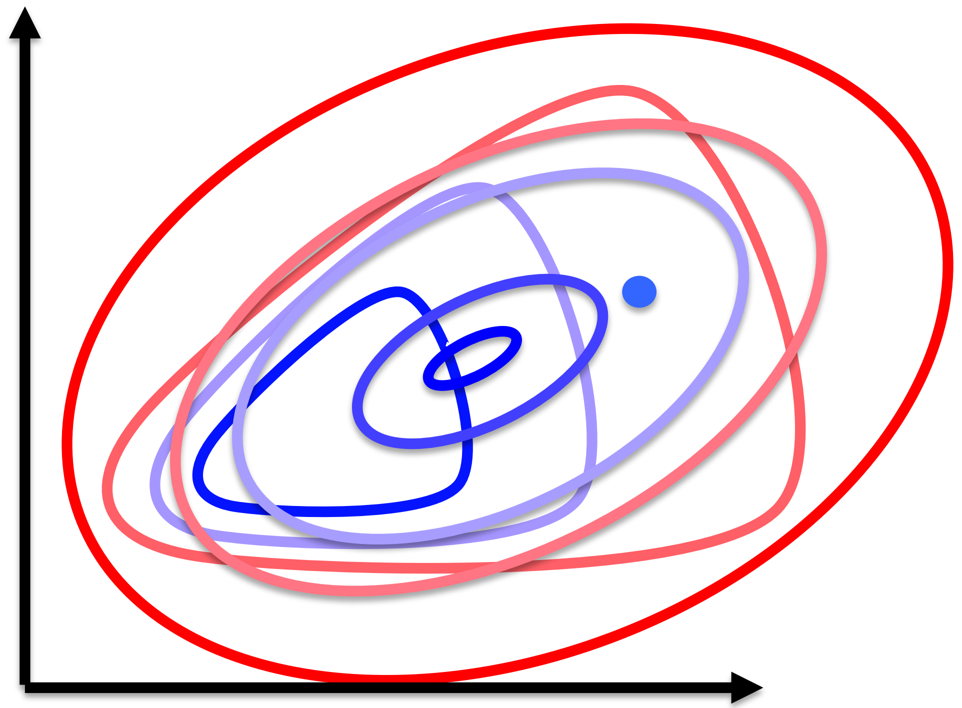
A Simple Example: Least Squares

- Solution given by the normal equations



Solution = $\boxed{\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{y}}$ $\nabla \mathbf{f}\left(\mathbf{x}^*\right) = \mathbf{0}$

Newton's Method



Choose best descent direction according to quadratic approximation

Newton's Method

- How do we get an approximation ?
- Taylor Series!

$$f\left(\mathbf{x}^c + \Delta\mathbf{x}\right) \approx f\left(\mathbf{x}^c\right) + \Delta\mathbf{x}^T\mathbf{g} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}$$

$$\boxed{\nabla f\big|_{\mathbf{x}^c}}$$

$$\boxed{H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \frac{\partial^2 f}{\partial x_2\,\partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\,\partial x_1} & \frac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}}$$

Newton's Method

- We minimize the model problem
- Find where the gradient is zero

$$f\left(\mathbf{x}^c\right) + \Delta\mathbf{x}^T\mathbf{g} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}$$

Newton's Method

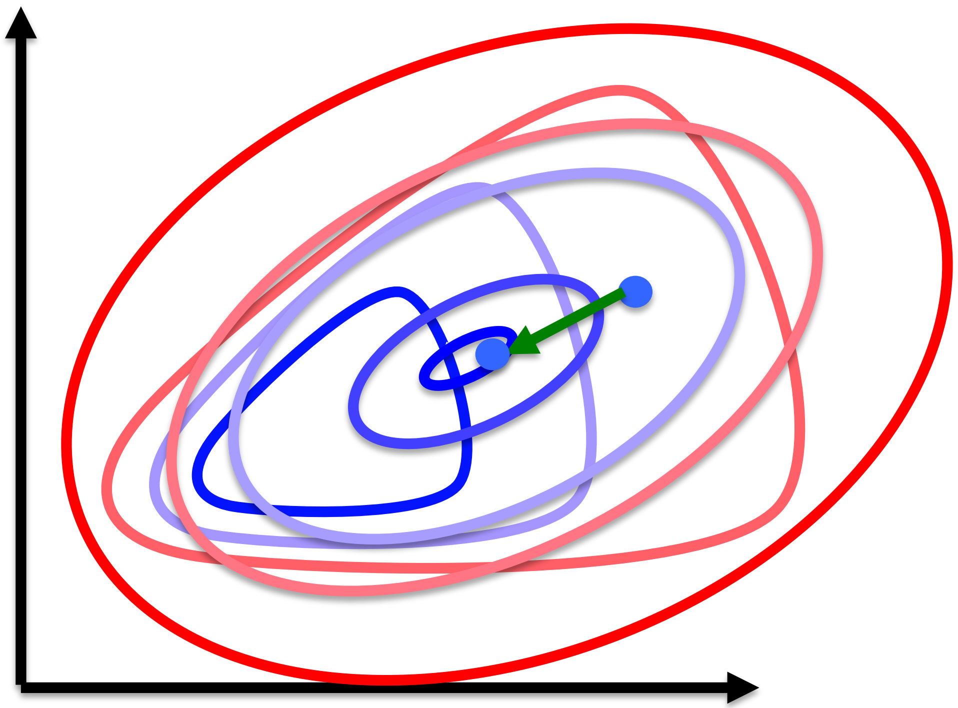
- We minimize the model problem
- Find where the gradient is zero

Model: $f\left(\mathbf{x}^c\right) + \Delta\mathbf{x}^T\mathbf{g} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}$

Gradient: $\mathbf{H}\Delta\mathbf{x} + \mathbf{g} = \mathbf{0}$

Increment: $\Delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{g}$

Newton’s Method



Newton's Method

- Initialize $\mathbf{x}^c$
- While not at optimal point
  - Compute gradient (**g**) and Hessian (**H**)
  - Compute $\Delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{g}$
  - Update $\mathbf{x}^c = \mathbf{x}^c + \Delta\mathbf{x}$

Gradient Descent vs. Newton's Method

- Gradient Descent is very simple
- Newton's Method converges faster (near solution, quadratic vs linear)
- Available Newton's Method Implementations:
  - MATLAB: fminunc
- Quasi-Newton Methods
  - LBFGS: http://www.chokkan.org/software/liblbfgs/

Examples of Optimization in Engineering

- Static Equilibrium: Find the minimum energy state of a deformable object
- Typically done using a Newton's method

Examples from Engineering

ENGINEERPEDIA

Types of Optimization

- Continuous vs. Discrete
- Constrained vs. Unconstrained

Constrained Optimization

- Optimization involves finding an “optimal value”

- i.e. Maximizing a profit, minimizing an area etc…

$$\min f\left(x\right)$$
$$s.t\ \boxed{\mathbf{c}_i\left(\mathbf{x}\right)} = 0$$

Equality Constraints

Constrained Optimization

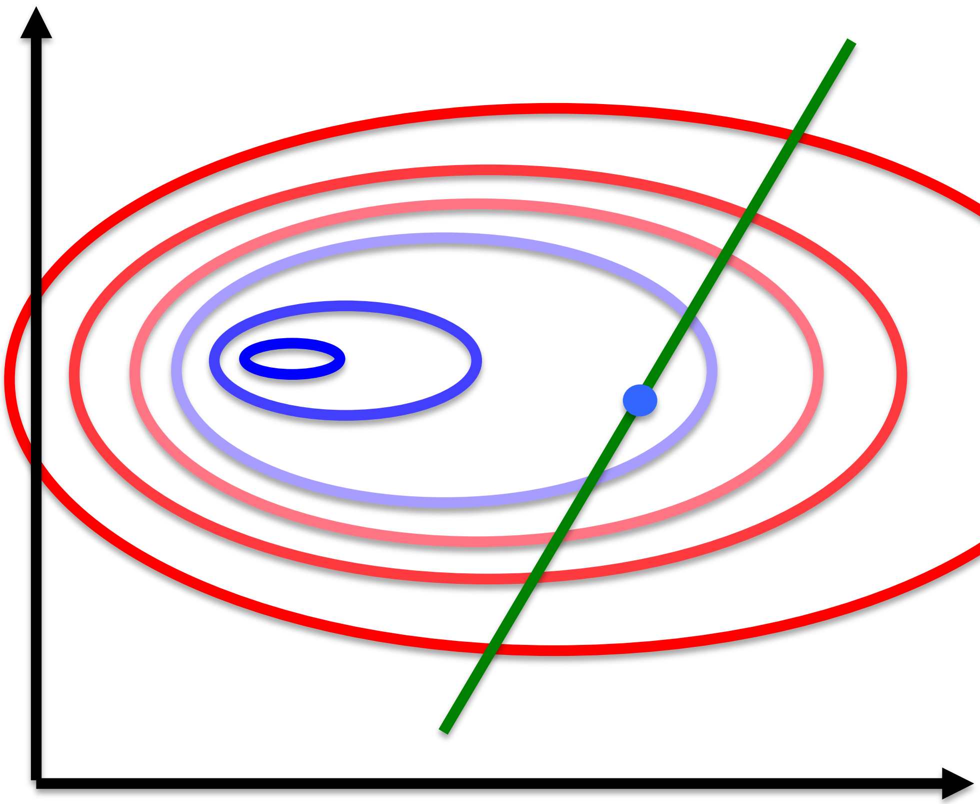
- Optimization involves finding an “optimal value”

- i.e. Maximizing a profit, minimizing an area etc…

$$\min f\left(x\right)$$
$$s.t\ \boxed{Ax} = \mathbf{b}$$

Equality Constraints

Constrained Optimization



Constrained Optimization: method of Lagrange multipliers

- Solve a different, *unconstrained* optimization problem instead

- Compute stationary (critical) points of the Lagrangian:

$$f + \boxed{\lambda} c$$

Lagrange Multipliers!

At Optimal Solution, we have:

$$\mathbf{c}_i(\mathbf{x}) = 0$$

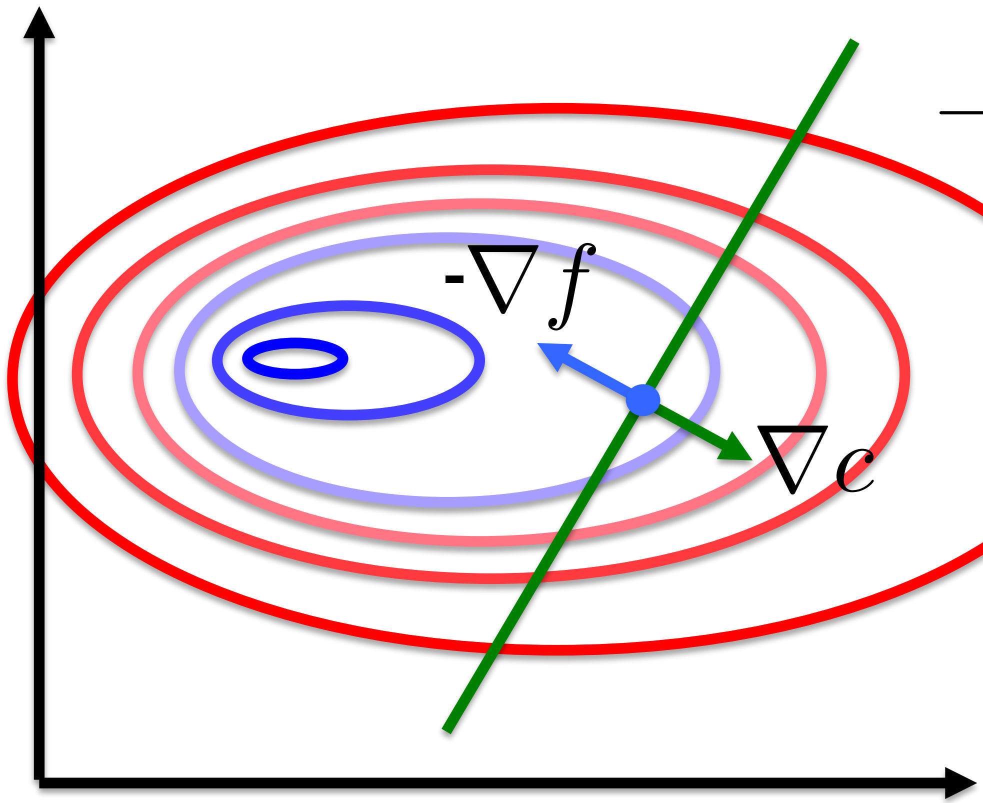
$$-\nabla f = \lambda \nabla c$$

Constrained Optimization

At Optimal Point

$$\mathbf{c}_i\left(\mathbf{x}\right) = 0$$

$$-\nabla f = \lambda \nabla c$$



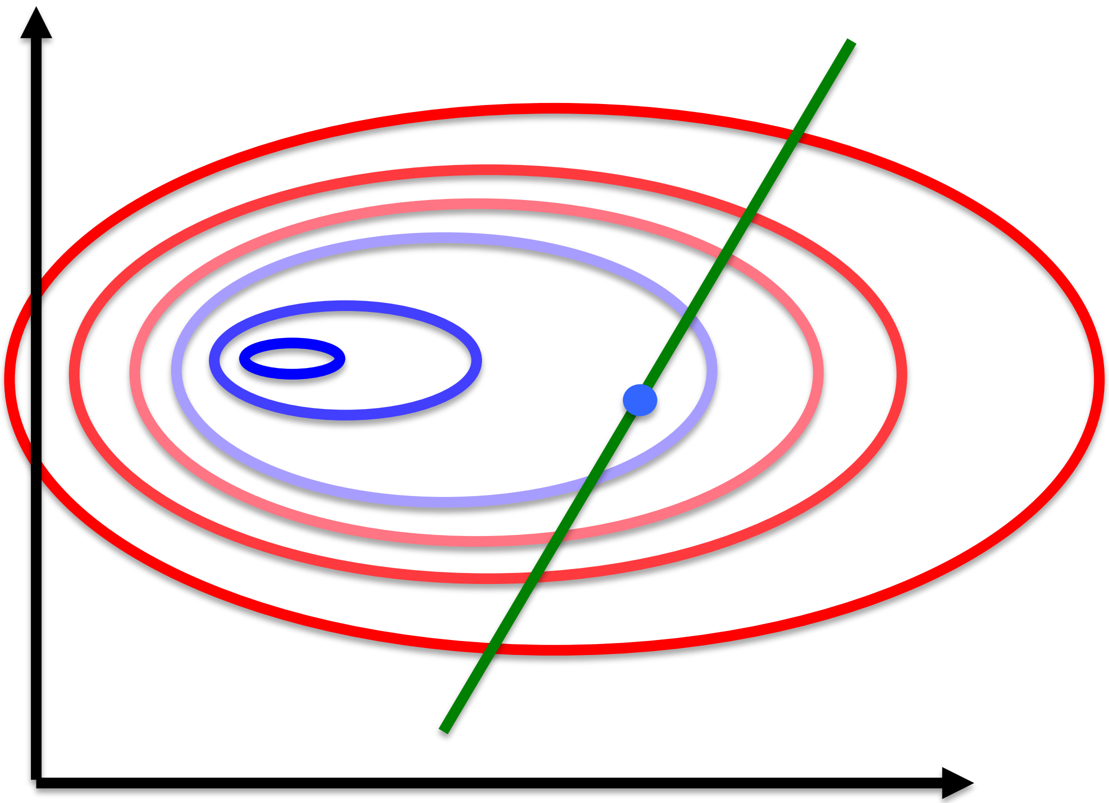
Constrained Optimization

- Very useful for general equality constrained problems
- Available in MATLAB as fmincon
- Easy to modify unconstrained Newton Code

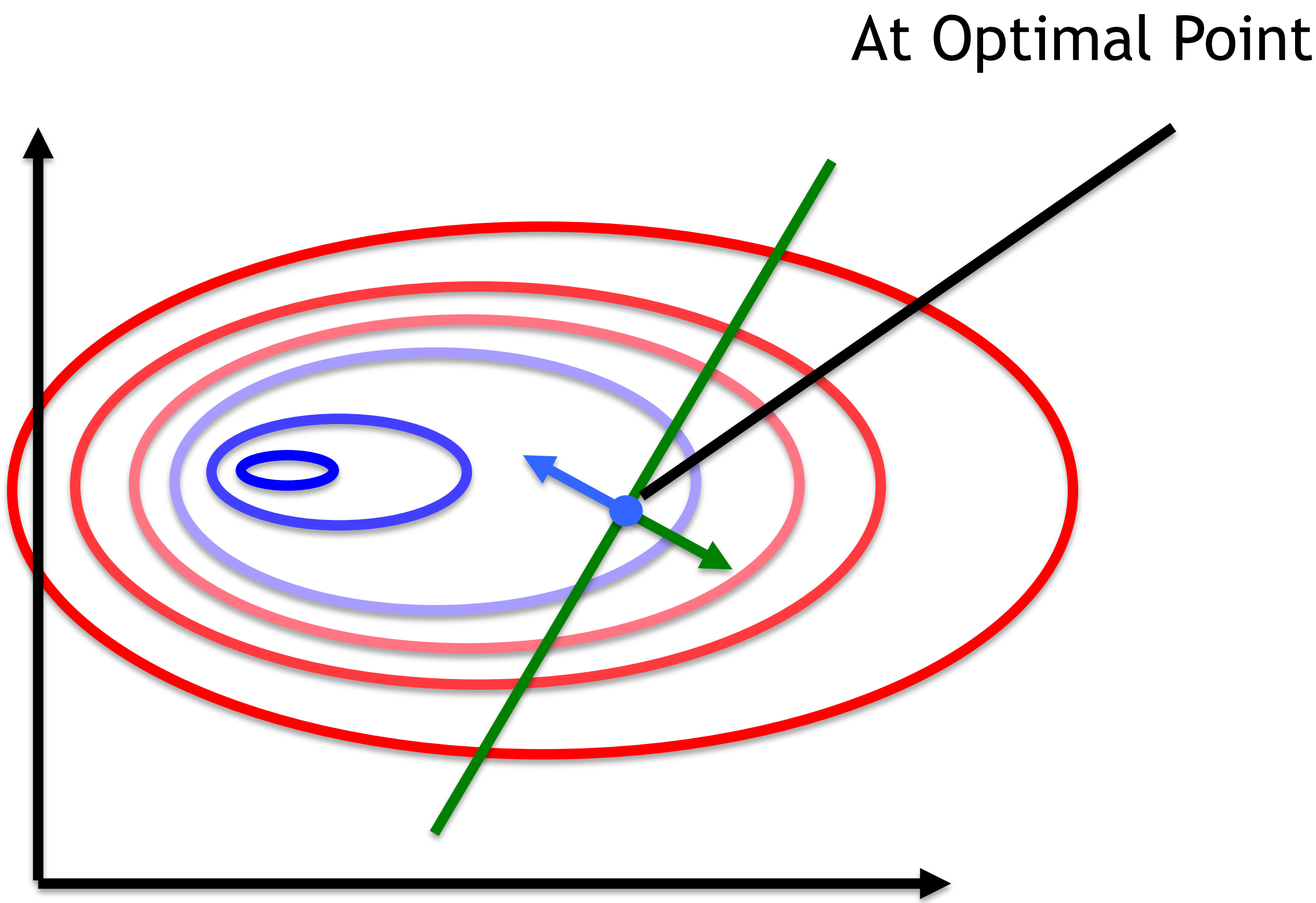
Next: Inequality Constrained Optimization

- Specifically we will work up to a particular type of problem called a Quadratic Program

Constrained Optimization



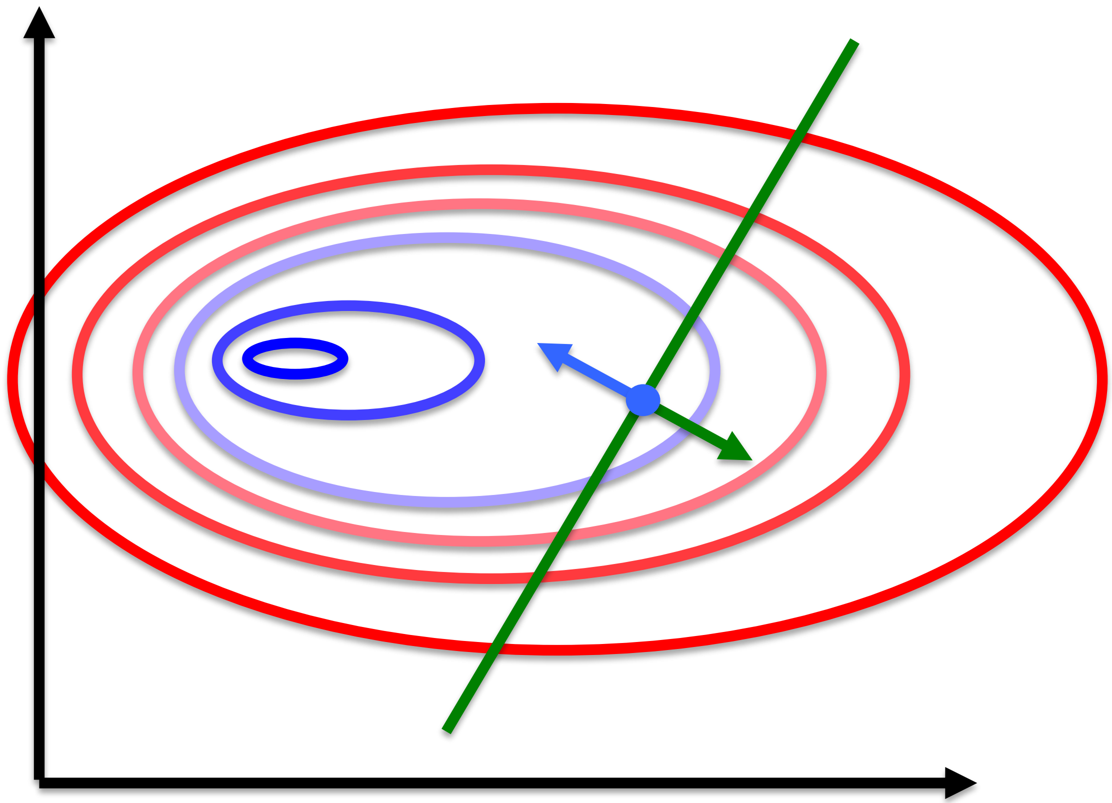
Constrained Optimization



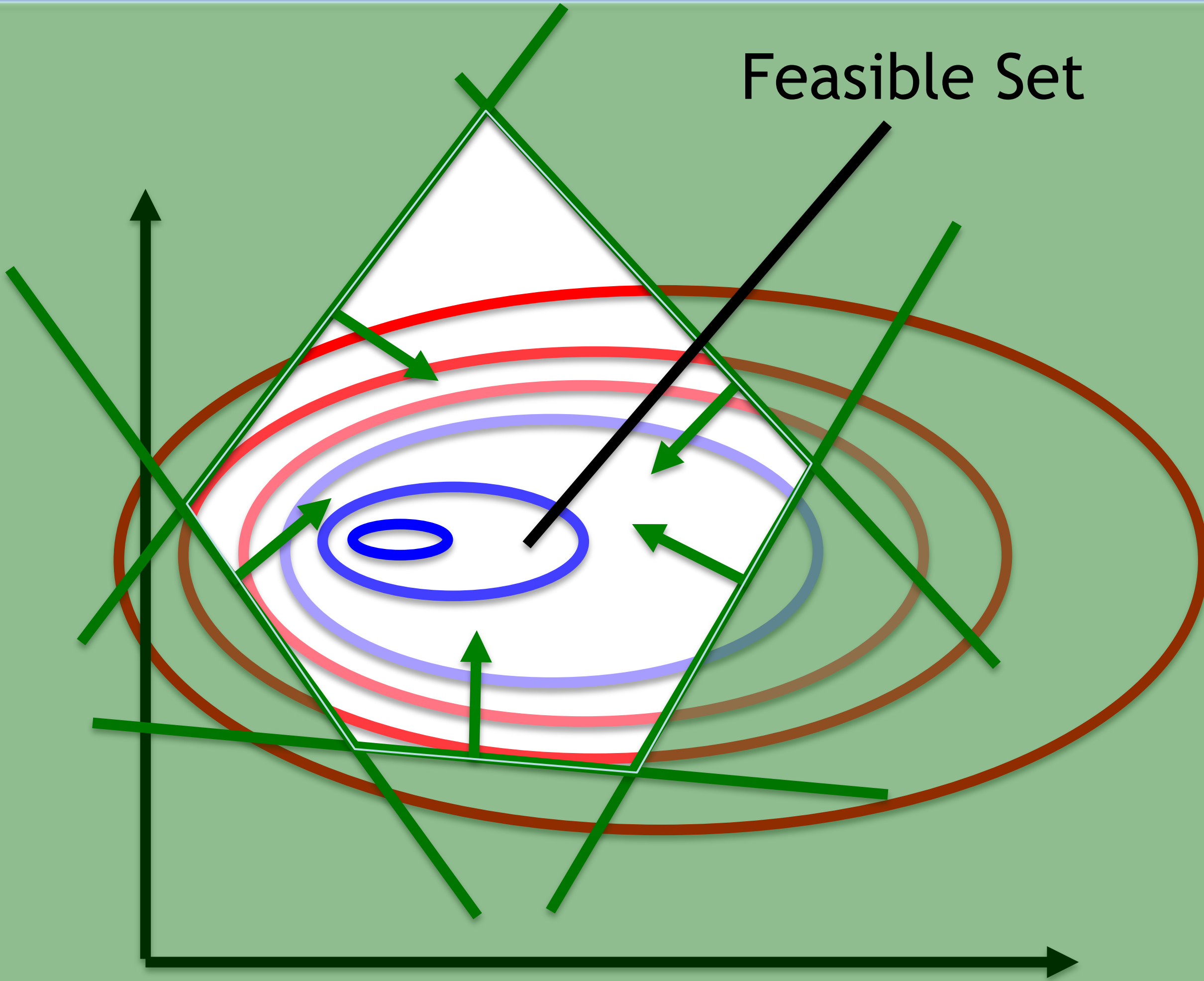
Inequality Constrained Optimization

$$\min f(x)$$
$$s.t\ c_i(\mathbf{x}) \boxed{\leq} 0$$

Equality Constrained Optimization



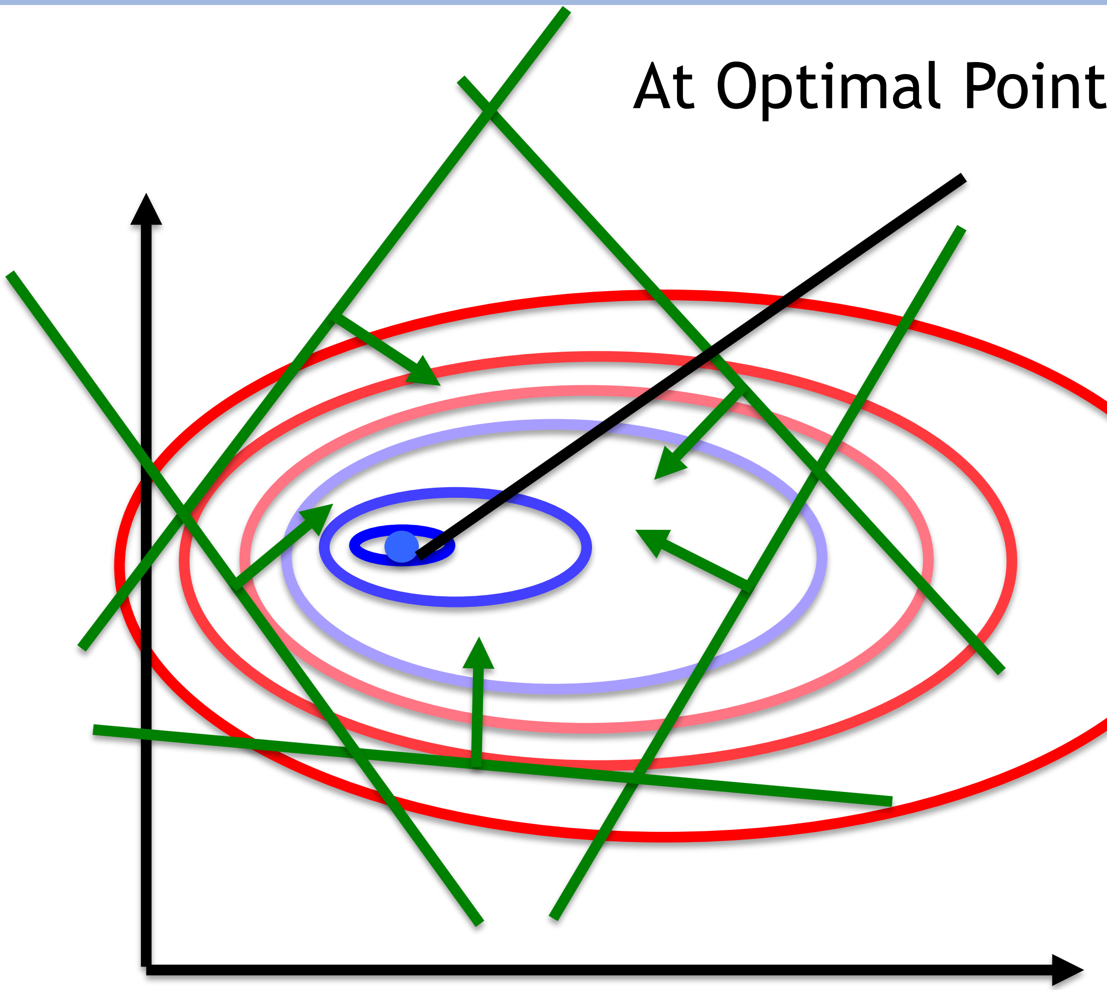
Inequality Constrained Optimization



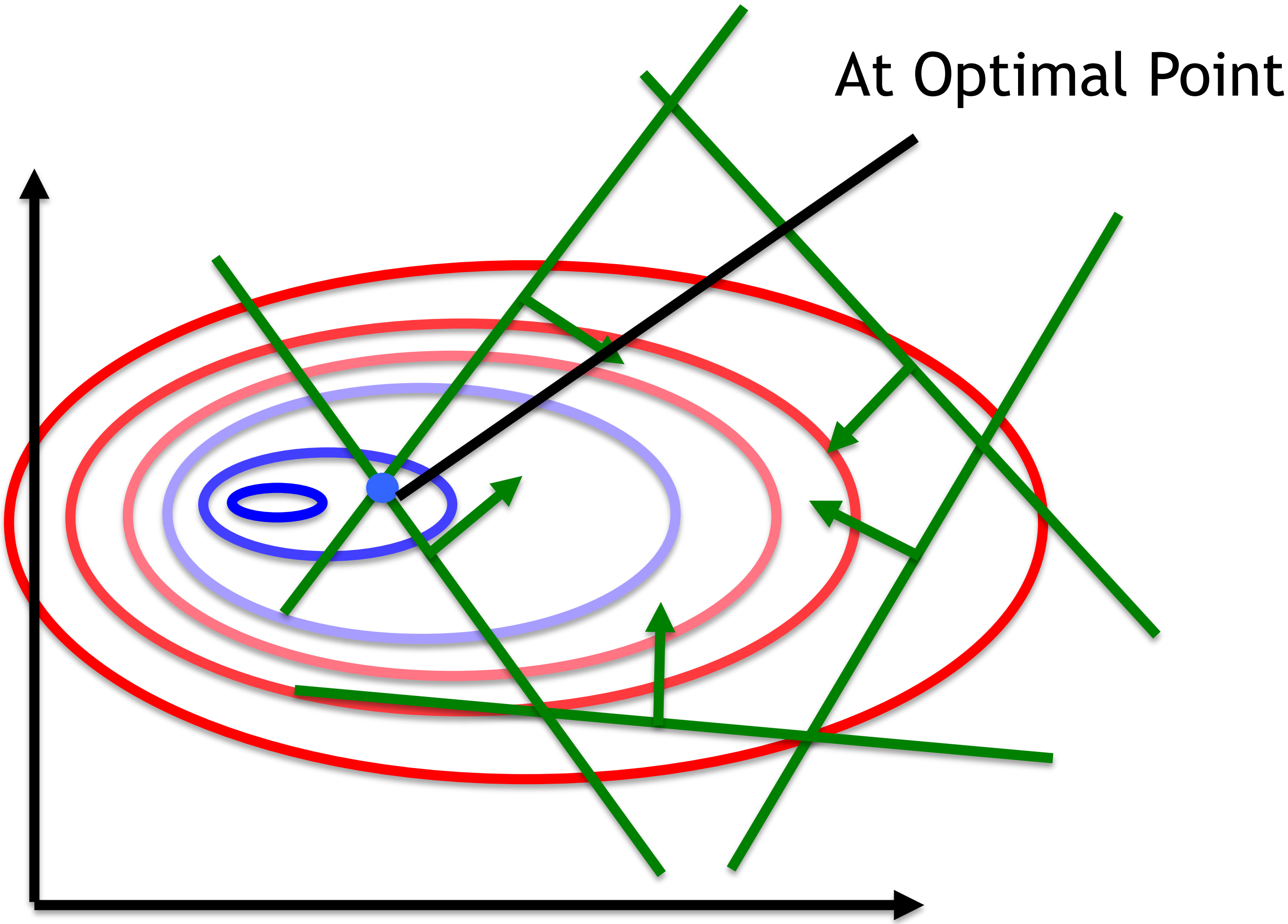
The Active Set

- Hidden inside of each inequality constrained optimization is an equality constrained optimization
- There are two cases for our optimal point…

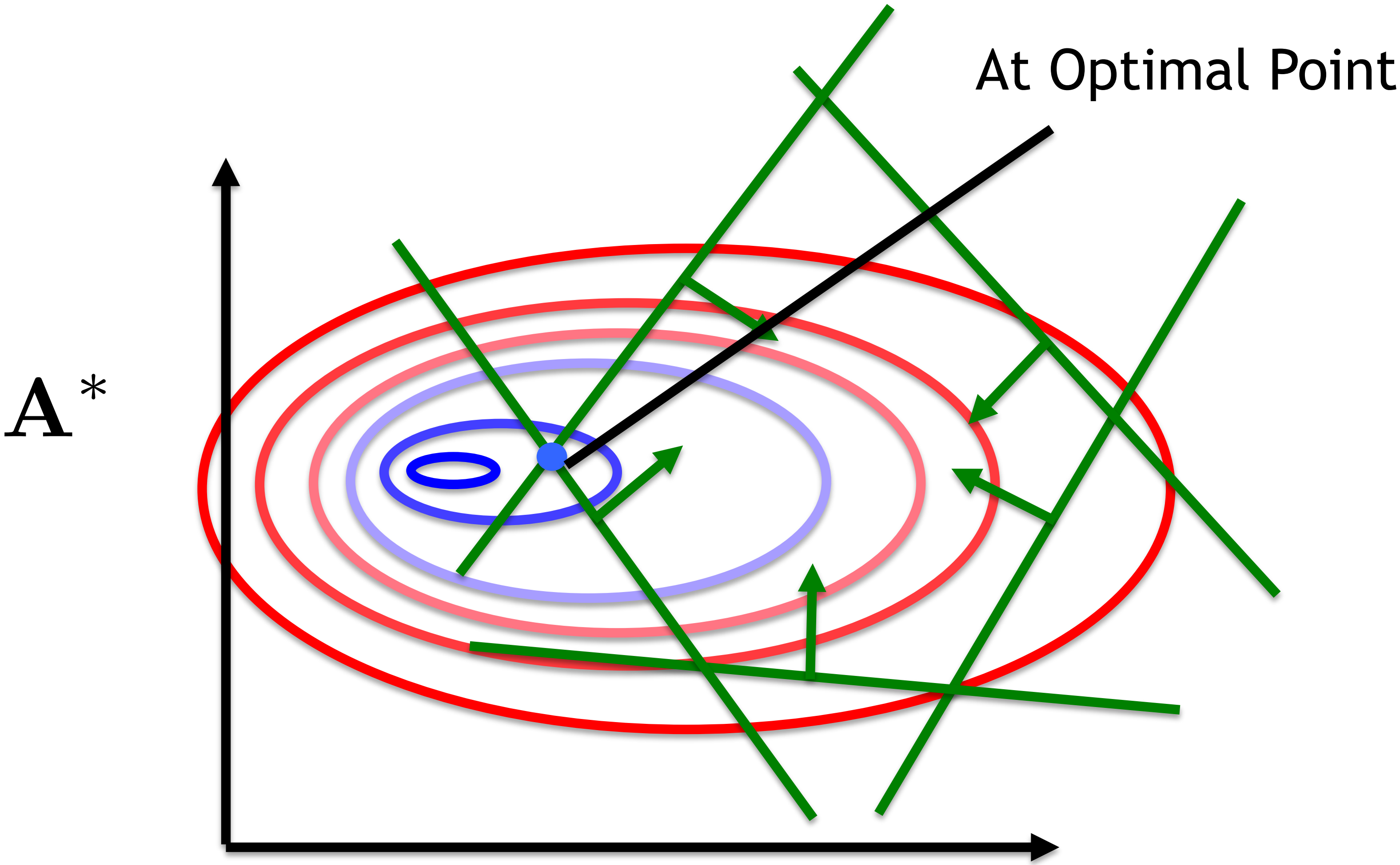
Case 1: Optimal Value Inside Feasible Set



Case 2: Optimal Value On Boundary



Case 2: Optimal Value On Boundary



The Active Set

- On the boundary we satisfy

$$\min f(x)$$

$$s.t\ \boxed{\mathbf{A}}^{*}\mathbf{x} = \mathbf{b}$$

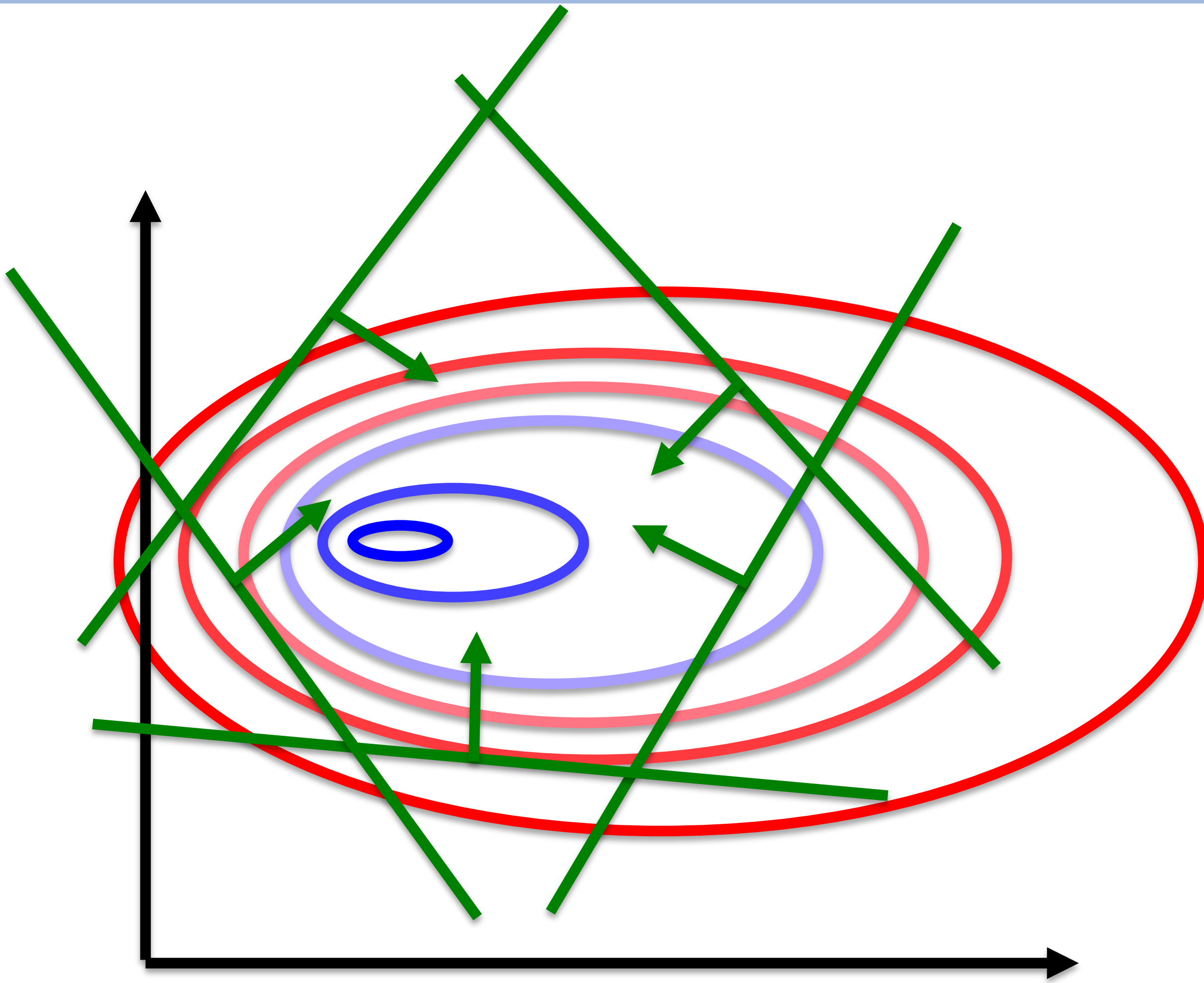
Active Set

Quadratic Programs

- Quadratic objective, linear constraints

$$\min \mathbf{x}^T \mathbf{H} \mathbf{x} + \mathbf{x}^T \mathbf{d}$$
$$s.t.\ \mathbf{A}\mathbf{x} = \mathbf{b}$$
$$s.t.\ \mathbf{L}\mathbf{x} \leq \mathbf{m}$$

Quadratic Programs



Quadratic Program

- How do we solve this ?
- Active Set: Try different combinations of constraints until the minimum is found
- Interior Point: …

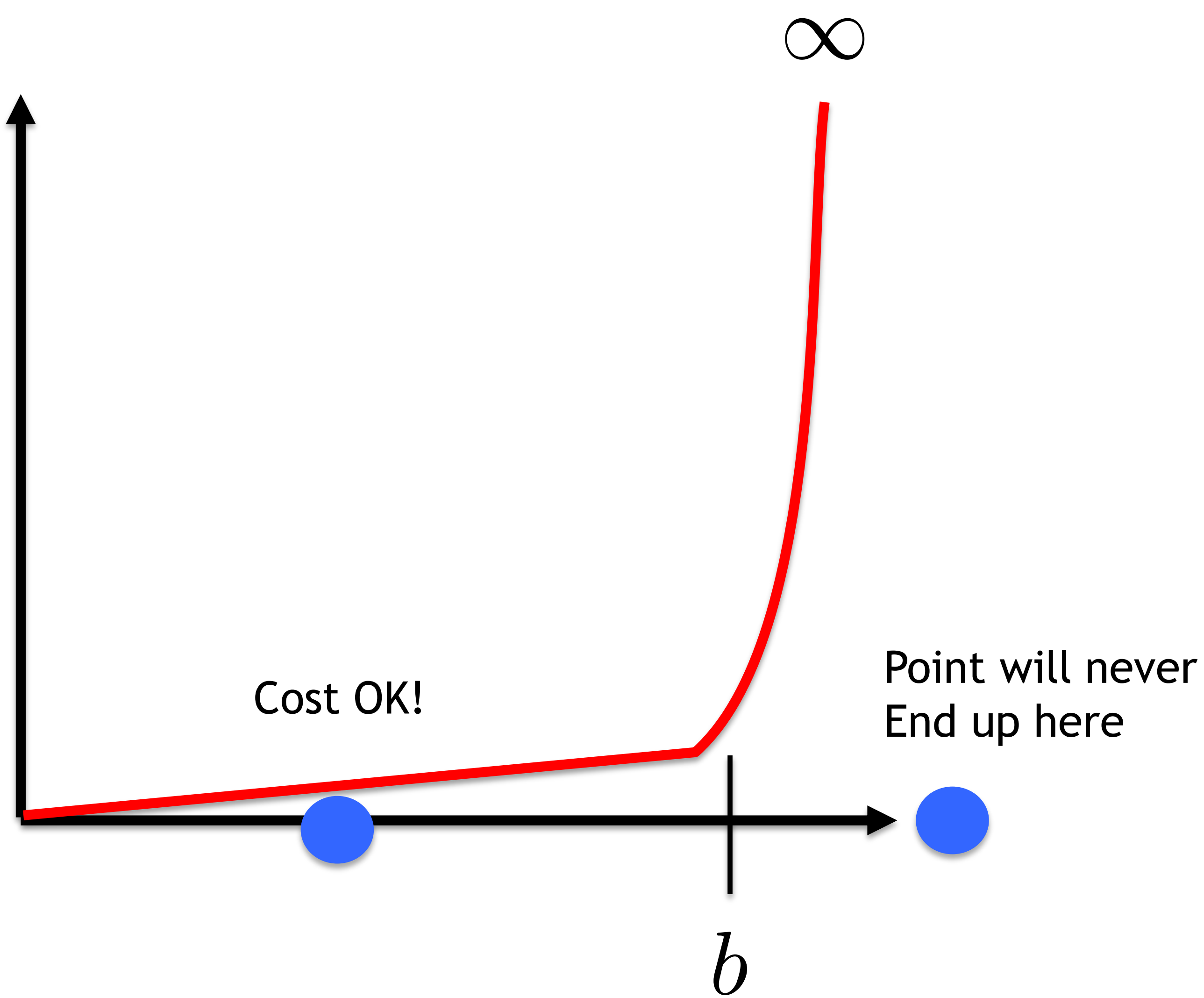
Interior Point Methods

- Replace inequality constraints with special barrier functions

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + (\mathbf{A}\mathbf{x} - \mathbf{b})^T \lambda + \sum_i \boxed{c_i}(\mathbf{x})$$

Special "Constraint" Function

Interior Point: barrier functions



Interior Point Methods

- Replace inequality constraints with special barrier functions

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + (\mathbf{A}\mathbf{x} - \mathbf{b})^T \lambda + \sum_i \boxed{c_i}(\mathbf{x})$$

Special "Constraint" Function

- Now use Newton's method

Quadratic Programs and Interior Point

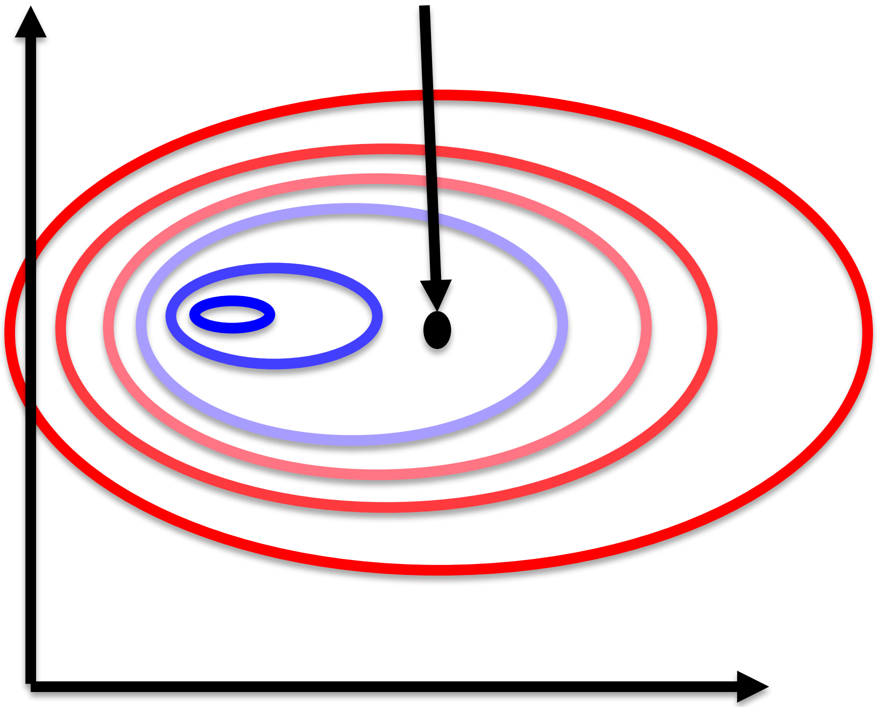
- Quadratic Programs (Active Set)
  - Quadprog++ (http://quadprog.sourceforge.net)
  - MATLAB: quadprog
- Interior Point
  - Ipopt (https://projects.coin-or.org/Ipopt)

Types of Optimization

- Continuous vs. Discrete
- Constrained vs. Unconstrained

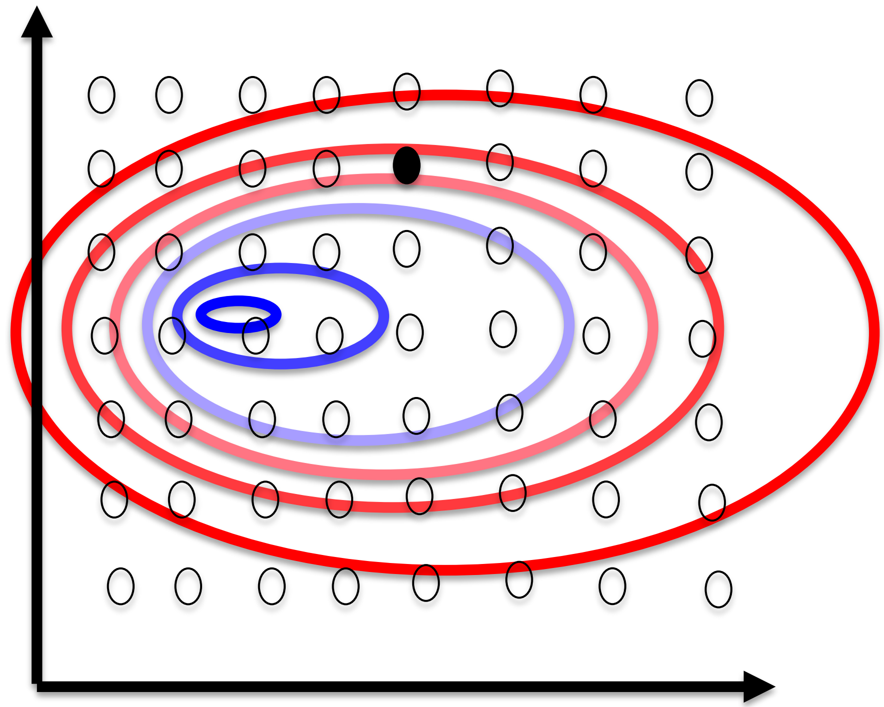
Continuous

This point can move smoothly



Discrete

Choose from discrete points in parameter space



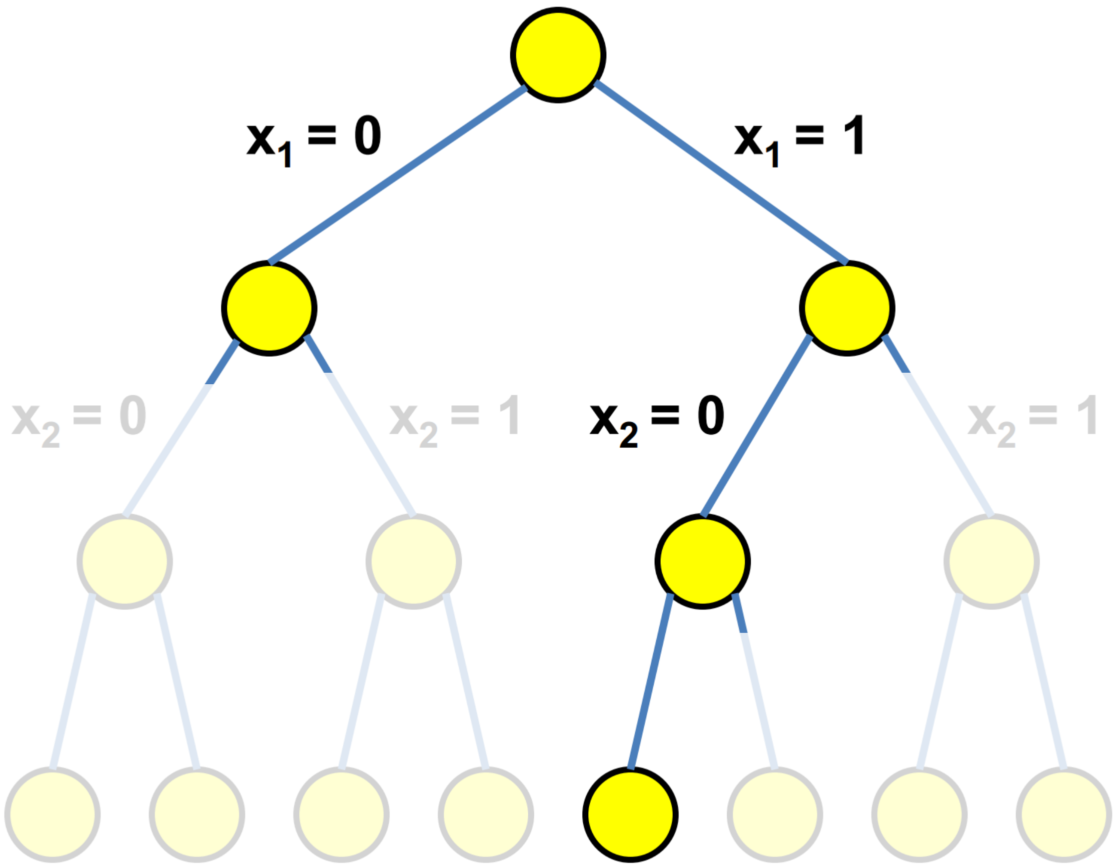
Branch and Bound Optimizations

- most commonly used tool for solving NP-hard optimization problems
- An optimization technique with 3 phases
  - Branch (divide the solution space into a number of subspaces)
  - Bound (compute some upper and lower bound for the cost of each subspace – worst case, conservative but feasible solution, best case, optimistic, usually through relaxation)
  - Prune (remove subspaces with upper bounds worse than the lower bounds of other subspaces)

Branch and Bound Example

$$\begin{array}{ll} \text{maximize} & 15x_1 + 12x_2 + 4x_3 + 2x_4 \\ \text{subject to} & 8x_1 + 5x_2 + 3x_3 + 2x_4 \leq 10 \\ & x_k \text{ binary for } k = 1 \text{ to } 4 \end{array}$$

Branch and Bound Example



Enumerating all solutions is not feasible – exponential explosion

Simulated Annealing

- Has four ingredients
  - Cost function
  - Configuration (made of discrete or continuous elements)
  - Neighbor Generator
  - Annealing Schedule

Simulated Annealing

- Basic Idea taken from cooling of materials in metallurgy
- At high “heat” atoms undergo rigorous motion
- As they are cooled they move less
- Explore trade-off between exploration and exploitation

Simulated Annealing

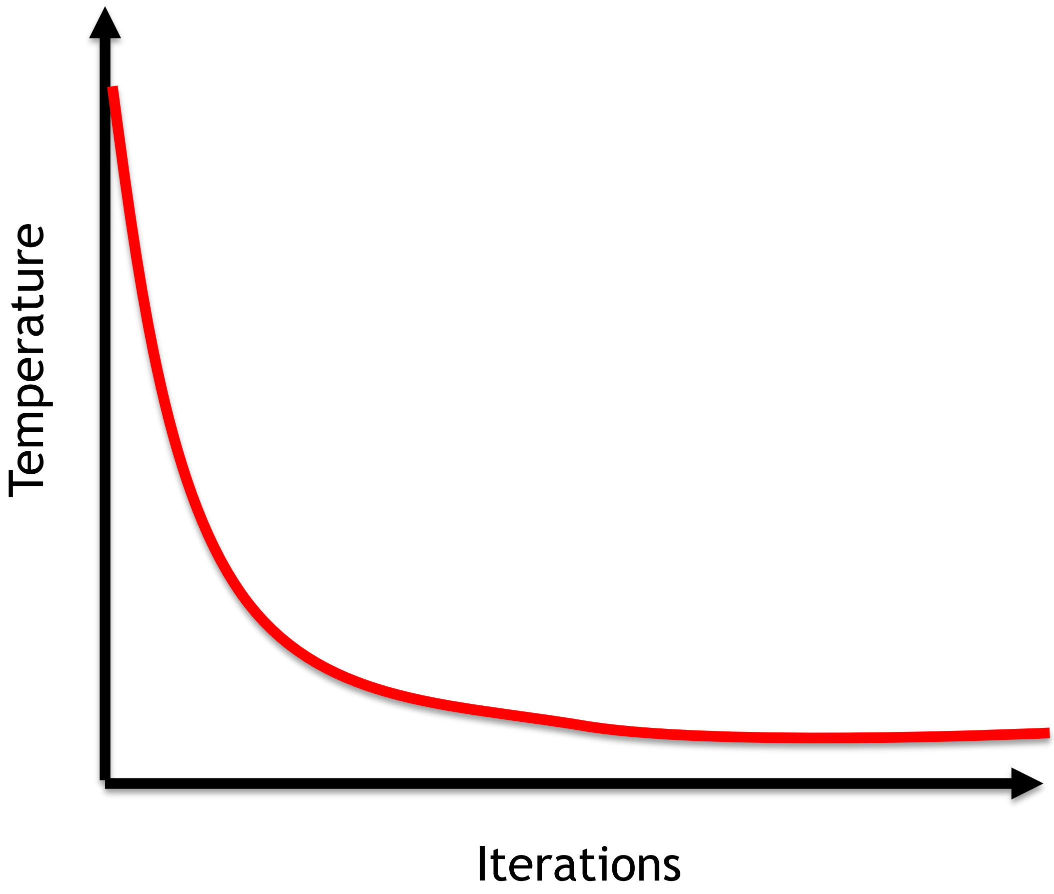
- Cost function: $f(\mathbf{q})$

Simulated Annealing

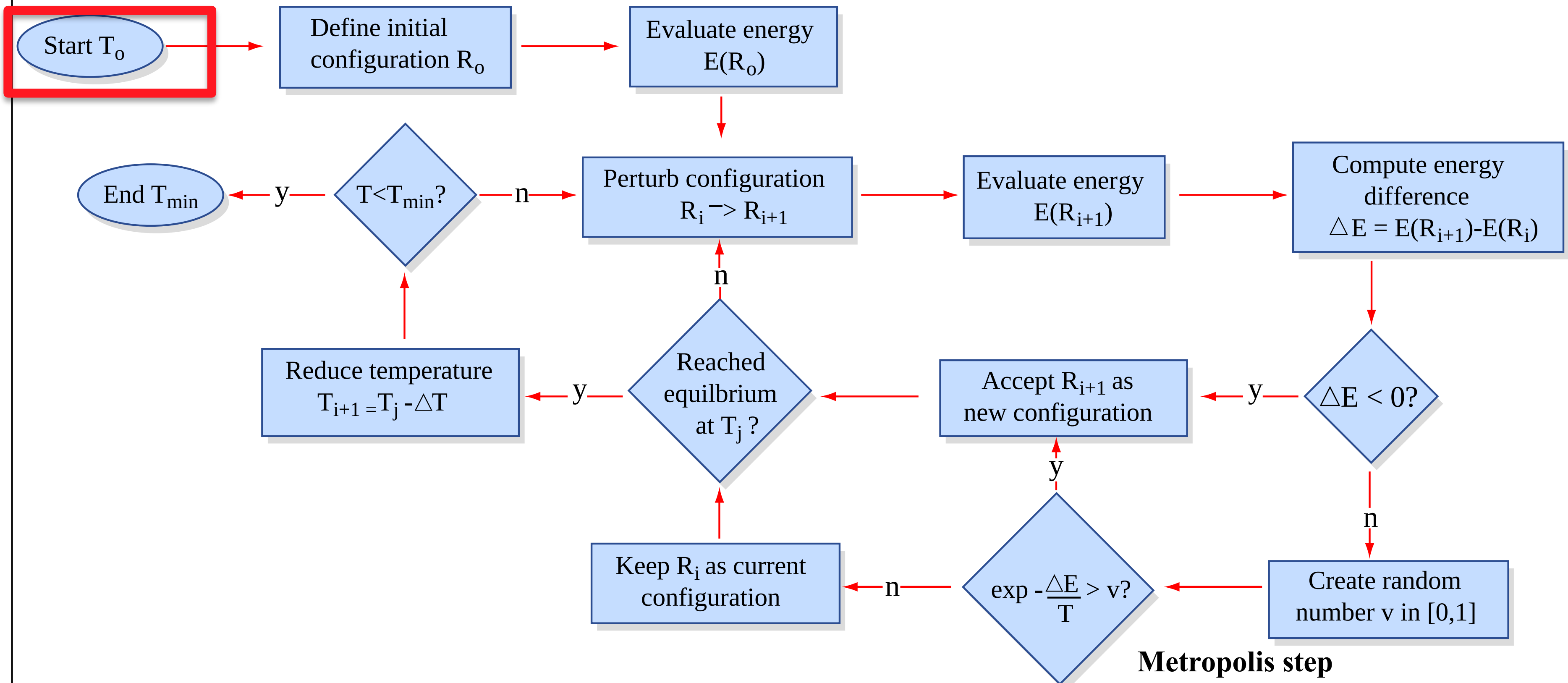
- Cost function: $f(\mathbf{q})$
- Neighbor Generator: Rearrange Configuration
  - Change **q** to something nearby
- Annealing Schedule

Simulated Annealing

- Annealing Schedule



Simulated Annealing



SA Block Diagram

Image by MIT OpenCourseWare.

Simulated Annealing

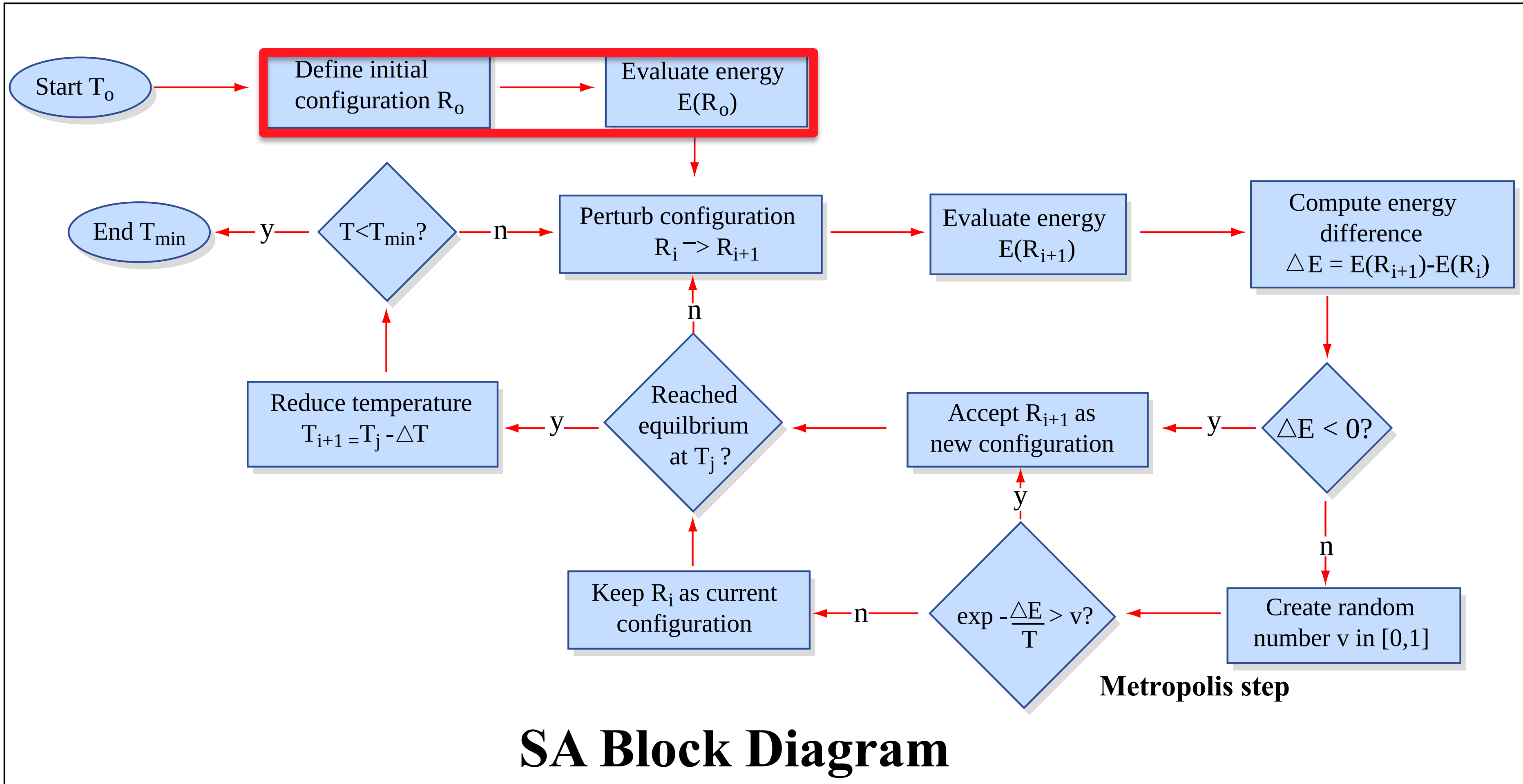


Image by MIT OpenCourseWare.

Simulated Annealing

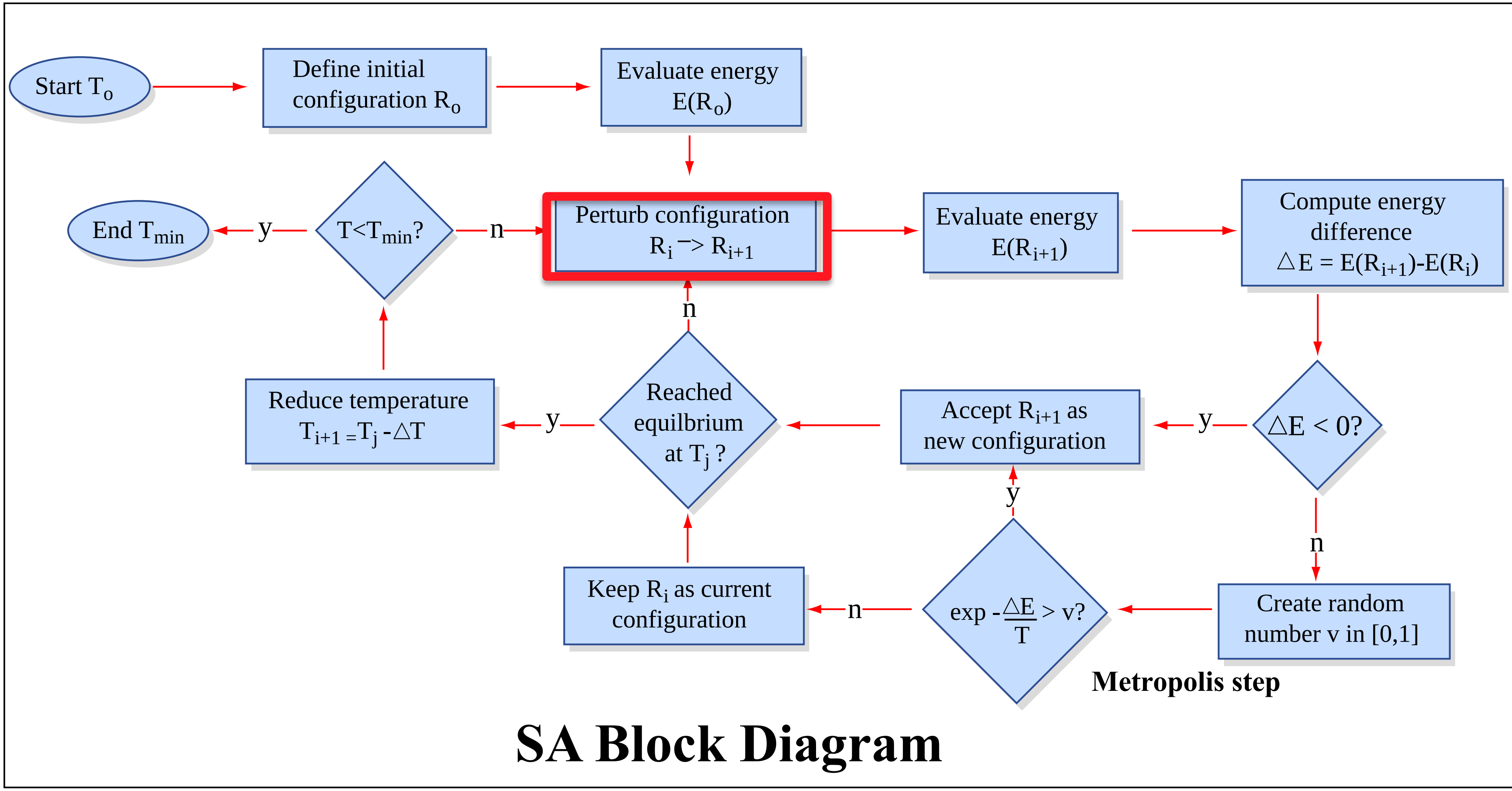


Image by MIT OpenCourseWare.

Simulated Annealing

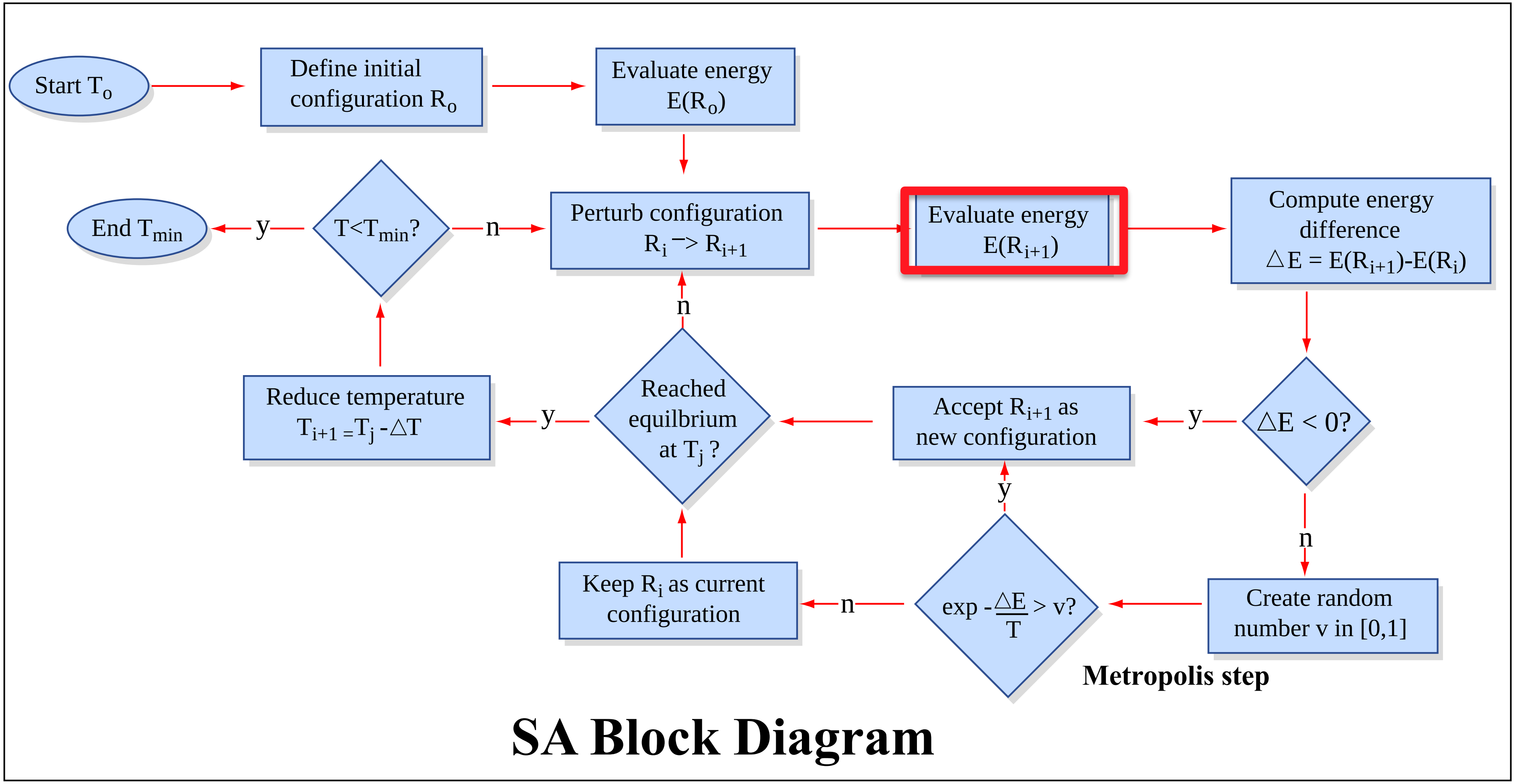


Image by MIT OpenCourseWare.

Simulated Annealing

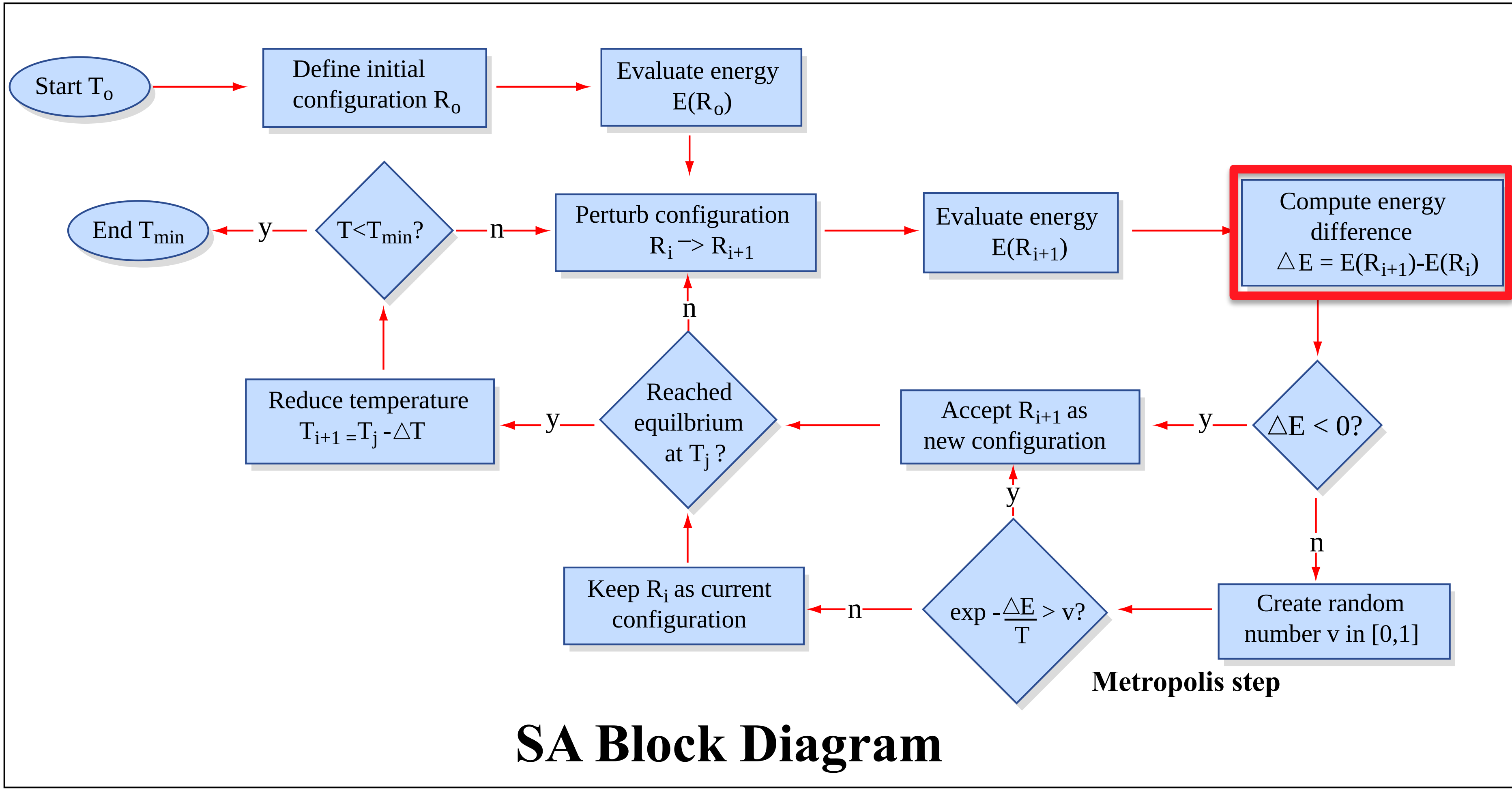
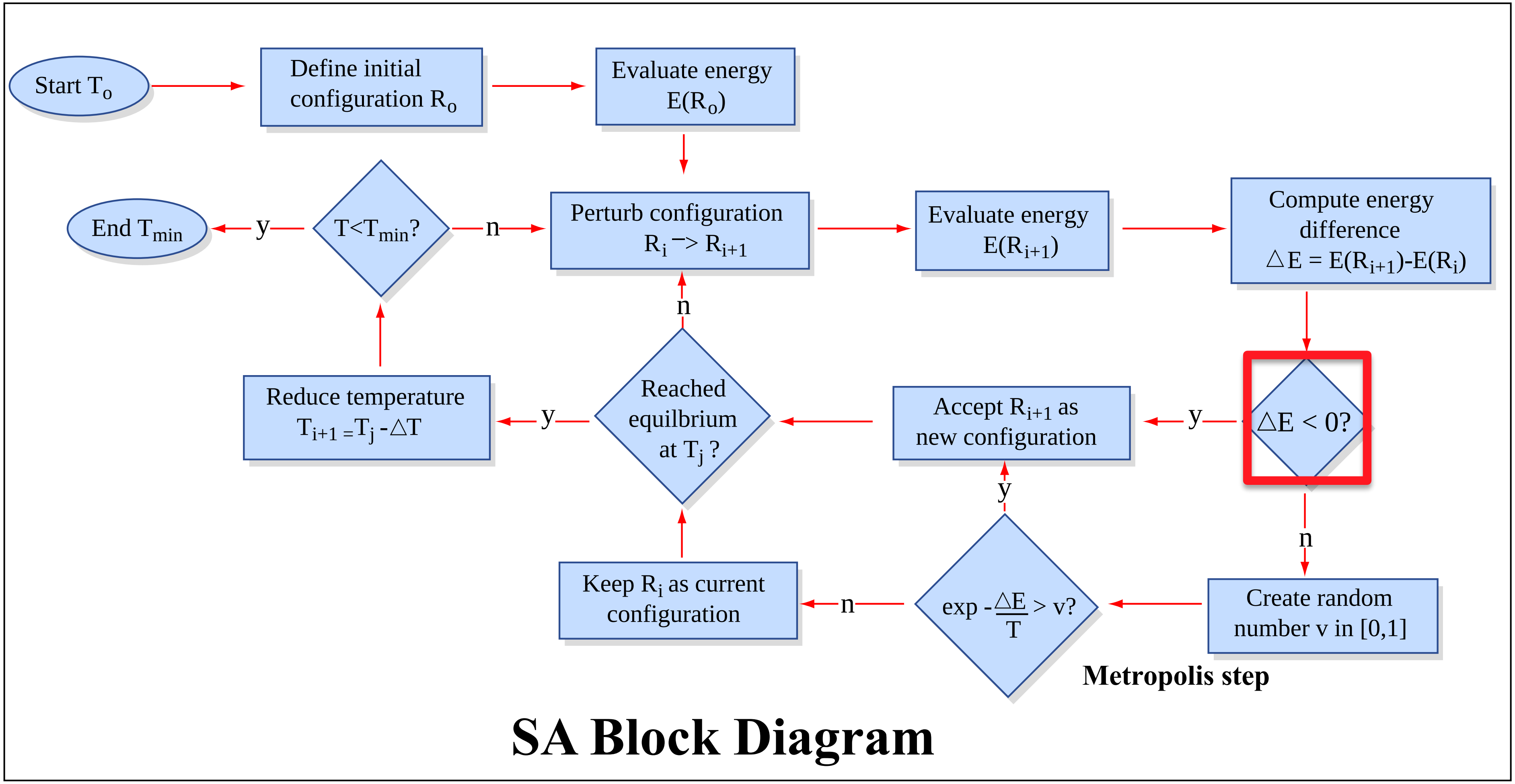


Image by MIT OpenCourseWare.

Simulated Annealing



Simulated Annealing

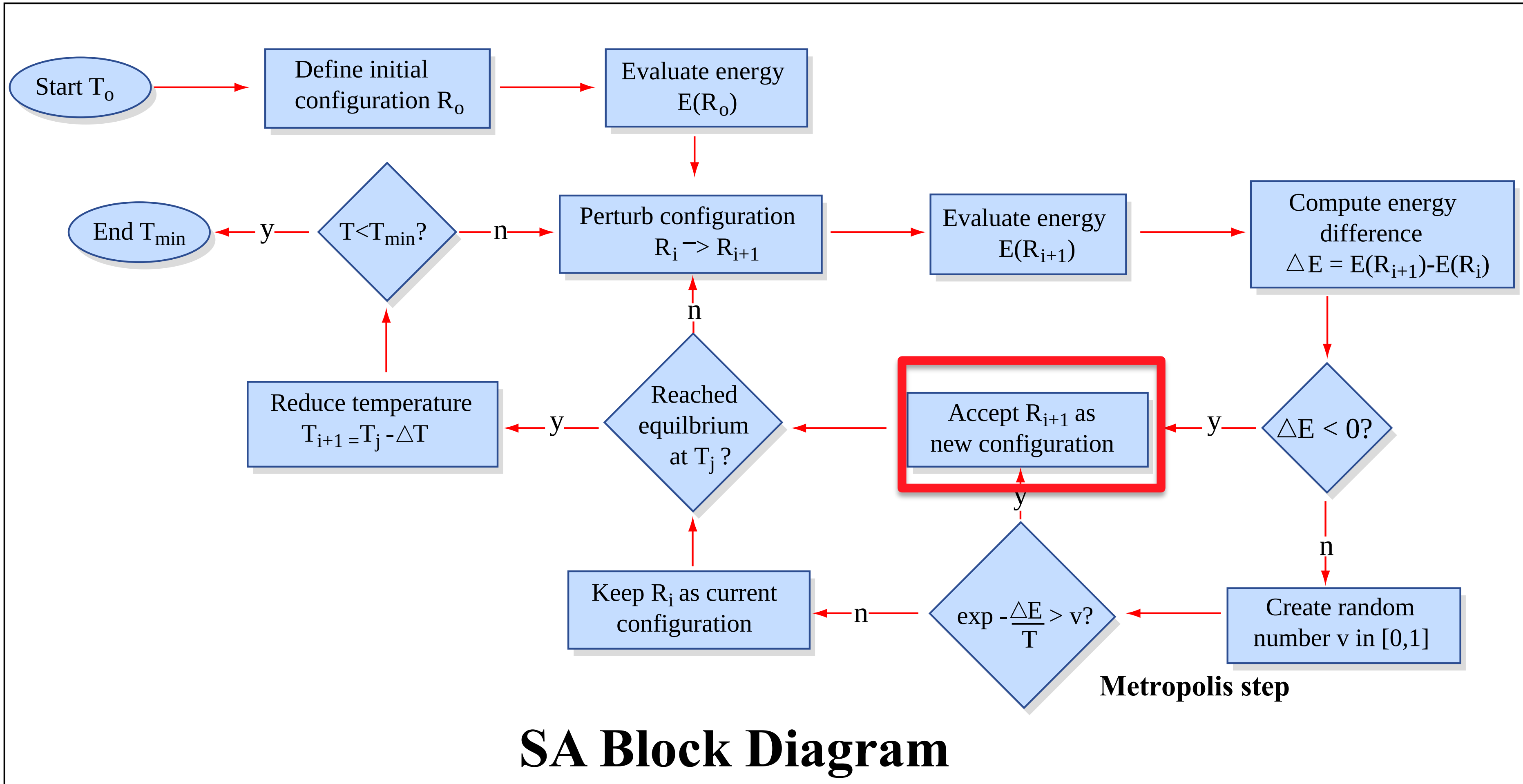


Image by MIT OpenCourseWare.

Simulated Annealing

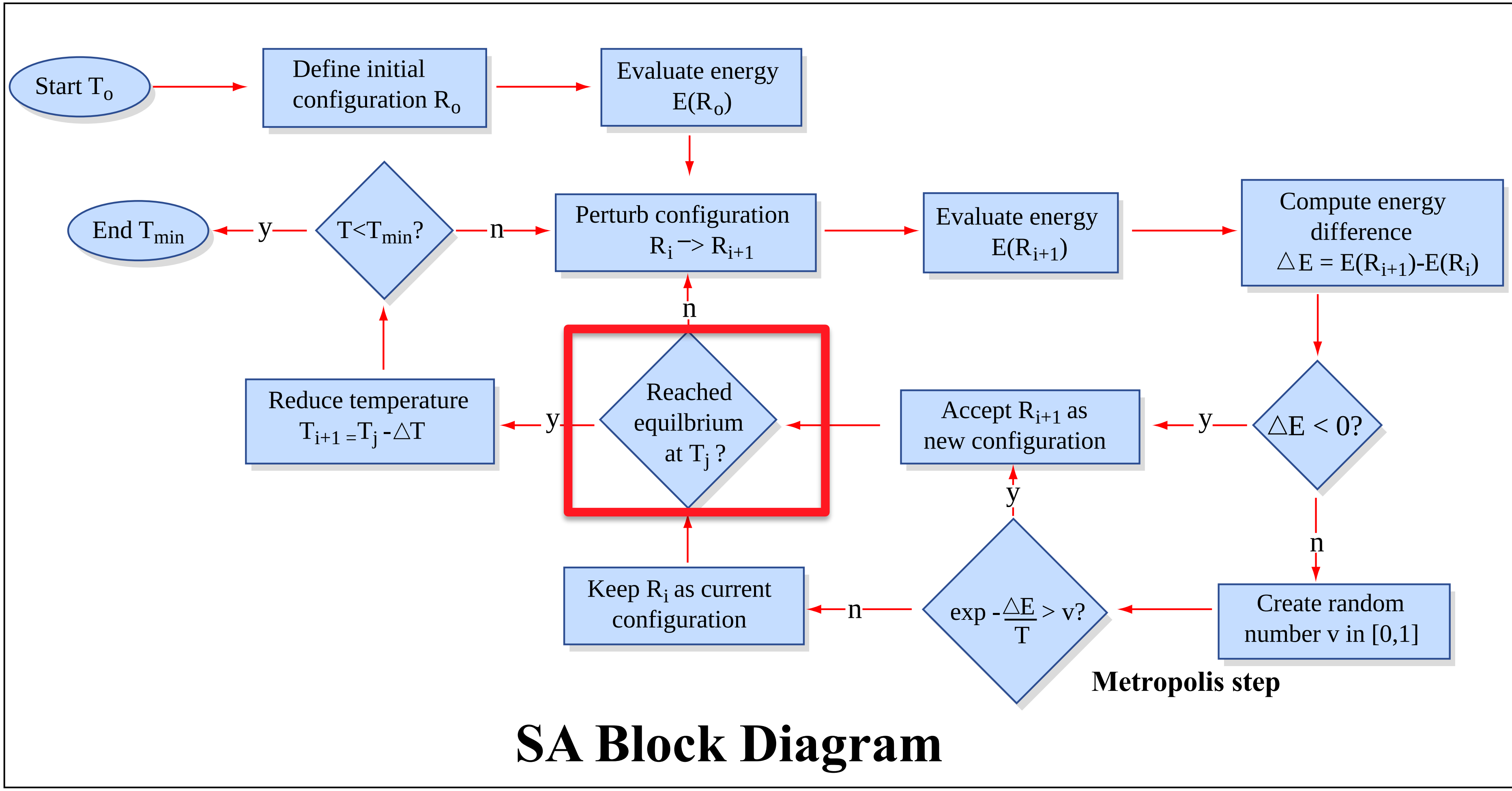


Image by MIT OpenCourseWare.

Simulated Annealing

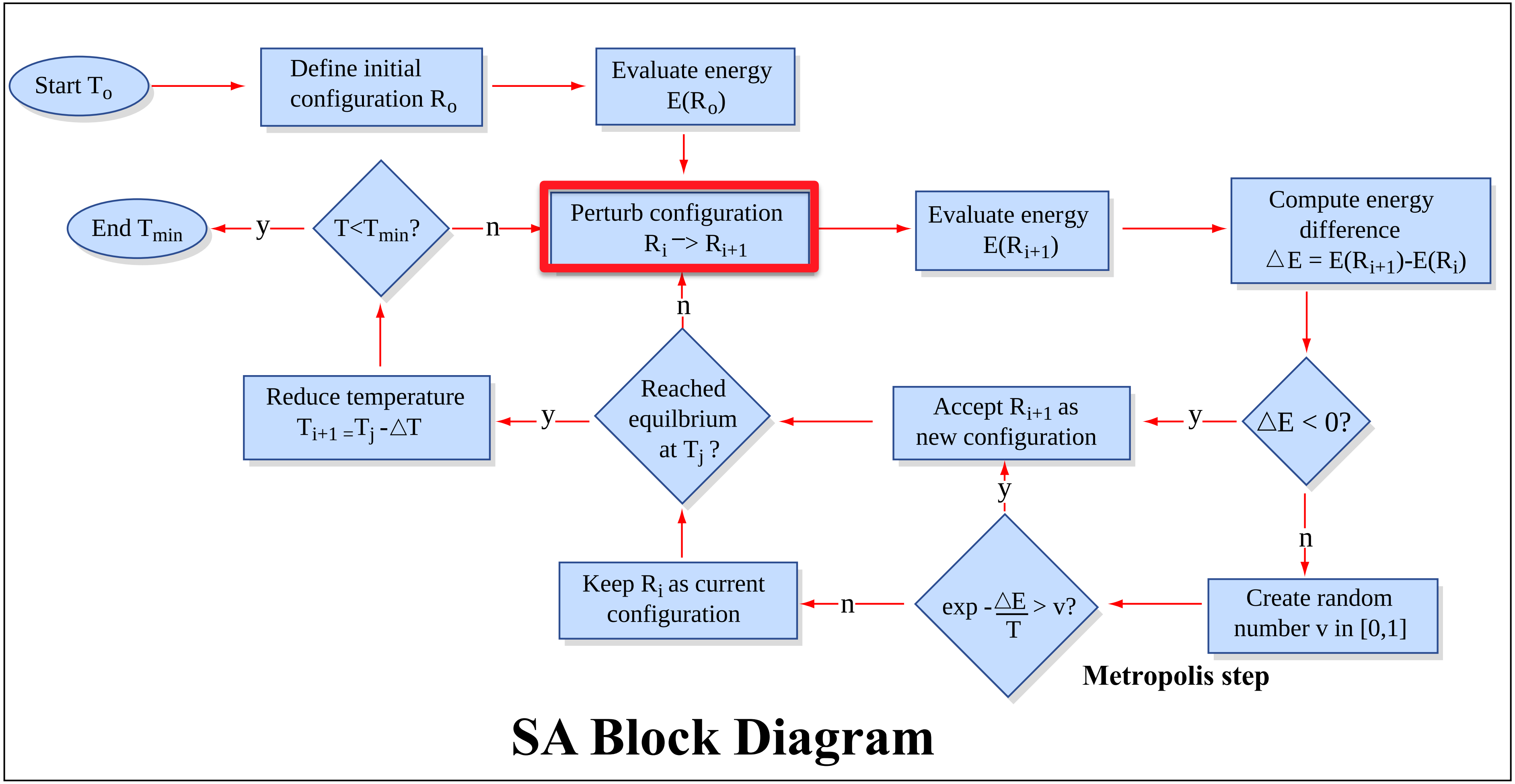
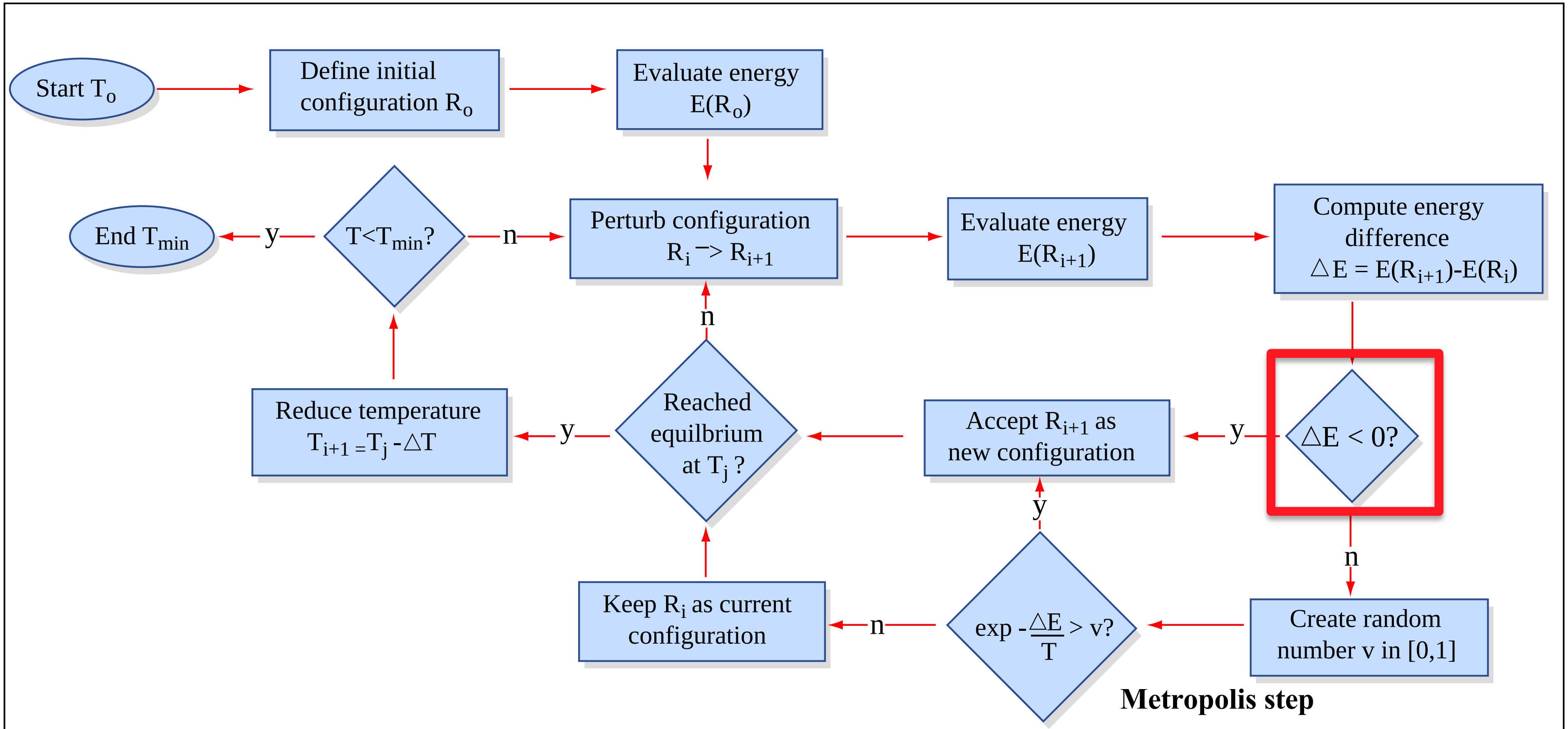


Image by MIT OpenCourseWare.

Simulated Annealing



SA Block Diagram

Image by MIT OpenCourseWare.

Simulated Annealing

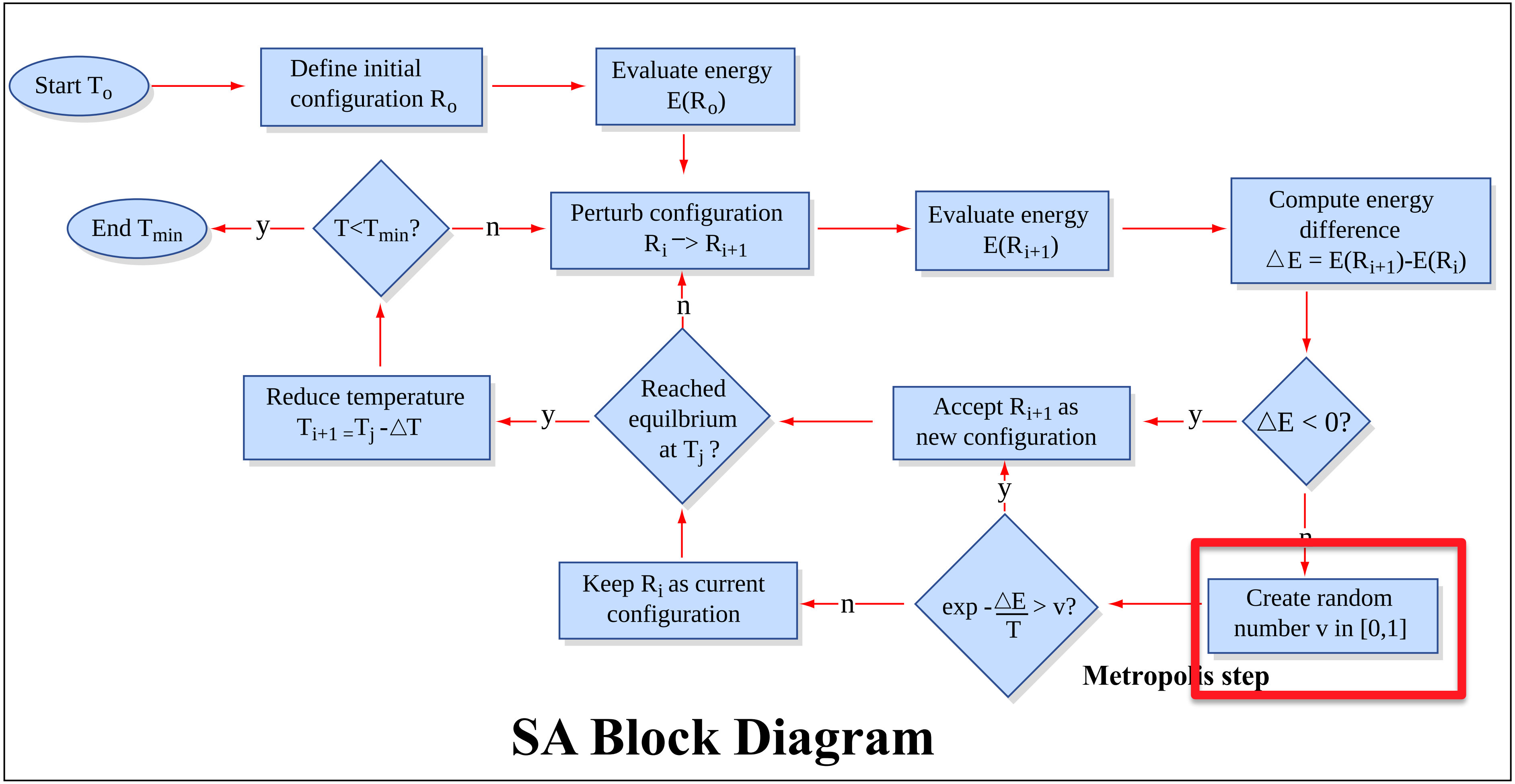
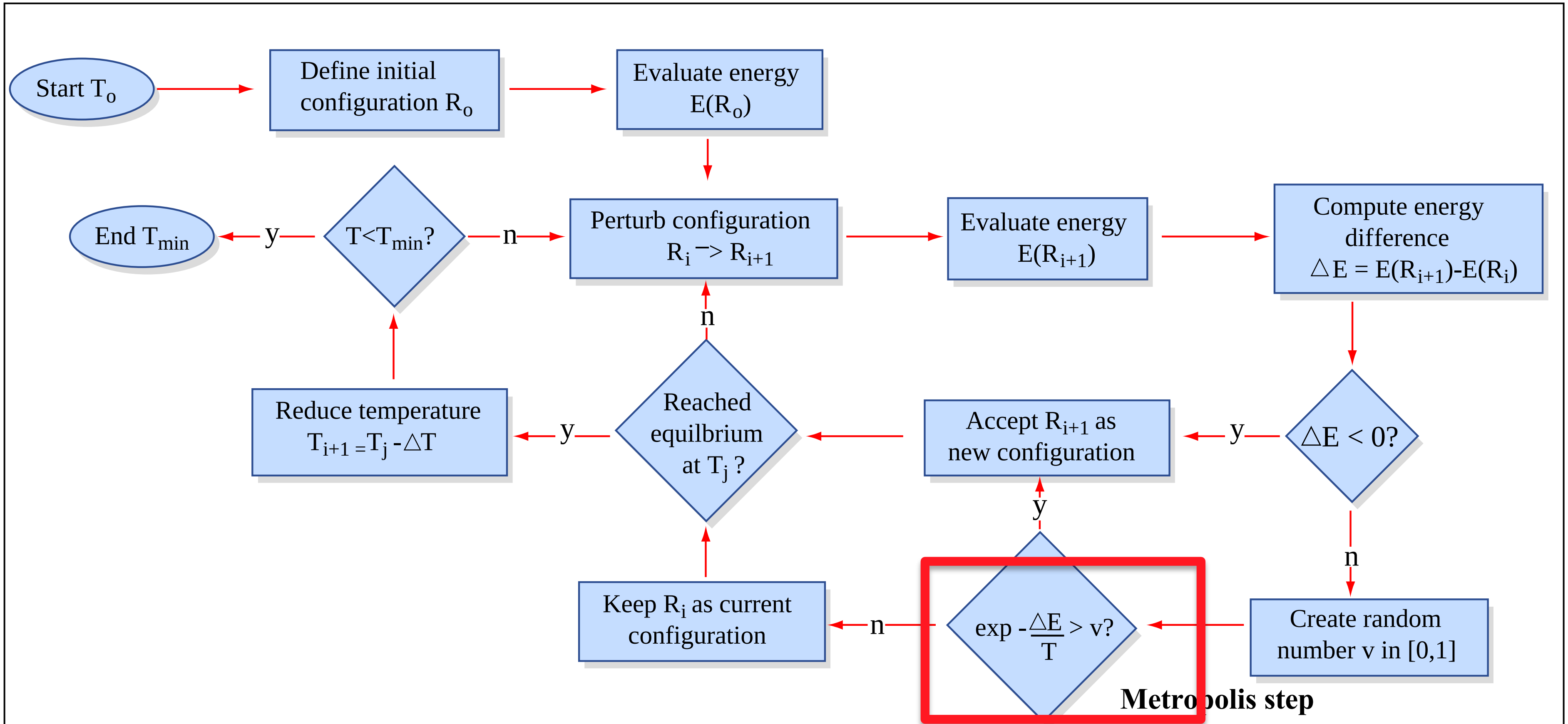


Image by MIT OpenCourseWare.

Simulated Annealing



SA Block Diagram

Image by MIT OpenCourseWare.

Simulated Annealing

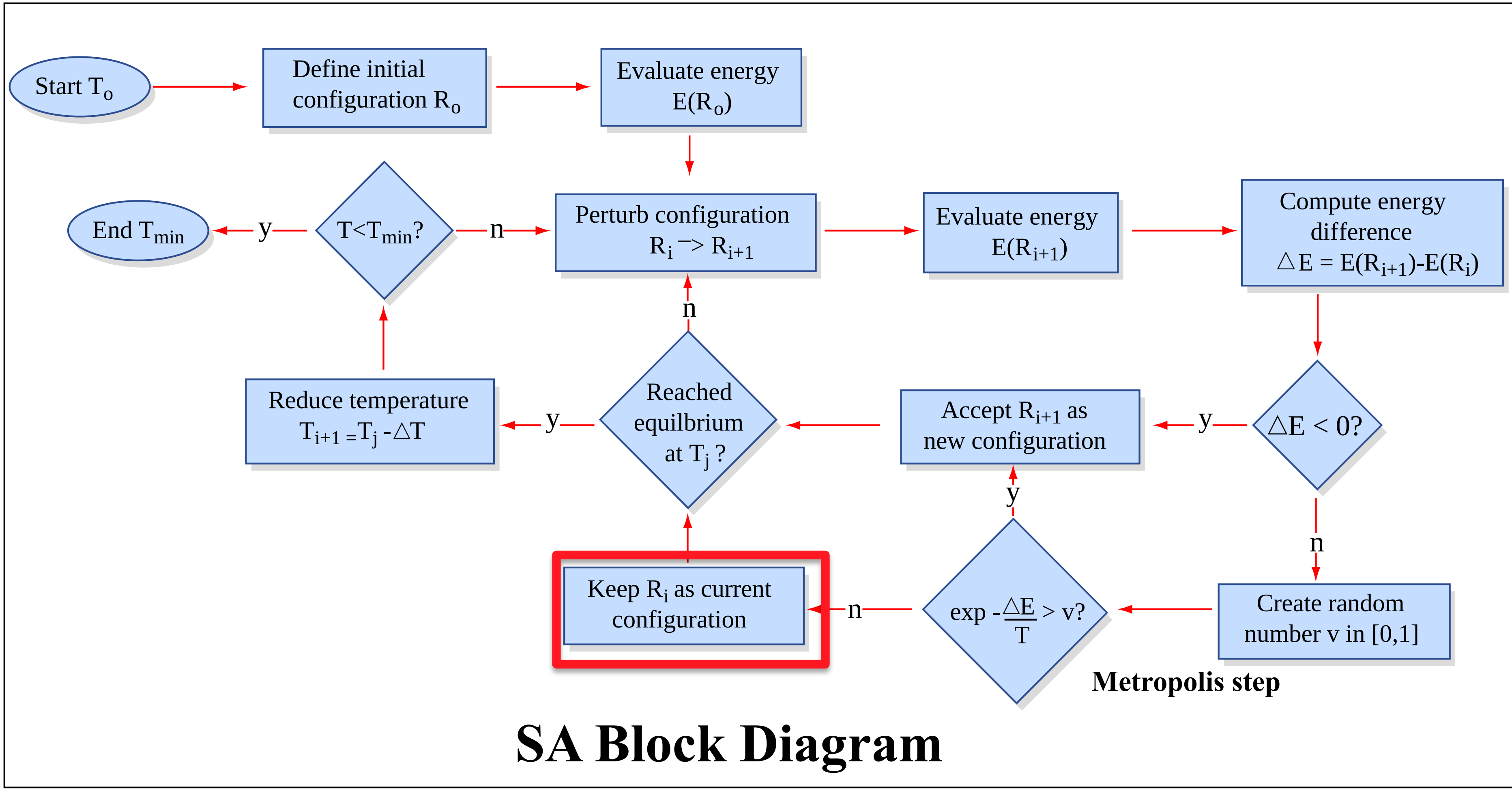
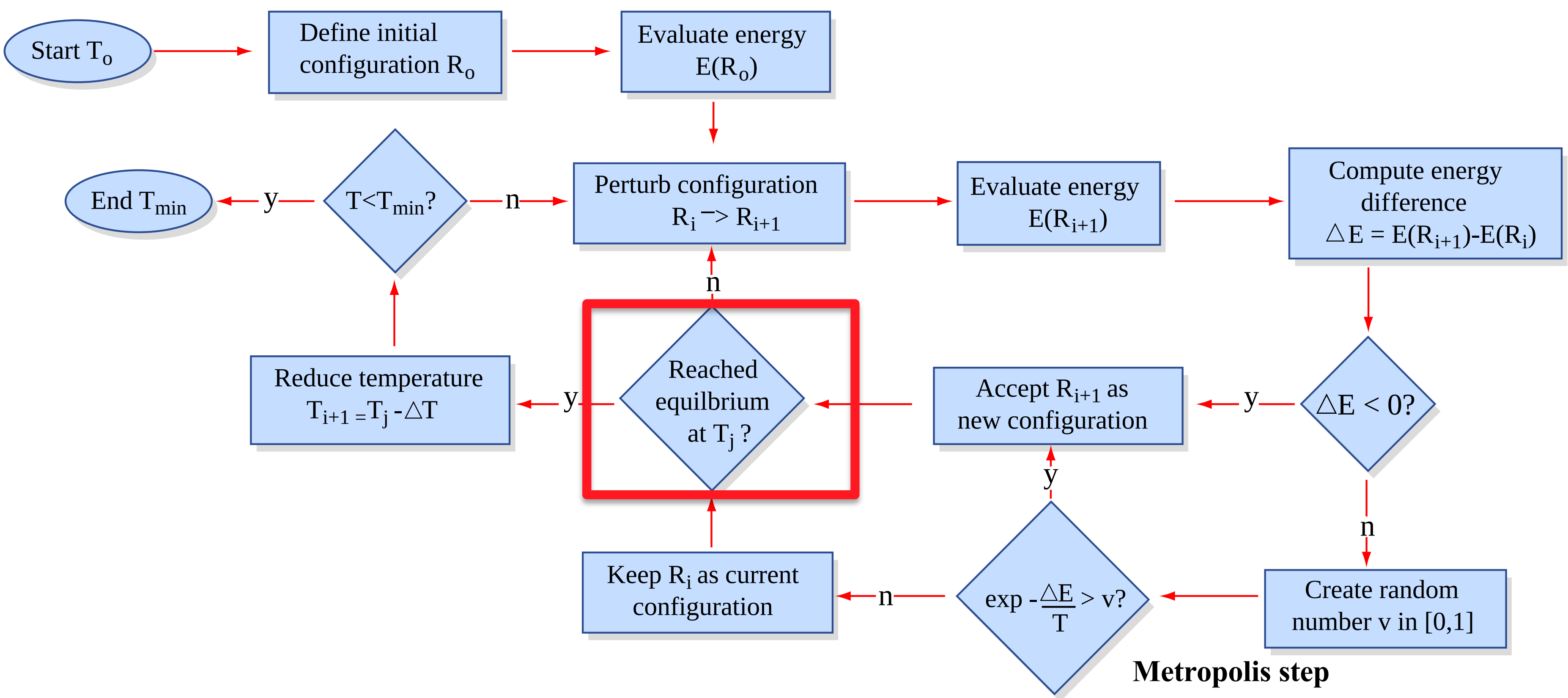


Image by MIT OpenCourseWare.

Simulated Annealing



SA Block Diagram

Image by MIT OpenCourseWare.

Simulated Annealing

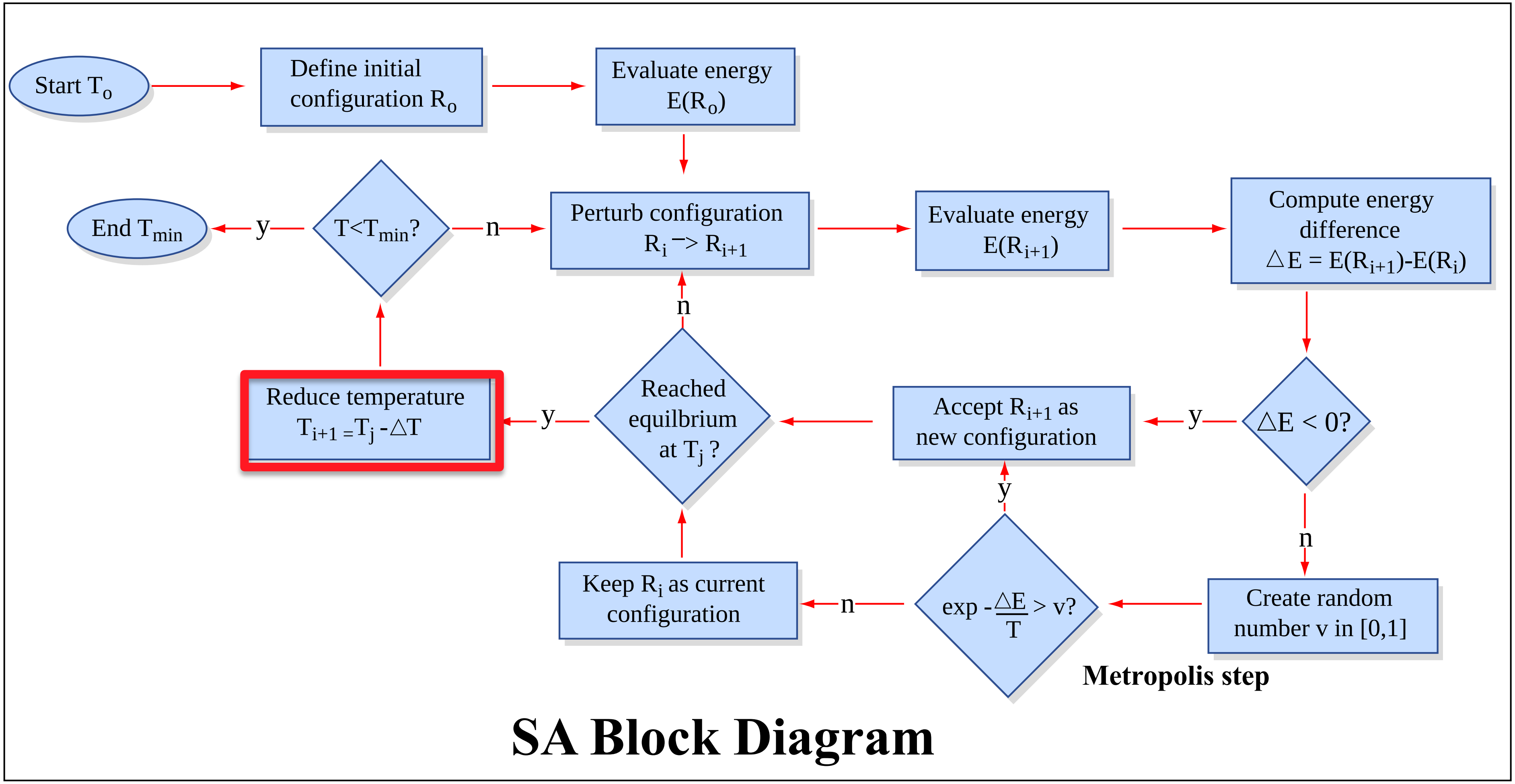


Image by MIT OpenCourseWare.

Simulated Annealing

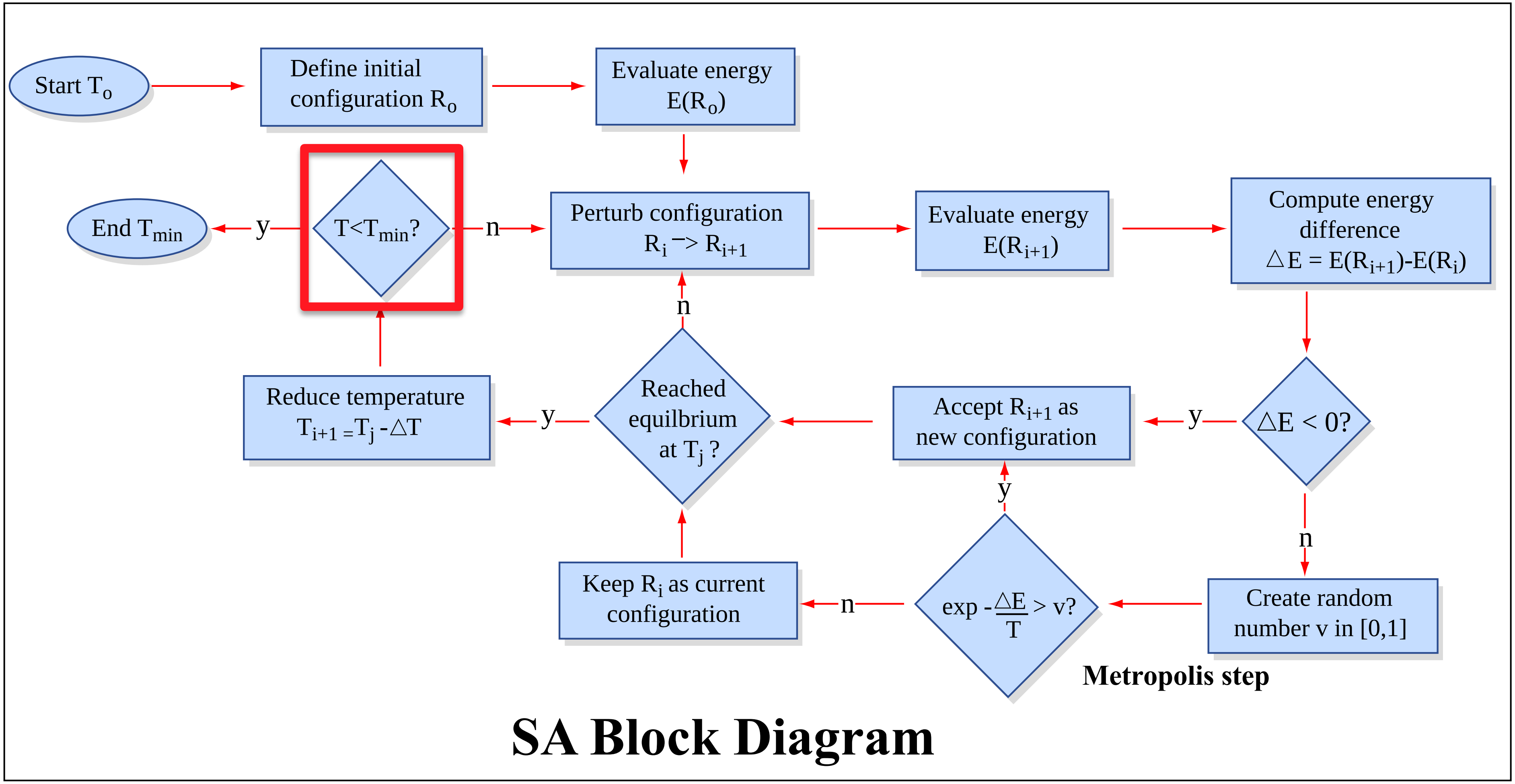
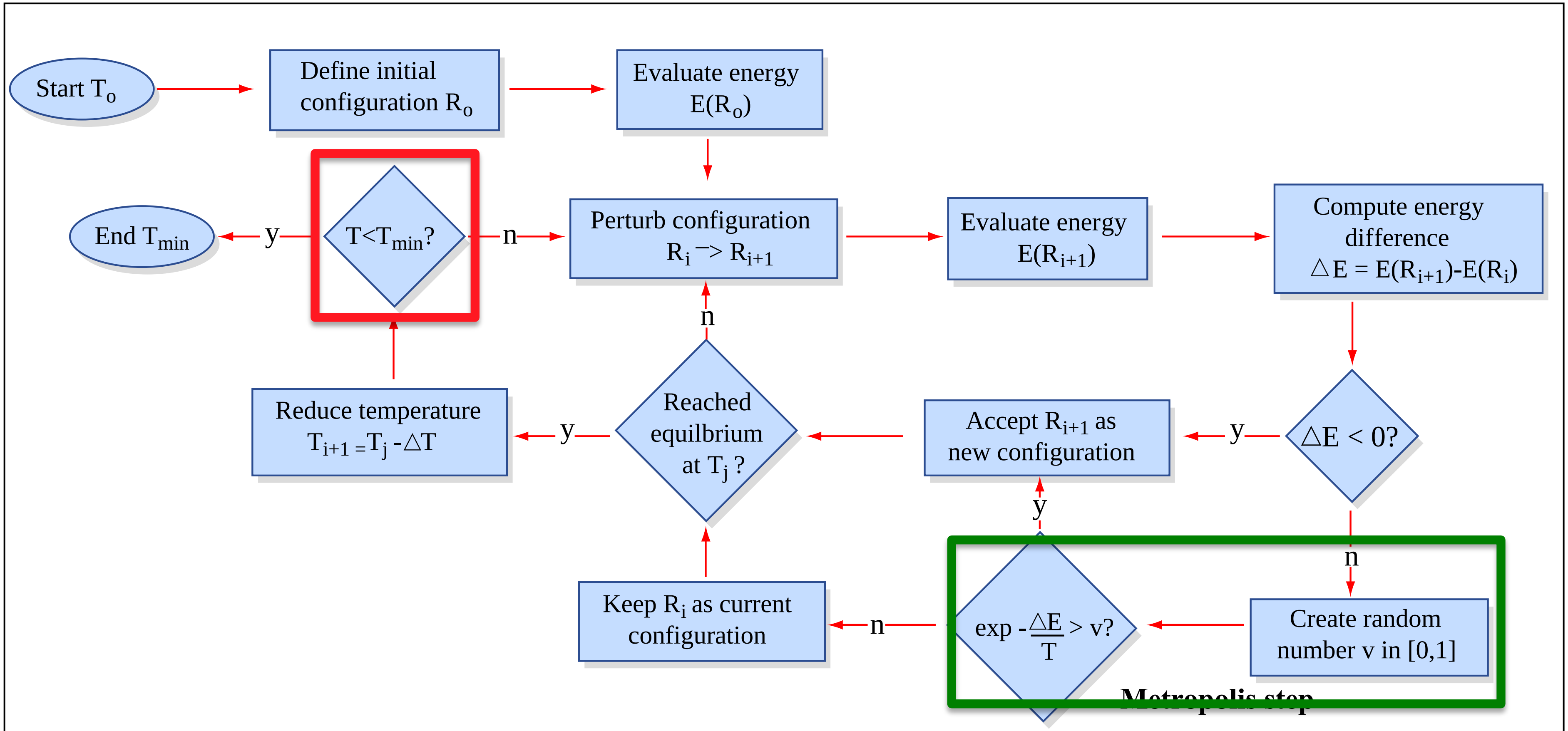


Image by MIT OpenCourseWare.

Simulated Annealing



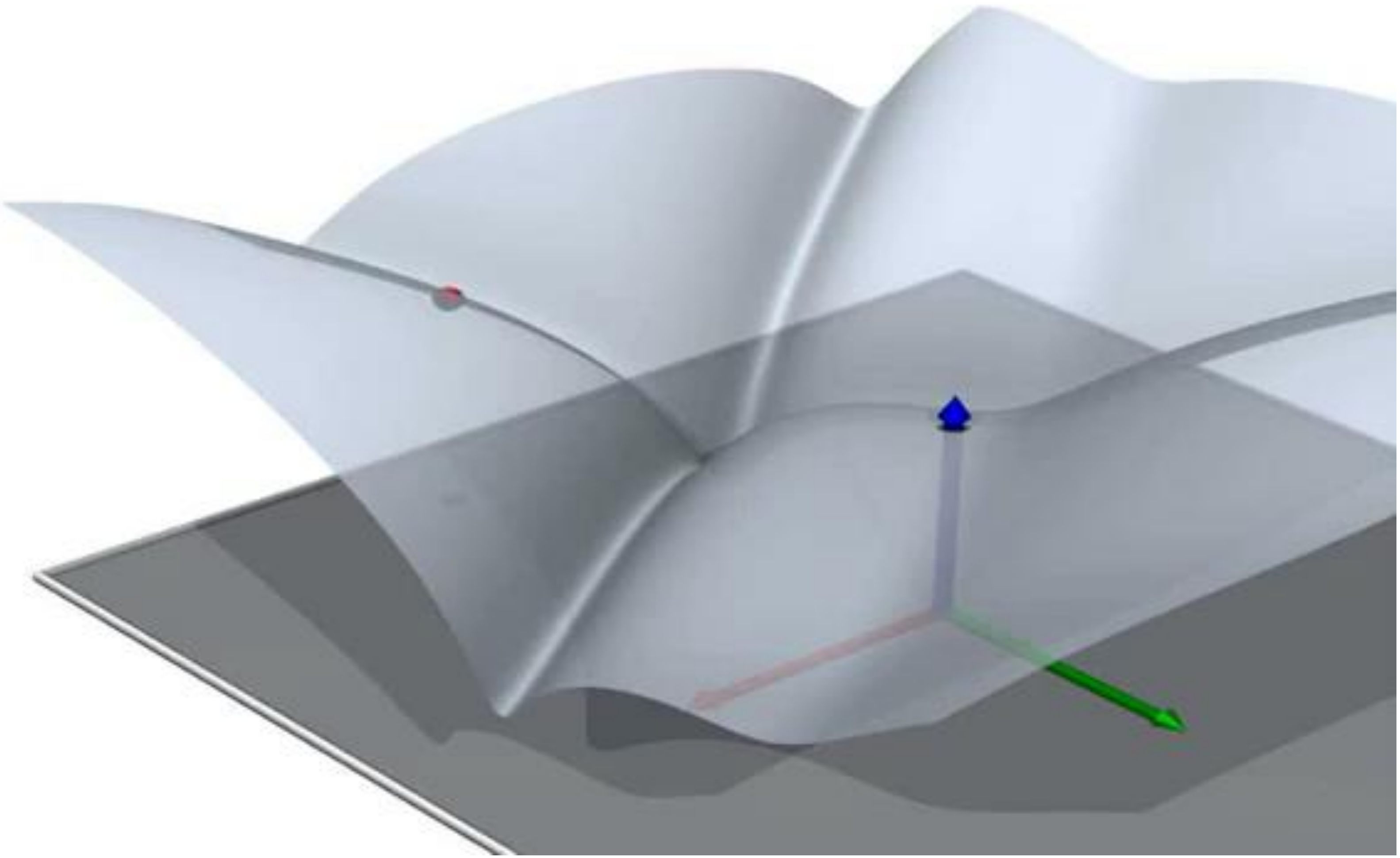
SA Block Diagram

Image by MIT OpenCourseWare.

Simulated Annealing

- Global optimization
- Combinatorial optimization
- Difficult to define good annealing schedule and neighbor generation scheme

Simulated Annealing



Examples from Graphics