Optimization Methods via Simulation

Optimization problems are very important in science, engineering, industry, ….

- Examples:
  - "Traveling salesman" problem
  - Circuit-board design
  - Car-Parrinello "ab initio" MD
  - Protein folding, molecular geometries, crystal structures.
- Concepts from simulation have provided new algorithms:
  - Steepest descent
  - Simulated annealing
  - Genetic algorithms

See lecture notes, particularly on genetic algorithms.

Numerical Recipes chapter on minimization.

Optimization

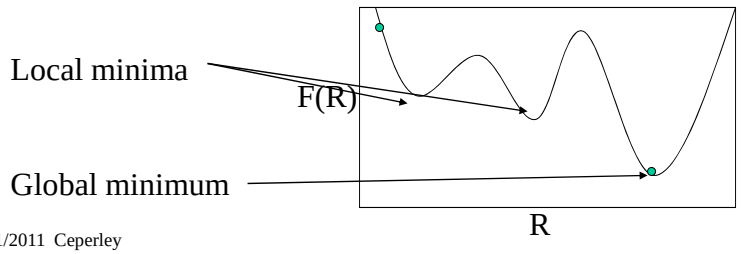
General problem: given a "cost" function, f(R), determine the value of R such that f is minimized. R can be either a multi-dimensional continuum variable and/or discrete variables.

If f(R) is convex, it's easy- just go downhill. *See Numerical Recipes.*

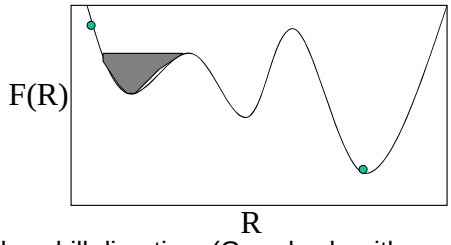
Otherwise it may be a very difficult (insoluble) problem.

Global min. optimization is NP-complete (non-deterministic, polynomial in time, complete) and no bound has been found that is N^k in computer cost, where N is number of DOF.

Two strategies: Divide-and-Conquer or Iterative.



Steepest descent



- Go in the downhill direction. (Greedy algorithm - never go uphill).
- Like MD - but cool system so temperature is close to zero.
- Will go to the nearest local minimum - *defines a basin of attraction*.
- A *frustrated* or *glassy* system has many, many local minima.
- **Goal is to find the lowest or nearly the lowest energy.**

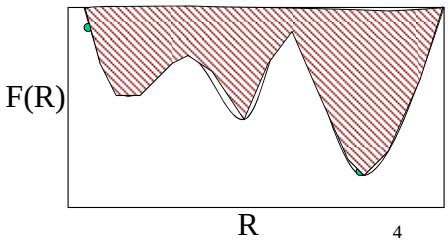
Simulated annealing

Kirkpatrick et al, Science 220, 671 (1983)

Use MD or MC to minimize by slowly cooling. Taking the temperature to zero via some *annealing schedule*.

- Guaranteed to get *eventually* the global ground state.
- System has a chance to reach equilibrium in a given basin.
- At certain temperature the system will move into a lower energy state if we cool slow enough. Entropy is related to phase transitions.
- *Annealing schedule* is important.
- Convergence if $\beta_k \leq c \ln(k)$. --> very slow convergence.

Very general powerful method.
Will give a reasonable answer.



MC or MD for Simulated Annealing

- Dynamics can be chosen for convenience – MD or MC.
- The focus is not on the naturalness of dynamics but how fast (CPU time) it moves from basin to basin.
- We don't necessarily need to have detailed balance. Just some rules that:
  - Favor movement to lower energy states.
  - Allow *some* fluctuations to higher energy to get over barriers.
- With Metropolis, we can speak the vocabulary of statistical mechanics: temperature, entropy, ergodicity, …
- We want global moves (cluster MC) that aren't too expensive in energy.

(SA continued)

- To reach the optimal (or ground state) think how nature achieves ground state structures.
  - **Slow cooling (annealing)** a liquid allows us to access the global ground state, just as happens in the real world.
  - **Fast cooling (quenching)** a liquid leads to a metastable, local minimum (just like glass).
- For MD:
  - Choose annealing schedule (i.e. cooling rate).
  - Solve Newton's Eqs. for each atom by numerical integration.
  - Control T via coupling to heat bath, like Nose-Hoover Thermostat.
- For MC: with Metropolis, we can speak of temperature, entropy, ergodicity, …
  - Reject or Accept according to Metropolis Algorithm
    [$p = \min(1, e^{-E/kT})$] which obey microscopic reversibility.
  - Choose annealing schedule (i.e. cooling rate).

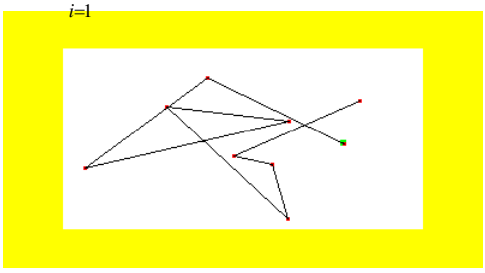
If T is low, method is like Steepest Descents--you don't move uphill.

Traveling salesman problem

What is the optimal path starting from a given city, which visits the other cities once and only once.

- Problem is provably difficult. (N!) possibilities to examine.
- Key issues:
  - Update rules: Fast and effective
  - Annealing schedule: Just slow enough

$$Energy = L = \sum_{i=1}^{N} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} - K(s_i - s_{i+1})^2$$



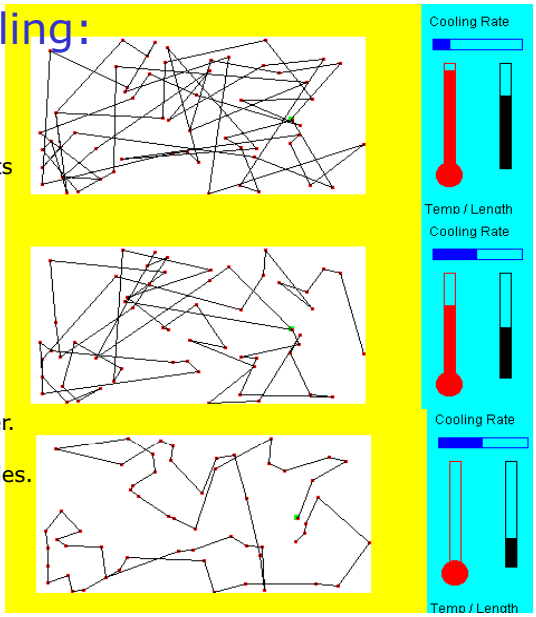
K is additional penalty for crossing between sides of rivers, or between processors in circuit.

Example of cooling:

Using **Genetic algorithm** concepts extremely useful for more rapid solution of this type of problem.

Namely, in combination with simulated annealing, you do evolutionary type of moves:

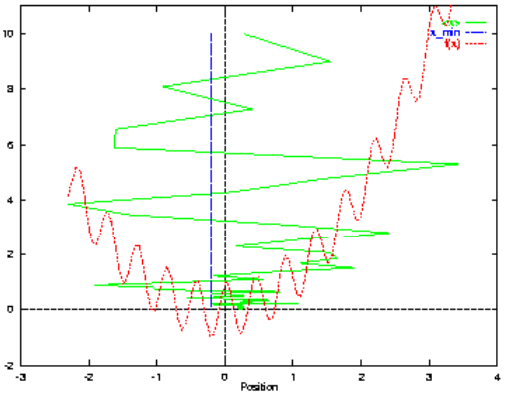
- **rearrangement:** a randomly picked ***sub-string of cities*** is
  - inversion: ran in opposite order.
  - crossover: ran between two randomly picked cities.

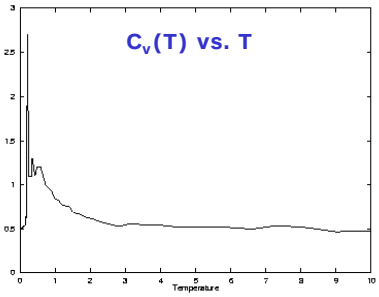


Simulated Annealing for Complex Potentials

- Potential in Red.
- Average position at T in Green.
  - final state is near Abs. Min.
- Absolute minimum in Blue.

- Will I get same answer every time?
- Will answer depend on annealing?
- Can *thermodynamic analogy* be used effectively via $C_v(T)$? How?
- Is there same as "critical slowing down"?





Why the spike in C_v at low T?

- Use C_v to control dT steps!

Did you find the global minima?

- Does it matter? (Engineering criterium-just give me a reasonable answer quickly).
- Finish off with steepest descent at the end (for continuous variables).
- Do it many times and see if you always end up in the same place. Either
  - you have the global minima, or
  - your dynamics is very bad.
  - One can apply statistical criteria to decide if you have the global minimum.
- For some problems in nature, such as protein folding, the idea is not to find the global minima of the energy, but to see the sequences of folded structures. Maybe nature cannot find the global mimimum either.

Car-Parrinello molecular dynamics

- MD for ions and LDA (i.e. quantum mechanics) for electrons.
- We need to solve an *eigenvalue* (e.v.) problem at each step.
- We can formulate as an optimization principle-minimize the electronic energy; the electrons are in the T=0 in the quantum ground state.
- Do simulated annealing with MD.
- Use a thermostat to keep the "electron temperature" pretty cold. (close to the ground state.)
- By doing it this way (rather than as an e.v. problem) you use the slow steady change of electronic states to speed up code.

Genetic algorithms (GA)

- General technique to make lots of copies in region as the cost function gets smaller (Branching)
- Do mutations of those copies (random walk)
- Based on evolutionary/genetic analogy.
- Relies on making mutations that have a reasonable chance of staying close without high cost.
- See lecture notes for more details of genetic algorithms.