Stats 318: Lecture

Agenda: Markov Chain Monte Carlo

- The traveling salesman problem
- Solution by simulated annealing

The traveling salesman problem

- d cities labelled $\{1, 2, \ldots, d\}$
- distances between city i and city j: $d(i, j)$.
- Problem find a tour of all the cities (starting and ending in city 1, say) such that the distance travelled is minimum.

A tour is a permutation $\{s(2), \ldots, s(d)\}$ of $\{2, \ldots, d\}$ and the distance traveled is

$$\sum_{i=1}^{d} d(s(i), s(i+1)),$$

with the convention $s(1) = s(d+1) = 1$

Solution by simulated annealing

```
% Solution of the traveling salesman problem by simulated anealing

% location of the cities
l = 10; d = l.^2;
cities = zeros(d,2);
for k = 0:(d-1),
  cities(k+1,:) = [fix(k/l) rem(k,l)]/l;
end

% distances between cities
dist = zeros(d);
for k1 = 1:d,
  for k2 = 1:d,
    dist(k1,k2) = norm(cities(k1,:)-cities(k2,:));
  end
end
```

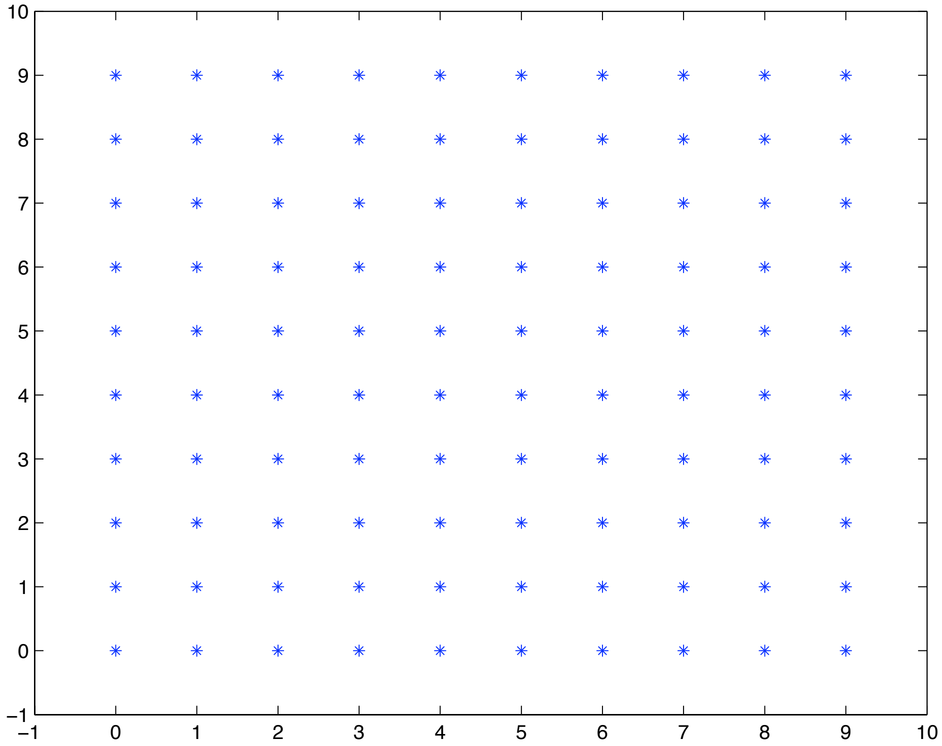
```
% Initial tour
tour = [1, 1+randperm(d-1), 1];
figure, plot(cities(:,1),cities(:,2),'*'), axis([-.1 1 -.1 1])
hold on
plot(cities(tour,1),cities(tour,2),'r')
hold off

% Number of steps and cooling schedule
nsteps  = 800000;
Tinit = 5;
Tfinal = .01;
n = 1:nsteps;
T = Tinit*(Tfinal/Tinit).^(n/nsteps); % geometric schedule
% T = Tinit + (Tfinal-Tinit).*n/nsteps; % linear schedule

Length = zeros(1,nsteps+1);
Length(1) = LengthTour(tour,dist);
```

```
for n = 1:nsteps,
  % Pick two cities at random
  pair = randsample(2:d,2);
  i = min(pair); j = max(pair);

  % Replace segment
  new.tour = tour; new.tour(i:j) = tour(j:-1:i);

  % Compute length of new tour
  if rem(n,1000) == 0,
      new.length = LengthTour(new.tour,dist);
      Delta = new.length - Length(n);
  else
      Delta = dist(tour(i-1),tour(j)) + dist(tour(i),tour(j+1))
            - dist(tour(i-1),tour(i)) - dist(tour(j),tour(j+1));
      new.length = Length(n) + Delta;
  end
```

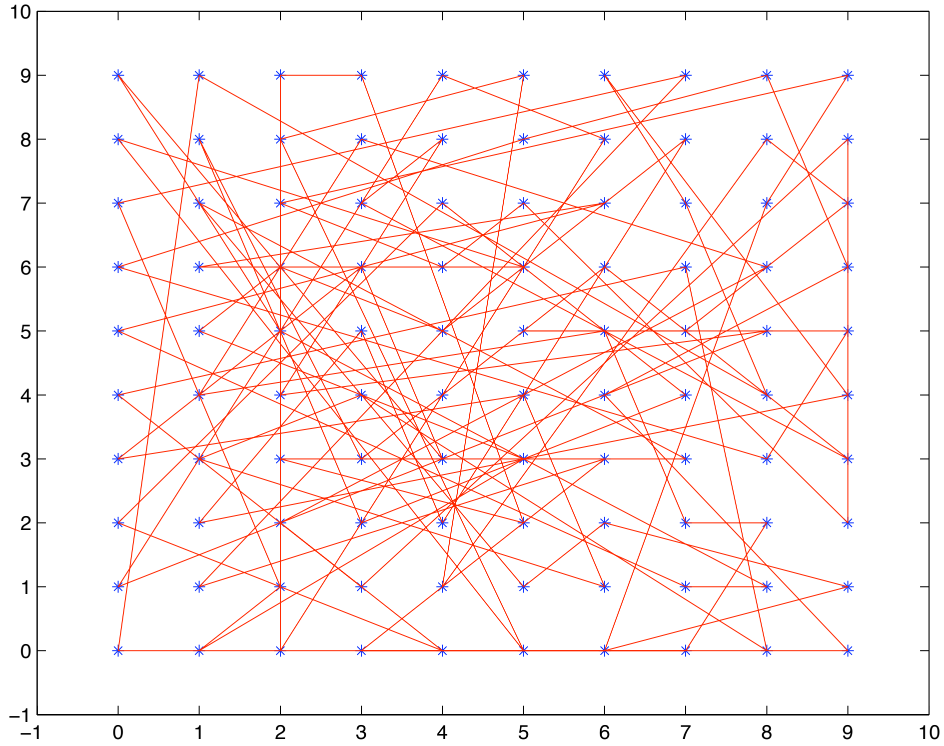
```
  % Decide whether or not to accept the new tour
  if rand(1) < min(exp(-Delta/T(n)),1),
    tour = new.tour;
    Length(n+1) = new.length;
  else
    Length(n+1) = Length(n);
  end
end
```

```
function Length = LengthTour(tour,dist)
% Calculate the length of a tour

Length = 0;
for k = 1:(length(tour)-1),
  Length = Length + dist(tour(k),tour(k+1));
end
```

Initial tour

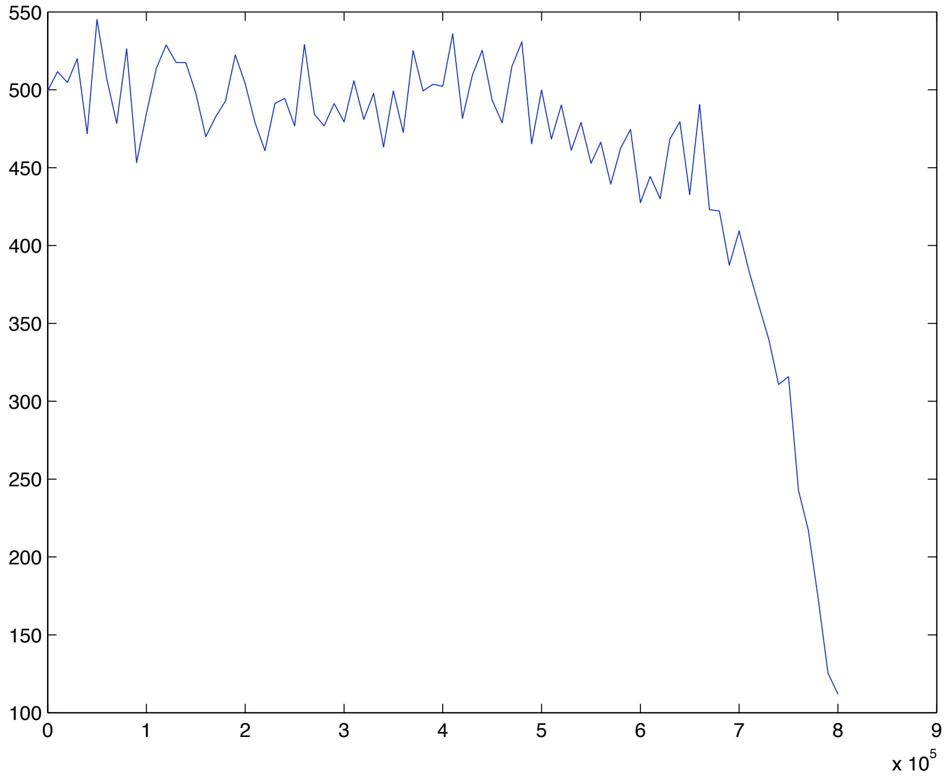


Cities



Initial tour

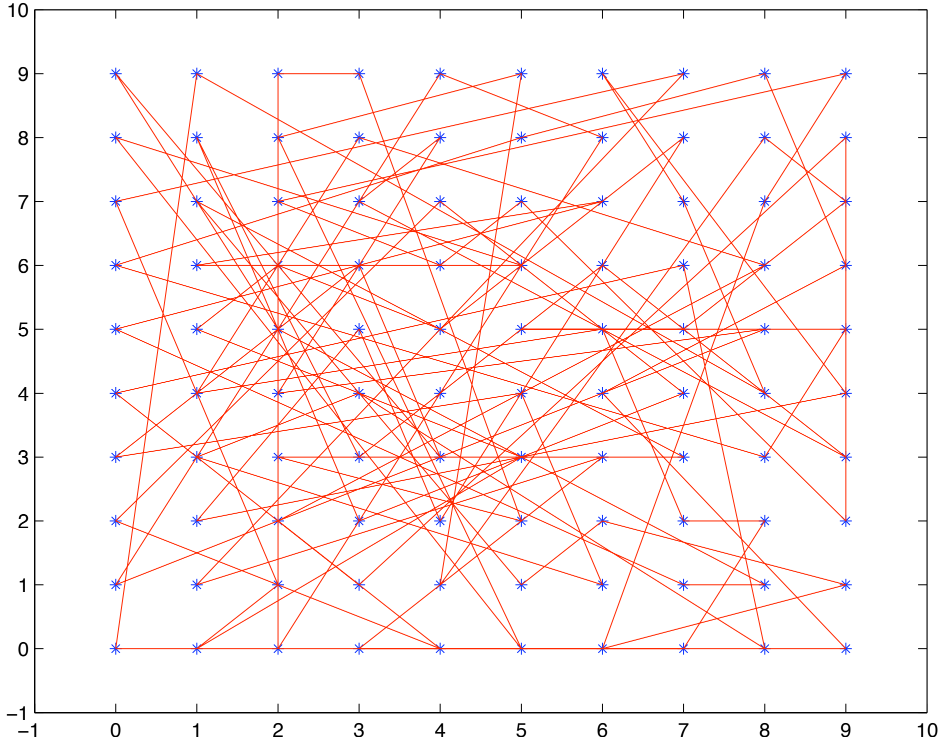
Dynamics of tour length



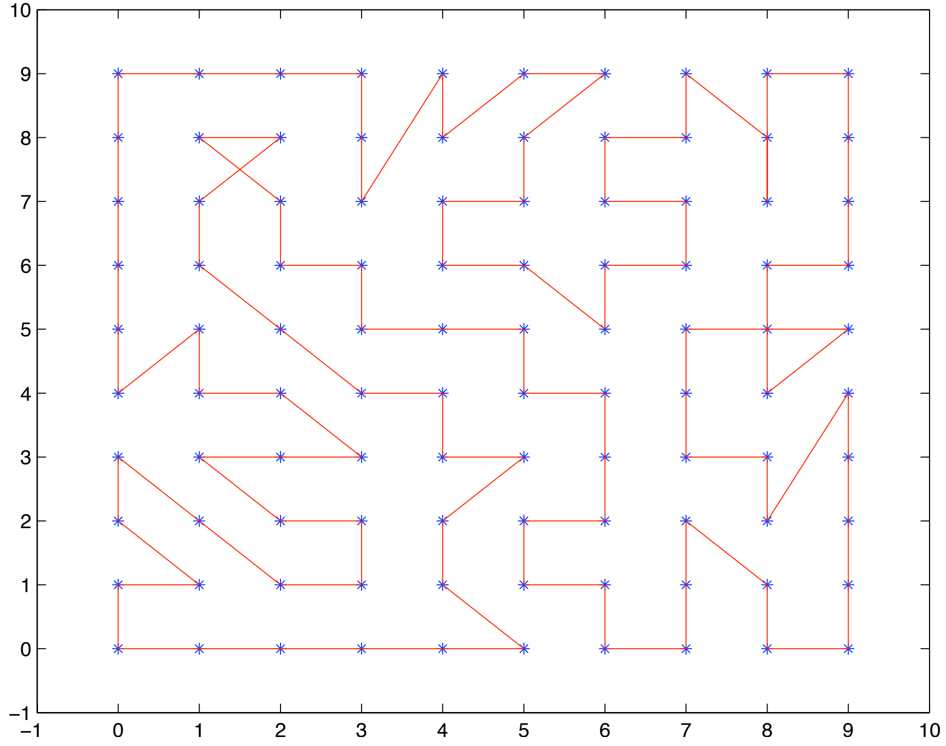
Length of the tour (sampled every 10,000 steps)
as a function of time (number of steps)

- Final value is 111.9
- Optimal value is 100

After 800,000 steps



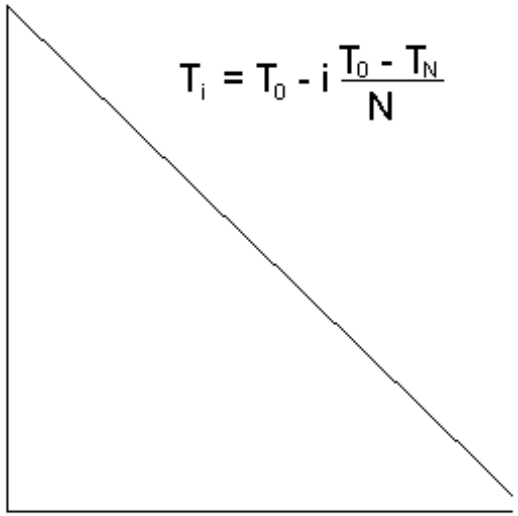
Initial tour



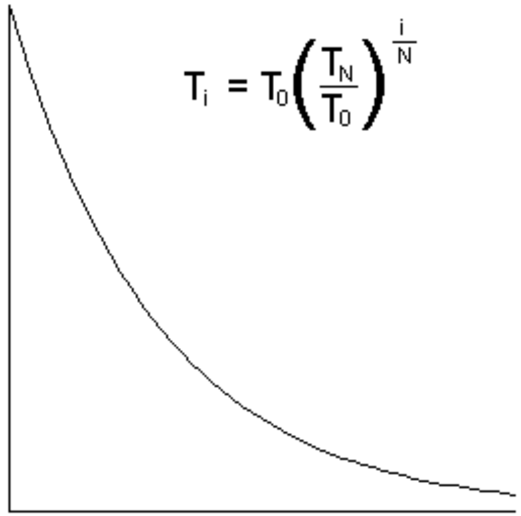
Tour after 800,000 steps

Example of cooling schedules (Brian T. Luke)

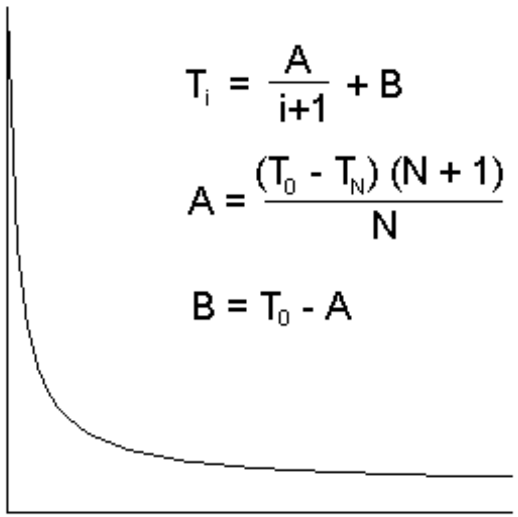
Cooling Schedule 0



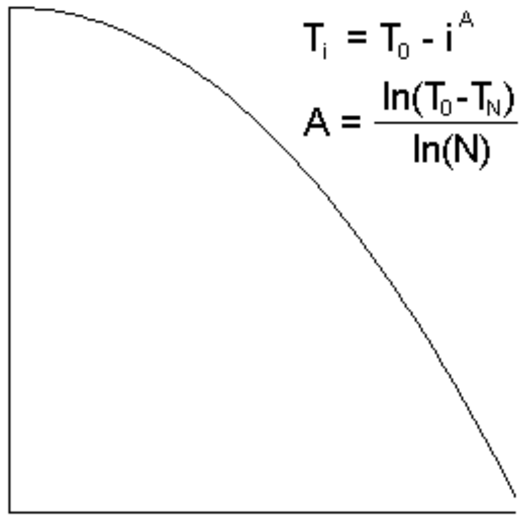
Cooling Schedule 1

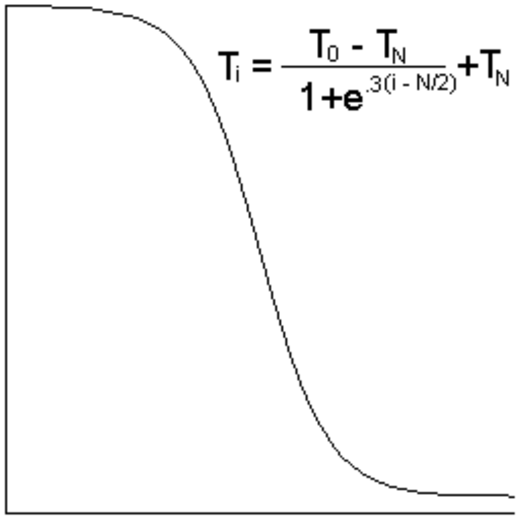


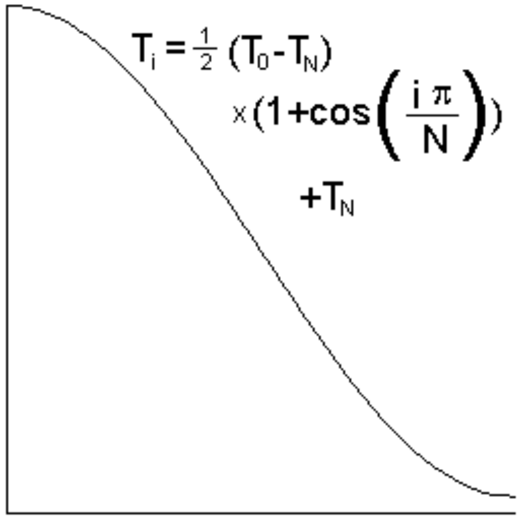
Cooling Schedule 2



Cooling Schedule 3

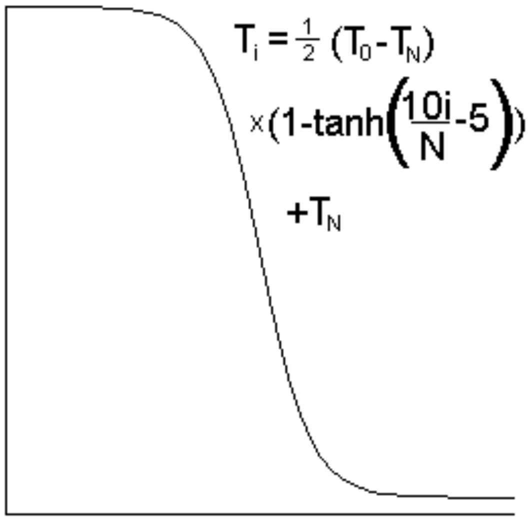


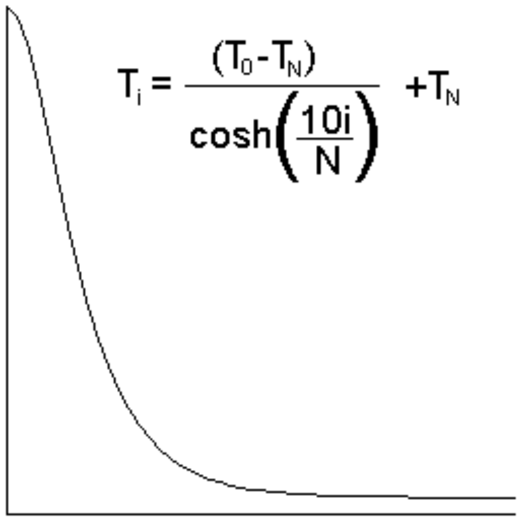




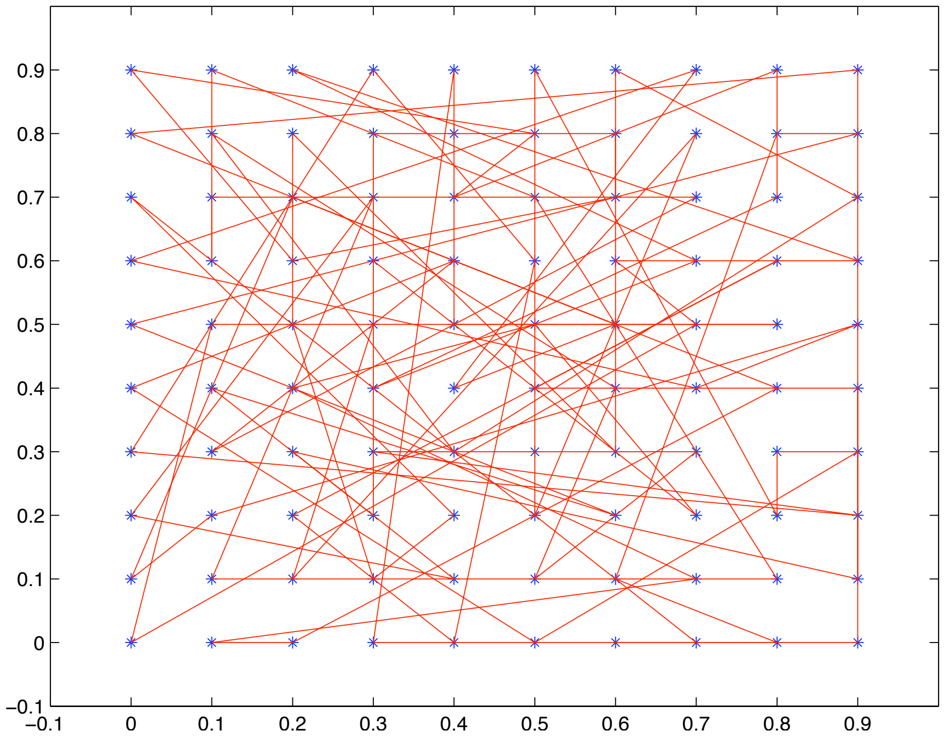
Cooling Schedule 6

Cooling Schedule 7

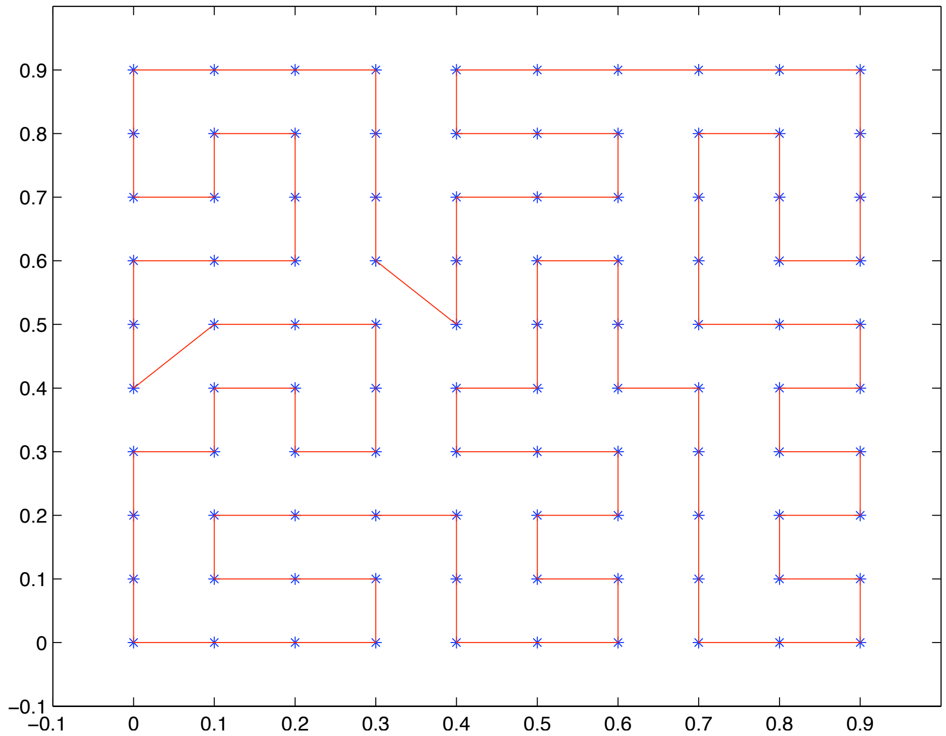




Another run with a geometric cooling schedule



Initial tour



Tour after 800,000 steps

- Very close to the solution: optimal length = 10; current value 10.08
- I performed many such simulations! Often times, not that lucky!