Monte Carlo Methods for Computation and Optimization

Final Presentation

# Simulated Annealing for Constrained Global Optimization

H. Edwin Romeijn & Robert L.Smith (1994)

Presented by

Ariel Schwartz

# Objective of Talk

- Briefly discuss local search optimization methods and their limitations
- Explain the need for randomness for global optimization
- Introduce the Hit & Run and Hide & Seek algorithms
- Show that the Monte-Carlo approach leads to a simulated annealing process
- Disscuss considerations for implementing simulated annealing
- Highlight connection to many topics discussed in class
- Present a visualization of simulated annealing
- Discuses the effectiveness of simulated annealing

# The Problem

Solving the following constrained global optimization problem:

$$\min \ f(x)$$

$$\text{subject to } x \in S$$

# The Setting

$$\min f(x)$$
$$\text{subject to } x \in S$$

- $S$ is a compact body in $\mathbb{R}^d$ (a non-empty, closed, bounded subset)
- $f(x)$ is a continuous real-valued function on $S$
- $S$ is not necessarily convex or even connected
- $f(x)$ is not necessarily differentiable

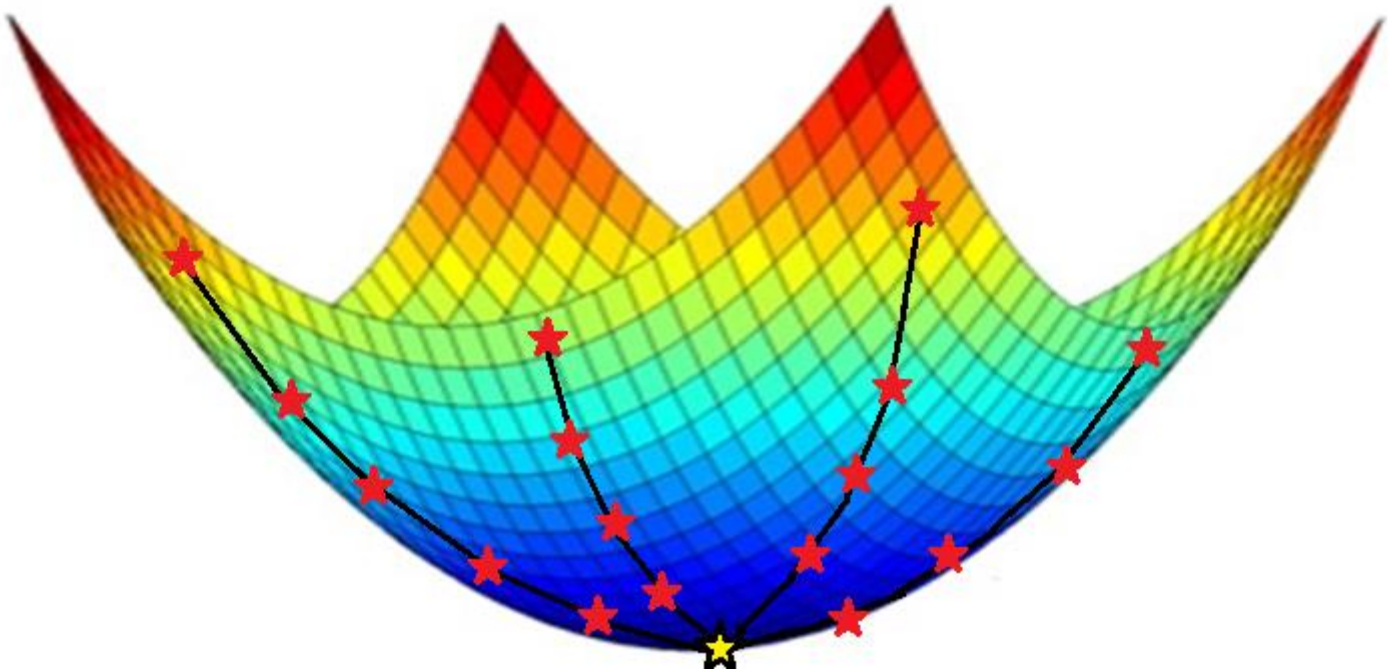
# Differentiable Functions

- For a differentiable function we can calculate the gradient
- Gradient is zero at an interior local minimum
- Global minimum is either a local minimum or a boundary point
- Analytical solution possible in some simple cases
- Numerical solution needed in general

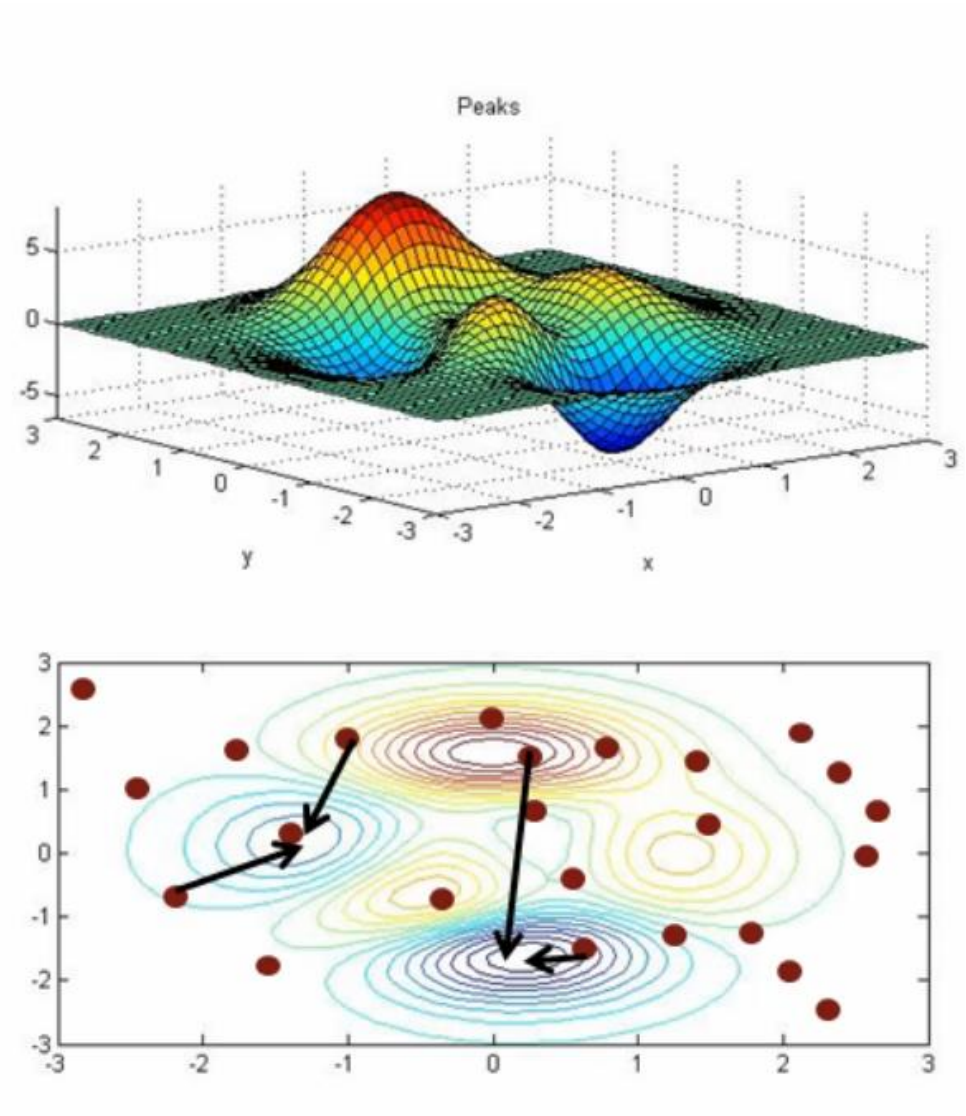
# Numerical Local Search

- “Gradient Descent” based algorithms
- Require functions to be differentiable
- Moves in the direction opposite to the gradient
- Direction of steepest descent
- For a small enough step size decrease in value is guaranteed
- Adaptive step size can guarantee a decrease at each iteration
- At local minima the gradient is zero and algorithm stops
- Convergence to a local minima guaranteed
- For convex problems converges to global minima is guaranteed

# Convex Case



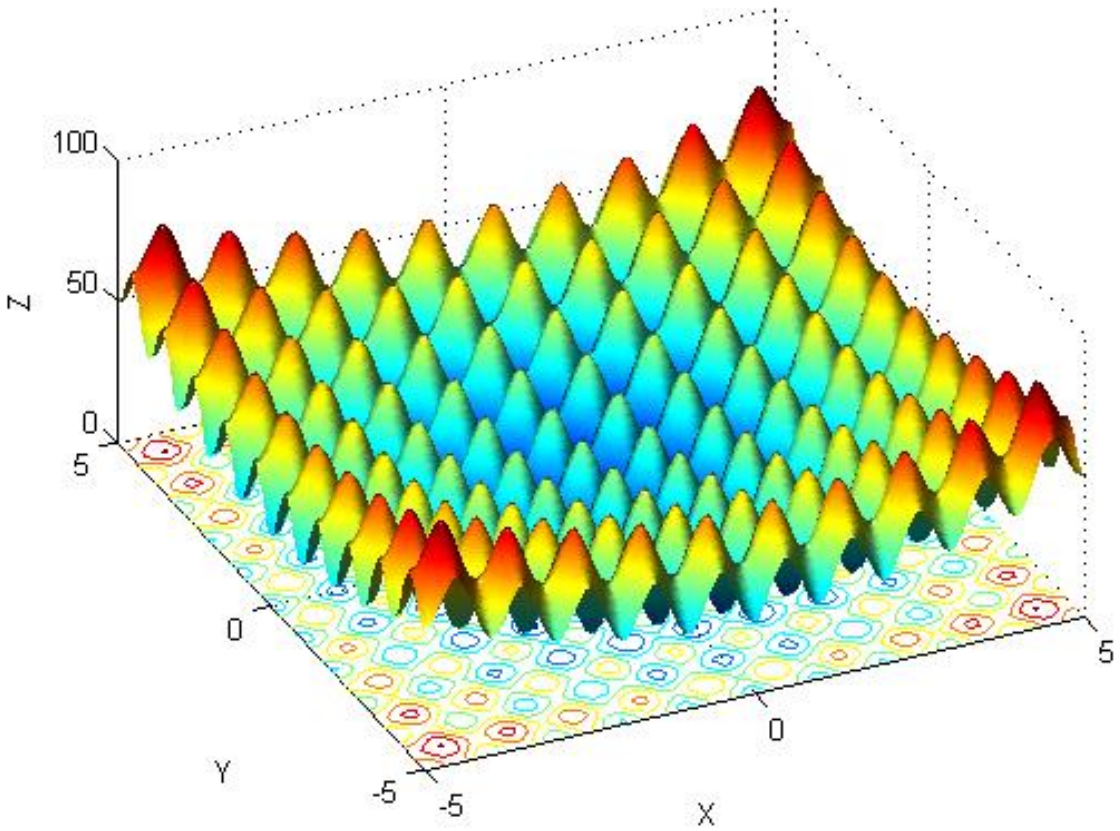
# Non-Convex Case



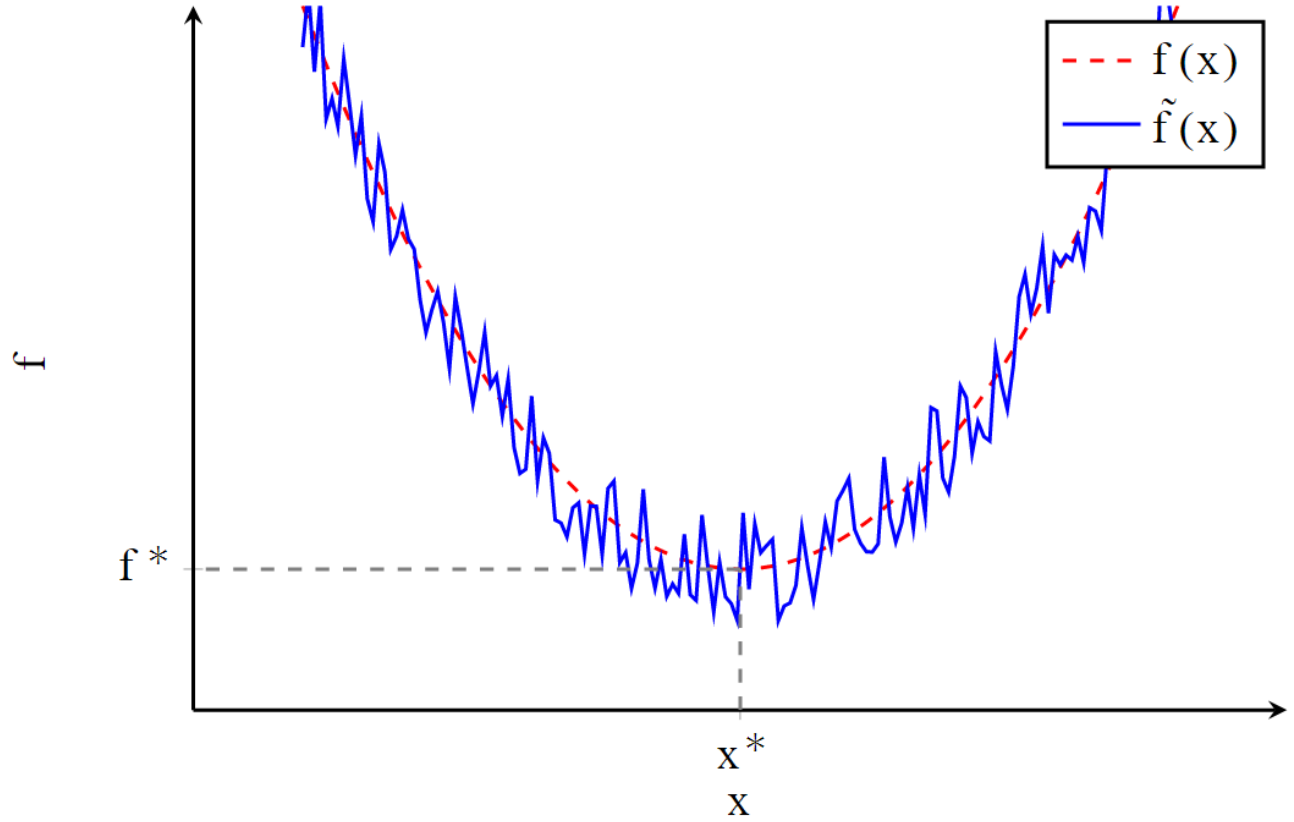

# Multi-Start Algorithms

- Perform local search starting from different points
- Each local search converges to a local minimum
- Take that lowest local minimum as an approximation of the global minimum
- Can not guarantee global minima
- Requires a method for “covering” feasible area (parametrization & grid)
- When function has multiple local minima we need a lot of starting points
- Becomes completely impractical in the presence of any noise

# Multi-Start Algorithms




# Gradient Sensitivity to Noise

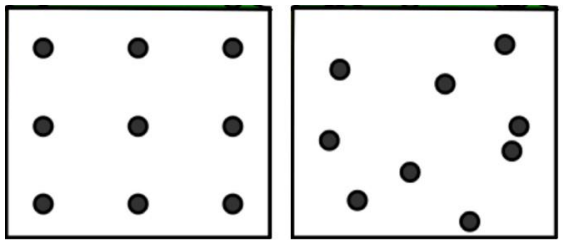



# The Need for Randomness

- Need a method for generating points in $S$
- Easier mathematical handling of constraints (Accept-Reject)
- Multi-start can use random starting points
- Noise is still a problem
- Randomness is needed in the descent process itself
- Sometimes one “wrong” move discovers a whole area of better values
- “Mistakes” allow exploration but eventually we should “settle down” and “exploit” what you found so far
- The Exploration-Exploitation tradeoff
- Much like dating when your young and later getting married

# The Monte-Carlo Approach

- Perform a randomized search
- Instead of sampling feasible set on a deterministic grid sample it stochastically
- Can reduced the problem of constraint handling
- If the sampling PDF is strictly positive on $S$ and $f\left(x\right)$ is continuous it is guaranteed to converges to global minimum
- Random uniform search is proven to have better convergence properties than determinitstic grid search for high dimensions



# Random Variable Generation

- Distribution specific methods - LCG, Box-Muller, Binomial, Geometric
- Inverse-Transform Method – Good for simple distributions:

$$F^{-1}(y) \triangleq \inf\{x : F(x) \geq y\} \qquad y \in [0,1]$$
$$x = F^{-1}(U) \qquad U \sim U[0,1]$$

- Accept-Reject method – Sample a candidate point from a proposal distribution $h(x)$ such that $f(x) \leq C \cdot g(x)$ for some $C > 1$.
  Then accept this point as being from $f(x)$ with probability:

$$P(x) = \frac{f(x)}{C \cdot h(x)}$$

- We do not know bound on $f(x)$ and AR is inefficient in high dimensions
- If we over estimate becomes inefficient

# The MCMC Method

- Sample from the limiting distribution of homogenous continuous Markov chain
- Reversibility, aperiodicity and irreducibility ensure the convergence to a single limiting stationary distribution regardless of initial point
- We wish to construct a chain with a uniform stationery distribution:

$$s(x) = \frac{1}{|s|} I_{x \in S}(x)$$

- "Convergence" implies a certain waiting time called the "Mixing Time"

# Reversibility Condition

- Reversibility (or the detailed balance) condition:

$$p(y \mid x)s(x) = p(x \mid y)s(y)$$

- For $x, y \in \mathbb{R}^d$ where $s(x)$ is the stationary limit distribution over $\mathbb{R}^d$
- We need to construct a chain with $p(y \mid x)$ such that the reversibility condition holds for the desired $s(x)$

# Proposal Chain

- We will make two assumptions about $p(y \mid x)$ which we later show we can easily fulfill in a number of different ways:
- Symmetry on feasible set:

$$p(y \mid x) = p(x \mid y) \qquad x, y \in S$$

- Will not "step out" of feasible set:

$$p(y \mid x) = 0 \qquad x \in S, y \in S^C$$

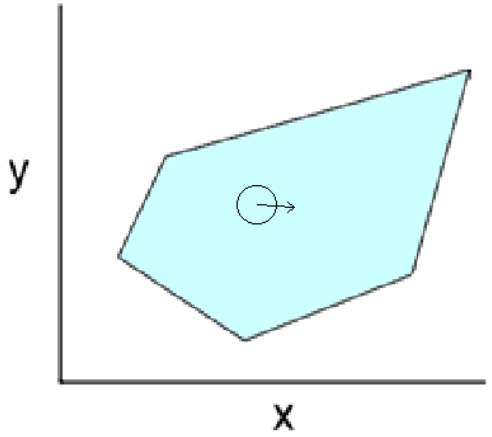
- Given this the stationery distribution over $S$ is uniform
- Many chains fulfill this but they might differ in mixing time
- Requires an initial point in $S$

# Proposed Chain Transitions

- We are left with building a chain with transition probability $p(y \mid x)$
- Needs to not leave $S$
- Needs to be symmetric
- Needs to be easy to sample from
- A large single-step variance should reduce mixing time
- We first chose a direction for a step and then perform A-R as to not violate constraints. This time A-R is in 1D which does not suffer from the dimensionality curse
- If direction selection is symmetric then transition probability is also symmetric

# Choosing a Direction

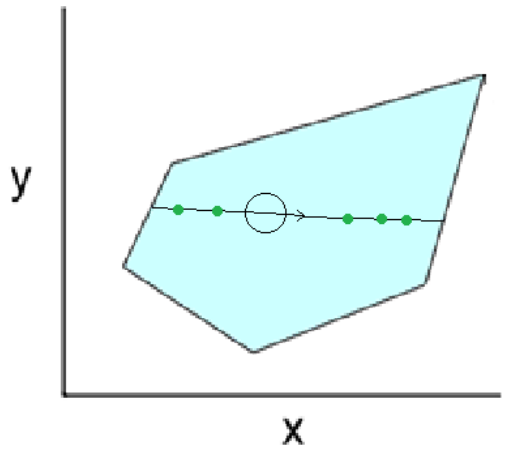
- We directions around a point
- We sample randomly on a:
- D1 – Hyper-sphere
- D2 – Hyper-ellipsoid
- D3 – Hyper-cube

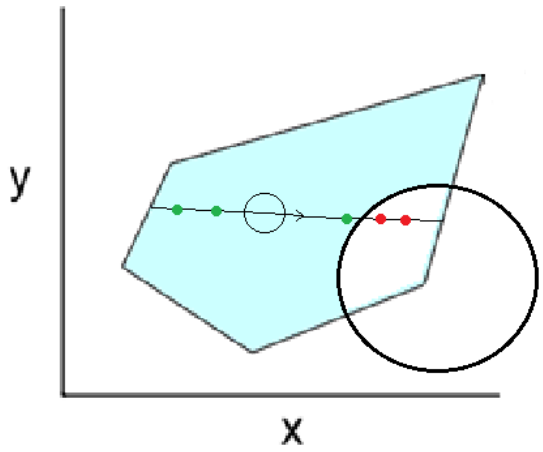



- Coordinate Selection
- Ellipsoid and cube allow scaling to better fit problem
- Coordinate Selection is more computationally efficient but does not ensure eventual convergence to global minima

# Step Size

- Find the intersection of the selected direction with the set of points fulfilling all **linear constraints** (possible due to compactness, bound by box)
- Select a point uniformly on feasible segment using Accept-Reject (fulfilling non-linear constraints)
- This sub problem is 1-D, the Accept-Reject method is reasonably efficient for this task




# Hit & Run

- A random walk that can globally reach across the support of the distribution in one step is called a Hit-and–Run sampler.
- If direction selection is symmetric then the transition probability is symmetric
- Convergence to uniform distribution
- Positive step PDF, every point is reachable in one step
- Large steps are just as probable as small steps, leads to faster mixing time
- Offers the promise of faster convergence to the target distribution than conventional small step
- Eventually converges to global minima with probability 1

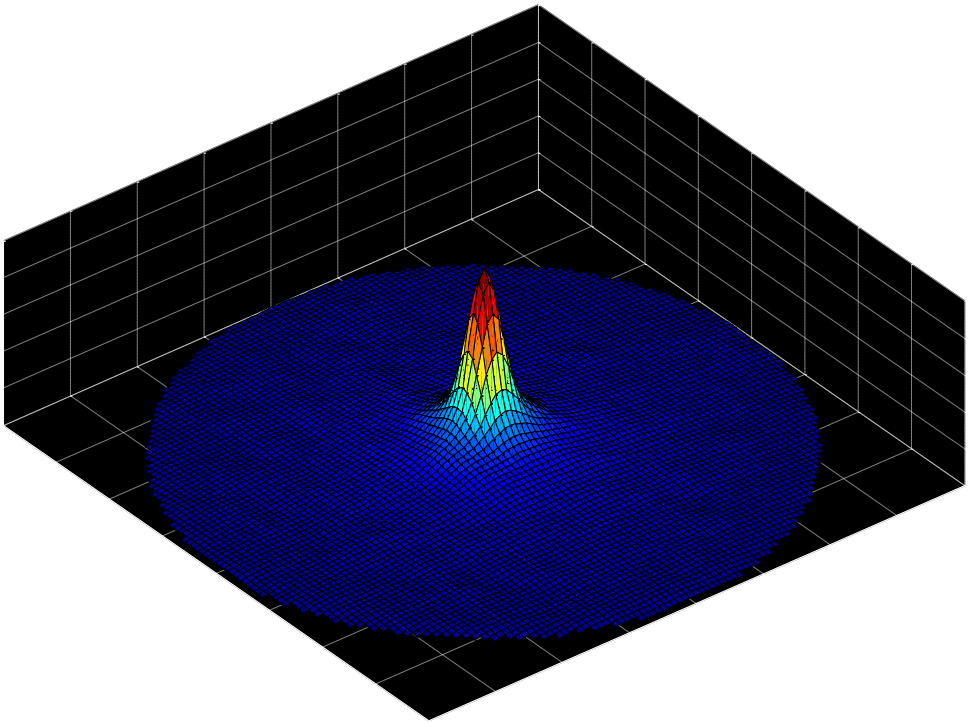
# Transition Probability – D1

$$\frac{\pi^{d/2}}{\Gamma\left(\frac{n}{2}+1\right)}\left[\left(r+dr\right)^{d}-r^{d}\right]\cdot p\left(r\right)=\frac{1}{R}\cdot dr$$

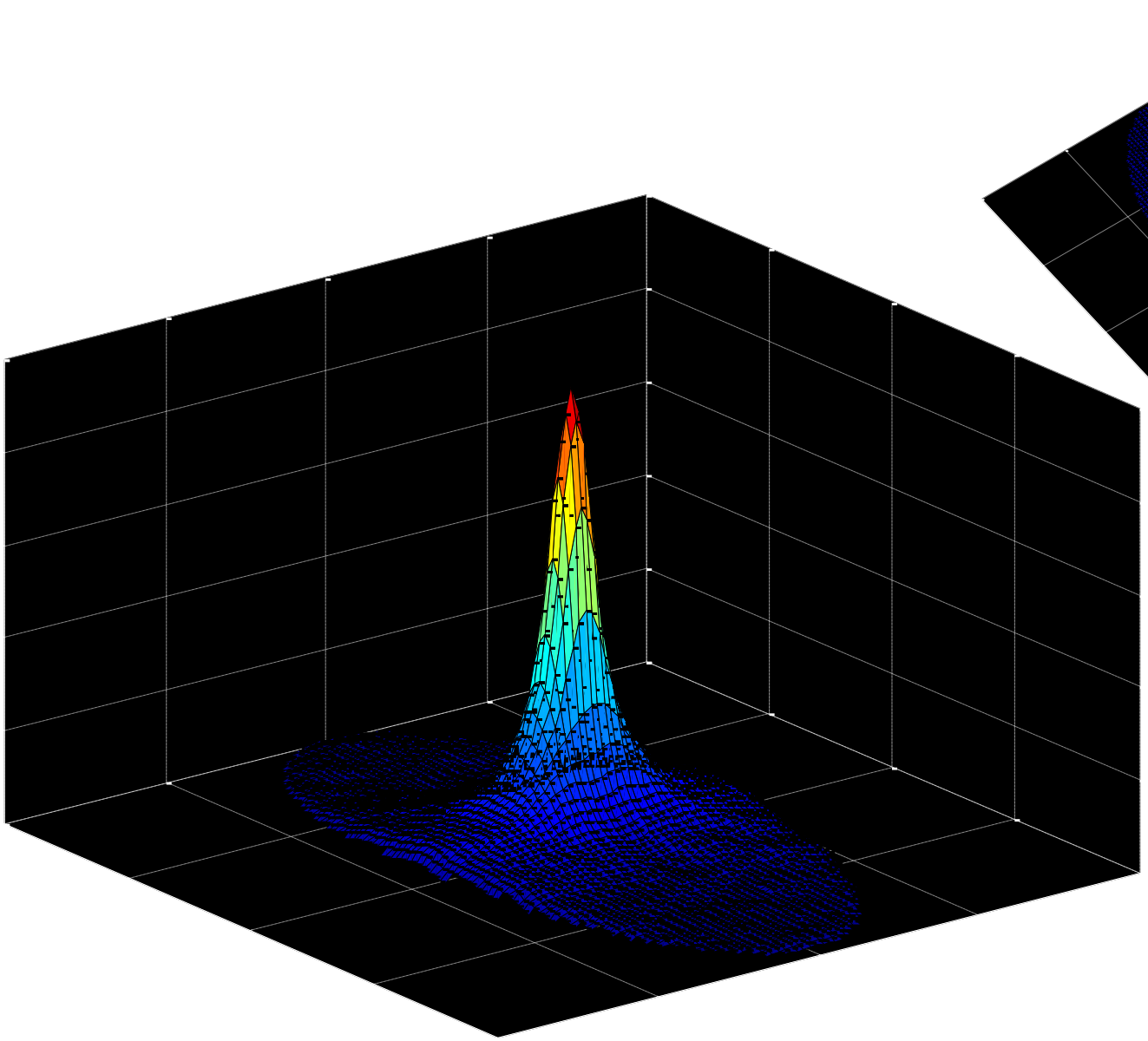
$$p\left(r\right)=\lim_{dr\to+0}\frac{1}{R}\cdot dr\,\frac{\Gamma\left(\frac{n}{2}+1\right)}{\pi^{d/2}\left[\left(r+dr\right)^{d}-r^{d}\right]}$$

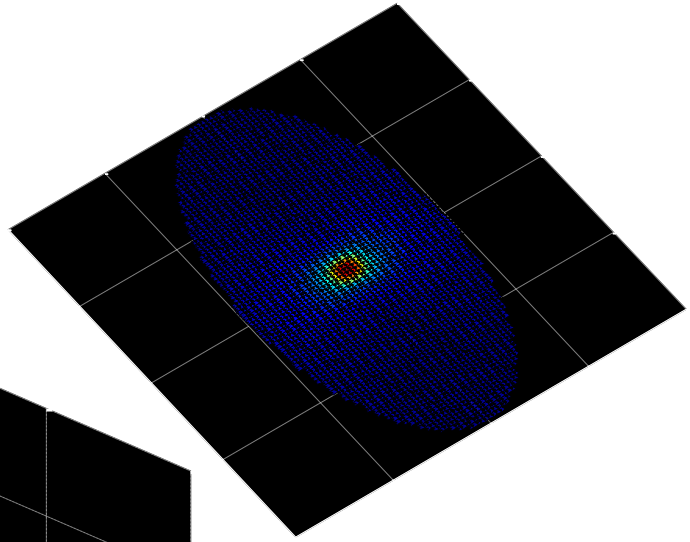
$$=\lim_{dr\to+0}\frac{1}{R}\cdot dr\,\frac{\Gamma\left(\frac{n}{2}+1\right)}{\pi^{d/2}\left[d\cdot r^{d-1}\cdot dr\right]}$$

$$=\frac{1}{R}\,\frac{\Gamma\left(\frac{n}{2}+1\right)}{\pi^{d/2}\cdot d}\cdot\frac{1}{r^{d-1}}$$

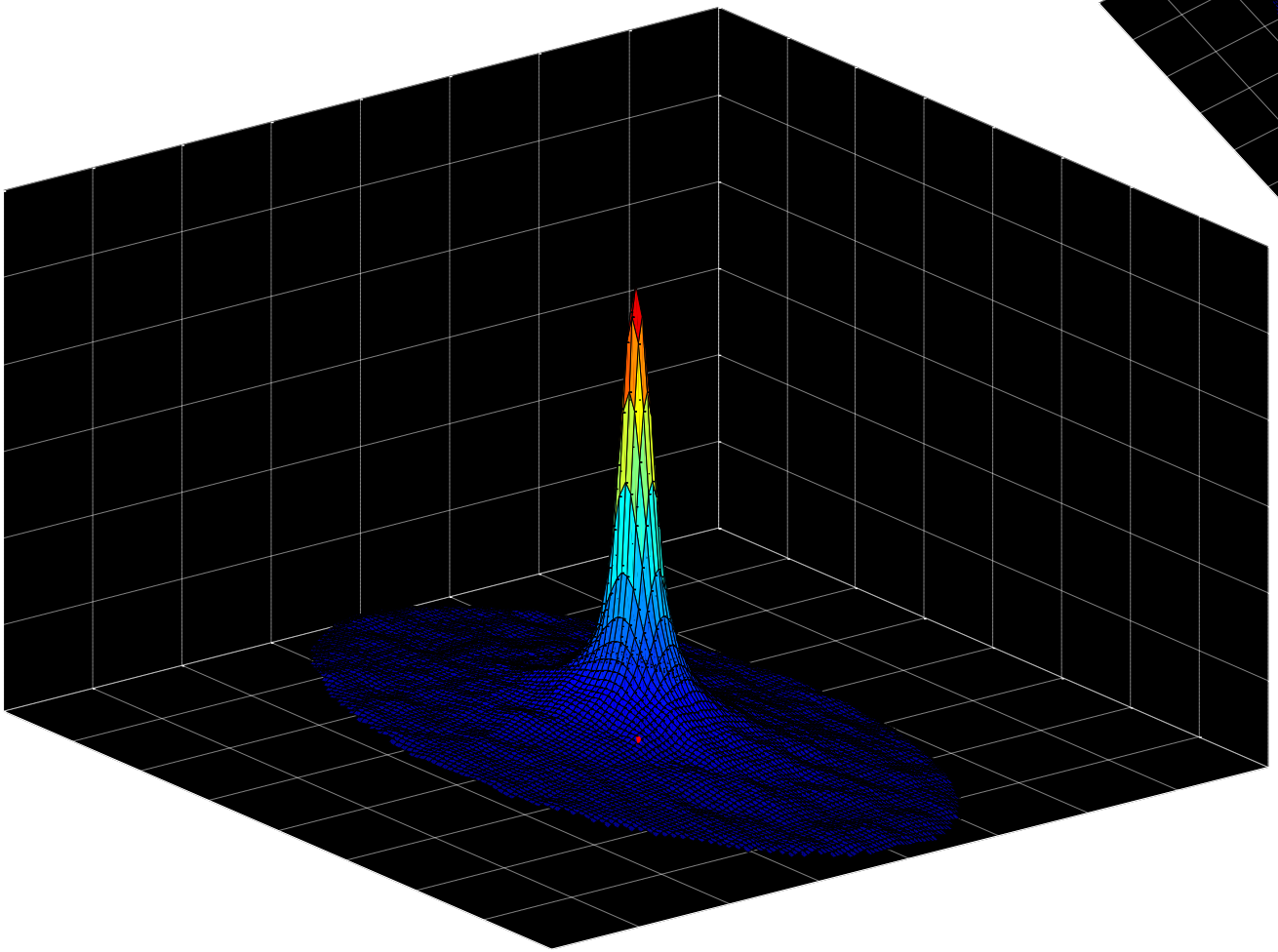


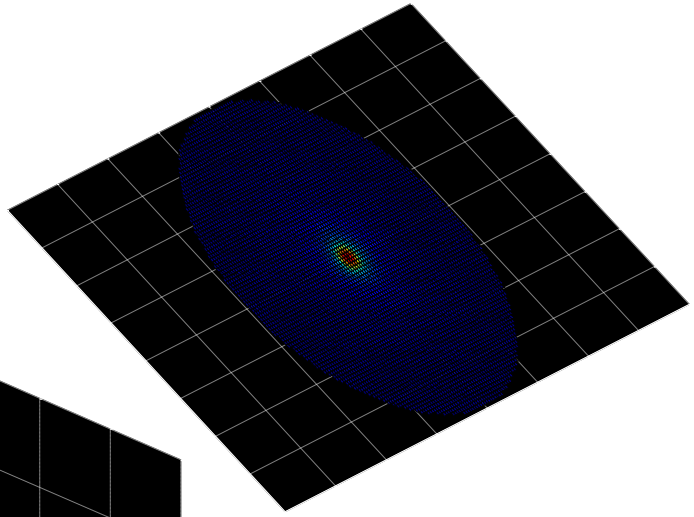
# Transition Probability – D1



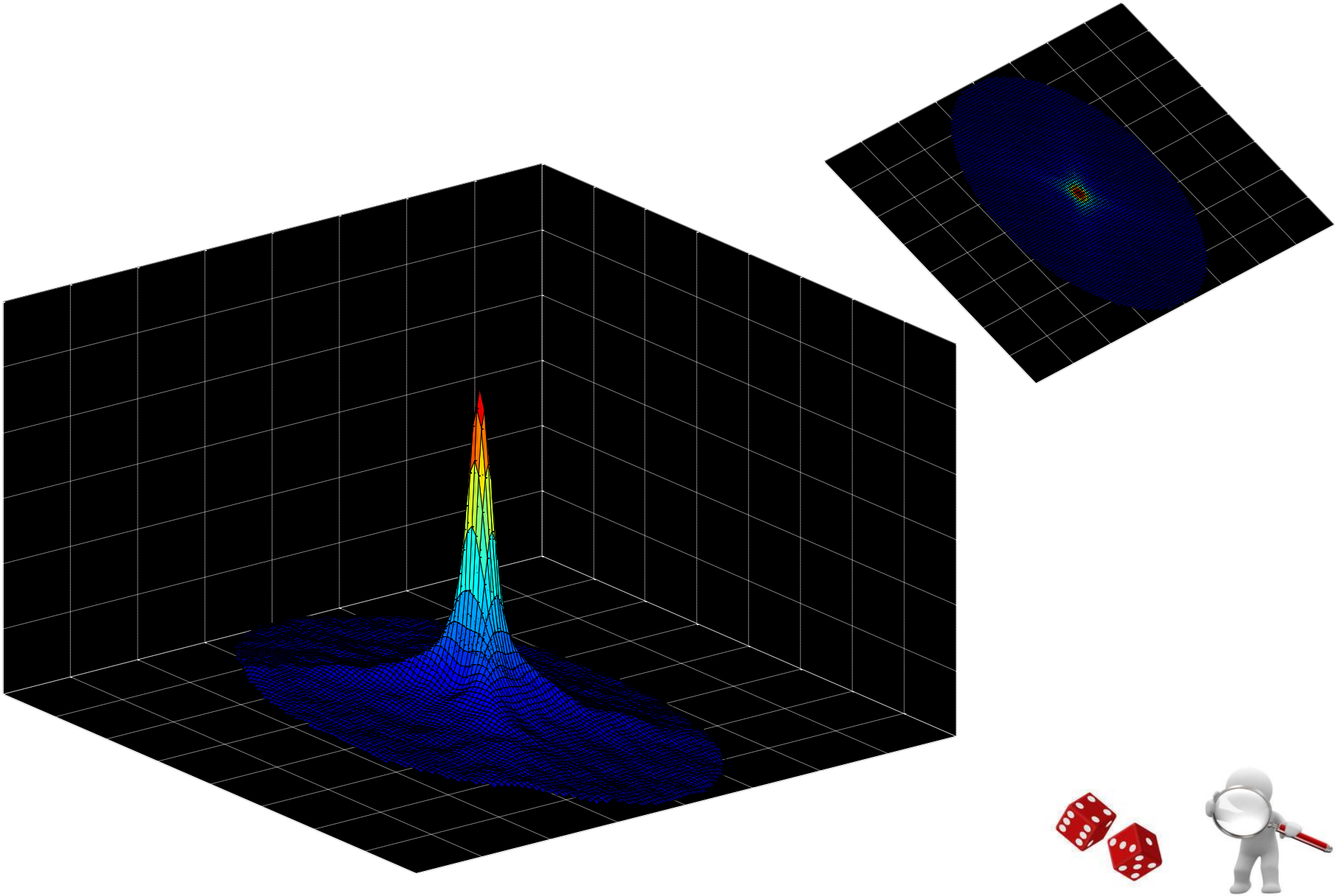


# Transition Probability – D2





# Transition Probability – D3



# Improving Hit & Run

- Transitions are still quite local
- Moving to a better position makes sense since the function is continuous
- Same as simple Hit & Run but will only move  if cost function is reduced
- Here using a Hit & Run transition probability is critical for convergence
- Coordinate Sampling is not guaranteed to converge
- Asymptotically equivalent to uniformly sampling and then performing Accept-Reject
- Asymptotically produces uniform samples in higher level set
- Number of required **accepted samples** is linear in dimension

# Non-Uniform Sampling

- We look at the following PDF:

$$g_T(x) = \frac{e^{-f(x)/T}}{\int_S e^{-f(z)/T} dz}$$

- We would rather sample according to this function instead of uniformly
- $T$ is the "Temperature" parameter – (just an arbitrary name for now)
- $\int_S e^{-f(z)/T} dz$ is a normalization constant, usually impossible to calculate
- We look at the following expectation values:

$$\overset{*}{x}_T = E_{g_T}[x] = \int_S x \cdot g_T(x) dx$$

$$\overset{*}{f}_T = E_{g_T}[f(x)] = \int_S f(x) \cdot g_T(x) dx$$

# Mathematical Properties

- For high temperatures:

$$\lim_{T\to\infty} f_T^* = \frac{1}{|S|}\int_S f(x)\,dx$$

- For low temperatures:

$$\lim_{T\to 0^+} f_T^* = \max_{x\in S}\left(f(x)\right) = f^*$$

- As the temperature grow the expected cost grows:

$$\frac{\partial}{\partial T} f_T^* = \frac{\partial}{\partial T} E_{g_T}\left[f(x)\right] = \frac{\mathrm{var}_{g_T}\left[f(x)\right]}{T^2}$$

# Mathematical Properties

- For high temperatures:

$$\lim_{T\to\infty} \mathrm{var}_{g_T}\left[ f\left(x\right)\right] = \frac{1}{|S|}\int_S \left( f\left(x\right) - \lim_{T\to\infty} f^*_T \right)^2 dx$$

- For low temperatures:

$$\lim_{T\to 0^+} \mathrm{var}_{g_T}\left[ f\left(x\right)\right] = 0$$

- We conclude that if we could indeed sample from $g_T\left(x\right)$ for $T \to +0$ we could approximate with $f^*$ arbitrary accuracy and confidence
- How can we sample from it? Luckily we learned just that in class…

# Metropolis-Hastings Algorithm

- We assume we already have a proposed Markov chain which we can easily sample from with transition probability function:

$$p(y \mid x)$$

- In general the reversibility principle for this chain does not hold:

$$p(y \mid x) g_T(x) \neq p(x \mid y) g_T(y)$$

- We wish to "correct" the transition probability with an additional factor:

$$\beta_T(y \mid x)$$

Such that:

$$p(y \mid x) \beta_T(y \mid x) g_T(x) = p(x \mid y) \beta_T(x \mid y) g_T(y)$$

# Metropolis-Hastings Algorithm

- Metropolis-Hastings gives the best $\beta_T(y \mid x)$ such in the sense that the most transitions are accepted (most efficient)
- If proposal transition are symmetrical we get:

$$\beta_T(y \mid x) = \min\left(1, g_T(y) / g_T(x)\right)$$

- Moves that reduce the reduce cost are always accepted but ones that increase cost are also accepted with some probability depending on the gap and the temperature.
- Direction and step size generation produce a candidate point
- We then need to choose if to accept candidate or not according to the M-H criteria
- If point is accepted we move to it, if not we stay in place

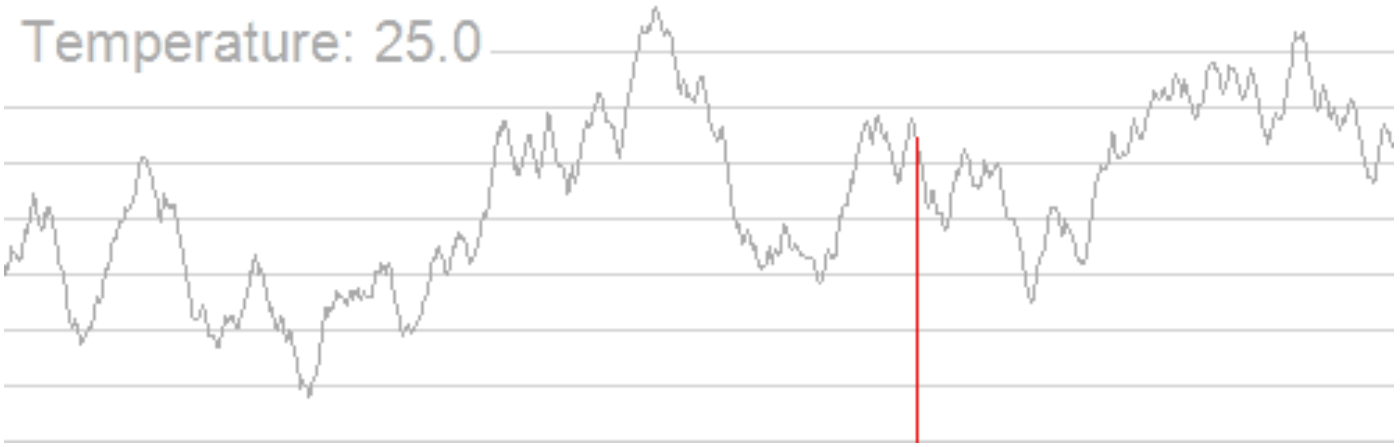
# Hide & Seek

- We assume that we have a starting point in the feasible set
- Choose direction randomly
- Find valid segment (intersect with linear bounding constraints)
- Generate a point uniformly on segment
- Perform 1D Accept-Reject to fulfill additional non-linear constraints
- Perform  another Accept-Reject according to M-H (gap & temperature dependent)

# The Effect of Temperature

- At high temperatures all transitions are accepted – Hit & Run
- Asymptotically converges to uniform distribution over $S$
- Cost function has ono effect on random walk, only affected by constraints
- At low temperatures only improvements are accepted – Improving Hit & Run
- Since step probability is local the chance of exiting a local minima and finding a new one is low (eventually will happen)
- Neither extreme is ideal, Hide & Seek  Allows for a gradual transition
- A cooling schedule specifies a decrement function for lowering temperature
- Typically at high temperatures the gross structure of the design emerges which is then refined at lower temperatures

# 1D Example



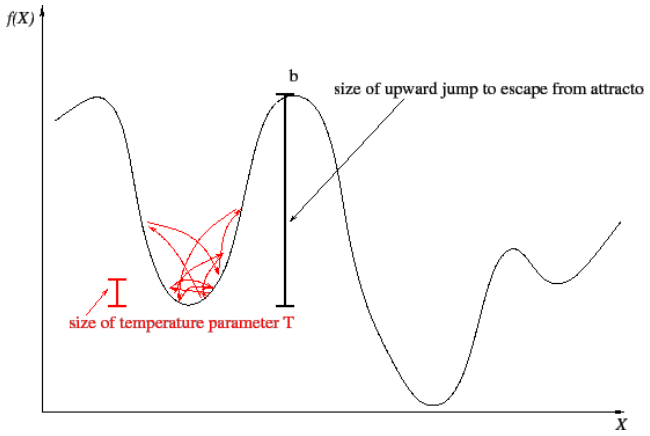

# Physical Annealing

In metallurgy and materials science, is a heat treatment that involves heating a material to above its recrystallization temperature, maintaining a suitable temperature, and then cooling in a gradual manner.

In annealing, atoms migrate in the crystal lattice and the number of dislocations decreases, leading to the change in ductility and hardness.

The controlled of metal allows atoms to detach from locally optimal structures and later realign to an ordered crystalline (globally) lower energy state.

Amazingly we get to an analogues result in a completely Monte-Carlo oriented approach



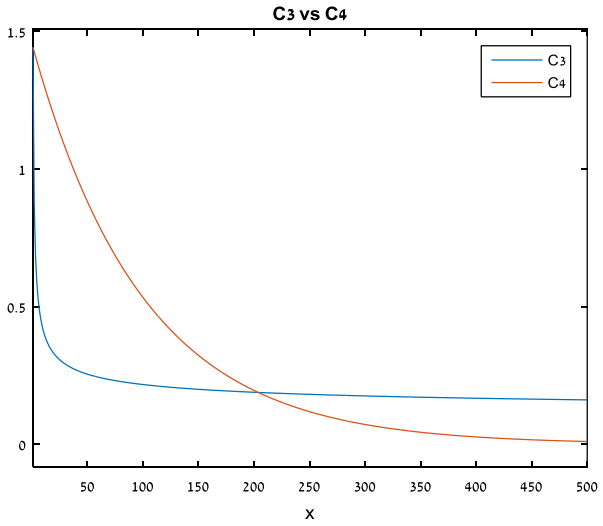

# Universal Cooling Schedules

- C5 – Temperature is zero (Corresponds to Improving Hit & Run)
- C4 – Geometric cooling:

$$T(k) = T_0 \cdot \alpha^k \qquad 0 < \alpha < 1$$

- C3 –Inverse-Logarithmic:

$$T(k) = \frac{T_0}{\ln(k+1)}$$




# Adaptive Cooling

- Temperature needs to be adapted to function values
- No schedule is universally good
- At high temperatures minima do not attract
- When lowering temperature minima start to gradually attract
- At freezing temperature all minim are attractive
- We would like the global minima to attract more then others
- Although some heuristics exist it is still a bit of an art form

# Adaptive Schedules

- Take second order approximation of the function
- Set temperature in such a way that the limit distribution concentrates mainly but no solely on function values superior to the current value
- We would like 99% of the probability to be at higher function values
- After change of variables can be shown to be equivalent to the probability:

$$\Pr\{\|U\|^2 \leq 2(f^* - f(x))/T\}$$

- We use the well known Chi-Squared distribution values and set:

$$T(k) = \frac{2\left(f^* - f(x)\right)}{\chi^2_{1-p}(d)}$$

# Cooling Schedules

- C1 – Theoretical adaptive schedule

$$T(k) = \frac{2\left(f^* - f(x)\right)}{\chi^2_{1-p}(d)}$$

- C2- Practical adaptive schedule

$$T(k) = \frac{2\left(\hat{f}^* - f(x)\right)}{\chi^2_{1-p}(d)}$$

$$\hat{f}^*(x_0, x_1, \ldots x_k) = (x_0, x_1, \ldots x_k)_{(k)} + \frac{\left(f(x_0), f(x_1), \ldots f(x_k)\right)_{(k)} - \left(f(x_0), f(x_1), \ldots f(x_k)\right)_{(k-1)}}{const}$$

# Finding a Starting Point

- Run Hide & Seek only constrained by box
- Objective function reflects constraint violation

$$\max_{x \in S_1} \left( \sum_{j=1}^{m_L} \min(0, b_j - a_j' x) + \sum_{j=1}^{m_{NL}} \min(0, -c_j(x)) \right)$$

- All feasible space is a global minimum of cost 0
- Algorithm can stop immediately when a feasible point is found
- Cost function is not differentiable but this is not required anyway
- Easy to generate a uniform starting point in box

# Results

1. Objective function:

$$f(x) = -\frac{1}{x_1^2 x_2 x_3}$$

Feasible region:

$$0.44098x_1 + 28.46x_1^2 + 6158.4x_1^2x_2 + 0.0037018x_3 + 5.4474x_3^2 + 0.032236x_1x_3 + 2.92x_2x_3 + 0.44712x_2 + 37.964x_2^2 + 42.876x_1x_2 - 1 \leqslant 0$$

$$0 \leqslant x_1 \leqslant 0.18745$$

$$0 \leqslant x_2 \leqslant 0.16230$$

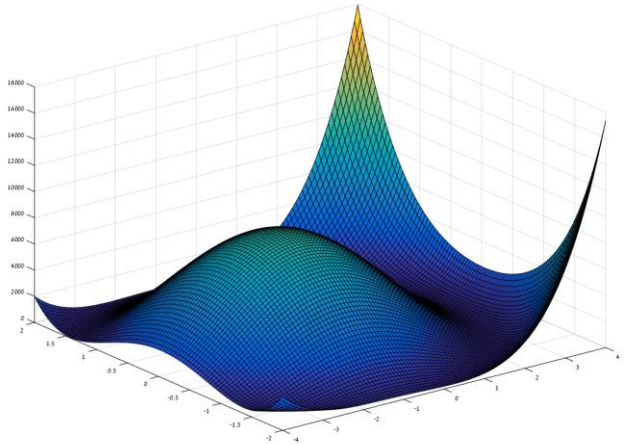
$$0 \leqslant x_3 \leqslant 0.42846$$

| p | f.e. | c.e. | new | rec | time | best | |
|---|---|---|---|---|---|---|---|
| 1 | 0.0 | 10.1 | 4.3 | 3.5 | 0.10 | | |
| | 530.6 | 1,866.2 | 24.9 | 14.8 | 9.87 | | |
| | 530.6 | 1,876.3 | 29.2 | 18.3 | 9.97 | C1 | D3 |

| | f.e | c.e | time | best | |
|---|---|---|---|---|---|
| start | 0 | 11.08 | | | |
| opt | 579.74 | 0 | | | |
| total | 579.74 | 11.08 | 0.179 | C5 | D3 |

| | D1 | D2 | D3 |
|---|---|---|---|
| C1 | 0.3364 | 0.2013 | 0.2036 |
| C2 | 0.4654 | 0.3417 | 0.2795 |
| C3 | 0.3229 | 0.2018 | 0.1825 |
| C4 | 0.3194 | 0.2208 | 0.2183 |
| C5 | 0.3148 | 0.233 | 0.179 |

# Results



13. Objective function:

$$f(x) = -(9x_1^2 + 36x_1x_2 + 52x_2^2 + 30(x_1^2 + 4x_2^2 + 2x_1 - 1$$

Feasible region:

$$-4 \leqslant x_1 \leqslant 4$$

$$2 \leqslant x_2 \leqslant 2$$

| | f.e. | new | rec | time | best | |
|---|---|---|---|---|---|---|
| 13 | 547.4 | 29.0 | 13.0 | 4.69 | C1 | D2 |

| | f.e | c.e | time | best | |
|---|---|---|---|---|---|
| start | 0 | 0 | | | |
| opt | 583.1 | 0 | | | |
| total | 583.1 | 0 | 0.0602 | C4 | D3 |

| | D1 | D2 | D3 |
|---|---|---|---|
| C1 | 0.933 | 0.1227 | 0.0796 |
| C2 | 0.1449 | 0.1002 | 0.1301 |
| C3 | 0.0879 | 0.0721 | 0.0764 |
| C4 | 0.0834 | 0.0782 | 0.0602 |
| C5 | 0.0658 | 0.0706 | 0.0693 |

# Results

6. Objective function:

$$f(x) = -2x_1 - 6x_2 + x_1^3 + 8x_2^2$$

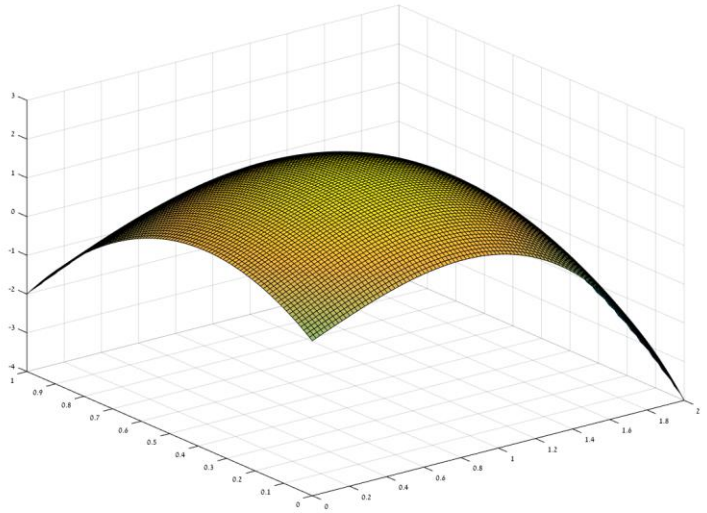
Feasible region:

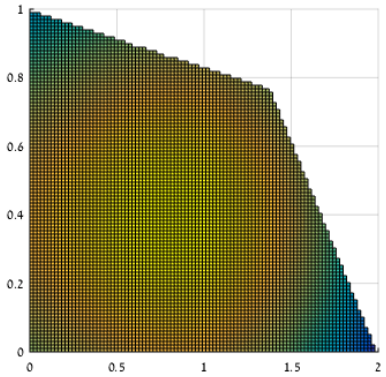
$$x_1 + 6x_2 \leqslant 6$$

$$5x_1 + 4x_2 \leqslant 10$$

$$0 \leqslant x_1 \leqslant 2$$

$$0 \leqslant x_2 \leqslant 1$$





| | f.e. | c.e. | new | rec | time | best | |
|---|---|---|---|---|---|---|---|
| 6 | 0.0 | 3.6 | 0.5 | 0.5 | 0.02 | | |
| | 174.4 | 0.0 | 36.4 | 36.4 | 2.05 | | |
| | 174.4 | 3.6 | 36.9 | 36.9 | 2.07 | C5 | D1 |

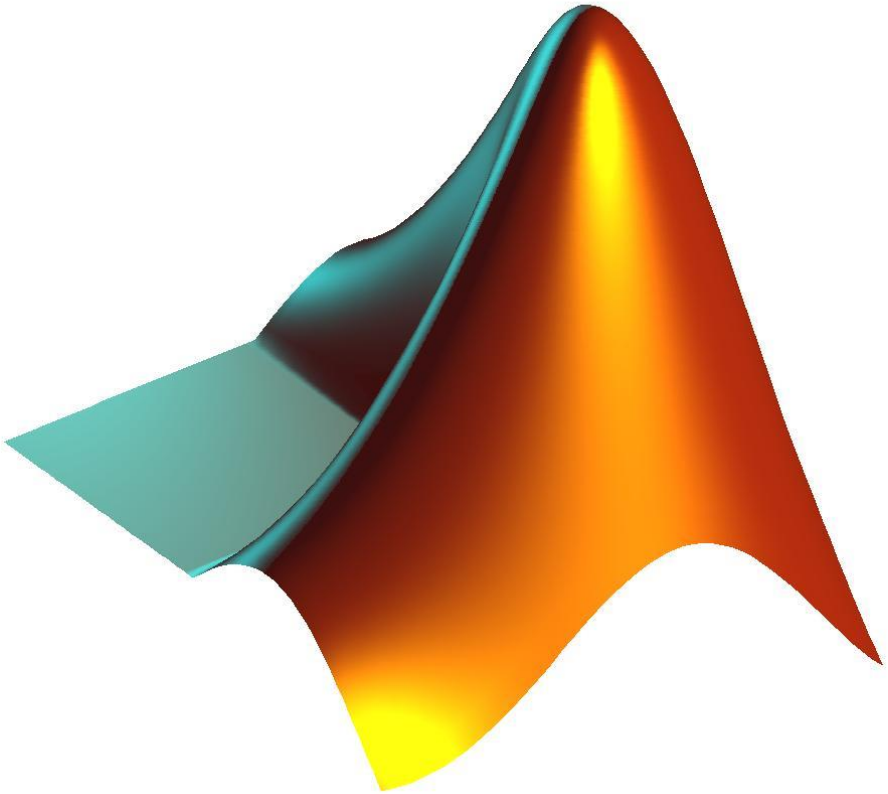
| | f.e | c.e | time | best | |
|---|---|---|---|---|---|
| start | 0 | 1.64 | | | |
| opt | 284.84 | 0 | | | |
| total | 284.84 | 1.64 | 0.0296 | C5 | D1 |

| | D1 | D2 | D3 |
|---|---|---|---|
| C1 | 0.0303 | 0.0353 | 0.0303 |
| C2 | 0.0925 | 0.1105 | 0.0993 |
| C3 | 0.0556 | 0.0591 | 0.0608 |
| C4 | 0.0938 | 0.0807 | 0.0932 |
| C5 | 0.0296 | 0.0338 | 0.036 |

# Results

- CPU time till convergence is the best criteria for efficiency
- Combines number of iterations and function/constraint evaluations
- D2 & D3 outperform D1 almost always
- C1 performs well in most cases but its approximation C2 does not perform very poorly
- C3 & C4 performs poorly as well
- C5 performs remarkably well and in many cases outperforms C1, especially in low dimensional cases

# Visualization



# Discussion & Conclusion

- Although visualized for 2D the method is mainly suited for higher dimensions.
- For low dimensions Improving Hit & Run works very well and even better
- Results in paper are not statistically significant, seems to be intentional. Not enough samples were made and results are presented in a unclear manner
- The paper proposes a theoretical cooling schedule which performs similarly to IH&R, the practical approximation for it performs poorly so adaptive method presented is useless
- The need for a cooling schedule is not evident from this paper
- Practical use cases require bounding in a “box” which might be hard to do
- Annealing is useful for low noise or significant global minima not good if local minima are similar
- Convergence speed affected by ability to tightly bound by linear constraints due to A-R on segment

# Questions

# References


1. Aarts, E. H. L. and P. J. M. van Laarhoven (1989), Simulated annealing: an introduction, Statistica Neerlandica 43, 31-52.
2. H.E. Romeijn, R.L. Smith, Simulated annealing for constrained global optimization, J. Global Optim., 5 (1994), pp. 101–126
3. Pincus, M. (1968). A Closed Form Solution of Certain Dynamic Programming Problems. Operations Research, 16, 690-694.
4. Romeijn, H.E. and R. L. Smith (1990), Sampling through random walks. Technical Report 90-02. Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI.
5. Rubinstein, R. Y. (1981), Simulation and the Monte Carlo Method, Wiley, New York, NY.
6. Zabinsky, Z. B., R. L. Smith, J. F. McDonald, H. E. Romeijn, and D. E. Kaufman (1993), Improving Hit-and-Run for global optimization, Journal of Global Optimization 3, 171-192.
7. http://www.sfu.ca/~ssurjano