Probabilistic Inverse Theory

Lecture 10

W. Dębski

31.05.2023

Exploring *a posteriori* probability

- searching for maximum of $\sigma(\mathbf{m})$
- marginal distributions (sampling) $\sigma(\mathbf{m})$

Efficient methods of calculation multi-dimensional integrals needed !

Marginal *a posteriori* distribution/Sampling

- 1D marginals

$$\sigma_i(m_i) = \int_{\mathbf{m} \neq m_i} \sigma(\mathbf{m}) \, \mathrm{d}\mathbf{m}$$

- 2D marginals

$$\sigma_{ij}(m_i, m_j) = \int_{\mathbf{m} \neq m_i, m_j} \sigma(\mathbf{m}) \, \mathrm{d}\mathbf{m}$$

- higher dimension marginals

Sampling *a posteriori* distribution

- geometric sampling - grid search
- adaptive grid search (near neighborhood algorithm)
- stochastic (Monte Carlo sampling)

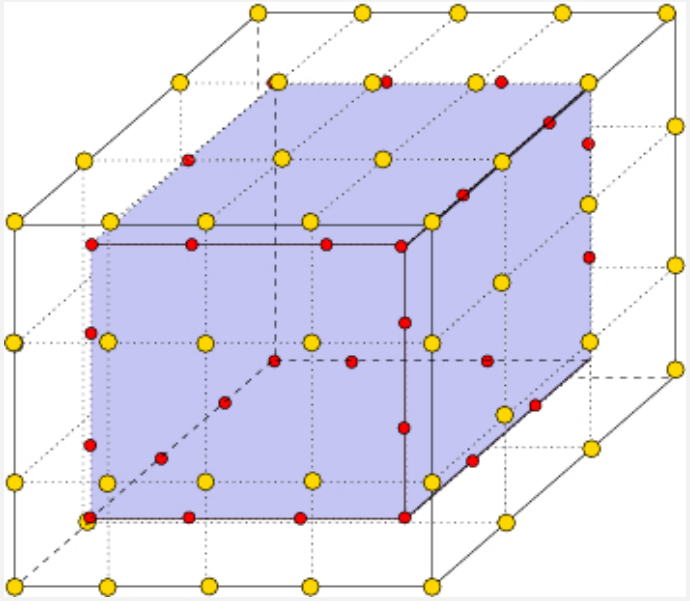
$$\sigma(\mathbf{m}) \Rightarrow \sigma_{i,j,k,\ldots} = \sigma(m_i, m_j, m_k, \ldots)$$

- very general
- only for small dimensional problem
- non-uniform sampling ...

Geometric sampling in multi-dimensional spaces

Non-uniform sampling:

$$\frac{N_V}{N} = \left(\frac{p-2}{p}\right)^N \overset{p \gg 2}{\approx} e^{-2N/p} \xrightarrow{N} 0$$

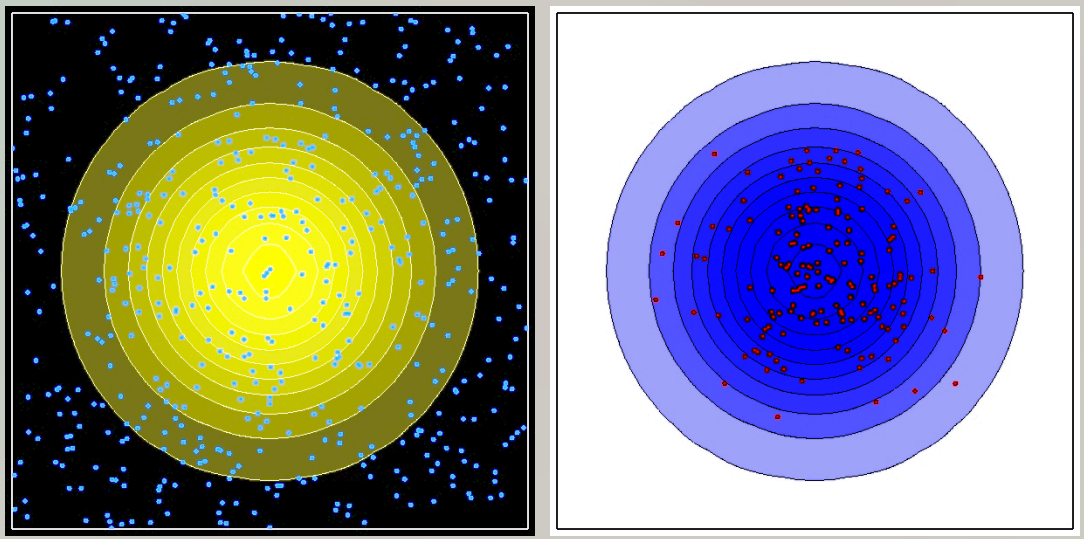


Regular grid sampling - curse of dimensionality problem

N/p	10	100	1000
2	0.64	0.96	0.996
3	0.51	0.94	0.994
5	0.33	0.90	0.990
10	0.10	0.82	0.980
100	$2.0\,10^{-10}$	0.13	0.818
1000	$1.2\,10^{-97}$	$1.6\,10^{-11}$	0.135
10000	—	$1.8\,10^{-88}$	$2\,10^{-9}$

Sampling *a posteriori* distribution

- geometric sampling - grid search
- adaptive grid search (near neighborhood algorithm)
- stochastic (Monte Carlo sampling)



Integral evaluation - homogeneous grid

$$I = \int_{\mathcal{M}} f(\mathbf{m})\sigma(\mathbf{m})\mathrm{d}\mathbf{m}$$

geometric sampling

$$\sigma_{i,j,k,\ldots} = \sigma(m_i, m_j, m_k, \ldots)$$

$$I \sim \Delta \sum_{i,j,k,\ldots} f_{i,j,k,\ldots} \cdot \sigma_{i,j,k,\ldots}$$

Integral evaluation - importance sampling

$$I = \int_{\mathcal{M}} f(\mathbf{m})\sigma(\mathbf{m})\mathrm{d}\mathbf{m}$$

Grid points genereated such that

$$N(\mathbf{m} \pm \delta\mathbf{h}) \approx N_{tot}\ \sigma(\mathbf{m})$$

importance sampling

$$I \sim \Delta(\delta\mathbf{h}) \sum_i f_i$$

Importance sampling - how to do it ?

Direct generation of points from general $\sigma(\mathbf{m})$ is possible only in a few cases (Gausian distribution, Weibul distribution, etc.) and essentially **only** in 1D.

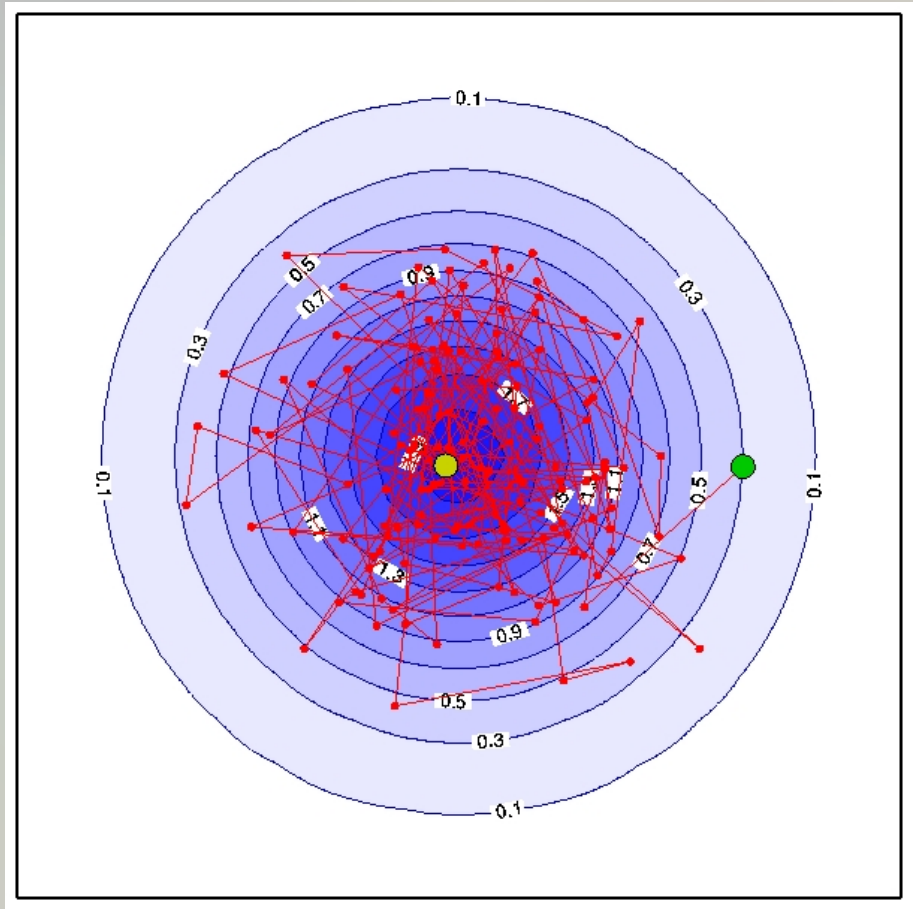
Thus, instead of the direct sampling we make a clever trick:
we "add " time t to our problem and consider a stochastic process

$$X(t) \in \mathcal{M}$$

constructed such a way that for a sufficient large t

$$P(X = \mathbf{m}) \approx \sigma(\mathbf{m})$$

Random walk over $\mathcal{M}$



$$(X_1, X_2, \cdots X_N)$$

$$I \sim \sum_i f(X_i)$$

For example::

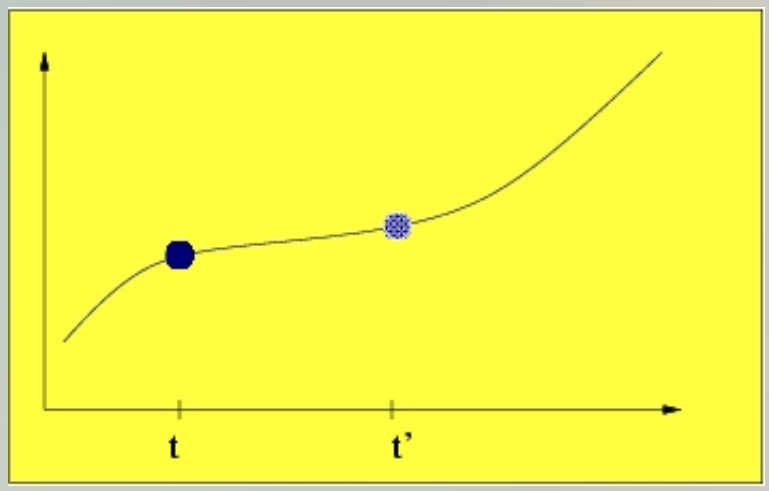
$$f(\mathbf{m}) = (\mathbf{m} - \bar{\mathbf{m}})^T \cdot (\mathbf{m} - \bar{\mathbf{m}})$$

$$\bar{\mathbf{m}} = \frac{1}{N} \sum_i X_i$$

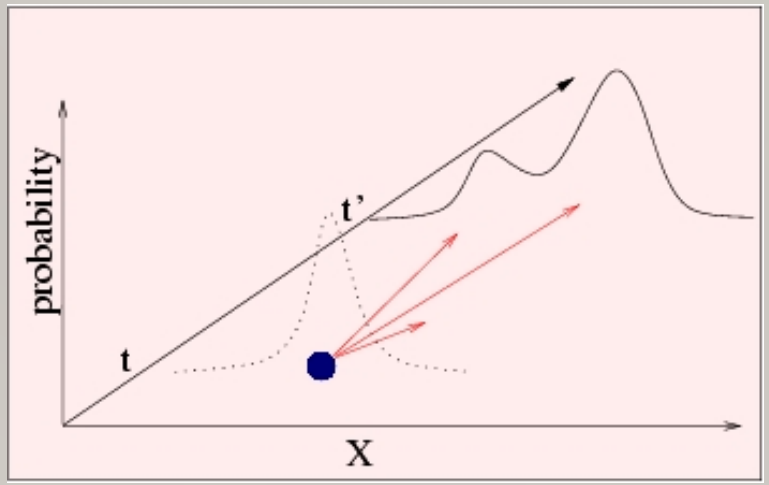
$$I \sim \sum_i (X_i - \bar{\mathbf{m}})^T \cdot (X_i - \bar{\mathbf{m}})$$

Stochastic processes (with short memory)

deterministic



stochastic



$$X(t+dt) = X(t) + f(X(t),t)dt \quad P(x',t') = \sum_x P(x,t)K(t',x';t,x)$$

Markow chains

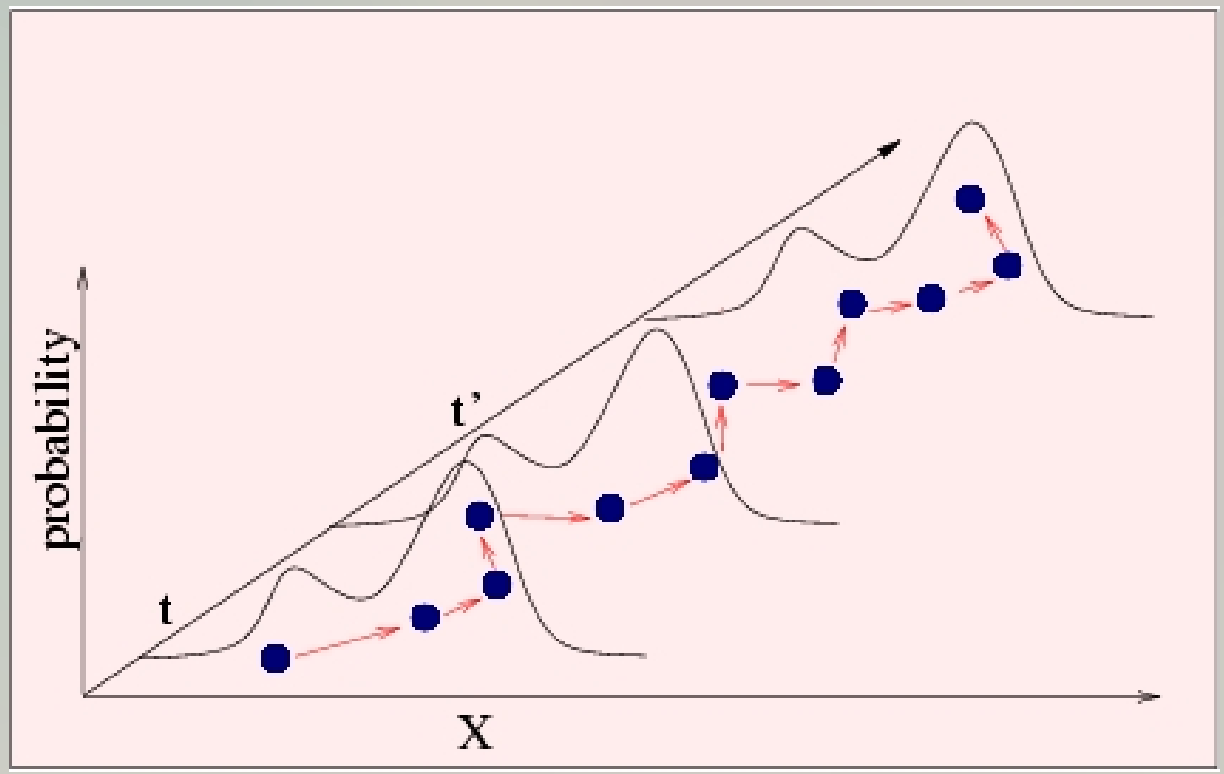
- discrete time: $t_1, t_2, \cdots$
- short memory $K(t_i, x; t_{i-1}, x_{i-1})$
- ergodic process:
  - ★ stationary $K(t, x; t', x') = K(x, x')$
  - ★ irreducible
  - ★ aperiodic
- Stationary probability distribution: $\pi(x') = \pi(x)$

$$\pi(x) = \int_{x'} \pi(x') K(x; x') dx'$$

Stationary Markov Chain

$$p(x_i) = \sum_k p(x_k)\, K(x_i; x_k)$$

$$p(x) = \int_{x'} \pi(x') K(x; x') dx'$$

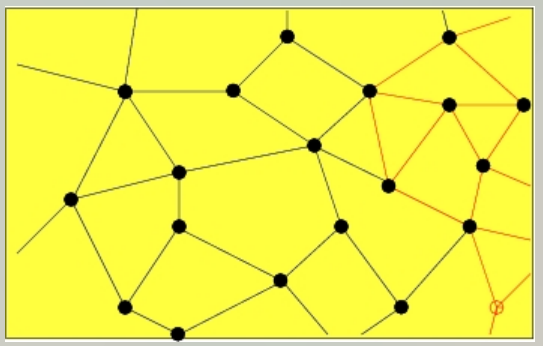
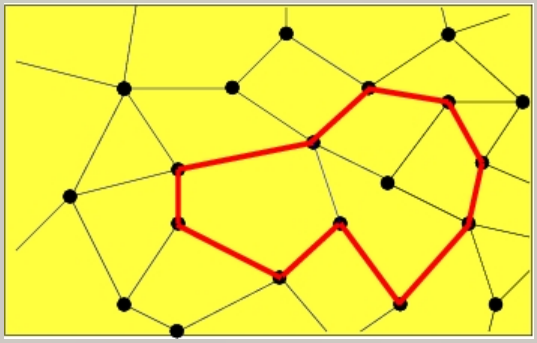
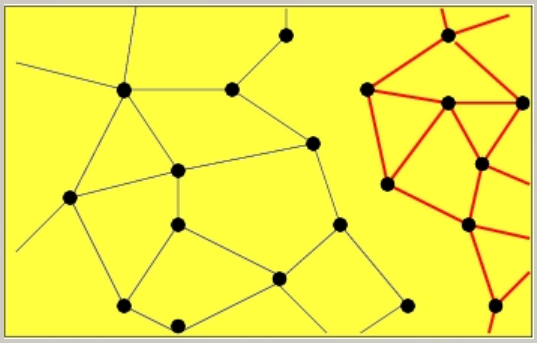


Markov chain - example

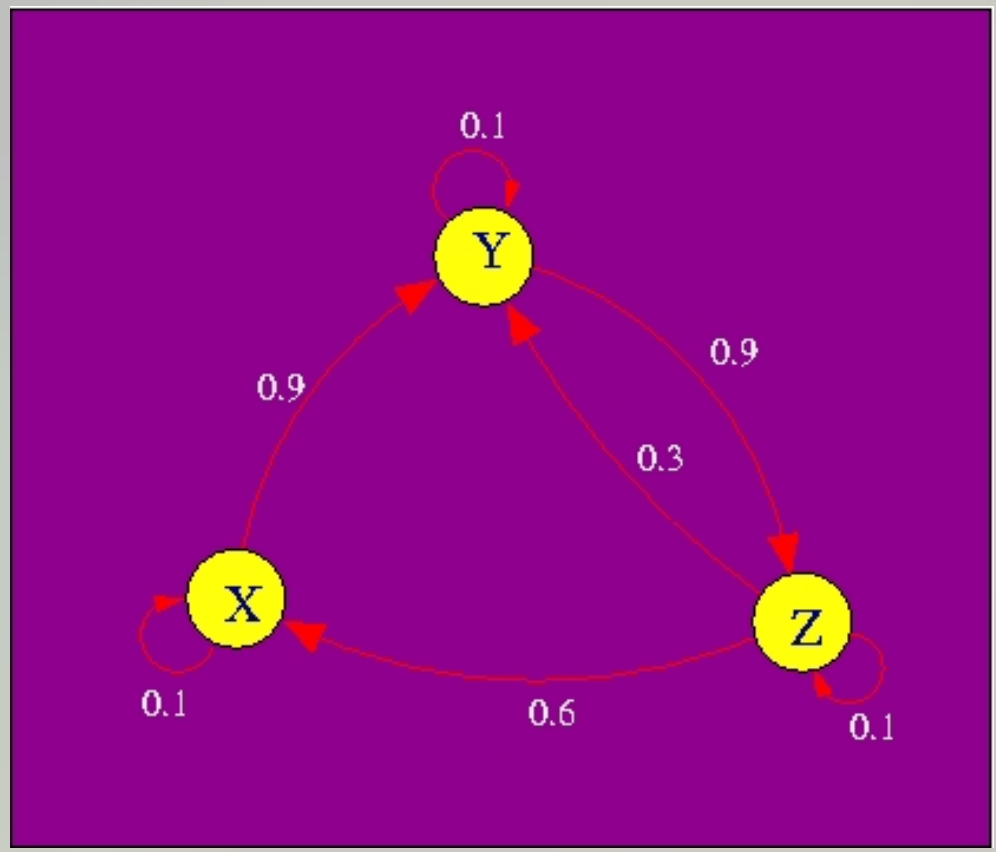
random walk

$$x_i \Longrightarrow x_{i+1} = x: \quad p_{x_i \to x_{i+1}} = K(x_{i+1}|x_i)$$

stationary aperiodic irrducible

Algorithm - example



Algorithm - example

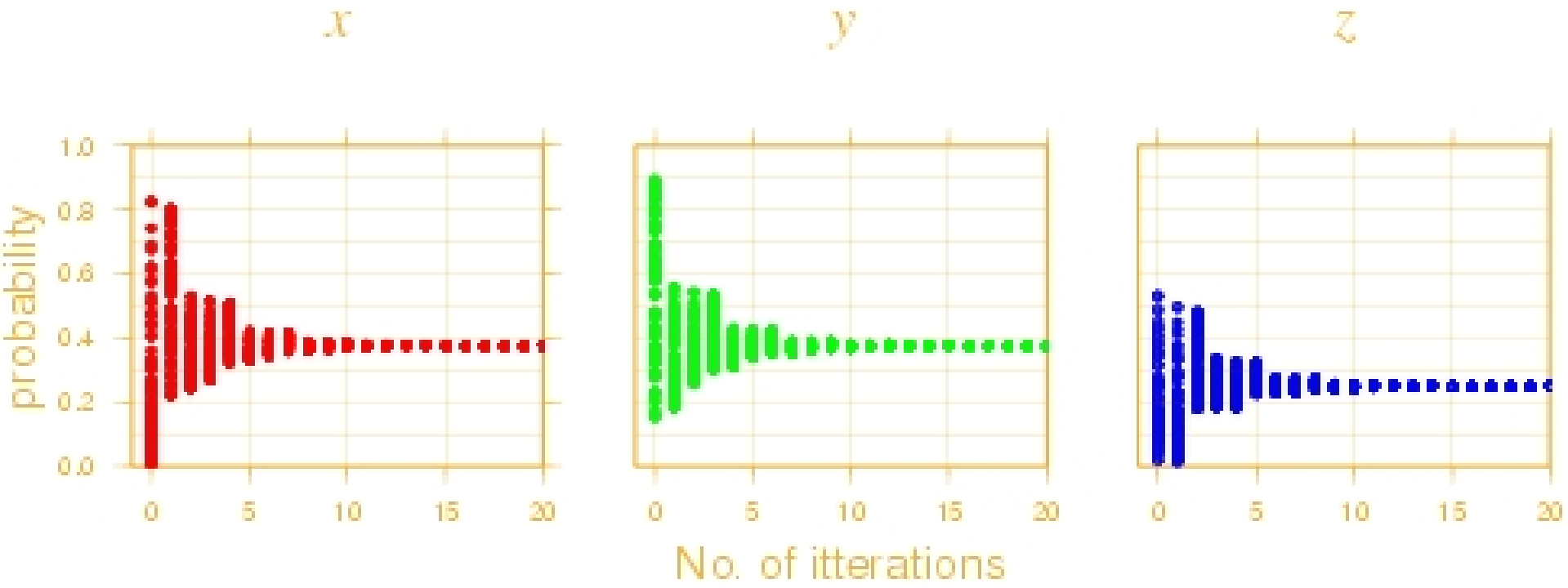
$$\mathbf{p}_i = (p_x, p_y, p_z)$$

Evolution equation

$$\mathbf{p}_{i+1} = \mathbf{p}_i \cdot \begin{pmatrix} 0.1 & 0.9 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.3 & 0.1 \end{pmatrix}$$

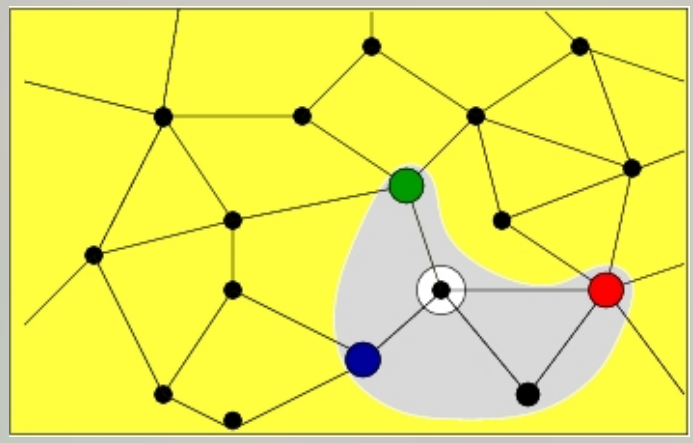
$\mathbf{p}_{i \to \infty} = ???$

Algorithm - example



Defining MC

1.



2.

$$K = \begin{pmatrix} K_{11} & \cdots & K_{1N} \\ K_{21} & \cdots & K_{2N} \\ \vdots & \vdots & \vdots \\ K_{M1} & \cdots & K_{MN} \end{pmatrix}$$

Metropolisa-Hastings (MH) algorithm

$$x_{i+1} = x_i + \delta x_i$$

$$K(x_i; x_{i+1}) = \min\left\{1, \frac{p(x_{i+1})}{p(x_i)}\right\}$$

Metropolis pseudo-code for sampling from $p(\mathbf{m})$

- Initialize sampling $\mathbf{m}^{\alpha} = \mathbf{m}^{0}$

★ generate test sample $\mathbf{m}^{\beta}$: $\mathbf{m}^{\beta} = \mathbf{m}^{\alpha} + \delta\mathbf{m}$
★ evaluate $\mathbf{m}^{\beta}$: $p_{\beta} = p(\mathbf{m}^{\beta}, T_k)$
★ create a new one $\mathbf{m}^{\alpha+1}$
➡ accept $\mathbf{m}^{\beta}$ with probability $p = min\,(1, p_{\beta}/p_{\alpha})$

$$\mathbf{m}^{\alpha+1} = \mathbf{m}^{\beta}$$

➡ if $\mathbf{m}^{\beta}$ rejected duplicate

$$\mathbf{m}^{\alpha+1} = \mathbf{m}^{\alpha}$$

- Repeat untill sufficient number of $\{\mathbf{m}^{i}\}$ is obtained

Metropolis Hasting algorithm

$$\sigma(\mathbf{m}) \;=\; f(\mathbf{m})L(\mathbf{m})$$

- Initialize $\mathbf{m}^0$
- Repeat
  - generate uniform random number $u \sim U(0,1)$
  - generate test sample $\mathbf{m}^\beta$ from $f(\mathbf{m})$
  - if $u < P(\mathbf{m}^\beta, \mathbf{m}^\alpha) \;=\; \min\left[1, \frac{L(\mathbf{m}^\beta)}{L(\mathbf{m}^\alpha)}\right]$

    $\mathbf{m}^{\alpha+1} = \mathbf{m}^\beta$

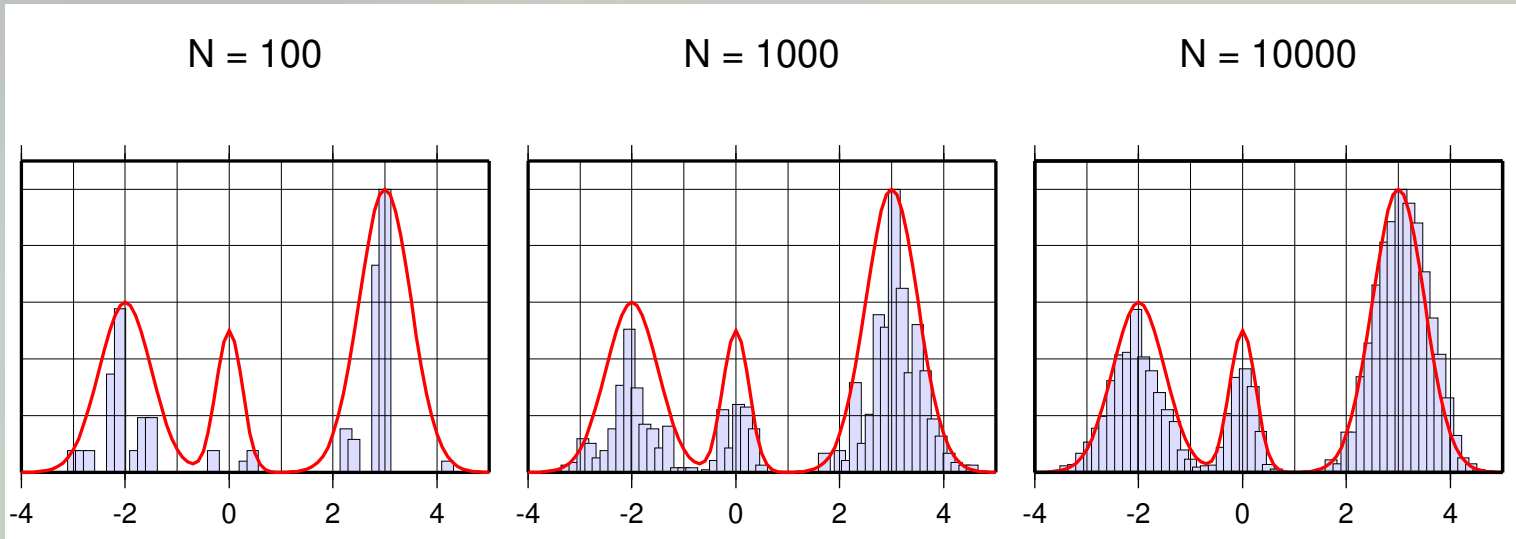
    otherwise

    $\mathbf{m}^{\alpha+1} = \mathbf{m}^\alpha$
- Continue until sufficient number of samples $\{\mathbf{m}^\alpha\}$ is generated

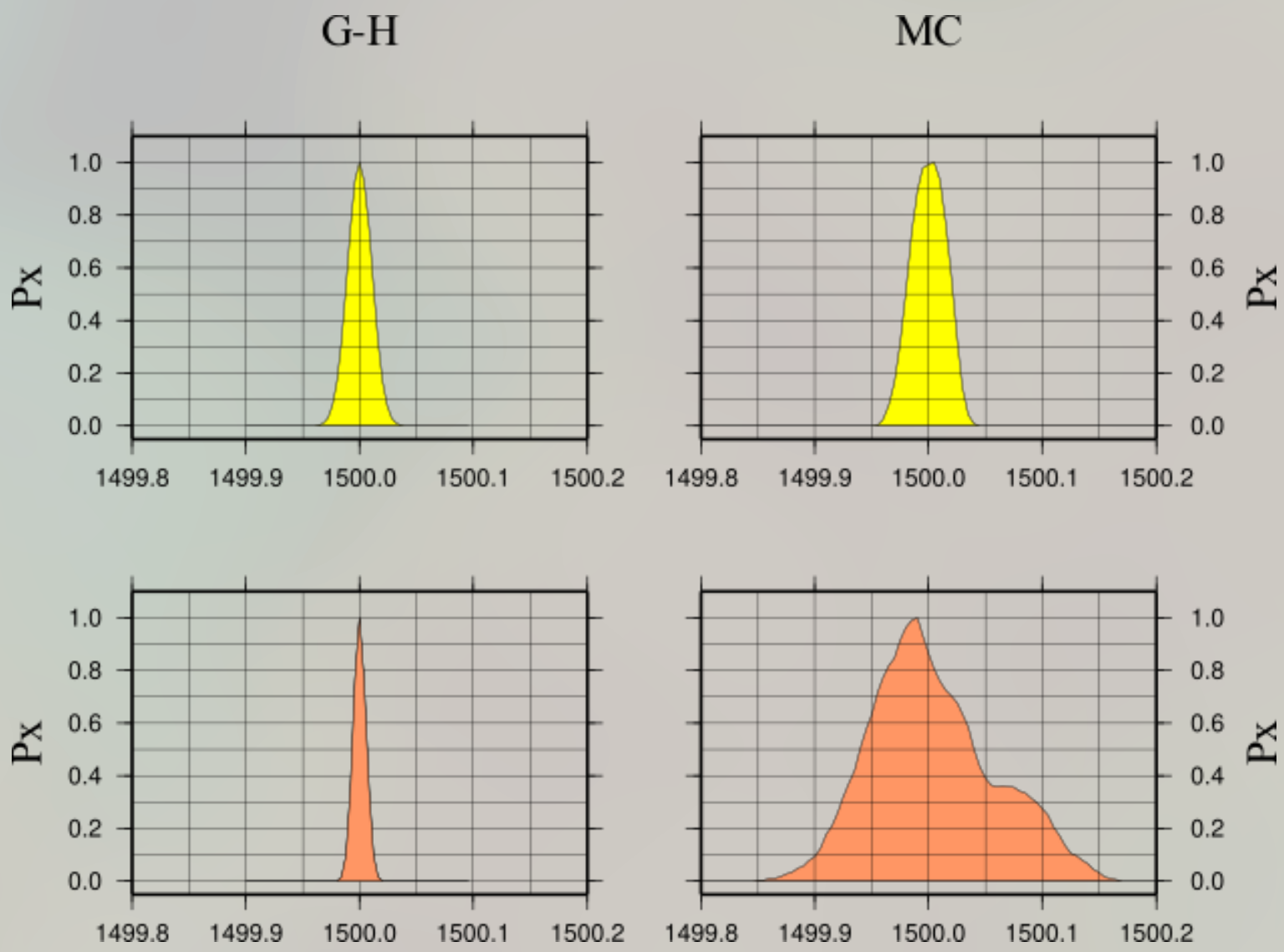
Metropolis Hasting algorithm -features

- after so called burn-in initial time generated samples follow $\sigma(\mathbf{m})$ probability distribution
- subsequent samples are strongly correlated - the chain must be run for a long time
- MH is optimum sampling algorithm if only $\sigma(\mathbf{m})$ is available.
- How many samples should be generated ?
- problem with generating "proper" test samples ($\mathbf{m}^{\beta}$)

MH algorithm - how many samples



MH algorithm - generating proposal samples

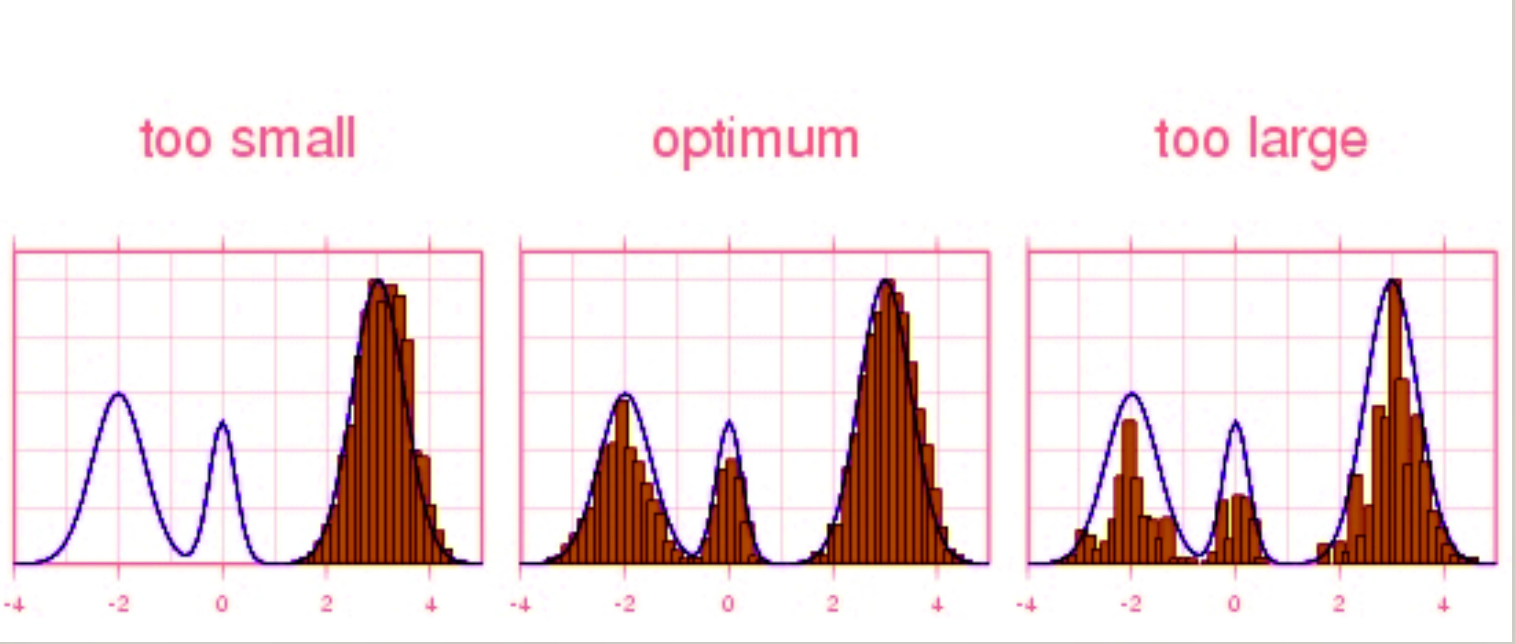


MH algorithm - generating proposal samples

"Narrow" $\sigma(\mathbf{m})$ requires small update of $\mathbf{m}^{\alpha}$

$$\mathbf{m}^{\beta} = \mathbf{m}^{\alpha} + \delta\mathbf{m}$$

but then we diminishing mixing property of the chain



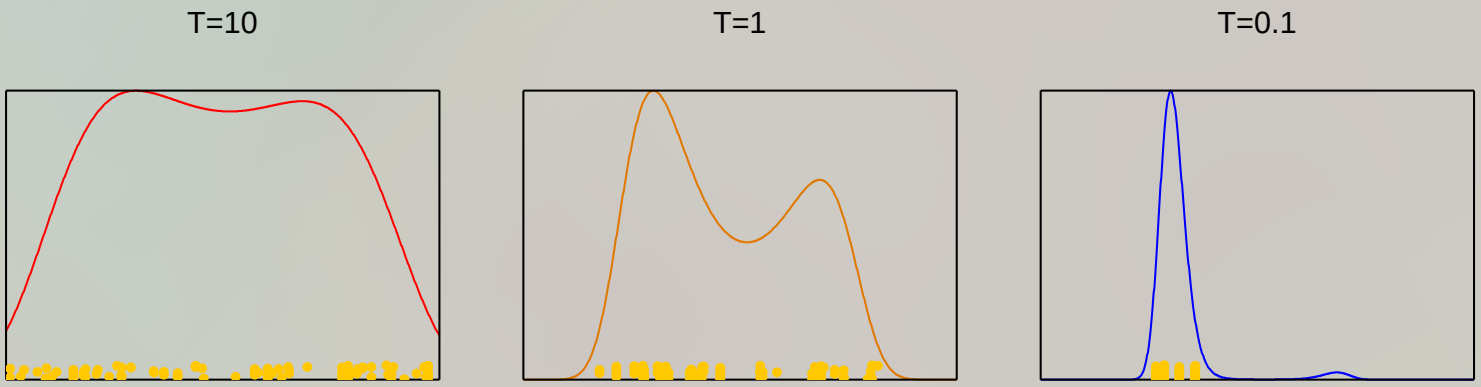
Non stationary MCMC

- discrete time: $t_1, t_2, \cdots$
- short memory $K(t_i, x; t_{i-1}, x_{i-1})$
- ergodic process:
  - ★ stationary $K(t, x; t', x') = K(x, x')$
  - ★ irreducible
  - ★ aperiodic

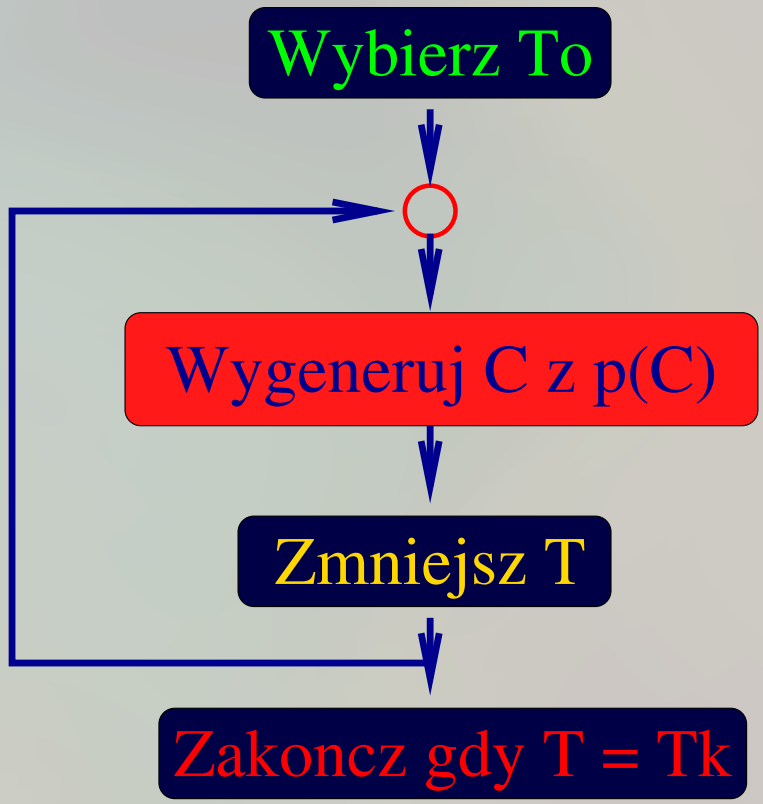
*** NO Stationary probability distribution:***

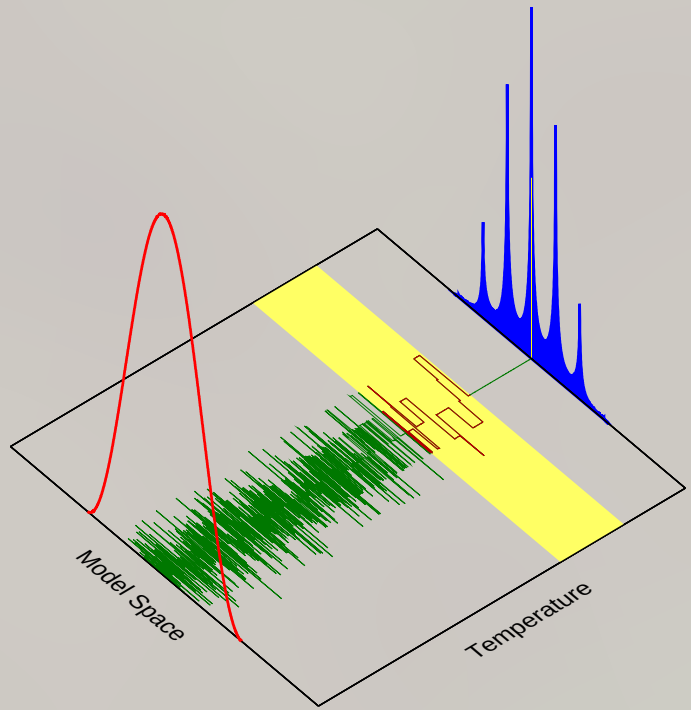
Non stationary MCMC - simulated annealing

$$p(\mathbf{m}) \;=\; \exp\left(-\frac{S(\mathbf{m})}{kT}\right) \qquad T = T(t)$$



Simulated annealing - an idea





SA - pseudo-code

- Create Boltzman-like distribution $p(\mathbf{m}, T) = \exp(-S(\mathbf{m})/T)$
- Set $T = T_o$

  - ★ generate test sample $\mathbf{m}^\beta$: $\mathbf{m}^\beta = \mathbf{m}^\alpha + \delta\mathbf{m}$
  - ★ evaluate it $\mathbf{m}^\beta$: $p_\beta = p(\mathbf{m}^\beta, T_k)$
  - ★ create a new one $\mathbf{m}^{\alpha+1}$
    - ➡ accept $\mathbf{m}^\beta$ with probability $p = min\,(1, p_\beta/p_\alpha)$

      $$\mathbf{m}^{\alpha+1} = \mathbf{m}^\beta$$

    - ➡ if $\mathbf{m}^\beta$ rejected duplicate

      $$\mathbf{m}^{\alpha+1} = \mathbf{m}^\alpha$$

- Keep the "best" $\mathbf{m}^\alpha$
- Decrease $T_k = \frac{T_0}{\ln(k)}$
- continue sampling untill $T_k > T_{final}$

Simulated annealing - elements

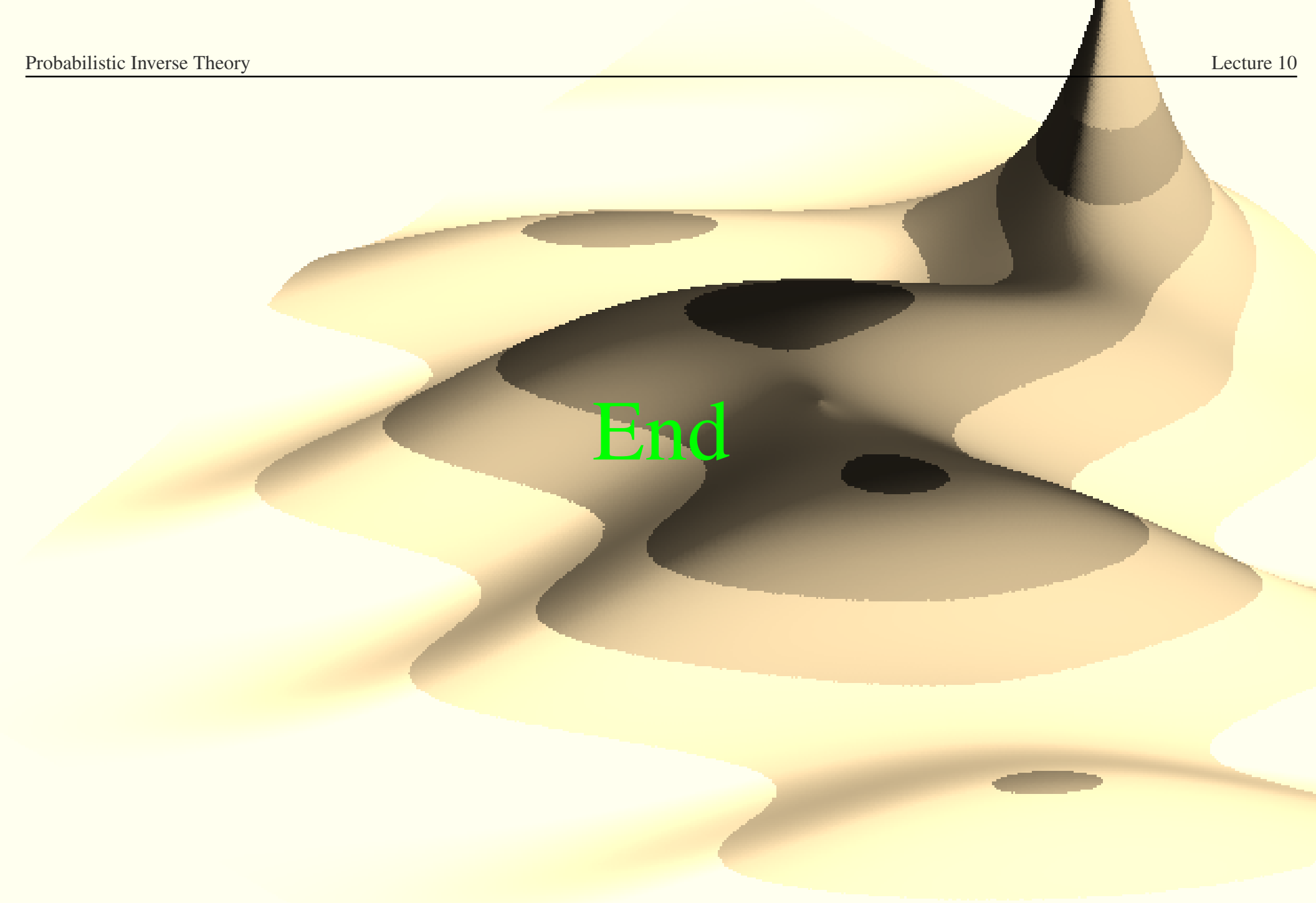
Two most important elements

1. generating sample from $p(\mathbf{m})$
2. schedule of T change

Efficient sampling - the Metropolis algorithm (generalized by Hastings)
Optimum cooling schedule (Kirkpatrick)

$$T_k = \frac{T_o}{\ln(k)}$$

Gobal Optimization Algorithm


End