# Statistical analysis of experimental data
## **Markov Chains**

Aleksander Filip Żarnecki

FACULTY OF PHYSICS
UNIVERSITY
OF WARSAW

**Lecture 14**

January 23, 2025

## Markov Chains

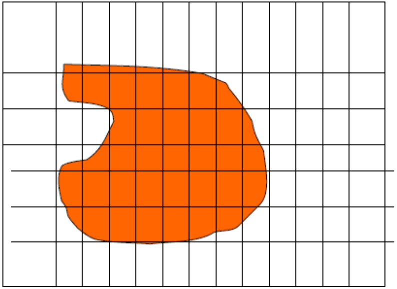
1. Markov Chains
2. Markov Chain Monte Carlo
3. Application to parameter fitting
4. Final exam

# Monte Carlo integration

## Applications

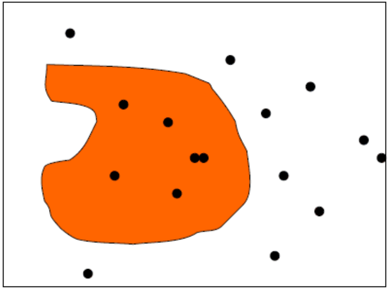
(lecture 05)

Described procedure can be used not only to calculate integrals of one-dimensional functions, it is much more general... How to calculate volume of a given shape?



Standard procedure:

scan all dimensions using dense point grid and sum cells with centers inside the volume



Monte Carlo integration:

Generate random points in the considered space and count points inside the volume

# Monte Carlo integration

## General case

Examples presented considered the special case: input random variables had uniform distribution and "test function" was binary (returning 0 or 1).

In the general case we want to determine an expectation value of a function $h(\mathbf{x})$ of random variable vector $\mathbf{x}$ described by $f(\mathbf{x})$ pdf:

$$\mu_h \equiv \mathbb{E}_f[h(\mathbf{x})] = \int d\mathbf{x}\; h(\mathbf{x})\; f(\mathbf{x})$$

Monte Carlo determination of $\mu_h$ assumes we can generate random variables from $f(\mathbf{x})$. We can then calculate:

$$\mu_{MC} = \lim_{N\to\infty} \frac{1}{N} \sum_i h(\mathbf{x}_i)$$

where $\mathbf{x}_i$, $i = 1, \ldots, N$ are random (input) variables generated from $f(\mathbf{x})$

## Weighted Monte Carlo

General method for generating random points in multi-dimensional space using acceptance–rejection technique can have very low efficiency, if probability distribution function $f(\mathbf{x})$ varies a lot, eg. has sharp peaks.

Assume we know how to generate random numbers from $g(x)$.
We can then apply the following procedure:

- generate $\mathbf{x}_i$ distributed according to $g(\mathbf{x})$
- accept all generated value $\mathbf{x}_i$,
  but consider them with additional weight: $w_i = f(\mathbf{x})/g(\mathbf{x})$

For example, when calculating the expectation value of $h(\mathbf{x})$:

$$\mu_{MC} \rightarrow \mu_{wMC} = \frac{\sum_i w_i \; h(\mathbf{x}_i)}{\sum_i w_i}$$

“unweighted” samples considered previously correspond to $w_i \equiv 1$

# Monte Carlo integration

## Weighted Monte Carlo

When using weighted Monte Carlo "events", number of events has to be replaced by sum of weights:

$$N \quad \rightarrow \quad N_w = \sum_i w_i$$

Variance of the sum of weights:

$$\mathbb{V}(N_w) \quad = \quad \sum_i w_i^2$$

Statistical power of the weighted Monte Carlo sample is equivalent to unweighted sample of:

$$N_{eq} \quad = \quad \frac{N_w^2}{\mathbb{V}(N_w)} \quad = \quad \frac{(\sum_i w_i)^2}{\sum_i w_i^2}$$

For Poisson distributed random number $\mathbb{V}(N) = N$

# Maximum Likelihood Method

## General problem

Presented above was a simple example of a more general problem: how to estimate parameters of the probability distribution function from the results of the experiment (measurements).

In many cases, parameter value can not be directly extracted from the measurement results.

In the general case, shape of the probability density function for measurement result **x**:

$$\mathbf{x} \quad = \quad (x_1, \ldots, x_n)$$

depends on a set of pdf parameters:

$$\boldsymbol{\lambda} \quad = \quad (\lambda_1, \ldots, \lambda_p)$$

so the probability density should be written as:

$$f(\mathbf{x}; \boldsymbol{\lambda})$$

# Maximum Likelihood Method

## Maximum Likelihood Method

The product:

$$L = \prod_{j=1}^{N} f(\mathbf{x}^{(j)};\boldsymbol{\lambda})$$

is called a likelihood function.

The most commonly used approach to parameter estimation is the maximum likelihood approach: as the best estimate of the parameter set $\boldsymbol{\lambda}$ we choose the parameter values for which the likelihood function has a (global) maximum.

Frequently used is also log-likelihood function

$$\ell = \ln L = \sum_{j=1}^{N} \ln f(\mathbf{x}^{(j)};\boldsymbol{\lambda})$$

we can look for maximum value of $\ell$ or minimum of $-2\,\ell = -2\ln L$

## Markov Chains

1. Markov Chains
2. Markov Chain Monte Carlo
3. Application to parameter fitting
4. Final exam

## General concept

(Bonamente)

Markov Chain is a stochastic process where we consider the sequence of measurements (random variables) $X^{(t)}$. Measurements at fixed time intervals are a frequent case...

Outcome of the measurement (also called "state" of the chain) has to belong to the defined "state space". It is our sample space...

However, the probability density for different states is not given a'priori! Instead, probability of the subsequent state (measurement at $t+1$) depends only on the current state of the system:

$$P(X^{(t+1)}) \quad = \quad P(X^{(t+1)}|x^{(t)})$$

## General concept

(Bonamente)

Markov Chain is a stochastic process where we consider the sequence of measurements (random variables) $X^{(t)}$. Measurements at fixed time intervals are a frequent case...

Outcome of the measurement (also called "state" of the chain) has to belong to the defined "state space". It is our sample space...

However, the probability density for different states is not given a'priori! Instead, probability of the subsequent state (measurement at $t+1$) depends only on the current state of the system:

$$P(X^{(t+1)}) \;=\; P(X^{(t+1)}|x^{(t)})$$

Probability can change in time, but it depends only on the current state of the chain, and not on any state of its earlier history!

This "short memory" property is known as the "Markovian property". As for particle decays!

## Simple chain example

Consider two boxes with a total of $N$ balls.

The state of the system can be defined by a number $n$ of balls which are placed in the first box, $0 \leq n \leq N$. The state space of the system has $N+1$ elements.

## Simple chain example

Consider two boxes with a total of $N$ balls.

The state of the system can be defined by a number $n$ of balls which are placed in the first box, $0 \leq n \leq N$. The state space of the system has $N+1$ elements.

The "random walk" chain can defined by the following procedure. At each step:

- select a box at random,
- move one ball from the selected box (if not empty) in the other one.

## Simple chain example

Consider two boxes with a total of $N$ balls.

The state of the system can be defined by a number $n$ of balls which are placed in the first box, $0 \leq n \leq N$. The state space of the system has $N+1$ elements.

The "random walk" chain can defined by the following procedure. At each step:

- select a box at random,
- move one ball from the selected box (if not empty) in the other one.

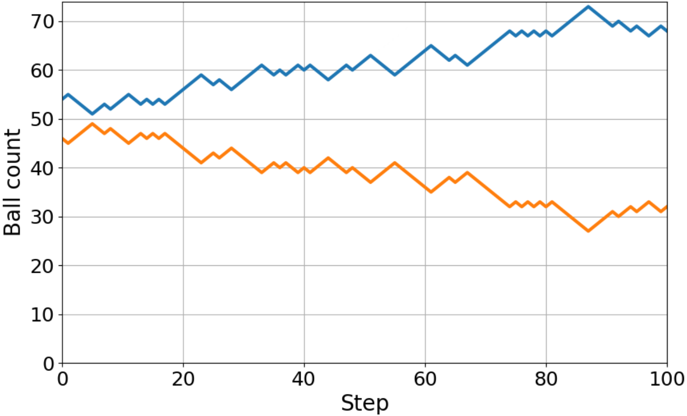
This chain can be presented in terms of the transition probabilities:

$$p(n^{(t+1)}) = \begin{cases} \frac{1}{2} \quad \text{for} & n^{(t+1)} = n^{(t)} \pm 1 \text{ and } n^{(t)} \neq 0 \text{ and } n^{(t)} \neq N \\ 1 & n^{(t+1)} = n^{(t)} \pm 1 \text{ and } (n^{(t)} = 0 \text{ or } n^{(t)} = N) \\ 0 & n^{(t+1)} \neq n^{(t)} \pm 1 \end{cases}$$

## Simple chain example

14_Simple.ipynb Open in Colab

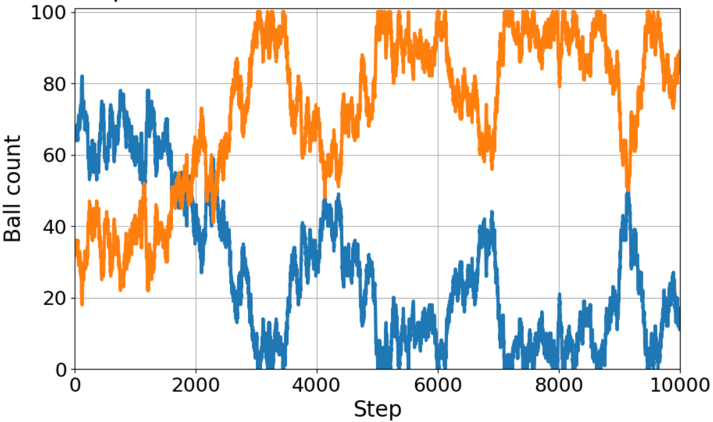
Result of the algorithm implementation



## Simple chain example

14_Simple.ipynb Open in Colab

Result of the algorithm implementation



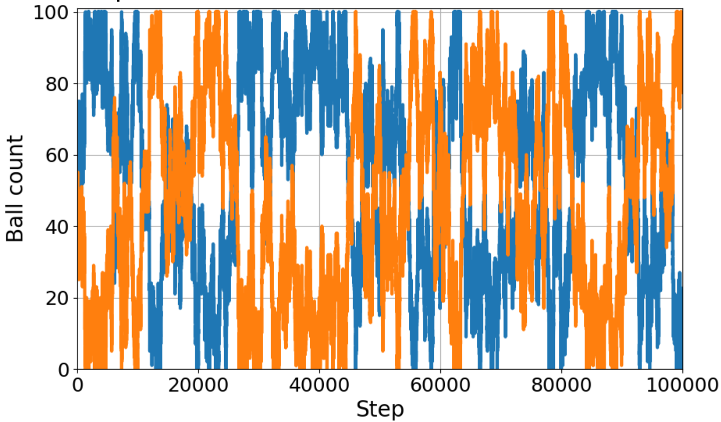
Looks like symmetry violation?...

## Simple chain example

14_Simple.ipynb

Open in Colab

Result of the algorithm implementation

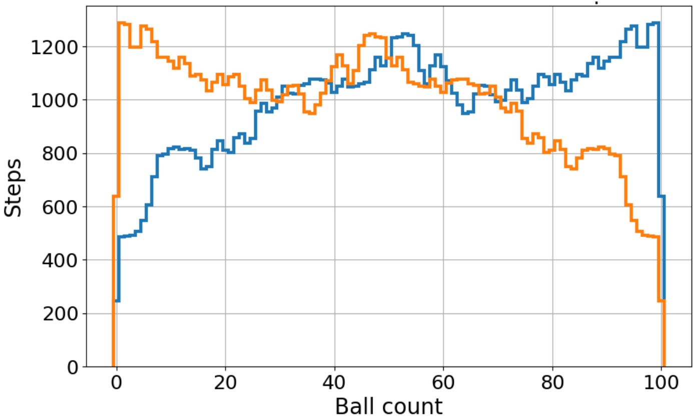


Large time scales for count fluctuations ⇒ symmetry restored on longer time scales

## Simple chain example

14_Simple.ipynb Open in Colab

Result of the algorithm implementation

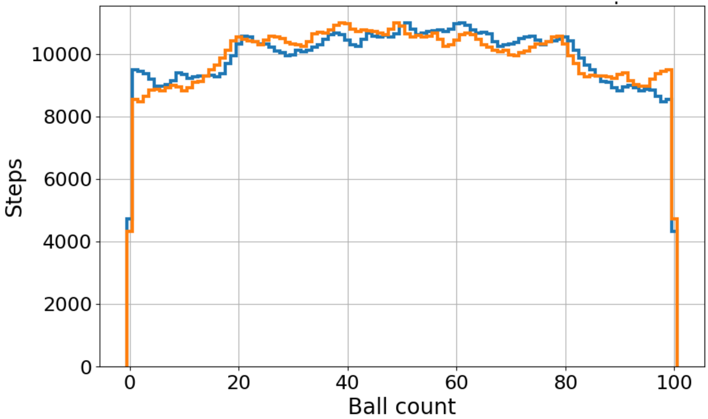


Fluctuations still visible in ball count distribution...

## Simple chain example

14_Simple.ipynb Open in Colab

Result of the algorithm implementation

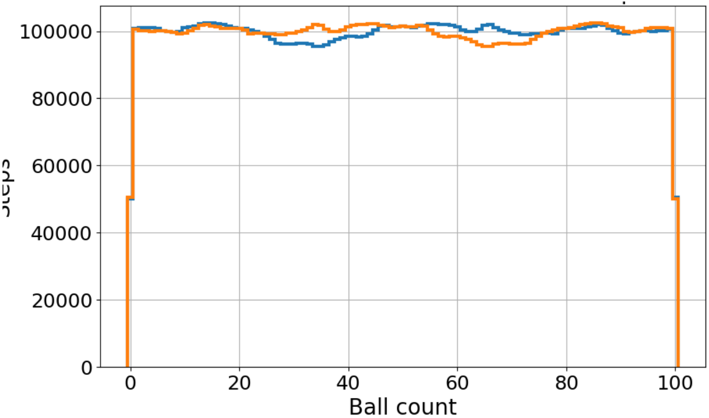


Decrease with the chain lenght...

## Simple chain example

14_Simple.ipynb Open in Colab

Result of the algorithm implementation

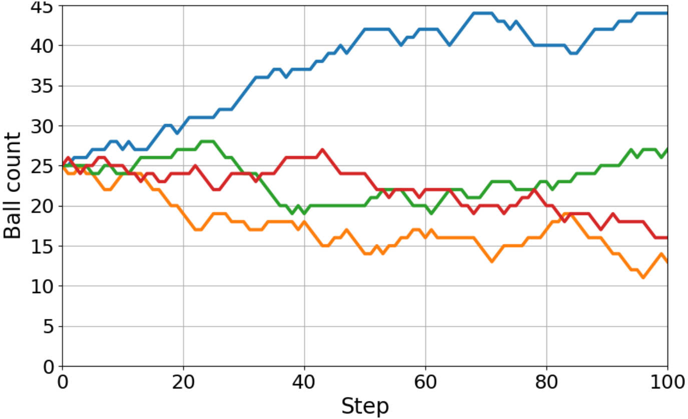


Decrease with the chain lenght...

## Simple chain example

14_Simple.ipynb Open in Colab

Result for the extended example (4 boxes)

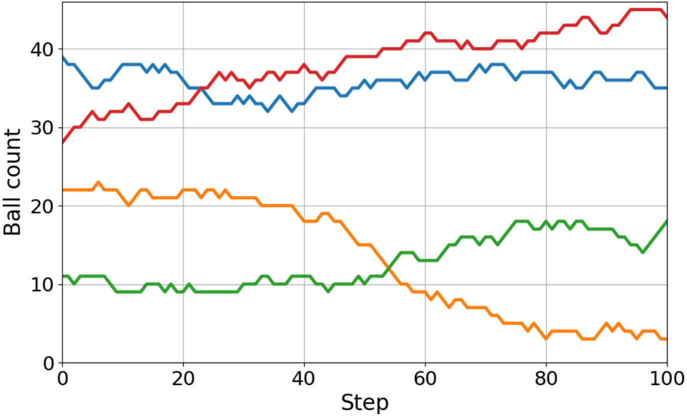


Even starting from even ball distribution, large fluctuations appear very soon...

## Simple chain example

14_Simple.ipynb Open in Colab

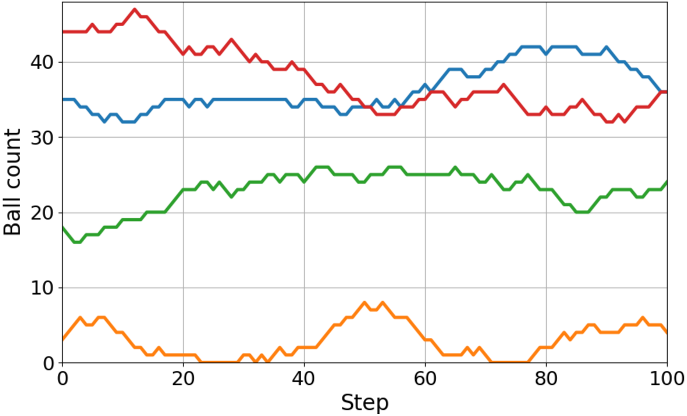
Result for the extended example (4 boxes)



## Simple chain example

14_Simple.ipynb Open in Colab

Result for the extended example (4 boxes)



# Markov Chains

## Ehrenfest chain

(Bonamente)

Simple model of diffusion: same case of two boxes with a total of $N$ balls, but different procedure for generating next step.

The state of the system is defined by a number $n$ of balls the first box, $0 \leq n \leq N$.

## Ehrenfest chain

(Bonamente)

Simple model of diffusion: same case of two boxes with a total of $N$ balls,
but different procedure for generating next step.

The state of the system is defined by a number $n$ of balls the first box, $0 \leq n \leq N$.

The Ehrenfest chain is defined by the following procedure. At each step:

- select a ball at random from either box, previously we were selecting a box
- move the selected ball in the other box.

## Ehrenfest chain

(Bonamente)

Simple model of diffusion: same case of two boxes with a total of $N$ balls, but different procedure for generating next step.

The state of the system is defined by a number $n$ of balls the first box, $0 \leq n \leq N$.

The Ehrenfest chain is defined by the following procedure. At each step:

- select a ball at random from either box,
- move the selected ball in the other box.

previously we were selecting a box

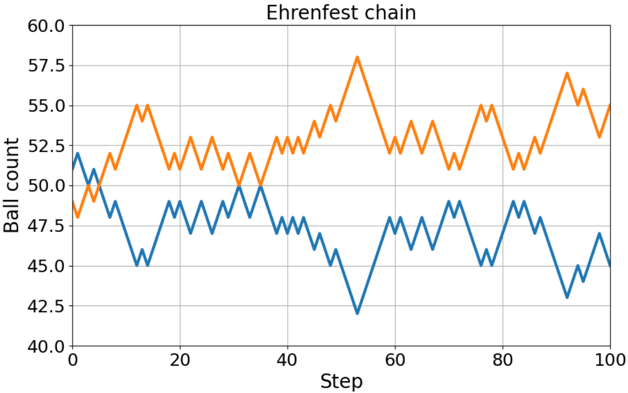
This chain can be presented in terms of the transition probabilities:

$$p(n^{(t+1)}) = \begin{cases} \frac{n^{(t)}}{N} & \text{for} \quad n^{(t+1)} = n^{(t)} - 1 \\ \frac{N-n^{(t)}}{N} & \qquad\;\; n^{(t+1)} = n^{(t)} + 1 \\ 0 & \qquad\;\; n^{(t+1)} \neq n^{(t)} \pm 1 \end{cases}$$

## Ehrenfest chain

14_Ehrenfest.ipynb Open in Colab

Result of the algorithm implementation

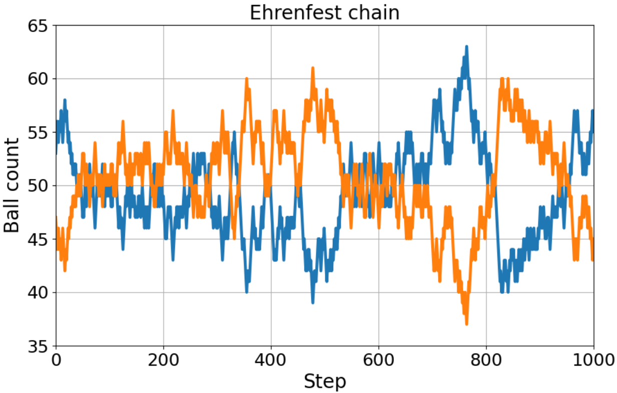


Looks again like symmetry violation?...

## Ehrenfest chain

14_Ehrenfest.ipynb Open in Colab

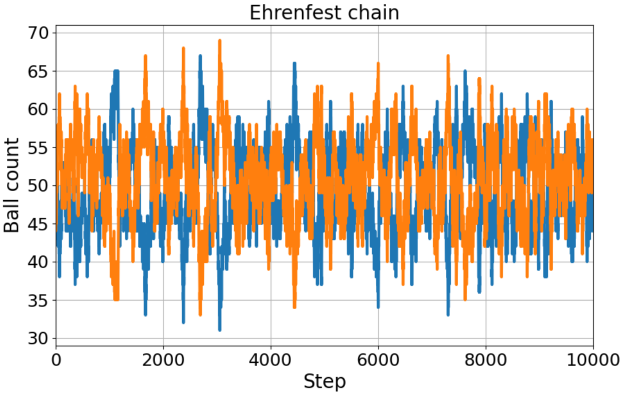
Result of the algorithm implementation



## Ehrenfest chain

14_Ehrenfest.ipynb Open in Colab

Result of the algorithm implementation

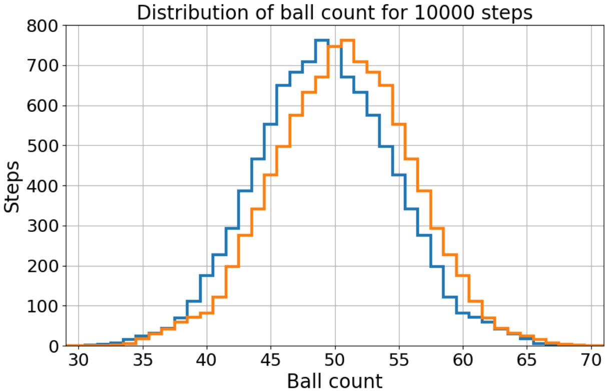


Symmetry restored on longer time scales...

## Simple example: Ehrenfest chain

14_Ehrenfest.ipynb Open in Colab

Result of the algorithm implementation

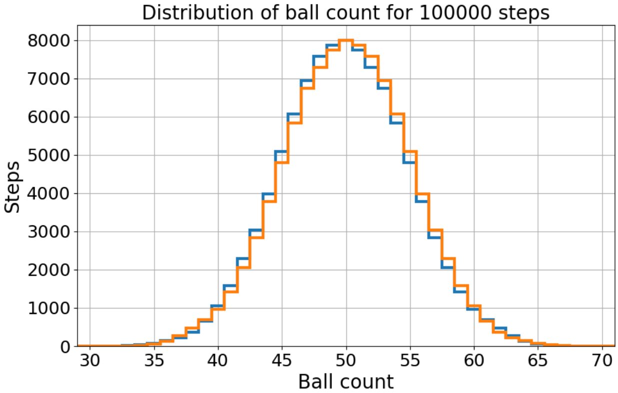


Fluctuations still visible in ball count distribution...

## Simple example: Ehrenfest chain

14_Ehrenfest.ipynb   Open in Colab

Result of the algorithm implementation

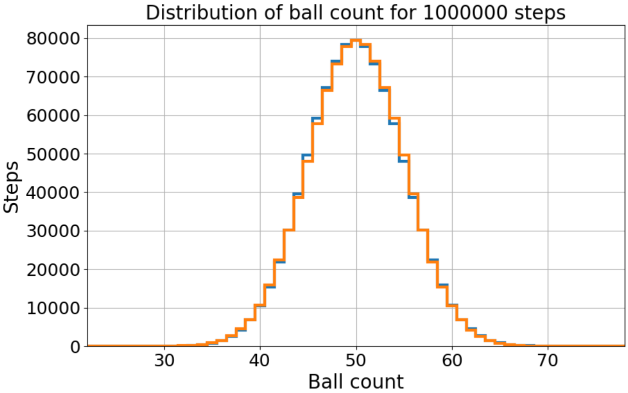


Decrease with the chain lenght...

## Simple example: Ehrenfest chain

14_Ehrenfest.ipynb Open in Colab

Result of the algorithm implementation



Decrease with the chain lenght...

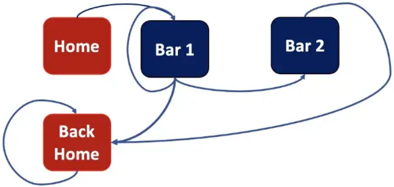
# Markov Chains

## Web example

Piero Paialunga in Towards Data Science

As a student you can go to the bar each Saturday.

And you need to go back home at some time...



We can consider the following "chain" of states (shown above):

- you always start from Home going to Bar 1 or Bar 2.
- after each drink in Bar 1 you have three choices:
  go Back Home, go to Bar 2 and order another drink in Bar 1.
- if you are already in Bar 2, you have only two choices after each round:
  go Back Home or order another drink (not shown).
- once you get Back Home, you stay there.

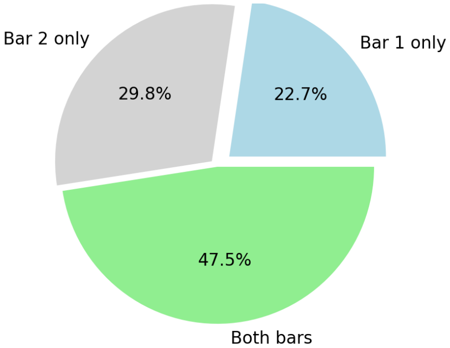
## Web example

14_TwoBar.ipynb Open in Colab

Even if all transition probabilities are known, it is not always possible to obtain statistical properties of the distribution directly...

But one can simulate Markov Chain state sequence many times...

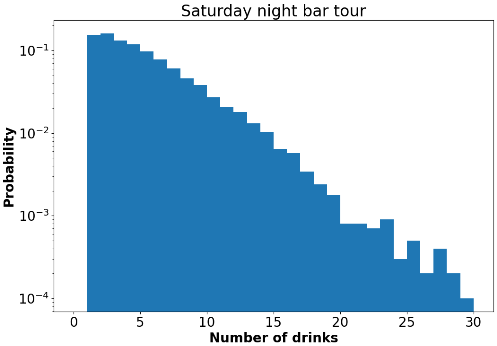
Probability of visiting bars:



## Web example

14_TwoBar2.ipynb Open in Colab

Probability density for the number of drinks:

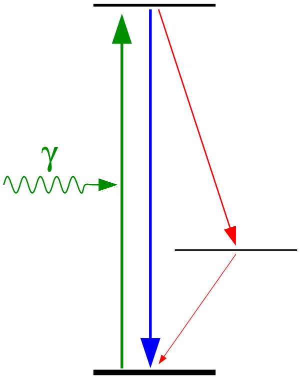


We can not only estimate the expected number of drinks (which we could also do from the known probabilities), but also the distribution...

## Another example

The chain in the web example always ended in the single 'Back Home' state.

Not very interesting...



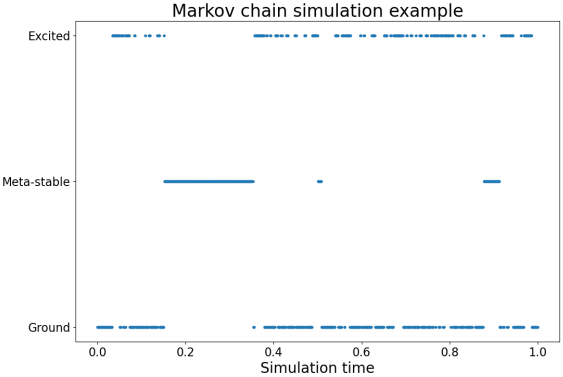
Consider an atom irradiated with the laser light tuned to the excitation energy:

- when in ground state, atom has certain probability (per time unit ≡ simulation step) to get excited
- when in the excited state, atom can radiate photon and go back to the ground state or, with lower probability, radiate softer photon and go to intermediate meta-stable state.
- when in the meta-stable state, probability of radiation (per unit of time) is very low.

## Another example

14_atom.ipynb Open in Colab

Example simulation results starting from ground state, 1000 time steps:

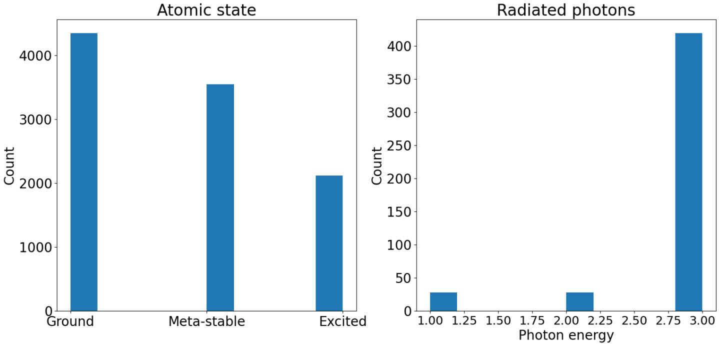


Fast oscillations between ground and excited state, longer stays in meta-stable...

## Another example

14_atom2.ipynb Open in Colab

Example simulation results starting from ground state, 10000 time steps:



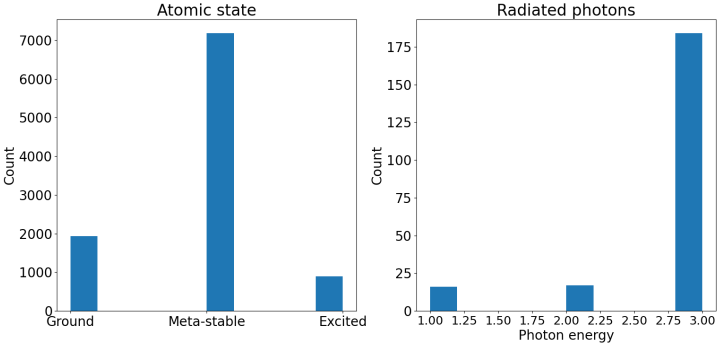
System "forgets" about the initial state fast. We can get distributions for different parameters...

## Another example

14_atom2.ipynb Open in Colab

Example simulation results starting from ground state, 10000 time steps:

After increasing meta-stable state lifetime:



System "forgets" about the initial state fast. We can get distributions for different parameters...

# Markov Chains

## Transition probability

Assume that the state space consists of $N$ states: $s_{(1)}, \ldots, s_{(N)}$.
Then, for each state $s_{(i)}$ on can define a set of on-step transition probabilities:

$$p_{ij} \quad = \quad p(X^{(t+1)} = s_{(j)} | X^{(t)} = s_{(i)})$$

We usually require that these probabilities are time-independent
(such chain is called time-homogeneous).

# Markov Chains

## Transition probability

Assume that the state space consists of $N$ states: $s_{(1)}, \ldots, s_{(N)}$.
Then, for each state $s_{(i)}$ on can define a set of on-step transition probabilities:

$$p_{ij} \;=\; p(X^{(t+1)} = s_{(j)} | X^{(t)} = s_{(i)})$$

We usually require that these probabilities are time-independent
(such chain is called time-homogeneous).

If we now describe state of the system by a $N$-component vector:

$$(s_{(i)})_j \;=\; \delta_{ij} \qquad \text{e.g.} \;\; s_{(1)} \;=\; (1, 0, 0, \ldots, 0)$$

## Transition probability

Assume that the state space consists of $N$ states: $s_{(1)}, \ldots, s_{(N)}$.
Then, for each state $s_{(i)}$ on can define a set of on-step transition probabilities:

$$p_{ij} \;=\; p(X^{(t+1)} = s_{(j)} | X^{(t)} = s_{(i)})$$

We usually require that these probabilities are time-independent
(such chain is called time-homogeneous).

If we now describe state of the system by a $N$-component vector:

$$(s_{(i)})_j \;=\; \delta_{ij} \qquad \text{e.g.} \;\; s_{(2)} \;=\; (0, 1, 0, \ldots, 0)$$

## Transition probability

Assume that the state space consists of $N$ states: $s_{(1)}, \ldots, s_{(N)}$.
Then, for each state $s_{(i)}$ on can define a set of on-step transition probabilities:

$$p_{ij} \;=\; p(X^{(t+1)} = s_{(j)} | X^{(t)} = s_{(i)})$$

We usually require that these probabilities are time-independent
(such chain is called time-homogeneous).

If we now describe state of the system by a $N$-component vector:

$$(s_{(i)})_j \;=\; \delta_{ij} \qquad \text{e.g.} \;\; s_{(N)} \;=\; (0, 0, 0, \ldots, 1)$$

## Transition probability

Assume that the state space consists of $N$ states: $s_{(1)}, \ldots, s_{(N)}$.
Then, for each state $s_{(i)}$ on can define a set of on-step transition probabilities:

$$p_{ij} \;=\; p(X^{(t+1)} = s_{(j)} | X^{(t)} = s_{(i)})$$

We usually require that these probabilities are time-independent
(such chain is called time-homogeneous).

If we now describe state of the system by a $N$-component vector:

$$(s_{(i)})_j \;=\; \delta_{ij} \qquad \text{e.g.} \;\; s_{(N)} \;=\; (0, 0, 0, \ldots, 1)$$

then probabilities for different states to proceed after state $s_{(i)}$ can be written as:

$$\mathbf{p} \;=\; s_{(i)} \cdot \mathbb{T} \qquad \text{where} \quad \mathbb{T} = (p_{ij})$$

is the transition matrix

# Markov Chains

## Chain properties

(Bonamente)

Probabilities of states after *n* time steps are then given by:

$$\mathbf{p}^{(n)} \quad = \quad s_{(i)} \cdot \mathbb{T}^{n}$$

## Chain properties

(Bonamente)

Probabilities of states after $n$ time steps are then given by:

$$\mathbf{p}^{(n)} = s_{(i)} \cdot \mathbb{T}^n$$

Let $u_k$ denote the probability that the system returns to the initial state $s_{(i)}$ in exactly $k$ time steps. We can define the total probability for returning to the initial state:

$$u = \sum_{k=1}^{\infty} u_k$$

## Chain properties

(Bonamente)

Probabilities of states after $n$ time steps are then given by:

$$\mathbf{p}^{(n)} = s_{(i)} \cdot \mathbb{T}^{n}$$

Let $u_k$ denote the probability that the system returns to the initial state $s_{(i)}$ in exactly $k$ time steps. We can define the total probability for returning to the initial state:

$$u = \sum_{k=1}^{\infty} u_k$$

States can be classified according to this probability:

- if $u = 1$ state $s_{(i)}$ is recurrent,
- if $u < 1$ state $s_{(i)}$ is transient.

If state is recurrent, it will certainly be observed again (even, if we have to wait very long), and the system will return to this state infinitely often.

# Markov Chains

## Chain properties

(Bonamente)

State $s_{(j)}$ is accessible from the initial state $s_{(i)}$, if there is a non-zero probability of reaching this state from the initial state in finite number of time steps:

$$\left(\mathbf{p}^{(m)}\right)_j \;=\; \left(s_{(i)} \cdot \mathbb{T}^m\right)_j \;>\; 0$$

for some natural number $m$.

## Chain properties

(Bonamente)

State $s_{(j)}$ is accessible from the initial state $s_{(i)}$, if there is a non-zero probability of reaching this state from the initial state in finite number of time steps:

$$\left(\mathbf{p}^{(m)}\right)_j = \left(s_{(i)} \cdot \mathbb{T}^m\right)_j > 0$$

for some natural number $m$.

If a state $s_{(j)}$ is accessible from a recurrent state $s_{(i)}$,
then $s_{(j)}$ is also recurrent, and $s_{(i)}$ is accessible from $s_{(j)}$.

## Chain properties

(Bonamente)

State $s_{(j)}$ is accessible from the initial state $s_{(i)}$, if there is a non-zero probability of reaching this state from the initial state in finite number of time steps:

$$\left(\mathbf{p}^{(m)}\right)_j \;=\; \left(s_{(i)} \cdot \mathbb{T}^m\right)_j \;>\; 0$$

for some natural number $m$.

If a state $s_{(j)}$ is accessible from a recurrent state $s_{(i)}$,
then $s_{(j)}$ is also recurrent, and $s_{(i)}$ is accessible from $s_{(j)}$.

If a Markov chain has a finite number of states and each state is accessible from any other state, then all states are recurrent.

## Chain properties

(Bonamente)

A chain is said to be irreducible if all states are accessible from others.

Possible states of reducible Markov Chain can be divided into two or more classes, which do not communicate with each other.

# Markov Chains

## Chain properties

(Bonamente)

A chain is said to be irreducible if all states are accessible from others.
Possible states of reducible Markov Chain can be divided into two or more classes, which do not communicate with each other.

A state $s_{(i)}$ is said to be periodic with period $T$ if system can return to this state only at times $t$ divisible by $T$:

$$\left(\mathbf{p}^{(t)}\right)_j = \begin{cases} p > 0 & \text{for} \quad t\%T = 0 \\ 0 & \quad\quad\; t\%T \; != 0 \end{cases}$$

All states of irreducible chain share the same period.

## Chain properties

(Bonamente)

A chain is said to be irreducible if all states are accessible from others.
Possible states of reducible Markov Chain can be divided into two or more classes, which do not communicate with each other.

A state $s_{(i)}$ is said to be periodic with period $T$ if system can return to this state only at times $t$ divisible by $T$:

$$\left(\mathbf{p}^{(t)}\right)_j = \begin{cases} p > 0 \quad \text{for} \quad t\%T = 0 \\ 0 \qquad\qquad\quad t\%T\ != 0 \end{cases}$$

All states of irreducible chain share the same period.

A chain is said to be aperiodic, if return to a given state can occur at any time
(corresponding to $T = 1$ in definition above).

## Stationary distribution

In most cases, we do not care about the initial system state, we want to calculate the set of probabilities for a system after a large number $n$ of steps:

$$\mathbf{p}^{\infty} \;=\; \lim_{n\to\infty} \mathbf{p}^{(n)}$$

This probabilities are called limiting probabilities.

## Stationary distribution

In most cases, we do not care about the initial system state, we want to calculate the set of probabilities for a system after a large number $n$ of steps:

$$\mathbf{p}^{\infty} = \lim_{n\to\infty} \mathbf{p}^{(n)}$$

This probabilities are called limiting probabilities.

For a irreducible aperiodic Markov Chain with recurrent states, limiting probabilities correspond to the stationary distribution:

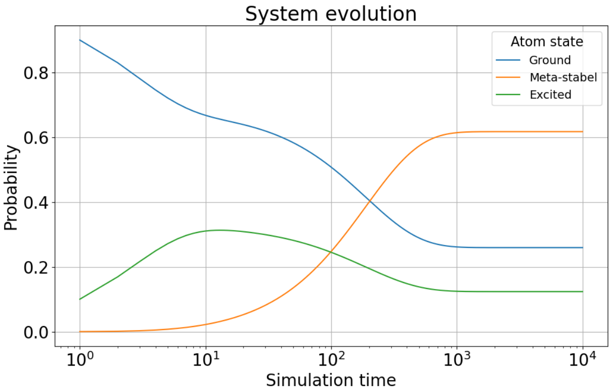
$$\boldsymbol{\pi} = \boldsymbol{\pi} \cdot \mathbb{T}$$

and this distribution is unique. Regardless of the starting point of the chain, the same stationary distribution will eventually be reached.

## Stationary distribution

14_atom3.ipynb Open in Colab

Evolution of state probabilities for system starting at 'Ground' state at $t = 0$



Stationary state reached for $t \sim 1000$

Note logarithmic time scale!

## Stationary distribution

this is what we look for in most cases

There are three possible approaches to finding a stationary solution:

- by running multiple Markov Chain instances and looking at final state distribution, simple but time consuming

# Markov Chains

## Stationary distribution

this is what we look for in most cases

There are three possible approaches to finding a stationary solution:

- by running multiple Markov Chain instances and looking at final state distribution,
  simple but time consuming

- applying the transfer matrix many times, starting for arbitrary initial state vector

## Stationary distribution

this is what we look for in most cases

There are three possible approaches to finding a stationary solution:

- by running multiple Markov Chain instances and looking at final state distribution, simple but time consuming
- applying the transfer matrix many times, starting for arbitrary initial state vector
- by looking for analytic solution to the problem:

$$\pi_j = \sum_j \pi_i \, p_{ij} \quad \text{stationary distribution}$$

$$\sum_i \pi_i = 1 \quad \text{normalization constrain}$$

$$\pi_i \geq 0$$

## Stationary distribution

Herman Scheepers on Towards Data Science

In the analytic approach the problem can be presented as a set of equations:

$$\left(\begin{array}{ccc} & \mathbb{T}^{\mathsf{T}} - \mathbb{I} & \\ \hline 1 & \dots & 1 \end{array}\right) \cdot \boldsymbol{\pi} \quad = \quad \left(\begin{array}{c} 0 \\ \vdots \\ 0 \\ \hline 1 \end{array}\right)$$

$$\mathbb{A} \qquad \cdot \boldsymbol{\pi} \quad = \quad \mathbf{b}$$

which are, however, not independent (the problem is over-constrained).

# Markov Chains

## Stationary distribution

Herman Scheepers on Towards Data Science

In the analytic approach the problem can be presented as a set of equations:

$$\left(\begin{array}{ccc} & \mathbb{T}^{\mathsf{T}} - \mathbb{I} & \\ & & \\ \hline 1 & \ldots & 1 \end{array}\right) \cdot \boldsymbol{\pi} \;=\; \left(\begin{array}{c} 0 \\ \vdots \\ 0 \\ \hline 1 \end{array}\right)$$

$$\mathbb{A} \cdot \boldsymbol{\pi} \;=\; \mathbf{b}$$

which are, however, not independent (the problem is over-constrained).

The simple solution is to multiply both sides by $\mathbb{A}^{\mathsf{T}}$:

$$\mathbb{A}^{\mathsf{T}}\mathbb{A} \cdot \boldsymbol{\pi} \;=\; \mathbb{A}^{\mathsf{T}}\,\mathbf{b}$$

which can now be solved with standard linear algebra procedures...

## Markov Chains

1. Markov Chains
2. Markov Chain Monte Carlo
3. Application to parameter fitting
4. Final exam

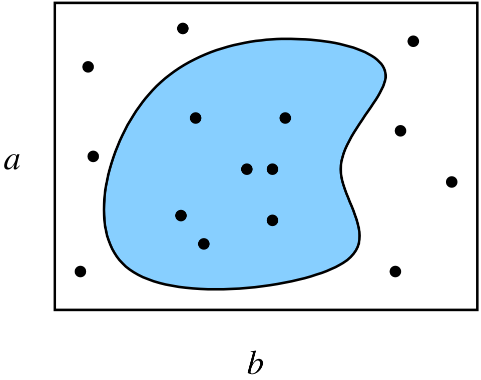
# Markov Chain Monte Carlo

## General concept

arXiv:0905.1629

We introduced Monte Carlo as an alternative method for integrating an arbitrary function.

Arbitrary parameter space can be considered.



Rejection technique

Generate uniformly distributed random points, select those in the considered parameter space...

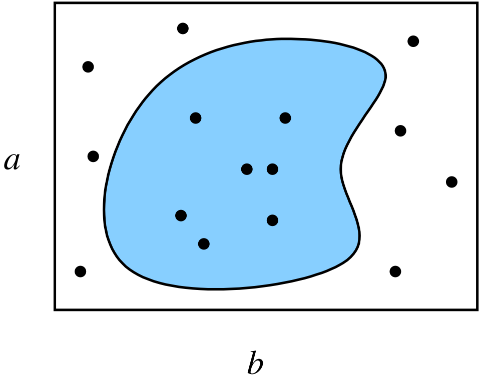
# Markov Chain Monte Carlo

## General concept

arXiv:0905.1629

We introduced Monte Carlo as an alternative method for integrating an arbitrary function.

Arbitrary parameter space can be considered.



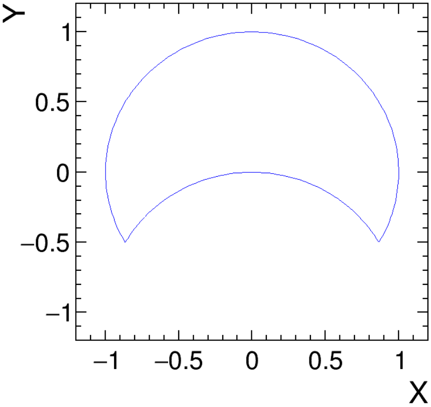
Rejection technique

Generate uniformly distributed random points, select those in the considered parameter space...

Efficiency can be low...

## Standard approach example

Generation of random points from the surface considered in lecture 05

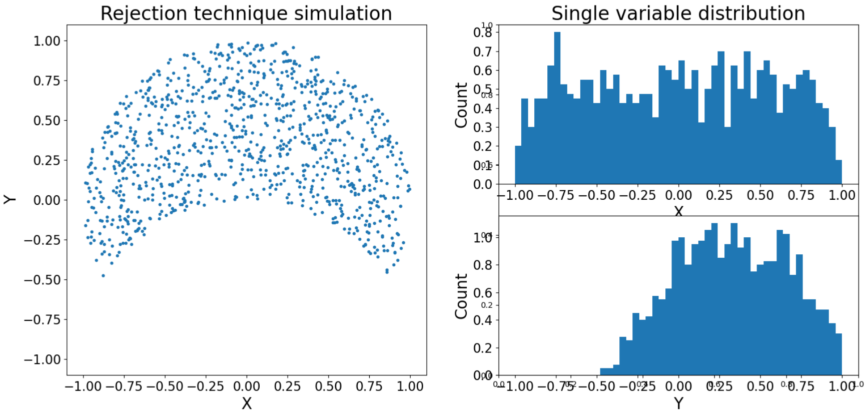


# Markov Chain Monte Carlo

## Standard approach example

14_mcmc.ipynb Open in Colab

Generation of random points from the surface considered in lecture 05



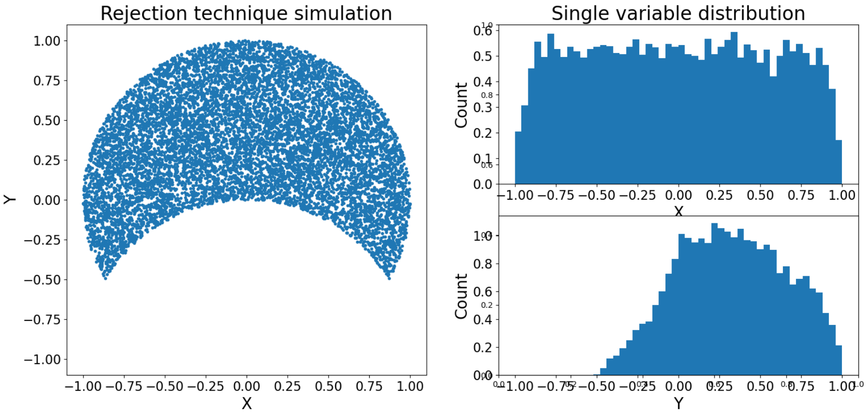
N=1 000

# Markov Chain Monte Carlo

## Standard approach example

14_mcmc.ipynb Open in Colab

Generation of random points from the surface considered in lecture 05

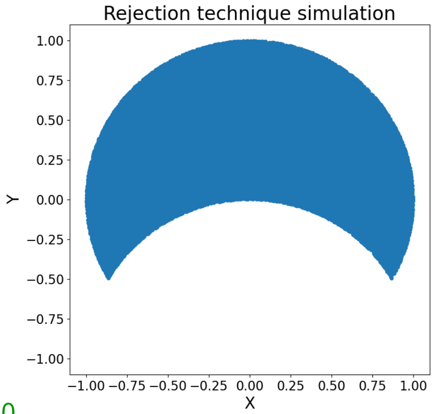


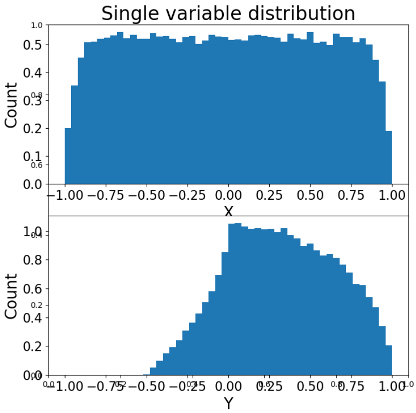
N=10 000

## Standard approach example

14_mcmc.ipynb Open in Colab

Generation of random points from the surface considered in lecture 05





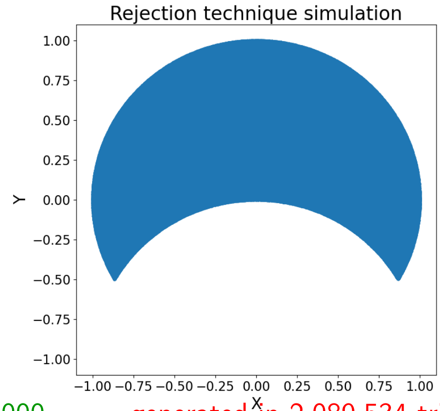
N=100 000

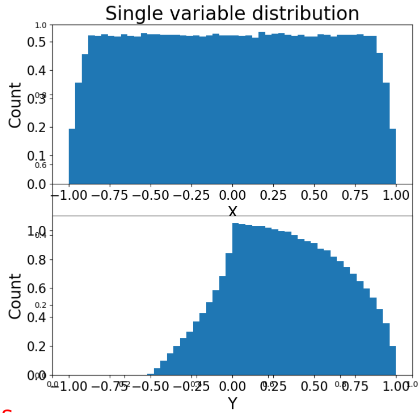
# Markov Chain Monte Carlo

## Standard approach example

14_mcmc.ipynb Open in Colab

Generation of random points from the surface considered in lecture 05





N=1 000 000

generated in 2 089 534 tries

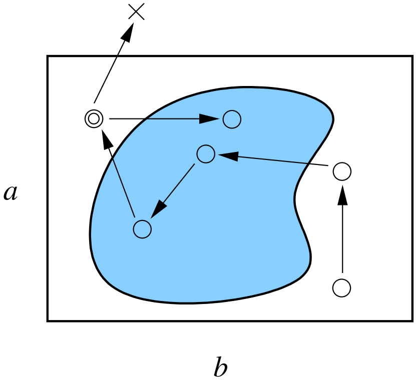
# Markov Chain Monte Carlo

## General concept

arXiv:0905.1629

We do not want to reject events!

Random move procedure: subsequent points generated by random variations of previous ones



Markov Chain Monte Carlo procedure

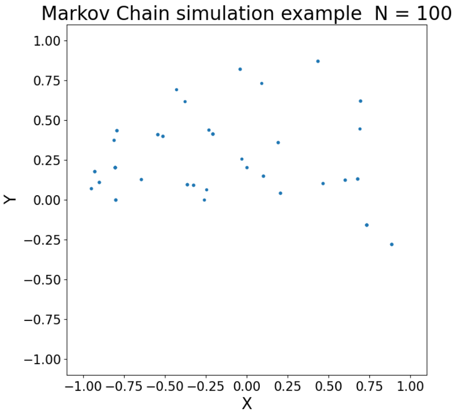
If the new point is outside the considered parameter space, do not reject it, but take the last point again (!)

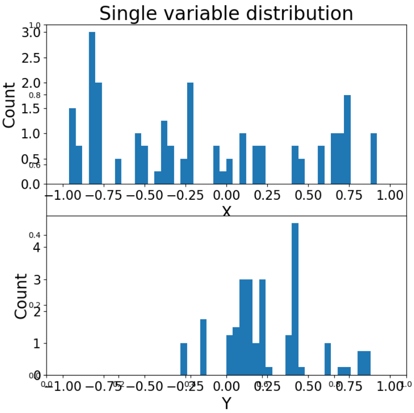
Can this procedure work ?

## Markov Chain MC example

14_mcmc.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



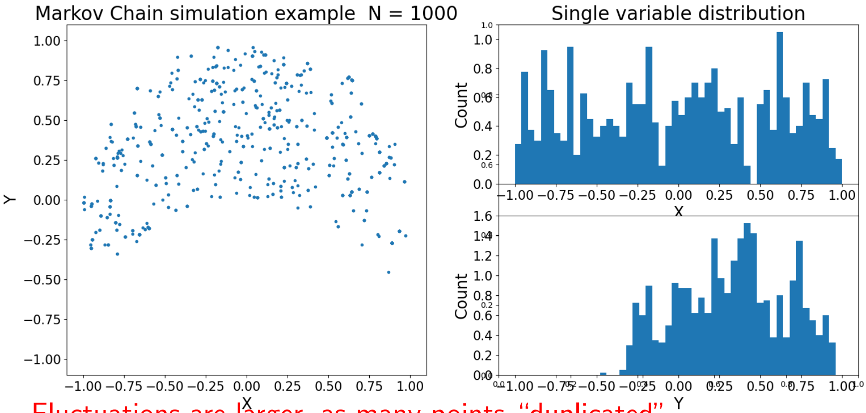


N=100

## Markov Chain MC example

14_mcmc.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$

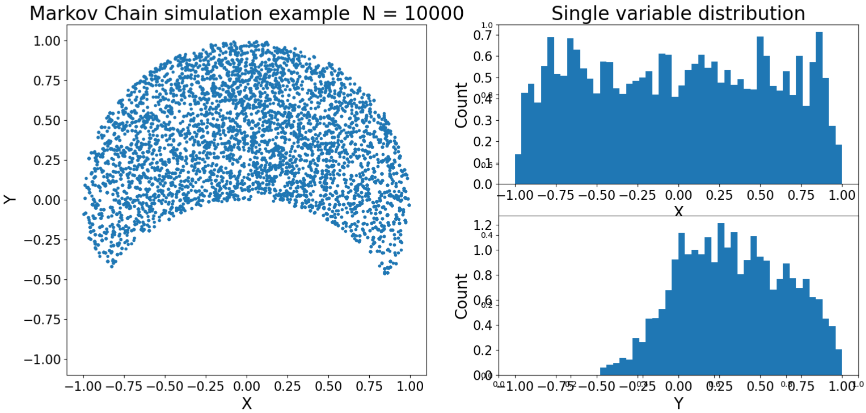


N=1 000 Fluctuations are larger, as many points “duplicated”

## Markov Chain MC example

14_mcmc.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$

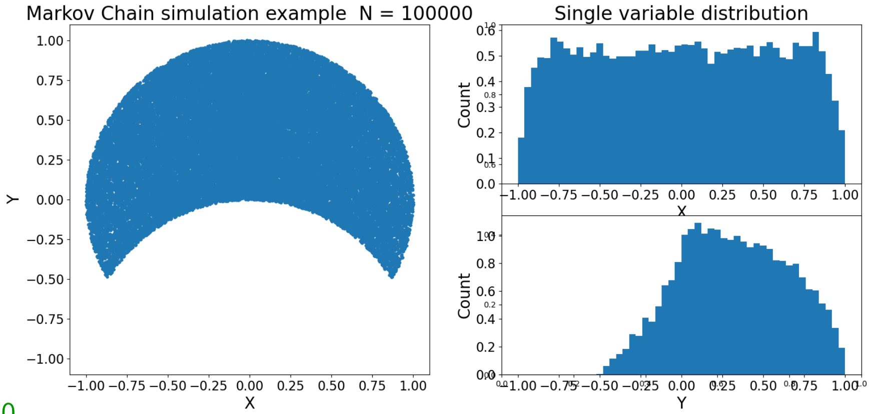


N=10 000

## Markov Chain MC example

14_mcmc.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$

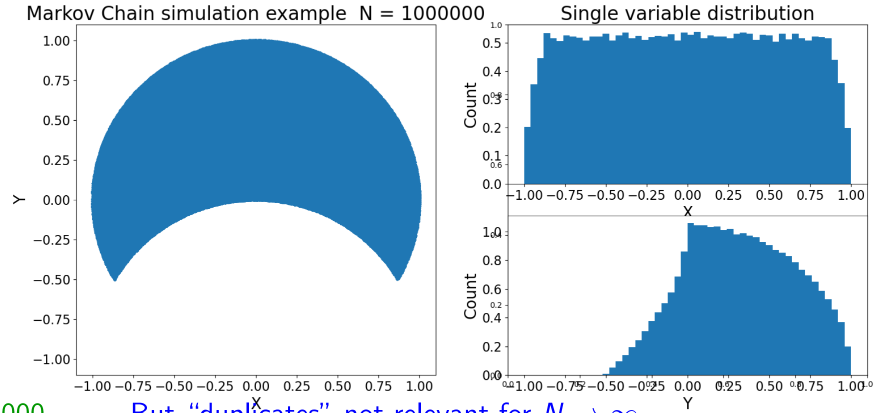


N=100 000

## Markov Chain MC example

14_mcmc.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



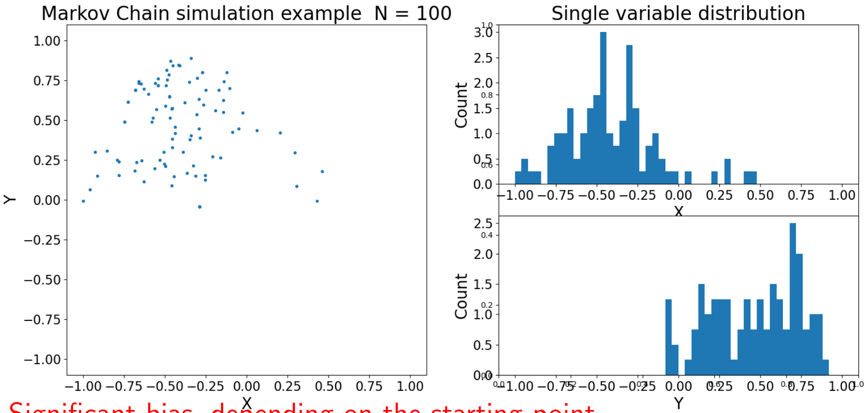
N=1 000 000

But “duplicates” not relevant for $N \to \infty$

# Markov Chain Monte Carlo

## Markov Chain example

We can reduce number of “duplicates” by reducing step: $\Delta x = \Delta y = 0.2$

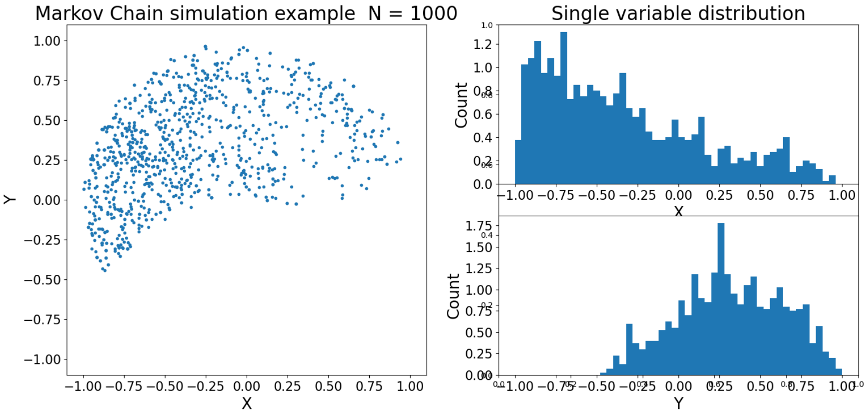


N=100 Significant bias, depending on the starting point...

# Markov Chain Monte Carlo

## Markov Chain example

We can reduce number of “duplicates” by reducing step: $\Delta x = \Delta y = 0.2$

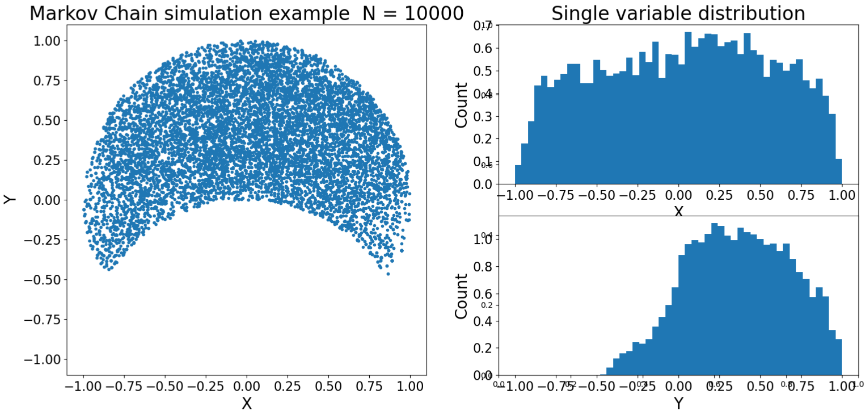


N=1 000

# Markov Chain Monte Carlo

## Markov Chain example

We can reduce number of “duplicates” by reducing step: $\Delta x = \Delta y = 0.2$

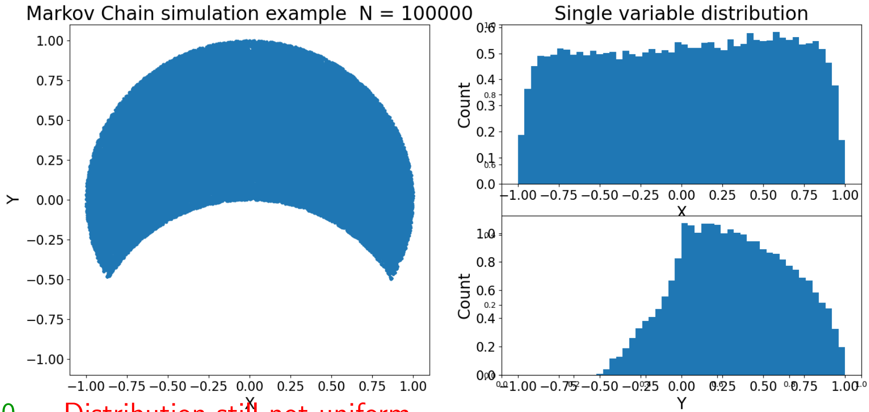


N=10 000

# Markov Chain Monte Carlo

## Markov Chain example

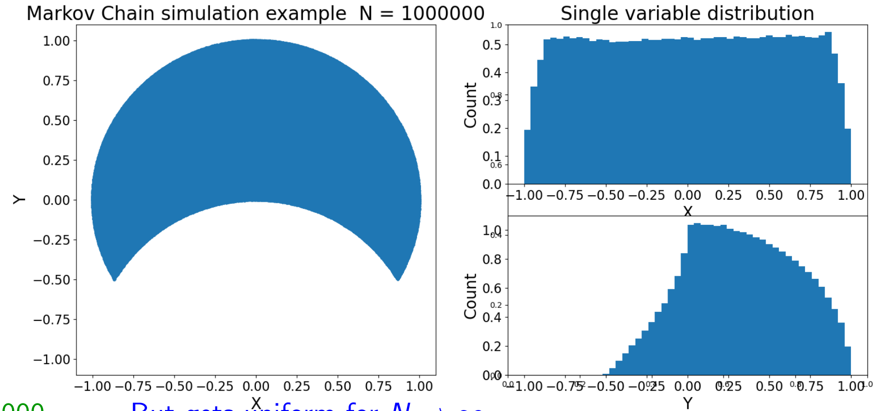
We can reduce number of “duplicates” by reducing step: $\Delta x = \Delta y = 0.2$



N=100 000  Distribution still not uniform...

## Markov Chain example

We can reduce number of "duplicates" by reducing step: $\Delta x = \Delta y = 0.2$



N=1 000 000

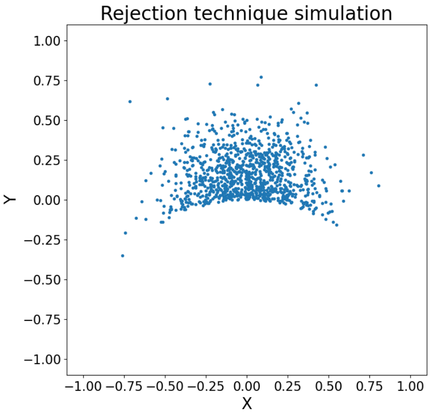
But gets uniform for $N \rightarrow \infty$

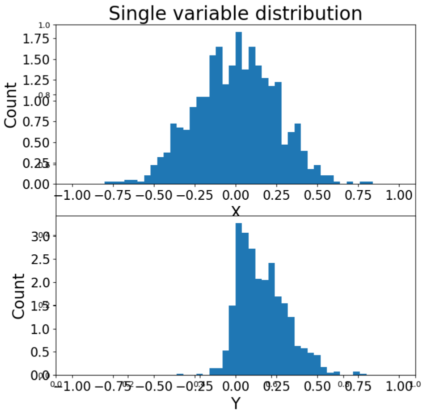
# Markov Chain Monte Carlo

## More general case

14_mcmc2.ipynb Open in Colab

Gaussian probability distribution in the considered parameter space





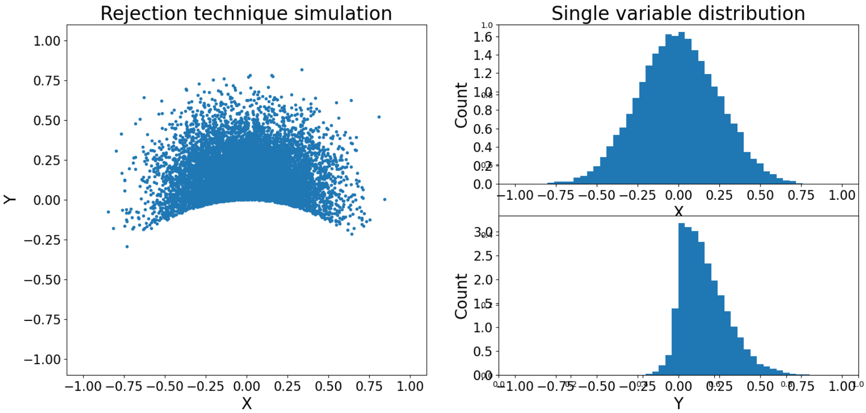
N=1 000

# Markov Chain Monte Carlo

## More general case

14_mcmc2.ipynb Open in Colab

Gaussian probability distribution in the considered parameter space



N=10 000

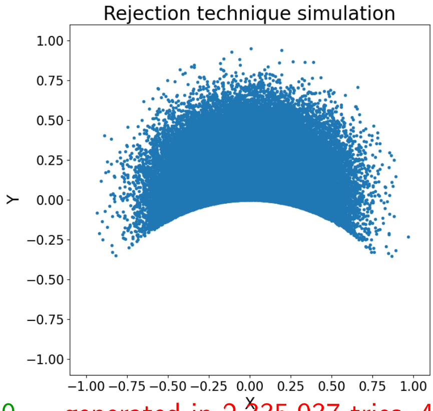
# Markov Chain Monte Carlo

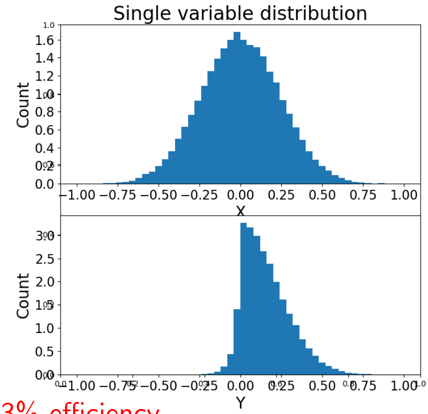
## More general case

14_mcmc2.ipynb

Open in Colab

Gaussian probability distribution in the considered parameter space





N=100 000 generated in 2 335 937 tries, 4.3% efficiency

## Metropolis–Hastings algorithm

(Givens)

Consider chain described by on-step transition probability $p(X^{(t+1)}|X^{(t)})$

To generate points distributed according to $f(X)$, for each step $t$:

- generate candidate point $X^\star$ from $p(X^\star|X^{(t)})$
- compute the Metropolis–Hastings ratio:

$$R = \frac{f(X^\star)\ \ p(X^{(t)}|X^\star)}{f(X^{(t)})\ p(X^\star|X^{(t)})}$$

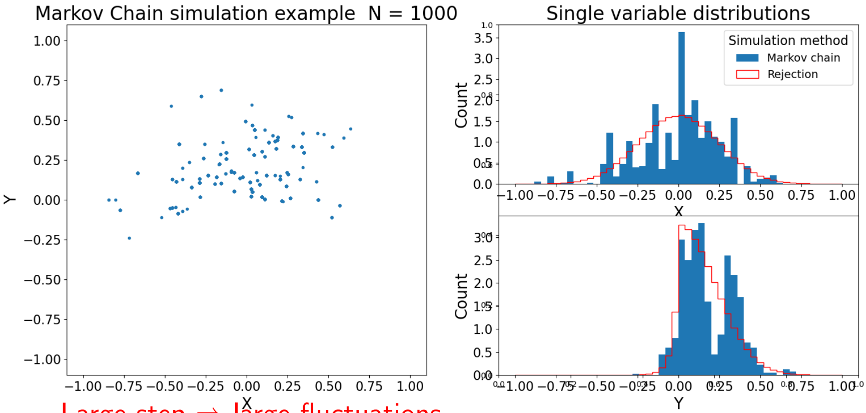
- for the next step take

$$X^{(t+1)} = \begin{cases} X^\star & \text{with probability } p^\star = \min\{R,1\} \\ X^{(t)} & \text{otherwise.} \qquad \text{with probability } 1-p^\star \end{cases}$$

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



N=1 000

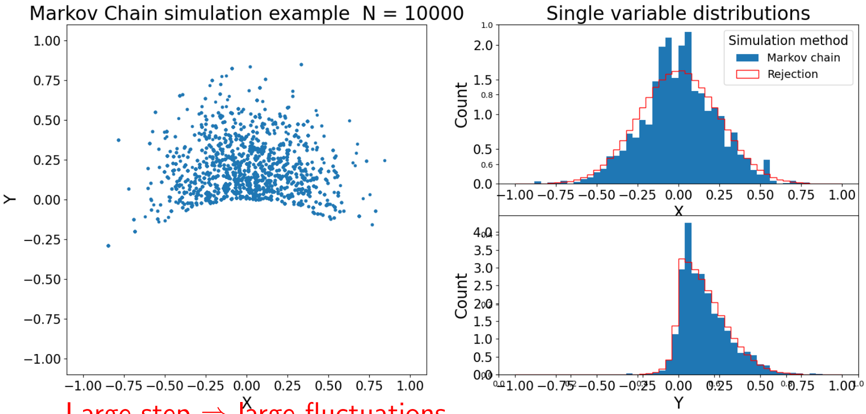
Large step ⇒ large fluctuations

# Markov Chain Monte Carlo

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



N=10 000 Large step ⇒ large fluctuations

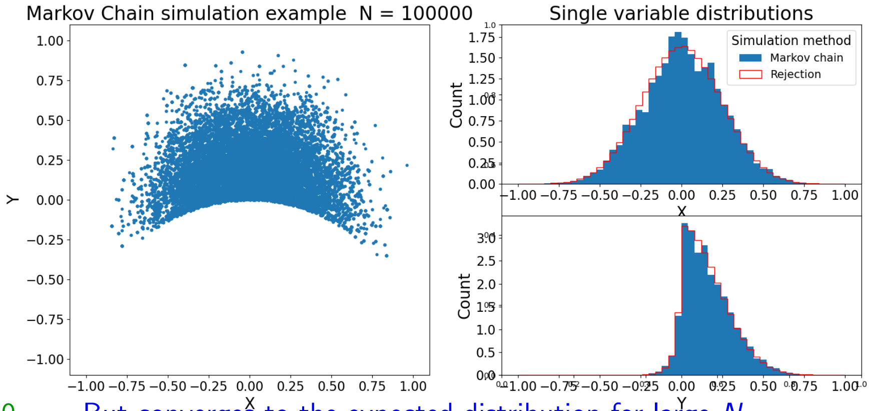
# Markov Chain Monte Carlo

## Markov Chain MC example (2)

14_mcmc3.ipynb

Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



N=100 000

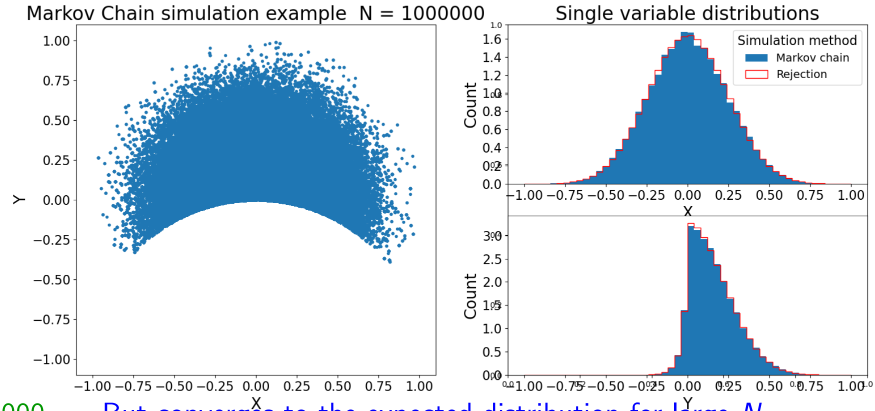
But converges to the expected distribution for large $N$

## Markov Chain MC example (2)

14_mcmc3.ipynb

Open in Colab

Using maximum step size: $\Delta x = \Delta y = 1$



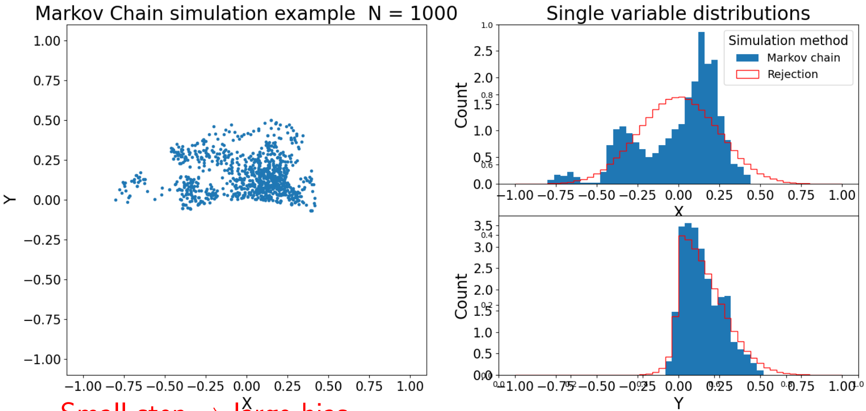
N=1 000 000

But converges to the expected distribution for large $N$

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.05$



N=1 000

Small step ⇒ large bias

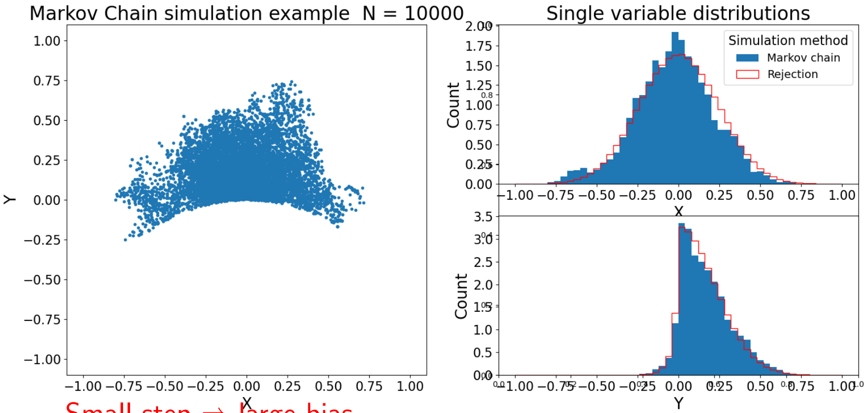
# Markov Chain Monte Carlo

## Markov Chain MC example (2)

14_mcmc3.ipynb

Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.05$



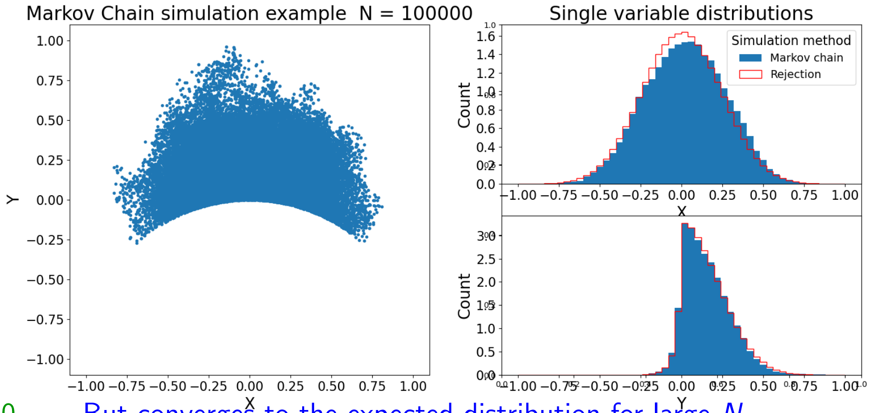
N=10 000

Small step ⇒ large bias

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.05$



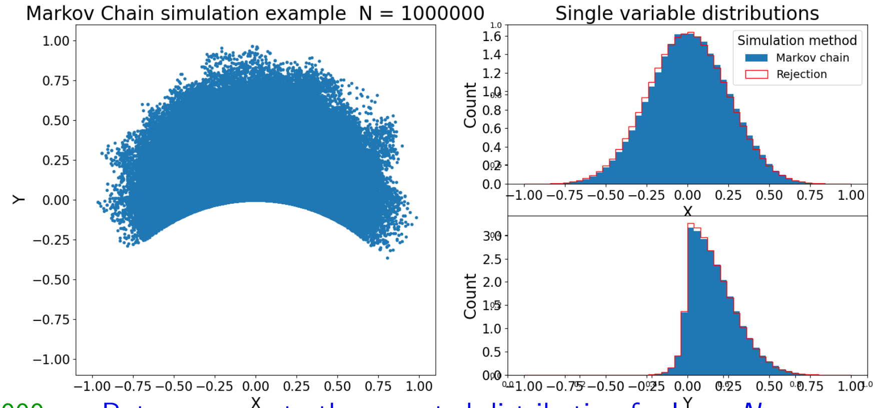
N=100 000

But converges to the expected distribution for large $N$

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.05$

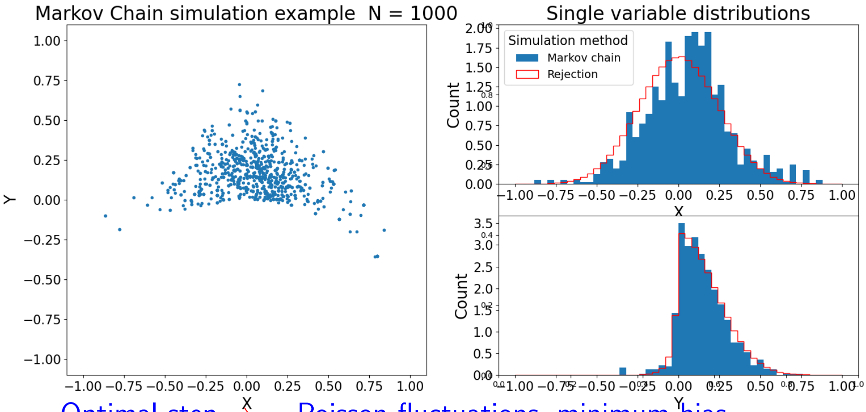


N=1 000 000 But converges to the expected distribution for large $N$

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.2$

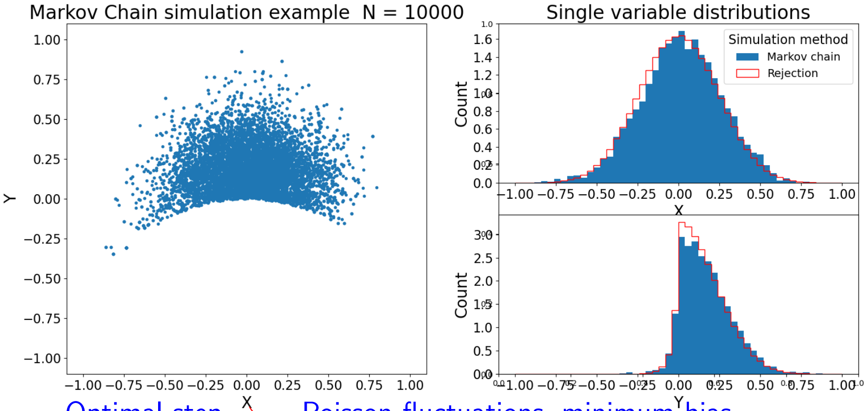


N=1 000 Optimal step $\Rightarrow \sim$ Poisson fluctuations, minimum bias

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.2$

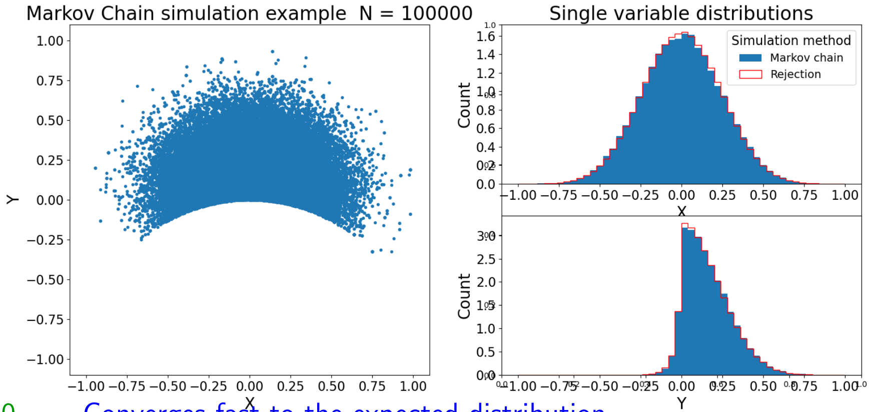


N=10 000 Optimal step ⇒ ∼ Poisson fluctuations, minimum bias

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.2$

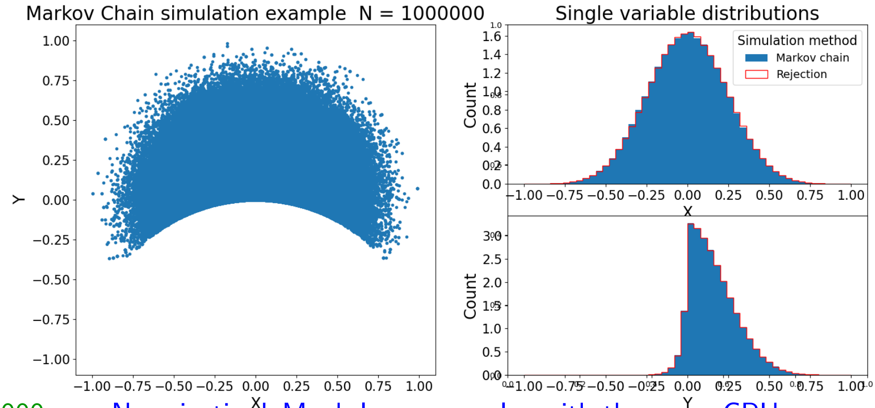


N=100 000 Converges fast to the expected distribution

## Markov Chain MC example (2)

14_mcmc3.ipynb Open in Colab

Using maximum step size: $\Delta x = \Delta y = 0.2$



N=1 000 000 No rejection! Much larger samples with the same CPU

## Markov Chains

1. Markov Chains
2. Markov Chain Monte Carlo
3. Application to parameter fitting
4. Final exam

# Application to parameter fitting

## Bayesian approach

(lecture 01)

Bayes theorem can be used to generalize the concept of probability.

In particular, one can consider "probability" of given hypothesis $H$ (theoretical model or model parameter, eg. Hubble constant) when taking into known outcome D (data) of the experiment

$$P(H|D) \;=\; \frac{P(D|H)}{P(D)} \cdot P(H)$$

There are two problems with this approach:

- $H$ can not be considered an event, sampling space can not be properly defined
- we need to make a subjective assumption about the "prior" $P(H)$ describing our initial belief in hypothesis $H$

For these reasons I try to avoid it, and do not refer to $P(H|D)$ as "probability".

Rather use "degree of belief" for results of the procedure applied to non random events

# Application to parameter fitting

## Maximum Likelihood Method

(lecture 06)

The likelihood function:

$$L(\boldsymbol{\lambda}, \mathbf{x}) \;=\; \prod_{j=1}^{N} f(\mathbf{x}^{(j)}; \boldsymbol{\lambda})$$

describes the probability of a given measurement results $\mathbf{x}$ for the selected parameter values $\boldsymbol{\lambda}$.

# Application to parameter fitting

## Maximum Likelihood Method

(lecture 06)

The likelihood function:

$$L(\boldsymbol{\lambda}, \mathbf{x}) \;=\; \prod_{j=1}^{N} f(\mathbf{x}^{(j)}; \boldsymbol{\lambda})$$

describes the probability of a given measurement results $\mathbf{x}$ for the selected parameter values $\boldsymbol{\lambda}$.

In the bayesian approach we can refer it to “probability distribution” for the parameters $\boldsymbol{\lambda}$:

$$f(\boldsymbol{\lambda}) \;\sim\; L(\boldsymbol{\lambda}, \mathbf{x}) \cdot p(\boldsymbol{\lambda})$$

where $p(\boldsymbol{\lambda})$ is the assumed prior distribution for parameters $\boldsymbol{\lambda}$. (usually flat)

# Application to parameter fitting

## Maximum Likelihood Method

(lecture 06)

The likelihood function:

$$L(\boldsymbol{\lambda}, \mathbf{x}) \;=\; \prod_{j=1}^{N} f(\mathbf{x}^{(j)}; \boldsymbol{\lambda})$$

describes the probability of a given measurement results $\mathbf{x}$ for the selected parameter values $\boldsymbol{\lambda}$.

In the bayesian approach we can refer it to “probability distribution” for the parameters $\boldsymbol{\lambda}$:

$$f(\boldsymbol{\lambda}) \;\sim\; L(\boldsymbol{\lambda}, \mathbf{x}) \cdot p(\boldsymbol{\lambda})$$

where $p(\boldsymbol{\lambda})$ is the assumed prior distribution for parameters $\boldsymbol{\lambda}$. (usually flat)

If we know $f(\boldsymbol{\lambda})$, we can construct Markov Chain in $\boldsymbol{\lambda}$ space.

# Application to parameter fitting

## Maximum Likelihood Method (lecture 06)

The likelihood function:

$$L(\boldsymbol{\lambda}, \mathbf{x}) \quad = \quad \prod_{j=1}^{N} f(\mathbf{x}^{(j)}; \boldsymbol{\lambda})$$

describes the probability of a given measurement results $\mathbf{x}$ for the selected parameter values $\boldsymbol{\lambda}$.

In the bayesian approach we can refer it to “probability distribution” for the parameters $\boldsymbol{\lambda}$:

$$f(\boldsymbol{\lambda}) \quad \sim \quad L(\boldsymbol{\lambda}, \mathbf{x}) \cdot p(\boldsymbol{\lambda})$$

where $p(\boldsymbol{\lambda})$ is the assumed prior distribution for parameters $\boldsymbol{\lambda}$. (usually flat)

If we know $f(\boldsymbol{\lambda})$, we can construct Markov Chain in $\boldsymbol{\lambda}$ space.

With Metropolis–Hastings algorithm, starting from arbitrary $\boldsymbol{\lambda}^{(0)}$ point, the chain should converge to $f(\boldsymbol{\lambda})$ distribution for $N \rightarrow \infty$.

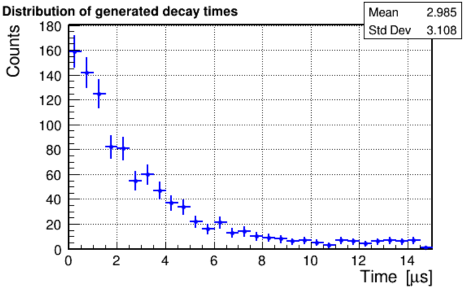
# Application to parameter fitting

## Example

1000 events were collected in the muon lifetime measurement. Distribution can be described by the formula:

$$\frac{dN}{dt} = \frac{N_{sig}}{\tau} e^{-\frac{t}{\tau}} + \frac{dN_{bg}}{dt}$$

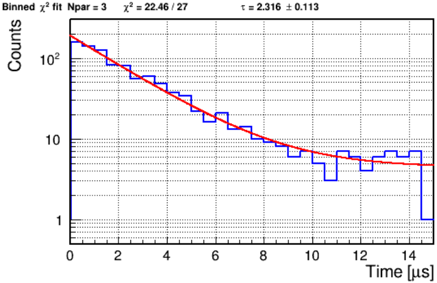
with flat background level known to be $\frac{dN_{bg}}{dt} = 10 \pm \Delta \; \mu s^{-1}$



# Application to parameter fitting

## Example

Histogram can be fitted using iterative $\chi^2$ minimization procedure (without bg constraint)



Fit results:

$$
\begin{aligned}
\tau &= 2.316 \pm 0.113\ \mu s \\
N_{sig} &= 430.773 \pm 16.611 \\
N_{bg} &= 4.399 \pm 0.424 \\
\chi^2 &= 22.460/27
\end{aligned}
$$

$$
\mathrm{Corr} = \begin{pmatrix} 1. & 0.279 & -0.392 \\ 0.279 & 1. & -0.309 \\ -0.392 & -0.309 & 1. \end{pmatrix}
$$

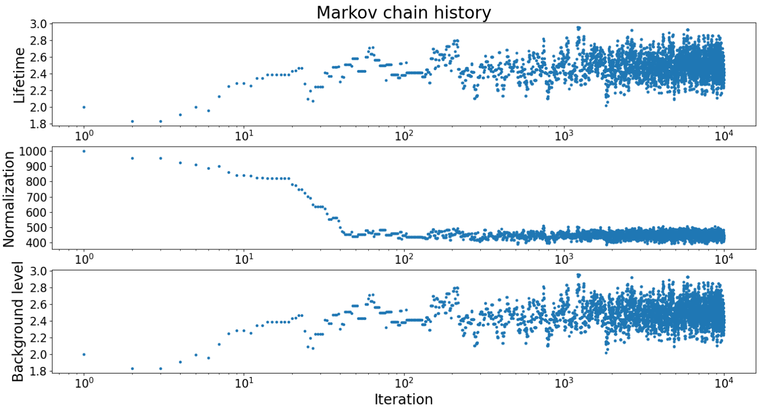
# Application to parameter fitting

**Example**

14_mcfit1.ipynb

Open in Colab

Parameter evolution in the Markov Chain



Stable distribution obtained already after about 100 iterations

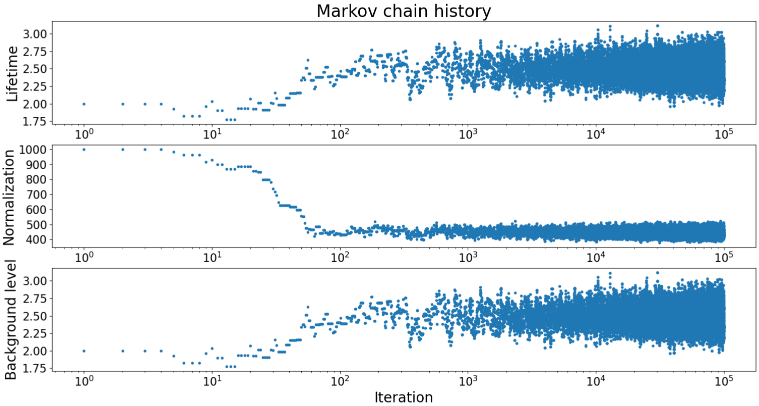
# Application to parameter fitting

## Example

14_mcfit1.ipynb

Open in Colab

Parameter evolution in the Markov Chain



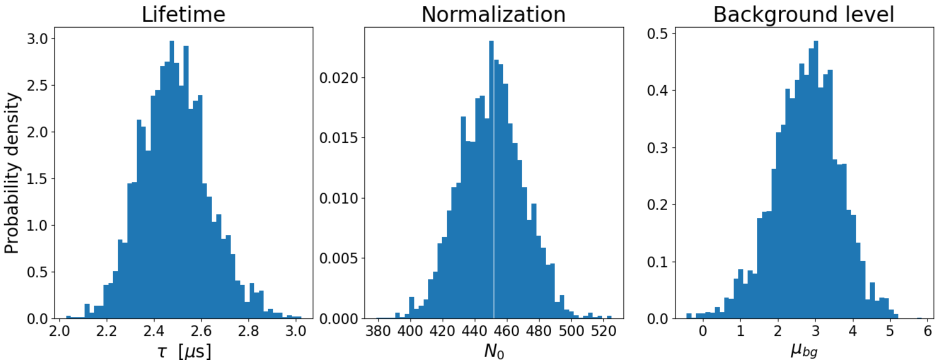
Stable distribution obtained already after about 100 iterations

# Application to parameter fitting

**Example**

14_mcfit2.ipynb Open in Colab

Parameter distributions after $N = 10\,000$ iterations (skipping first 100)



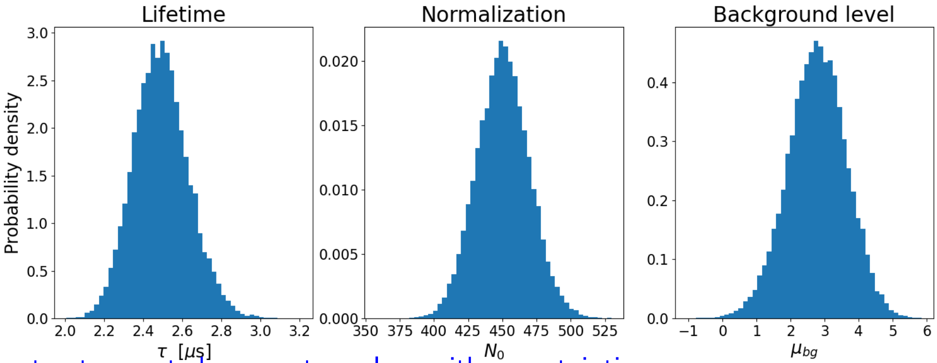
# Application to parameter fitting

**Example**

14_mcfit2.ipynb

Open in Colab

Parameter distributions after $N = 100\,000$ iterations (skipping first 1000)



We can extract expected parameter values with uncertainties...

but also identify problems, e.g. find multiple solutions...

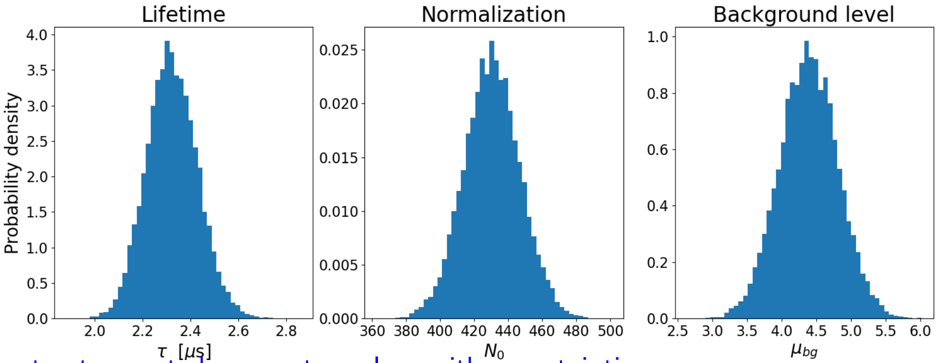
# Application to parameter fitting

**Example**

14_mcfit2.ipynb Open in Colab

Parameter distributions after $N = 100\,000$ iterations (skipping first 1000)

Including background level constraint



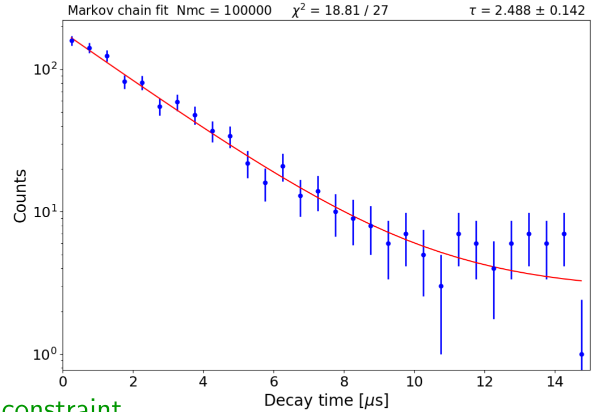
We can extract expected parameter values with uncertainties...

but also identify problems, e.g. find multiple solutions...

# Application to parameter fitting

**Example**

Nominal solution from Markov Chain (mean values of parameters)



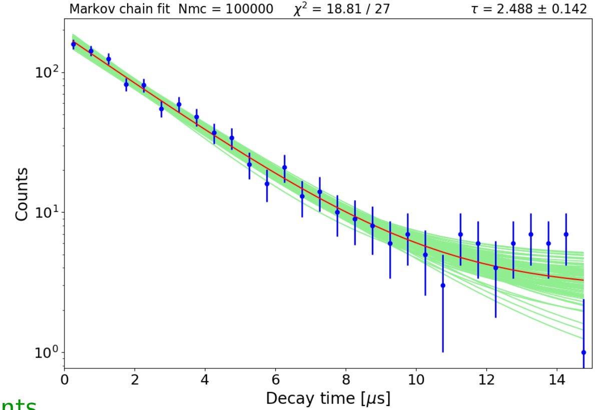
Without background constraint

# Application to parameter fitting

**Example**

14_mcfit3.ipynb Open in Colab

But we can also get the probability distribution of the fit results:



Last 100 chain elements

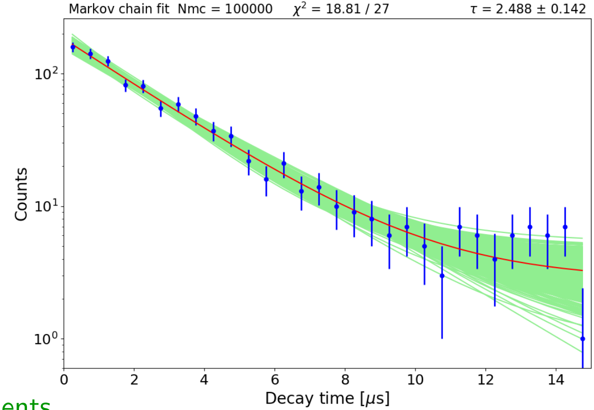
# Application to parameter fitting

**Example**

14_mcfit3.ipynb

Open in Colab

But we can also get the probability distribution of the fit results:



Last 1000 chain elements

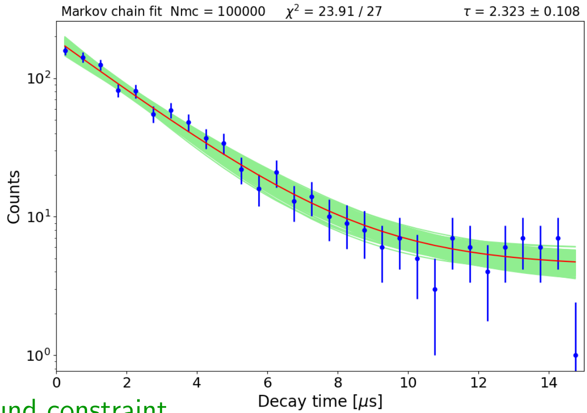
# Application to parameter fitting

## Example

14_mcfit3.ipynb

Open in Colab

But we can also get the probability distribution of the fit results:

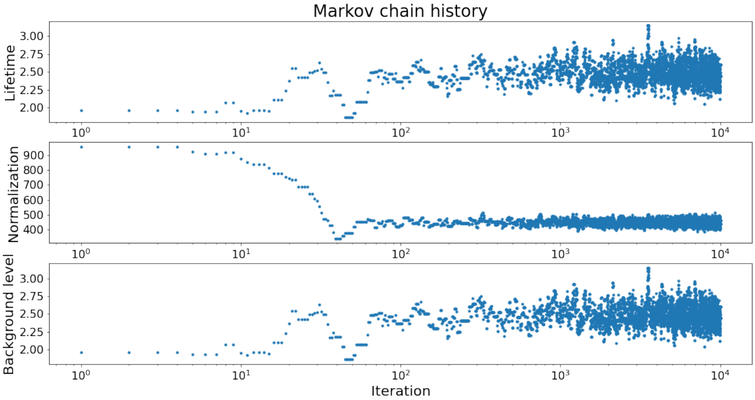


After adding background constraint

# Application to parameter fitting

## Example

Markov Chain Monte Carlo does not work "out of the box"

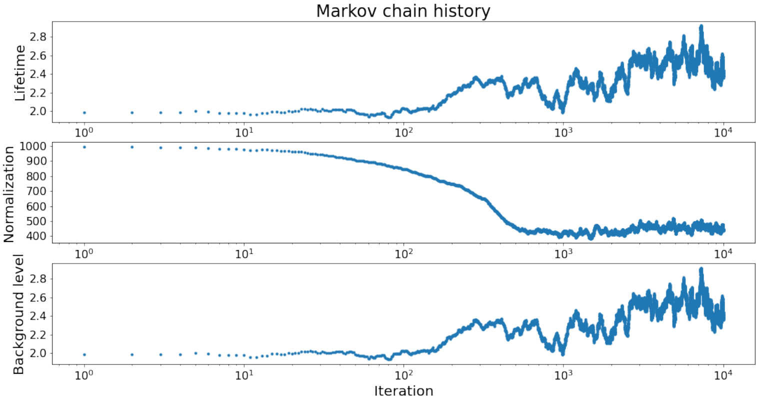


It converges fast with the proper choice of parameter variation steps

# Application to parameter fitting

## Example

Markov Chain Monte Carlo does not work “out of the box”

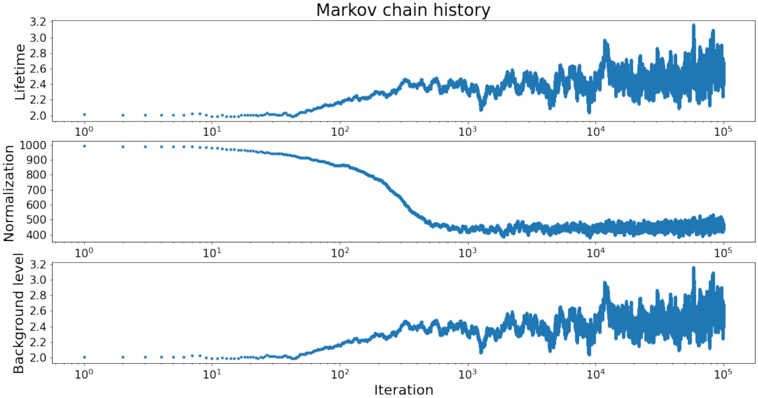


Convergence can be very slow, if parameter steps too small...

# Application to parameter fitting

## Example

Markov Chain Monte Carlo does not work “out of the box”

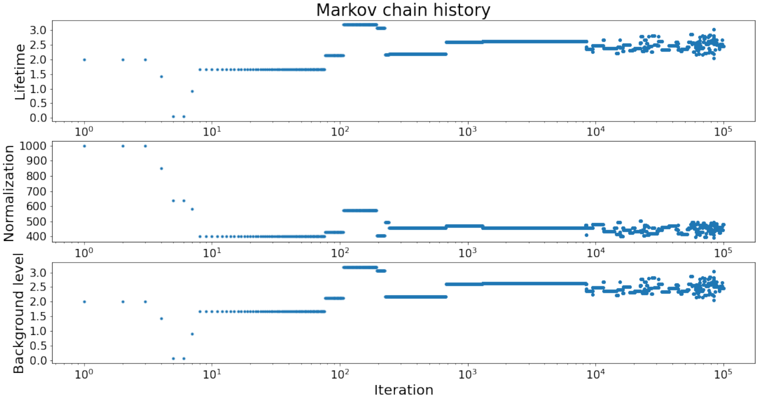


Convergence can be very slow, if parameter steps too small...

# Application to parameter fitting

## Example

Markov Chain Monte Carlo does not work “out of the box”



Fluctuations significantly increased, if steps are too large...

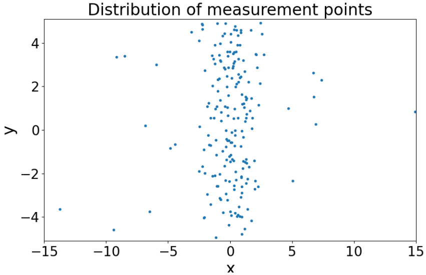
# Application to parameter fitting

## Example (2)

200 events were measured in the electron scattering experiment.
The expected distribution corresponds to that of the single slit diffraction:

$$p(x) \quad = \quad C\ a \cdot \left( \frac{\sin a(x - x_0)}{a\,(x - x_0)} \right)^2$$

where $a$ is the scaling factor and $x_0$ is the position of the maximum.

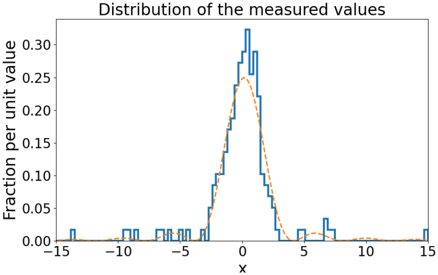


## Example (2)

200 events were measured in the electron scattering experiment.
The expected distribution corresponds to that of the single slit diffraction:

$$p(x) \quad = \quad C\; a \cdot \left( \frac{\sin a(x - x_0)}{a\,(x - x_0)} \right)^2$$

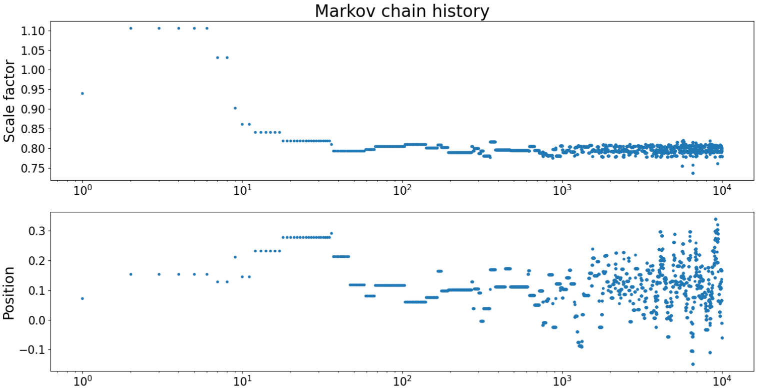
where $a$ is the scaling factor and $x_0$ is the position of the maximum.



# Application to parameter fitting

## Example (2)

Standard unbinned likeligood fit fails for this problem! Markov Chain Monte Carlo works...

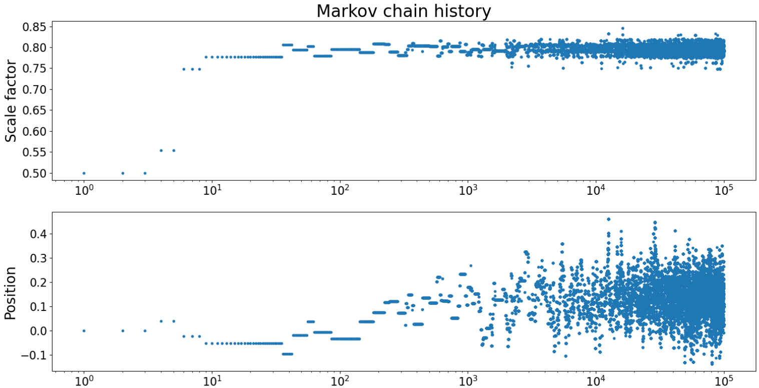


It converges fast with the proper choice of parameter variation steps

# Application to parameter fitting

## Example (2)

Standard unbinned likeligood fit fails for this problem! Markov Chain Monte Carlo works...

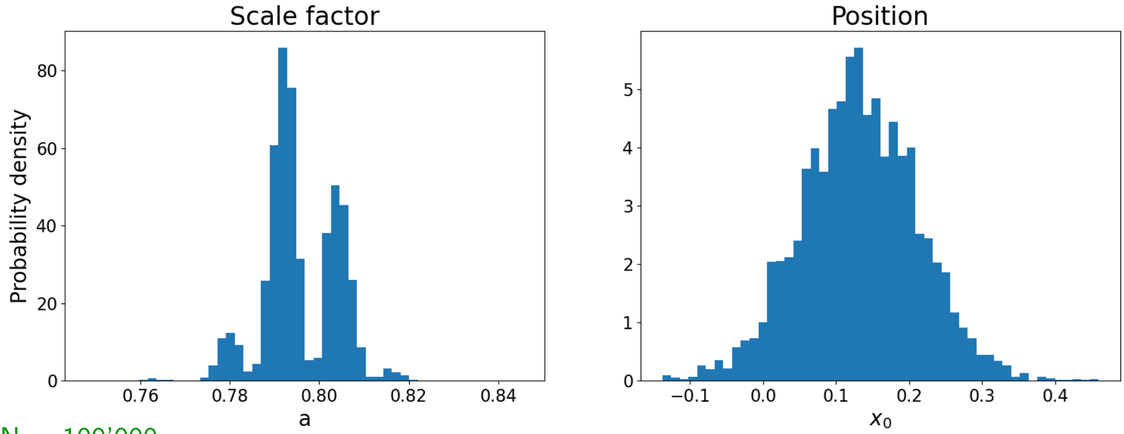


It converges fast with the proper choice of parameter variation steps

# Application to parameter fitting

## Example (2)

Standard unbinned likeligood fit fails for this problem! Markov Chain Monte Carlo works...

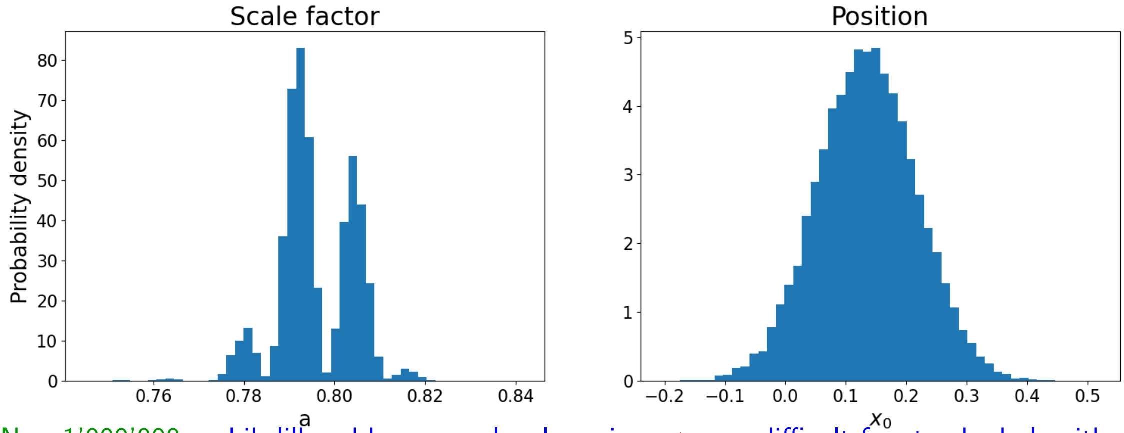


N = 100’000

# Application to parameter fitting

## Example (2)

Standard unbinned likeligood fit fails for this problem! Markov Chain Monte Carlo works...



N = 1'000'000 Likelilhood has many local maxima ⇒ very difficult for standard algorithms!

# Markov Chain Monte Carlo

## Final remarks

Markov Chains are powerful tools to solve many problems that are difficult to approach “directly”, using other numerical techniques

However, it is crucial to make sure they converge, before using their output for the analysis. Algorithm tuning may be required...

# Markov Chain Monte Carlo

## Final remarks

Markov Chains are powerful tools to solve many problems that are difficult to approach "directly", using other numerical techniques

However, it is crucial to make sure they converge, before using their output for the analysis. Algorithm tuning may be required...

Only the simplest approach was presented, many more advanced algorithms exist for more effective step generation. Probability $p(X^{(t+1)}|X^{(t)})$ does not need to be uniform!

# Markov Chain Monte Carlo

## Final remarks

Markov Chains are powerful tools to solve many problems that are difficult to approach "directly", using other numerical techniques

However, it is crucial to make sure they converge, before using their output for the analysis. Algorithm tuning may be required...

Only the simplest approach was presented, many more advanced algorithms exist for more effective step generation. Probability $p(X^{(t+1)}|X^{(t)})$ does not need to be uniform!

Events generated with Markov Chain MC are not independent!
One should not use subsequent events together in the analysis (eg. for background estimates)

## Markov Chains

1. Markov Chains
2. Markov Chain Monte Carlo
3. Application to parameter fitting
4. Final exam

## Final exam

As described in the syllabus, assessment will be based on home exercises and the written exam. 50% of points collected from exercises and exam (with same weights) required to pass.

For the written exam, you will have to solve five problems similar to those in homeworks (maybe a little bit more complex, as you get 13 points for each).

Problems will be put on Kampus on Sunday, February 2nd, and you should upload solutions to Kampus (each one as a separate file) within one week, till Sunday, February 9th (23:55).

By uploading the solutions to Kampus you declare that they resulted from your own work and that you have not shared nor discussed them with anyone.