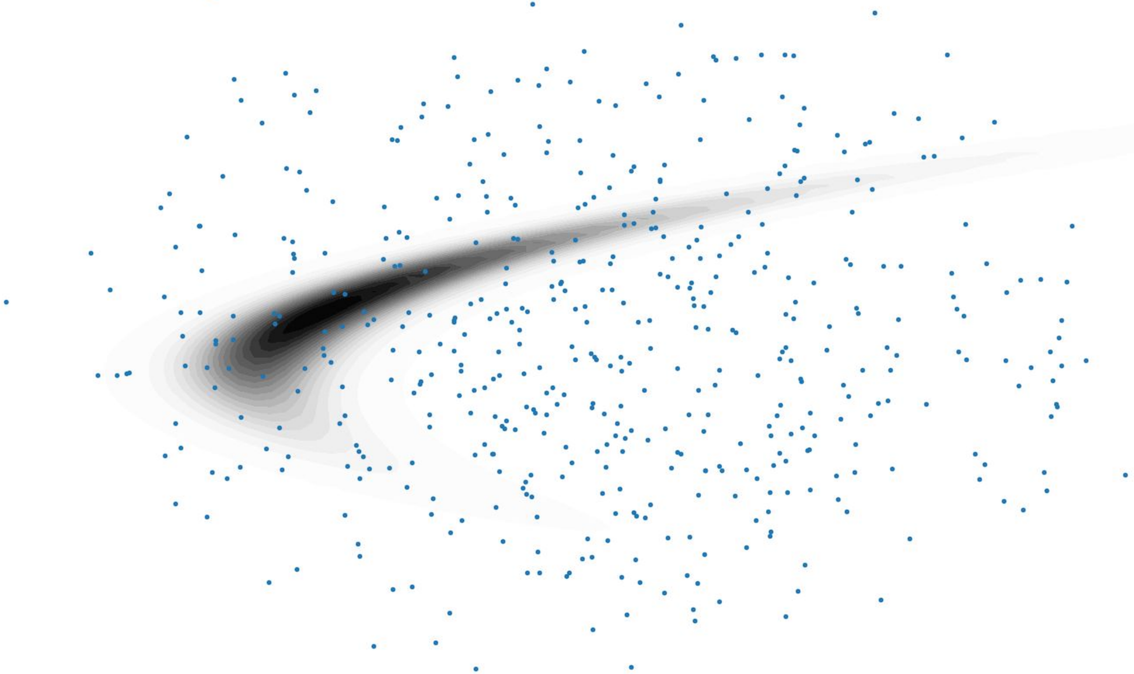
Vzorčenje v visokih dimenzijah:
Mikrokanonični Hamiltonski Monte Carlo



Jakob Robnik

mentor: Uroš Seljak

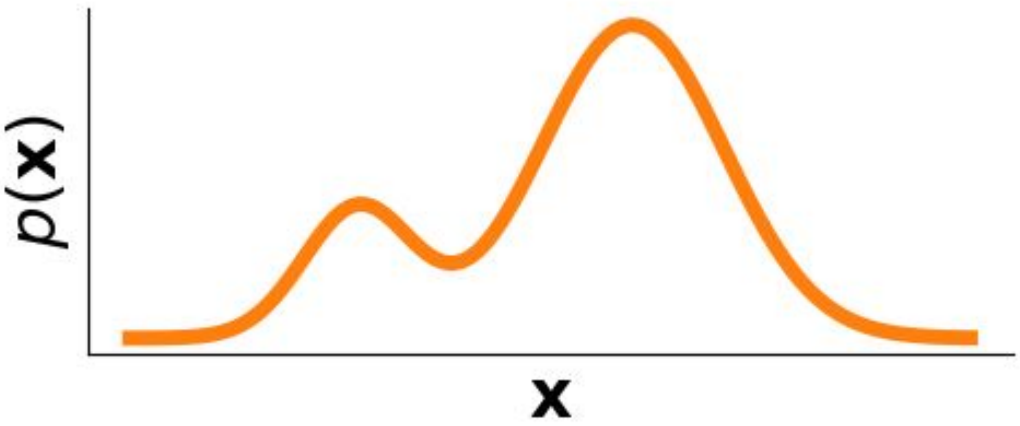
ostali sodelavci: Eva Silverstein, Bruno De Luca, Jaime Ruiz Zapatero, Reuben Cohn-Gordon, Qijia Jiang, Adrian Bayer, Rohith Karur,

HEISING-SIMONS FOUNDATION

Berkeley
UNIVERSITY OF CALIFORNIA

Vzorčenje

Naloga: generiraj vzorce, porazdeljene po dani porazdelitvi.

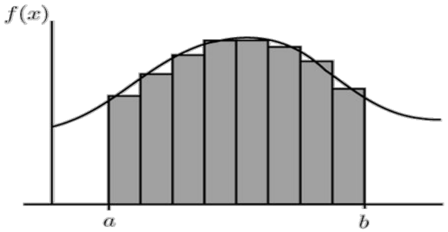


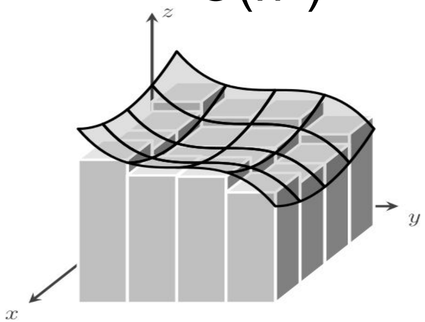
Zakaj je to pomemben problem?

Integriranje

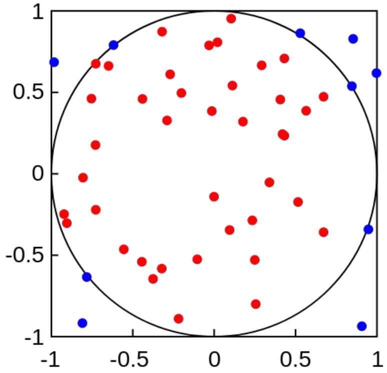
Število izračunov funkcije za neko natančnost:

O(n) O(n^2) … O(n^d)





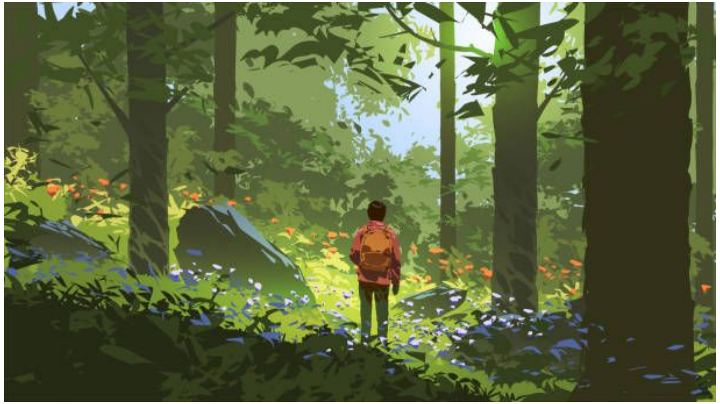
Monte Carlo: $$\int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} \overset{\boldsymbol{x}_i \sim p}{\approx} \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}_i)$$

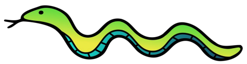


O(n^2)

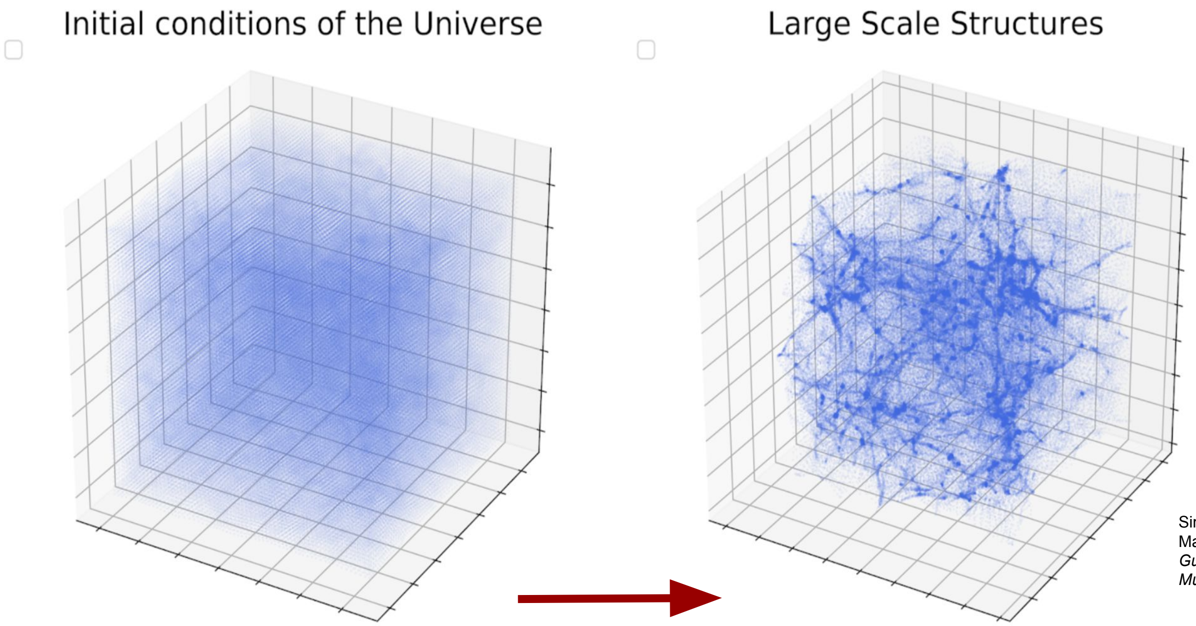
1. Marginalna porazdelitev

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}, \boldsymbol{z}) dz$$



	smrtno	ni smrtno
kobra	7 (5 %)	12 (8.5 %)
krait	8 (5.7 %)	13 (9.2 %)
Russellov gad	22 (15.6 %)	52 (36.9 %)
žagast gad	3 (2.1 %)	27 (19.1 %)
marginalno	26 %	74 %

Chaudhari, Tejendra S., et al. "Predictors of mortality in patients of poisonous snake bite: experience from a tertiary care hospital in central India." *International journal of critical illness and injury science* 4.2 (2014): 101.

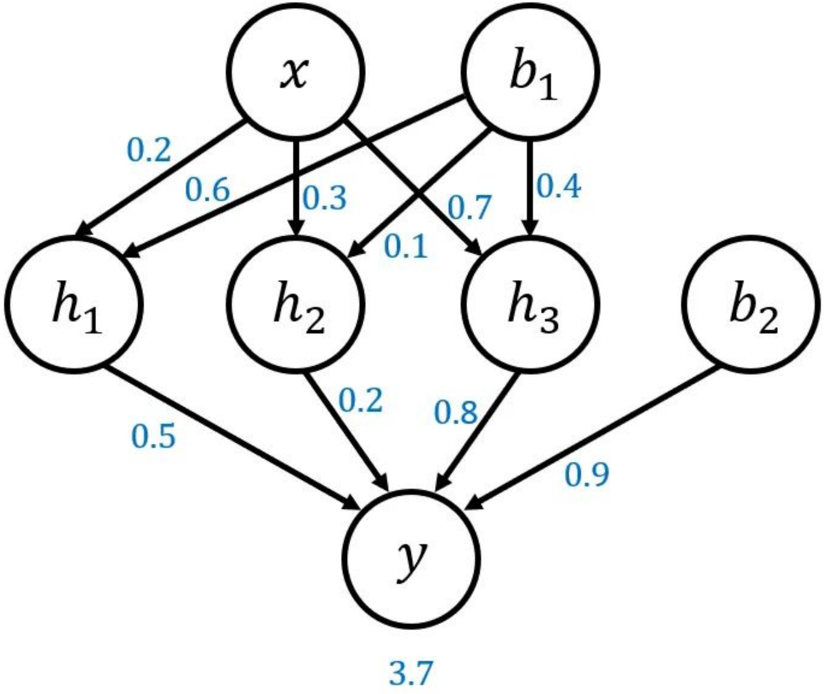


Simulating the Universe in TensorFlow
March 06, 2020
Guest post by Chirag Modi, François Lanusse, Mustafa Mustafa, Uroš Seljak

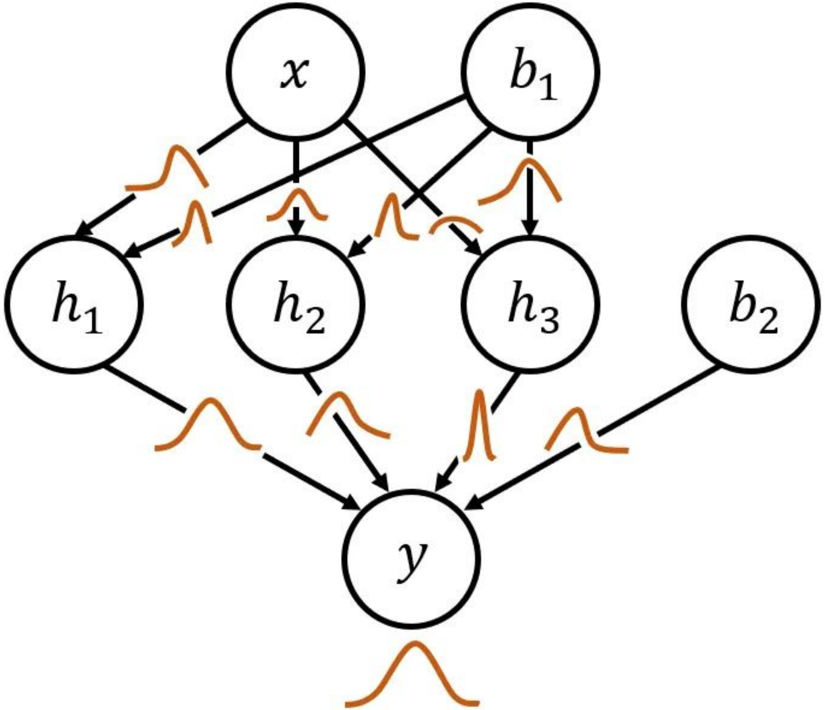
x = (začetni pogoji, gostota snovi, enačba stanja za temno energijo, …)

Bayesovsko sklepanje: $p(\boldsymbol{x}|\mathrm{data}) \propto p(\mathrm{data}|\boldsymbol{x})p(\boldsymbol{x})$

Standard Neural Network



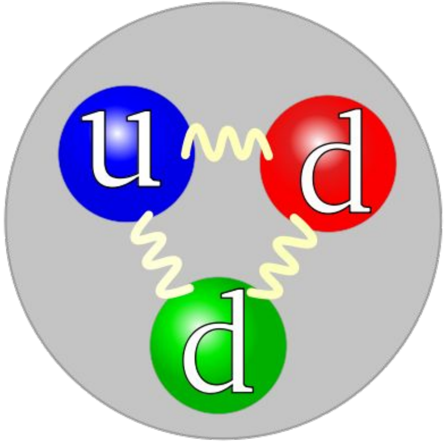
Bayesian Neural Network



Yeung Wong, Why You Should Use Bayesian Neural Network, Towards Data Science, 2021

2. Kvantna kromodinamika na mreži

- Naloga: izračun **mase sestavljenih delcev** (npr. protona) iz osnovnih zakonov
- **Integral prek vseh možnih konfiguracij** fermionskega in gluonskega kvantnega polja
- Diskretiziaramo prostor na mreži, polja aproksimiramo z visoko-dimenzionalnimi vektorji
- **>10^9** dimenzionalen integral, za izračun integranda potrebuješ **10^5 jeder**, to področje porablja **40% vse superračunalniške moči v ZDA**

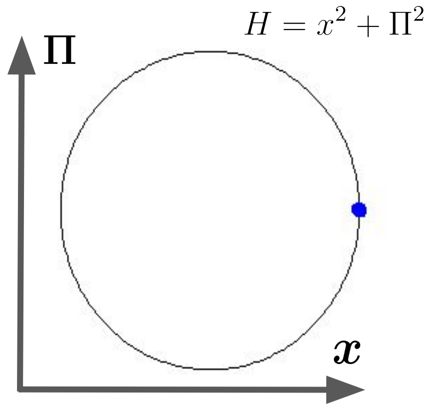


Kako vzorčiti?

Intermezzo: Hamiltonova dinamika

- Imejmo dva vektorja $\boldsymbol{x}, \boldsymbol{\Pi} \in \mathbb{R}^d$ in skalarno funkcijo $H(\boldsymbol{x}, \boldsymbol{\Pi})$
- Naj se časovno spreminjata po diferencialnih enačbah:

$$\frac{d\boldsymbol{x}}{dt} = \frac{\partial H(\boldsymbol{x}, \boldsymbol{\Pi})}{\partial \boldsymbol{\Pi}} \qquad \frac{d\boldsymbol{\Pi}}{dt} = -\frac{\partial H(\boldsymbol{x}, \boldsymbol{\Pi})}{\partial \boldsymbol{x}}$$



1. $\boldsymbol{x}(t), \boldsymbol{\Pi}(t)$ vedno ostaneta na ploskvi konstantnega H
2. **Ergodična hipoteza**: $p(\boldsymbol{x}, \boldsymbol{\Pi})$ je uniformna porazdelitev, omejena na ploskev konstanega H = **mikrokanonična porazdelitev**

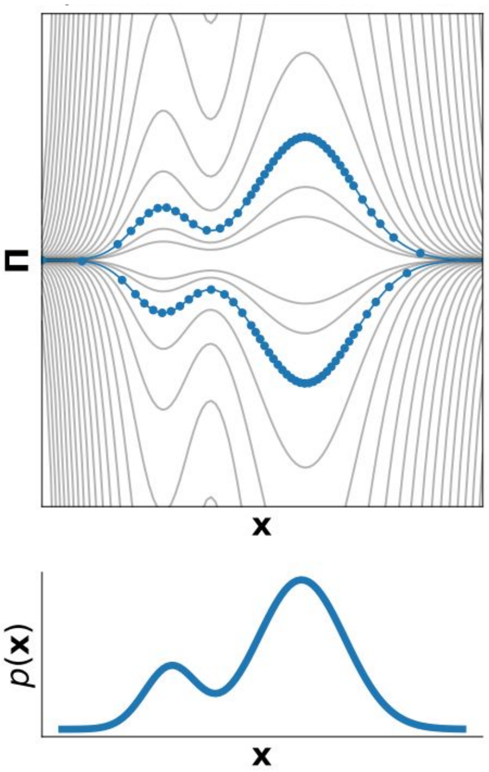
Znamo vzorčit iz **mikrokanonične** porazdelitve!

Je to dovolj za vzorčenje iz **poljubne** porazdelitve?

Mikrokanonični Hamiltonski Monte Carlo (MCHMC)

- Želimo vzorčit iz $p(\boldsymbol{x})$
- Vpeljemo dodatno spremenljivko $\boldsymbol{\Pi}$ in funkcijo $H(\boldsymbol{x}, \boldsymbol{\Pi})$
- Simuliramo Hamiltonovo dinamiko → konvergiramo k mikrokanonični porazdelitvi
- H nastavimo tako, da je **marginalna x-porazdelitev** $p(\boldsymbol{x})$:

$$H = \frac{d}{2} \log \frac{|\boldsymbol{\Pi}|^2}{d} - \log p(\boldsymbol{x})$$

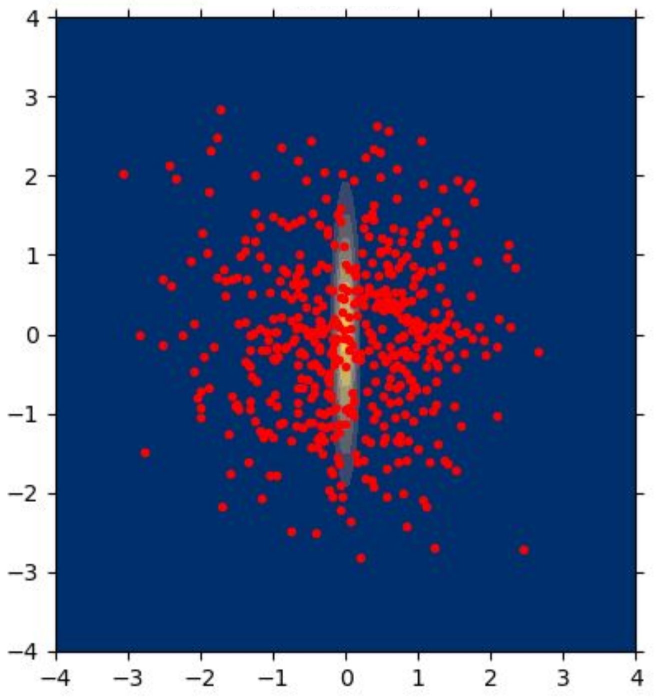


Implementacija

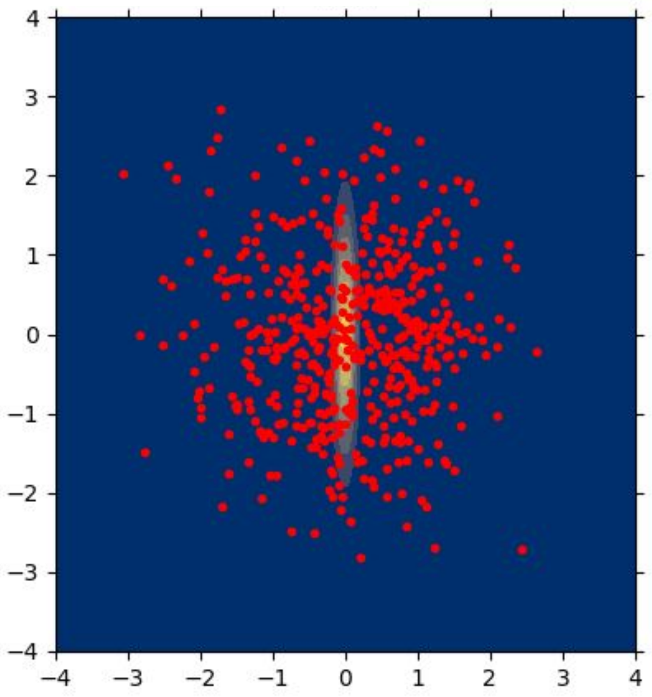
1. Integracija Hamiltonovih enačb: kako izbrat časovni korak? napaka na energiji ↔ pristranost vzorcev

2. Ergodičnost počasna ali je ni → dodajmo dekoherenco momenta! (pogoj tipa No U-turn)

Hamiltonova dinamika
+ šum na momentu

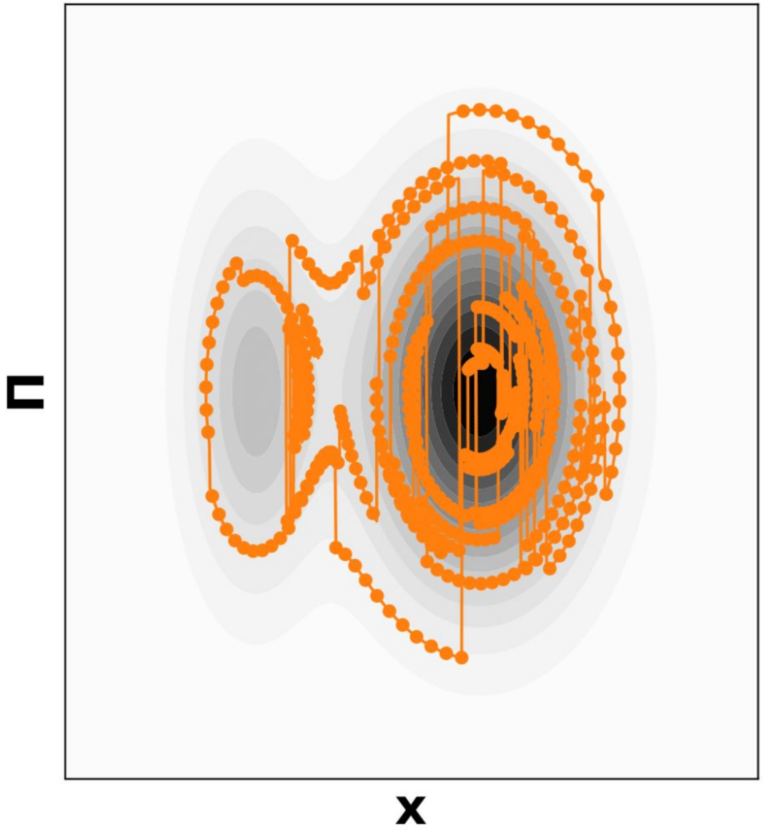


Hamiltonova dinamika



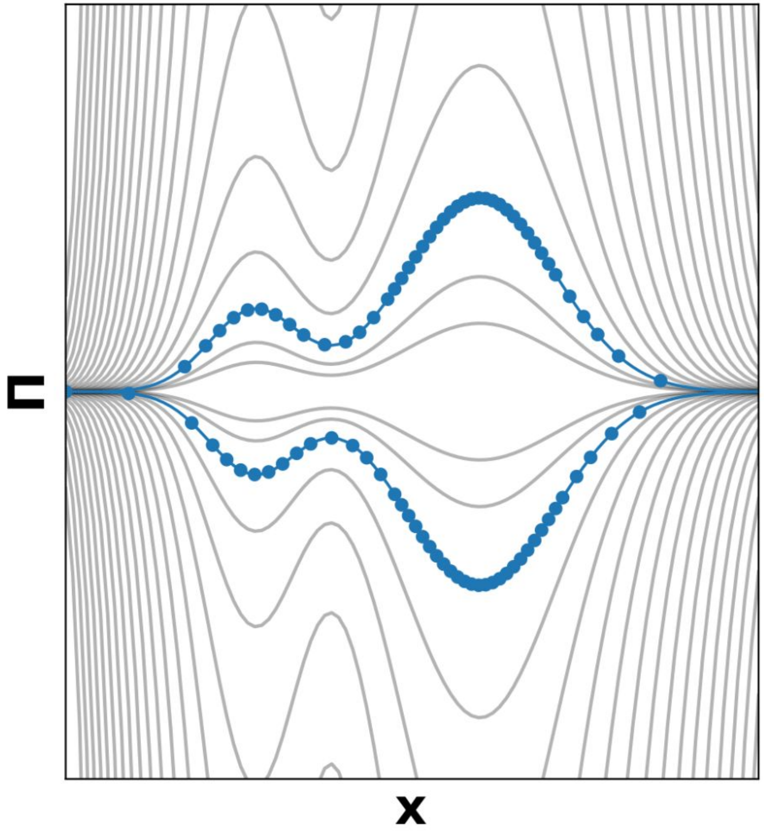
Canonical HMC

$p(\mathbf{x}, \mathbf{\Pi}) \propto e^{-H(\mathbf{x}, \mathbf{\Pi})}$



Microcanonical HMC

$p(\mathbf{x}, \mathbf{\Pi}) \propto \delta(H(\mathbf{x}, \mathbf{\Pi}) - E)$



HMC

```
Initialize x

For i in range(n)
    u[k]  ~ N(0, 1)
    xold = x

    For j in range(L):
        u = update_momentum(stepsize/2, u, grad_nlogp(x))
        x = update_position(stepsize, x, u)
        u = update_momentum(stepsize/2, u, grad_nlogp(x))

    x = acccept/reject(xold, x, energy change)
    save(x)
```

```
def update_momentum(eps, u, g):
    return u - eps * g
```

MCHMC

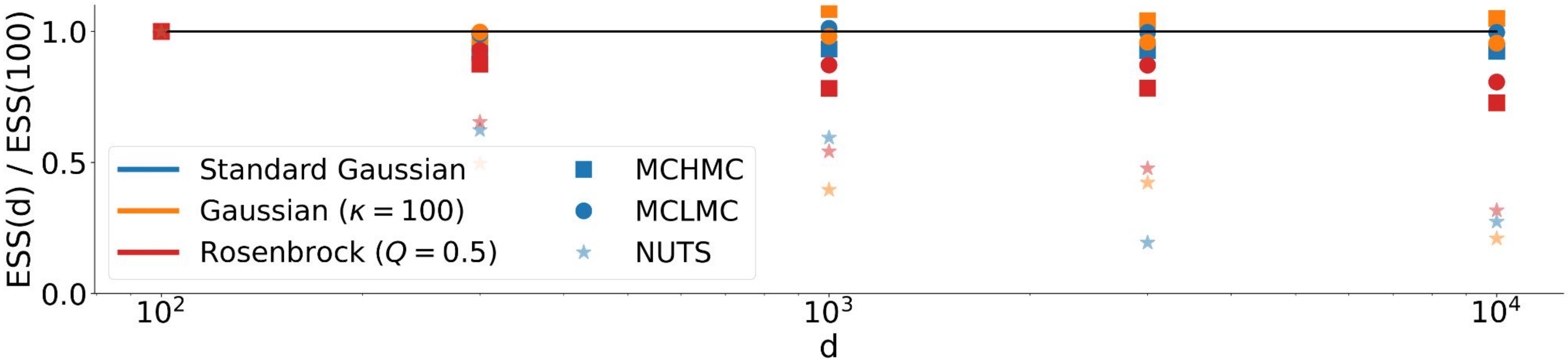
```
Initialize x

For i in range(n):
    u[k]  ~ N(0, 1)
    u = u / norm(u)

    For j in range(L):
        u = update_momentum(stepsize/2, u, grad_nlogp(x))
        x = update_position(stepsize, x, u)
        u = update_momentum(stepsize/2, u, grad_nlogp(x))

        save(x)
```

```
def update_momentum(eps, u, g):
    g_norm = sqrt(sum(square(g)))
    e = - g / g_norm
    ue = dot(u, e)
    delta = eps * g_norm / (d-1)
    zeta = exp(-delta)
    uu = e *(1-zeta)*(1+zeta + ue * (1-zeta)) + 2*zeta* u
    return uu / sqrt(sum(square(uu)))
```

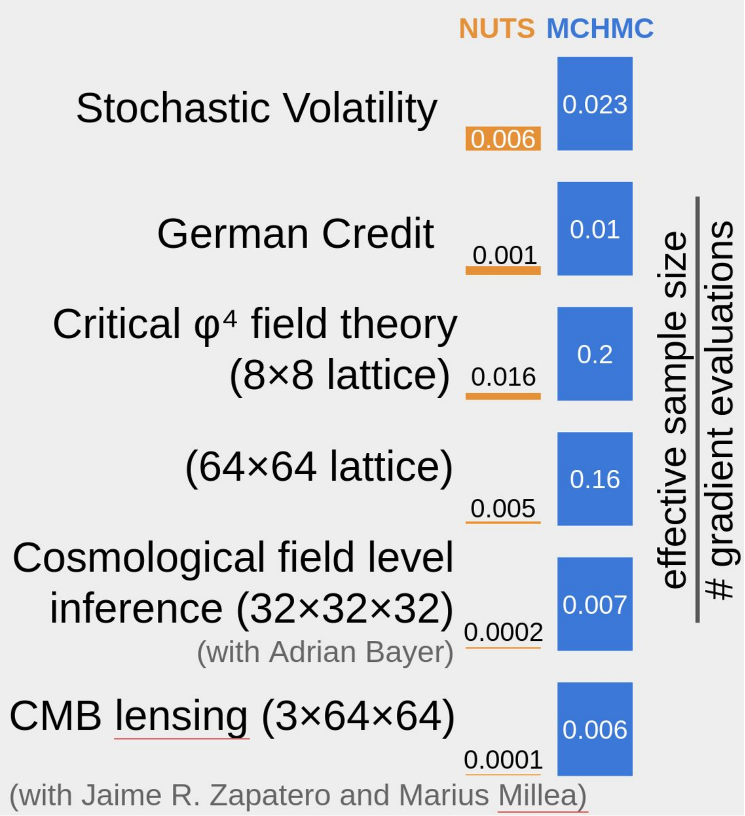
Želimo natančnost pričakovanih vrednosti: $\int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} \approx \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}_i)$

ESS = “učinkovitost” = “natančnost” / “računska cena”
= efektivno število vzorcev / število evalvacij ∇p

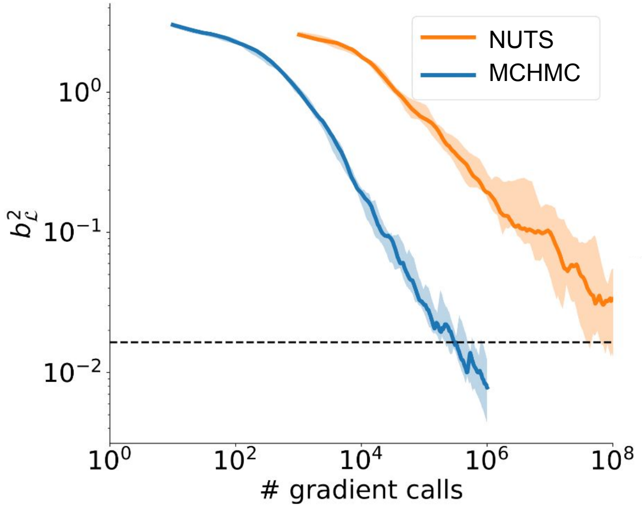
HMC: $O(d^{-1/4})$ MCHMC: $O(1)$



Učinkovitost neodvisna od dimenzije!

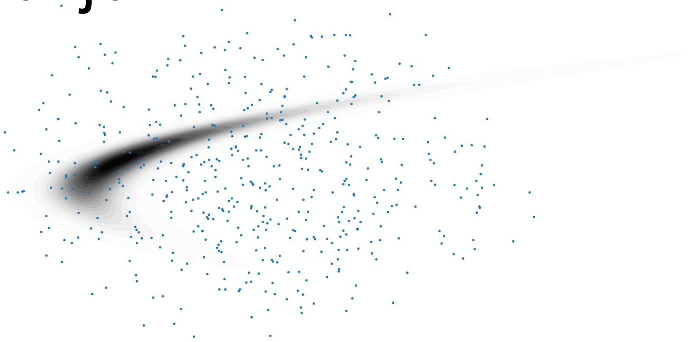


1000-dimenzionalna Cauchy-jeva porazdelitev



Paralelno vzorčenje

- Namesto veliko korakov z eno verigo,
  lahko delamo malo korakov z več verigami
- Odlično za GPU in clustre CPU-jev



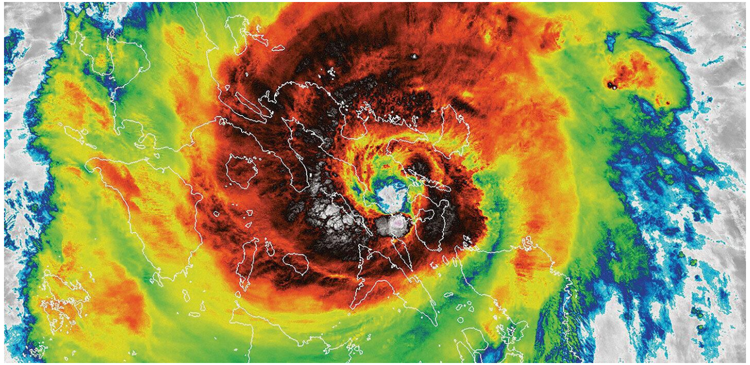
	NUTS	ChEES	MEADS	MCLMC
Banana	320	264	390	**22**
Ill Conditioned Gaussian	9846	5138	5520	**402**
Sparse Logistic Regression	2716	1168	610	**298**
Brownian Motion	2159	600	410	**122**
Item Response Theory	1162	537	790	**454**

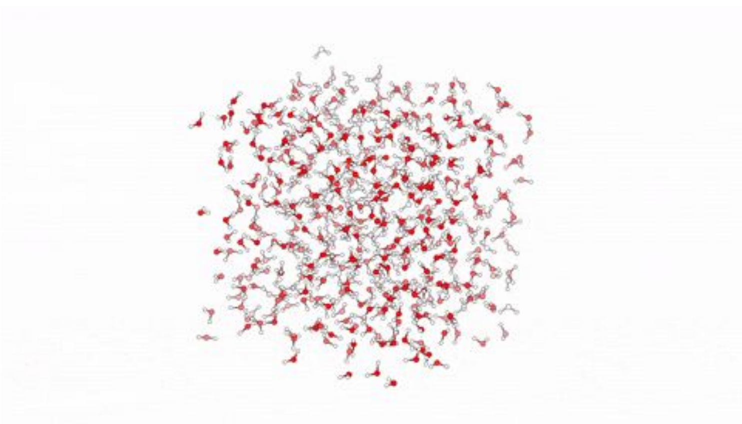
Table 1: Grads to low max bias ($b^2_{\text{max}} = 0.01$), normalized per chain.

Vzorčenje ni več drago!

Visoko-dimenzionalni problemi

- Napoved vremena
- Kvantna kromodinamika
- Bayesovsko sklepanje
- Bayesovske nevronske mreže
- Molekularna dinamika



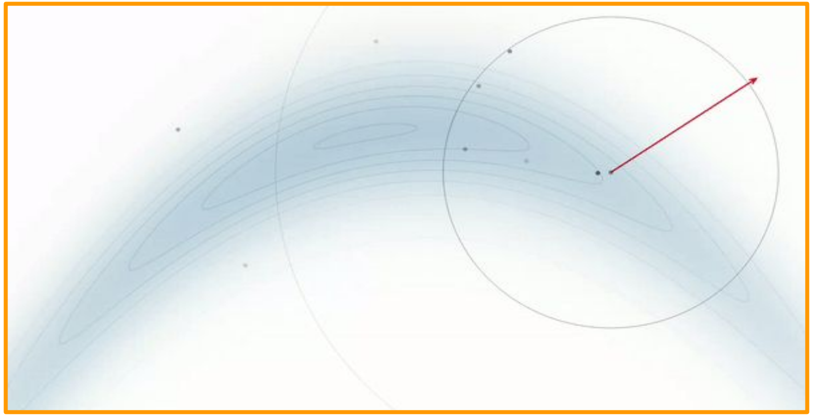


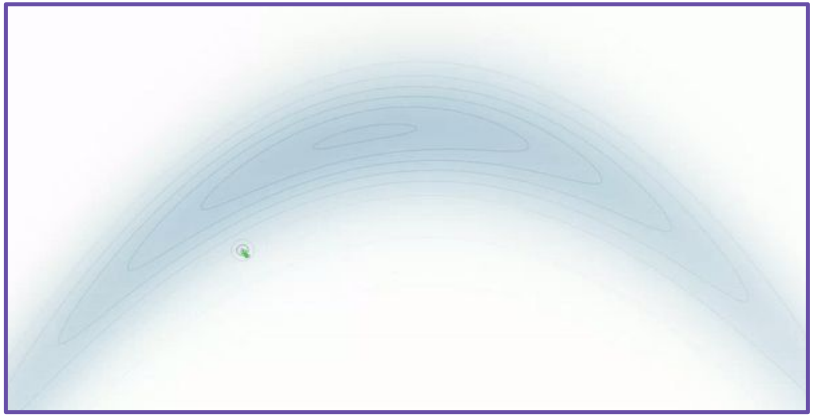
- Python (jax) in C++: https://github.com/JakobRobnik/MicroCanonicalHMC
- Julia: https://github.com/JaimeRZP/MicroCanonicalHMC.jl
- Spletna stran s tutoriali: https://main--zesty-daffodil-572c7d.netlify.app/
- Delamo na vključitvi v PPL-je kot so numpyro, STAN in pymc3.
- Članki:
  - Robnik, De Luca, Silverstein and Seljak, *Microcanonical Hamiltonian Monte Carlo*, 2022 (arXiv: 2212.08549)
  - Robnik and Seljak, *Microcanonical Langevin Monte Carlo*, 2023 (arXiv: 2303.18221)
  - Bayer, Seljak and Modi, *Field-Level Inference with Microcanonical Langevin Monte Carlo*, 2023 (arXiv: 2307.09504)

Prvi poskus: Metropolis-Hastings algoritem

- V vsakem koraku predlagamo nov vzorec $x_{n+1} \sim q(\cdot|x_n)$ in ga sprejmemo z verjetnostjo $min\{1, \frac{p(x_{n+1})q(x_n|x_{n+1})}{p(x_n)q(x_{n+1}|x_n)}\}$.

- Problem 1: če je korak predolg, so predlogi zavrnjeni, če je prekratek, se počasi premikamo
- Problem 2: ESS = **O(1/d)**



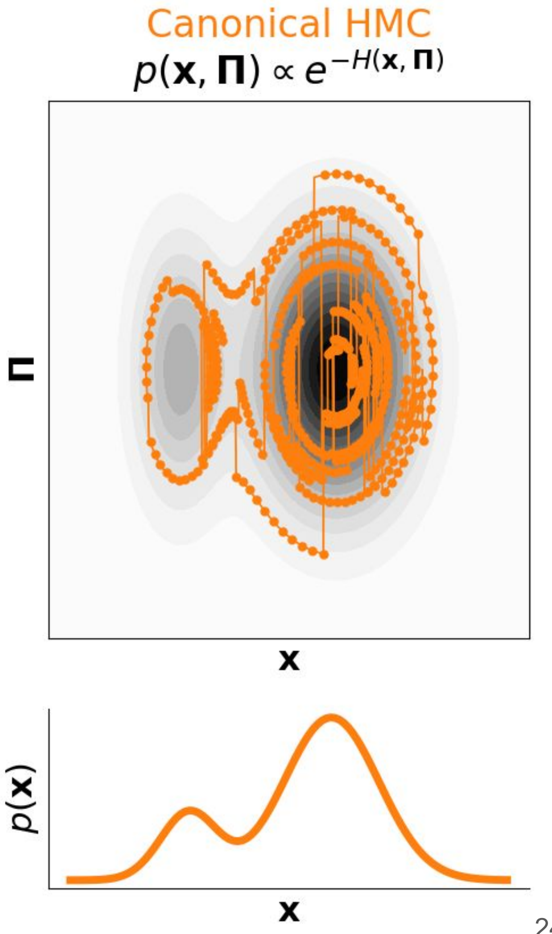


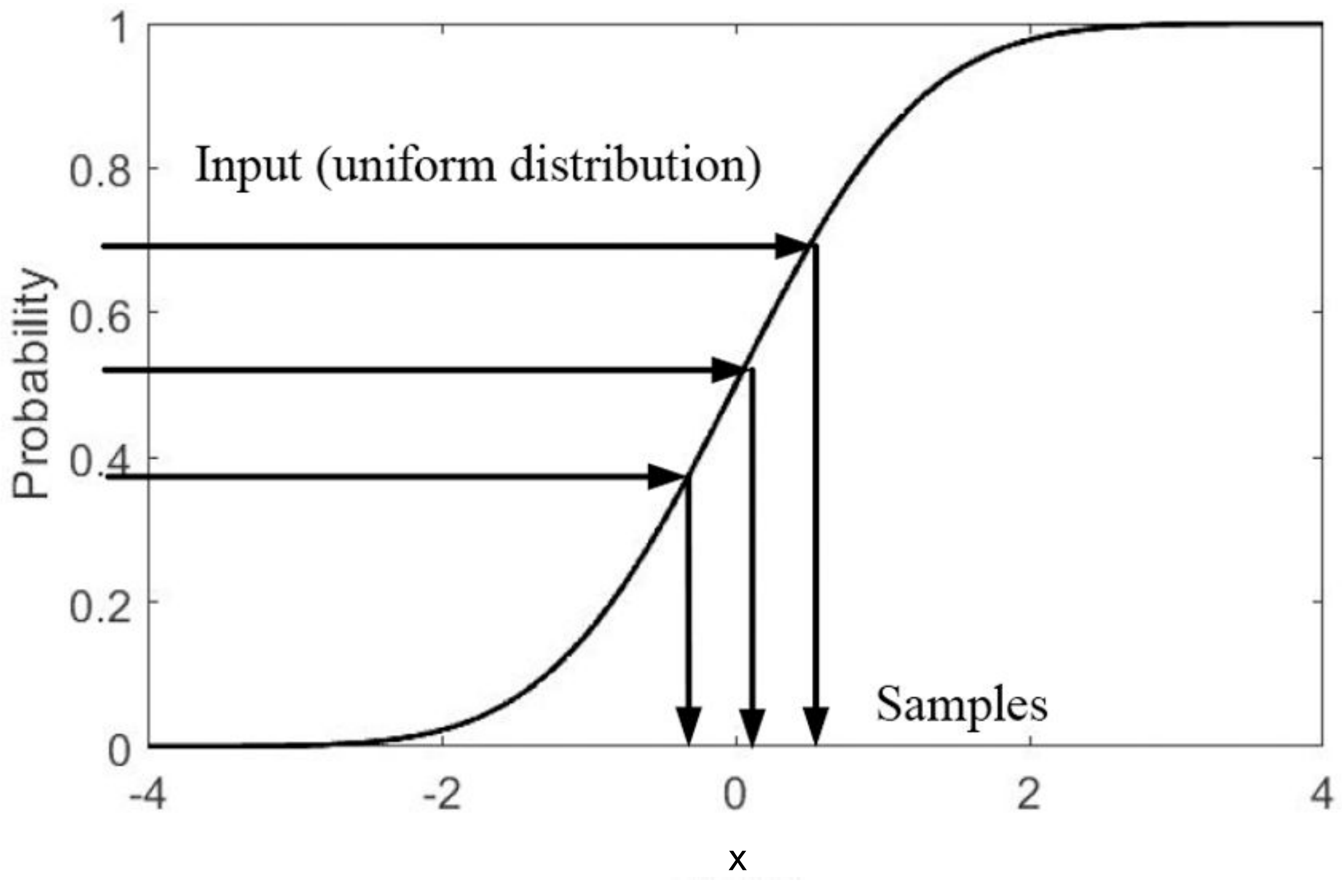
Chi-Feng, MCMC Interactive Gallery

Hamiltonski Monte Carlo (HMC)

- Želimo vzorčit iz $p(\boldsymbol{x})$
- Vpeljemo dodatno spremenljivko $\boldsymbol{\Pi}$ in funkcijo $H(\boldsymbol{x}, \boldsymbol{\Pi})$
- Hamiltonova dinamika omogoča učinkovito vzorčenje **mikrokanonične** porazdelitve
- Če dodamo občasno **prevzorčenje** momenta konvergiramo h **kanoninčni** porazdelitvi $p(\boldsymbol{x}, \boldsymbol{\Pi}) \propto e^{-H(\boldsymbol{x}, \boldsymbol{\Pi})}$
- H nastavimo tako, da je **marginalna x-porazdelitev** $p(\boldsymbol{x})$:

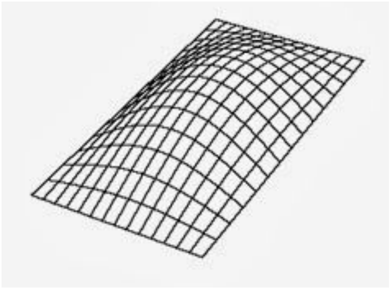
$$H = \frac{1}{2}|\boldsymbol{\Pi}|^2 - \log p(\boldsymbol{x})$$



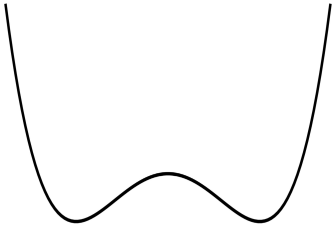

1
0.8
0.6
0.4
0.2
0
Probability
Input (uniform distribution)
Samples
-4
-2
0
2
4
x

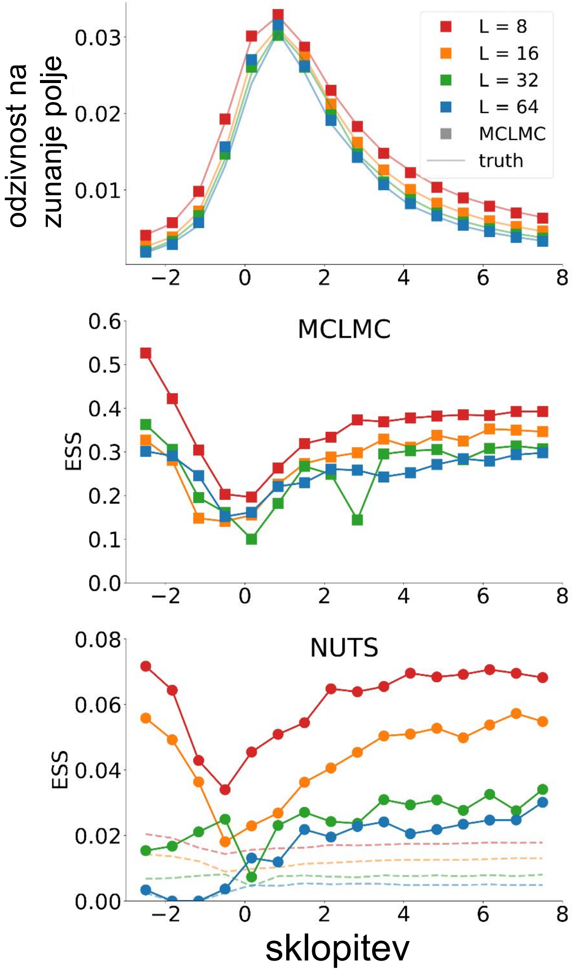
Kako to deluje na pravih problemih?

ϕ^4 teorija polja



fazni prehod





2. Statistična fizika: Boltzmanova porazdelitev

$$p(\boldsymbol{x}, \boldsymbol{\Pi}) \propto e^{-H(\boldsymbol{x}, \boldsymbol{\Pi})/T}$$

Formacija proteinov