Self-tuning Hamiltonian Monte Carlo

Henrik Christiansen
with Federico Errica and Francesco Alesiani

NEC Labs Europe, Heidelberg

November 2023

Hamiltonian Monte Carlo?!

Molecular Dynamics

- Discretization of Newton's equation $\Rightarrow$ effective shadow Hamiltonian and numeric errors
- Difficult to control temperature $\leftrightarrow$ Canonical in x and v? Disturbed dynamics?

Hamiltonian Monte Carlo?!

Molecular Dynamics

- Discretization of Newton's equation $\Rightarrow$ effective shadow Hamiltonian and numeric errors
- Difficult to control temperature $\leftrightarrow$ Canonical in x and v? Disturbed dynamics?

Monte Carlo

- "Directly" sample from target distribution
- Need "good" set of move proposals $\leftrightarrow$ no (direct) dynamic interpretation

Hamiltonian Monte Carlo?!

Molecular Dynamics

- Discretization of Newton's equation $\Rightarrow$ effective shadow Hamiltonian and numeric errors
- Difficult to control temperature $\leftrightarrow$ Canonical in x and v? Disturbed dynamics?

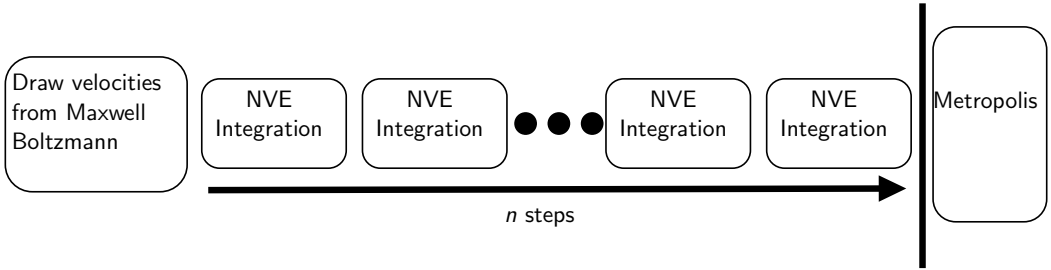
Monte Carlo

- "Directly" sample from target distribution
- Need "good" set of move proposals $\leftrightarrow$ no (direct) dynamic interpretation

Hamiltonian Monte Carlo

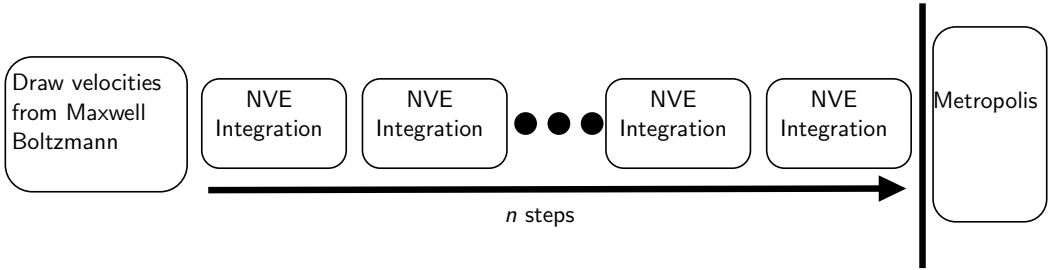
- What if we combine elements of both? Profit?
- Provides structured way to set-up moves based on microcanonical MD simulations
- Guaranteed to sample from the target distribution by Monte Carlo accept/reject

How does it work?



A priori, arbitrary integrator can be used. We restrict ourselves to velocity verlet.

How does it work?



A priori, arbitrary integrator can be used. We restrict ourselves to velocity verlet. At least two free parameters: number of integrations steps n and timestep Δt of integrator $\rightarrow$ performance of HMC crucially depends on these parameters

Target distribution:

$$P^{\mathrm{eq}}(\boldsymbol{x}) = \frac{1}{Z} e^{-U(\boldsymbol{x})/(kT)}$$

Target distribution:

$$P^{\mathrm{eq}}(\boldsymbol{x}) = \frac{1}{Z} e^{-U(\boldsymbol{x})/(kT)}$$

Metropolis criterion ($\boldsymbol{\xi} = (\boldsymbol{x}, \boldsymbol{v})$):

$$P_{\mathrm{acc}} = \min\left(1, \frac{P^{\mathrm{eq}}(\boldsymbol{\xi}_j)}{P^{\mathrm{eq}}(\boldsymbol{\xi}_i)}\right)$$

Target distribution:

$$P^{\mathrm{eq}}(\boldsymbol{x}) = \frac{1}{Z} e^{-U(\boldsymbol{x})/(kT)}$$

Metropolis criterion ($\boldsymbol{\xi} = (\boldsymbol{x}, \boldsymbol{v})$):

$$P_{\mathrm{acc}} = \min\left(1, \frac{P^{\mathrm{eq}}(\boldsymbol{\xi}_j)}{P^{\mathrm{eq}}(\boldsymbol{\xi}_i)}\right)$$

NVE Integrator (velocity verlet):

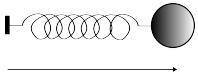
$$\boldsymbol{v}_i\left(t + \frac{1}{2}\Delta t_i\right) = \boldsymbol{v}_i(t) + \frac{1}{2}\boldsymbol{a}_i(t)\Delta t_i \quad (1a)$$

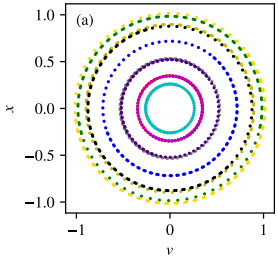
$$\boldsymbol{x}_i(t + \Delta t_i) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i\left(t + \frac{1}{2}\Delta t_i\right)\Delta t_i \quad (1b)$$

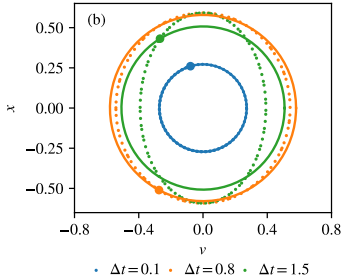
Recalculate $\boldsymbol{a}_i(t + \Delta t_i)$ from the new positions with $\boldsymbol{a}_i = -1/m\partial/\partial \boldsymbol{x}_i U(\boldsymbol{x})$ (1c)

$$\boldsymbol{v}_i(t + \Delta t_i) = \boldsymbol{v}_i\left(t + \frac{1}{2}\Delta t_i\right) + \frac{1}{2}\boldsymbol{a}_i(t + \Delta t_i)\Delta t_i \quad (1d)$$

One Dimensional Harmonic Oscillator





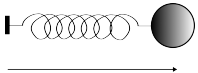


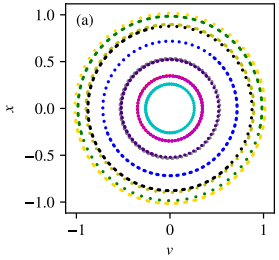
- Hamiltonian (in $d = 1$ with $m = k = 1$)

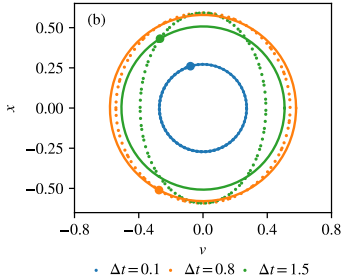
$$\mathcal{H} = U(x) + T(v) = 0.5(x^2 + v^2),$$

simulated at constant temperature $T = 0.5$

One Dimensional Harmonic Oscillator





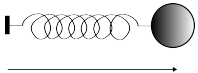


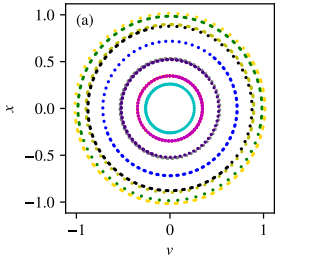
- Hamiltonian (in $d = 1$ with $m = k = 1$)

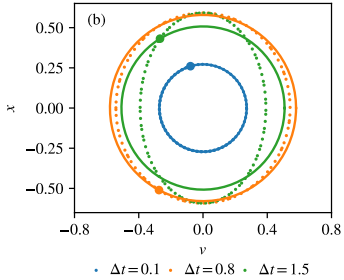
$$\mathcal{H} = U(x) + T(v) = 0.5(x^2 + v^2),$$

  simulated at constant temperature $T = 0.5$
- Every new proposal is obtained by setting a different total energy level via the picking of velocities from Maxwell-Boltzmann, as shown in (a).

One Dimensional Harmonic Oscillator







- Hamiltonian (in $d = 1$ with $m = k = 1$)

$$\mathcal{H} = U(x) + T(v) = 0.5(x^2 + v^2),$$

  simulated at constant temperature $T = 0.5$
- Every new proposal is obtained by setting a different total energy level via the picking of velocities from Maxwell-Boltzmann, as shown in (a).
- Using different Δt has major effect on simulated system ("shadow" Hamiltonian), see (b) $\rightarrow$ reflected in acceptance rate of MH.

What are ideal parameters of HMC?

- General goal for tuning MC: Reduce autocorrelation $\leftrightarrow$ smaller autocorrelation time τ for some observable $\mathcal{O}$ $\leftrightarrow$ Can run shorter simulations to get same accuracy.

What are ideal parameters of HMC?

- General goal for tuning MC: Reduce autocorrelation $\leftrightarrow$ smaller autocorrelation time τ for some observable $\mathcal{O}$ $\leftrightarrow$ Can run shorter simulations to get same accuracy.
- Problem for tuning: Need "long" time series to estimate autocorrelation time τ.

What are ideal parameters of HMC?

- General goal for tuning MC: Reduce autocorrelation $\leftrightarrow$ smaller autocorrelation time τ for some observable $\mathcal{O}$ $\leftrightarrow$ Can run shorter simulations to get same accuracy.
- Problem for tuning: Need "long" time series to estimate autocorrelation time τ.
- Solution: Come up with proxy "loss" of τ that can be calculated "locally"

$$L_n = -p_n |\boldsymbol{x}'_n - \boldsymbol{x}_0|^b$$

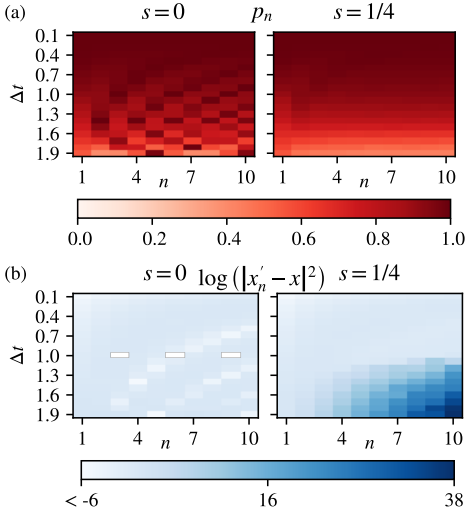
What are ideal parameters of HMC?

- General goal for tuning MC: Reduce autocorrelation $\leftrightarrow$ smaller autocorrelation time τ for some observable $\mathcal{O}$ $\leftrightarrow$ Can run shorter simulations to get same accuracy.
- Problem for tuning: Need "long" time series to estimate autocorrelation time τ.
- Solution: Come up with proxy "loss" of τ that can be calculated "locally"

$$L_n = -p_n |\boldsymbol{x}'_n - \boldsymbol{x}_0|^b$$

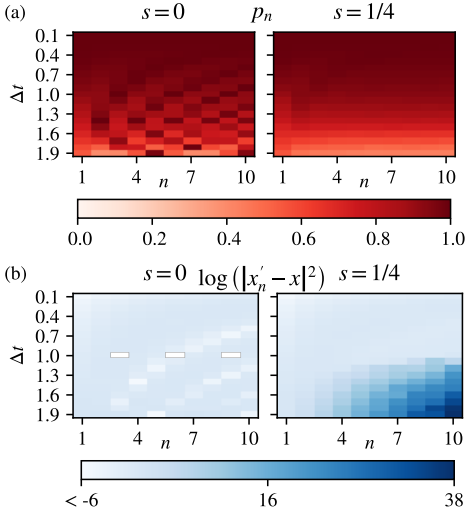
- This loss multiplies the acceptance rate p_n times the movement in spatial space $|\boldsymbol{x}'_n - \boldsymbol{x}_0|$ to some power b
  - b controls how important large jumps are compared to small ones

Jitter vs no-jitter



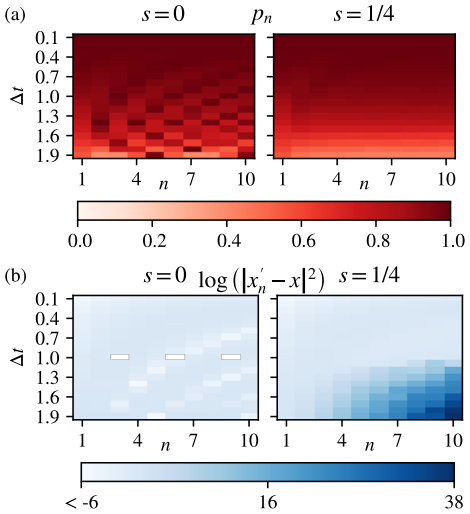
- Several minima/maxima in the acceptance rate and jump distance.

Jitter vs no-jitter



- Several minima/maxima in the acceptance rate and jump distance.
- Unsuited as "loss" since we would just run into the next local minima.

Jitter vs no-jitter

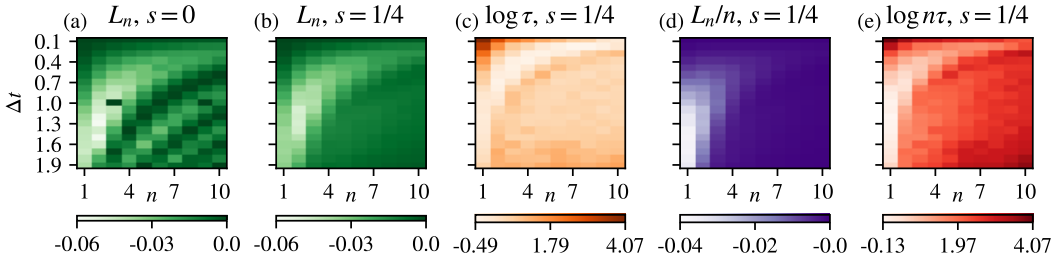


- Several minima/maxima in the acceptance rate and jump distance.
- Unsuited as "loss" since we would just run into the next local minima.
- Solution: Introduce "jittering" of Δt, i.e., pick for every iteration the timestep from a normal distribution

$$\Delta t' \sim \mathcal{N}(\Delta t, s\Delta t),$$

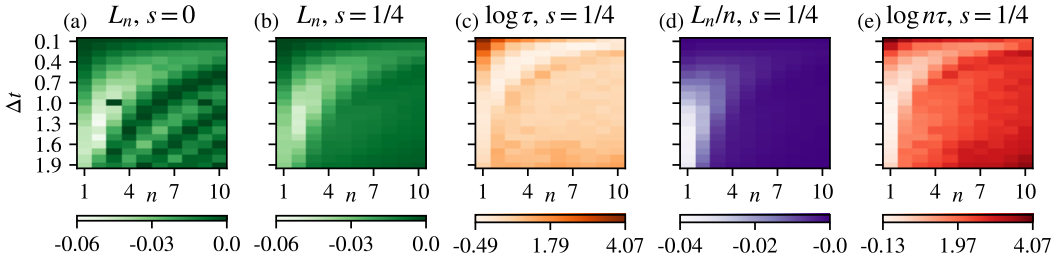
with relative variance s.

Loss definition



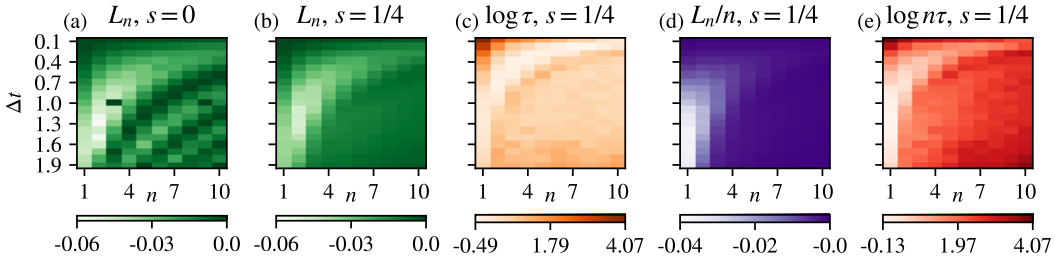
- With "jittering" smooth loss surface (a) vs (b)

Loss definition



- With "jittering" smooth loss surface (a) vs (b)
- Relatively well aligned with autocorrelation time of potential energy τ

Loss definition



- With "jittering" smooth loss surface (a) vs (b)
- Relatively well aligned with autocorrelation time of potential energy τ
- "Real objective": Autocorrelation time in units of computational effort $n\tau$, so consider "loss" per computational effort L_n/n

How do we learn?

- Fully differentiable framework + automatic differentiation + backpropagation (Adam)

How do we learn?

- Fully differentiable framework + automatic differentiation + backpropagation (Adam)
- What about learning the number of integration steps (not directly part of computation graph)?

How do we learn?

- Fully differentiable framework + automatic differentiation + backpropagation (Adam)
- What about learning the number of integration steps (not directly part of computation graph)?
- Use attention like loss function:

$$L = \sum_{n=1}^{N} c_n L_n / n,$$

where c_n are the weights of the particular number of integration steps and N is the maximal number of integration steps considered during training.

How do we learn?

- Fully differentiable framework + automatic differentiation + backpropagation (Adam)
- What about learning the number of integration steps (not directly part of computation graph)?
- Use attention like loss function:

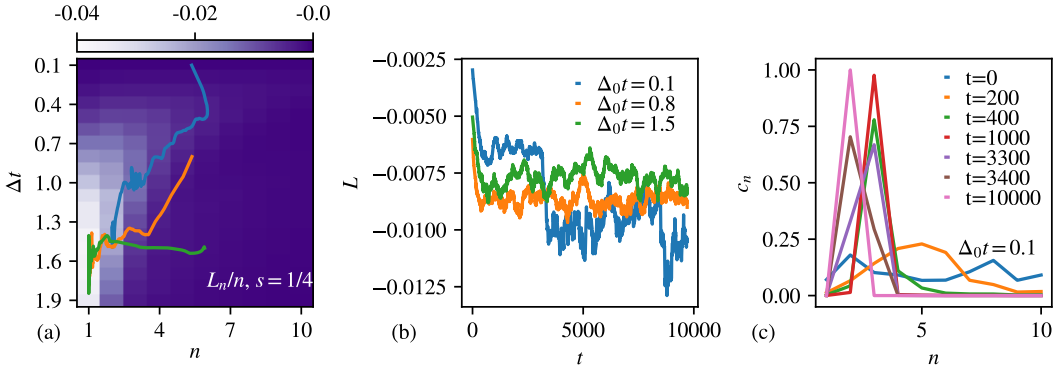
$$L = \sum_{n=1}^{N} c_n L_n / n,$$

where c_n are the weights of the particular number of integration steps and N is the maximal number of integration steps considered during training.

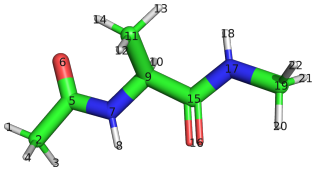
- The c_n are in practice obtained as softmax of unrestricted parameters C_n, i.e.,

$$c_n = \sigma(W_n) = \frac{e^{C_n}}{\sum_{n=1}^{N} e^{C_n}}$$

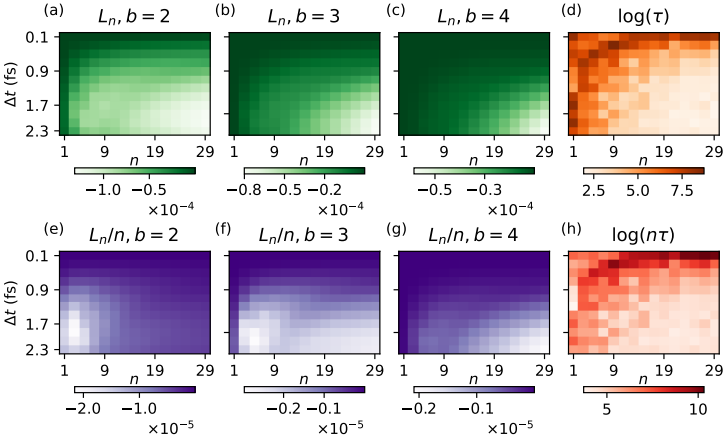
Learning Trajectories for the Harmonic Oscillator



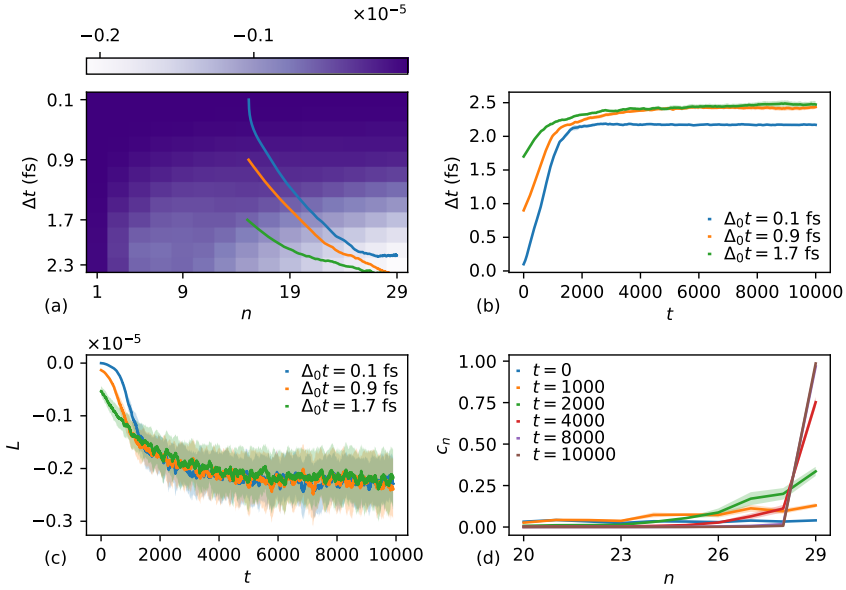
Alanine dipeptide loss surface and autocorrelation



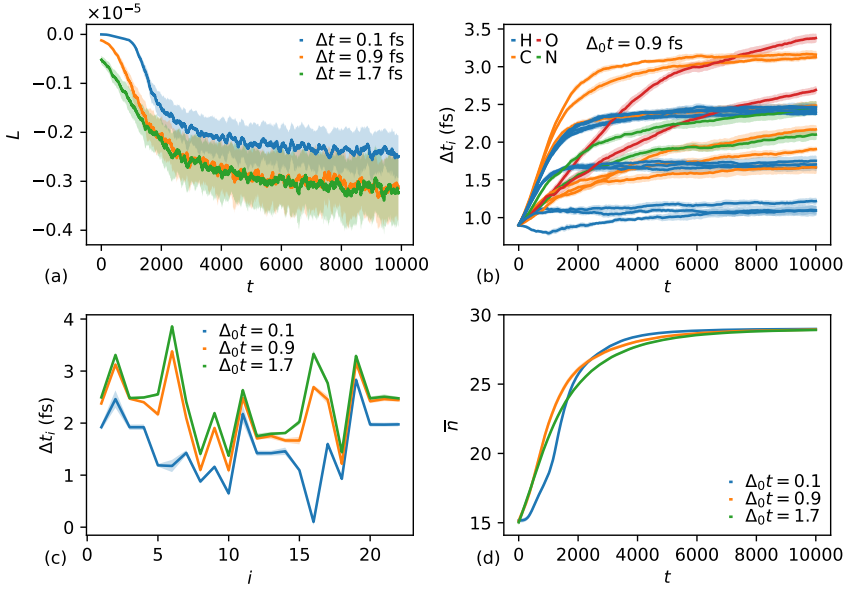
- Small peptide with 22 atoms.
- Common test system for novel methods.
- Simulated at $T = 300$ K.



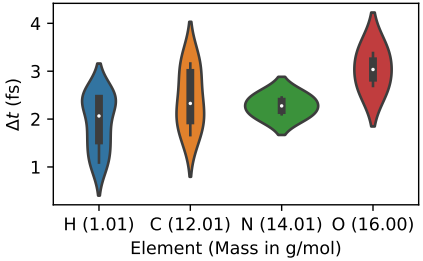
Learning for alanine dipeptide



Atom dependent timestep



Timesteps and Performance



$\Delta_0 t$	0.1 fs	0.9 fs	1.7 fs
τ for atom based Δt	12.7(2.6)	7.5(9)	7.5(1)
τ for global Δt	12.1(1.8)	10.0(1.0)	9.9(1.3)

Conclusion & Outlook

- Fully differentiable framework for learning both classical parameters of HMC

Conclusion & Outlook

- Fully differentiable framework for learning both classical parameters of HMC
- Extended to atom-based timesteps – heuristic optimizers would have a hard time

Conclusion & Outlook

- Fully differentiable framework for learning both classical parameters of HMC
- Extended to atom-based timesteps – heuristic optimizers would have a hard time
- Atom-based timesteps provide speed-up of simulation – without additional runtime penalty at time of "production"

H. Christiansen, F. Errica, F. Alesiani, J. Chem. Phys. **159**, 234109 (2023)


Orchestrating a brighter world
NEC