Tuning HMC parameters with gradients

James C. Osborn
Argonne National Laboratory

Lattice 2023, FNAL

Tuning HMC Parameters

- Hybrid Monte Carlo (HMC)
  - Choose random gaussian momentum, ***p***
  - Integrate Hamiltonian, $H(p,U) = p^2/2 + S(U)$, along equations of motion
    - $p', U' = F(p, U, \alpha)$
    - α = integration time, τ, other MD parameters
  - Conditionally accept new configuration
    - $P_{acc}(p', U', p, U) = \min[\,1, \exp(-\Delta H)\,]$
    - $\Delta H = H(p', U') - H(p, U)$
- Tuning parameters effects efficiency of update
  - Integration time, step size, higher order schemes, multiple time scales
  - Also action parameters (fermion action w/ Hasenbusch ratios)
- Can be tuned by hand, using measurements of error terms, forces
- Experimenting with autotuning using gradient information

Integrator parameters

- For example: 2 step integrator
  - ABABA pattern
  - A = gauge field update
  - B = momentum update with force term
- $e^{\varepsilon (A+B) + E} = e^{\varepsilon\lambda A} e^{\varepsilon/2\, B} e^{\varepsilon(1-2\lambda) A} e^{\varepsilon/2\, B} e^{\varepsilon\lambda A}$
  - $E = \varepsilon^3 \{ (1-6\lambda+6\lambda^2)/12\, [A,[B,A]] + (1-6\lambda)/24\, [B,[B,A]] \} + \varepsilon^5 \ldots$
- Minimum norm solution (I. P. Omelyan, I. M. Mryglod, R. Folk 2001)
  - Assume similar magnitude of operators ***[A,[B,A]]*** and ***[B,[B,A]]***
  - Minimize norm of error term ($\lambda \approx 0.193\ldots$)
  - In LQCD, terms typically not equal, need to tune
- Can also consider higher order integrators (force gradient) with more parameters

Cost function

- MD integration time: τ
- After ***N*** HMC updates (momentum refresh, accept/reject)
  - effective MD integration time $\approx \tau\, sqrt(<P_{acc}>N)$
- Number of updates needed to go ***T*** effective MD time units
  - $N_T = T^2 / (<P_{acc}>\, \tau^2)$
- Cost $= N_T \times$ (cost per update)
- For simplicity using
  - (cost per update) = # force evaluations (N_{force})
  - $T = 1$

Loss function optimization

- Want to minimize
  - Cost = $N_{force} / (\langle P_{acc} \rangle \tau^2)$
- Involves average in denominator, inconvenient
- Instead minimize
  - Loss $= - \langle P_{acc} \rangle \tau^2$
- Using methods from ML
  - Calculate gradient of loss, use to update parameters
  - Using Adam optimizer, accumulates gradient during update
  - Single update stream (no batch)
  - Currently updating parameters after every step

Calculating gradients

$\varepsilon = \tau / n$

- Need gradient of P_{acc} , function of $H(p',U') \equiv H'$
  - $(p', U') = [\, e^{\varepsilon\lambda A}\, e^{\varepsilon/2\, B}\, e^{\varepsilon(1-2\lambda)\, A}\, e^{\varepsilon/2\, B}\, e^{\varepsilon\lambda A}\,]^n\, (p, U)$
- Using back propagation
- Generate gradients from chain rule
- $\partial H'/\partial\alpha = \sum_k (\partial H'/\partial q_k)\, (\partial q_k/\partial\alpha$ [fixed q_{k-1}]$)$
- $q_k = (p_k\, , U_k\,)$, state after k^{th} update step (A or B)
- $\partial H'/\partial q_k = (\partial H'/\partial q_{k+1}\,)\, (\partial q_{k+1}/\partial q_k)$
- Need gradients of all update steps (gauge update, forces, …)
- Need to save all intermediate states

Gradient of gauge force

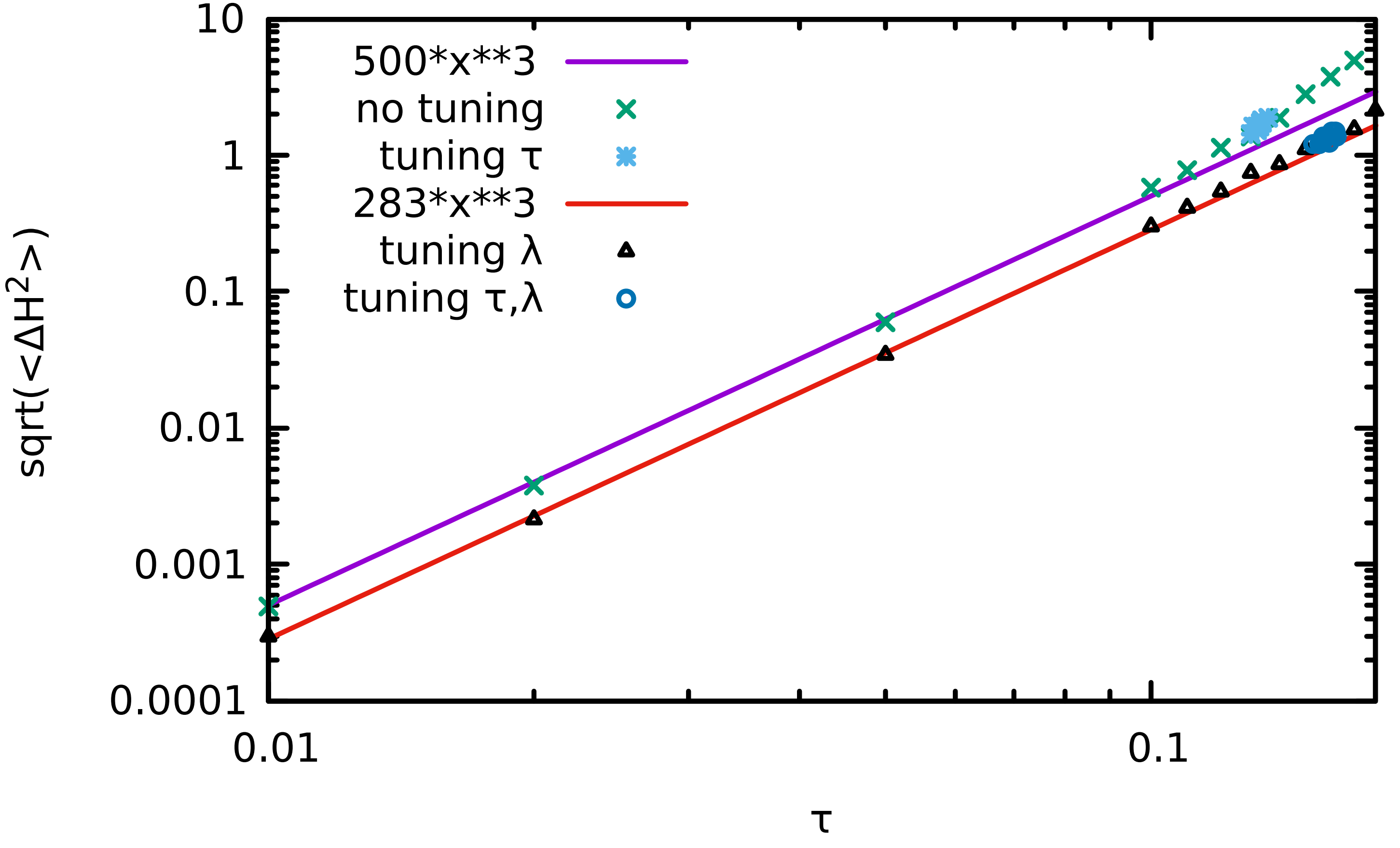
- Gauge force
  - Main piece is calculating staples
  - $S = U_\nu(x)\, U_\mu(x+\nu)\, U^\dagger_\nu(x+\mu)$ + **backward staple**
- Chain rule
  - $dH'/dU = (dH'/dS)\,(dS/dU) = C\, dS/dU$
  - $dH'/dU = C_\nu(x)\, U_\mu(x+\nu)\, U^\dagger_\nu(x+\mu)$
    $+ U_\nu(x)\, C_\mu(x+\nu)\, U^\dagger_\nu(x+\mu)$
    $+ U_\nu(x)\, U_\mu(x+\nu)\, C^\dagger_\nu(x+\mu)$
    + **backward**
- Also need adding/scaling/multiplying fields, traceless anti-Hermitian projection
  - Relatively easy to work out gradients

Tests on pure gauge

- Implemented in QEX
  - LFT framework in Nim language
  - https://github.com/jcosborn/qex
  - Using simple "tape" implementation, save list of operations
  - Run forwards to do update, run backwards for gradient
- 12^3 x 24 lattice (running on single desktop)
- Plaquette action at β = 5.6
- Starting from thermalized config
- 200 tuning updates, 400 measurement updates

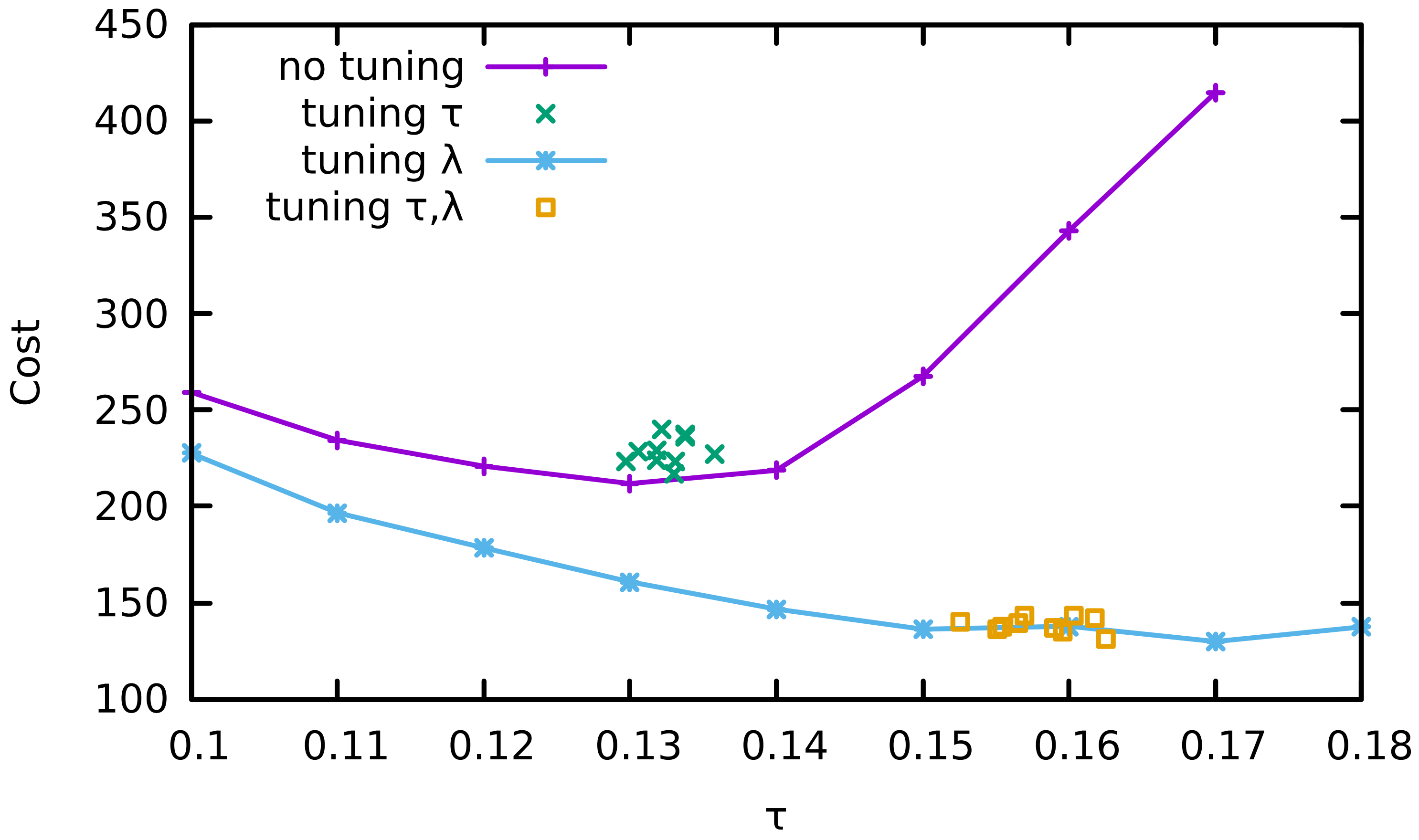
Gauge ABABA: Error vs. trajectory length

Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)



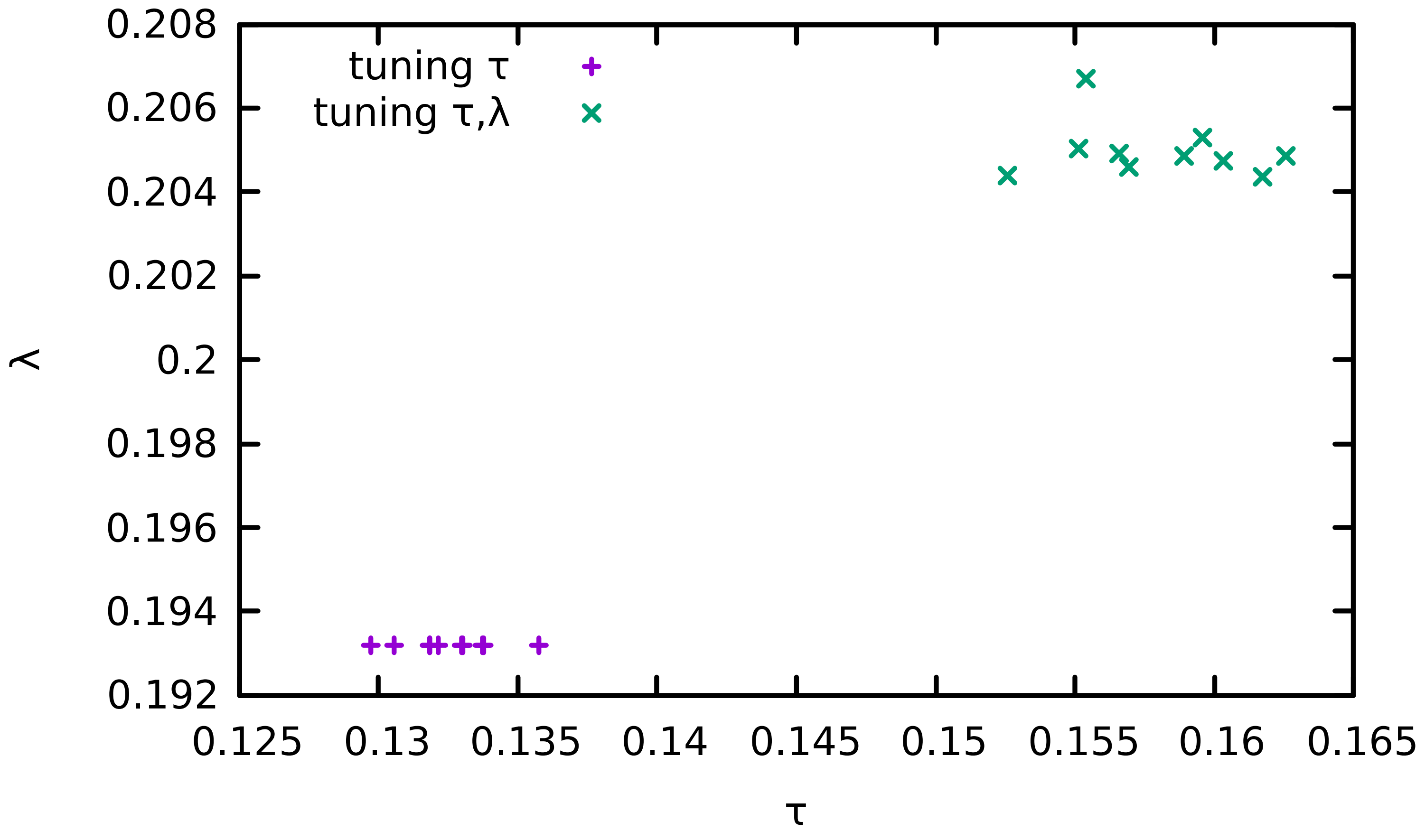
Gauge ABABA: Cost vs. trajectory length

Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)



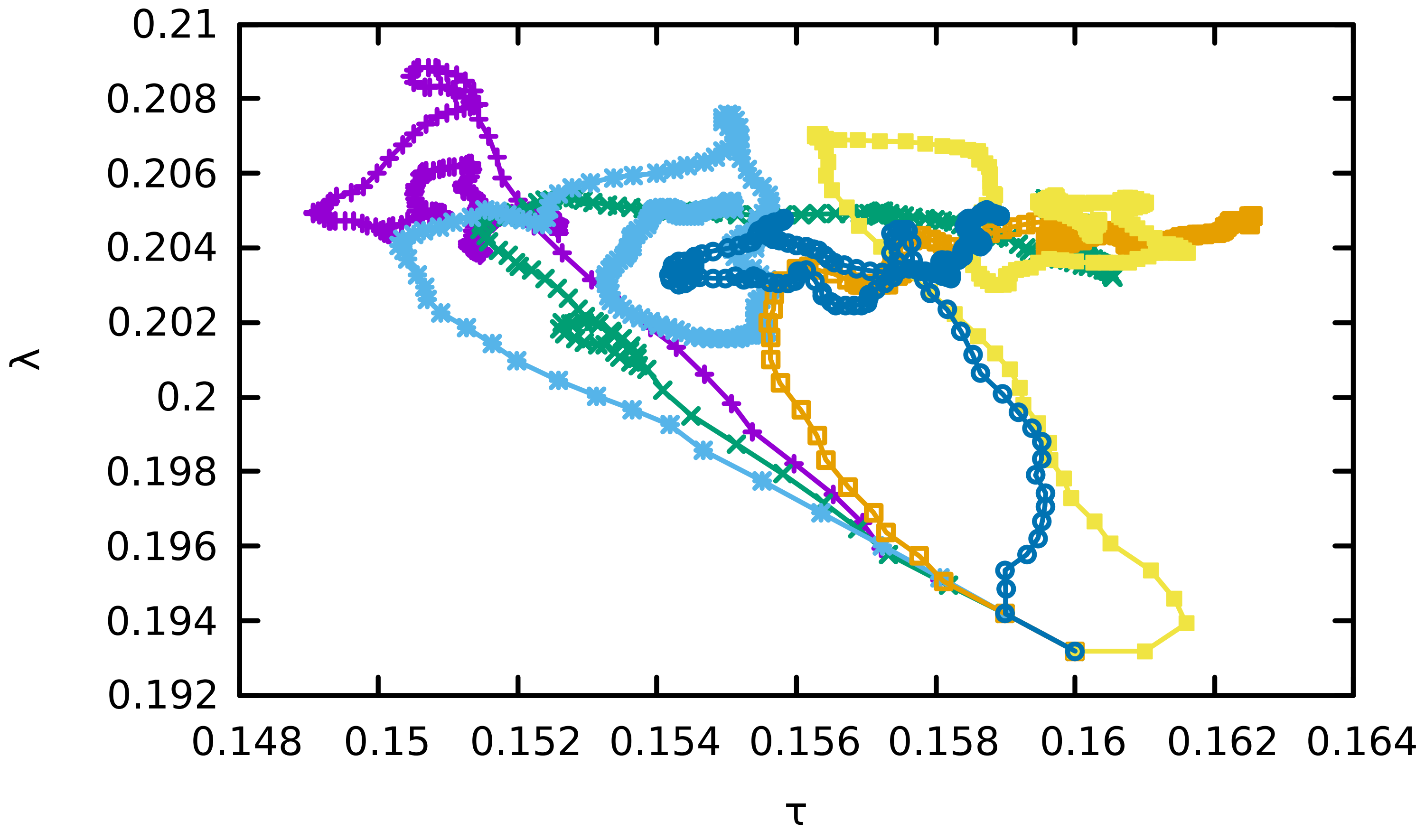
Gauge ABABA: Tuned parameters

Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)

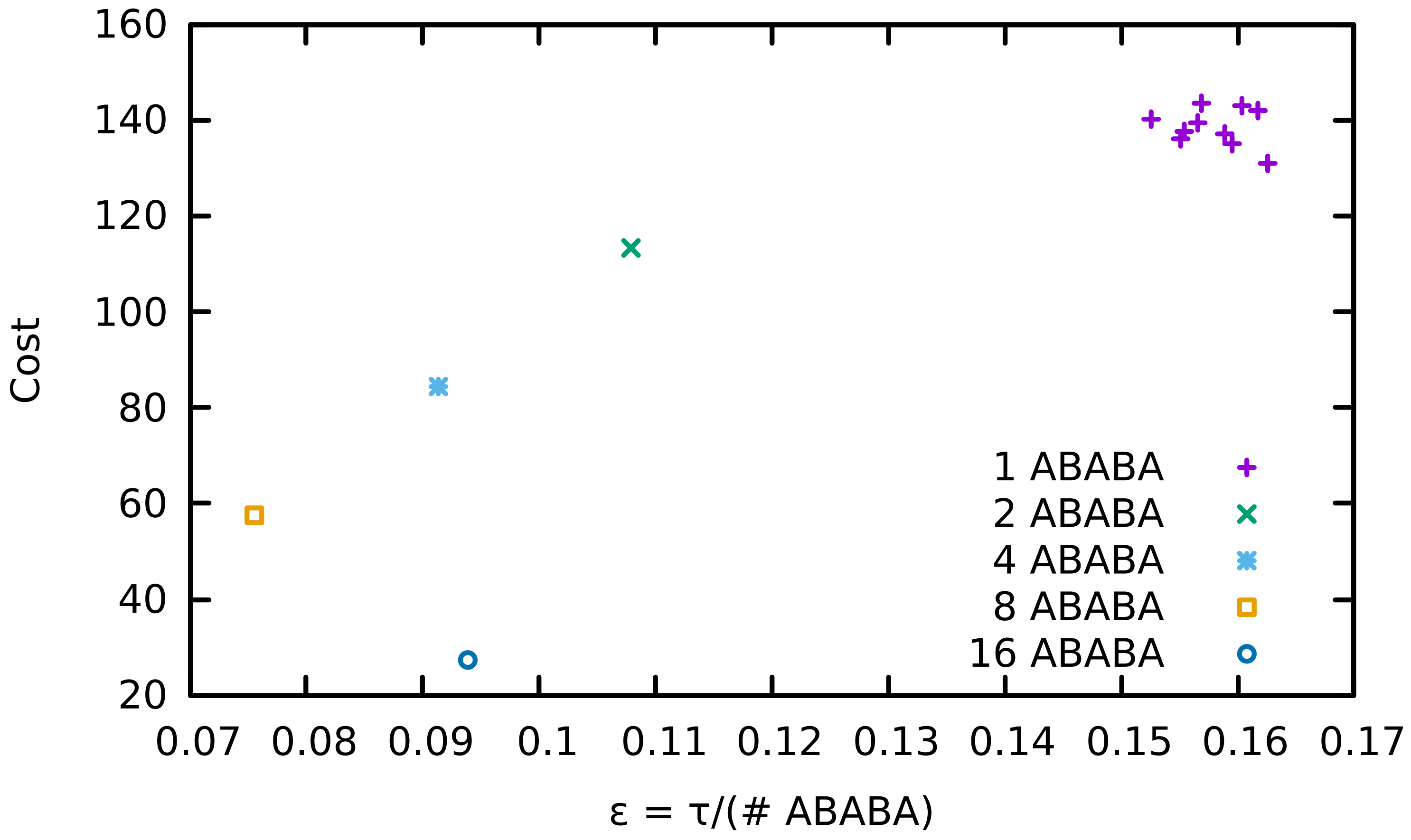


Gauge ABABA: Parameter paths during tuning

Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)



Gauge ABABA: Multiple copies



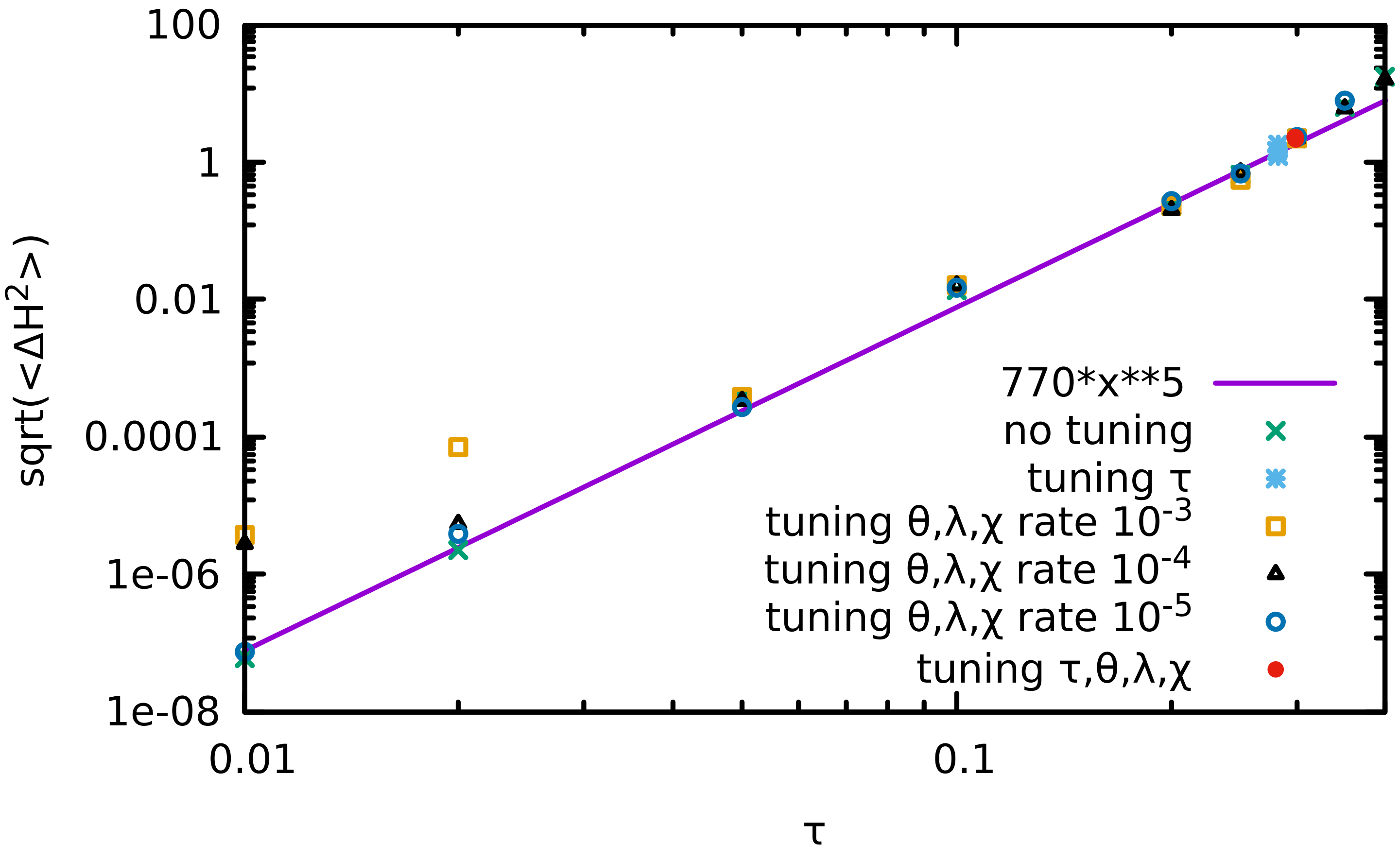
4^{th} order integrator

- Using Omelyan, et al.'s recommended 4th order integrator

  (I. P. Omelyan, I. M. Mryglod, R. Folk 2002)
- Uses force gradient term
  - Approximated using 2 force evaluations (H. Yin, R. D. Mawhinney 2011)
- ABACABA pattern
  - C = force gradient
  - $e^{\varepsilon\vartheta A}\, e^{\varepsilon\lambda B}\, e^{\varepsilon/2(1-2\vartheta) A}\, e^{\varepsilon(1-2\lambda) B + (\varepsilon\text{^}3)\chi C}\, e^{\varepsilon/2(1-2\vartheta) A}\, e^{\varepsilon\lambda B}\, e^{\varepsilon\vartheta A}$
  - 3 parameters: ϑ, λ, χ
  - Minimum norm solution cancels 3^{rd} order error, minimizes 5^{th} order

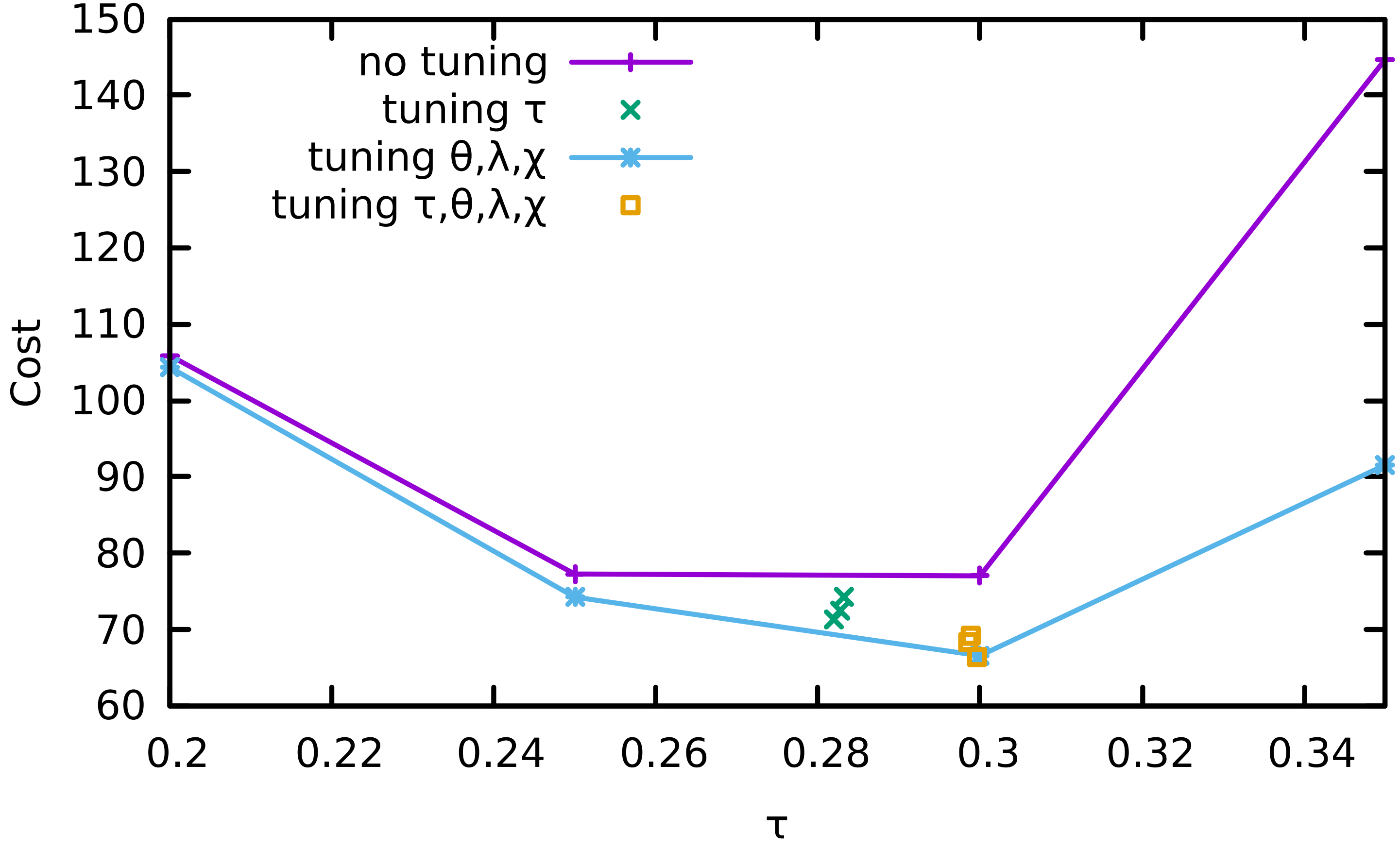
Gauge ABACABA: Error vs. trajectory length

Single integrator copy (n=1, $\varepsilon = \tau$)



Gauge ABACABA: Cost vs. trajectory length

Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)



Gradient of fermion force

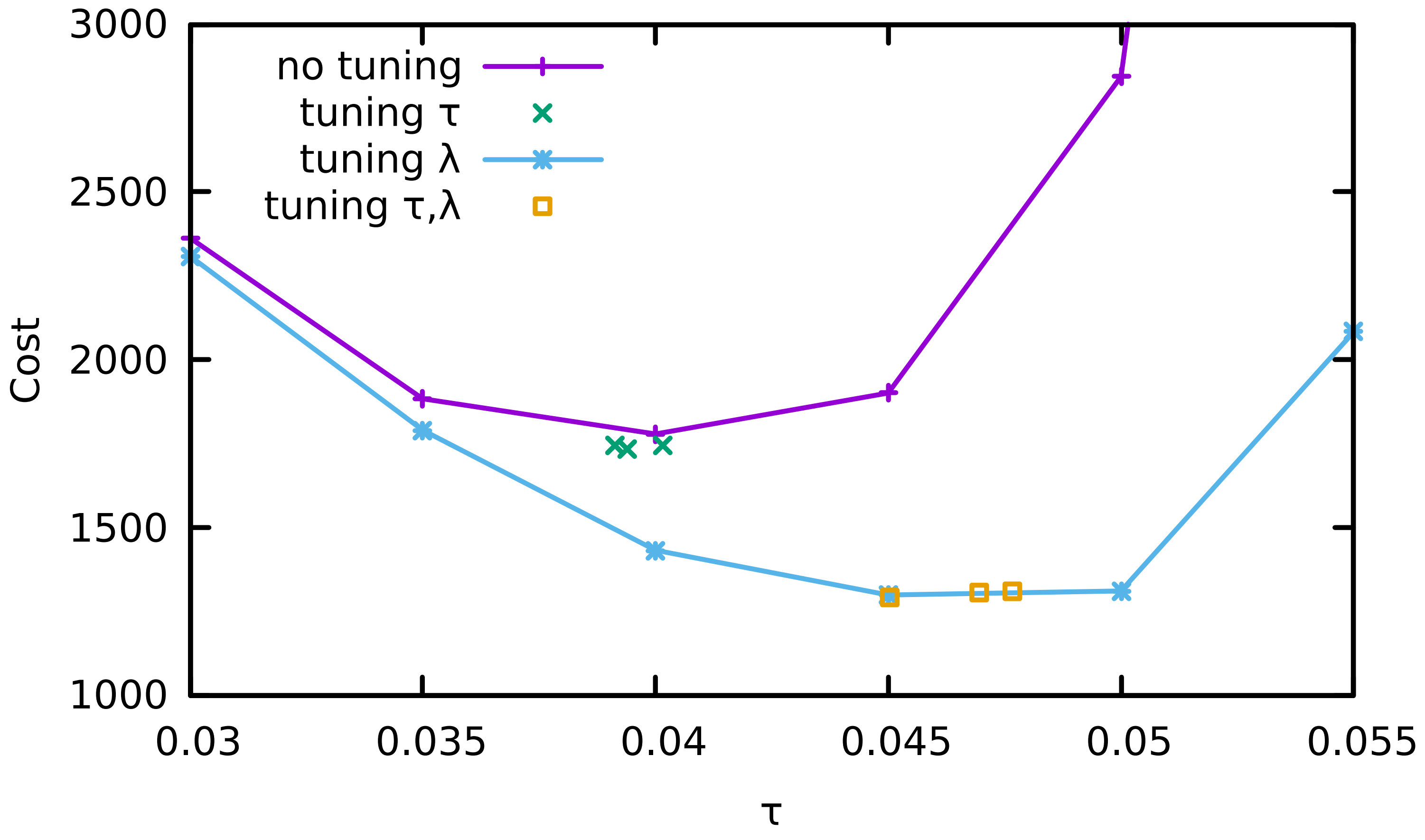
- Action term
  - $S = \phi^\dagger (D^\dagger D)^{-1} \phi$
- Force
  - $F_X = -\partial S/\partial U_X = \phi^\dagger (D^\dagger D)^{-1} (\partial D^\dagger D/\partial U_X) (D^\dagger D)^{-1} \phi + h.c.$
- Gradient of momentum update
  - $\partial H'/\partial U_Y = (\partial H'/\partial F_X) (\partial F_X/\partial U_Y) = C_X \, \partial F_X/\partial U_Y$
- Break into 2 pieces
  - $\psi = (D^\dagger D)^{-1} \phi \quad \Rightarrow \quad C_W \, \partial\psi_W/\partial U_Y = \phi^\dagger (D^\dagger D)^{-1} (\partial D^\dagger D/\partial U_Y) (D^\dagger D)^{-1} C_W$
  - $C_X \, \partial/\partial U_Y \, \psi^\dagger (\partial D^\dagger D/\partial U_X) \psi = \partial/\partial U_Y \, \psi^\dagger (D^\dagger D_C + D_C^\dagger D) \psi$
  - $D_C = C_X \, \partial D/\partial U_X$

Fermion tests

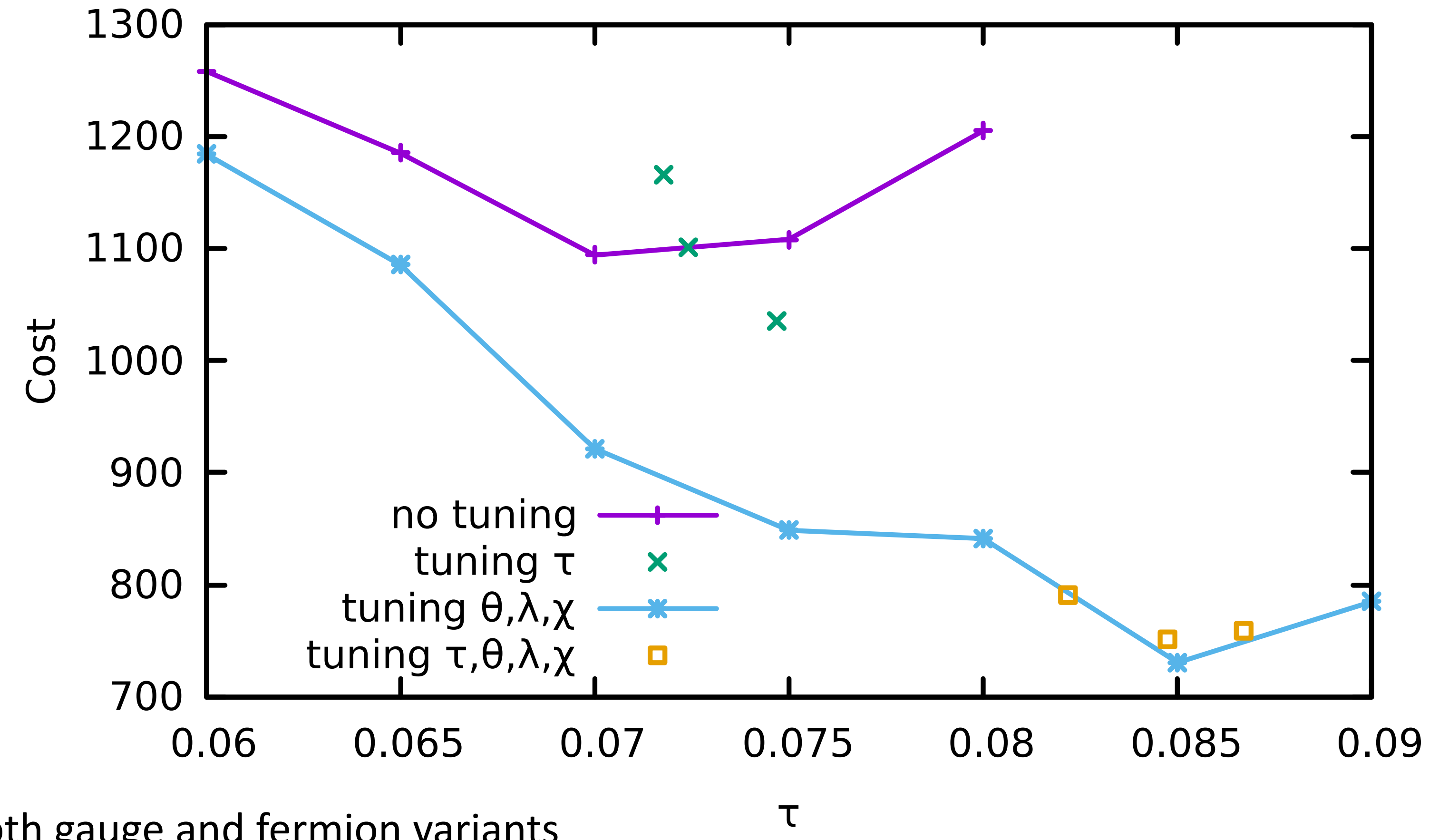
- Plain staggered, no rooting, 4 continuum flavors
- 12^3 x 24 lattice
- β = 5.6
- m = 0.04
  - a M_π ~ 0.5
- Cost function only counts fermion force evaluations (not gauge)

Staggered ABABA: Cost vs. trajectory length

Single integrator copy (n=1, ***ε = τ***)

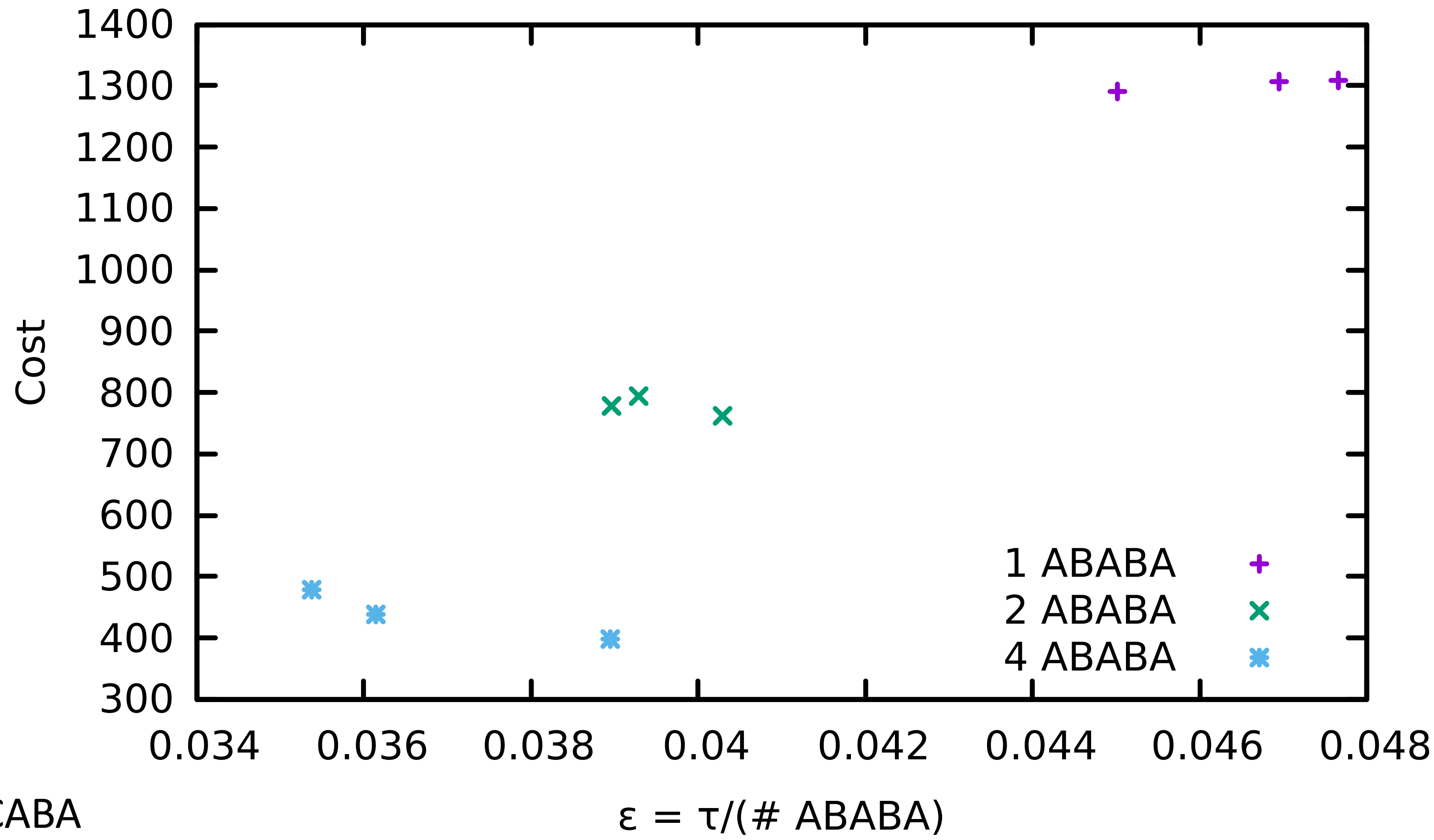


Staggered ABACABA: Cost vs trajectory length
Single integrator copy (n=1, $\boldsymbol{\varepsilon = \tau}$)



Note: λ and χ have both gauge and fermion variants

Staggered ABABA: Multiple copies



Note 2 ABABA ≈ 1 ABACABA
on small volume,
but only after tuning

Summary

- Tuning HMC parameters using gradient information seems to work well in simple cases
- Very convenient way to tune HMC,
  after initial investment in implementation
- Force gradient integrators may work much better when tuned, could be competitive even on small volumes
- Need to try with mass parameters in Hasenbusch ratios
- Plan to implement other actions (improved gauge, smearing)