Automated tuning for HMC mass ratios

J. Torsiello[1,2], G. Fleming[3], S. Foreman[1], X.-Y. Jin[1] and J.C. Osborn[1]

[1] Argonne National Laboratory

[2] Temple University

[3] Fermilab

Lattice 2024, Liverpool

Previous work on autotuning HMC parameters

- Demonstrated autotuning of HMC parameters
  - ML methods + gradient of full HMC trajectory over parameters
- Tuned MD trajectory length, $\boldsymbol{\tau}$, and integrator parameters
- Tested 2 MD integrators (***A***=Kinetic, ***B***=Forces, ***C***=Force gradient, $\boldsymbol{\varepsilon = \tau / n_s}$)

$$e^{\varepsilon(A+B)} \approx e^{\varepsilon\lambda A}\, e^{\frac{\varepsilon}{2}B}\, e^{\varepsilon(1-2\lambda)A}\, e^{\frac{\varepsilon}{2}B}\, e^{\varepsilon\lambda A} \quad \text{(ABABA) tune } \lambda$$

$$e^{\varepsilon(A+B)} \approx e^{\varepsilon\theta A}\, e^{\varepsilon\lambda B}\, e^{\frac{\varepsilon}{2}(1-2\theta)A}\, e^{\varepsilon(1-2\lambda)B+\varepsilon^3\chi C} e^{\frac{\varepsilon}{2}(1-2\theta)A}\, e^{\varepsilon\lambda B} e^{\varepsilon\theta A} \quad \text{(ABACABA) tune } \theta, \lambda, \chi$$

- Pure gauge and 4 flavor staggered HMC
- Autotuning worked well, very convenient compared to manual tuning

Now adding Mass Preconditioning

[Hasenbusch (2001)]

- Staggered fermion action (4 flavors)

$D = D_{oe}$

$$\left|D^\dagger D + m^2\right| = \int d\phi e^{-\phi^\dagger\left[D^\dagger D + m^2\right]^{-1}\phi}$$

- Hasenbusch mass preconditioning

$$\left|D^\dagger D + m^2\right| = \left|\frac{D^\dagger D + m^2}{D^\dagger D + h_1^2}\right| \dots \left|\frac{D^\dagger D + h_{n-1}^2}{D^\dagger D + h_n^2}\right| \left|D^\dagger D + h_n^2\right| =$$

$$\int d\phi_0 e^{-\phi_0^\dagger \frac{D^\dagger D + h_1^2}{D^\dagger D + m^2}\phi_0} \dots \int d\phi_{n-1} e^{-\phi_{n-1}^\dagger \frac{D^\dagger D + h_n^2}{D^\dagger D + h_{n-1}^2}\phi_{n-1}} \int d\phi_n e^{-\phi_n^\dagger\left[D^\dagger D + h_n^2\right]^{-1}\phi_n}$$

- Tune h’s to optimize efficiency

Evaluating HMC efficiency

- MD integration time: $\boldsymbol{\tau}$
- After $\boldsymbol{N}$ HMC update trajectories (momentum refresh, accept/reject)
  - effective MD integration time (average acceptance probability: $\langle \boldsymbol{P_{acc}} \rangle$)

$$\boldsymbol{T_{eff} \equiv \tau\sqrt{\langle P_{acc}\rangle N}}$$

- Solve for $\boldsymbol{N(T_{eff})}$

$$\boldsymbol{N\left(T_{eff}\right) = \frac{T_{eff}^2}{\langle P_{acc}\rangle \tau^2}}$$

- Cost $\equiv \boldsymbol{N\left(T_{eff} = 1\right)}$ **×** (cost per update trajectory)

Cost function

- Cost per update trajectory
  - Ideally measure the time for each update operation (gauge update, force evaluations, …)
  - Fermion forces require solver
    - Model solver time as function of solver mass

- For simplicity currently only considering CG solver time
  - (relative cost per update) $\approx \sum_{m_{CG}} \frac{N_{CG}(m_{CG})}{N_{CG}(m)} \approx \sum_{m_{CG}} \frac{\frac{1}{m_{CG}}}{\frac{1}{m}} = \sum_{m_{CG}} \frac{m}{m_{CG}} \equiv C_S$

Loss function optimization

- Want to minimize Cost

$$\textbf{Cost} = \frac{c_S}{\langle P_{acc} \rangle \tau^2}$$

- Involves average in denominator, inconvenient
- Instead minimize Loss

$$\textbf{Loss} = -\frac{1}{Cost} = -\frac{\langle P_{acc} \rangle \tau^2}{c_S}$$

- Using methods from ML
  - Calculate gradient of loss (using autograd implementation), use gradient to update parameters
  - Using Adam optimizer, accumulates gradient over update trajectories
  - Allow different learning rates for h's vs. other parameters (set at 4x others)

Gradient of pseudofermion

- Pseudofermions depend on h's

$$\int d\phi_{n-1} e^{-\phi_{n-1}^{\dagger}\frac{D^{\dagger}D+h_n^2}{D^{\dagger}D+h_{n-1}^2}\phi_{n-1}} \quad \rightarrow \quad \phi_{n-1} = \mathrm{P_e}(\mathrm{D_0}+\mathrm{h_n})^{-1}(\mathrm{D_0}+\mathrm{h_{n-1}})\eta_{n-1}$$

$\mathrm{P_e}$=project even

$\mathrm{D_0}$=full staggered operator at start of trajectory

η_{n-1}= Gaussian, independent of parameters

- Gradient of fermion force includes gradient of pseudofermion

Fermion tests

- Plain staggered, no rooting, 4 continuum flavors
- 12^3 x 24 lattice
- β = 5.4
- m = 0.04
- CG iterations scale with $1/m_{CG}$ to within 10-15% accuracy
- 200 tuning trajectories
- 400 measurement trajectories
- Always tuning $\boldsymbol{\tau}$ and MD integrator parameters

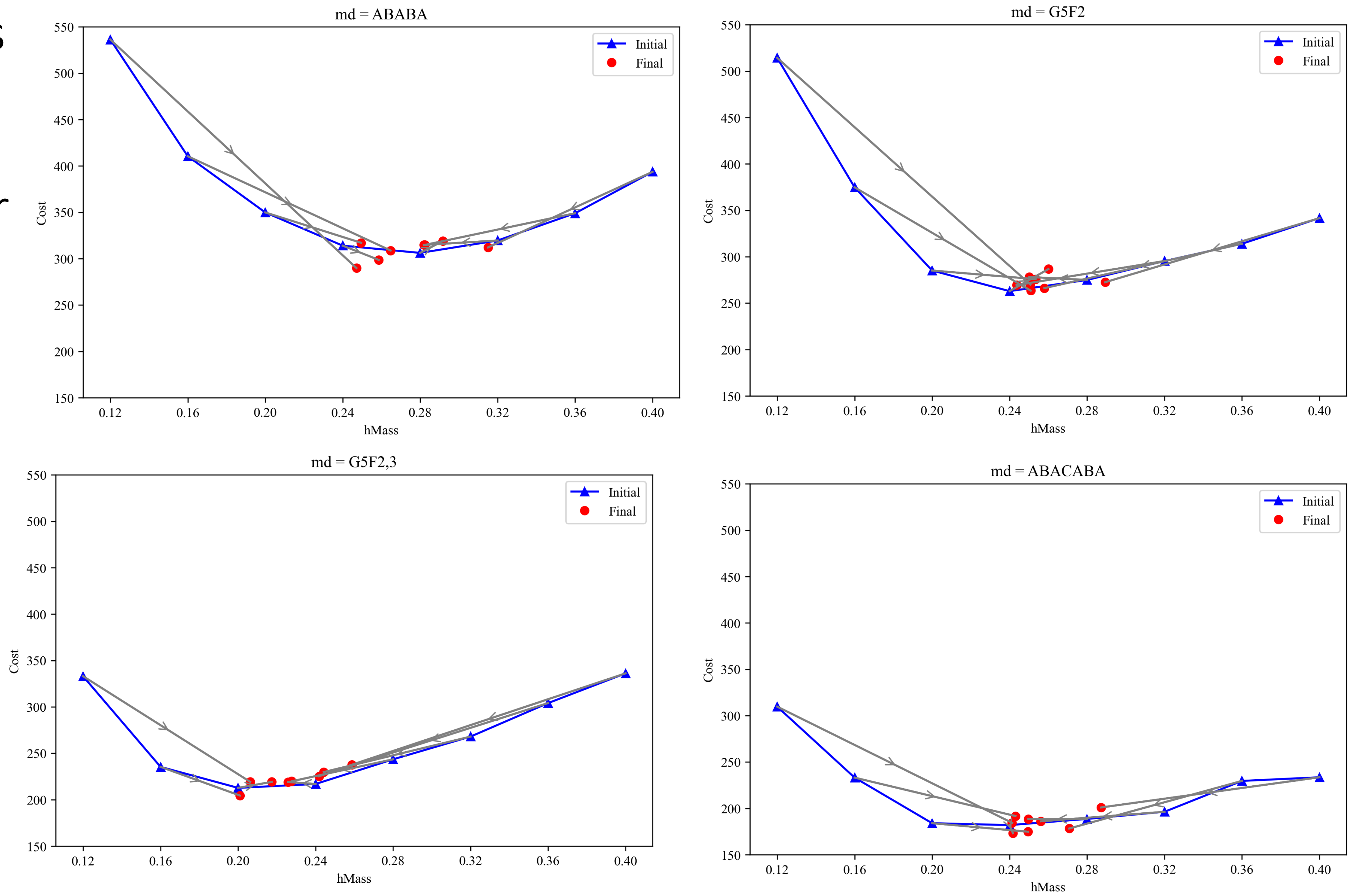
1 Hasenbusch mass
n_s=1 MD step

Changing integrator
ABABA:
- 2 gauge force
- 2 fermion force

G5F2:
- 5 gauge force
- 2 fermion force

G5F2,3:
- 5 gauge force
- 2 light fermion force
- 3 heavy fermion force

ABACABA:
- force gradient



Force-gradient implementation

- Using Hessian-free version (H. Yin, R. D. Mawhinney 2011)

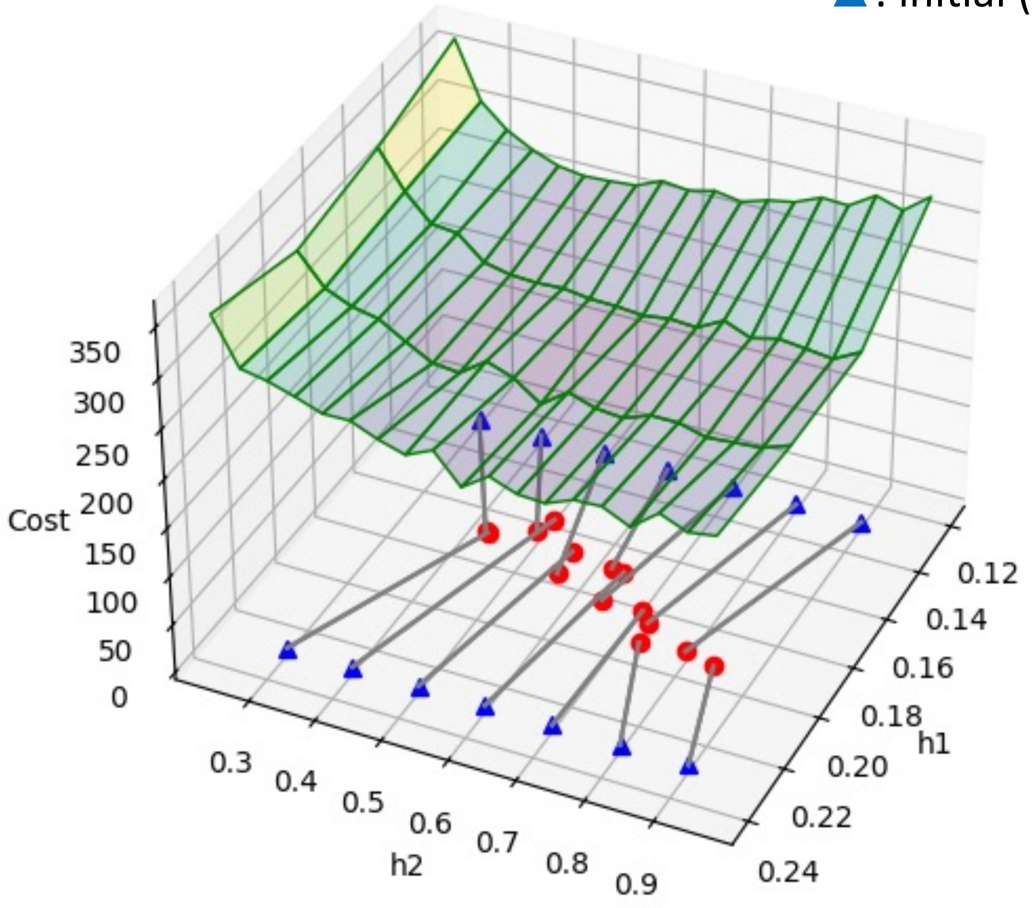
$$U' = e^{aF(U)}U$$
$$p' = p + bF(U')$$

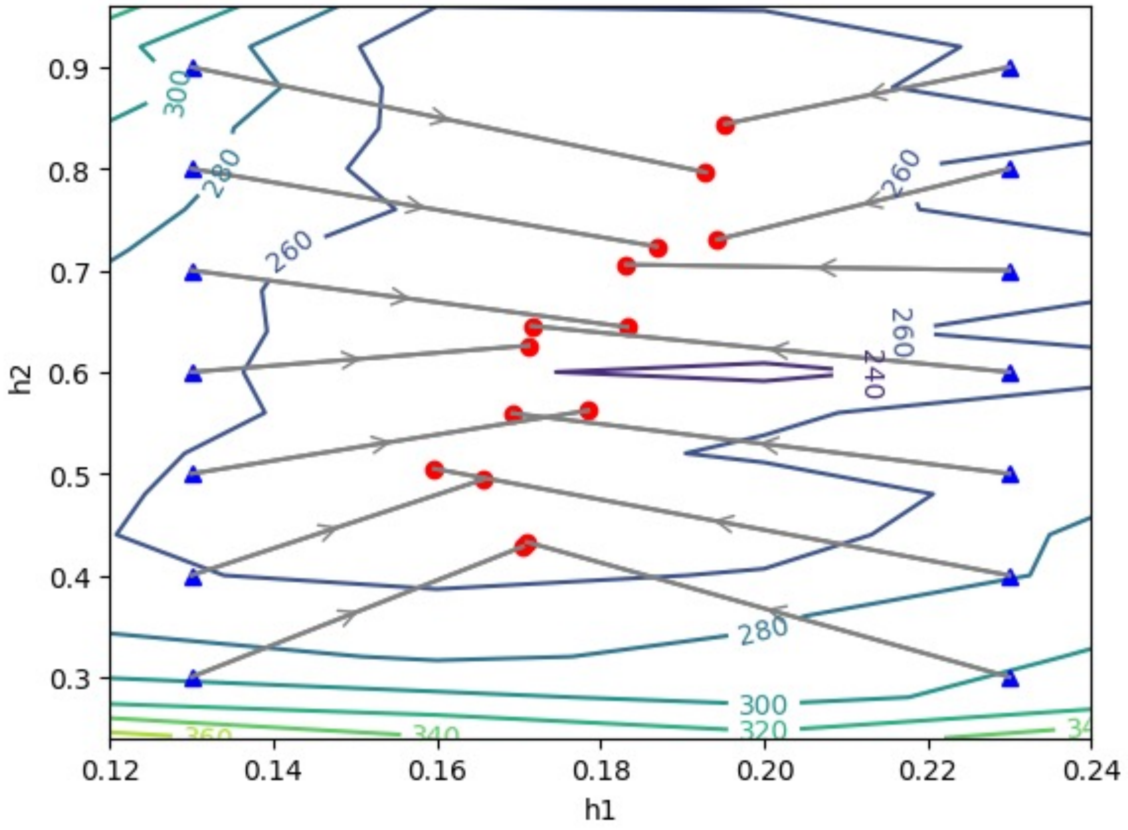
- Using combined force-gradient for gauge and fermion

$$U' = e^{a_G F_G(U) + a_F F_F(U)}U$$
$$p' = p + b_G F_G(U') + b_F F_F(U')$$

2 Hasenbusch masses, n_s=1 MD step, ABABA

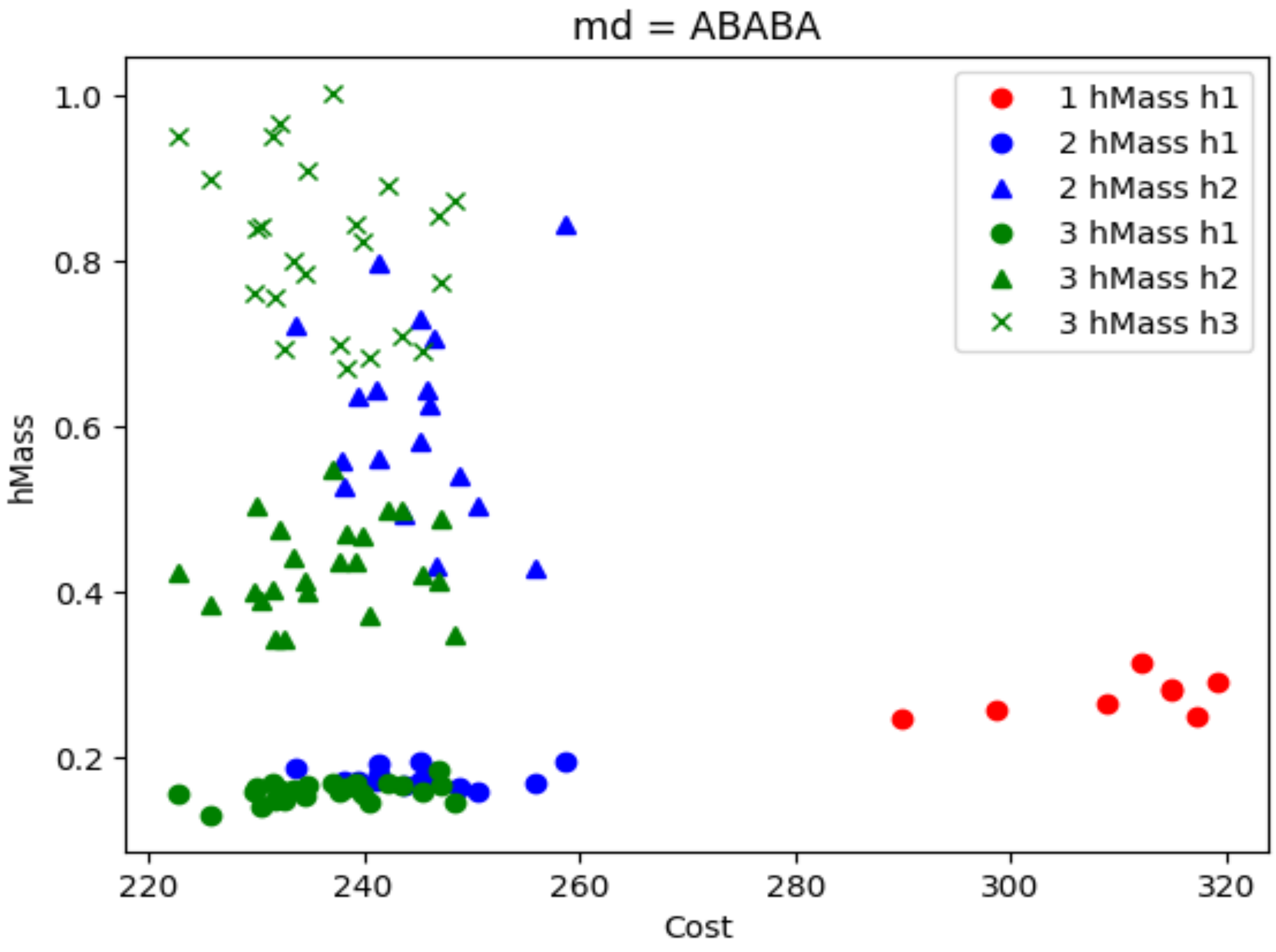
▲: initial (before tuning) ●: final (after tuning)





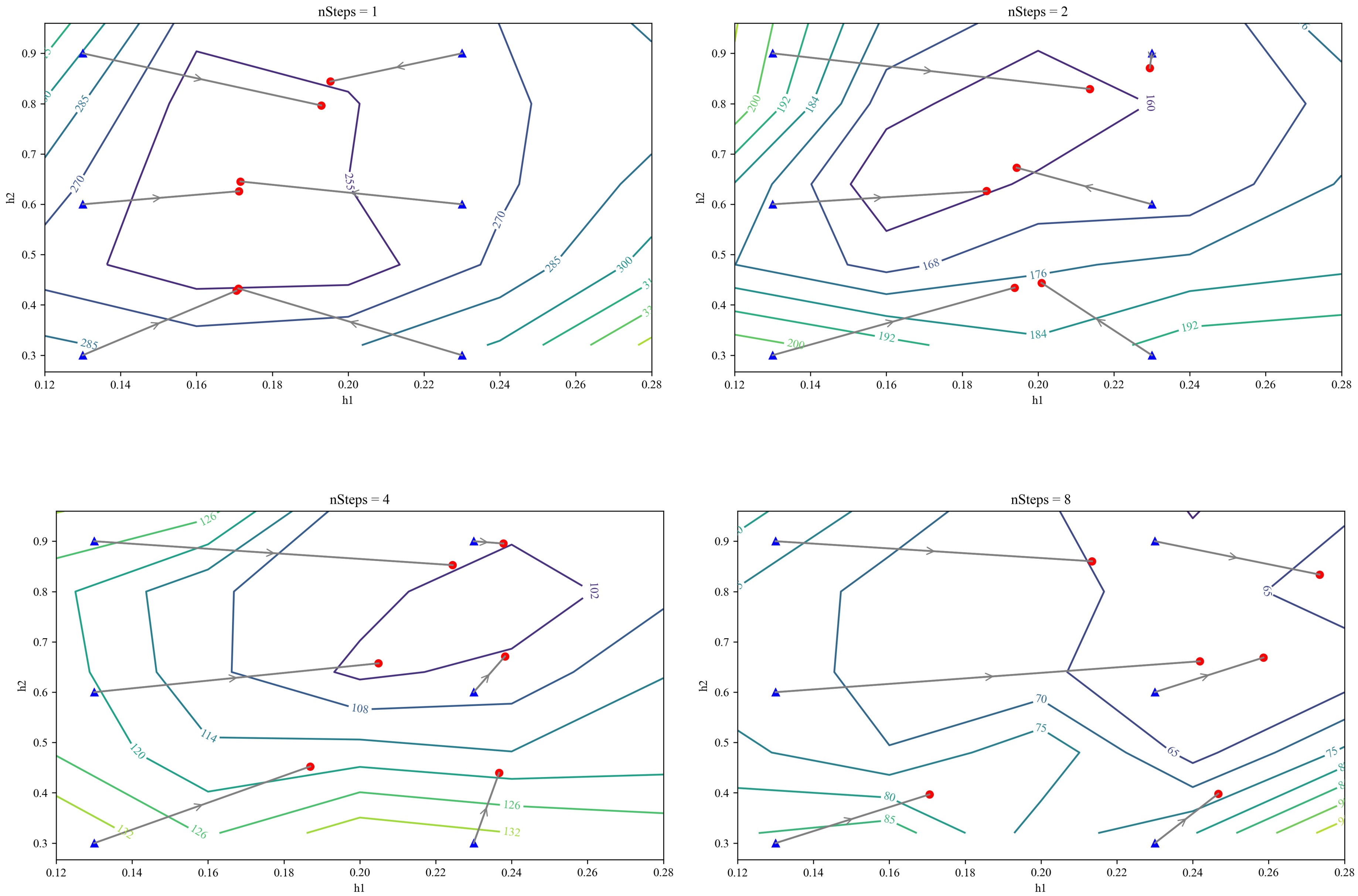
Tuned values for 1, 2 & 3 Hasenbusch Masses n_s=1 step ABABA

- 1 Hasenbusch mass
  - h_1 well determined
  - spread in Cost mainly statistical
- 2 Hasenbusch masses
  - Cost noticeably smaller
  - h_2 not well determined
- 3 Hasenbusch masses
  - Cost not much lower
  - h_1 similar for 2 H. masses
  - h_2 and h_3 split from 2 H. mass h_2
- Need larger, lighter ensemble to fully explore more H. masses



Dependence on number of MD steps

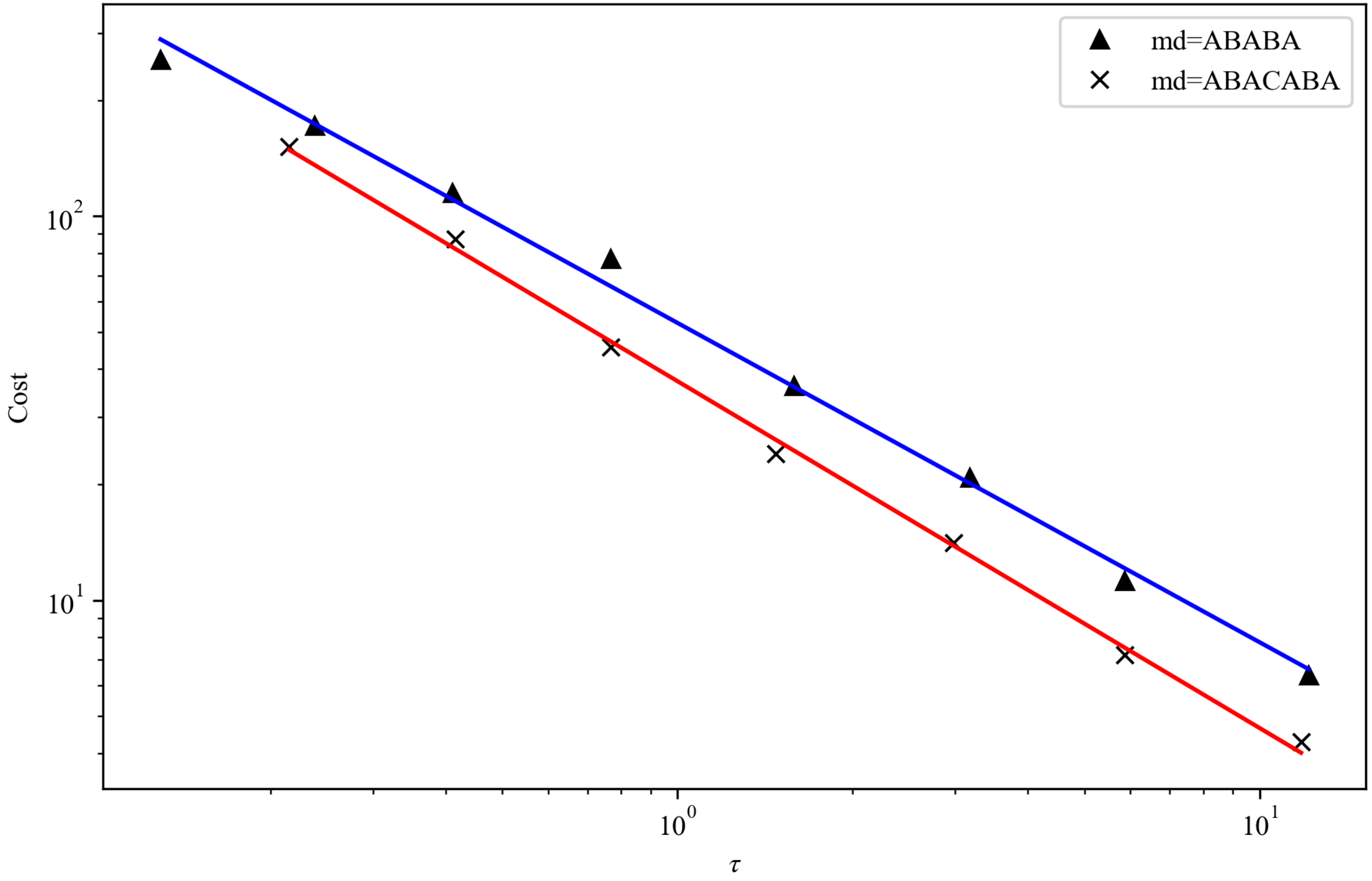
- Tuned values for 2 Hasenbusch masses
- ABABA
- n_s = 1, 2, 4 & 8 steps per trajectory
- Cost minimum moves to larger h_1 possibly larger h_2
- n_s = 8 likely influenced by large statistical fluctuations



Tuned Cost vs. trajectory length for increasing number of steps

$$\textbf{Cost} = \frac{C_S}{\langle P_{acc} \rangle \tau^2}$$

- At fixed $\boldsymbol{\varepsilon = \tau / n_s}$
  - Force evaluations $\boldsymbol{C_S \sim n_s = \tau/\varepsilon}$
  - Ideal scaling
    Cost ~ $\boldsymbol{\tau}^{-1}$
- ABABA
  Cost ~ $\boldsymbol{\tau}^{-0.83}$
- ABACABA
  Cost ~ $\boldsymbol{\tau}^{-0.90}$
- Want to limit trajectory length based on observable autocorrelation time
- Need to more accurately model effect of momentum refresh on autocorrelation



Summary

- Tuning HMC parameters using gradient information is working well so far
- Very convenient way to tune HMC, after initial investment in implementation
- Cost is very sensitive to lowest Hasenbusch mass, less so for higher ones
  - Should become more sensitive for larger volume, lighter quark mass
- Plan to implement other actions (improved gauge, smearing, RHMC) and test on production ensembles
  - RHMC: propagate gradient through rational function approximation
- Improve cost estimate
  - Model full HMC costs & observable autocorrelation time