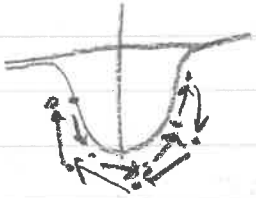
NUTS - No U-Turn Sampler

- (Brief) review of HMC
  - Physical particle simulation (potential + Kinetic energy)
  - Benefits
    - Reduce auto-correlation a.k.a. get good samples from the target distr. faster
    - Reject far fewer samples, since energy is conserved

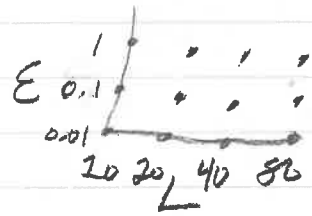
Good, right?

Not so fast!!

  - Requirements:
    - Differentiable log prob (for gradients)
    - Hyper-parameters: ε + L
      - Must be chosen
      - Bad values result in updates near our starting position!



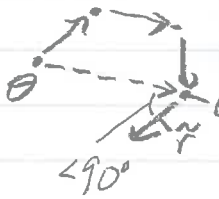
- So... How do we pick good values?
  - Naive/standard/fallback method: grid search



  - Offline, i.e., must re-run HMC for each pair and compare results. Computationally Expensive
  - What we want: online method, i.e., the parameters are tweaked as the algo. runs. No need to run multiple times - the algo figures it out for us!
  - Application: inference engines like BUGS, JAGS, STAN

- Heuristics for "good" values of ϵ, L

Sec. 3.2
- ϵ: as ϵ approaches 0, we approach the true Hamiltonian and reduce the error between initial energy + target energy to 0.
  - Recall: error between these two values affects acceptance ratio, so there is a correspondence between $\epsilon \to h_\epsilon$, h_ϵ is avg. acceptance rate for ϵ.
  - Pick a target acceptance rate, δ, and use the heuristic $\delta - h_\epsilon$ (to increase or decrease ϵ)
  - Use "dual averaging method" to deal w/ the two modes of MCMC:
    - Converging to target distr.
    - Already at target distr.
- L:
  - Idea: Want to get as far away from current sample, θ, as possible
  - If we start turning back, i.e., make a U-turn, don't take anymore steps!



$(\tilde{\theta} - \theta) \cdot \tilde{r} < 0$

Naive method to optimize L (bottom of p. 1597)
- Just sample r, then continue until a U-turn happens.
- Why not? Not time reversible, so not a valid MCMC alg.
- Maybe worth spending some time to unpack this. I don't quite understand this, since it seems similar enough to HMC.

NUTS - No U-Turn Sampler

So, we want to tune L, and we need to do it s.t. we still have a valid MCMC algo.

NUTS - no U-turn sampler:

- New variables:
  - u - "slice," I call it "error threshold"
  - B - set of all leapfrog states visited during the physics simulation
  - $C \subseteq B$, the set of valid moves from $\theta \to \theta'$
- Algorithm: (p. 1599)
  - $r \sim \mathcal{N}(0, I)$ (as in HMC)
  - $u \sim \text{Uniform}\left(0, \exp\left\{\mathcal{L}(\theta^t) - \frac{1}{2} r \cdot r\right\}\right)$
  - B: (Key: simulate backwards in time as well as forward)
    - for $j \leftarrow 0 \ldots \infty$
      - Sample direction, $v_j \sim \text{Uniform}(\{-1, 1\})$
      - Simulate for 2^j steps from the corresponding endpoint already in B (front or back) using $v_j \varepsilon$ as step size
    - (Note: conceptually a tree, fig. 1)): Add these to B.
    - Until a stop criteria is met
      - (1) one of the newly-added intermediate states either
        - (a) Had error above the threshold

$$\mathcal{L}(\theta) - \frac{1}{2} r \cdot r - \log(u) < -\Delta_{max} \quad \text{(p. 1601, eqn. 3)}$$

        - (b) Made a U-turn
      - (2) The two end-points make a U-turn, but no intermediate states make a U-turn
  - C:
    - Deterministic, given B
    - Prunes the possible new states to ensure good "next" states + a valid MCMC method
    - Conditions at the bottom of p. 1599

→

- Show fig. 2
- Sample $\theta' \sim T(\theta^t; r, \epsilon)$. T can be uniform
- Optimizations
  - or more complicated to "weight" states that are further away
    This is where the "tree" property comes in handy
- Experiments + results
  - How should we design the experiments
  - What metrics are we interested in?
    - Comparison w/ HMC w/ various hand-picked L.
    - Computational efficiency. How many accepted samples per grad. calc.?
    - Are we getting "good" samples in the first place?