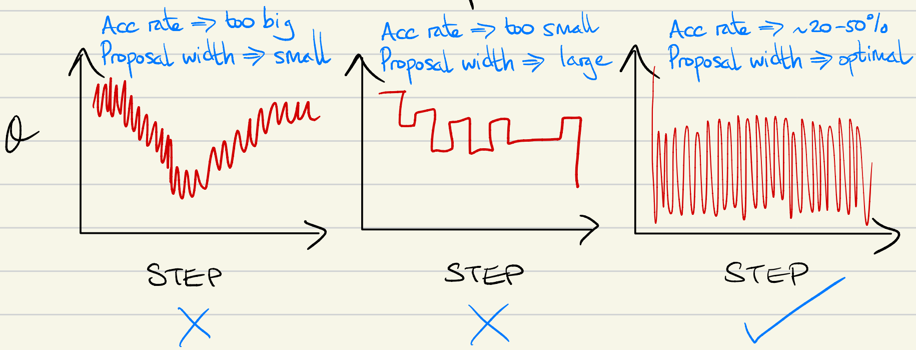
Lecture 11

02/22/2022

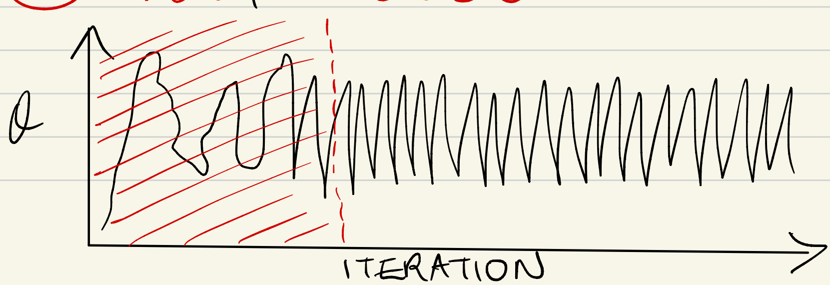
1. Check acceptance rate
   - ↳ ~20 - 50% is OPTIMAL.

2. Check traceplots



3. Check chain autocorrelation length
   - ↳ measures the number between independent samples in the chain.
   - ⇒ we want independent samples from the POSTERIOR for MC integration.

4. Dump the burn-in



Optimizing sampling

(i) ADAPTIVE METROPOLIS

* Use the chain to tune the width of the proposal distribution.

⇒ Estimate $N_p \times N_p$ covariance matrix, C
⇒ Factorize to take square root, $C = LL^T$.
⇒ $\theta_{i+1} = \theta_i + \alpha L U$

$\alpha = 2.38/d_{param}$

U = N_p-vector of draws from $N(0,1)$.

(ii) SINGLE COMPONENT ADAPTIVE METROPOLIS (SCAM)

* PCA on chain to identify important directions in parameter space!

⇒ $C = D \Lambda D^T$

eigen-matrix

$\text{diag}(\sigma_\Lambda^2)$ eigenvalues.

⇒ $\theta_{i+1} = \theta_i + 2.38 D_j u_j$

randomly chosen column of D.

$u_j \sim N(0, \sigma_\Lambda^j)$

(iii) DIFFERENTIAL EVOLUTION (DE)

$$\theta_{i+1} = \theta_i + \beta (x_{r1} - x_{r2})$$

usually 1

2 randomly chosen points from chain history.

- Practical MCMC

⇒ emcee
- v. popular
- good for small problems
- not great in high-D.

⇒ PyMC
- super fancy
- lots of automated optimizations
- overkill for many situations

⇒ PTMCMCSampler
- bare-bones
- manual control
- can use parallel tempering
- built for PTA GW searches

$\boxed{\text{Experience and trial/error is a great teacher. MCMC can be an art form.}}$