Introduction to Monte Carlo Markov Chains (Part 1)

Class 19

Grant Wilson

Fitting reminder: the goal

- Given some data, $\vec{y}$, and a model function, $f(\vec{x}, \vec{a})$, find the set of $\vec{a}$ that maximizes the probability of getting that data set given that model: $P(\vec{y}|\vec{a})$.

- For independent data points with normally distributed uncertainties with variance, σ_i^2, we have

$$P(\vec{y}|\vec{a}) = \prod_{i=0}^{N-1} e^{-\left(\frac{\left(y_i - f(x_i, \vec{a})\right)^2}{2\sigma_i^2}\right)}$$

How do we find the best $\vec{a}$?

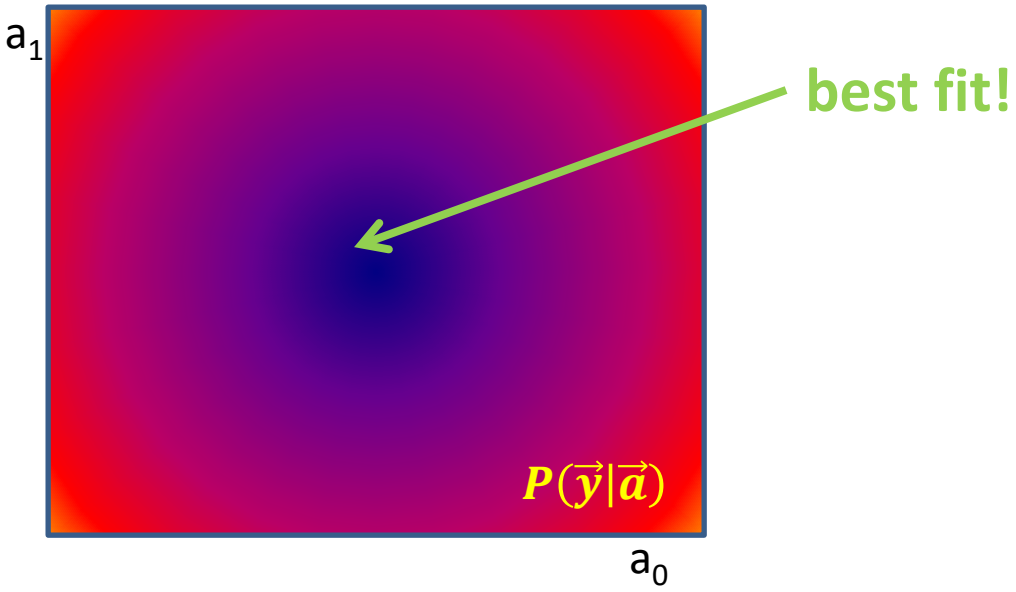
1. If our model is linear in the parameters then use SVD.

2. If our model is nonlinear in the parameters then use Levenberg-Marquardt

But why work so hard?

What about this?

- Take your favorite range of parameter space and grid it up.
- Calculate $P(\vec{y}|\vec{a})$ at every point in the grid.



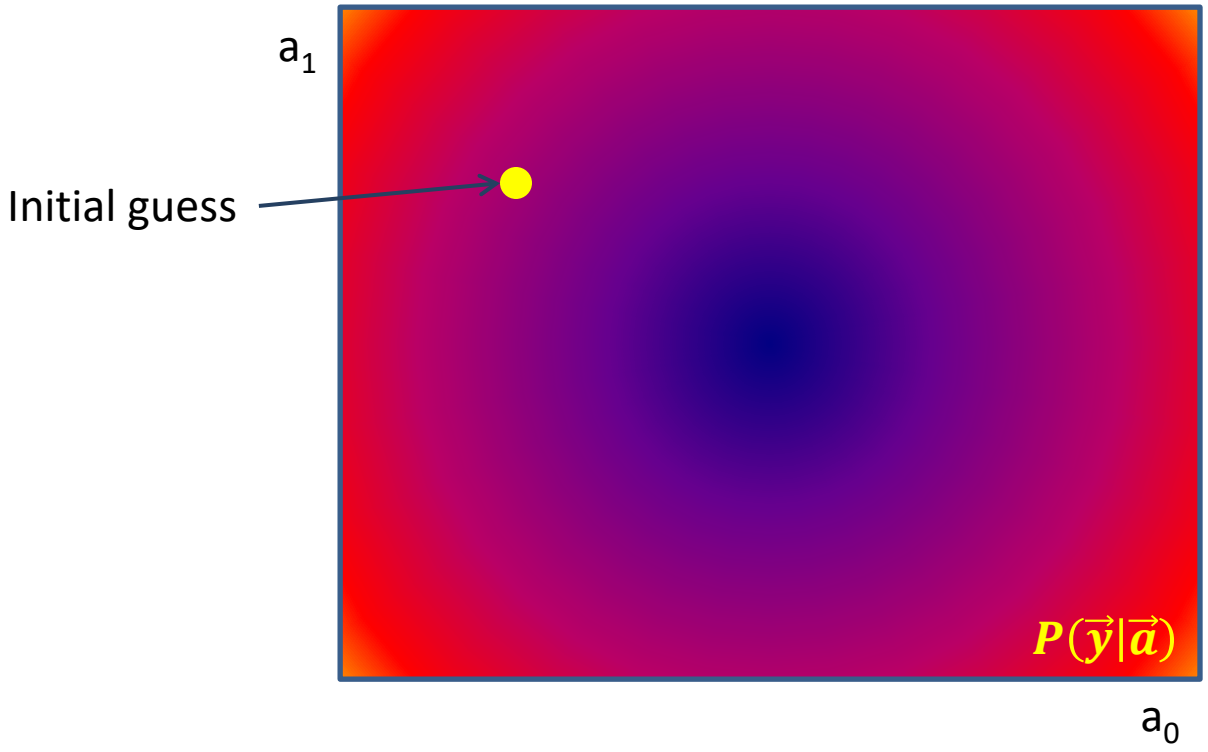
Problems with this approach …

- This is called the “grid search method”

- It is very expensive computationally. For M free parameters and *m* grid cells per parameter this scales as m^M.

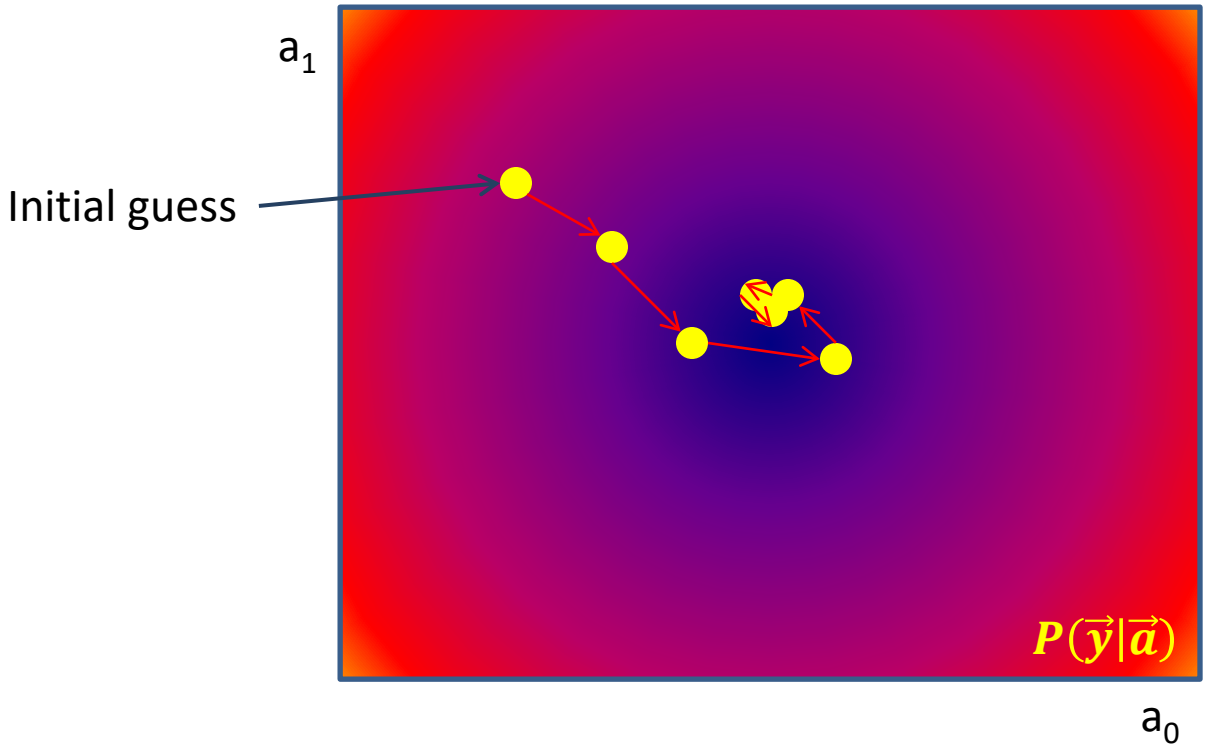
What is Levenberg Marquardt doing?

- Levenberg Marquardt is an approach to go downhill starting from a guess to the best fit point. That's fine as long as there is only one minimum.



What is Levenberg Marquardt doing?

- Levenberg Marquardt is an approach to go downhill starting from a guess to the best fit point. That's fine as long as there is only one minimum.



This finds the minimum but does not tell you anything about the shape of $P(\vec{y}|\vec{a})$

- Under this approach, how do we then learn about the posterior probability distribution?

- Under this approach, how do we then learn about the posterior probability distribution?

- How is this different from just sampling a gridded out parameter space?

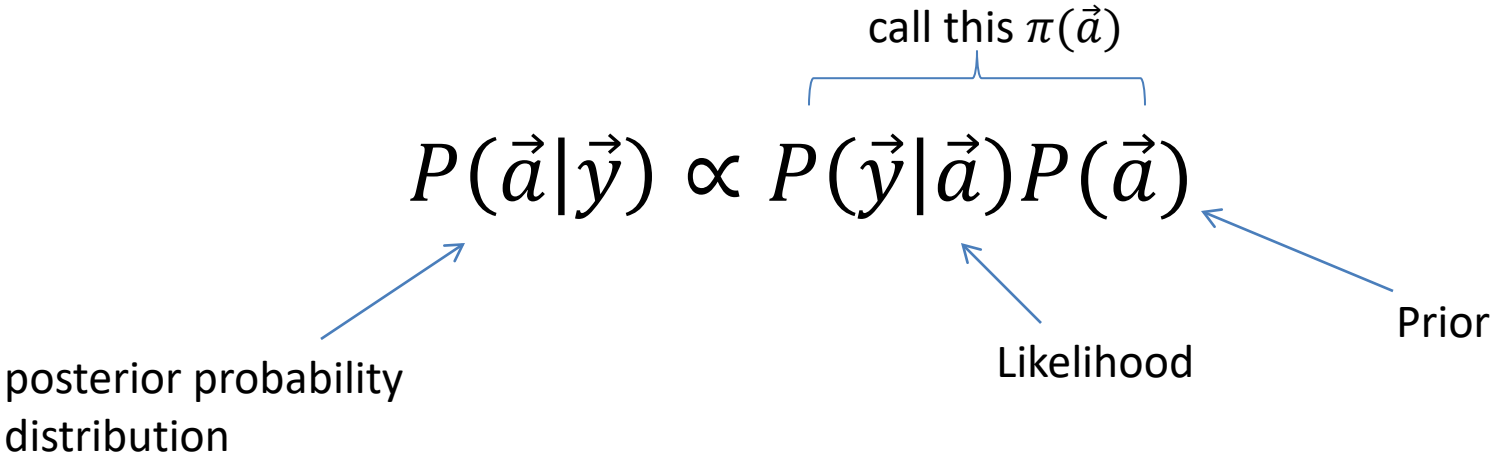
- so calculating $P(\vec{y}|\vec{a})$ at every point is too slow …

- and going straight to the minimum doesn't tell us about the shape of $P(\vec{y}|\vec{a})$ (at least not without doing a simulation …)

- Enter Monte Carlo Markov Chains, a technique for efficiently sampling the parameter space.

MCMC – the Idea

- Our goal is to learn about $P(\vec{a}|\vec{y})$.



We don't care about the amplitude of $P(\vec{a}|\vec{y})$, just its peak and its shape.

- MCMC is a technique for taking a “biased random walk” through the space of $\pi(\vec{a})$

- It is “random” in the sense that we will draw from probability distributions to guess which way to go.

- It is “biased” in the sense that we will allow $\pi(\vec{a})$ to influence how often we choose the uphill direction.

Ergodic Chains

- The set of accepted positions in our biased random walk through parameter space is called a "chain"

- An "ergodic chain" has the following two properties:

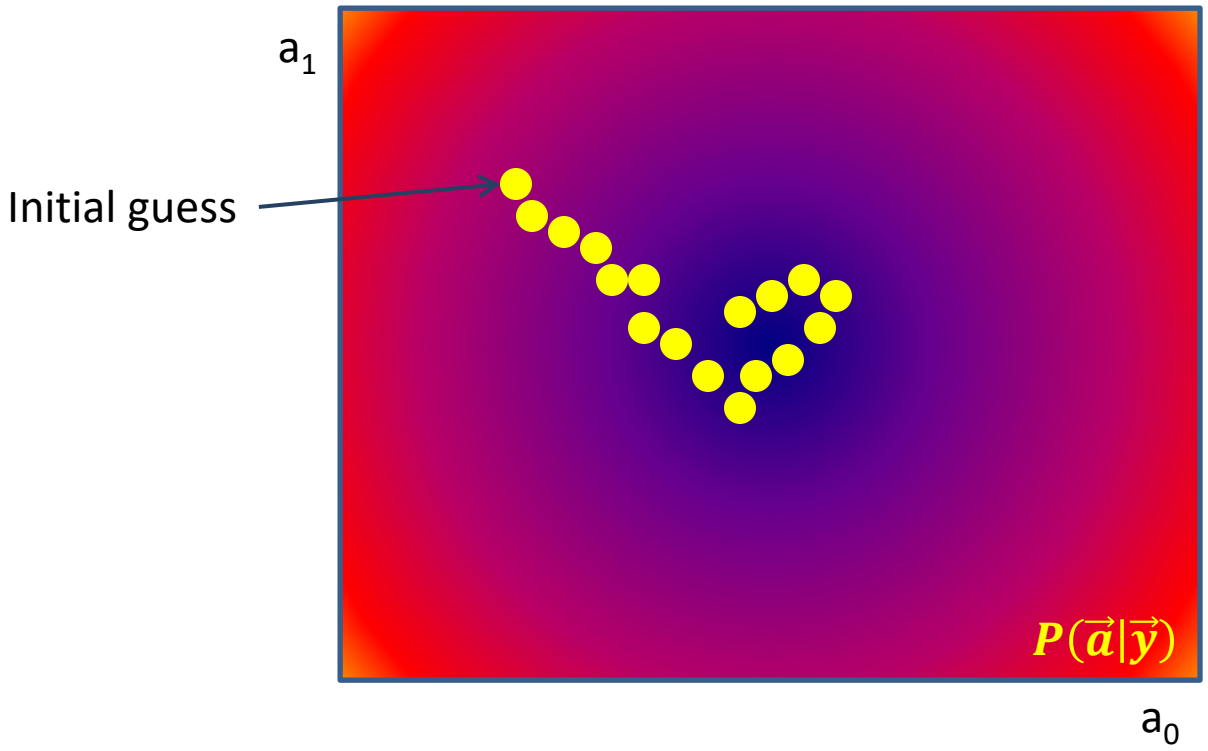
    1. the probability of visiting point $\overrightarrow{a_i}$ depends only on the previous point in the chain, $\overrightarrow{a_{i-1}}$. This will be governed by a "transition probability", $p(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})$

    2. the chain has the potential to visit every point in parameter space where $P(\vec{a}) \neq 0$

- Both properties are achieved by choosing our transition probability so that it satisfies the detailed balance equation:

$$p(\overrightarrow{a_{i-1}}|\overrightarrow{a_i})\pi(\overrightarrow{a_i}) = p(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})\pi(\overrightarrow{a_{i-1}})$$

- Again, we start out with a guess. This time we allow the chain to random walk through parameter space (biased by $\pi(\vec{a})$), and we keep track of where it goes.



Over time, the chain will fill in the space with a density of visits proportional to the posterior probability distribution.

- Here is a recipe following the Metropolis-Hastings Algorithm
  1. Choose a proposal distribution $q(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})$ (a good choice is often to start with a multivariate normal distribution centered on $\overrightarrow{a_{i-1}}$).

  2. Draw a new candidate position, $\overrightarrow{a_{ic}}$, from $q(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})$.

  3. Calculate an acceptance probability, $\alpha(\overrightarrow{a_{i-1}}|\overrightarrow{a_{ic}})$, according to

$$\alpha(\overrightarrow{a_{i-1}}|\overrightarrow{a_{ic}}) = \min\left(1, \frac{\pi(\overrightarrow{a_{ic}})}{\pi(\overrightarrow{a_{i-1}})} \frac{q(\overrightarrow{a_{i-1}}|\overrightarrow{a_{ic}})}{q(\overrightarrow{a_{ic}}|\overrightarrow{a_{i-1}})}\right)$$

  4. Draw a uniform deviate, u, between 0 and 1.
     1. if $u < \alpha(\overrightarrow{a_{i-1}}|\overrightarrow{a_{ic}})$ then accept the candidate step: $\overrightarrow{a_i} = \overrightarrow{a_{ic}}$
     2. if $u > \alpha(\overrightarrow{a_{i-1}}|\overrightarrow{a_{ic}})$, then reject the candidate step: $\overrightarrow{a_i} = \overrightarrow{a_{i-1}}$

  5. Go back to step #2 and take the next step

Doing this results in a transition probability of:

$$p(\overrightarrow{a_i}|\overrightarrow{a_{i-1}}) = q(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})\alpha(\overrightarrow{a_i}|\overrightarrow{a_{i-1}})$$

Exercise #1

- Of course we start with a simple linear fit.

- See the class19 exercises on the Moodle page for instructions.