Numerical Optimization

Until now, we have discussed evaluating likelihoods.

- We've seen that in many cases maximizing a likelihood can be challenging and require numerical procedures.

Given that we can compute the value of a likelihood, for a $\vec{\theta}$ and $\vec{y}$ we will now discuss numerical optimization techniques.

- Numerical methods typically won't find the exact MLE.
- We will need to set a tolerance level for the quality of our approximation.

Grid Search

Let our parameter vector $k\,\vec{\theta} \in \mathbb{R}^k$. We define a k-dimensional "grid" or hypercube of points, $\textcircled{H}$

- We can define a univariate grid, $\textcircled{H}^{(w)}$ for $\theta_1 \in \vec{\theta}.\{\theta_{i,1}, \theta_{i,2}, \ldots, \theta_{i,m_i}\}$
- $\textcircled{H} = \textcircled{H}^{(1)} x \textcircled{H}^{(2)} x \ldots x \textcircled{H}^{(k)}$
- Ex: $N(\mu, \sigma^2)$

$$\begin{array}{ccccc} -5 & . & \mu & . & 5 \\ . & & . & & . \\ . & . & . & . & . \\ \sigma^2 & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{array}$$

- Often such a grid search is simply equally spaced, but this is not required

Ex. AR(1)

Suppose $c = 0$ and $\sigma^2 = 1$. $\vec{\theta} = \phi \implies k = 1$

- The grid can be a set of values equally spaced between -0.99 and 0.99
- Given a data set $\vec{y}$, we can compute the exact or conditional likelihood for each ϕ_i in the grid and pick the one that yields the highest value of the log likelihood: ϕ^*.
- Refine the grid search around ϕ^* until our tolerance is reached.

If the likelihood is concave, then we can use a more efficient algorithim:

- Binary search
    1. Pick two adjacent points, θ_j and θ_{j+1}, in the middle of the grid and evaluate the likelihood

2. If $\mathcal{L}(\theta_{j+1}) < \mathcal{L}(\theta_j)$, set the lower bound of the grid to be θ_j and otherwise set the upper bound to be θ_{j+1}.
3. Return to step 1, until the lower and upper bounds of the grid are sperated by no more than one point.

$$\begin{array}{cccccccccc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & \uparrow & & & & & \uparrow \\ & & & & L & & & & & u \end{array}$$

- Golden Search: see Heer and Maussner

  Grid search is very ineffective when k is large because the number of grid points grows exponentially: doubling the number of points in each dimension results in 2^k as many points.

- This is called the curse of dimensionality.

<u>Newton-Raphson</u>

Define:

$$\vec{g}(\vec{\theta}) = \nabla \ell(\vec{\theta}) = \frac{\gamma \ell(\vec{\theta})}{\gamma \vec{\theta}}$$

$$H(\vec{\theta}) = \nabla^2 \ell(\vec{\theta}) = \nabla \vec{g}(\vec{\theta}) = \frac{\gamma^2 \ell(\vec{\theta})}{\gamma \vec{\theta} \vec{\theta}'}$$

Suppose $H(\vec{\theta})$ is positive definite:

$$\vec{x}' H(\vec{\theta}) \vec{x} > 0 \quad \vec{x} \in \mathbb{R}^k$$

We can approximate $\ell H(\vec{\theta})$ with a 2nd order Taylor expansion around $\vec{\theta}^{(0)}$:

$$\tilde{\ell}(\vec{\theta}) = \ell(\vec{\theta}^{(0)}) + \vec{g}(\vec{\theta}^{(0)})'(\vec{\theta} - \vec{\theta}^{(0)}) + \frac{1}{2}(\vec{\theta} - \vec{\theta}^{(0)})' H(\vec{\theta}^{(0)})(\vec{\theta} - \vec{\theta}^{(0)})$$

The Newton-Raphson method chooses $\vec{\theta}^{(1)}$ to maximize $\tilde{\ell}(\vec{\theta})$:

$$\left.\frac{\gamma \tilde{\ell}(\vec{\theta})}{\gamma \vec{\theta}}\right|_{\vec{\theta}=\vec{\theta}^{(1)}} = \vec{g}(\vec{\theta}^{(0)}) + H(\vec{\theta}^{(0)})(\vec{\theta}^{(1)} - \vec{\theta}^{(0)}) = 0$$

$$\implies \vec{\theta}^{(1)} - \vec{\theta}^{(0)} = -H(\vec{\theta}^{(0)})^{-1} \vec{g}(\vec{\theta}^{(0)})$$

$$\implies \vec{\theta}^{(1)} = \vec{\theta}^{(0)} - H(\vec{\theta}^{(0)})^{-1} \vec{g}(\vec{\theta}^{(0)})$$

Newton-Raphson begins with a guess $\vec{\theta}^{(0)}$ and iteratively computes:

$$\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - H(\vec{\theta}^{(i)})^{-1} \vec{g}(\vec{\theta}^{(i)})$$

until $||\vec{\theta}^{(i+1)} - \vec{\theta}^{(i)}|| < \tau$

where τ is some tolerance level.

- Newton-Raphson converges fast if the likelihood is concave and the initial guress is good enough.

- A modified version of NR computes:

  $$\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - sH(\vec{\theta}^{(i)})^{-1} \vec{g}(\vec{\theta}^{(i)})$$

  for various values of s and chooses $\vec{\theta}^{(i+1)}$ that yields the biggest log likelihood.

- Various modified NR methods have been proposed which substitute other positive definite matrices for H^{-1}.

- These are advantageous if H is not possible to compute or invert.
- Typically these are slower but more robust.

If analytical derivatives are not possible, numerical derivatives are an option.

- e.g. the ith element of $\vec{g}(\vec{\theta})$ can be approximated with

  $g_i(\vec{\theta}) = \frac{1}{\Delta}\left(\ell(\theta_1, \ldots, \theta_i + \Delta, \ldots, \theta_k) - \ell(\theta_1, \ldots, \theta_i, \ldots, \theta_k)\right)$ for some small Δ.
- The Hessian can be computed numerically from $\vec{g}$ in a similar fashion.