Maximum Likelihood Estimation and Likelihood Ratio Confidence Sets

Ken Judd, March 15, 2022

Likelihood function

Suppose we have data generated by a process governed by some unknown vector of parameters θ. Suppose we have a likelihood function for θ, $L(\theta)$, based on our data

Current practice

Find max:

Use numerical methods to maximize L(θ), BUT take little care to avoid finding only a local max instead of the global max.

Compute curvature:

Compute the Wald confidence interval, which depends only on the curvature of L(θ) at the max θ - a purely local construction.

Problem:

L(θ) is costly to compute.

Judd-Mueller-Reich approach

Global max:

Use a global maximization method to find the value of θ that maximizes L(θ).

LR Confidence Sets:

The likelihood ratio confidence set is a set defined by

$\{ \theta \mid L(\theta) \geq LR\}$

for appropriate LR. Its boundary is a level set of L(θ), that is,

$\{ \theta \mid L(\theta)=LR\}$

We compute confidence sets and/or their level sets

Our computational approach

Approximate L(θ)

Evaluate: compute L(θ) at a set of θ -- use massive parallelization

Construct a surrogate function: interpolate/regress to compute approximation

Maximize the surrogate

First use comparison methods -- use massive parallelization, multiple initial guesses

Then use DFO methods -- use massive parallelization, multiple initial guesses

Last use Newton-style methods

Confidence Sets

A confidence set is the level set for L(θ) = LR, for appropriate LR values based on max L.

Approximate the boundary by finding many points on the boundary via multiple 1-D searches

Our goal

Create a software framework that would

1: take an economists likelihood function (coded up in Fortran, C, C++, Python, Matlab, whatever we can use)

2: construct the surrogate approximation

3: find its max

4: find the confidence sets

Procedure

Step 1: Explore likelihood function
Evaluate likelihood function at a variety of points in order to get an idea of where it has nontrivial values.
PURPOSES: Get information about likelihood function, AND debug the code for the likelihood function.

Step 2: Apply a DFO optimization method
Apply a DFO method (e.g., Nelder-Mead, finsearch, DIRECT) to the likelihood function BUT with a loose stopping criterion.
Use parallelism by starting it at many different points. The starting points can be points examined in step 1 which have nontrivial values.
PURPOSES: Get a rough estimate of the maximum likelihood value AND increase our database of likelihood function evaluations AND debug the code.

Step 3: Choose cutoff values for generating a database of likelihood values
Step 2 gives us a rough idea of the maximum likelihood. Using that as an estimate of the maximum likelihood, choose likelihood values that correspond to 99% confidence levels (use 99% because it is the lowest likelihood level we are likely to examine and defines the largest set). Also, this should be the lowest 99% value across the different degrees of freedom we are likely to examine.

Step 4: Choose a domain for approximation
One natural choice is a hypercube that encloses the nontrivial values of the likelihood function evaluated in steps 1 and 2, and also satisfy the cutoffs chosen in step 3.

Step 5:
Use an adaptive approximation method (such as a sparse grid method) to construct a surrogate function.

Step 6:
Use surrogate to construct boundaries of confidence sets