Subject: TNMIN limits
Posted by biophys on Mon, 03 Dec 2007 02:56:47 GMT
View Forum Message <> Reply to Message

Hi, Folks

I've been doing some maximum likelihood fitting with TNMIN. It's kind of working fine except that I have to manually adjust the scaling of my parameters to get the fitting going. Without proper scaling, it seems that it will evaluate my function(s) outside of the parainfo limits where it might be much more computational intensive to evaluate or even undefined so that it doesn't converge or takes ridiculous amount of time to converge. I am having a hard time to understand the implementation details of TNMIN. Why does TNMIN insist evaluate my function beyond my set limits? Is there a way to avoid this? If the only way to avoid this is scaling, how to do automatic/adaptive scaling to get it run in a for loop of a series of TNMIN fitting. I'd appreciate any comments/hints!

Thanks!
BP

Subject: Re: TNMIN limits
Posted by Brian Larsen on Tue, 04 Dec 2007 19:55:34 GMT
View Forum Message <> Reply to Message

> Hi Brian;
> Would you please compare GA(genetic algorithm) with amoeba or SA
> method? Which on is faster? Which one is reliable? Is in the amoeba
> method any divergence problems exist like in GA or SA?
> Cheers
> Dave

> Hi Brian;
> Would you please compare GA(genetic algorithm) with amoeba or SA
> method? Which on is faster? Which one is reliable? Is in the amoeba
> method any divergence problems exist like in GA or SA?
> Cheers
> Dave

Dave, this is no small feat... What I now comes from Numerical recipes
http://www.nr.com/oldverswitcher.html
and from some experiences both research based and coursework...

In general the more information the algorithm requires the faster it is. After all you could always get the right answer if you tested

every possible option (I have done this too). Every algorithm is trying to get to the right answer in the least number of steps. Think of this practically, if you want to get to the ocean you walk downhill. You will certainly get there but certainly not the shortest way (fewest steps). If you look at a map you can then choose the shortest path. These are all that way also.

Looking at derivatives is another piece of the map making you need fewer steps. But with that comes the requirement that the function is differentiable. For Amoeba you don't assume that so you need more steps than a faster/smarter algorithm like Conjugate gradient. I always start my search for the minimum is several places regardless of the method to assure that I get the same answer. If not then which is the right answer? Also it is often good to try more than one method but who has the time? Well if you find the new minimum after the paper is published then you sure feel dumb.

I like to use amoeba as I understand it well and trust it. I am willing to take the efficiency hit for not having to think as hard or code as long. But if you are putting it inside a 10^6 for loop than you had better find a better algorithm.

GA algorithms are really really slow but work great.

This is a bit from an old course note that I happened to pull up. (NR = Numerical Recipes in C)

Function minimization in N dimensions

(1) Downhill simplex, a.k.a. amoeba (NR §10.4)
 * O(N^2) storage
 * robust, simple
 * Lecture/handout already prepared
(2) Conjugate gradient (NR §10.6)
 * Uses Brent's method (NR §10.3) in 1-D
 * O(N) storage ==> large problems
 * fast/efficient
 * requires derivatives (==> smoothness assumed)
(3) Simulated annealing (NR §10.9)
 * most robust and simplest of algorithms!
 * often finds global minima, even of rough functions
 * O(N) storage ==> large problems
 * slow as molasses in January
 * independent variables need not be continuous!

Cheers,

Brian

---------- ----------
Brian Larsen
Boston University
Center for Space Physics