Numerical Methods in Economics and Finance
Lecture V: Function Solving, Optimization, Numerical Integration and Differentiation

Xi Wang
School of Economics
Peking University

May 15, 2019

SETUP

- Investigating methods for optimizing a function with respect to a finite number of variables, say

$$\max_{x \in X} f(x) \tag{1}$$

Think about

$$\max_{k' \in R} \phi(k') = u(f(k) - k') + \beta v(k') \tag{2}$$

And we may have various kinds of capital, which we can model this value funtion as a multi-dimensional function.

  - Maximization of profit and utility functions
  - Maximize social surplus
  - Maximize Likelihood Function

- Optimization problem is pretty close to function solving problem
  - Derivative free
  - Newton-Type methods
  - Foundation (Wierstrass Theorem): if f is countinuous and X is nonempty, compact(close and bounded in real), then f attains max and min on X. Local max(min). If f is concave, X is convex , the local optimum is global.

DERIVATIVE-FREE

- Bracket iteratively.... , repeat it here.
- Motivation problem: want to find an optimal x^* to maximize concave function f on inverval $[a, b]$.
- Pick two points $x_1 < x_2$ in $[a, b]$
- Replace your original bracket by $[a, x_2]$ if $f(x_1) > f(x_2)$
- Replace your original bracket by $[x_1, b]$ if $f(x_1) \leq f(x_2)$
- Repeat until you got interval small enough.
- How to pick up x_i?
  - Set $x_i = a + \alpha_i(b - a)$
  - And

$$\alpha_1 = \frac{3 - \sqrt{5}}{2} \text{ and } \alpha_2 = \frac{\sqrt{5} - 1}{2} \tag{3}$$

WHY GOLDEN?

- To be Added

DF: NELDER-MEAD ALGORITHM

- Evaluate function on $N+1$ points, and shrinkage the distance...; And these $N+1$ points forms a space so called Simplex in N-dimensional Control variable space.

  Alogrithm
  - Given $N+1$ points, determines the worst point
  - I will replace the worst by a better one
  - Reflect
  - Fails then contract
  - If Fails again then shrink
  - Iterate until get the simplex small enough

NELDER-MEAD ALGORITHM

- Order your N+1 Points as, if you did not break this iteration

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \cdots \leq f(\mathbf{x}_{n+1}) \tag{4}$$

- Compute the centroid of all points except x_1
- Reflect

$$\mathbf{x}_r = \mathbf{x}_o + \alpha(\mathbf{x}_o - \mathbf{x}_1), \ \alpha > 0 \tag{5}$$

If $f(x_2) < f(x_r) < f(x_{n+1})$, go to step 1
If $f(x_{n+1}) \leq f(x_r)$, I will expand
If $f(x_r) \leq f(x_2)$, I will contract

- Expand: compute

$$\mathbf{x}_e = \mathbf{x}_o + \gamma(\mathbf{x}_r - \mathbf{x}_o) \text{ with } \gamma > 1 \tag{6}$$

If $f(x_e) > f(x_r)$, then replace x_1 by x_e, go to step 1
If $f(x_e) < f(x_r)$, then replace x_1 by x_r, go to step 1

- Contract:

$$\mathbf{x}_c = \mathbf{x}_o + \rho(\mathbf{x}_1 - \mathbf{x}_o) \text{ with } 0 < \rho \leq 0.5 \tag{7}$$

If $f(x_c) > f(x_2)$, then replace x_1 by x_c, go to step 1
If $f(x_c) < f(x_2)$, then Shrink

- Shrink: Replace all points except x_{n+1} by

$$\mathbf{x}_i = \mathbf{x}_{n+1} + \sigma(\mathbf{x}_i - \mathbf{x}_{n+1}) \tag{8}$$

COMPASS SEARCH

- Usually people use Compass Search for min, which is dual to max.
- Begin with a point say x_0, and step size in each direction $e_i = [0, 0 \ldots, h_i, 0 \ldots, 0]$
- Evaluate
- Move/Shrink
- Iterate until steps are small enough

Newton Method

- For this part, we investigate those Methods explore derivatives
- Newton-Raphson method, which use quadratic approximations to the objectivem and maximize the approximation.
  - Approximation:

$$f(x) \approx f\left(x^{(k)}\right) + f'\left(x^{(k)}\right)\left(x - x^{(k)}\right) + \frac{1}{2}\left(x - x^{(k)}\right)^{\top} f''\left(x^{(k)}\right)\left(x - x^{(k)}\right) \tag{9}$$

  - Optimize (F.O.C)

$$f'\left(x^{(k)}\right) + f''\left(x^{(k)}\right)\left(x - x^{(k)}\right) = 0 \tag{10}$$

  - Iterate

$$x^{(k+1)} \leftarrow x^{(k)} - \left[f''\left(x^{(k)}\right)\right]^{-1} f'\left(x^{(k)}\right) \tag{11}$$

- In theory, this method converges if $f \in C^2$, and x_0 is close to the true optimum.
- Second and First order Derivatives are all invovled, and the Hessian should be well-conditioned at the optimum.
- No guarantee that the objective function increases in the direction

QUASI-NEWTON METHODS

- This one employs a similar strategy to the Newton-Raphson method
  - In the equation (1), $d = \left[f''\left(x^{(k)}\right)\right]^{-1} f'\left(x^{(k)}\right)$ is called search direction
  - Now I will use

$$d^{(k)} = -A^{(k)} f'\left(x^{(k)}\right) \tag{12}$$

  - Where $A^{(k)}$ is an approximation to the inverse Hessian of f at x
  - However, you do not need to take a full search step, one could use a linear searching, namely, to have a s to maximize $f(x^k + sd^k)$, hence I will take

$$x^{(k+1)} = x^{(k)} + s^{(k)} d^{(k)} \tag{13}$$

GRADIENT DECENT

- What is A^k?, The simplest quasi-Newton method is $A = -I$, Hence optimum is updated as

$$x^{(k+1)} = x^{(k)} + s^{(k)} f'(x_k) \tag{14}$$

  - Why?
  - Crazily but Useful Modification
  - Generally not good for complex model.

Gradient Decent

- To be added
  -

Curvature Information

- Can we explore the curvature of f? like Newton Method?

$$d^{(k)} \approx f''^{-1}\left(x^{(k)}\right)\left(-f'\left(x^{(k)}\right)\right) \tag{15}$$

- And surely inverse Hessian should be symmetric and negative-definite.
  - Davidson-Fletcher-Powell (DFP): I update my A as

$$A \leftarrow A + \frac{dd^T}{d^T u} - \frac{Auu^T A}{u^T B u} \tag{16}$$

  where $d = x^{k+1} - x^k$, and $u = f'\left(x^{(k+1)}\right) - f'\left(x^{(k)}\right)$
  - Broyden-Fletcher-Goldfarb-Shano (BFGS): Again A as:

$$A \leftarrow A + \frac{1}{d^T u}\left(wd^T + dw^T - \frac{w^T u}{d^T u} dd^T\right) \tag{17}$$

  where $w = d - Au$, See `http://sims.princeton.edu/yftp/optimize/`

MORE ON LINEAR SEARCH

- Newton step recommend a search direction, but not on the strike.
  - One can use brute force to search best λ
  - Or Golden search to get the best λ
  - Or Armijo search
    Looking for the smallest j s.t

$$\frac{f(x+\rho^j d)-f(x)}{\rho^j} \geq \mu f'(x)^T d,\ \rho, d \in (0, .5) \tag{18}$$

    backtrack from a step size of 1 until the slope on the left hand side is a given fraction of the slope on the right hand side.
  - Or Goldstein search:
    Find any s s.t.

$$\mu_0 f'(x)^T d \leq \frac{f(x+sd)-f(x)}{s} \leq \mu_1 f'(x)^T d,\ 0 < \mu_0 \leq 0.5 \leq \mu_1 < 1 \tag{19}$$

CONSTRAINED OPTIMIZATION

- Our Objective is

$$\max_{a \leq x \leq b} f(x) \tag{20}$$

- You see, there are constraints, meaning that it is possible to have corner solution.
  - The optimal $x^* \in [a, b]$
  - If $x^* > a$, then $f'(x_i) \geq 0$
  - If $x^* < b$, then $f'(x_i) \leq 0$

Solving Functions

- For us the most common equations to be solve are

$$f(x) = 0, \quad f : \mathbb{R}^n \to \mathbb{R}^n \tag{21}$$
$$g(x) = x, \quad g : \mathbb{R}^n \to \mathbb{R}^n \tag{22}$$

- They are equivalent
  - The First case, for any f, I define $g = f - x$
  - The Second case, for any g, I define $g = f + x$
- Example?

$$u'(C_t) = \beta U(C_{t+1})[f'(f(K_t) + (1-\delta)K_t - C_t) + 1 - \delta] \tag{23}$$

- Similar to opitmizing? If I have a vector of function to solve then

$$\arg\min_x \frac{1}{2} \sum_i f_i(x)^2 \tag{24}$$

The min will be equal to zero only at the root.

Methods

- Bisection Method
- Function Iteration
- Newton's Method Again, but they are different
- Quasi-Newton Methods

Setup

- Motivation, Differentiation is common
- But why integration?
-
- Two approximation from staring definition

$$f'(x) = \lim_{u \to 0} \frac{f(x+u) - f(x)}{u} \tag{25}$$

$$\int_a^b f(x)dx = \lim_{\pi \to 0} \sum_i f(x_i)(x_{i+1} - x_i) \tag{26}$$

- The last one have several variants.

Differentiation

- The basic definition can be numerically implemented or $\frac{f(x+h)-f(x-h)}{2h}$

$$f(x+h) = f(x) + f'(x)h + f''(x)\frac{h^2}{2} + O\left(h^3\right) \tag{27}$$

$$f(x-h) = f(x) - f'(x)h + f''(x)\frac{h^2}{2} + O\left(h^3\right) \tag{28}$$

- How about $f''(x)$? $\frac{f(x+h)+f(x-h)-2f(x)}{h^2}$, from the same Equations.
- Or Say to approximate $f''(x)$ by $f(x), f(x+h), f(x-h)$
- Set $f''(x) = af(x+h) + bf(x) + cf(x-h)$, Taylor expansion every term and match coefficient.
- $a = h^{-2}, b = -2h^{-2}, c = h^{-2}$
- More generally, you do not need to approximate $f''(x)$ by $f(x), f(x+h), f(x-h)$ Right?

Differentiation II

- More generally, you can approximate $f'(x)$ by $f(x), f(x+h), f(x+\lambda h)$
- Same logic, you can solve the a, b, c like

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \frac{1}{h\lambda(1-\lambda)} \begin{bmatrix} \lambda^2 - 1 \\ -\lambda^2 \\ 1 \end{bmatrix} \tag{29}$$

- what if $\lambda = -1$?
- What if $\lambda = 2$?
- Similarly, you can approximate $f''(x)$ by $f(x), f(x+h), f(x+\lambda h), f(x+\eta h)$

$$f''(x) = af(x) + bf(x+h) + cf(x+\lambda h) + df(x+\eta h) \tag{30}$$

Differentiation III

- Again, Taylor Expansion and substitute

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \frac{2}{h^2} \begin{bmatrix} \frac{1+\lambda+\eta}{\lambda\eta} \\ \frac{\lambda^2-\eta^2}{(\lambda-1)(\eta-1)(\eta-\lambda)} \\ \frac{\lambda^2-\eta^2}{\lambda(\lambda-1)(\eta-1)(\eta-\lambda)} \\ \frac{1-\lambda^2}{\eta(\lambda-1)(\eta-1)(\eta-\lambda)} \end{bmatrix} \tag{31}$$

- How about f_{ij}?

Partial Differentiation

- Why do we need it?

$$h_{ij} := \frac{\partial^2 f(\bar{x})}{\partial x_i \partial x_j},\ i,j = 1,\ldots,N \tag{32}$$

- First the diagnoal Elements are fine

$$h_{ii} \simeq \frac{f\left(\bar{x} + \mathbf{e}_i h_i\right) + f\left(\bar{x} - \mathbf{e}_i h_i\right) - 2f(\bar{x})}{h_i^2} \tag{33}$$

- Off-Diagnoal

$$\begin{aligned}
f\left(\bar{x} + h_i + h_j\right) \approx & f(\bar{x}) + f_i(\bar{x})h_i + f_j(\bar{x})h_j + (1/2)h_{ii}h_i^2 \\
& + (1/2)h_{jj}h_j^2 + h_{ij}h_i h_j \\
f\left(\bar{x} + h_i - h_j\right) \approx & f(\bar{x}) + f_i(\bar{x})h_i - f_j(\bar{x})h_j + (1/2)h_{ii}h_i^2 \\
& + (1/2)h_{jj}h_j^2 - h_{ij}h_i h_j \\
f\left(\bar{x} - h_i + h_j\right) \approx & f(\bar{x}) - f_i(\bar{x})h_i + f_j(\bar{x})h_j + (1/2)h_{ii}h_i^2 \\
& + (1/2)h_{jj}h_j^2 - h_{ij}h_i h_j \\
f\left(\bar{x} - h_i - h_j\right) \approx & f(\bar{x}) - f_i(\bar{x})h_i - f_j(\bar{x})h_j + (1/2)h_{ii}h_i^2 \\
& + (1/2)h_{jj}h_j^2 + h_{ij}h_i h_j
\end{aligned}$$

Partial Differentiation General

- General

$$h_{ij} \simeq \frac{1}{4h_i h_j}\left[f\left(\overline{x}+\mathbf{e}_i h_i+\mathbf{e}_j h_j\right)-f\left(\overline{x}-\mathbf{e}_i h_i+\mathbf{e}_j h_j\right)\right.$$
$$\left.-f\left(\overline{x}+\mathbf{e}_i h_i-\mathbf{e}_j h_j\right)+f\left(\overline{x}-\mathbf{e}_i h_i-\mathbf{e}_j h_j\right)\right]$$

- You can set $i = j$ too.

Differentiation

- Same but we need more points
- How to choose h? As small as possible. But remember the finite precision of computer arithmetic.

Quadrature

- General Setup:

$$\int_D f(x)dx \tag{34}$$

  where $f : \mathbb{R}^n \to \mathbb{R}$
- General Solution: Quadrature (sometimes called cubature) methods approximate this integral via

$$\sum_{i=1}^{m} \omega_i f(x_i) \approx \int_D f(x)dx \tag{35}$$

- Generally different methods diff in
  - points $\{x_i\}_{i=1}^N$
  - weights $\{\omega_i\}_{i=1}^N$

Quadrature Methods

- Generally u have several choices
  - Newton-Coates: integrate piecewise (p-w) polynomial approximations.
  - Gaussian: integrate polynomial interpolants
  - Adaptive: add points when integration seems difficult
  - Monte Carlo (MC) and psuedo-MC: random points
  - Creative rule

Newton-Cotes Methods

- Newton-Coates rules
  - Use approximations of the function: piece-wise polynomial
  - Integrate the approximate
- There are three kinds of NC Methods:
  - Midpoint
  - Trapezoid
  - Simpson's

Midpoint

- This method is for my teaching purpose
- The rule states as

$$\int_a^b f(x)dx \approx (b-a)f\left(\frac{a+b}{2}\right) + \frac{(b-a)^3}{24}f''(\xi) \tag{36}$$

  - I did not use any end points, I use not knots
  - But this can work if $|b-a|$ is small or f'' is small
    - Why? Since then it would be a linear function, Then the integral is a ...
    - U do not know where is ξ

Midpoint: How will u use it?

- Because using only one-point is far too coarse (for most functions), it is very common to use a composite rule.
- Composite rules break $[a, b]$ into smaller intervals and use the rule for each and sum them up.

Midpoint

- Partition $a = z_1 < \ldots < z_{n+1} = b$
- And select out the midpoint $x_i = .5\left(z_i + z_{i+1}\right)$
- Hence for each interval, I have size $h_i = z_{i+1} - z_i$
- Formula

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} \left(h_i f\left(x_i\right) + \frac{h_i^3}{24} f''\left(\xi_i\right) \right) \tag{37}$$

where $\xi_i \in [z_i, z_{i+1}]$
How to graph it?

TRAPEZOID RULE

- Trapezoid rule.

$$\int_a^b f(x)dx \approx \frac{b-a}{2}(f(a)+f(b)) - \frac{(b-a)^3}{12}f''(\xi) \tag{38}$$

  where $\xi \in [a, b]$
- Still pretty brute force

TRAPEZOID RULE

- Direct Partition $a = x_1 < \ldots < x_n = b$
- Hence the interval size would be $h_i = x_{i+1} - x_i$
- Formula

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n-1} \left(\frac{h_i}{2} \left(f\left(x_{i+1}\right) + f\left(x_i\right) \right) - \frac{h_i^3}{12} f''\left(\xi_i\right) \right) \tag{39}$$

  where $\xi_i \in [x_i, x_{i+1}]$
- U see I am using $\sum_{i=1}^{n} \omega_i f\left(x_i\right)$
- Both methods, you will choose ξ_i

SIMPSON'S RULE

- Generally
  - midpoint rule as an approximation by step functions
  - trapezoid rule as an approximation by p-w linear functions
  - But one can use quadratic approximation, this is Simpson
- Formula

$$\int_a^b f(x)dx \approx \frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)+\frac{1}{90}\left(\frac{b-a}{2}\right)^5 \left|f^4(\xi\right.$$

Newton-Cotes Methods Wrappup

- To approximate the integral of a real valued function defined on a bounded interval [a, b] of the real line
- Set $h = (b-a)/n$ and $x_i = a + (i-1)b$
- Trapezoid:

$$\int_{x_i}^{x_{i+1}} f(x) \approx \frac{f(x_i) + f(x_{i+1})}{2} h$$

- Then

$$\int_a^b f(x) \approx \sum_{i=1}^{N+1} w_i f(x_i), \quad w_{i \neq 1, N+1} = h, w_1 = w_{N+1} = h/2 \tag{40}$$

- Simpson's rule: use three points and quadratic form to approximate the function (preferred to the trapezoid rule when the integrand f is smooth)
- Hence you need even number of intervals
- In other words, odd number of points

$$\int_{x_{2i-1}}^{x_{2i+1}} f(x)dx = \frac{h}{3}[f(x_{2i-1}) + 4f(x_{2i}) + f(x_{2i+1})] \tag{41}$$

Gaussian Rules

- Use n points to exactly integrate all polynomials of degree $2n-1$ or less.
- The integration is performed with respect to a positive weight function $w(.)$
  Say

$$\int g(x)dx = \int f(x)w(x)dx \tag{42}$$

- Why?
- E.g.

$$\int_{-1}^{1} g(x)dx \text{ for } g(x) = \log(x+1) \tag{43}$$

GAUSSIAN QUADRATURE

Theorem

$\{p_i\}_{i=0}^n$ are orthogonal polynomials with degree i, If the quadrature nodes $\{x_i\}$ are the n roots of p_n, then there exits $\{\omega_i\}_{i=1}^N$ s.t.

$$\int_D h(x)w(x)dx = \sum_{i=1}^{n} \omega_i h(x_i) \tag{44}$$

where h is s any polynomial of degree $2n-1$ or less

Gaussian Quadrature Theorem

- Knots

$$\{x_i | p_n(x_i) = 0\} \tag{45}$$

- Weights?

$$\omega_i = \frac{a_n}{a_{n-1}} \frac{\langle p_{n-1}, p_{n-1} \rangle}{p_n'(x_i) p_{n-1}(x_i)} \tag{46}$$

where a_n is the coefficient of x^n in p_n

- U can take Chebyshev polynomials right?

$$T_0(x) = 1, T_1(x) = x, T_2 = 2x^2 - 1$$
$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x)$$

- Hence $a_n = 2a_{n-1}$
- Different Methods use different weighting functions and for different domains.

GAUSSIAN QUADRATURE

- Gaussian quadrature rules are constructed with respect to specific weighting functions.
- Given weighting function $w(x)$, weights and points w_i, x_i are determined through

$$\int_I x^k w(x)dx = \sum_i w_i x_i^k, \ for\ k = 0, 1, \ldots 2n-1 \tag{47}$$

- Types
  - Gauss-Legendre: $w(x) = 1$; if f can be closely approximated by a polynomial, a Gauss-Legendre quadrature should provide an accurate approximation to the integral. Not good with kinks.
  - One can also interprete the w as density. How shall we approximate a standard normal dist? 6 moments.

$$x_1 = -\sqrt{3}, x_2 = 0, x_3 = \sqrt{3}, w_1 = 1/6, w_2 = 2/3, w_3 = 1/6 \tag{48}$$

  - Gauss-Hermite $w(x) = exp(-x^2)$
  - Density– Simulation

GAUSS-LEGENDRE

- This formula tells u

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} \omega_i f(x_i) \tag{49}$$

- x_i are roots of Legendre polynomials and weights are

$$\omega_i = \frac{2}{(1-x_i)^2 P_n'(x_i)^2} \tag{50}$$

- What.....

LEGENDRE POLY

- Legendre Poly are

$$P_{n+1}(x) = \frac{(2n+1) \times P_n(x) - nP_{n-1}(x)}{n+1} \tag{51}$$

$$P_0 = 1,\ P_1(x) = x \tag{52}$$

- Derivatives

$$P'_n(x) = \frac{-nxP_n(x) + nP_{n-1}(x)}{1-x^2} \tag{53}$$

- Ok what are their roots? Bisection method to find out and save it ,or Wiki it....

Monte Carlo

- Draw r.v from the density
- Take the mean

$$\lim_{n\to\infty} \frac{1}{n} \sum_i f(x_i) = \int_a^b f(x)dx = E[x] \tag{54}$$

How is x distributed?

- U may want to look into my MCMC slides

ODE

- I will show several methods to solve ODE, which is common when one use Continuous time model
- To start with, I will solve ODE with initial condition
- Another type? BVP
- with $x(0) = x_0$, $x(t)$ solves $x'(t) = f(x,t)$
- What if sec order $y'' = f(y', y, t)$?

$$\begin{pmatrix} y' \\ z' \end{pmatrix} = \begin{pmatrix} z \\ f(z,y,t) \end{pmatrix} = g(z,y,t) \tag{55}$$

Euler's Method

- Solve it iteratively

$$x_{i+1} = x_i + hf(x_i, t_i), t_i = iT/N \tag{56}$$

  For rough approximation, fine
- Runge-Kutta Method is the most used one.
- High order RK is tedious but straightforward. As illustration, I will use order 2

$$x(t+h) = x(t) + x'(t)h + \frac{1}{2}x''(t)h^2 + O(h^3) \tag{57}$$

$$= x + fh + \frac{1}{2}(f_1' + f_2'f) + O(h^3) \tag{58}$$

- And $f(t+\lambda h, x+\lambda hf) = f(t,x) + \lambda h(f_1' + f_2'f) + O(h^2)$

Runge-Kutta Method

- And $f(t+\lambda h, x+\lambda h f) = f(t,x) + \lambda h(f_1' + f_2' f) + O(h^2)$ Then

$$x(t+h) = x + h[(1 - \frac{1}{2\lambda})f(t,x) + \frac{1}{2\lambda} f(t+\lambda h, x+\lambda h f)] \quad (59)$$

- Set $\lambda = 2/3$ then

$$x(t+h) = x + h/4[f(t,x) + 3f(t+2h/3, x+2hf/3)] \quad (60)$$

- One should not feel too surprise on the last equation, for example:

$$\begin{aligned} x(t+h) &= x(t) + \int_t^{t+h} f(x(\tau),\tau)d\tau \\ &= x(t) + \frac{h}{2}(f(x_t,t) + f(x_{t+h}, t+h)) \\ &= x + \frac{h}{2}[f(x_t,t) + f(x_t + fh, t+h)] \end{aligned}$$

Last equation with $\lambda = 1$

Runge-Kutta Method

- The most widely used Runge-Kutta method is the classical fourth-order method

$$\begin{aligned} x(t+h) &= x + (y_1 + 2(y_2 + y_3) + y_4)/6 \\ y_1 &= hf(t, x) \\ y_2 &= hf(t + h/2, x + y_1/2) \\ y_3 &= hf(t + h/2, x + y_2/2) \\ y_4 &= hf(t + h, x + y_3) \end{aligned}$$

- Numerous other methods, Read ODE32, ODE45 from Matlab
- But I will not cover continuous time model here.

For Further Reading

- http://ab-initio.mit.edu/wiki/index.php/NLopt_Algorithms