CS 450: Numerical Anlaysis[1]

Numerical Optimization

University of Illinois at Urbana-Champaign

[1] *These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book "Scientific Computing: An Introductory Survey" by Michael T. Heath (slides).*

Numerical Optimization

- Our focus will be on *continuous* rather than *combinatorial* optimization:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0} \quad \text{and} \quad \boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{0}$$

- We consider linear, quadratic, and general nonlinear optimization problems:

Local Minima and Convexity

- Without knowledge of the analytical form of the function, numerical optimization methods at best achieve convergence to a *local* rather than *global* minimum:

- A set is *convex* if it includes all points on any line, while a function is (strictly) convex if its (unique) local minimum is always a global minimum:

Existence of Local Minima

- *Level sets* are all points for which f has a given value, *sublevel sets* are all points for which the value of f is less than a given value:

- If there exists a closed and bounded sublevel set in the domain of feasible points, then f has has a global minimum in that set:

Optimality Conditions

- If $\boldsymbol{x}$ is an interior point in the feasible domain and is a local minima,

$$\nabla f(\boldsymbol{x}) = \left[\frac{df}{dx_1}(\boldsymbol{x}) \quad \cdots \frac{df}{dx_n}(\boldsymbol{x}) \right]^T = \mathbf{0} :$$

- *Critical points* $\boldsymbol{x}$ satisfy $\nabla f(\boldsymbol{x}) = \mathbf{0}$ and can be minima, maxima, or saddle points:

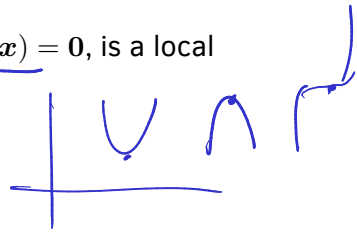
Hessian Matrix

$\min_x f(x)$

- To ascertain whether a critical point $\boldsymbol{x}$, for which $\nabla f(\boldsymbol{x}) = \mathbf{0}$, is a local minima, consider the *Hessian matrix*:

$H_f(x) = H_f(x')$ QP

$H_{f(x)_{ij}} = \left[\frac{\partial^2 f}{\partial x_i \partial x_j}\right]$



- If $\boldsymbol{x}^*$ is a minima of f, then $\boldsymbol{H}_f(\boldsymbol{x}^*)$ is positive semi-definite:

sufficient

necessary

positive definiteness $x^T H x > 0 \Rightarrow$ local minimum

negative definiteness $\Rightarrow$ maximum

indefiniteness $\lambda_i(H) > 0$, $\lambda_j(H) < 0$ $\Rightarrow$ saddle point

$f(x+\epsilon) \geq f(x)$

$\Rightarrow$ local minima

cannot have $\nabla f(x) \neq 0$

$f(x \pm \epsilon \nabla f(x)) \overset{<}{\underset{>}{\text{or}}} f(x)$

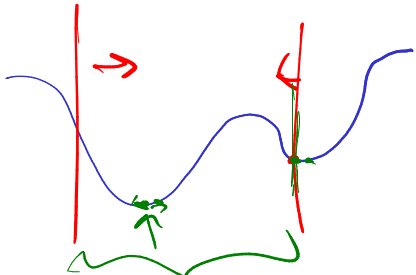
cannot have $s^T H_f(x) s < 0$

$f(x + \epsilon s) = f(x) + \epsilon \nabla f(x)^T s + \frac{\epsilon^2}{2} s^T H_f(x) s + O(\epsilon^3)$

≥ 0

$\nabla f(x)^T s = 0$

$f(x+\epsilon s) < f(x)$



equality $g(x) = 0$

inequality $h(x) \leq 0$

$$g(x + \epsilon s) = g(x) + \epsilon J_g(x) s + O(\epsilon^2)$$

$$-\nabla f(x) = J_g^T(x) \lambda \quad \text{for some } \lambda$$

$$\begin{bmatrix} 0 \\ -3 \\ 0 \\ \vdots \\ 2 \end{bmatrix}$$

$$g(x + \epsilon s) = \underset{0}{g(x)} + \underset{0}{\epsilon J_g(x) s}$$

Optimality on Feasible Region Border

- Given an equality constraint $\boldsymbol{g}(\boldsymbol{x}) = \mathbf{0}$, it is no longer necessarily the case that $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$. Instead, it may be that directions in which the gradient decreases lead to points outside the feasible region:

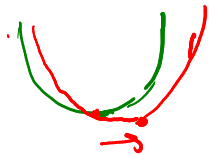
$$\exists \boldsymbol{\lambda} \in \mathbb{R}^n, \quad -\nabla f(\boldsymbol{x}^*) = \boldsymbol{J}_{\boldsymbol{g}}^T(\boldsymbol{x}^*)\boldsymbol{\lambda}$$

- Such *constrained minima* are critical points of the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$, so they satisfy:

$$\nabla \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}) = \begin{bmatrix} \nabla f(\boldsymbol{x}^*) + \boldsymbol{J}_{\boldsymbol{g}}^T(\boldsymbol{x}^*)\boldsymbol{\lambda} \\ \boldsymbol{g}(\boldsymbol{x}^*) \end{bmatrix} = \mathbf{0}$$

Sensitivity and Conditioning

- The condition number of solving a nonlinear equations is $1/f'(x^*)$, however for a minimizer x^*, we have $f'(x^*) = 0$, so conditioning of optimization is inherently bad:



- To analyze worst case error, consider how far we have to move from a root x^* to perturb the function value by ϵ:

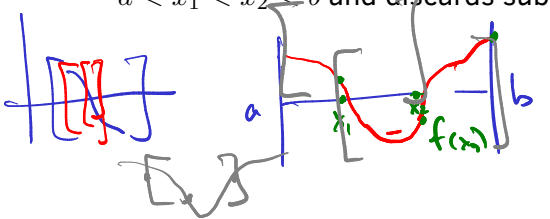
$$\epsilon = f(x^* + h) - f(x^*) = \underbrace{f'(x^*)}_{0} h + \frac{1}{2} f''(x) h^2 + O(h^3)$$

$$\epsilon \sim h^2 \quad \Rightarrow \quad h = O(\sqrt{\epsilon})$$

Golden Section Search

Demo: *Golden Section Proportions*

- Given bracket $[a, b]$ with a unique local minimum (f is *unimodal* on the interval), *golden section search* considers consider points $f(x_1), f(x_2)$, $a < x_1 < x_2 < b$ and discards subinterval $[a, x_1]$ or $[x_2, b]$:



$f(a)$
$f(x_1)$? which is larger
$f(x_2)$
$f(b)$

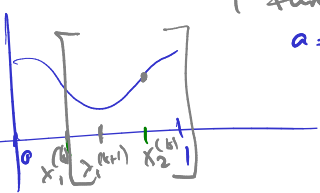
$[a, x_2]$ $[b, x_1]$

- Since one point remains in the interval, golden section search selects x_1 and x_2 so one of them can be effectively reused in the next iteration:

$b - a = 1$

linear conv rate

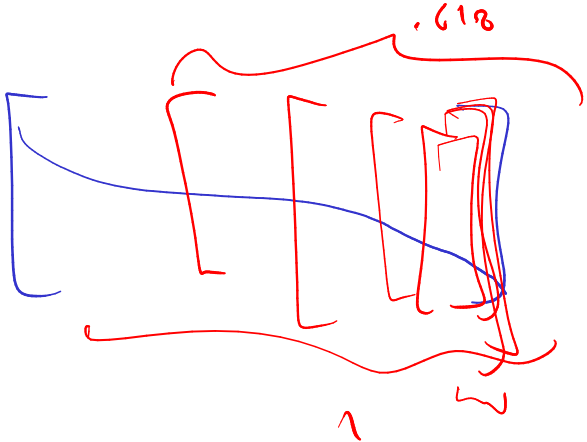
$C = \frac{\sqrt{5}-1}{2} \approx .618$



1 fun-eval / iteration

$a = 0, b = 1$

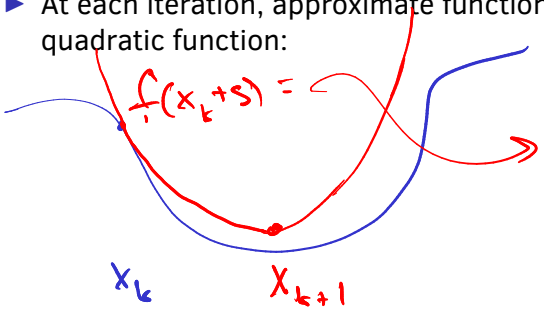
$x_2 = \frac{\sqrt{5}-1}{2}$ $x_1 = 1 - x_2$


.618
1

Newton's Method for Optimization

Demo: Newton's Method in 1D

- At each iteration, approximate function by quadratic and find minimum of quadratic function:



$$f(x_k) + f'(x_k)\,s + \frac{1}{2} f''(x_k)\,s^2$$

$$s = -f'(x_k) / f''(x_k)$$

$$-h(x) / h'(x)$$

- The new approximate guess will be given by $x_{k+1} - x_k = -f'(x_k)/f''(x_k)$:

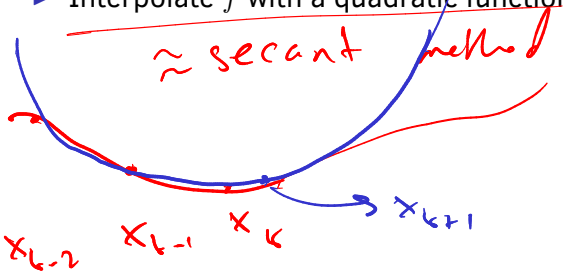
= Newton's method

$f'(x) = 0$

$h(x)$

Successive Parabolic Interpolation

- Interpolate f with a quadratic function at each step and find its minima:

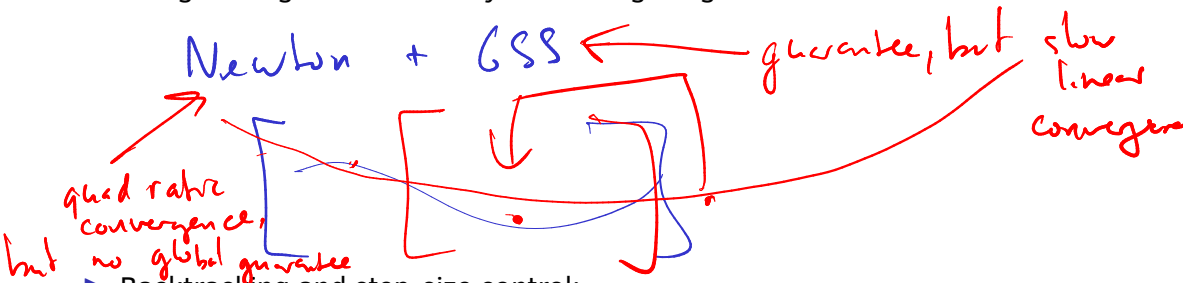


- The convergence rate of the resulting method is roughly 1.324

superlinear
quadratic

Safeguarded 1D Optimization

- Safeguarding can be done by bracketing via golden section search:



- Backtracking and step-size control:

$$x_{k+1} = x_k - \alpha_k f'(x_k) / f''(x_k)$$

- backtrack α_k if $f(x_{k+1}) > f(x_k)$

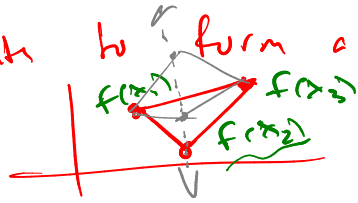
General Multidimensional Optimization

Demo: *Nelder-Mead Method*

- Direct search methods by simplex (*Nelder-Mead*):

n-dim opt

n+1 points to form a simplex



- Steepest descent: find the minimizer in the direction of the negative gradient:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

$$\min_{\alpha_k} g(\alpha_k) = f(x_k - \alpha_k \nabla f(x_k))$$

1D optimization

Convergence of Steepest Descent

- Steepest descent converges linearly with a constant that can be arbitrarily close to 1:

  - rate depends on Hessian at x^*
  - if gradient changes rapidly, local approximation is poor

- Given quadratic optimization problem $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x}$ where $\boldsymbol{A}$ is symmetric positive definite, the error $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}^*$ satisfies

$$\|e_{k+1}\|_A = e_{k+1}^T A e_{k+1} = \left(\frac{\sigma_{max} - \sigma_{min}}{\sigma_{max} + \sigma_{min}}\right) \|e_k\|_A$$

$$= \frac{\kappa(A) - 1}{\kappa(A) + 1} \|e_k\|_A$$

Gradient Methods with Extrapolation

- We can improve the constant in the linear rate of convergence of steepest descent by leveraging *extrapolation methods*, which consider two previous iterates (maintain *momentum* in the direction $\boldsymbol{x}_k - \boldsymbol{x}_{k-1}$):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1})$$

- The *heavy ball method*, which uses constant $\alpha_k = \alpha$ and $\beta_k = \beta$, achieves better convergence than steepest descent:

$$\|e_{k+1}\|_A = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \|e_k\|_A$$

Nesterov's gradient optimization

Conjugate Gradient Method

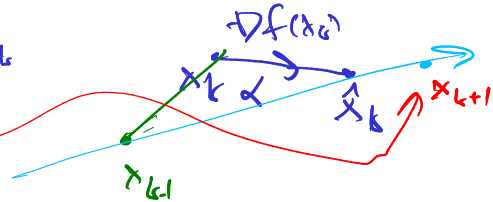
- The *conjugate gradient method* is capable of making the optimal choice of α_k and β_k at each iteration of an extrapolation method:

$$(\alpha_k, \beta_k) = \underset{\alpha_k, \beta_k}{\operatorname{argmin}} f(x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}))$$

Krylov / Lanczos

- *Parallel tangents* implementation of the method proceeds as follows

1. steepest descent $\hat{x}_k$
2. x_{k+1} :



Nonlinear Conjugate Gradient

- Various formulations of conjugate gradient are possible for nonlinear objective functions, which differ in how they compute β below
- Fletcher-Reeves is among the most common, computes the following at each iteration
  1. Perform 1D minimization for α in $f(\boldsymbol{x}_k + \alpha \boldsymbol{s}_k)$
  2. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \boldsymbol{s}_k$
  3. Compute gradient $\boldsymbol{g}_{k+1} = \nabla f(\boldsymbol{x}_{k+1})$
  4. Compute $\beta = \boldsymbol{g}_{k+1}^T \boldsymbol{g}_{k+1} / (\boldsymbol{g}_k^T \boldsymbol{g}_{k+1})$
  5. $\boldsymbol{s}_{k+1} = -\boldsymbol{g}_{k+1} + \beta \boldsymbol{s}_k$

Conjugate Gradient for Quadratic Optimization

- Conjugate gradient is an optimal iterative method for quadratic optimization, $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$
- For such problems, it can be expressed in an efficient and succinct form, computing at each iteration
  1. $\alpha = \boldsymbol{r}_k^T\boldsymbol{r}_k / \boldsymbol{s}_k^T\boldsymbol{A}\boldsymbol{s}_k$
  2. $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha\boldsymbol{s}_k$
  3. Compute gradient $\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k\boldsymbol{A}\boldsymbol{s}_k$
  4. Compute $\beta = \boldsymbol{r}_{k+1}^T\boldsymbol{r}_{k+1} / (\boldsymbol{r}_k^T\boldsymbol{r}_{k+1})$
  5. $\boldsymbol{s}_{k+1} = \boldsymbol{r}_{k+1} + \beta\boldsymbol{s}_k$
- Note that for quadratic optimization, the negative gradient $-\boldsymbol{g}$ corresponds to the residual $\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$

Krylov Optimization

Demo: *Conjugate Gradient Parallel Tangents as Krylov Subspace Method*

- Conjugate Gradient finds the minimizer of $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$ within the Krylov subspace of $\boldsymbol{A}$:

$$\mathcal{K}(A,b) = \text{span}(b, Ab, \dots A^{r-1}b) = \text{span}(Q)$$

$$\min_{x \in \mathcal{K}(A,b)} f(x) = \min_{y \in \mathbb{R}^r} = \frac{1}{2} y^T Q^T A Q y - b^T Q y$$

$$x = Qy$$

T (tridiag)

$$x = Qy$$

$$T y^* = Q^T b = \|b\|_2 e_1$$

$$y^* = \|b\|_2 T^{-1} e_1$$

Newton's Method

Demo: Newton's Method in n dimensions

- Newton's method in n dimensions is given by finding minima of n-dimensional quadratic approximation:

= solving $\nabla f(x) = 0$

$$f(x_k + s) \approx \hat{f}(s) = f(x_k) + s^T \nabla f(x_k) + \frac{1}{2} s^T H_f(x_k) s$$

$$0 = \nabla \hat{f}(s) = \nabla f(x_k) + H_f(x_k)\, s$$

$$H_f(x_k)\, s = -\nabla f(x_k)$$

?

$$x_{k+1} = x_k + s = x_k - H_f(x_k)^{-1} \nabla f(x_k)$$

quad. conv.

Quasi-Newton Methods

- *Quasi-Newton* methods compute approximations to the Hessian at each step:

$$x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f(x_k)$$

α_k – line search

- The *BFGS* method is a secant update method, similar to Broyden's method:

rank-2 update to B_k, using $s_k = x_{k+1} - x_k$, $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}$$

$O(n^2)$ work / step by Sherman-Morrison

Nonlinear Least Squares

- An important special case of multidimensional optimization is *nonlinear least squares*, the problem of fitting a nonlinear function $f_{\boldsymbol{x}}(t)$ so that $f_{\boldsymbol{x}}(t_i) \approx y_i$:

- We can cast nonlinear least squares as an optimization problem and solve it by Newton's method:

Gauss-Newton Method

- The Hessian for nonlinear least squares problems has the form:

- The *Gauss-Newton* method is Newton iteration with an approximate Hessian:

- The Levenberg-Marquardt method incorporates Tykhonov regularization into the linear least squares problems within the Gauss-Newton method.

Constrained Optimization Problems

- We now return to the general case of *constrained* optimization problems:

- Generally, we will seek to reduce constrained optimization problems to a series of unconstrained optimization problems:
  - *sequential quadratic programming*:
  - *penalty-based methods*:
  - *active set methods*:

Sequential Quadratic Programming

- *Sequential quadratic programming* (SQP) corresponds to using Newton's method to solve the equality constrained optimality conditions, by finding critical points of the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$,

- At each iteration, SQP computes $\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{s}_k \\ \boldsymbol{\delta}_k \end{bmatrix}$ by solving

Inequality Constrained Optimality Conditions

- The *Karush-Kuhn-Tucker (KKT)* conditions hold for local minima of a problem with equality and inequality constraints, the key conditions are

- To use SQP for an inequality constrained optimization problem, consider at each iteration an *active set* of constraints:

Penalty Functions

- Alternatively, we can reduce constrained optimization problems to unconstrained ones by modifying the objective function. *Penalty* functions are effective for equality constraints $\boldsymbol{g}(\boldsymbol{x}) = 0$:

- The augmented Lagrangian function provides a more numerically robust approach:

Barrier Functions

- *Barrier functions* (*interior point methods*) provide an effective way of working with inequality constraints $\boldsymbol{h}(\boldsymbol{x}) \leq \mathbf{0}$: