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:

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

- 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

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

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

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

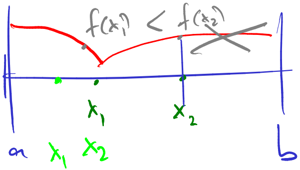
- The condition number of solving a nonlinear equations is $1/f'(x^*)$, however for a minimizer x^*, we have $f'(x^*) = 0$, so conditioning of optimization is inherently bad:

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

Demo: *Golden Section Proportions*

Golden Section Search

- Given bracket $[a, b]$ with a unique 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 $[x_2, b]$:



- 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:

$(\sqrt{5}-1)/2$

$(b-a)(\sqrt{5}-1)/2$ is distance from a to x_2

x_1 to b

Newton's Method for Optimization

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

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

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

Safeguarded 1D Optimization

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

- Backtracking and step-size control:

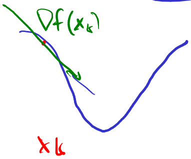
General Multidimensional Optimization

Demo: *Nelder-Mead Method*

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

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

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



via Golden Section
or any 1D optimization scheme

Convergence of Steepest Descent

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

in worst case $\alpha_k \approx 0$, always $\alpha_k > 0$

how fast ∇f changes guides converge

- 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

min $f(x)$

quadratic $\downarrow$ linear term $\downarrow$

$$\underbrace{\|e_k\|_A}_{} = \underbrace{\frac{\sigma_{max}(A) - \sigma_{min}(A)}{\sigma_{max}(A) + \sigma_{min}(A)}}_{\kappa(A) - 1 \,/\, \kappa(A) + 1} \|e_{k-1}\|_A$$

$e_k^T A e_k$, only norm if A is SPD

$\nabla f(x) = Ax + c$

$\underbrace{0 = Ax + c}_{\text{optimality condition}}$

$h(x_k + s) = h(x_k) + \underbrace{\nabla h(x_k)}_{c} s + \frac{1}{2} s^T H_h(x_k) s + \ldots$

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 $x_k - x_{k-1}$):

$$x_{k+1} = f\Big(x_k - \overbrace{\alpha_k \underbrace{\nabla f(x_k)}_{\text{gradient}} + \beta_k}^{\text{parameters}} \underbrace{(x_k - x_{k-1})}_{\text{momentum}}\Big)$$

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

$$\|e_k\|_A = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\|e_{k-1}\|_A \quad \left.\right\} \begin{array}{l}\text{near-optimal} \\ \text{Same as Conjugate Gradient}\end{array}$$

Conjugate Gradient Method

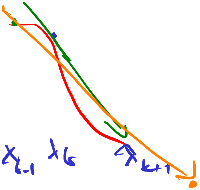
- The *conjugate gradient method* is capable of making the optimal choice of α_k and β_k at each iteration of an extrapolation method:

$$x_{k+1} = \underbrace{\operatorname*{argmin}_x}_{\text{optimal}} \Big(f\big(x_k - \alpha_k \underbrace{\nabla f(x_k)}_{\text{gradient}} + \beta_k \underbrace{(x_k - x_{k-1})}_{\text{momentum}} \big) \Big)$$

optimal α_k, β_k at each step, Lanczos implicitly

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

  - steepest descent from x_k to generate $\hat{x}_k$
  - minimize along the line from x_{k-1} to $\hat{x}_k$ to produce x_{k+1}



CG converges in n steps

Conjugate Gradient as a Krylov Subspace Method

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

$$x^* = \underset{x \in \mathcal{K}_k(A,c)}{\operatorname{argmin}} \; f(x) = \underset{\substack{x = Q_k y \\ y \in \mathbb{R}^k}}{\operatorname{argmin}} \underbrace{\frac{1}{2} y^T Q_k^T A Q_k y + c^T Q_k y}_{\frac{1}{2} y^T T_k y \; + \; \underbrace{c^T Q}_{\|c\|_2 e_1} y}$$

$$\nabla f(y) = T_k y + \|c\|_2 e_1 = 0$$

$$y = -\|c\|_2 T_k^{-1} e_1$$

$$x_k = Q_k y = -\|c\|_2 Q_k T_k^{-1} e_1$$

***Demo:** Newton's Method in n dimensions*

Newton's Method

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

Taylor expansion

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

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

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

$$x_{k+1} = x_k + s$$

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

equivalent Newton

quadratic convergence

Quasi-Newton Methods

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

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

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:

Levenberg-Marquardt Method

- The *Levenberg-Marquardt* modifies the Gauss-Newton method to use Tykhonov regularization:
- The scalar μ controls the step size through the least squares problem:

Constrained Optimization Problems

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

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

- Generally, we will seek to reduce constrained optimization problems to a series of unconstrained optimization problems:

Lagrangian Duality

- The Lagrangian function with constraints $\boldsymbol{g}(\boldsymbol{x}) = \mathbf{0}$ and $\boldsymbol{h}(\boldsymbol{x}) \leq \mathbf{0}$ is

- The Lagrangian dual problem is an unconstrained optimization problem:

$$\max_{\boldsymbol{\lambda}} q(\boldsymbol{\lambda}), \quad q(\boldsymbol{\lambda}) = \begin{cases} \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) & \text{if } \boldsymbol{\lambda} \geq \mathbf{0} \\ -\infty & \text{otherwise} \end{cases}$$

  The unconstrained optimality condition $\nabla q(\boldsymbol{\lambda}^*) = \mathbf{0}$, implies

Sequential Quadratic Programming

- *Sequential quadratic programming (SQP)* reduces a nonlinear equality constrained problem to a sequence of constrained quadratic programs via a Taylor expansion of the Lagrangian function $\mathcal{L}_f(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$:

- SQP ignores the constant term $\mathcal{L}_f(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ and minimizes $\boldsymbol{s}$ while treating $\boldsymbol{\delta}$ as a Lagrange multiplier:

Sequential Quadratic Programming

- From a different viewpoint, sequential quadratic programming corresponds to using Newton's method to solve the nonlinear equations,

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

Active Set Methods

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

- The Karush-Kuhn-Tucker (KKT) optimality conditions given the generalized Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \boldsymbol{\mu}^T \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\nu}^T \boldsymbol{h}(\boldsymbol{x})$ are

$$
\begin{aligned}
\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) &= \mathbf{0} \\
\boldsymbol{g}(\boldsymbol{x}) &= \mathbf{0} \\
\boldsymbol{h}(\boldsymbol{x}) &\leq \mathbf{0} \\
\boldsymbol{\nu} &\geq \mathbf{0} \\
\boldsymbol{\nu}^T \boldsymbol{h}(\boldsymbol{x}) &= \mathbf{0}
\end{aligned}
$$

  at an optimal point, we must have that for either the ith inequality constraint is active, so $h_i(\boldsymbol{x}) = \mathbf{0}$ or it is inactive, but its Lagrange multiplier $\nu_i = 0$.

Penalty Functions

- 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

- A drawback of penalty function methods is that they can produce infeasible approximate solutions, which is problematic if the objective function is only defined in the feasible region:

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