

Numerical Optimization Methods

Chapter 1 Introduction

Optimize : Investment portfolios (risk vs return)
 Manufacturing efficiency
 Minimum energy under constraints
 Maximum time under constraints.

Objective Function : A quantitative measure of the performance of the system under study.

Examples : profit, time, potential energy, error probability

Modeling : The process of identifying the objective, variables, and constraints.

Algorithm : The procedure used to solve the optimization problem characterized by the model.

Optimality Conditions : The set of mathematical expressions that must be satisfied by the optimal solution of the problem at hand.

Sensitivity Analysis : A study of how the solution changes with perturbations in the model and data.

(2)

* Mathematical Formulation: Optimization problems consist of minimizing/maximizing an objective function with respect to (w.r.t.) its variables, subject to some constraints (equality or inequality).

x : vector of variables (unknowns, parameters)

$f(x)$: objective function (scalar-valued)

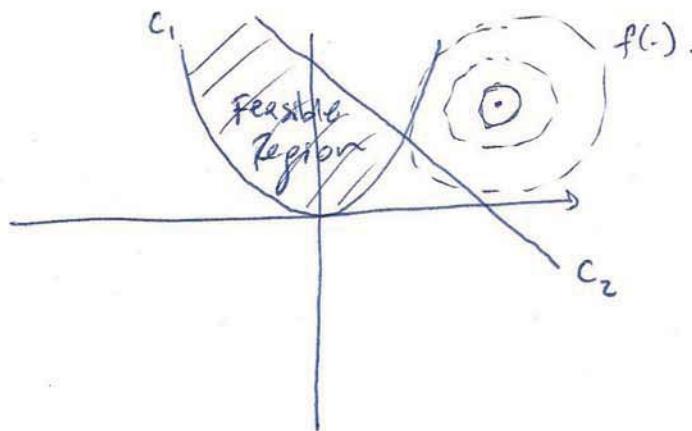
$c_i(x)$: constraint functions (scalar-valued)

* Example: $\min_{x \in \mathbb{R}^n} f(x)$ subject to $c_i(x) = 0 \quad i \in \mathcal{E}$
 $c_i(x) \geq 0 \quad i \in \mathcal{I}$

Here \mathcal{E} and \mathcal{I} are the index sets for equality and inequality constraints.

* Ex Consider $\min_{x_1, x_2} (x_1 - 2)^2 + (x_2 - 1)^2$ s.t. $x_1^2 - x_2 \leq 0$
 $x_1 + x_2 \leq 2$

Here $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$, $c(x) = \begin{cases} -x_1^2 + x_2 \\ -x_1 - x_2 + 2 \end{cases}$
 $\mathcal{E} = \emptyset$, $\mathcal{I} = \{1, 2\}$.



* Continuous vs Discrete optimization

In some optimization problems, the variables make sense only if they take integer values:

e.g. x_i = the number of type i items to be produced

In such problems, $x_i \in \mathbb{Z}$ (the set of integers)

or in others $x_i \in \{0, 1\}$ (binary-valued)

These are called integer programming problems.

- * In such discrete optimization problems, the feasible set consists of (usually very large) finite set of possible/allowed parameter values.
- * In contrast, our focus is on continuous optimization problems where the feasible set has uncountably infinite members.

* Constrained vs Unconstrained Optimization

If the problem does not impose any constraints on x (i.e. $\mathcal{E} = \mathcal{I} = \emptyset$) then we have an unconstrained problem.

$\mathcal{I} = \emptyset \neq \mathcal{E}$ yields a problem with equality constraints.

If $f(\cdot)$ and $c_i(\cdot)$ are all linear in x , then we have a linear programming problem at hand.

(4)

Global and Local Optimization

Many algorithms for nonlinear optimization seek only a local solution. Their convergence to the global optimum is not guaranteed.

In general global solutions are difficult to recognize and locate. For convex programming problems (including linear programming), there exists a unique local solution, which is also the global optimum.

For global optimization, annealing techniques make use of repeated local optimization problem solvers.

Stochastic and Deterministic Optimization

If exact values of some parameters in the model are unknown, instead of using a single best guess, we can employ probabilistic models. This leads to stochastic optimization (typical in statistical inference).

Chance-constrained optimization: $x \in$ feasible set with pre-specified probability.

Robust optimization: $x \in$ feasible set with probability = 1. or for all cases.

(5)

Convexity

Convex feasible sets : S convex iff $\forall x, y \in S, \alpha x + (1-\alpha)y \in S$

Convex Objective functions. $f(\cdot)$ convex iff $\forall x, y \quad f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$

$f(\cdot)$ is strictly convex if $<$ for $\alpha \in (0, 1)$.

$f(\cdot)$ is concave if $-f$ is convex.

Fact: If f is convex and the feasible set S is convex,
then the local minimum is also the global minimum.

Convex Programming : Objective f is convex

c_i for $i \in E$ are linear

c_i for $i \in I$ are concave

Optimization Algorithms

These are iterative procedures that start from an initial guess and generate a sequence of improved guesses called iterates. Different algorithms have differing strategies in moving to the next iterate. Desirable properties are:

Robustness: Perform well for a wide variety of problems and initial points.

Efficiency: Do not require excessive ~~storage~~ and computation

Accuracy: Identify a solution with precision without being too sensitive to errors in data or rounding.

(6)

Appendix A: Background Material

Vectors and Matrices: We will primarily deal with vectors and matrices composed of real numbers.

$$x \in \mathbb{R}^n \Leftrightarrow x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad A \in \mathbb{R}^{n \times m} \Leftrightarrow A = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}$$

* x is assumed to be a column vector by default.

$x \geq 0$ means $x_i \geq 0 \forall i$ ($\geq, >$ apply elementwise for vectors).

* Inner product of $x, y \in \mathbb{R}^n$: $x^T y = \sum_{i=1}^n x_i y_i$
Square

* Matrix $A \in \mathbb{R}^{n \times n}$ is symmetric iff $A = A^T$ ($A_{ij} = A_{ji}$).
~~(\leftrightarrow)~~ there exists

Defn: Square matrix A is positive definite iff $\exists \alpha \in \mathbb{R}$

~~such that~~ $\xrightarrow{\text{def}} \forall x \in \mathbb{R}^n \quad x^T A x \geq \alpha x^T x$

It is positive semidefinite if $x^T A x \geq 0 \quad \forall x \in \mathbb{R}^n$

Thm: Symmetric matrix A is positive definite ($A > 0$) iff all of its eigenvalues are positive ($\lambda_A > 0$).

Similarly, $A \geq 0$ (pos. semi-def.) iff $\lambda_A \geq 0$.

We can define negative (semi-) definite similarly.

(7)

Diagonal A, Lower Triangular A, Upper Triangular

$$\begin{bmatrix} a_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & a_{nn} \end{bmatrix}$$

$$\begin{bmatrix} \Delta & 0 \\ \times & \Delta \end{bmatrix}$$

$$\begin{bmatrix} 0 & \Delta \\ 0 & \times \end{bmatrix}$$

I denotes the identity matrix $\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$.

Defn: A square $n \times n$ matrix A is nonsingular if $\forall b \in \mathbb{R}^n, \exists x \in \mathbb{R}^n \nexists Ax = b$.

Fact: If $A \in \mathbb{R}^{n \times n}$ is nonsingular, then $\exists B \in \mathbb{R}^{n \times n} \nexists$.

$AB = BA = I$. We denote $B = A^{-1}$, the inverse of A .

Fact: $(A^T)^{-1} = (A^{-1})^T$

Defn: $Q \in \mathbb{R}^{n \times n}$ is orthogonal iff $QQ^T = Q^TQ = I$.

* The inverse of an orthogonal matrix is its transpose.

Norms L_p -norm for vectors: $\|x\|_p = \left(\sum_i |x_i|^p \right)^{1/p}$

$$L_1: \|x\|_1 = \sum_i |x_i|$$

$$L_2: \|x\|_2 = \left(\sum_i x_i^2 \right)^{1/2} \text{ Euclidean norm}$$

$$L_\infty: \|x\|_\infty = \max_i |x_i|$$

Defn: $\|\cdot\|: \mathbb{R}^n \rightarrow \mathbb{R}_+ \cup \{0\}$ where

$$\|x+z\| \leq \|x\| + \|z\| \quad \forall x, z \in \mathbb{R}^n \quad \checkmark \quad \text{Triangle inequality}$$

$$\|x\| = 0 \Leftrightarrow x = 0$$

$$\|\alpha x\| = |\alpha| \cdot \|x\| \quad \forall \alpha \in \mathbb{R}, x \in \mathbb{R}^n \quad \text{Scaling.}$$

Equality holds iff
 $z = \beta x, \beta \in \mathbb{R} \setminus \{0\}$

(8)

Cauchy-Schwarz Inequality: $|x^T z| \leq \|x\|_2 \cdot \|z\|_2$.

with equality iff $z = \beta x$, $\beta \in \mathbb{R}_+ \cup \{0\}$.

$$\text{Proof: } 0 \leq \|\alpha x + z\|^2 = \alpha^2 \|x\|^2 + 2\alpha x^T z + \|z\|^2$$

Since the right side is a quadratic polynomial of α , its roots must be both complex-valued for the inequality to hold $\forall x, z, \alpha$. Then. ($\alpha x^2 + b x + c \overset{\text{no real roots}}{\Rightarrow} b^2 \leq 4ac$)

$$(2x^T z)^2 \leq 4\|x\|^2 \cdot \|z\|^2$$

$$\Leftrightarrow |x^T z| \leq \|x\| \cdot \|z\| \quad \square.$$

Equality occurs iff α has a single (repeated) real root, which is when $\alpha x + z = 0$ for some α .

Dual Norm: Any norm $\|\cdot\|$ has a dual norm $\|\cdot\|_D$ defined by

$$\|x\|_D = \max_{\|y\|=1} x^T y$$

Fact: $\|\cdot\|_1$ and $\|\cdot\|_\infty$ are duals of each other

$\|\cdot\|_2$ is its own dual.

Matrix Norms: $\|A\| \triangleq \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}$ A matrix norm defined in this manner is said to be consistent with vector norms.

Ex $\|A\|_1 = \max_j \sum_i |A_{ij}|$ consistent with vector norms.

$$\|A\|_2 = \text{Largest eigenvalue of } (A^T A)^{1/2}$$

$$\|A\|_\infty = \max_i \sum_j |A_{ij}|$$

Frobenius Norm
 $\|A\|_F = \left(\sum_i \sum_j A_{ij}^2 \right)^{1/2}$

Not consistent with any vector norm!

(9)

Fact: For the Euclidean norm $\|AB\|_2 \leq \|A\|_2 \cdot \|B\|_2$.

Defn: The condition number of a nonsingular matrix is $K(A) \triangleq \|A\| - \|A^{-1}\|$ for any norm $\|\cdot\|$.

By convention $K(A)$ means $K_2(A)$ using $\|\cdot\|_2$.

Others are $K_1(A)$, $K_\infty(A)$, etc.

Fact: For some continuous scalar/vector/matrix-valued function $F(\cdot)$, we have $\left\| \int_a^b F(t) dt \right\| \leq \int_a^b \|F(t)\| dt$ for any norm $\|\cdot\|$.

Subspaces: Given the Euclidean space \mathbb{R}^n , the subset $S \subset \mathbb{R}^n$ is a subspace of \mathbb{R}^n iff

$$\bullet x, y \in S \Leftrightarrow \alpha x + \beta y \in S \quad \forall \alpha, \beta \in \mathbb{R}$$

Ex: Given $a_i \in \mathbb{R}^n$, $i=1, 2, \dots, m$ ($m < n$), then

$$S = \left\{ w \in \mathbb{R}^n \mid a_i^T w = 0, \forall i \right\} \text{ is a subspace}$$

However

$$S = \left\{ w \in \mathbb{R}^n \mid a_i^T w \geq 0 \quad \forall i \right\} \text{ is } \underline{\text{not}} \text{ a subspace.}$$

* Consider \mathbb{R}^3 and $a_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, $a_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$. Then $S = \left\{ w \in \mathbb{R}^3 \mid w_1^2 + w_2^2 = 0 \right\}$

$$S = \left\{ w \in \mathbb{R}^3 \mid \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right\} \equiv \left\{ w \in \mathbb{R}^3 \mid w = \begin{bmatrix} 0 \\ 0 \\ w_3 \end{bmatrix}, w_3 \in \mathbb{R} \right\}$$

(10)

Defn: A set of vectors $\{s_1, s_2, \dots, s_m\}$ in \mathbb{R}^n is called a linearly independent set if $\nexists \alpha_1, \dots, \alpha_m \in \mathbb{R}$ such that

$$\alpha_1 s_1 + \dots + \alpha_m s_m = 0 \text{ except } \alpha_1 = \alpha_2 = \dots = \alpha_m = 0.$$

Fact: If $\{s_1, s_2, \dots, s_m\}$ is a linearly independent set, then none of these vectors can be written as a linear combination of the others. (Ask for proof!)

Defn: The set $\{s_1, \dots, s_m\}$ in \mathbb{R}^n is a spanning set for $S \subset \mathbb{R}^n$ if $\forall s \in S \exists \alpha_1, \dots, \alpha_m \in \mathbb{R}$ such that

$$s = \alpha_1 s_1 + \dots + \alpha_m s_m.$$

Defn: If $\{s_1, \dots, s_m\}$ is both linearly independent and is a spanning set for S , then it is a basis set for S (or basis of S).

In this case, S is a subspace of \mathbb{R}^n and $\dim S = m$, the number of vectors in its basis set.

Note The dimension of a subspace is constant but one can choose different bases for the same subspace, all with the same number of vectors, $\dim S$.

(11)

Defn: If A is any real matrix, then its nullspace is defined as $\text{Null}(A) = \{w \mid Aw = 0\}$. Its range space is $\text{Range}(A) = \{w \mid w = Av \text{ for some } v\}$.

Fundamental Theorem of Linear Algebra: $\text{Null}(A) \oplus \text{Range}(A^T) = \mathbb{R}^n$ for $A \in \mathbb{R}^{m \times n}$. Here \oplus denotes the direct sum of two sets ($A \oplus B = \{x+ty \mid x \in A, y \in B\}$).

Fact: If $A \in \mathbb{R}^{n \times n}$ (square) and nonsingular, we have $\text{Null } A = \text{Null } A^T = \{0\}$ and $\text{Range } A = \text{Range } A^T = \mathbb{R}^n$. In this case columns of A and A^T both form a basis for \mathbb{R}^n .

Eigenvalues, Eigenvectors, Singular Value Decomposition

Defn: $\lambda \overset{\text{E}}{\in}$ is an eigenvalue of $A \in \mathbb{R}^{n \times n}$ if $\exists q \neq 0$ $Aq = \lambda q$. Then q is called an eigenvector of A (corresponding to λ).

Fact: If A is nonsingular all of its eigenvalues are nonzero. If A is symmetric, all of its eigenvalues are real-valued ($\lambda \in \mathbb{R}$).

(12)

Fact: If $A > 0$ then $\lambda_A > 0$ (all eigenvalues positive real).

SVD: All matrices $A \in \mathbb{R}^{m \times n}$ (assume $m > n$ without loss of generality - wlog) can be decomposed as a product of three matrices.

$$A = U \begin{bmatrix} S \\ 0 \end{bmatrix} V^T \quad \left(\begin{array}{l} A = U \{S \circ\} V^T \\ \text{if } m \geq n \end{array} \right)$$

where U, V are $m \times m$ and $n \times n$ orthogonal and S is $n \times n$ diagonal.

$$\text{Let } S = \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{bmatrix} \text{ and } U = [u_{:,1} \dots u_{:,n}] \\ V = [v_{:,1} \dots v_{:,n}]$$

then equivalently $A = \sum_{i=1}^n \sigma_i u_{:,i} v_{:,i}^T$.

By convention we sort these as $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$.

The σ_i 's are called the singular values of A .

Defn: $\sigma_1 / \sigma_n = K_{\text{SVD}}(A)$ is a condition number of A .

Note that $K_{\text{SVD}}(A) = K_2(A)$ when A is square and nonsingular.

Fact: If $A \in \mathbb{R}^{n \times n}$ is symmetric, then its SVD is its eigendecomposition: $A = Q \Lambda Q^T = \sum_{i=1}^n \lambda_i q_i q_i^T$

where $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ with $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and $Q = [q_1, \dots, q_n]$ is orthogonal. [If $A \geq 0$, then $\lambda_i \geq 0$ as well.]

(13)

Fact: If $A > 0$ with $\lambda_1 > \lambda_2 > \dots > \lambda_n > 0$

then $\lambda_1(x^T x) \geq x^T A x \geq \lambda_n(x^T x) \quad \forall x \in \mathbb{R}^n$.

Fact: If Q is orthogonal, then $\|Qx\|_2 = \|x\|_2$.

Therefore all singular values of Q are 1.

Determinant and Trace

Defn: $A \in \mathbb{R}^{n \times n}$. $\text{trace}(A) = \sum_{i=1}^n A_{ii}$

Fact: Let $\{\lambda_1, \dots, \lambda_n\}$ be the eigenvalues of A .

$\text{trace}(A) = \sum_{i=1}^n \lambda_i$ Trace is the sum of e-values

~~Det~~ $\det A = \prod_{i=1}^n \lambda_i$ Determinant is the product of e-values.

Facts: $\det A = 0 \Leftrightarrow A$ is singular

$$\det AB = \det A \cdot \det B$$

$$\det A^{-1} = 1/\det A, \quad \det A^T = \det A$$

Fact: Let Q be orthogonal so that $Q^{-1} = Q^T$.

$$\text{Then } \det Q^{-1} = \frac{1}{\det Q} = \det Q^T = \det Q$$

$$\therefore \det Q = \pm 1.$$

Matrix factorizations

We have seen the SVD above. There are other useful factorizations for analysis and algorithm design.

Defn: Permutation matrices are orthogonal matrices with all zeros except a single 1 at each row/column.

$$\text{Ex } P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Fact PA and AP operators permute the rows and columns of A, respectively.

LU factorization: $PA = LU$

where P is an $n \times n$ permutation matrix

L is unit lower triangular (diagonals = 1)

U is upper triangular

Application: Solve for $Ax = b$ efficiently.

$$1) \tilde{b} = Pb \text{ so that } PAx = Pb = \tilde{b} \quad (\underline{L} \underline{U} \underline{x} = \tilde{b})$$

2) Solve $Lz = \tilde{b}$ using triangular forward substitution.

3) Solve $Ux = z$ by triangular back-substitution.

See Algorithm A.1 Gaussian Elimination with Row Partial Pivoting

Cholesky Factorization: If $A > 0$ (symmetric, pos. definite).

then it is possible to achieve $A = LL^T$

where L is a lower triangular matrix with diagonal entries uniquely determined by this equation.

(see Alg. A2)

Symmetric permutation can be used to improve the sparsity of L : $P^TAP = \tilde{L}\tilde{L}^T$.

QR Decomposition: $A \in \mathbb{R}^{m \times n}$, $AP = QR$

where P is $n \times n$ permutation

Q is $m \times m$ orthogonal

R is $m \times n$ upper triangular

$$\text{Fact: } \|A\|_2 = \|QR P^T\|_2 \leq \|Q\|_2 \|R\|_2 \|P^T\|_2 = \|R\|_2$$

$$\text{Also } \|R\|_2 = \|Q^T AP\|_2 \leq \|Q^T\|_2 \|A\|_2 \|P\|_2 = \|A\|_2$$

Therefore $\|A\|_2 = \|R\|_2$ so the condition number of ~~A~~ R can be calculated using R , which is upper triangular.

Symmetric Indefinite Factorization: If A is symmetric but indefinite (mixed-sign eigenvalues), Cholesky factorization will not be available since Alg. A2 will try to take the square root of a negative number.

(16)

In this case we can achieve

$$PAP^T = LBL^T$$

where L is unit lower triangular, B is block diagonal with blocks of dimensions 1 or 2, and P is a permutation matrix.

Sherman-Morrison-Woodbury Formula

Also known as the matrix inversion lemma --

Special case : $\bar{A} = A + ab^T \Leftrightarrow \bar{A}^{-1} = A^{-1} - \frac{A^{-1}ab^TA^{-1}}{1 + b^TA^{-1}a}$

For higher-rank updates consider $U, V \in \mathbb{R}^{n \times p}$ $1 \leq p \leq n$.

$$\hat{A} = A + UV^T \Leftrightarrow \hat{A}^{-1} = A^{-1}U(I + V^TA^{-1}U)^{-1}V^TA^{-1}$$

iff $(I + V^TA^{-1}U)$ is nonsingular.

Interlacing Eigenvalue Theorem

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and let $z \in \mathbb{R}^n$ be a vector with $\|z\|=1$ and $\alpha \in \mathbb{R}$ be a scalar. Then if we denote the eigenvalues of $A + \alpha z z^T$ by $\xi_1 \geq \xi_2 \geq \dots \geq \xi_n$ for $\alpha > 0$,

we have that $\xi_1 \geq \lambda_1 \geq \xi_2 \geq \lambda_2 \geq \xi_3 \geq \dots \geq \xi_n \geq \lambda_n$ \square

with $\sum_{i=1}^n (\xi_i - \lambda_i) = \alpha$

If $\alpha < 0$, we have $\lambda_1 \geq \xi_1 \geq \lambda_2 \geq \xi_2 \geq \dots \geq \lambda_n \geq \xi_n$.

* Read the book (pp 614-617) for error analysis, conditioning and stability.

(A7)

Elements of Analysis, Geometry, Topology

Sequences Suppose that $\{x_k\}$ is a sequence of points in \mathbb{R}^n . we say that $\{x_k\}$ converges to x , $\lim_{k \rightarrow \infty} x_k = x$ if $\forall \epsilon > 0, \exists K \in \mathbb{N} \text{ s.t. } \|x_k - x\| \leq \epsilon \quad \forall k \geq K$

Ex Let $x_k = \begin{bmatrix} 1 - 2^{-k} \\ 1/k^2 \end{bmatrix}, \lim_{k \rightarrow \infty} x_k = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Defn. Given an index set $S \subset \{1, 2, 3, \dots\}$, we can define a subsequence of $\{x_k\}$ corresponding to S by $\{x_{k_i}\}_{i \in S}$.

We say that $\hat{x} \in \mathbb{R}^n$ is an accumulation point or limit point for $\{x_k\}$ if there is an infinite set of indices k_1, k_2, k_3, \dots such that the subsequence $\{x_{k_i}\}_{i=1,2,\dots}$ converges to $\hat{x}; \lim_{i \rightarrow \infty} x_{k_i} = \hat{x}$.

Ex Consider the sequence $\{\frac{1}{3}\}, \{\frac{1}{2}\}, \{\frac{1}{1}\}, \{\frac{1}{4}\}, \{\frac{1}{1}\}, \{\frac{1}{8}\}, \dots$. This sequence has exactly two limit points: $\{\frac{1}{1}\}$ and $\{\frac{0}{0}\}$. A sequence can have infinitely many limit points. For instance every point in the interval $[-1, 1]$ is a limit point for $x_k = \sin k$.

Defn. A sequence x_k converges iff it has exactly one limit point.

Defn A sequence $\{x_k\}$ is said to be a Cauchy sequence if for any $\epsilon > 0$, there exists an integer K such that $\|x_k - x_l\| \leq \epsilon$ for all indices $k, l \geq K$.

Fact. A sequence converges iff it is a Cauchy sequence.

Defn A scalar sequence $\{t_k\}$ where $t_k \in \mathbb{R} \ \forall k$ is said to be bounded above if $\exists u \in \mathbb{R} \ni t_k \leq u \ \forall k$; bounded below if $\exists v \in \mathbb{R} \ni t_k \geq v \ \forall k$. The sequence $\{t_k\}$ is nondecreasing if $t_{k+1} \geq t_k \ \forall k$; nonincreasing if $t_{k+1} \leq t_k \ \forall k$.

Fact. If $\{t_k\}$ is nondecreasing and bounded above, then it converges; $\lim_{k \rightarrow \infty} t_k = t$ for some $t \in \mathbb{R}$. Similarly if $\{t_k\}$ is nonincreasing and bounded below, it converges.

Defn The supremum of $\{t_k\}$ is the smallest real number $u \ni t_k \leq u \ \forall k$; $u = \sup_k t_k$. Similarly, $v = \inf_k t_k$ is the infimum of $\{t_k\}$, and is the largest real number such that $v \leq t_k \ \forall k$.

Defn The sequence of suprema $\{u_i\}$ is $u_i \stackrel{\text{def}}{=} \sup_k \{t_k | k \geq i\}$. Clearly u_i is a nonincreasing sequence.

Fact $\{x_k\}$ converges iff $\{x_k\}$ is Cauchy

Proof: (i) $\{x_k\}$ converges $\Rightarrow \{x_k\}$ is Cauchy

Since $\{x_k\}$ converges, $\exists! x \in \mathbb{R}$ for any $\epsilon > 0 \exists K$ with $\|x_k - x\| \leq \epsilon \forall k \geq K$. Consider

$$\|x_k - x_l\| = \|x_k - x + x - x_l\| \leq \|x_k - x\| + \|x - x_l\| \leq 2\epsilon \quad \forall k, l \geq K.$$

$\therefore \{x_k\}$ is Cauchy.

(ii) $\{x_k\}$ is Cauchy $\Rightarrow \{x_k\}$ converges

* This is true for metric spaces and the proof is similar. Consider a subsequence $\{x_{n_k}\}$ such that

$$\|x_{n_k} - x_m\| \leq \epsilon_k \quad \forall m \geq n_k. \text{ So for } n_e > n_k \text{ we have}$$

$\|x_{n_k} - x_{n_e}\| \leq \epsilon_k$. Suppose that the subsequence $\{x_{n_k}\}$ is selected such that $\epsilon_k < \frac{\epsilon_e}{2}$. Then the two balls

$B_{\epsilon_k}(x_{n_k})$ and $B_{\epsilon_e}(x_{n_e})$ satisfy $B_{\epsilon_e}(x_{n_e}) \subset B_{\epsilon_k}(x_{n_k})$.

The intersection of all these balls is a singleton x , i.e.

$$x = \bigcap_{k=1}^{\infty} B_{\epsilon_k}(x_{n_k})$$

since the balls are nested and shrink; $\epsilon_k \rightarrow 0$.

So x is a candidate for the sequence limit.

Clearly $\lim_{k \rightarrow \infty} \|x_{n_k} - x\| = 0$. Notice that $t_{n_k} \stackrel{\Delta}{=} \|x_{n_k} - x\|$

is a monotonically decreasing/increasing subsequence of $t_k = \|x_k - x\|$. Furthermore t_k is bounded from below by 0 ($t_k \geq 0$). $\therefore t_k$ converges to where t_{n_k} converges; that is $\lim_{k \rightarrow \infty} t_k = \lim_{k \rightarrow \infty} t_{n_k} = 0$. So $\{x_k\}$ converges.

(13)

If $\{a_i\}$ is also bounded below, then it converges to $\bar{a} \in \mathbb{R}$, which is called the liminf of $\{t_{k_i}\}$; denoted by $\bar{a} = \liminf t_k$.

Defn. Similarly $v_i = \inf_k \{t_k | k \geq i\}$, which is nondecreasing.

If $\{v_i\}$ is bounded above, it converges to $\bar{v} \in \mathbb{R}$, which is called $\bar{v} = \limsup t_k$.

Ex Consider the sequence $1, \frac{1}{2}, 1, \frac{1}{3}, 1, \frac{1}{4}, \dots$. Thus sequence has $\liminf = 0$ and $\limsup = 1$.

Note. Basically, the sequence "oscillates" between its liminf and limsup as $k \rightarrow \infty$.

Question: Is the following statement true?

$\{t_k\}$ converges iff $\limsup t_k = \liminf t_k$.

Rates of Convergence Let $\{x_k\}$ be a sequence in \mathbb{R}^n that converges to x^* .

Defn we say that the convergence is α -linear if $\exists r \in \mathbb{R}_{>0}$ such that $\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq r \quad \forall k \geq K$ for some K .

Ex $t_k = 1 + 0.5^k$ converges to 1 α -linearly with $r=0.5$ ($\alpha = \text{quotient}$)

Defn The convergence is Q-superlinear if

$$\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$$

Ex $\lim_{k \rightarrow \infty} (1 + k^{-k}) = 1$ converges superlinearly to 1.

Proof ~~Not yet~~

$$\lim_{k \rightarrow \infty} \frac{|1 + (k+1)^{-k+1} - 1|}{|1 + k^{-k} - 1|} = \lim_{k \rightarrow \infty} \frac{(k+1)^{-k+1}}{k^{-k}} = \frac{1}{(k+1)} \left(\frac{k+1}{k}\right)^{-k} = 0$$

Defn Convergence is Q-quadratic if $\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} \leq M \quad \forall k \geq K$.

Here M does not need to be less than one.

Ex $1 + (0.5)^{2^k}$ converges Q-quadratically.

Defn In general, the Q-order of convergence is p if

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} \leq M \quad \forall k \geq K.$$

Clearly higher order convergence implies lower order ones.

Ex Q-quadratic \Rightarrow Q-superlinear \Rightarrow Q-linear

We will typically omit the "Q-" and simply refer to these as quadratic convergence, superlinear convergence etc.

Defn A weaker form of convergence is as follows.

If $\exists v_k \geq 0 \quad \forall k$ such that $\|x_k - x^*\| \leq v_k \quad \forall k$ and $\lim_{k \rightarrow \infty} v_k = 0$ then we say $\{\|x_k - x^*\|\}$ is dominated by $\{v_k\}$ and converges Q-linearly.

Then we say $\{\|x_k - x^*\|\}$ is dominated by $\{v_k\}$ and converges Q-linearly.

(21)

Defn Similarly, we say $\{x_k\}$ converges to x^*

R-superlinearly if $\{\|x_k - x^*\|\}$ is dominated by

$\{v_k\}$ that converges to zero Q-superlinearly;

R-quadratically if $\{v_k\}$ converges to zero

Q-quadratically, etc.

Note that R-linear convergence allows the error not to decrease at every iteration, while Q-linear convergence forces the error to decrease at every step.

Topology of the Euclidean Space \mathbb{R}^n

Defn The set F is bounded if $\exists M > 0 \text{ s.t. } \|x\| \leq M \quad \forall x \in F$.

Defn A subset $F \subset \mathbb{R}^n$ is open if $\forall x \in F \exists \epsilon > 0 \text{ s.t. } B_\epsilon(x) \subset F$. $B_\epsilon(x)$ is the ball centered at x with radius ϵ .

$$B_\epsilon(x) \subset F \Leftrightarrow \{y \in \mathbb{R}^n \mid \|y - x\| < \epsilon\} \subset F.$$

Defn The set F is closed if for all possible sequences $\{x_n\}$ in F , all limit points of $\{x_n\}$ are elements of F .

Ex $F = (0, 1) \cup (2, 10)$ is open while $F = [0, 1] \cup \{2, 5\}$ is closed.

(22)

$F = (0, 1]$ is neither open nor closed.

Defn The interior of F is the largest open set contained in F . The closure of F is the smallest closed set containing F . These are denoted by $\text{int } F$ and $\text{cl } F$.

Fact $x \in \text{cl } F$ if $\lim_{k \rightarrow \infty} x_k = x$ for some $\{x_k\}$ in F .

Ex $F = (-1, 1] \cup [2, 4) \Rightarrow \text{cl } F = [-1, 1] \cup [2, 4]$
 $\text{int } F = (-1, 1) \cup (2, 4)$

Note F is open $\Leftrightarrow \text{int } F = F$

F is closed $\Leftrightarrow \text{cl } F = F$

Facts The union of finitely many closed sets is closed.
Any intersection of closed sets is closed.

The intersection of finitely many open sets is open.

Any union of open sets is open.

Ex Consider the closed sets $[2^k, 2^{k+1}]$. The union of infinitely many such closed sets $\bigcup_{k=1}^{\infty} [2^k, 2^{k+1}]$ ends up being open, while any finite union is closed.

(23)

Defn F is compact if every $\{x_k\}$ in F has at least one limit point and all such limit points are in F .

Central Result in Topology $F \subset \mathbb{R}^n$ is closed and bounded
 $\Rightarrow F$ is compact.

Defn Given $x \in \mathbb{R}^n$, we call $N \subset \mathbb{R}^n$ a neighborhood of x if it is an open set containing x . The open ball of radius ϵ around x , denoted by $B_\epsilon(x)$ or $B(x; \epsilon)$, is an especially useful neighborhood.

$$B(x; \epsilon) = \{y \in \mathbb{R}^n \mid \|y - x\| < \epsilon\}$$

Defn Given $F \subset \mathbb{R}^n$, we say N is a neighborhood of F if $\exists \epsilon > 0$ such that $\bigcup_{x \in F} B(x; \epsilon) \subset N$.

Convex Sets in \mathbb{R}^n

Defn. A convex combination of a finite set of vectors

$\{x_1, \dots, x_m\}$ in \mathbb{R}^m is any vector x of the form

$$x = \sum_{i=1}^m \alpha_i x_i \text{ where } \sum_i \alpha_i = 1, \alpha_i \geq 0 \quad \forall i=1, \dots, m.$$

The convex hull of $\{x_1, \dots, x_m\}$ is the set of all convex combinations of these vectors.

Defn. A cone is a set $F \subset \mathbb{R}^n$ for which we have

$$x \in F \Rightarrow \alpha x \in F, \forall \alpha > 0$$

Ex $F = \{(x_1, x_2) \mid x_1 > 0, x_2 \geq 0\} \subset \mathbb{R}^2$ is a cone.

* Note that cones are not necessarily convex.

Ex $F = \{(x_1, x_2) \mid x_1 \geq 0 \text{ OR } x_2 \geq 0\}$ is a cone

but is not convex. So convexity depends on the apex angle.

Ex The cone generated by $\{x_1, x_2, \dots, x_n\}$ is the set of all vectors x of the form $x = \sum_i \alpha_i x_i, \alpha_i \geq 0 \forall i$. Note that all cones of this form are convex. by construction. (Prove this as exercise.).

Defn. An affine set in \mathbb{R}^n is the set of all vectors $\{x\} + S$ where $x \in \mathbb{R}^n$, $S \subset \mathbb{R}^n$ is a subspace.

↳ This extends the span by adding x into the basis of S .

Defn Given $F \subset \mathbb{R}^n$, the affine hull of F ($\text{aff } F$) is the smallest affine set containing F ; equivalently it is the intersection of all affine sets containing F or the set of all affine combinations of the elements of F .

$$\text{aff } F = \left\{ \sum_i \alpha_i x_i \mid x_i \in F, \alpha_i \in \mathbb{R}, \sum_i \alpha_i = 1 \right\}$$

Defn. The relative interior $\text{ri } F$ is its interior relative to $\text{aff } F$. If $x \in F$, then $x \in \text{ri } F$ if $\exists \epsilon > 0$ such that $(x + \epsilon B) \cap \text{aff } F \subset F$.

(25)

Continuity and Limits. Consider $f: D \rightarrow \mathbb{R}^m$, $D \subset \mathbb{R}^n$.

For some $x_0 \in \text{cl } D$ let $\lim_{x \rightarrow x_0} f(x) = f_0$, which means $\forall \epsilon > 0 \exists \delta > 0$ such that $\|x - x_0\| < \delta$ and $x \in D \Rightarrow \|f(x) - f_0\| \leq \epsilon$.

Defn. f is continuous at $x_0 \in D$ and $f(x_0) = f_0$ if $\lim_{x \rightarrow x_0} f(x) = f_0$.

f is continuous on D if f is continuous $\forall x_0 \in D$.

Defn f is Lipschitz continuous on $N \subset D$ if $\exists L > 0$ such that $\|f(x_1) - f(x_0)\| \leq L \|x_1 - x_0\| \quad \forall x_0, x_1 \in N$.

L is called the Lipschitz constant. The function f is locally Lipschitz continuous at $\bar{x} \in \text{int } D$

if $\exists N \subset D$, a neighborhood of \bar{x} with the Lipschitz condition above hold for some $L > 0$.

Fact If g and h are two functions mapping $D \subset \mathbb{R}^n$ to \mathbb{R}^m which are Lipschitz continuous on $N \subset D$ with L_g, L_h , then $g+h$ is Lipschitz continuous on N with $L_g + L_h$. Also gh is Lipschitz continuous on N if g, h are Lipschitz cont. on N and both are bounded on N .

Proof Consider $g h$

$$|g(x_0)h(x_0) - g(x_1)h(x_1)| = |g(x_0)h(x_0) - g(x_1)h(x_0) + g(x_1)h(x_0) - g(x_1)h(x_1)|$$

triangle
inequality

$$\begin{aligned} &\leq |g(x_0)h(x_0) - g(x_1)h(x_0)| + |g(x_1)h(x_0) - g(x_1)h(x_1)| \\ &= |h(x_0)| \cdot |g(x_0) - g(x_1)| + |g(x_1)| \cdot |h(x_0) - h(x_1)| \\ &\leq 2M L \|x_0 - x_1\| \end{aligned}$$

where $M = \max_x (\sup |g(x)|, \sup_x |h(x)|)$, $L = \max(L_g, L_h)$.

Derivatives Let $\phi: \mathbb{R} \rightarrow \mathbb{R}$. Consider $\phi'(\alpha)$

$$\frac{d\phi}{d\alpha} = \phi'(\alpha) \stackrel{\Delta}{=} \lim_{\epsilon \rightarrow 0} \frac{\phi(\alpha + \epsilon) - \phi(\alpha)}{\epsilon}$$

Chain Rule Let $\alpha(\beta)$ be a reparameterization.

Then $\frac{d\phi(\alpha(\beta))}{d\beta} = \frac{d\phi}{d\alpha} \frac{d\alpha}{d\beta} = \phi'(\alpha) \alpha'(\beta)$

Multivariate functions: $f: \mathbb{R}^n \rightarrow \mathbb{R}$

We say that f is differentiable at x if there exists a vector $g \in \mathbb{R}^n$ such that

$$\lim_{y \rightarrow 0} \frac{f(x+y) - f(x) - g^T y}{\|y\|} = 0$$

where $\|\cdot\|$ is any vector norm. This is called Frechet differentiability. If such g exists, it is called the gradient of f at x ; $\nabla f(x) = [\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n}]$

(27)

Here, each partial derivative is obtained by

$$\frac{\partial f(x)}{\partial x_i} = \lim_{\epsilon \rightarrow 0} \frac{f(x + \epsilon e_i) - f(x)}{\epsilon}$$

where $e_i = [0 \dots 0 \underset{i^{\text{th}} \text{ entry}}{\underset{\nwarrow}{1}} 0 \dots 0]^T$ is the i^{th} canonical basis vector.

Hessian Matrix The matrix of second derivatives is called the Hessian of f . Let $\nabla^2 f \triangleq \left[\frac{\partial^2 f}{\partial x_i \partial x_j} \right]$.

Then $\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots \\ \vdots & \ddots \end{bmatrix} = H_f(x)$ Also used sometimes

Defn. $f \in C^0 \Leftrightarrow f$ is continuous

$f \in C^1 \Leftrightarrow f$ and ∇f are continuous
we say f is continuously differentiable.

$f \in C^2 \Leftrightarrow f, \nabla f, \nabla^2 f$ are continuous

we say f is twice continuously differentiable.

Fact If f is twice continuously differentiable at x then $\nabla^2 f(x)$ is a symmetric matrix. The converse is also true.

Jacobian Matrix Consider $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$. The Jacobian

$\nabla_x f(x)$ is an $m \times n$ matrix whose i^{th} row is $\nabla f_i(x)$, that is the gradient of $f_i(x)$, the i^{th} component of f .

$\nabla f(x)$ is also called $J_f(x)$ sometimes.

Chain rule: $\nabla_z f_m^n(x(z)) = \nabla_x f(x) \nabla_z x(z)$

$(m \times n) \quad (m \times n) \cdot (n \times n)$

Directional Derivatives Let $f: \mathbb{R}^n \rightarrow \mathbb{R}$ be a mapping (23) and $p \in \mathbb{R}^n$ a direction vector. ~~of unit length, $\|p\|=1$~~ .

Then $D_p f(x) = \lim_{\epsilon \rightarrow 0} \frac{f(x+\epsilon p) - f(x)}{\epsilon}$

Fact If f is continuously differentiable in a neighborhood of x , we have $D_p f(x) = \nabla f(x) \cdot p$ (Prove this as an exercise).

Mean Value Theorem Given a continuously differentiable function $\phi: \mathbb{R} \rightarrow \mathbb{R}$ and $x_0, x_1 \in \mathbb{R}$ & $x_1 > x_0$, we have

$$\frac{\phi(x_1) - \phi(x_0)}{x_1 - x_0} = \phi'(z) \quad \left| \begin{array}{l} \text{OR} \\ \phi(x_1) = \phi(x_0) + \phi'(z)(x_1 - x_0) \end{array} \right.$$

for some $z \in (x_0, x_1)$.

(Could there be more than one such z ?)

* For $f: \mathbb{R}^n \rightarrow \mathbb{R}$ and any vector p , if f is cont. diff. we have $f(x+p) = f(x) + \nabla f(x+\alpha p) \cdot p$ for some $\alpha \in (0, 1)$

(Prove by defining $\phi(\alpha) = f(x+\alpha p)$, $\alpha_0=0$, $\alpha_1=1$.)

* If f is twice cont. differentiable, then

$$f(x+p) = f(x) + \nabla f(x) \cdot p + \frac{1}{2} p^T \nabla^2 f(x+\alpha p) \cdot p$$

for some $\alpha \in (0, 1)$.

(See pg 630 for an approximate extension to multivariate cases.)

Implicit Function Theorem

Let $h: \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ be a function such that

- i) $h(z^*, 0) = 0$ for some $z^* \in \mathbb{R}^n$
- ii) $h(\cdot, \cdot)$ is cont. diff. in some neighborhood of $(z^*, 0)$
- iii) $\nabla_z h(z, t)$ is nonsingular at $(z, t) = (z^*, 0)$

Then there exist open sets $N_z \subset \mathbb{R}^n$ and $N_t \subset \mathbb{R}^m$ containing z^* and 0 respectively, and a continuous function $z: N_t \rightarrow N_z$ such that $z^* = z(0)$ and $h(z(t), t) = 0$ for all $t \in N_t$.

Further, $z(t)$ is uniquely defined. Finally, if h is p times cont. diff. w.r.t. both its arguments for some $p > 0$, then $z(t)$ is also p times cont. diff. w.r.t. t and

$$\nabla z(t) = -\nabla_t h(z(t), t) [\nabla_z h(z(t), t)]^{-1}$$

for all $t \in N_t$.

Ex Consider a parametrized system of linear equations

$M(t)z = g(t)$ where $M(\cdot) \in \mathbb{R}^{n \times n}$ with $M(0)$ nonsingular, $g(\cdot) \in \mathbb{R}^n$. Define $h(z, t) = M(t)z - g(t)$.

If $M(\cdot)$ and $g(\cdot)$ are cont. diff. in some neighborhood of $t=0$, then the theorem implies that $z(t) = M(t)^{-1}g(t)$ is a cont. function of t in some neighborhood of $t=0$.

Order Notation $\mathcal{O}(\cdot), o(\cdot), \mathcal{R}(\cdot)$

Given two nonnegative infinite sequences $\{\eta_k\}, \{v_k\}$
 we write $\eta_k = \mathcal{O}(v_k)$ if $\exists C > 0 \text{ s.t. } |\eta_k| \leq C|v_k|$
 $\forall k \geq K$ for some K .

we write $\eta_k = o(v_k)$ if $\lim_{k \rightarrow \infty} \frac{\eta_k}{v_k} = 0$.

we write $\eta_k = \mathcal{R}(v_k)$ if $\exists c_0, c_1 \text{ s.t. } 0 < c_0 \leq c_1 < \infty \text{ and}$
 $c_0|v_k| \leq |\eta_k| \leq c_1|v_k|$

The same notation can be used when η & v depend on each other continuously.

$$\eta(v) = \mathcal{O}(v) \Leftrightarrow \exists C \text{ s.t. } |\eta(v)| \leq C|v| \quad \forall v \in \mathbb{R}$$

$$\eta(v) = o(v) \Leftrightarrow \lim_{\substack{v \rightarrow 0 \\ \text{or } v \rightarrow \infty}} \frac{\eta(v)}{v} = 0$$

Also $\eta_k = \mathcal{O}(1) \Leftrightarrow \exists C \text{ s.t. } |\eta_k| \leq C \quad \forall k$

while $\eta_k = o(1) \Leftrightarrow \lim_{k \rightarrow \infty} \eta_k = 0$

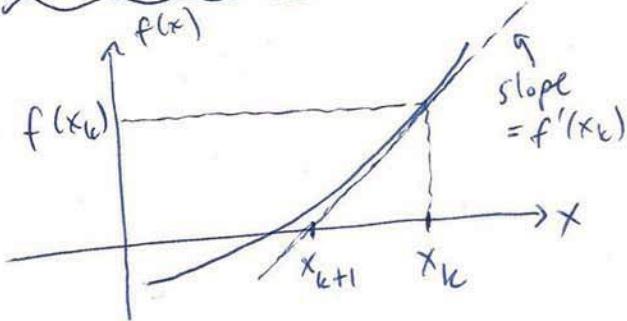
* If, instead of scalar, vectors and matrices are used in the notation, it is meant to apply to their norms.

$$\text{Ex } f(x) = \mathcal{O}(\|x\|) \Leftrightarrow \exists C > 0 \text{ s.t. } \|f(x)\| \leq C\|x\|$$

Root Finding for Scalar Equations

Solve $f(x) = 0$
where $f: \mathbb{R} \rightarrow \mathbb{R}$.

Newton's Method: $x_{k+1} \leftarrow x_k - \frac{f(x_k)}{f'(x_k)}$

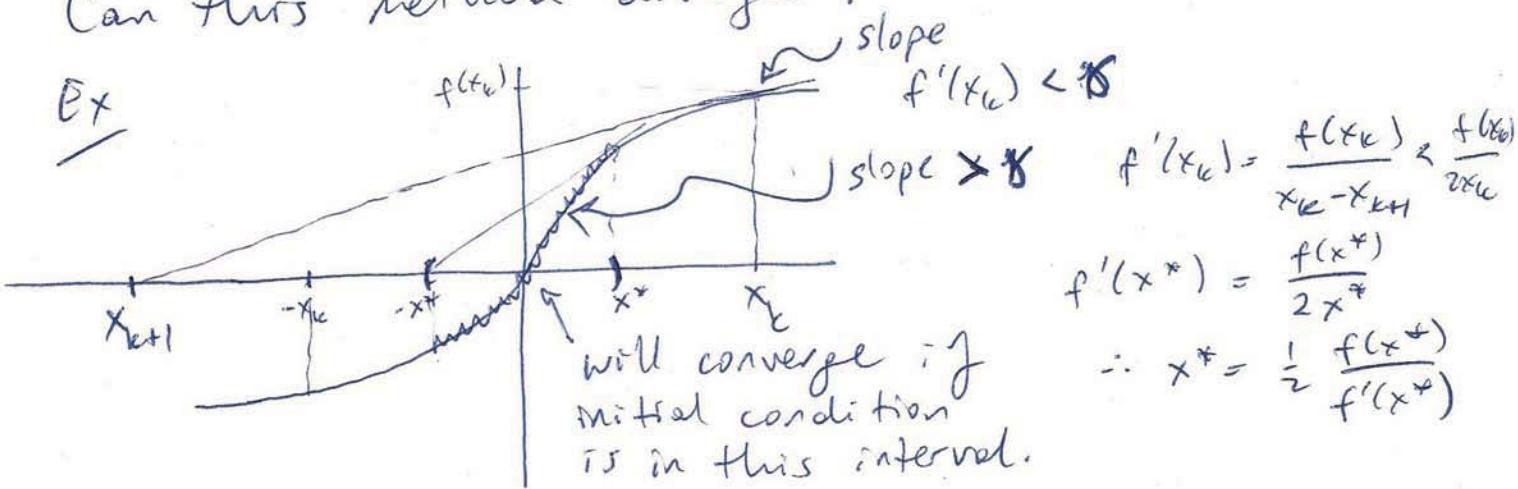


From the figure, we see that

$$\frac{f(x_k)}{x_k - x_{k+1}} = f'(x_k) \quad \dots$$

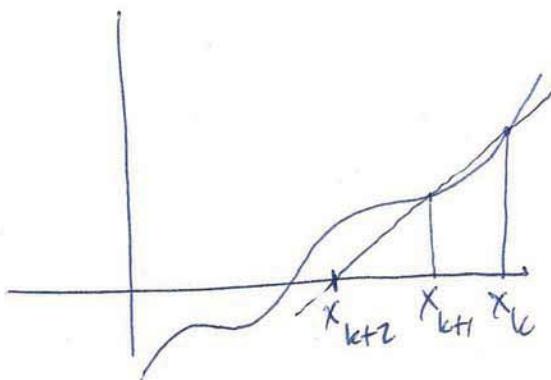
Can this method diverge?

Ex



Secant Method:

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



Note that

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

is used in Newton's method.

Chapter 2 : Fundamentals of Unconstrained Optimization

We want to solve $\min_x f(x)$

where $x \in \mathbb{R}^n$ with $n \geq 1$ and $f: \mathbb{R}^n \rightarrow \mathbb{R}$ which is a smooth function.

Ex Curve fitting: Suppose that we measured sample values $\{y_1, y_2, \dots, y_m\}$ at times $\{t_1, t_2, \dots, t_m\}$ from a signal. It is known a priori that the signal exponential-oscillatory behavior, so we choose the model

$$\phi(t; x) = x_1 + x_2 e^{-(x_3 - t)^2/x_4} + x_5 \cos(x_6 t)$$

We would like to choose x such that the residual errors $r_j(x) = y_j - \phi(t_j; x)$ are small.

In least-squares regression, we solve for

$$\min_x f(x) = \sum_{i=1}^m r_i^2(x)$$

Since $\phi(t; x)$ is a nonlinear function of x , this is called a nonlinear least-squares problem. If m is large, the evaluation of $f(x)$ is computationally expensive.

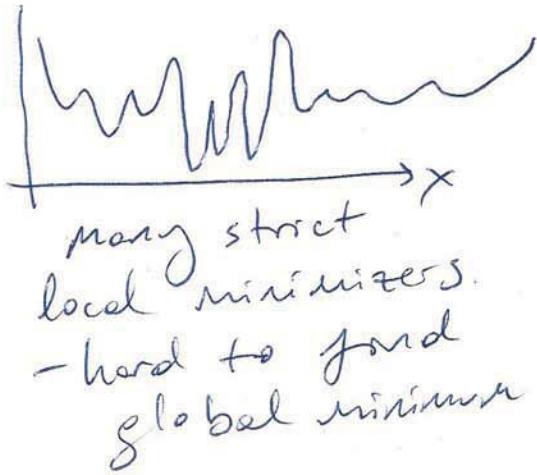
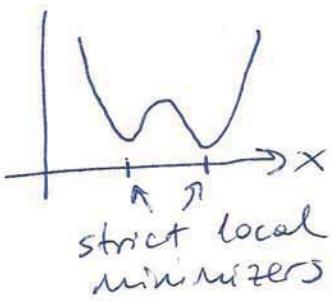
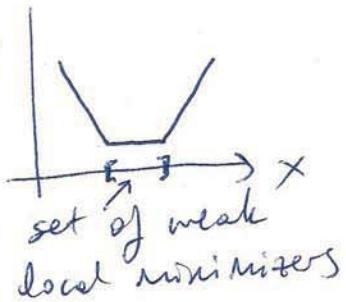
What is a solution?

Defn. A point x^* is a global minimizer of $f(x)$ iff $f(x^*) \leq f(x) \forall x \in \mathbb{R}^n$ (or the feasible set in general)

Since we can never be certain that $f(\cdot)$ makes a sharp dip in some region of \mathbb{R}^n , by sampling $f(\cdot)$ either randomly or systematically, we can never be certain if f ,^{or not} some point with a smaller objective value than what we have encountered so far.

Defn. A point x^* is a local minimizer of $f(x)$ iff \exists some neighborhood N of x^* such that $f(x^*) \leq f(x) \forall x \in N$.

(x^* is a strict local minimizer iff $f(x^*) < f(x) \forall x \neq x^*$)



Ex $f(x) = -\cos x^2$, $f(0)=0$ has a sequence of strict local minimizers at $x=0$ and its vicinity in the form $x_e = \frac{1}{\pi e}$; so $\lim_{e \rightarrow \infty} x_e = 0$.

Fact All isolated local minimizers are strict.
All strict local minimizers are NOT isolated.

* Note that while I personally like the gradient vector to be defined as a row to be consistent with the Jacobian matrix, I will stick to the column vector definition/notation of the book.

Consider $f(x)$ and its gradient $\nabla f(x)$ and Hessian $\nabla^2 f(x)$. The following theorems tell us the conditions for recognizing a local minimum.

Theorem 2.1 Taylor's theorem

Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is continuously differentiable and $p \in \mathbb{R}^n$. Then $f(x+p) = f(x) + \nabla f(x+t_p)p$ for some $t \in (0, 1)$. If f is twice continuously differentiable then $\nabla f(x+p) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp)p dt$ and $f(x+p) = f(x) + \nabla f^\top(x)p + \frac{1}{2} p^\top \nabla^2 f(x+tp)p$ for some $t \in (0, 1)$.

Theorem 2.2 (First-order necessary conditions)

If x^* is a local minimizer and f is cont. diff. in an open neighborhood of x^* , then $\nabla f(x^*) = 0$.

Proof: (By contradiction)

Assume that $\nabla f(x^*) \neq 0$. Let $p = -\nabla f(x^*)$ and note that $p^T \nabla f(x^*) = -\|\nabla f(x^*)\|^2 < 0$. Since ∇f is continuous near x^* , $\exists T > 0$ s.t.

$$p^T \nabla f(x^* + tp) < 0 \quad \forall t \in [0, T]$$

For any $\bar{t} \in (0, T]$, from Taylor's theorem:

$$f(x^* + \bar{t}p) = f(x^*) + \bar{t} p^T \nabla f(x^* + tp) \quad \text{for some } t \in (0, \bar{t}).$$

Therefore, $f(x^* + \bar{t}p) < f(x^*) + \bar{t} \in (0, T]$. Since we identified a direction going from x^* to $x^* + Tp$ for which f decreases, x^* is not a local minimizer and we have a contradiction \square .

Defn: We call x^* a stationary point of f if $\nabla f(x^*) = 0$. Any local minimizer must be a stationary point.

Theorem 2.3 (Second-order necessary conditions) (36)

If x^* is a local minimizer of f and $\nabla^2 f$ exists and is continuous in an open neighborhood of x^* , then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \geq 0$.

Proof: We know from Theorem 2.2 that $\nabla f(x^*) = 0$.

For contradiction assume that $\nabla^2 f(x^*)$ is not positive semidefinite. Let p be a vector such that $p^T \nabla^2 f(x^*) p < 0$. Since $\nabla^2 f$ is continuous near x^* , $\exists T > 0 \ni p^T \nabla^2 f(x^* + tp) p < 0 \quad \forall t \in [0, T]$. Using the Taylor expansion around x^* , we have $\forall \bar{t} \in (0, T]$ and for some $t \in (0, \bar{t})$

$$f(x^* + \bar{t}p) = f(x^*) + \bar{t} p^T \cancel{\nabla f(x^*)} + \frac{1}{2} \bar{t}^2 p^T \nabla^2 f(x^* + tp) p$$

$$= f(x^*) + \frac{1}{2} \bar{t}^2 \underbrace{p^T \nabla^2 f(x^* + tp) p}_{< 0} \cancel{< f(x^*)}$$

Since we found a direction where f decreases as we go away from x^* , this point is not a local minimum, so there is a contradiction. \square

Theorem 2.4 (Second-order sufficient conditions)

Suppose that $\nabla^2 f$ is continuous in an open neighborhood of x^* and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$. Then x^* is a strict local minimizer of f .

Proof: Since the Hessian is continuous and positive definite $\forall x \in D = \{z \mid \|z - x^*\| < r\}$.

For any nonzero vector $p \in D$, we have $x^* + p \in D$ and

$$\begin{aligned} f(x^* + p) &= f(x^*) + p^T \cancel{\nabla f(x^*)} + \frac{1}{2} p^T \nabla^2 f(z) p \\ &= f(x^*) + \frac{1}{2} p^T \nabla^2 f(z) p \end{aligned}$$

where $z = x^* + tp$ for some $t \in (0, 1)$. Since $z \in D$ we have $p^T \nabla^2 f(z)p > 0$ and therefore $f(x^* + p) > f(x^*)$. \square

Ex $f(x) = x^4 \Rightarrow \nabla f(x) = 4x^3$ and $\nabla^2 f(x) = 12x^2$

At $x=0$, $\nabla f(0) = 0$ and $\nabla^2 f(0) = 0$.

So $x=0$ satisfies the necessary conditions but not the sufficient ones.

Ex $f(x) = x^T A x$ where $A \geq 0$ with $Ap = 0$ (i.e. p is an eigenvector with zero-eigenvalue). Then $f(x_p) = 0$, so the line along p , passing through the origin is a local min. set.

Thm 2.5: When f is convex, any local minimizer x^* is a global minimizer of f . If in addition f is differentiable, then any stationary point x^* is a global minimizer of f .

Proof: (i) Suppose that x^* is a local but not global minimizer of a convex f . Then $\exists z \in \mathbb{R}^n$ s.t. $f(z) < f(x^*)$. Consider $x = \lambda z + (1-\lambda)x^*$ for some $\lambda \in (0, 1)$. By convexity of f , we have

$$f(x) \leq \lambda f(z) + (1-\lambda)f(x^*) < f(x^*)$$

Any neighborhood of x^* contains a piece of the line segment above so $\exists x \in N_{x^*}$ for which $f(x) < f(x^*)$ hence x^* is not a local minimizer.

(ii) Suppose a stationary point x^* is not a global minimizer of convex, differentiable f . Then $\exists z \in \mathbb{R}^n$ s.t. $f(z) < f(x^*)$. From convexity,

$$\begin{aligned} \nabla f^T(x^*) (z - x^*) &= \frac{d}{d\lambda} f(x^* + \lambda(z - x^*))|_{\lambda=0} \\ &= \lim_{\lambda \downarrow 0} \frac{f(x^* + \lambda(z - x^*)) - f(x^*)}{\lambda}|_{\lambda=0} \\ &\leq \lim_{\lambda \downarrow 0} \frac{\lambda f(z) + (1-\lambda)f(x^*) - f(x^*)}{\lambda}|_{\lambda=0} \\ &= f(z) - f(x^*) < 0 \end{aligned}$$

Therefore $\nabla f(x^*) \neq 0$, x^* is not stationary. \square

Overview of Algorithms

Given an initial estimate x_0 , the algorithms generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$ that terminate when no progress is made or when the solution is approximated with some accuracy.

Most commonly, we will require that $f(x_{k+1}) < f(x_k)$.

Line Search strategy: The algorithm chooses a direction p_k and searches along this direction from x_k . Ideally, one solves the following problem

$$\min_{\alpha \geq 0} f(x_k + \alpha p_k)$$

If one solves this problem accurately, the algorithm gets maximum benefit from this direction. However exact minimization may be impossible or very expensive. An approximate α that reasonably approximates the optimal value (ideal step size) can be found with little cost/effort.

Trust Region Strategy: The information gathered about f is used to construct a model function m_k

whose behavior near x_k is similar to that of f . Since m_k may not be a good approximation of f far from x_k , we solve for

$$\min_p m_k(x_k + p) \text{ where } (x_k + p) \in \text{Trust region}$$

If $f(x_k + p) \geq f(x_k)$ or if $f(x_k + p) < f(x_k)$ but not sufficiently smaller, then we conclude that the trust region is too large. In this case, we shrink the trust region and re-solve.

Usually, the trust region is $TR = \{x \mid \|x - x_k\| \leq \Delta\}$ so $\|p\| < \Delta$ where $\Delta > 0$ is the trust region radius.

$\|\cdot\|$ here could be $\|\cdot\|_\infty$, $\|\cdot\|_2$, $\|\cdot\|_{\text{Mahalanobis}}$, etc.

Most conveniently, $m_k(x_k + p) = f_k + p^T \nabla f_k + \frac{1}{2} p^T B_k p$ where f_k , ∇f_k , and B_k are the function, its gradient, and its Hessian at x_k (or some approximation of these).

Search Directions for Line Search Methods: The steepest descent direction $-\nabla f(x_k)$ is the most obvious choice for search direction. Consider

$$f(x_k + \alpha p) = f(x_k) + \alpha p^T \nabla f_k + \frac{1}{2} \alpha^2 p^T \nabla^2 f(x_k + t p) p$$

for some $t \in (0, \alpha)$

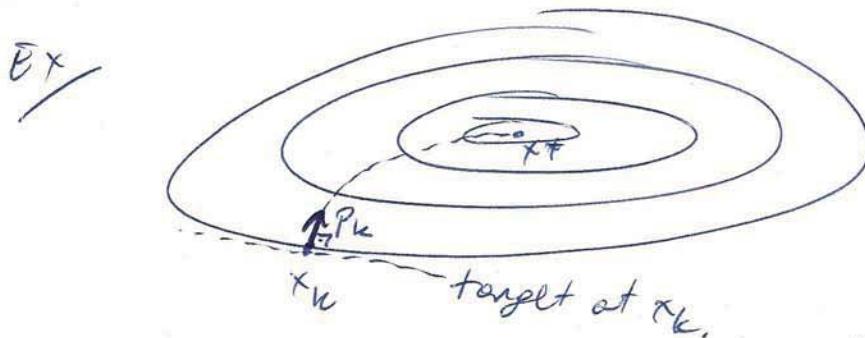
The unit direction of most rapid decrease^{in f} is given by

$$\min_p p^T \nabla f_k \text{ subject to } \|p\|=1$$

Since $p^T \nabla f_k = \|p\| \cdot \|\nabla f_k\| \cos \theta = \|\nabla f_k\| \cos \theta$, where θ is the angle between p and ∇f_k , clearly, the minimizer is attained ~~is~~ when $\cos \theta = -1$, so

$$p = -\frac{\nabla f_k}{\|\nabla f_k\|}$$

The steepest descent method uses update directions that are orthogonal to the equilevel contours of f .



fact Steepest descent requires only the calculation of the gradient and α_k can be chosen to guarantee that $f(x_{k+1}) < f(x_k)$.

$$\text{Proof: } f(x_k + \epsilon p_k) = f(x_k) + \epsilon p_k^T \nabla f_k + O(\epsilon^2)$$

where $p_k = -\nabla f_k$. Since x_k is not a local minimizer, then $\exists N_{x_k}^{(if)}$, some neighborhood where $x_k + \epsilon p_k \in N_{x_k}^{(if)}$ and $f(x_k + \epsilon p_k) < f(x_k)$ for sufficiently small ϵ .

In general, one could use any update direction P_k as long as $P_k^T \nabla f_k < 0$ (i.e. update is towards the inside of equilevel contours locally).

An important choice for P_k is the Newton direction : $P_k = -(\nabla^2 f_k)^{-1} \nabla f_k$. It is derived from the ~~a.~~ 2nd-order Taylor series. Clearly, this direction is accurate when the 2nd-order Taylor series approximation of the function is relatively close to f .

Notice that $P_k^T \nabla f_k = -\nabla f_k^T (\nabla^2 f_k)^{-1} \nabla f_k < 0$
~~if $\nabla^2 f_k \neq 0$~~ if $\nabla^2 f_k > 0$ so in the ~~convex~~ concave region around the local minimizer, this can be safely used.

The convergence rate of Newton updates is typically quadratic. The requirement for inverse-Hessian is a drawback and quasi-Newton algorithms use computationally cheaper approximations.

Recall that (from Taylor theorem by adding/subtracting $\nabla^2 f(x)\rho$)

$$\nabla f(x+\rho) = \nabla f(x) + \nabla^2 f(x)\rho + \int_0^1 [\nabla^2 f(x+t\rho) - \nabla^2 f(x)]\rho dt$$

Since $\nabla f(\cdot)$ is continuous, the integral is $\mathcal{O}(\|p\|)$ and letting $x = x_k$ and $p = x_{k+1} - x_k$, we get

$$\nabla f_{k+1} = \nabla f_k + \nabla^2 f_k(x_{k+1} - x_k) + \mathcal{O}(\|x_{k+1} - x_k\|).$$

When x_k and x_{k+1} are in a region near x^* where $\nabla^2 f \geq 0$, the $\nabla^2 f_k(x_{k+1} - x_k)$ dominates this expression and

$$\nabla^2 f_k(x_{k+1} - x_k) \approx \nabla f_{k+1} - \nabla f_k$$

We can then choose a Hessian approximation B_{k+1} that mimics this property by requiring the so-called secant equation:

$$B_{k+1} s_k = y_k$$

where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$

Additionally, we impose that B_{k+1} is symmetric and that $B_{k+1} - B_k$ is low-rank.

* Symmetric rank-one (SR1):

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

* BFGS (Broyden, Fletcher, Goldfarb, Shanno)

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

(This one is a rank-2 update).

$B_k > 0$ and $s_k^T y_k > 0$
$B_k > 0$

In quasi-Newton methods, $P_k = -B_k^{-1} \nabla f_k$.

If we update the inverse of B_k directly (let $H_k \stackrel{\Delta}{=} B_k^{-1}$):

$$H_{k+1} = (I - p_k s_k y_k^T) H_k (I - p_k y_k s_k^T) + p_k s_k s_k^T$$

$$\text{where } p_k = \frac{1}{y_k^T s_k}$$

then $P_k = -H_k \nabla f_k$.

* Conjugate gradient methods use $P_k = -\nabla f_k + B_k P_{k-1}$ where P_k and P_{k-1} are conjugate (discussed in Chapter 5). Conjugate vectors are orthogonal to each other in a Mahalanobis inner product.

Models for Trust Region Methods:

If we use a linear local model ($B_k = 0$) and the Euclidean norm to define the trust region:

$$\min_P f_k + p^T \nabla f_k \text{ s.t. } \|p\|_2 \leq \Delta_k$$

The solution in closed form is: $P_k = -\frac{\Delta_k \nabla f_k}{\|\nabla f_k\|}$

This is steepest descent with step size Δ_k .

Another choice for B_k is the exact Hessian $\nabla^2 f_k$. This is called the trust region Newton method.

(45)

Scaling The performance of an algorithm may depend on how the problem is formulated. A problem is poorly scaled if changes in x in certain directions are much larger than others.

Ex $f(x) = 10^9 x_1^2 + x_2^2$ is $10^{9/2}$ times more sensitive to perturbations in x_1 than in x_2 .

Defining $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 10^{9/2} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ yields

$f(z) = z_1^2 + z_2^2$, which is better scaled.

* Steepest descent is extremely sensitive to scaling. In poorly scaled problems, its convergence speed drops drastically.

* Newton's method is unaffected by scaling since the Hessian-inverse takes this into account.

Exercises

$$2.1) f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

Compute $\nabla f(x)$ and $\nabla^2 f(x)$.

$$\nabla f(x) = \begin{bmatrix} 200(x_2 - x_1^2)(-2x_1) + 2(1 - x_1)(-1) \\ 200(x_2 - x_1^2) \end{bmatrix} = \begin{bmatrix} -400x_1(x_2 - x_1^2) - 2(1 - x_1) \\ 200(x_2 - x_1^2) \end{bmatrix}$$

$$\nabla^2 f(x) = \begin{bmatrix} -400x_2 + 800x_1^2 + 2 & -400x_1 \\ -400x_1 & 200 \end{bmatrix} = \begin{bmatrix} (1200x_1^2 - 400x_2 + 2) & -400x_1 \\ -400x_1 & 200 \end{bmatrix}$$

(46)

At x^* we have $\nabla f(x^*) = 0$

$$\begin{aligned} -400x_1(x_2-x_1^2) - 2(-x_1) &= 0 \Rightarrow -2(-x_1) = 0 \Rightarrow x_1 = 1 \\ 200(x_2-x_1^2) &= 0 \Rightarrow \boxed{x_2 = x_1^2} \quad \Downarrow \quad x_2 = 1 \end{aligned}$$

Only at $x^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ we have $\nabla f(\cdot; \cdot) = 0$.

At this point $\nabla^2 f(\cdot; \cdot) = \begin{bmatrix} (1200-400+2) & -400 \\ -400 & 200 \end{bmatrix} = \begin{bmatrix} 802 & -400 \\ -400 & 200 \end{bmatrix}$

$$\lambda I - \begin{bmatrix} 802 & -400 \\ -400 & 200 \end{bmatrix} = \begin{bmatrix} (\lambda - 802) & 400 \\ 400 & (\lambda - 200) \end{bmatrix}$$

$$\begin{aligned} \det(\lambda I - \nabla^2 f(x^*)) &= (\lambda - 802)(\lambda - 200) - 400^2 \\ &= \lambda^2 - 1002\lambda + 200 \cdot 802 - 400 \cdot 400 \\ &= \lambda^2 - 1002\lambda + 400 \end{aligned}$$

$$\lambda = \frac{-(-1002)}{2} \pm \frac{1}{2}\sqrt{1002^2 - 4 \cdot 400} = 501 \pm \frac{1}{2}\sqrt{(501^2 - 400)}$$

so both λ are > 0 , hence $\nabla^2 f(x^*) > 0$.

2.3) $f_1(x) = a^T x$, $f_2(x) = x^T A x$ Ansymmetric.

$$\nabla f_1(x) = a \qquad \nabla f_2(x) = 2Ax$$

(I would write $\nabla f_1(x) = a^T$) (I would write $2x^T A$)

$$\nabla^2 f_1(x) = 0 \qquad \nabla^2 f_2(x) = 2A$$

2.7) $f(x) = x^T Q x$ where $Q \geq 0$ is symmetric. Show that f is convex.

$f(y + \alpha(x-y)) = f(\alpha x + (1-\alpha)y)$. Consider $\alpha \in [0, 1]$.

$$\begin{aligned} f(\alpha x + (1-\alpha)y) &= (\alpha x + (1-\alpha)y)^T Q (\alpha x + (1-\alpha)y) = (\alpha x + (1-\alpha)y)^T Q (\alpha x + (1-\alpha)y) \\ &= \cancel{\alpha^2 x^T Q x} + \cancel{(1-\alpha)^2 y^T Q y} + 2\alpha(1-\alpha)x^T Q y \cancel{+ 2\alpha(1-\alpha)x^T Q y} \end{aligned}$$

(47)

Show $f(\alpha x + (1-\alpha)y) - \alpha f(x) - (1-\alpha)f(y) \leq 0$ for $\alpha \in [0,1]$.
 $\forall x, y \in \mathbb{R}^n$.

$$\begin{aligned}
 & \alpha^2 x^T Q x + (1-\alpha)^2 y^T Q y + 2\alpha(1-\alpha)x^T Q y \\
 & - \alpha x^T Q x - (1-\alpha)y^T Q y \stackrel{?}{\leq} 0 \\
 \equiv & -\alpha(1-\alpha)x^T Q x - \alpha(1-\alpha)y^T Q y + 2\alpha(1-\alpha)x^T Q y \stackrel{?}{\leq} 0 \\
 \equiv & -\alpha(1-\alpha)[x^T Q x + y^T Q y - 2x^T Q y] \stackrel{?}{\leq} 0 \quad \text{yes for } \alpha=0 \\
 \equiv & x^T Q x + y^T Q y - 2x^T Q y \stackrel{?}{\geq} 0 \quad \text{for } \alpha \in (0,1) \quad \text{since } \alpha(1-\alpha) > 0 \\
 \equiv & (x-y)^T Q(x-y) \stackrel{?}{\geq} 0 \quad \text{yes since } Q \geq 0.
 \end{aligned}$$

2-8) Show that if f is convex then its global minimizers form a convex set.

Let a and ~~b~~ b be any two global minimizers of f . This means $f(a) = f(b)$ and $\forall x \in \mathbb{R}^n, f(x) \geq f(a)$. Notice that for $x = \alpha a + (1-\alpha)b$ for $\alpha \in [0,1]$,

$$f(x) = f(\alpha a + (1-\alpha)b) \leq \alpha f(a) + (1-\alpha)f(b) = f(a) \quad \forall \alpha \in [0,1].$$

Then any point on the line segment connecting a to b is also a global minimizer of f . \square

Alternative proof by contradiction: Let a and b be two global minimizers of some convex f . Assume that the set of global minimizers is not convex. Then there must exist some a, b such that $\exists x = \alpha a + (1-\alpha)b$ where $f(x) > f(a)$. But $\forall f(x) = f(\alpha a + (1-\alpha)b) > f(a) = \alpha f(a) + (1-\alpha)f(b)$ that means

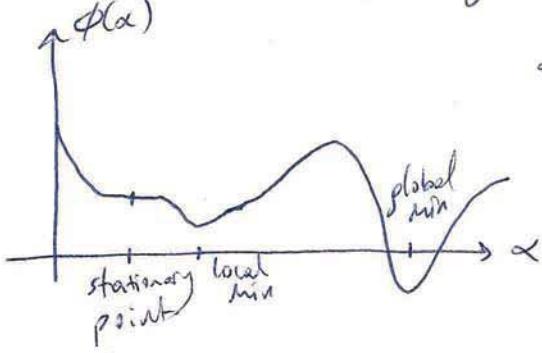
Then f is not convex; this is a contradiction. \square

Chapter 3: Line Search Methods

In iterative optimization, once an update direction p_k^* is selected for $x_{k+1} = x_k + \alpha_k p_k$, a positive scalar α_k , called the step length or step size, needs to be determined for stable and fast convergence. Here, since we deal with problems where we want to minimize the objective function, it is assumed that p_k is a local step direction: $p_k^T \nabla f_k < 0$. Since the update direction generally has the form $p_k = -B_k^{-1} \nabla f_k$ where B_k is symmetric and nonsingular, we need $p_k^T \nabla f_k = -\nabla f_k^T B_k \nabla f_k < 0$. When $B_k > 0$ this condition is guaranteed.

Step length: Given p_k we want $\min_{\alpha} f(x_k + \alpha p_k)$ subject to $\alpha > 0$.

At iteration k , define $\phi(\alpha) \stackrel{\Delta}{=} f(x_k + \alpha p_k)$.



Seeking the closest stationary point where $\phi(\alpha) < \phi(0)$ might not give the global optimum value for the step length.

(49)

Also in general using a sequence $\{\alpha_k\}$ such that $f(x_k + \alpha_k p_k) < f(x_k) \quad \forall k \in \mathbb{N}$ is not sufficient to achieve convergence to the local minimum.

Consider $\{\alpha_k\}$ such that the convex $f(\cdot)$ shown here, which has a minimum value of -1 is ~~sufficient~~ to be minimized with attempted $f(x_k) = \frac{5}{k}, k=0, 1, \dots$

Although at each iteration f_k decreases, $\lim f_k = 0$ and this is not the minimum value. Here, $\{\alpha_k\}$ is chosen such that the decrease in f at each iteration is insufficient. We will define the concept of sufficient decrease.

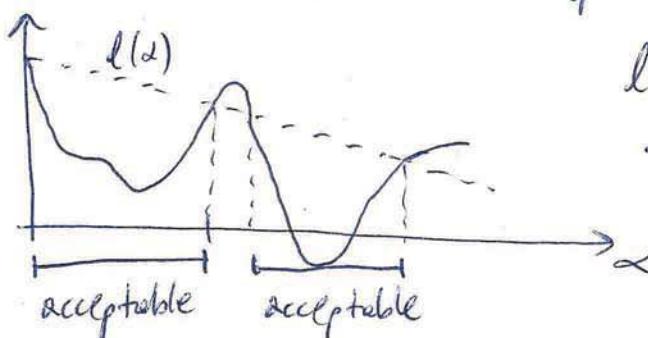
The Wolfe Conditions: A step size should give sufficient decrease in f as measured by the following:

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k$$

for some $c_1 \in (0, 1)$. In words, the reduction in f should be more than a decrease one would obtain proportional to the step length α_k and the directional derivative $p_k^T \nabla f_k$.

$$\phi(\alpha) = f(x_k + \alpha p_k)$$

$$\text{Acceptable } \alpha \iff \phi(\alpha) \leq l(\alpha)$$



$$l(\alpha) = f(x_k) + c_1 \alpha \nabla f_k^T p_k, c_1 \text{ (const)}$$

This is also called the Armijo condition.

(50)

In practice, c_1 is chosen to be very small (e.g. $c_1 = 10^{-4}$).

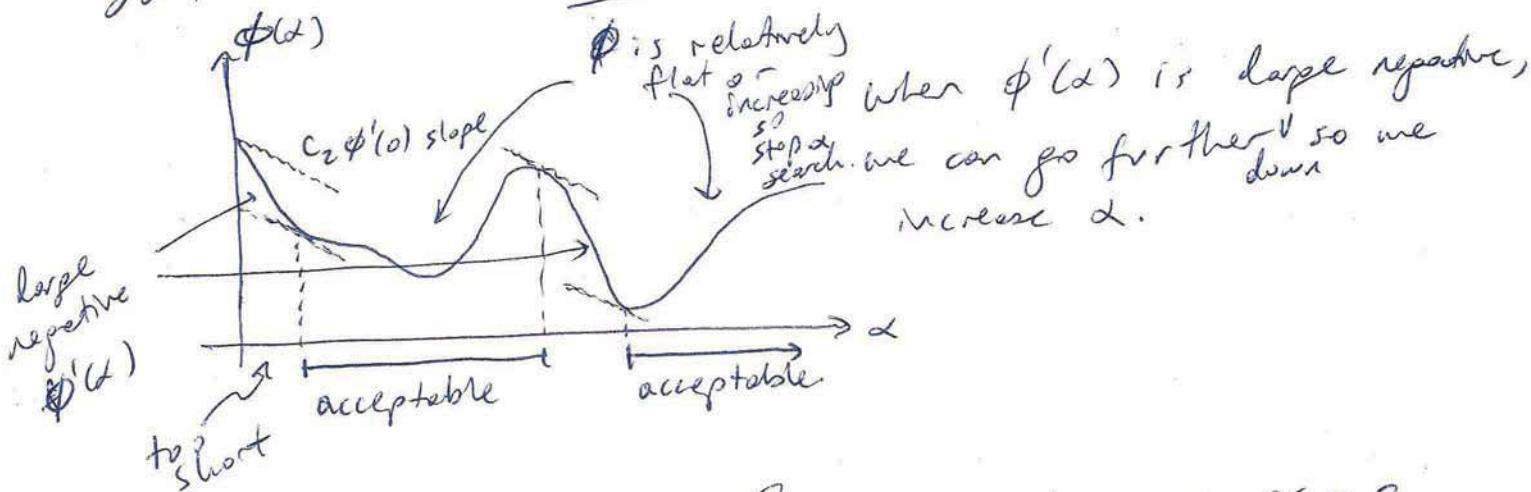
Note: In practice, "very small" depends on the problem and its parameter scales, so this statement and the example of $c_1 = 10^{-4}$ should be interpreted with caution.

In order to rule out unacceptably short steps, we introduce the curvature condition:

$$(\phi'(\alpha_k) =) \quad \nabla^T f(x_k + \alpha_k p_k) p_k \geq c_2 \nabla^T f_k p_k \quad (= c_2 \phi'(0))$$

for some $c_2 \in (c_1, 1)$. Note that $\nabla^T f(x_k + \alpha_k p_k) p_k = \phi'(\alpha_k)$.

So this condition ensures that $\phi'(\alpha_k) \geq c_2 \phi'(0)$.

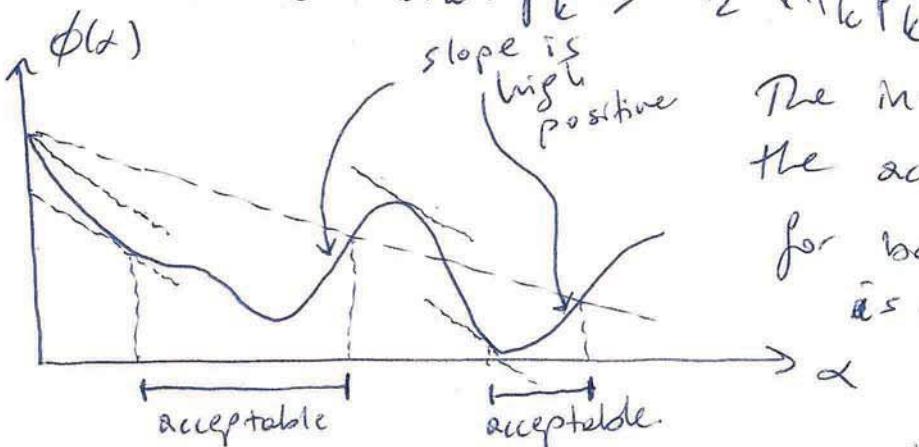


Typically, for Newton & quasi-Newton: $c_2 \approx 0.9$
for Nonlinear conjugate gradient: $c_2 \approx 0.1$.

* Collectively, these are called the Wolfe conditions:

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k$$

$$\nabla^T f(x_k + \alpha_k p_k) p_k \geq c_2 \nabla^T f_k p_k \quad \text{with } 0 < c_1 < c_2 < 1$$



The intersection of the acceptable regions for both conditions is as shown.

Notice that in the acceptable regions, there are places where the slope is large positive - this can be prevented by introducing the strong Wolfe conditions:

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla^T f_k p_k$$

$$|\nabla^T f(x_k + \alpha_k p_k) p_k| \leq c_2 |\nabla^T f_k p_k| \quad \text{with } 0 < c_1 < c_2 < 1$$

This modification disqualifies large positive $\phi'(\alpha)$ so the acceptable α values are closer to stationary points.

Lemma 3.1 Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is cont. diff. Let

p_k be a descent direction at x_k and assume that f is bounded below along $\{x_k + \alpha p_k \mid \alpha \geq 0\}$. Then if $0 < c_1 < c_2 < 1$, 3 intervals of step lengths satisfying the Wolfe conditions and the strong Wolfe conditions.

(i.e. Wolfe conditions do not yield empty set for acceptable α .)

Proof: $\phi(\alpha) = f(x_k + \alpha p_k)$ is bounded below $\forall \alpha > 0$.

Since $0 < c_1 < 1$, the line $l(\alpha) = f(x_k) + \alpha c_1 \nabla f_k^T p_k$ is unbounded below (assuming $\nabla f_k^T p_k \neq 0$); therefore $\phi(\alpha)$ and $l(\alpha)$ must intersect at least once at some $\alpha' > 0$. Consider the smallest such α' .

$$f(x_k + \alpha' p_k) = f(x_k) + \alpha' c_1 \nabla f_k^T p_k$$

Clearly the sufficient decrease condition holds for $0 < \alpha < \alpha'$. By the mean value theorem, $\exists \alpha'' \in (0, \alpha')$

$$\nabla f(x_k + \alpha' p_k) - \nabla f(x_k) = \alpha' \nabla f(x_k + \alpha'' p_k)^T p_k$$

Combining the two equations above, we get

$$\nabla f(x_k + \alpha'' p_k)^T p_k = c_1 \nabla f_k^T p_k > c_2 \nabla f_k^T p_k$$

since $0 < c_1, c_2 < 1$ and $\nabla f_k^T p_k < 0$. Therefore α'' satisfies the Wolfe conditions and the inequalities both hold strictly. Hence, by the assumption that $f \in C^1$, \exists an interval around α'' for which the Wolfe conditions hold.

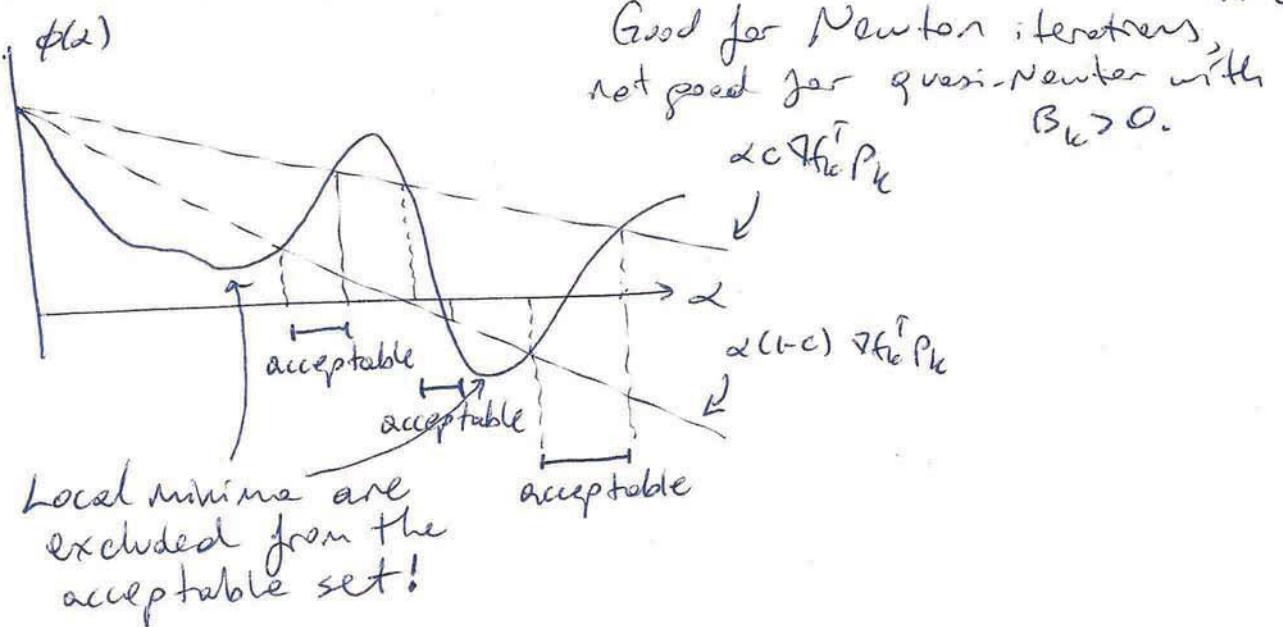
Moreover, since $\nabla f(x_k + \alpha'' p_k)^T p_k < 0$ ~~and~~ ~~strongly~~ $\nabla f_k^T p_k < 0$, the strong Wolfe conditions hold as well, in the same interval around α'' . \square

Note: The Wolfe conditions can be used with any line search algorithm, and are especially useful for Newton-type iterations.

The Goldstein Conditions: These conditions also ensure that the step length α achieves sufficient decrease but is not too short. Goldstein suggests:

$$f(x_k) + (1-c)\alpha_k \nabla f_k^T p_k \leq f(x_k + \alpha_k p_k) \leq f(x_k) + c\alpha_k \nabla f_k^T p_k$$

where $0 < c < 1/2$.



Sufficient Decrease and Backtracking

Using backtracking, one can let go of the lower bound conditions and just use the sufficient decrease rule.

Alg. 3.1: Backtracking Line Search

Choose $\bar{\alpha} > 0$, $\rho \in (0, 1)$, $c \in (0, 1)$.

Set $\alpha \leftarrow \bar{\alpha}$

Repeat until $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f_k^T p_k$

end (repeat)

Set $\alpha_h = \alpha$.

In this algorithm, an acceptable step size will be found after a finite number of iterations since for some ℓ , $\alpha = \rho^{\ell} \bar{\alpha}$ will satisfy the sufficient decrease condition.

The contraction factor ρ can be varied at each iteration of the line search as long as for some preset $\rho_{\text{low}}^{(t+1)}$ and $\rho_{\text{high}}^{(t+1)}$, one satisfies $\rho \in [\rho_{\text{low}}, \rho_{\text{high}}]$ where $\rho_{\text{low}} < \rho_{\text{high}}$.

This method works well with Newton method when $\bar{\alpha} = 1$ (typically), but is not suited for quasi-Newton and gradient methods.

Convergence of line search methods

Let θ_k be the angle between p_k and $-\nabla f_k$:

$$\cos \theta_k = \frac{-\nabla f_k^T p_k}{\|\nabla f_k\| \cdot \|p_k\|}$$

The following theorem has important consequences for the global convergence of steepest descent and other algorithms.

Thm 3.2 (Zoutendijk's Theorem)

Consider any iteration of the form $x_{k+1} = x_k + \alpha_k p_k$, where p_k is a descent direction ($\nabla f_k^T p_k < 0$) and α_k satisfies the Wolfe conditions. Suppose that f is bounded below in \mathbb{R}^n

(55)

and that f is continuously differentiable in an open set N containing the level set $L \triangleq \{x | f(x) \leq f(x_0)\}$, where x_0 is the starting point of the iteration.

Assume also that the gradient ∇f is Lipschitz. cont. on N , that is, $\exists L > 0$ such that

$$\|\nabla f(x) - \nabla f(\bar{x})\| \leq L \|x - \bar{x}\| \quad \forall x, \bar{x} \in N$$

Then $\sum_{k \geq 0} \cos^2 \theta_k \|\nabla f_k\|^2 < \infty$

Proof: From the curvature condition (2^{nd} Wolfe condition) we have $\nabla f_{k+1}^T P_k \geq c_2 \nabla f_k^T P_k$. Subtracting $\nabla f_k^T P_k$ from both sides: $(\nabla f_{k+1} - \nabla f_k)^T P_k \geq (c_2 - 1) \nabla f_k^T P_k$. From the Lipschitz condition: $\|\nabla f_{k+1} - \nabla f_k\| \leq L \alpha_k \|P_k\|$.

$$\text{So } (\nabla f_{k+1} - \nabla f_k)^T P_k \stackrel{\text{Cauchy-Schwarz}}{\leq} \|\nabla f_{k+1} - \nabla f_k\| \cdot \|P_k\| \stackrel{\text{Lipschitz}}{\leq} \alpha_k L \|P_k\|^2$$

$$\therefore (c_2 - 1) \nabla f_k^T P_k \leq (\nabla f_{k+1} - \nabla f_k)^T P_k \leq \alpha_k L \|P_k\|^2$$

$$\Rightarrow \alpha_k \geq \frac{(c_2 - 1) \nabla f_k^T P_k}{L \|P_k\|^2} \quad \text{substitute } \begin{matrix} \downarrow f_{k+1} \\ \downarrow f_k \end{matrix}$$

The sufficient decrease condition says $f(x_k + \alpha_k P_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T P_k$

$$\therefore f_{k+1} \leq f_k + c_1 \underbrace{\frac{(c_2 - 1)}{L}}_C \underbrace{\frac{(\nabla f_k^T P_k)^2}{\|P_k\|^2}}_{\cos^2 \theta_k \|\nabla f_k\|^2} = f_k + c_1 \|\nabla f_k\|^2 \cos^2 \theta_k$$

By summing this expression for all indices 0 to k :

$$f_{k+1} \leq f_0 - c \sum_{j=0}^k \cos^2 \theta_j \| \nabla f_j \|^2$$

Since f is bounded from below $f_0 - f_{k+1} < M$ for some $M > 0$. Then.

$$\begin{aligned} \sum_{j=0}^k \cos^2 \theta_j \| \nabla f_j \|^2 &\leq \frac{f_0 - f_{k+1}}{c} < \frac{M}{c} \quad \forall k \\ \therefore \sum_{j=0}^{\infty} \cos^2 \theta_j \| \nabla f_j \|^2 &< \infty \end{aligned}$$

This last inequality is called the Touterdijk condition.

This condition implies that $\cos^2 \theta_k \| \nabla f_k \|^2 \xrightarrow{k \rightarrow \infty} 0$.

If our method for selecting p_k ensures that $\cos \theta_k > 0$
then $\exists S \ni \cos \theta_k \geq s > 0 \quad \forall k$. (descent direction)

Consequently, we get $\lim_{k \rightarrow \infty} \| \nabla f_k \|^2 = 0$ (Globally Convergent)

* Clearly, steepest descent, if used with a line search method that satisfies the Wolfe conditions, yields a sequence of gradients that converge to 0.

* Other algorithms produce gradient sequences that converge to 0 if $\cos \theta_k > 0 \quad \forall k$ and Wolfe conditions are satisfied.

Fact: For line search with $p_k^T \nabla f_k < 0$, we can only show that (as above), the stationary points attract the iterations. The use of $\nabla^2 f_k$ is needed to strengthen these results.

57

For Newton and quasi-Newton methods we have

$\cos \theta_k \geq \frac{1}{M}$ where $\|\mathcal{B}_k\| \cdot \|\mathcal{B}_k^{-1}\| \leq M \forall k$ with the update $P_k = -\mathcal{B}_k^{-1} \nabla f_k$. So these algorithms are also globally convergent ($\lim_{k \rightarrow \infty} \|\nabla f_k\| = 0$). (exercise 3-5)

Convergence Rate of Steepest Descent

Suppose we have $f(x) = \frac{1}{2} x^T Q x - b^T x$, $Q > 0$, symmetric. Then $\nabla f(x) = Qx - b$ and the minimizes is $x^* = Q^{-1}b$.

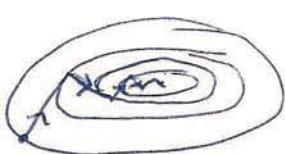
Consider $f(x_k + \alpha \nabla f_k) = \frac{1}{2} (x_k - \alpha \nabla f_k)^T Q (x_k - \alpha \nabla f_k) - b^T (x_k - \alpha \nabla f_k)$

Taking the derivative of this w.r.t α and setting to 0:

$$\alpha_k = \frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k} \text{ is the optimal step size.}$$

Then

$$x_{k+1} = x_k - \left(\frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k} \right) \nabla f_k \quad \text{where } \nabla f_k = Qx_k - b.$$



Define $\|x\|_Q^2 \triangleq x^T Q x$. Then, since $x^* = Q^{-1}b$,

$$\frac{1}{2} \|x - x^*\|_Q^2 = \frac{1}{2} (x - x^*)^T Q (x - x^*) = f(x) - f(x^*)$$

Noting that $Q(x_k - x^*) = \nabla f_k$, we have

$$\|x_{k+1} - x^*\|_Q^2 = \left[1 - \frac{(\nabla f_k^T \nabla f_k)^2}{(\nabla f_k^T Q \nabla f_k)(\nabla f_k^T Q^{-1} \nabla f_k)} \right] \cdot \|x_k - x^*\|_Q^2$$

(exercise 3-7)

Thm 3.3 When steepest descent with exact line search is applied to the strongly convex quadratic function $f(x) = \frac{1}{2}x^T Q x - b^T x$, the error norm satisfies

$$\|x_{k+1} - x^*\|_Q^2 \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \right)^2 \|x_k - x^*\|_Q^2$$

where $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ are the eigenvalues of Q .

$\therefore f(x)$ converges to $f(x^*)$ linearly.

* If $Q = \lambda I$, then convergence is achieved in one step.

* As $R(Q) = \lambda_n / \lambda_1$ increases, convergence speed degrades.

(Prove this theorem as an exercise.)

Thm 3.4 Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is twice cont. diff. and that the iterates generated by steepest descent and exact line search converge to x^* at which $\nabla^2 f(x^*) > 0$. Let r be any scalar $r \in \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}, 1 \right)$ where $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ are the eigenvalues of $\nabla^2 f(x^*)$. Then for all k sufficiently large, we have

$$\{f(x_{k+1}) - f(x^*)\} \leq r^2 \{f(x_k) - f(x^*)\}$$

Newton's Method. Consider $p_k^N = -\nabla f_k^{-1} \nabla f_k$. In general, since $\nabla^2 f_k$ is not positive definite, p_k^N may not be a descent direction at times.

Here we discuss the local convergence properties of the Newton method.

Thm 3.5 Suppose that f is twice cont. diff. and $\nabla^2 f(x)$ is Lipschitz continuous in a neighborhood of x^* at which the sufficient conditions are satisfied ($\nabla f(x^*)=0, \nabla^2 f(x^*) \geq 0$). Consider the iteration $x_{k+1} = x_k + P_k^{-1}$. Then

- if the starting point x_0 is sufficiently close to x^* , the sequence of iterates converges to x^* ;
- the rate of convergence of $\{x_k\}$ is quadratic;
- the sequence of gradient norms $\{\|\nabla f(x)\|\}$ converges quadratically to zero.

$$\text{Note: } \nabla f_x = 0.$$

Proof: $x_k + P_k^{-1} - x^* = x_k - x^* - \nabla f_k^{-1} \nabla f_k$

$$= \nabla f_k^{-1} \left[\nabla^2 f_k (x_k - x^*) - (\nabla f_k - \nabla f_x) \right]$$

From Taylor's Thm: $\nabla f_k - \nabla f_x = \int_0^1 \nabla^2 f(x_k + t(x^* - x_k))(x_k - x^*) dt$.

$$\begin{aligned} \|\nabla^2 f(x_k)(x_k - x^*) - (\nabla f_k - \nabla f_x)\| &= \left\| \int_0^1 [\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))] (x_k - x^*) dt \right\| \\ &\leq \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))\| \cdot \|x_k - x^*\| dt \\ &\leq \|x_k - x^*\|^2 \int_0^1 L t dt = \frac{1}{2} L \|x_k - x^*\|^2 \end{aligned}$$

Since $\nabla^2 f_x$ is nonsingular $\exists r > 0$ s.t. $\|\nabla f_k^{-1}\| \leq 2\|\nabla^2 f_x^{-1}\| \quad \forall x_k$ with $\|x_k - x^*\| \leq r$. Substituting in the first equation ...

(60)

$$\|x_k + p_k - x^*\| \leq L \|\nabla^2 f(x^*)^{-1}\| \cdot \|x_k - x^*\|^2 = \tilde{L} \|x_k - x^*\|^2$$

where $\tilde{L} \triangleq L \|\nabla^2 f_*^{-1}\|$. If $\|x_0 - x^*\| \leq \min(\tau, \frac{1}{2\tilde{L}})$

then (by induction) the sequence $\{x_k\}$ converges to x^* , and the convergence rate is quadratic.

Using $x_{k+1} - x_k = p_k$ and $\nabla f_k + \nabla^2 f_k p_k = 0$:

$$\|\nabla f_{k+1}\| = \|\nabla f(x_{k+1}) - \nabla f_k - \nabla^2 f(x_k) p_k\|$$

$$= \left\| \int_0^1 \nabla^2 f(x_k + t p_k) (x_{k+1} - x_k) dt - \nabla^2 f(x_k) p_k \right\|$$

$$\leq \int_0^1 \|\nabla^2 f(x_k + t p_k) - \nabla^2 f(x_k)\| \cdot \|p_k\| dt$$

$$\text{if } \leq \frac{1}{2} L \|p_k\|^2 \leq \frac{1}{2} L \|\nabla^2 f(x_k)^{-1}\| \cdot \|\nabla f_k\|^2$$

$$\leq 2L \|\nabla^2 f_*^{-1}\| \cdot \|\nabla f_k\|^2$$

∴ the gradient norm converges quadratically. □

Quasi-Newton Methods

$$p_k = -B_k^{-1} \nabla f_k, B_k > 0, \text{sym.}$$

Thm 3.6: Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is twice cont. diff. Consider the iteration $x_{k+1} = x_k + \alpha_k p_k$ where p_k is a descent direction ($\nabla f_k^T p_k < 0$) and α_k satisfies the Wolfe conditions with $c_1, 2, \gamma_2$. If $\{x_k\}$ converges to x^* $\nabla f_* = 0$ and $\nabla^2 f_* > 0$ and if

$$\lim_{k \rightarrow \infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0$$

then

- i) $\alpha_k = 1$ is admissible $\forall k > k_0$;
- ii) if $\alpha_k = 1 \quad \forall k > k_0$, $\{x_k\}$ converges to x^* superlinearly.

* Note that if $c_1 > 1/2$, the Wolfe conditions would exclude the minimizer of a quadratic and $\alpha_k = 1$ may not be admissible.

* If p_k is a quasi-Newton direction then the limit condition in Thm 3.6 ~~comes~~ is equivalent to

$$\lim_{k \rightarrow \infty} \frac{\|(\mathbf{B}_k - \nabla^2 f^*) p_k\|}{\|p_k\|} = 0 \quad (*)$$

hence a quasi-Newton method could achieve superlinear convergence even if $\{\mathbf{B}_k\}$ does not converge to $\nabla^2 f^*$. It suffices that \mathbf{B}_k become increasingly accurate approximations of $\nabla^2 f^*$ (i.e. $\|\mathbf{B}_k - \nabla^2 f^*\|$ monotonically decreasing but not converging to zero).

(*) above is necessary and sufficient for the superlinear convergence of quasi-Newton methods.

Thm 3.7: Suppose $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is twice cont. diff. Consider $x_{k+1} = x_k + p_k$ (i.e. $\alpha_k = 1$) and $p_k = -\mathbf{B}_k^{-1} \nabla f_k$. Assume $\{x_k\}$ converges to $x^* \neq \nabla f^* = 0$ and $\nabla^2 f^* > 0$. Then $\{x_k\}$ converges superlinearly iff $\lim_{k \rightarrow \infty} \frac{\|(\mathbf{B}_k - \nabla^2 f^*) p_k\|}{\|p_k\|} = 0$.

(see pg 471 in book for proof).

(3.4) Newton's Method with Hessian Modification

Away from a local solution, if the Hessian $\nabla^2 f_k$ is not positive definite, $P_k^N = -\nabla^2 f_k^{-1}$ may be a non-definite matrix (or a negative-definite one). Then, the Newton direction may be a non-descent direction. Note that $\nabla^2 f_k P_k^N = -\nabla f_k$ is the system of equations that define P_k^N . The update direction can be obtained by, for instance, Gaussian elimination.

Alg. 3.2 Line Search Newton with Modification

Given initial point x_0 .

for $k=0, 1, 2, \dots$

Factorize $B_k = \nabla^2 f_k + E_k$ where $E_k = 0$ if $\nabla^2 f_k > 0$, otherwise E_k is chosen to ensure that $B_k > 0$.

Solve $B_k P_k = -\nabla f_k$

Set $x_{k+1} \leftarrow x_k + \alpha_k P_k$ where α_k satisfies Wolfe cond.
end

Thm. 3.8 Let f be twice cont. diff. on an open set D , and assume that the starting point x_0 of Alg. 3.2 is such that the level set $L = \{x \in D : f(x) \leq f(x_0)\}$ is compact. Then if the bounded modified factorization property holds we have that $\lim_{k \rightarrow \infty} \nabla f_k = 0$.

Defn. $\{B_k\}$ satisfies the bounded modified factorization property when $\{B_k\}$ have bounded condition number whenever $\{\nabla^2 f_k\}$ ~~have~~ are bounded; that is

$$K(B_k) = \|B_k\| \cdot \|B_k^{-1}\| \leq C \quad \text{for some } C > 0 \text{ \forall k.}$$

Eigenvalue Modification

Example: Consider $\nabla f_k = \begin{bmatrix} 1 & 3 \\ -3 & 2 \end{bmatrix}$ for some x_k and

$\nabla^2 f_k = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & -1 \end{bmatrix}$, which is indefinite. By spectral decomposition theorem ($\alpha = I$ and $\Lambda = \text{diag } \nabla^2 f_k$):

$$\nabla^2 f_k = \alpha \Lambda \alpha^T = \sum_{i=1}^3 \lambda_i q_i q_i^T$$

The Newton step is $P_k^N = -\begin{bmatrix} 10 & 3 \\ 3 & -1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ -3 \end{bmatrix} = \begin{bmatrix} -0.1 \\ 1 \\ 2 \end{bmatrix}$.

$P_k^{NT} \nabla f_k = \{-0.1, 1, 2\} \begin{bmatrix} 1 \\ -3 \end{bmatrix} = -0.1 - 3 + 4 > 0$ so P_k^N is not a descent direction. Suppose we replace the negative eigenvalue of $\nabla^2 f_k$ by $\delta = 10^{-8}$:

$$B_k = \sum_{i=1}^2 \lambda_i q_i q_i^T + \delta q_3 q_3^T = \text{diag}(10, 3, 10^{-8})$$

The search direction is now $P_k = -B_k^{-1} \nabla f_k = -\sum_{i=1}^2 \frac{1}{\lambda_i} q_i (q_i^T \nabla f_k)$

Although $P_k^T \nabla f_k < 0$, the $\approx -(2 \times 10^8) q_3$ huge length of $\|P_k\|$ violates the spirit of Newton updates that assume local quadratic approximations.

If A is a symmetric matrix with spectral decomposition $A = Q \Lambda Q^T$, then ΔA with minimum Frobenius norm that ensures that $\lambda_{\min}(A + \Delta A) \geq \delta$ is given by $\Delta A = Q \text{diag}(\tau_i) Q^T$ with

$$\tau_i = \begin{cases} 0, & \lambda_i \geq \delta \\ \delta - \lambda_i, & \lambda_i < \delta \end{cases}$$

Here $\lambda_{\min}(A)$ denotes the smallest eigenvalue of A .

Similarly ΔA with minimum Euclidean norm is given by $\Delta A = \tau I$ with $\tau = \max(0, \delta - \lambda_{\min}(A))$ and we use $A + \tau I$ as the modified matrix.

Adding a Multiple of the Identity Matrix

In practice, estimating the smallest eigenvalue of the Hessian might be computationally expensive or numerically ill-conditioned. The following algorithm tries successively larger τ values.

Alg 3.3 (Cholesky with Added Multiple of Identity)

Choose $\beta > 0$

if $\min_i a_{ii} > 0$, set $\tau_0 \leftarrow 0$, else $\tau_0 = -\min_i (a_{ii}) + \beta$, end
for $k = 0, 1, 2, \dots$

Attempt Cholesky decomposition $L L^T = A + \tau_k I$

if the factorization is successful

stop and return L)

else

$$\tau_{k+1} \leftarrow \max(2\tau_k, \beta);$$

end (if)

end (for)

could be
larger

heuristically
selected

Modified Cholesky Factorization

Every symmetric positive definite matrix A , can be written as $A = LDL^T$ where L is lower triangular and $D > 0$ is diagonal.

Ex

$$\begin{bmatrix} a & c \\ c & b \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ l & 1 \end{bmatrix} \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \begin{bmatrix} 1 & l \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ l & 1 \end{bmatrix} \begin{bmatrix} d_1 & ld_1 \\ 0 & d_2 \end{bmatrix} = \begin{bmatrix} d_1 & ld_1 \\ ld_1 & (l^2d_1 + d_2) \end{bmatrix}$$

$$\therefore d_1 = a, \quad l = c/d_1 = c/a, \quad d_2 = b - l^2 d_1 = b - \frac{c^2}{a^2} a = b - \frac{c^2}{a^2}$$

Alg 3.4 (Cholesky Factorization)

for $j=1, 2, \dots, n$

$$c_{jj} \leftarrow a_{jj} - \sum_{s=1}^{j-1} d_s l_{js}^2; \quad d_j \leftarrow c_{jj};$$

for $i=j+1, \dots, n$

$$c_{ij} \leftarrow a_{ij} - \sum_{s=1}^{j-1} d_s l_{is} l_{js}; \quad l_{ij} \leftarrow c_{ij}/d_j;$$

end end

(66)

If A is indefinite, this algorithm becomes unstable. We can modify A during factorization to ensure d_{ii} are sufficiently positive and that D and L do not have very large entries. $(s > 0, \beta > 0)$

Given two quality control parameters s and β , we require that when computing the j^{th} columns of $L \& D$ (i.e. for each j), we satisfy $d_{jj} \geq s$, $|m_{ij}| \leq \beta$

where $m_{ij} = l_{ij} \sqrt{d_j}$. To achieve these, the rule computing d_{jj} is changed as

$$d_j = \max \left(|c_{jj}|, \left(\frac{\theta_j}{\beta} \right)^2, s \right) \text{ with } \theta_j = \max_{i \in \mathcal{I}} |c_{ij}|$$

This algorithm generates a modified matrix

$$PAP^T + E = L D L^T = M M^T \text{ with } M = L D^{1/2}$$

and P -permutation.

* See pg 54-56 in book for Modified Symmetric Indefinite Factorization.

Step Length Selection Algorithms

Consider $\min_{\lambda > 0} \phi(\lambda)$ where $\phi(\lambda) = f(x_k + \lambda p_k)$.

We assume that p_k is a descent direction ($\phi'(0) < 0$) so we only consider $\lambda > 0$.

(67)

for $f(x) = \frac{1}{2} x^T Q x - b^T x$, the exact solution is given by $\alpha_k = - \frac{\nabla f_k^T P_k}{P_k^T Q P_k}$.

for general nonlinear functions, iterative procedures are needed. These procedures can use only ϕ -values or they can also make use of ϕ' -values.

Interpolation: This line search procedure is based on the interpolation of known function and derivative values of the function ϕ . The procedure generates a decreasing sequence of $\{\alpha_i\}$ where at each step the sufficient decrease condition is satisfied and α_i is not much smaller than α_{i-1} .

The sufficient decrease condition is

$$\phi(\alpha_k) \leq \phi(0) + c_1 \alpha_k \phi'(0)$$

We would like to keep ϕ and ϕ' evaluations at a minimum to achieve computational efficiency.

* For an initial guess α_0 if we have

$$\phi(\alpha_0) \leq \phi(0) + c_1 \alpha_0 \phi'(0)$$

we terminate the search and proceed with α_0 .

(68)

otherwise, the interval $[0, \alpha_0]$ contains acceptable step lengths. Using $\phi(0)$, $\phi'(0)$, and $\phi(\alpha_0)$ we form a quadratic approximation:

$$\phi_q(\alpha) = \left(\frac{\phi(\alpha_0) - \phi(0) - \alpha_0 \phi'(0)}{\alpha_0^2} \right) \alpha^2 + \phi'(0) \alpha + \phi(0)$$

The minimizer of $\phi_q(\alpha)$ is: $\alpha_1 = - \frac{\phi'(0) \alpha_0^2}{2[\phi(\alpha_0) - \phi(0) - \phi'(0) \alpha_0]}$

is the new trial value. If

$$\phi(\alpha_1) \leq \phi(0) + c, \alpha_1, \phi'(\alpha_1)$$

then we proceed with step length α_1 . Otherwise a cubic function fitting $\phi(0)$, $\phi'(0)$, $\phi(\alpha_0)$, $\phi(\alpha_1)$ is obtained: $\phi_c(\alpha) = a \alpha^3 + b \alpha^2 + \phi'(0) \alpha + \phi(0)$

$$\text{where } \begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{\alpha_0^2 \alpha_1^2 (\alpha_1 - \alpha_0)} \begin{bmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{bmatrix} \begin{bmatrix} \phi(\alpha_1) - \phi(0) - \phi'(0) \alpha_1 \\ \phi(\alpha_0) - \phi(0) - \phi'(0) \alpha_0 \end{bmatrix}$$

Differentiating $\phi_c(\alpha)$, we find α_2 , the minimizer of $\phi_c(\alpha)$ in $[0, \alpha_1]$:

$$\alpha_2 = \frac{-b + \sqrt{b^2 - 3a\phi'(0)}}{3a}$$

If α_2 satisfies $\phi(\alpha_2) \leq \phi(0) + c, \alpha_2, \phi'(\alpha_2)$, then we use it; otherwise we could repeat the cubic-interpolation process using $\phi(0)$, $\phi'(0)$, $\phi(\alpha_1)$, $\phi(\alpha_2)$.

If ϕ and ϕ' are computationally equally expensive, then cubic interpolator could use ~~both~~ these at two end points. Cubic interpolator usually produces quadratic convergence to the minimizing α .

Initial Step Length

for Newton and quasi-Newton methods, $\alpha_0 = 1$ is a good choice. For gradient methods, one must use current information. For instance, if we assume that the first order change in f at x_k will be the same as that obtained at the previous step, we choose α_0 so that $\alpha_0 \nabla f_k^T P_k = \alpha_{k-1} \nabla f_{k-1}^T P_{k-1}$.

Then $\alpha_0 = \alpha_{k-1} \frac{\nabla f_{k-1}^T P_{k-1}}{\nabla f_k^T P_k}$ where α_0 is the initial guess for α_k .

Another strategy is to interpolate a quadratic to the data $f(x_{k-1})$, $f(x_k)$, and $\nabla f_{k-1}^T P_{k-1}$ to define

$$\alpha_0 = \frac{2(f_k - f_{k-1})}{\phi'(c_0)}$$

which is the minimizer of the quadratic.

If $x_k \rightarrow x^*$ superlinearly, then the ratio above converges to 1. If we use $\alpha_0 = \min(1, 1.01 \frac{2(f_k - f_{k-1})}{\phi'(c_0)})$, the unit step $\alpha_0 = 1$ will eventually be tried and superlinear convergence properties of Newton and quasi-Newton methods will be achieved.

A Line Search Algorithm for the Wolfe Conditions

We assume p is a descent direction, f is bounded from below in the direction of p and $0 < c_1, 2c_2 < 1$. This algorithm begins with a trial estimate α_i and increases it until an acceptable step length or an interval that brackets desired lengths is found.

In the latter case, zoom is called to successively decrease the length until an acceptable value is found.

Here α_{\max} is a user supplied maximum allowed length. The resulting α_* satisfies the strong Wolfe conditions.

Alg 3.5 Line Search Algorithm

Choose $\alpha_{\max} > 0$, set $\alpha_0 \leftarrow 0$, choose $\alpha_i \in (0, \alpha_{\max})$.

$i \leftarrow 1$

repeat

Evaluate $\phi(\alpha_i)$

if $\phi(\alpha_i) > \phi(0) + c_1 \alpha_i \phi'(0)$ OR $\phi(\alpha_i) > \phi(\alpha_{i-1})$ for $i > 1$

$\alpha_* \leftarrow \text{zoom}(\alpha_{i-1}, \alpha_i)$ and stop

Evaluate $\phi'(\alpha_i)$

if $|\phi'(\alpha_i)| \leq -c_2 \phi'(0)$

Set $\alpha_* \leftarrow \alpha_i$ and stop

if $\phi'(\alpha_i) \geq 0$

Set $\alpha_* \leftarrow \text{zoom}(\alpha_i, \alpha_{i-1})$ and stop

Choose $\alpha_{i+1} \in (\alpha_i, \alpha_{\max})$

$i \leftarrow i + 1$

end (repeat)

(71)

The sequence of trial steps $\{\alpha_i\}$ is monotonically increasing. The interval (α_{i-1}, α_i) which is sent to zoom contains step lengths satisfying the strong Wolfe conditions if one of the following conditions is satisfied:

- i) α_i violates the sufficient decrease condition
- ii) $\phi(\alpha_i) \geq \phi(\alpha_{i-1})$
- iii) $\phi'(\alpha_i) \geq 0$

The last step of the algorithm selects the next trial value α_{i+1} . The selection could be based on interpolation or a multiple of α_i .

The algorithm $\text{zoom}(\alpha_{l_0}, \alpha_{h_0})$ performs fine tuning in the interval $(\alpha_{l_0}, \alpha_{h_0})$ in which

- a) step lengths that satisfy the Wolfe conditions exist
- b) α_{l_0} , among all step lengths generated so far and satisfying the sufficient decrease condition, is the one giving the smallest function value
- c) α_{h_0} is chosen so that $\phi'(\alpha_{l_0})(\alpha_{h_0} - \alpha_{l_0}) < 0$

At each step of zoom, the iterate α_j replaces one of the end points while maintaining all three of (a-c).

Alg 3.6 zoom

Repeat

Interpolate (using quadratic, cubic, bisection) to find
a trial step length α_j between α_{l_0} and α_{u_0} .

Evaluate $\phi(\alpha_j)$

if $\phi(\alpha_j) > \phi(\alpha_0) + c_1 \alpha_j \phi'(\alpha_0)$ or $\phi(\alpha_j) \geq \phi(\alpha_{l_0})$
 $\alpha_{u_0} \leftarrow \alpha_j$

else

Evaluate $\phi'(\alpha_j)$

if $|\phi'(\alpha_j)| \leq -c_2 \phi''(\alpha_0)$

Set $\alpha_* \leftarrow \alpha_j$ and stop

if $\phi'(\alpha_j)(\alpha_{u_0} - \alpha_{l_0}) \geq 0$

$\alpha_{u_0} \leftarrow \alpha_j$

$\alpha_{l_0} \leftarrow \alpha_j$

end (repeat)

If the new estimate α_j satisfies the strong Wolfe conditions, zoom terminates with $\alpha_* = \alpha_j$. If α_j has a lower function value than α_{l_0} , then we set $\alpha_{l_0} \leftarrow \alpha_j$ to maintain (b). If this violates (c), we then set $\alpha_{u_0} \leftarrow \alpha_{l_0}$.

* Derivative-free line search algorithms exist (see Golden section and Fibonacci search).

Exercises

3.3) Show that for $f(x) = \frac{1}{2} x^T Q x - b^T x$, the minimizing $\alpha_k > 0$ along $x_k + \alpha P_k$ is $\alpha_k = -\frac{\nabla f(x_k)^T P_k}{P_k^T Q P_k}$

$$f(x_k + \alpha P_k) = \phi(\alpha) = \frac{1}{2} (x_k + \alpha P_k)^T Q (x_k + \alpha P_k) - b^T (x_k + \alpha P_k)$$

$$\frac{d\phi(\alpha)}{d\alpha} = \frac{1}{2} 2 (x_k + \alpha P_k)^T Q P_k - b^T P_k$$

$$= x_k^T Q P_k + \alpha P_k^T Q P_k - b^T P_k = 0$$

$$\Rightarrow \alpha = \frac{(b^T - x_k^T Q) P_k}{P_k^T Q P_k} = -\frac{\nabla f(x_k)^T P_k}{P_k^T Q P_k} \text{ where } \nabla f(x_k) = Q x_k - b.$$

3.4) Consider $f(x) = \frac{1}{2} x^T Q x - b^T x$. Show that if $x_0 - x^*$ is parallel to an eigenvector of Q , then steepest descent with exact line search will find the solution in one step.

$$\nabla f(x) = Qx - b, \quad x_0 - x^* = \beta q, \text{ where } Qq = \lambda q.$$

Since $x_0 = x^* + \beta q$ for some β , $\nabla f_0 = Q(x^* + \beta q) - b$

However since $Qx^* = b$ and $Qq = \lambda q$, we get

$$\begin{aligned} \nabla f_0 &= \beta \lambda q. \quad \text{Then } x_1 = x_0 - \alpha \nabla f_0 = (x^* + \beta q) - \alpha \beta \lambda q \\ &= x^* + \beta(1 - \alpha \lambda) q \end{aligned}$$

So if $\alpha = \frac{1}{\lambda}$, then $x_1 = x^*$.

3.5) I assume that the book assumes $\| \cdot \| \equiv \| \cdot \|_2$ in this case.

Result 1: Consider any matrix norm that is consistent with a vector norm. Then $\| A \| = \sup_x \frac{\| Ax \|}{\| x \|}$.

So $\| A \| \geq \frac{\| Ax \|}{\| x \|} \quad \forall x, \text{ except when } x=0.$

\therefore In general, $\| Ax \| \leq \| A \| \cdot \| x \|$; this is also true for $\| \cdot \|_2$.

$$\text{Now } \frac{\| Bx \|}{\| x \|} = \frac{\| Bx \|}{\| B^{-1}Bx \|} \geq \frac{\| Bx \|}{\| B^{-1} \| \cdot \| Bx \|} = \frac{1}{\| B^{-1} \|} \therefore \| Bx \| \geq \frac{\| x \|}{\| B^{-1} \|}.$$

Now for $\| \cdot \|_2$, consider the claim $\cos \theta_k \geq \frac{1}{M}$ for the Newton update where $\kappa(B_k) = \| B_k \| - \| B_k^{-1} \| \leq M$.

$$\begin{aligned} \cos \theta_k &= \frac{\nabla f_k^T B_k^{-1} \nabla f_k}{\| \nabla f_k \| \cdot \| B_k^{-1} \nabla f_k \|} \geq \frac{\lambda_1^{-1} \| \nabla f_k \|^2}{\| \nabla f_k \| \cdot \| B_k^{-1} \nabla f_k \|} \geq \frac{\lambda_1^{-1} \| \nabla f_k \|}{\| B_k^{-1} \| \cdot \| \nabla f_k \|} \geq \frac{\lambda_1^{-1}}{\lambda_n^{-1}} \\ &= \frac{\lambda_n}{\lambda_1} = \frac{1}{\kappa(B_k)} \geq \frac{1}{M}. \end{aligned}$$

In the above sequence of inequalities, we have the eigenvalues of B_k as $\lambda_1 \geq \dots \geq \lambda_n$ and the eigenvalues of B_k^{-1} are $\lambda_1^{-1} \leq \dots \leq \lambda_n^{-1}$. Recall that for symmetric positive definite matrices (ASD)

$$\| A \|_2 = \lambda_{\max}^{(A)}$$

$$\text{so } \| B_k \|_2 = \lambda_1 \quad \text{and} \quad \| B_k^{-1} \|_2 = \lambda_n^{-1}.$$

Chapter 4: Trust-Region Methods

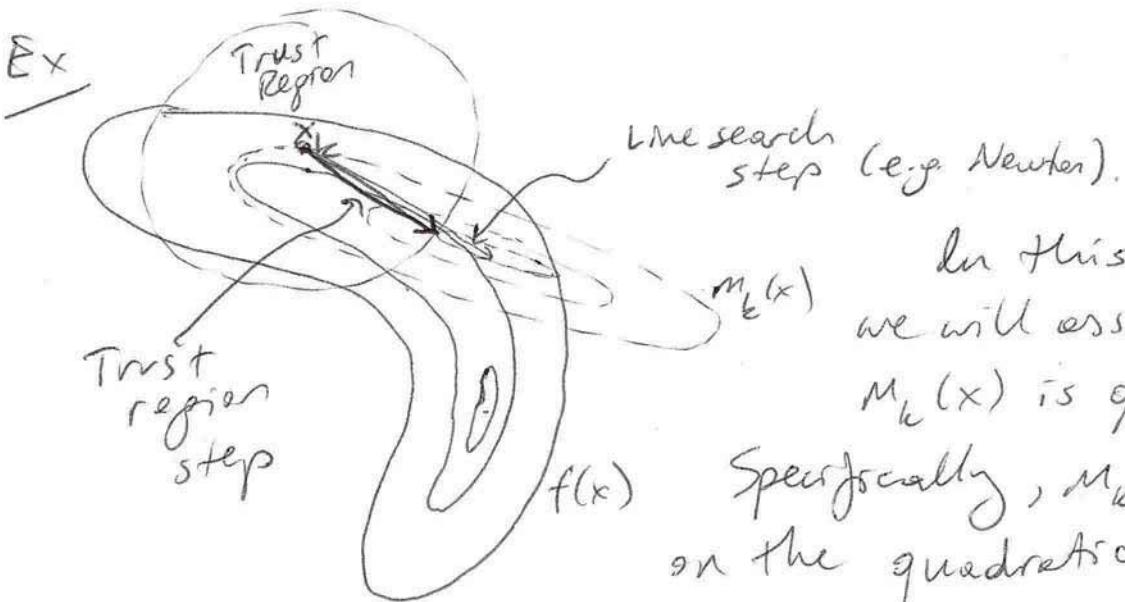
Line search methods use a local quadratic model of f to find a search direction and focus on finding a suitable step length along this direction.

Trust region methods define a region around the current iterate within which they trust the model to be adequate representation, then choose a step to the approximate minimizer in this region.

If the step is not acceptable, the size of the trust region is reduced and a new minimizer is found. When the trust region is altered, in general, the direction of the step changes.

If trust region is too small, an opportunity to make a large step is missed. If it is too large, then the chances of redoing the step with a smaller region becomes large, reducing computational efficiency again.

After repeated successful steps, the size of the region could be increased to seek more ambitious. If the step fails, then the model is not adequate for the large region and the region is reduced in size.



In this discussion, we will assume that $m_k(x)$ is quadratic.

Specifically, $m_k(x)$ is based on the quadratic Taylor expansion of $f(x)$ around x_k :

$$f(x_k + p) = f_k + g_k^T P + \frac{1}{2} p^T \nabla^2 f(x_k + t_p) p$$

where $f_k \triangleq f(x_k)$ and $g_k \triangleq \nabla f(x_k)$ and $t \in (0, 1)$.

If $B_k \approx \nabla^2 f(x_k + t_p)$, then $m_k(x) = f_k + g_k^T P + \frac{1}{2} p^T B_k p$,

where B_k is some symmetric matrix. The difference between $m_k(p)$ and $f(x_k + p)$ is $\mathcal{O}(||p||^2)$. When

$B_k = \nabla^2 f(x_k)$, then the error is $\mathcal{O}(||p||^3)$. In

this latter case, we refer to the method as the trust-region Newton method.

To obtain each step, we solve

$$\min_{p \in \mathbb{R}^n} m_k(p) = f_k + g_k^T P + \frac{1}{2} p^T B_k p \text{ s.t. } \|p\| \leq \Delta_k$$

where $\Delta_k > 0$ is the trust region radius. If $\|I - I_2\|$ is used for the trust region, then both the model and the region are quadratic in p .

When $B_k > 0$ and $\|B_k^{-1}g_k\| \leq \Delta_k$, the minimum of $m_k(x)$ is in the trust region and the step is simply $P_k^B = -B_k^{-1}P_k$ (which is the Newton step for $m_k(x)$ as if there is no trust-region constraint).

In this case, P_k^B is called the full step. In other cases (i.e. $\|B_k^{-1}g_k\| > \Delta_k$ so the solution of the constrained optimization problem is on the boundary of the trust region), the solution or a good approximation can be found easily.

Given a step P_k , define the ratio

$$\rho_k \triangleq \frac{f(x_k) - f(x_k + P_k)}{m_k(0) - m_k(P_k)} \quad (\text{actual reduction})$$

Since P_k is selected such that $m_k(P_k) \leq m_k(0)$, the predicted reduction is always ~~is~~ nonnegative.

Therefore, $\rho_k < 0 \Rightarrow f(x_k + P_k) > f(x_k)$ and the step P_k must be rejected. If $\rho_k \approx 1$, then there is a good agreement between the model and the function. In that case, the trust region can be extended in the next iteration.

If $0 < p_k \leq 1$, then we can perform the current iteration without altering the trust region but we shrink it for the next iteration.

Alg. 4.1 Trust Region Adjustment

Given $\hat{\Delta} > 0$, $\Delta_0 \in (0, \hat{\Delta})$, $\gamma \in \{0, \frac{1}{4}\}$

for $k = 0, 1, 2, \dots$

Obtain p_k by approximately solving

$$\min_{p \in \mathbb{R}^n} M_k(p) \text{ s.t. } \|p\| \leq \Delta_k$$

$$\text{Evaluate } f_k = \frac{f(x_k) - f(x_k + p_k)}{M_k(0) - M_k(p_k)}$$

if $p_k < \frac{1}{4}$, $\Delta_{k+1} = \Delta_k / 4$ (shrink trust region)

Note that
we always
have $\Delta_k \in (0, \hat{\Delta}]$.

else if $p_k > \frac{3}{4}$ and $\|p_k\| = \Delta_k$, $\Delta_{k+1} = \min(2\Delta_k, \hat{\Delta})$

$$\text{else } \Delta_{k+1} = \Delta_k$$

if $p_k > \gamma$, $x_{k+1} = x_k + p_k$

else $x_{k+1} = x_k$

end (for)

if $\|p_k\| < \Delta_k$
we do not
change the
trust region
since the minimum
is inside.

There $\hat{\Delta}$ is an overall bound on the step lengths.

At each iteration of Alg. 4.1, we solve the following subproblem:

$$(*) \quad \min_{p \in \mathbb{R}^n} m(p) \triangleq f + g^T p + \frac{1}{2} p^T B p \text{ s.t. } \|p\|_2 \leq \Delta$$

Thm 4.1: The vector p^* is a global solution of the trust-region problem $(*)$ iff p^* is feasible and $\exists \lambda \geq 0$ s.t. the following are satisfied

$$(i) \quad (B + \lambda I) p^* = -g$$

$$(ii) \quad \lambda(\Delta - \|p^*\|) = 0$$

$$(iii) \quad (B + \lambda I) \geq 0$$

To prove this result, first consider the following:

Lemma 4.7: Let $m(p) = g^T p + \frac{1}{2} p^T B p$ where B is symmetric. Then the following are true:

(i) m attains a minimum iff $B \geq 0$ and $g \in \text{range}(B)$. If $B \geq 0$, then every p

~~satisfying~~ satisfying $Bp = -g$ is a global minimizer of m .

(ii) m has a unique minimizer iff $B > 0$.

Proof: (\Leftarrow) Since $g \in \text{Range}(B)$, $\exists p$ with $Bp = -g$.

$\forall w \in \mathbb{R}^n$, we have

$$\begin{aligned} m(p+w) &= g^T(p+w) + \frac{1}{2}(p+w)^T B(p+w) \\ &= (g^T p + \frac{1}{2}p^T B p) + g^T w + (Bp)^T w + \frac{1}{2}w^T B w \\ &= m(p) + \frac{1}{2}w^T B w \\ &\geq m(p) \quad \text{since } B \geq 0. \end{aligned}$$

Hence, p is a minimizer of m .

(\Rightarrow) Let p be a minimizer of m . Since $\nabla m(p) = 0$ we have $\nabla m(p) = Bp + g = 0 \Rightarrow Bp = -g$ so $g \in \text{Range}(B)$. Also $\nabla^2 m(p) = B \geq 0$.

~~so by the second order sufficiency conditions,~~

For part (ii) (\Leftarrow) as in (i), $w^T B w > 0$ since $B \geq 0$ when $w \neq 0$. For (\Rightarrow), as in (i) we find that $B \geq 0$. However if $B \geq 0$ is not true then $\exists w \neq 0 \ni Bw = 0$, hence ~~B~~ $m(p+w) = m(p)$ so the minimizer p is not unique, which is a contradiction. \square

(81)

Proof of Thm 4.1 (\Leftarrow)

Assume that $\exists \lambda \geq 0$ s.t. the

conditions (i), (ii), (iii) in Thm 4.1 are satisfied.

Then, due to Lemma 4.7 (i), p^* is a global

$$\begin{aligned} \text{minimum of } \hat{m}(p) &\triangleq g^T p + \frac{1}{2} p^T (B + \lambda I) p \\ &= m(p) + \frac{\lambda}{2} p^T p \end{aligned}$$

since $\hat{m}(p) \geq \hat{m}(p^*)$, we have (substitute directly)

$$m(p) \geq m(p^*) + \frac{\lambda}{2} [p^{*T} p^* - p^T p]$$

Also, since $\lambda(\Delta - \|p^*\|) = 0$ and therefore $\lambda(\Delta^2 - p^{*T} p^*) = 0$

(multiply by $(\Delta + \|p^*\|)$)

$$\text{we have } m(p) \geq m(p^*) + \frac{\lambda}{2} (\Delta^2 - p^T p)$$

Hence, from $\lambda \geq 0$, we have $m(p) \geq m(p^*) \quad \forall p$ with $\|p\| \leq \Delta$. Therefore p^* is a global minimizer of (*), the constrained trust-region problem.

(skip) For the converse (\Rightarrow) assume that p^* is a global solution of (*). In the case $\|p^*\| < \Delta$,

p^* is an unconstrained minimizer of m , so

$$\nabla m(p^*) = B p^* + g = 0, \quad \nabla^2 m(p^*) = B \geq 0$$

and therefore the properties (i)-(iii) hold for $\lambda=0$.

In the case $\|p^*\| = \Delta$, the condition (ii) is true and p^* solves $\min m(p)$ s.t. $\|p\| = \Delta$.

Using Lagrange's method for this constrained problem (we will discuss this later in detail), we find that $\exists \lambda$ such that

$$L(p, \lambda) = m(p) + \frac{\lambda}{2} (p^T p - \Delta^2)$$

has a stationary point at p^* . Setting $\nabla_p L(p^*, \lambda) = 0$ we obtain $Bp^* + g + \lambda p^* = 0 \Rightarrow (B + \lambda I)p^* = -g$ so condition (i) holds.

Since $m(p) \geq m(p^*)$ $\forall p$ with $p^T p = p^{*\top} p^* = \Delta^2$, we have $m(p) \geq m(p^*) + \frac{\lambda}{2} (p^{*\top} p^* - p^T p)$ as before. Substituting $g = -(B + \lambda I)p^*$ into this, we get $\frac{1}{2}(p - p^*)^T (B + \lambda I)(p - p^*) \geq 0$.

This shows (since true for many p 's) condition (iii) holds.

From Lemma 4.7(i), we have that p^* minimizes \tilde{m} so $m(p) \geq m(p^*) + \frac{\lambda}{2} (p^{*\top} p^* - p^T p)$ as before. If there were only negative values of λ that satisfied conditions (i) and (iii), we would have $m(p) \geq m(p^*)$ when $\|p\| \geq \|p^*\| = \Delta$. Since p^* minimizes m for $\|p\| = \Delta$, p^* is a global unconstrained minimizer of m . But then $Bp = -g$ and $B \geq 0$ so (i) & (iii) are true for $\lambda = 0$, which contradicts $\lambda < 0$, so we conclude that $\lambda \geq 0$. \square

Consider the three conditions in Thm 4.1:

$$(i) (B + \lambda I)p^* = -g$$

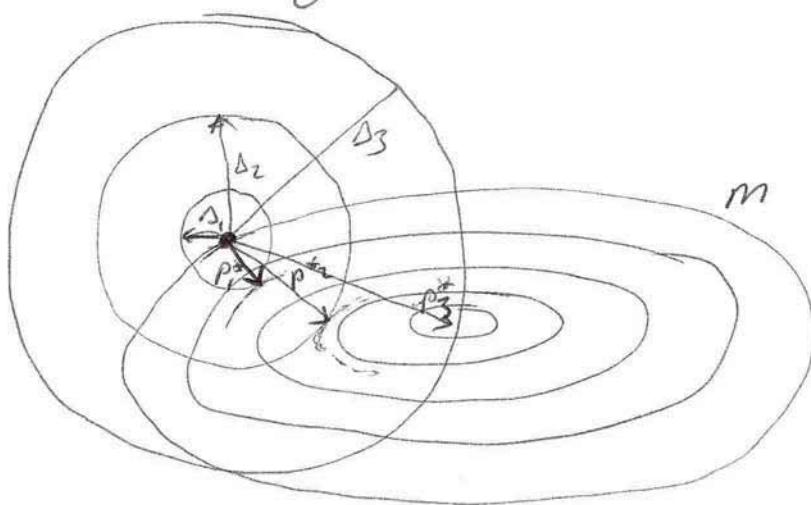
$$(ii) \lambda(\Delta - \|p^*\|I) = 0$$

$$(iii) (B + \lambda I) \geq 0$$

Condition (ii) guarantees that either $\lambda = 0$ or $(\Delta - \|p^*\|I) = 0$. When the solution to the unconstrained quadratic problem lies inside the trust region, we have $\Delta - \|p^*\|I > 0$, so we must have $\lambda = 0$.

Then, we must also have $Bp^* = -g$ (from (i)) and $B \geq 0$ (from (iii)). If the trust region is small such that $\Delta - \|p^*\|I \leq 0$, then $\lambda > 0$ in general and $\nabla p^* = -Bp^* - g = -\nabla m(p^*)$. Thus, when $\lambda > 0$, p^* is collinear with the negative gradient of m .

Ex



(84)

The Cauchy Point : As we have seen, line search methods can be globally convergent even when we don't use the optimal step length. Similarly loose conditions apply to the trust region methods.

An update that remains within the trust region and ^{that} gives a sufficient reduction is satisfactory,

Alg. 4-2 (Cauchy Point Calculation)

Find the vector p_k^s that solves

$$P_k^s = \underset{p \in \mathbb{R}^n}{\operatorname{argmin}} f_k + g_k^T p \text{ s.t. } \|p\| \leq \delta_k$$

Calculate $\tau_k > 0$ that minimizes $M_k(\tau p_k^s)$ s.t. satisfying the trust-region bound:

$$\tau_k = \underset{\tau \geq 0}{\operatorname{argmin}} M_k(\tau p_k^s) \text{ s.t. } \|\tau p_k^s\| \leq \delta_k$$

$$\text{Set } p_k^c = \tau_k p_k^s$$

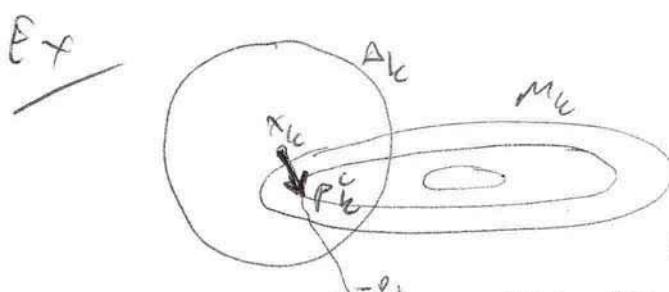
Note: The Cauchy point is simply the minimizer of M_k along the steepest descent direction $-g_k$, subject to the trust region bound.

Notice that in Alg 4.2, $P_k^S = -\frac{\Delta_k}{\|g_k\|} g_k$.

- * If $g_k^T B_k g_k \leq 0$, then $M_k(\tau P_k^S)$ decreases monotonically with τ whenever $g_k \neq 0$, so τ_k is the largest value that satisfies the trust region bound; $\tau_k = 1$.
- * If $g_k^T B_k g_k > 0$, then $M_k(\tau P_k^S)$ is a convex quadratic in τ so τ_k is ~~the~~ the smaller of 1 and $\frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}$, the latter being the unconstrained minimizer.

Summary: $P_k^C = -\tau_k \frac{\Delta_k}{\|g_k\|} g_k$

where $\tau_k = \begin{cases} 1 & \text{if } g_k^T B_k g_k \leq 0 \\ \min\left(-\frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}, 1\right) & \text{o.w.} \end{cases}$



The Cauchy step is easy to calculate. A trust region method is globally convergent if its steps P_k give a reduction in M_k that is at least some fixed positive multiple of the decrease attained by the Cauchy step.

Notice that the Cauchy step is simply steepest descent with a particular choice of the step length. Therefore, its convergence rate might be poor.

If $B_k > 0$, we could improve it by ~~taking~~.

using $P_k^B = -B_k^{-1}g_k$ whenever possible with $\|P_k^B\| \leq \Delta_k$.

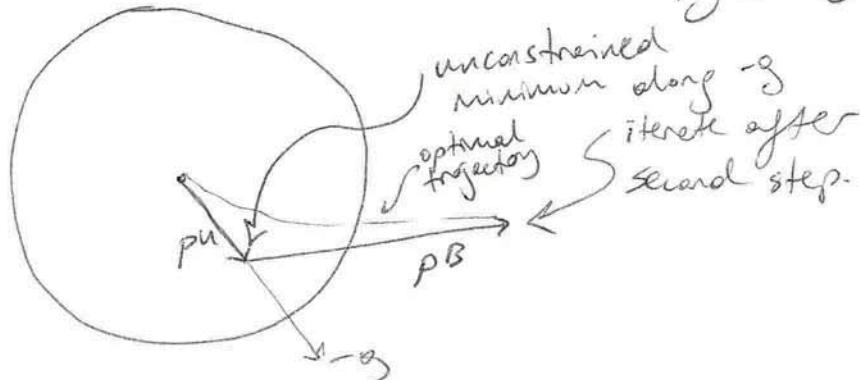
If $B_k = \nabla^2 f_k$, this will yield superlinear convergence.

The Dogleg Method: This can be used if $B > 0$.

$B > 0 \Rightarrow P^B = -B^{-1}g$ is the unconstrained minimizer of m . If $\Delta \geq \|P^B\|$, then $P^*(\Delta) = P^B$.

When $\Delta \ll \|P^B\|$, the first region $\|P\| \leq \Delta$ implies that the quadratic term in m has little effect. Then, we get the approximate solution

$$P^*(\Delta) \approx -\frac{\Delta}{\|P\|} g \quad (\Delta \ll \|P^B\|)$$



The first intermediate step is the unconstrained minimizer of m in the direction of $-g$:

$$p^u = -\frac{g^T g}{g^T B g} g$$

The second intermediate step is $p^B = -B^{-1}g$

Specifically, we have (for $\tau \in \{0, 1\}$)

$$\tilde{p}(\tau) = \begin{cases} \tau p^u & 0 \leq \tau \leq 1 \\ p^u + (\tau-1)(p^B - p^u) & 1 \leq \tau \leq 2 \end{cases}$$

The optimal curved trajectory (dashed) that occurs for intermediate Δ values, is thus approximated by two piecewise linear steps.

Lemma 4.2 Let $B > 0$. Then

(i) $\|\tilde{p}(\tau)\|$ is increasing with τ

(ii) $m(\tilde{p}(\tau))$ is decreasing with τ .

Proof: For $\tau \in \{0, 1\}$, both (i) and (ii) clearly hold. For $\tau \in \{1, 2\}$, define $h(\alpha)$ as

$$\begin{aligned} h(\alpha) &= \frac{1}{2} \|\tilde{p}(1+\alpha)\|^2 = \frac{1}{2} \|p^u + \alpha(p^B - p^u)\|^2 \\ &= \frac{1}{2} \|p^u\|^2 + \alpha p^{uT} (p^B - p^u) + \frac{1}{2} \alpha^2 \|p^B - p^u\|^2 \end{aligned}$$

We need to show that $h'(\alpha) \geq 0$ for $\alpha \in (0, 1)$.

$$\begin{aligned} h'(\alpha) &= -p^u T (p^u - p^B) + \alpha \|p^u - p^B\|^2 \\ &\geq -p^u T (p^u - p^B) = \frac{g^T g}{g^T B g} g^T \left(\frac{-g^T g}{g^T B g} g + B^{-1} g \right) \\ &= g^T g \frac{g^T B^{-1} g}{g^T B g} \left\{ 1 - \frac{(g^T g)^2}{(g^T B g)(g^T B^{-1} g)} \right\} \geq 0 \end{aligned}$$

For (ii), let $\hat{h}(\alpha) = m(\tilde{p}(1+\alpha))$ and show ^{Cauchy-Schwarz} that $\hat{h}'(\alpha) \leq 0$ for $\alpha \in (0, 1)$,

$$\begin{aligned} \hat{h}'(\alpha) &= (\rho^B - \rho^u)^T (g + B\rho^u) + \alpha (\rho^B - \rho^u)^T B (\rho^B - \rho^u) \\ &\leq (\rho^B - \rho^u)^T (g + B\rho^u + B(\rho^B - \rho^u)) \\ &= (\rho^B - \rho^u)^T (g + B\rho^B) = 0 \end{aligned}$$

If $\|\tilde{p}(\tau)\| \leq \Delta$, then we take the full step p^B .

otherwise, the intersection of the trust region boundary with the direction of p^B is used:

$$\|p^u + (\tau-1)(\rho^B - \rho^u)\|^2 = \Delta^2$$

needs to be solved to find this intersection for τ .

Two-Dimensional Subspace Minimization

If $B > 0$, the dogleg strategy can be improved by ordering the search for p to the entire two-dimensional space spanned by p^u and p^B (i.e. $-g$ and $-B^{-1}g$, equivalently):

$$\min_p m(p) = f + g^T p + \frac{1}{2} p^T B p \text{ s.t. } \|p\| \leq \delta \\ p \in \text{span}\{g, B^{-1}g\}$$

Clearly, the Cauchy point p^c is feasible for this problem, hence this update will be at least as good, i.e. will converge.

Global Convergence

We will investigate the decrease in the model function when using approximate solutions to the trust region problem.

Lemma 4.3. The Cauchy point p_k^c satisfies

$$m_k(0) - m_k(p_k^c) \geq \frac{1}{2} \|g_k\| \min\left(\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right)$$

(90)

Proof: Consider the following for $m(p) = f + g^T p + \frac{1}{2} p^T B p$

$$\begin{aligned}
 \boxed{\text{Assume } g^T B g < 0} \quad m(p^c) - m(0) &= m\left(-\frac{\Delta g}{\|g\|}\right) - f \\
 &= -\frac{\Delta}{\|g\|} \|g\|^2 + \frac{1}{2} \frac{\Delta^2}{\|g\|^2} g^T B g \quad \xrightarrow{\text{substitute }} m(p^c) \\
 &\leq -\Delta \|g\| \quad \xrightarrow{\text{if } g^T B g < 0} \\
 &\leq -\|g\| \min\left(\Delta, \frac{\|g\|}{\|B\|}\right) \leq -\frac{1}{2} \|g\| \min\left(\Delta, \frac{\|g\|}{\|B\|}\right)
 \end{aligned}$$

Assume
 $g^T B g > 0$

Let $g^T B g > 0$ and $\frac{\|g\|^3}{\Delta(g^T B g)} \leq 1$. Then, from

the definition of τ_k in the Cauchy point, we have

$$\tau = \frac{\|g\|^3}{\Delta(g^T B g)} \quad \text{and} \quad p_k^c = -\tau \frac{\Delta}{\|g\|} g. \quad \text{Then}$$

$$\begin{aligned}
 m(p^c) - m(0) &= -\frac{\|g\|^4}{g^T B g} + \frac{1}{2} g^T B g \frac{\|g\|^4}{(g^T B g)^2} \\
 &= -\frac{1}{2} \frac{\|g\|^4}{g^T B g} \quad \lesssim -\frac{1}{2} \frac{\|g\|^4}{\|B\| - \|\rho\|^2} = -\frac{1}{2} \frac{\|g\|^2}{\|B\|} \\
 &\quad \text{CAUCHY} \\
 &\quad \text{triangle} \\
 &\leq -\frac{1}{2} \|g\| \min\left(\Delta, \frac{\|g\|}{\|B\|}\right)
 \end{aligned}$$

Let $g^T B g > 0$ and $\frac{\|g\|^3}{\Delta(g^T B g)} > 1 \quad (\equiv g^T B g < \frac{\|g\|^3}{\Delta})$

Then $\tau = 1$ and we have $\Delta(g^T B g) > \frac{\|g\|^3}{\Delta}$

$$\begin{aligned}
 m(p^c) - m(0) &= -\frac{\Delta}{\|g\|} \|g\|^2 + \frac{1}{2} \frac{\Delta^2}{\|g\|^2} g^T B g \leq -\Delta \|g\| + \frac{1}{2} \frac{\Delta^2}{\|g\|^2} \frac{\|g\|^3}{\Delta} \\
 &= -\frac{1}{2} \Delta \|g\| \leq -\frac{1}{2} \|g\| \min\left(\Delta, \frac{\|g\|}{\|B\|}\right)
 \end{aligned}$$

D

(91)

Thm 4.4: Let p_k be any vector & $\|p_k\| \leq \Delta_k$ and

$$m_k(o) - m_k(p_k) \geq c_2 (m_k(o) - m_k(p_k^c)). \text{ Then}$$

$$m_k(o) - m_k(p_k) \geq \frac{c_2}{2} \|g_k\| \min(\Delta_k, \frac{\|g_k\|}{\|B_k\|}).$$

In particular, if p_k is the exact solution p_k^* of the trust-region problem

$$\min_{p \in \mathbb{R}^n} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \text{ s.t. } \|p\| \leq \Delta_k$$

then it satisfies $m_k(o) - m_k(p_k) \geq \frac{1}{2} \|g_k\| \min(\Delta_k, \frac{\|g_k\|}{\|B_k\|})$.

Proof: Since $\|p_k\| \leq \Delta_k$, from Lemma 4.3, we have.

$$\begin{aligned} m_k(o) - m_k(p_k) &\geq c_2 (m_k(o) - m_k(p_k^c)) \\ &\geq \frac{1}{2} c_2 \|g_k\| \min(\Delta_k, \frac{\|g_k\|}{\|B_k\|}) \end{aligned} \quad \square$$

Note that the dogleg method and its two-dimensional subspace minimization extension both satisfy

$$m_k(o) - m_k(p_k) \geq \frac{1}{2} \|g_k\| \min(\Delta_k, \frac{\|g_k\|}{\|B_k\|})$$

since they both provide approximate solutions for which we have $m_k(p_k) \leq m_k(p_k^c)$.

Convergence to Stationary Points

We study two cases: in Alg 4.1 (i) $\gamma=0$, (ii) $\gamma>0$.
 $\gamma=0 \Leftrightarrow$ no actual decrease in f is acceptable; $\gamma>0$ ^{actual decrease must} be positive.

(92)

We assume that the approximate Hessians $\{B_k\}$ are uniformly bounded in norm and f is bounded below on the level set $S \triangleq \{x \mid f(x) \leq f(x_0)\}$. Let

$$S(R_0) \triangleq \{x \mid \|x - y\| < R_0 \text{ for some } y \in S\}$$

be an open neighborhood of S where $R_0 > 0$.

To increase generality, we allow $\|\rho_k\| \leq \gamma \Delta_k$ for some $\gamma \geq 1$.

Thm 4.5: Let $\eta=0$ in Alg 4.1. Suppose that $\|B_k\| \leq \beta$ and f is bounded below on $S \triangleq \{x \mid f(x) \leq f(x_0)\}$ and Lipschitz continuously differentiable in $S(R_0)$ for some $R_0 > 0$, and that all approximate solutions of

$$\min_{p \in \mathbb{R}} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \quad \text{s.t. } \|\rho\| \leq \Delta_k$$

$$\text{satisfy } m_k(0) - m_k(\rho_k) \geq c_1 \|g_k\| \min(\Delta_k, \frac{\|\rho_k\|}{\|B_k\|})$$

and $\|\rho_k\| \leq \gamma \Delta_k$ for some c_1^* and γ^* . Then

$$\liminf_{k \rightarrow \infty} \|g_k\| = 0$$

(i.e. the sequence of gradients has a limit point at 0).

Proof: See page 80-82 in the book.

Note: Since with $\eta=0$, we did not enforce f to decrease strictly at each step, it is possible that the sequence $\{g_k\}$ has multiple limit points, one of which is 0.

Thm 4.6. Let $\gamma \in (0, \frac{1}{4})$ in Alg. 4.1. Suppose that $\|B_k\| \leq \beta$, f is bounded below on S and is Lipschitz continuously differentiable in $S(R_0)$ for some $R_0 > 0$, and that all approximate solutions p_k of the trust-region problem with quadratic $M_k(\cdot)$ and Euclidean-ball trust-region constraint satisfy the two inequalities in Thm 4.5. Then

$$\lim_{k \rightarrow \infty} g_k = 0$$

Proof: See pages 82-83 in the book.

Iterative Solution of the Subproblem

Recall from Thm 4.1 that the optimal solution to the trust-region problem satisfies $(B + \lambda I)p^* = -g$ for some $\lambda \geq 0$. We wish to find the/a λ which matches a given Δ , trust-region radius. The procedures we discussed above used cost-effective approximate solutions that exhibit nice convergence properties.

From Thm 4.1, we see that either $\lambda = 0$ with $\|p\| \leq \Delta$, or we define $p(\lambda) = -(B + \lambda I)^{-1}g$ for some sufficiently large λ such that $B + \lambda I > 0$ and $\|p(\lambda)\| = \Delta$. The last problem, $\|p(\lambda)\| = \Delta$, is a one-dimensional root-finding problem.

(94)

Since B is symmetric, $B = Q\Lambda Q^T$ where Λ has diagonal entries $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ and Q is orthogonal. Then $(B + \lambda I) = Q(\Lambda + \lambda I)Q^T$ and for $\lambda \neq \lambda_j$, we have

$$p(\lambda) = -Q(\Lambda + \lambda I)^{-1}Q^T g = -\sum_{j=1}^n \frac{q_j^T g}{\lambda_j + \lambda} q_j$$

where $Q = [q_1 \dots q_j \dots q_n]$ is the columnwise expression.

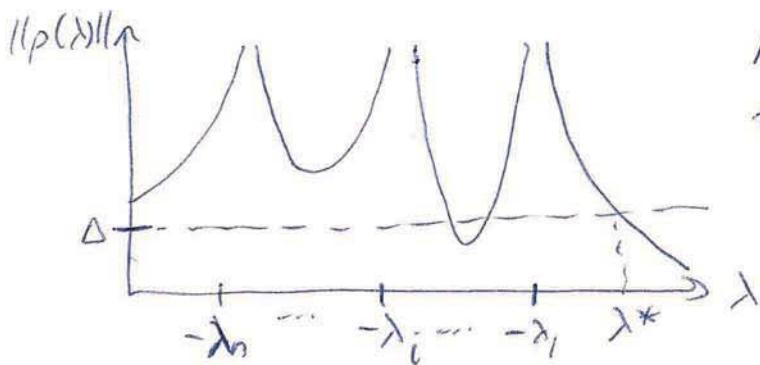
Since $Q^T Q = Q Q^T = I$, we have

$$\|p(\lambda)\|^2 = \sum_{j=1}^n \frac{(q_j^T g)^2}{(\lambda_j + \lambda)^2}$$

If $\lambda > -\lambda_1$, we have $\lambda_j + \lambda > 0 \quad \forall j \in \{1, 2, \dots, n\}$, so $\|p(\lambda)\|$ is a continuous, nonincreasing function of λ on $(-\lambda_1, \infty)$. In fact, we have $\lim_{\lambda \rightarrow \infty} \|p(\lambda)\| = 0$.

If ~~for $\lambda < -\lambda_1$~~ we have $q_j^T g \neq 0$, then $\lim_{\lambda \rightarrow -\lambda_1^-} \|p(\lambda)\| = \infty$.

So, an example profile of $\|p(\lambda)\|$ looks like



Now we can develop a procedure to find λ^* for which $\|p(\lambda^*)\| = \Delta$ when $q_1^T g \neq 0$.

(95)

Assume $q_1^T g \neq 0$. Note that when $B > 0$ and $\|B^{-1}g\| \leq \Delta$, $\lambda = 0$ satisfies the conditions of Thm 4.1. In this case $\lambda^* = 0$. Otherwise, we can use Newton's root-finding algorithm to find $\phi_1(\lambda) = \|p(\lambda)\| - \Delta = 0$ for $\lambda > -\lambda_1$. Notice that when $\lambda > -\lambda_1$ but very close to $-\lambda_1$, then $\phi_1(\lambda) \approx \frac{c_1}{\lambda + \lambda_1} + c_2$ with $c_1 > 0$. In this case, the Newton algorithm may be very slow since around λ^* the function is highly nonlinear.

$$\text{Now consider } \phi_2(\lambda) = \frac{1}{\Delta} - \frac{1}{\|p(\lambda)\|} \approx \frac{1}{\Delta} - \frac{\lambda + \lambda_1}{c_3}$$

for some $c_3 > 0$. Since ϕ_2 is nearly linear near $-\lambda_1$, Newton's root finding method will perform well.

$$\lambda^{(e+1)} = \lambda^{(e)} - \frac{\phi_2(\lambda^{(e)})}{\phi_2'(\lambda^{(e)})}$$

Alg. 4.3 Trust Region Subproblem

Given $\lambda^{(e)}$ and $\Delta > 0$

for $e = 0, 1, 2, \dots$

$$\text{Factor } B + \lambda^{(e)} I = R^T R$$

$$\text{Solve } R^T R P_e = -g ; R^T q_e = P_e$$

$$\text{Set } \lambda^{(e+1)} = \lambda^{(e)} + \left(\frac{\|P_e\|}{\|q_e\|} \right)^2 \left(\frac{\|P_e\| - \Delta}{\Delta} \right)$$

end (for)

Cholesky decomposition
here assumes $\lambda^{(e)} > -\lambda_1$.
For repeated eigenvalues,
thus does not work.
(see pg 87).

Convergence of Algorithms based on Nearly Exact Solutions

Alg 4.3 or its variations will only yield an approximate solution after a finite number of iterations. In the Newton root-finding process, we can require that

$$(*) \quad m(0) - m(p) \geq c, \quad (m(s) - m(p^*)) \cdot c, \quad c, \epsilon(0, 1] \\ \|p\| \leq \gamma D \quad \gamma > 0$$

so that convergence results of Thm 4.5 & 4.6 can be employed.

Thm 4.8: Suppose that the assumptions of Thm 4.6 are satisfied and in addition that f is twice cont. diff. in S . Suppose $B_k = \nabla^2 f(x_k)$ $\forall k$ and the approximate solution p_k of the first-order problem satisfies the two conditions (*) above for some $\gamma > 0$. Then

$\lim_{k \rightarrow \infty} \|g_k\| = 0$ (i.e. all limit points are stationary).

If, in addition, S is compact (closed and bounded), then either the algorithm terminates at a point x_k at which the 2^{nd} -order ^{local} necessary conditions (Thm 2.3) hold, or $\{x_k\}$ has a limit point $x^* \in S$ at which the 2^{nd} order necessary conditions hold.

Local Convergence of Trust-Region Newton Methods

(97)

If we show that the trust-region bound eventually stops interfering with the solution, then Newton-like convergence can be achieved.

Thm 4.9: Let f be twice Lipschitz cont. diff. in a neighborhood of x^* at which 2nd-order sufficiency conditions (Thm 2.4) are satisfied. Suppose $\{x_k\}$ converges to x^* and that Δ_k sufficiently large, the trust-region algorithm based on the quadratic model with $B_k = \nabla^2 f(x_k)$ chooses steps p_k that satisfy the Cauchy-point-based model reduction criterion and are asymptotically similar to Newton steps p_k^N whenever $\|p_k^N\| \leq \frac{1}{2} \Delta_k$ (i.e. $\|p_k - p_k^N\| = o(\|p_k^N\|)$). Then the trust-region bound Δ_k becomes inactive $\forall k$ sufficiently large and $\{x_k\}_{k=0}^\infty \rightarrow x^*$ superlinearly.

Proof: See pg 93-94.

Scaling: Optimization problems may be posed with poor scaling. This condition exhibits itself as the fact that x^* lies in a narrow valley with highly eccentric elliptical contours. Since the model $m_k(\cdot)$ is assumed to be accurate within the trust region, one could consider elliptical trust-regions: $\|Dp\| \leq \Delta_k$ where $D \succ 0$ is diagonal. Then

$$\min_{p \in \mathbb{R}^n} m_k(p) \stackrel{?}{=} f_k + g_k^T p + \frac{1}{2} p^T B_k p \text{ s.t. } \|Dp\| \leq \Delta_k$$

The diagonal entries of D_k could be derived from the diagonals of the local Hessian at iteration k . We need to impose some predetermined bound $\{\delta_{lo}, \delta_{hi}\}$.

Alg 4.4 Generalized Cauchy Point Calculation

$$\min_{p \in \mathbb{R}^n} f_k + g_k^T p \text{ s.t. } \|Dp\| \leq \Delta_k \Rightarrow \text{Let } p_k^S \text{ be the solution.}$$

$$\text{Let } \tau_k = \underset{\tau > 0}{\operatorname{argmin}} \quad m_k(\tau p_k^S) \quad \text{s.t. } \|\tau D p_k^S\| \leq \Delta_k$$

$$p_k^C = \tau_k p_k^S$$

$$\therefore p_k^S = -\frac{\Delta_k}{\|D^T g_k\|} D^{-2} g_k, \quad \tau_k = \begin{cases} 1 & ; \frac{g_k^T D^{-2} B_k D^{-2} g_k}{\|D^T g_k\|^2} \leq 0 \\ \min\left(\frac{\|D^T g_k\|^2}{\Delta_k g_k^T D^{-2} B_k D^{-2} g_k}, 1\right) & \text{otherwise} \end{cases}$$

Alternatively, we could define $\tilde{p} \triangleq D_p$. Then (99)

$$\min_{\tilde{p} \in R} \tilde{m}_k(\tilde{p}) \triangleq f_k + g_k^T D^{-1} \tilde{p} + \frac{1}{2} \tilde{p}^T D^{-1} B_k D^{-1} \tilde{p} \text{ s.t. } \|\tilde{p}\| \leq \Delta_k$$

Now, we have a spherical trust-region for \tilde{p} as before. The theory derived earlier applies exactly.

Trust Regions in Other Norms

We could use other norms for the trust regions.

$$\|\tilde{p}\|_1 \leq \Delta_k \text{ or } \|\tilde{p}\|_\infty \leq \Delta_k \text{ (or } \|D\tilde{p}\|_1 \leq \Delta_k).$$

The L_1 and L_∞ norm trust regions might be useful for problems with linear inequality constrained problems

$$\text{e.g. } \min_{x \in \mathbb{R}^n} f(x) \text{ s.t. } x \geq 0$$

This way, the local trust-region problems also have linear inequality constraints.

Exercises

4.6) Show that $\gamma \triangleq \frac{\|g\|^4}{(g^T B g)(g^T B^{-1} g)} \leq 1$, equality only if $g \parallel Bg$ ($g \parallel B^{-1}g$).

$$\text{Let } u = B^{1/2}g, v = B^{-1/2}g. \text{ Then } \gamma = \frac{(u^T v)^2}{(u^T u)(v^T v)} = \cos^2 \theta_{uv} \leq 1.$$

4.10) For symmetric B , show that $\exists \lambda \geq 0 \Rightarrow (B + \lambda I) > 0$. Since B is symmetric, $B = Q \Lambda Q^T$ is its eigendecomposition.

$$B + \lambda I = Q \Lambda Q^T + \lambda I = Q \Lambda Q^T + \lambda Q Q^T = Q (\Lambda + \lambda I) Q^T > 0 \text{ for } \forall \lambda > 0,$$

where $\lambda_1 \leq \dots \leq \lambda_n$ are the eigenvalues of B if $\lambda_i \leq 0$, $\forall \lambda > 0$ if $\lambda > 0$.

Chapter 5: Conjugate Gradient Methods

Conjugate gradient methods are very useful for solving large linear systems of equations as well as for solving nonlinear optimization problems. Linear conjugate gradient method is designed for solving linear systems with positive definite coefficient matrices. Its performance depends on the eigenvalue distribution, so in practice preconditioning is used.

Nonlinear conjugate gradient methods were introduced for solving large scale nonlinear optimization problems and they don't require any matrix storage yet they are faster than steepest descent.

Linear Conjugate Gradient Method

Consider a system of linear equations: $Ax = b$, where $A \succ 0$ is $n \times n$ symmetric. An equivalent problem is

$$\min \phi(x) \triangleq \frac{1}{2} x^T A x - b^T x$$

$$\therefore \nabla \phi(x) = Ax - b \triangleq r(x). \text{ So at } x = x_k, r_k = Ax_k - b.$$

Defn: A set of nonzero vectors $\{p_0, p_1, \dots, p_k\}$

is said to be conjugate with respect to the symmetric positive definite matrix A iff

$$p_i^T A p_j = 0 \quad \forall i \neq j.$$

Fact: Any set of vectors satisfying the conjugacy property above is also linearly independent.
(Ex. Prove this)

Fact: We can minimize $\phi(\cdot)$ in n steps by successively minimizing it along the individual directions in a conjugate set.

Proof: Given a starting point $x_0 \in \mathbb{R}^n$ and a set of conjugate directions $\{p_0, p_1, \dots, p_{n-1}\}$, generate a sequence $\{x_k\}$ by $x_{k+1} = x_k + \alpha_k p_k$ where α_k is the one-dimensional minimizer of $\phi(\cdot)$ along $x_k + \alpha p_k$. Specifically, $\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k}$.

Then, we have the following result.

Thm 5.1 For any $x_0 \in \mathbb{R}^n$, the sequence $\{x_k\}$ generated by the conjugate direction algorithm given above converges to the solution x^* of the linear system $Ax=b$ in at most n steps.

Proof: Since $\{p_i\}$ are conjugate, they are also lin. indep. Then they must span the whole space \mathbb{R}^n .

Consequently, $\exists (\sigma_0, \sigma_1, \dots, \sigma_{n-1}) \in \mathbb{R}^n$

$$x^* - x_0 = \sigma_0 p_0 + \sigma_1 p_1 + \dots + \sigma_{n-1} p_{n-1}$$

Premultiplying by $p_k^T A$ and using $p_i^T A p_j = 0 \forall i \neq j$, we get

$$\frac{p_k^T A (x^* - x_0)}{p_k^T A p_k} = \sigma_k, \text{ we now need}$$

to show that $\sigma_k = \alpha_k$. From the construction of x_k , we see that

$$x_k = x_0 + \alpha_0 p_0 + \dots + \alpha_{k-1} p_{k-1}$$

Premultiplying by $p_k^T A$ and using conjugacy:

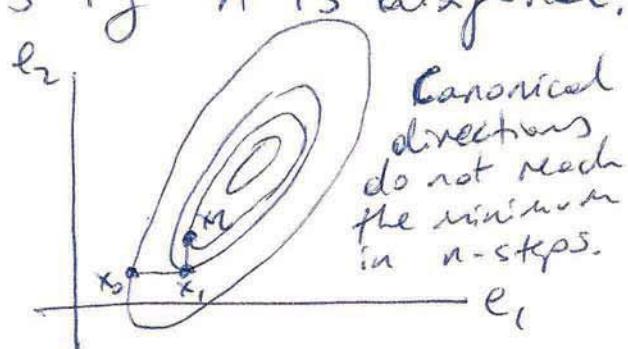
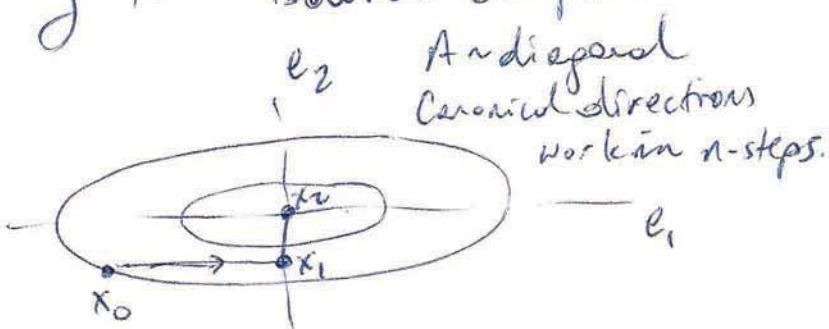
$$p_k^T A (x_k - x_0) = 0$$

$$\text{Therefore, } p_k^T A (\underbrace{x^* - x_0}_{+ x_k - x_k}) = p_k^T A (\overbrace{x^* - x_k}^{+ x_k - x_k}) = p_k^T (b - Ax_k) = -p_k^T r_k. \\ \therefore \sigma_k = \alpha_k \quad \square$$

* w.r.t. = with respect to

(103)

The conjugate directions correspond to axis directions of the isolovel elliptical contours if A is diagonal.



In the case when A is not diagonal, we can transform the problem (rotate x) such that the matrix in the new problem ~~too~~ is diagonal.

Let $\hat{x} = S^{-1}x$ where $S \triangleq [P_0 \ P_1 \ \dots \ P_{n-1}]$ and the set $\{P_0, \dots, P_{n-1}\}$ is conjugate w.r.t. A . The quadratic ϕ becomes

$$\hat{\phi}(\hat{x}) \triangleq \phi(S\hat{x}) = \frac{1}{2} \hat{x}^T (S^T A S) \hat{x} - (S^T b)^T \hat{x}$$

By the conjugacy of the columns of S , $S^T A S$ is diagonal. Therefore, we can find the minimizer of $\hat{\phi}$ by performing $n-1$ D minimizations along the canonical coordinate directions of \hat{x} . Consequently, canonical direction update in \hat{e}_i yields an update in the ϕ direction of P_{i-1} in the x -space.

(104)

Note that if the Hessian matrix of a quadratic criterion is diagonal, 1D minimization along each coordinate identifies one component of the solution. A similar conclusion can be drawn for general Hessians.

Thm 5.2: Expanding Subspace Minimization

Let $x_0 \in \mathbb{R}^n$ be any starting point and suppose that $\{x_k\}$ is generated by the conjugate direction algorithm ($x_{k+1} = x_k + \alpha_k p_k$; $\alpha_k = -(\gamma_k^T p_k) / (p_k^T A p_k)$). Then $\gamma_k^T p_i = 0$ for $i=0, 1, \dots, (k-1)$ and x_k is the minimizer of $\phi(x) = \frac{1}{2} x^T A x - b^T x$ over the set

$$S_{k-1}^{\Delta} \triangleq \left\{ x \mid x = x_0 + \text{span}\{p_0, \dots, p_{k-1}\} \right\}$$

Proof: We begin by showing that \tilde{x} minimizes ϕ over the set S_{k-1} iff $r^T(\tilde{x}) p_i = 0 \quad \forall i \in \{0, \dots, k-1\}$.

Let $h(\sigma) = \phi(x_0 + \sigma_0 p_0 + \dots + \sigma_{k-1} p_{k-1})$ where $\sigma = (\sigma_0, \dots, \sigma_{k-1})^T$.

Since $h(\sigma)$ is a strictly convex quadratic, it has a unique minimizer σ^* at which $\frac{\partial h(\sigma^*)}{\partial \sigma_i} = 0, i=0, \dots, k-1$. By the chain rule, this implies

$$\nabla \phi(x_0 + \sigma_0^* p_0 + \dots + \sigma_{k-1}^* p_{k-1})^T p_i = 0, \quad i=0, 1, \dots, k-1.$$

Recall that $r(\tilde{x}) = \nabla \phi(\tilde{x})$, so $r(\tilde{x})^T p_i = 0$. (105)

Now we use induction. Consider $k=1$. we have

$x_1 = x_0 + \alpha_0 p_0$ which minimizes ϕ along p_0 (i.e. $r_1^T p_0 = 0$).

Assume that we have $r_{k-1}^T p_i = 0$ for $i = 0, 1, \dots, (k-2)$.

We know that $r_k = r_{k-1} + \alpha_{k-1} A p_{k-1}$, so

$$p_{k-1}^T r_k = p_{k-1}^T r_{k-1} + \alpha_{k-1} p_{k-1}^T A p_{k-1} = 0$$

by the definition of α_{k-1} . Also, for other

p_i , $i = 0, 1, \dots, k-2$, we have

$$p_i^T r_k = p_i^T r_{k-1} + \alpha_{k-1} p_i^T A p_{k-1} = 0$$

where $p_i^T r_{k-1} = 0$ due to our assumption that $r_{k-1}^T p_i = 0$,
and the conjugacy of $\{p_i\}_{i=0}^{k-1}$. Thus, we have $r_k^T p_i = 0$
for $i = 0, 1, \dots, k-1$. \square

Note The current residual r_k at each step is orthogonal
to all previous search directions.

So far, we considered an arbitrary conjugate
set $\{p_0, \dots, p_{n-1}\}$. For large scale problems, we
~~could~~ generate such a set during the iterations using
Gram-Schmidt orthogonalization. However, the GS procedure
also requires storing the entire direction set.

The Conjugate Gradient Method

In this method, a conjugate direction p_k^{new} can be computed only using p_{k-1} , so it requires little storage and computation.

$$p_k = -r_k + \beta_k p_{k-1}$$

We choose β_k & p_k and p_{k-1} are conjugate w.r.t. A.

$$\Rightarrow \beta_k = \frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}$$

The initial direction is selected as $p_0 = -\nabla f(x_0)$.

Alg. 5.1 (CG Preliminary Version)

Given x_0 , set $r_0 \leftarrow A x_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$.
while $r_k \neq 0$

$$\alpha_k \leftarrow - (r_k^T p_k) / (p_k^T A p_k)$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k$$

$$r_{k+1} \leftarrow A x_{k+1} - b$$

$$\beta_{k+1} \leftarrow (r_{k+1}^T A p_k) / (p_k^T A p_k)$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$$

$$k \leftarrow k+1$$

end (while)

(107)

If we can show that $\{p_0, \dots, p_{n-1}\}$ generated by the CG procedure is indeed conjugate wrt A then by Thm 5.1, we know that Alg 5.1 will terminate in n steps.

Defn: The Krylov subspace of degree k for r_0 is defined as $K(r_0; k) \triangleq \text{span}\{r_0, Ar_0, \dots, A^k r_0\}$.

Thm 5.3 Suppose that the k^{th} iterate generated by the conjugate gradient method is not the solution point x^* . The following properties hold:

- 1) $r_k^T r_i = 0$ for $i = 0, 1, \dots, k-1$
- 2) $\text{span}\{r_0, r_1, \dots, r_k\} = \text{span}\{r_0, Ar_0, \dots, A^k r_0\}$
- 3) $\text{span}\{p_0, p_1, \dots, p_k\} = \text{span}\{r_0, Ar_0, \dots, A^k r_0\}$
- 4) $p_k^T A p_i = 0$ for $i = 0, 1, \dots, k-1$

Therefore, the sequence $\{x_k\}$ converges to x^* in at most n steps.

Proof: See pg 109-111 in the book.

Note: This theorem relies on the fact that p_0 is the steepest descent direction. The result does not hold for other choices. The gradients are \perp to each other ($r_i^T r_j = 0$); i.e. the search directions that are conjugate wrt A.

Practical CG Method.

The algorithm efficiency could be improved. Note that since $r_k^T p_i = 0$ for $i = 0, \dots, k-1$ (Thm 5.2) and $p_{k+1} = -r_{k+1} + \beta_{k+1} p_k$ (Alg 5.1), we get

$$\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k} = -\frac{r_k^T (-r_{k+1} + \beta_{k+1} p_k)}{p_k^T A p_k} = \frac{r_k^T r_{k+1}}{p_k^T A p_k}$$

We also had $r_{k+1} = r_k + \alpha_k A p_k$, so

$$\beta_{k+1} = \frac{r_{k+1}^T A p_k}{p_k^T A p_k} = \frac{r_{k+1}^T (r_{k+1} - r_k)}{\alpha_k (p_k^T A p_k)} = \frac{r_{k+1}^T r_{k+1} - r_{k+1}^T r_k}{r_k^T r_k} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$$

Alg 5.2 CG-algorithm

Given x_0 , set $r_0 \leftarrow Ax_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$

while $r_k \neq 0$

$$\alpha_k \leftarrow (r_k^T r_k) / (p_k^T A p_k)$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k$$

$$r_{k+1} \leftarrow r_k + \alpha_k A p_k$$

$$\beta_{k+1} \leftarrow (r_{k+1}^T r_{k+1}) / (r_k^T r_k)$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$$

$$k \leftarrow k+1$$

end (while)

Rate of Convergence

The algorithm will converge in at most n iterations. If the eigenvalues of A are favorable, convergence could occur in fewer iterations. From Thm 5-3, we know

that $x_{k+1} = x_0 + \alpha_0 p_0 + \dots + \alpha_k p_k$
 $= x_0 + \gamma_0 r_0 + \dots + \gamma_k A^k r_0$

for some constants γ_i . Define

$$P_k^*(A) = \gamma_0 I + \gamma_1 A + \dots + \gamma_k A^k.$$

Then $x_{k+1} = x_0 + P_k^*(A)r_0$. For some $z \in \mathbb{R}^n$, let $\|z\|_A^2 \stackrel{\Delta}{=} z^T A z$ be the norm-squared of z . Using the quadratic ϕ , we had

$$\frac{1}{2} \|x - x^*\|_A^2 = \frac{1}{2} (x - x^*)^T A (x - x^*) = \phi(x) - \phi(x^*)$$

From Thm 5-2, we know that x_{k+1} minimizes ϕ , hence $\|x - x^*\|_A^2$ over the set $x_0 + \text{span}\{p_0, p_1, \dots, p_k\}$, which is the same as $x_0 + \text{span}\{r_0, Ar_0, \dots, A^k r_0\}$ by Thm 5-3.

Consequently, the polynomial $P_k^*(.)$ solves

$$\min_{P_k} \|x_0 + P_k(A)r_0 - x^*\|_A$$

over the space of all possible polynomials of degree k .

Since $r_0 = Ax_0 - b = Ax_0 - Ax^* = A(x_0 - x^*)$, we have
 that $x_{k+1} - x^* = x_0 + P_k^*(A)r_0 - x^* = [I + P_k^*(A)](x_0 - x^*)$,

let $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ be the eigenvalues of A ,
 and let v_1, v_2, \dots, v_n be the corresponding orthonormal
 eigenvectors. Then $A = V \Lambda V^T = \sum_{i=1}^n \lambda_i v_i v_i^T$ and
 $x_0 - x^* = \sum_{i=1}^n \beta_i v_i$ for some coefficients β_i .

Note that, since v_i is an eigenvector of A , it
 is also an eigenvector of $P_k(A)$ for any polynomial P_k .

We have $P_k(A)v_i = P_k(\lambda_i)v_i$, $i=1, 2, \dots, n$, so

$$x_{k+1} - x^* = \sum_{i=1}^n [1 + \lambda_i P_k^*(\lambda_i)] \beta_i v_i.$$

Therefore, $\|x_{k+1} - x^*\|_A^2 = \sum_{i=1}^n \lambda_i [1 + \lambda_i P_k^*(\lambda_i)]^2 \beta_i^2$. Since
 P_k^* generated by the CG method is optimal with respect
 to this norm, we have

$$\|x_{k+1} - x^*\|_A^2 = \min_{P_k} \sum_i \lambda_i [1 + \lambda_i P_k(\lambda_i)]^2 \beta_i^2.$$

Let $\gamma \stackrel{\Delta}{=} \max_{1 \leq i \leq n} [1 + \lambda_i P_k(\lambda_i)]^2$. Then

$$\|x_{k+1} - x^*\|_A^2 \leq \min_{P_k} \gamma_{P_k} \left(\sum_{j=1}^n \lambda_j \beta_j^2 \right)$$

$$= \min_{P_k} \gamma_{P_k} \|x_0 - x^*\|_A^2$$

$$\text{since } \|x_0 - x^*\|_A^2 = \sum_j \lambda_j \beta_j^2.$$

So in the CG method, the convergence is governed by $\min_{P_k} \delta_{P_k}$; i.e. we search for a polynomial P_k that makes this expression as small as possible.

Theorem 5.4 If A has only r distinct eigenvalues, then the CG iteration will terminate at the solution in at most r iterations.

(skip if needed) Proof: Suppose that the eigenvalues $\lambda_1, \dots, \lambda_n$ take r distinct values $\tau_1 < \tau_2 < \dots < \tau_r$. Define the polynomial

$$Q_r(\lambda) = \frac{(-1)^r}{\tau_1 \cdots \tau_r} (\lambda - \tau_1)(\lambda - \tau_2) \cdots (\lambda - \tau_r)$$

Note that $Q_r(\tau_i) = 0$ for $i=1, 2, \dots, r$ and $Q_r(0) = 1$. Therefore, $Q_r(\lambda) - 1$ is a polynomial of degree r with a root at $\lambda = 0$. Then $\bar{P}_{r-1}(\lambda) \triangleq (Q_r(\lambda) - 1)/\lambda$ is a polynomial of degree $r-1$. Let $k=r-1$. Then

$$\begin{aligned} 0 &\leq \min_{P_{r-1}} \max_{1 \leq i \leq n} \left\{ 1 + \lambda_i P_{r-1}(\lambda_i) \right\}^2 \leq \max_{1 \leq i \leq n} \left\{ 1 + \lambda_i \bar{P}_{r-1}(\lambda_i) \right\}^2 \\ &\quad = \max_{1 \leq i \leq n} Q_r^2(\lambda_i) = 0 \end{aligned}$$

$$\therefore \|x_r - x^*\|_A^2 = 0, \text{ hence } x_r = x^*. \quad \square$$

Theorem 5.5 If A has eigenvalues $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$, then

$$\|x_{k+1} - x^*\|_A^2 \leq \left(\frac{\lambda_{n-k} - \lambda_1}{\lambda_{n-k} + \lambda_1} \right)^2 \|x_0 - x^*\|_A^2.$$

Fact. Another bound that characterizes the convergence of the CG method is

$$\|x_k - x^*\|_A \leq 2 \left(\frac{\sqrt{K(A)} - 1}{\sqrt{K(A)} + 1} \right)^k \|x_0 - x^*\|_A$$

where $K(A) = \|A\|_2 \cdot \|A^{-1}\|_2 = \lambda_n / \lambda_1$.

Note Thm 5.5 indicates that if the eigenvalues of A are clustered into r clusters, then CG iterates will solve the problem approximately in r steps (by selecting one eigenvalue from each cluster).

Preconditioning

CG method can be accelerated by improving the eigenvalue distribution of A . Consider a change of variables: $\hat{x} = Cx$. Then $\hat{\phi}(\hat{x}) = \frac{1}{2} \hat{x}^T (C^{-T} A C^{-1}) \hat{x} - (C^{-T} b)^T \hat{x}$. Now the solution is: $(C^{-T} A C^{-1}) \hat{x} = C^{-T} b$. The eigenvalues of $C^{-T} A C^{-1}$ control the convergence rate. Using the symmetric and positive definite matrix $M = C^T C$, an improved CG algorithm can be designed.

Here, C can be chosen to make the eigenvalue spread of $C^{-T} A C^{-1}$ clustered or to make the cond. number $K(C^T A C^{-1})$ more favorable than that of A .

Alg. 5.3 Preconditioned CG

113

Given x_0 and $M = C^T C$.

Set $r_0 \leftarrow Ax_0 - b$

Solve $My_0 = r_0$ for y_0

Set $p_0 \leftarrow -y_0$ and $k \leftarrow 0$

while $r_k \neq 0$

$$\alpha_k \leftarrow (r_k^T y_k) / (p_k^T A p_k)$$

$$x_{k+1} \leftarrow x_k + \alpha_k p_k$$

$$r_{k+1} \leftarrow r_k + \alpha_k A p_k$$

$$\text{Solve } M y_{k+1} = r_{k+1}$$

$$\beta_{k+1} \leftarrow (r_{k+1}^T y_{k+1}) / (r_k^T y_k)$$

$$p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$$

$$k \leftarrow k+1$$

end(while)

Notes

* For $M = I$, we get the standard CG method of Alg 5.2.

* Here, $r_i^T M^{-1} r_j = 0$ if $i \neq j$

- * Selection of preconditioners: There does not exist a one-fits-all type solution. See pg 120 in the book for a discussion. One interesting approach is to make C ~~upper triangular~~^($C = L^T$) so that $M = LL^T$ is a valid Cholesky decomposition and $C^{-T}AC^{-1} = L^{-1}A L^{-T} \approx I$.

Nonlinear Conjugate Gradient Methods

Q: How can we extend CG to $\min_x f(x)$?

The Fletcher-Reeves Method: Since the analytical solution for the optimal step length α_k is not known, a line search must be performed. The residual r_k must be replaced by the gradient of f at x_k .

Alg. 5.4 FR Method

Given x_0 , evaluate $f_0 = f(x_0)$, $\nabla f_0 = \nabla f(x_0)$.

Set $p_0 \leftarrow -\nabla f_0$, $k \leftarrow 0$.

while $\nabla f_k \neq 0$

Compute α_k and set $x_{k+1} = x_k + \alpha_k p_k$

Evaluate ∇f_{k+1}

$$\beta_{k+1}^{FR} \leftarrow \frac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}$$

$$p_{k+1} = -\nabla f_{k+1} + \beta_{k+1}^{FR} p_k$$

$$k \leftarrow k+1$$

Does this yield
a descent
direction?

end (while)

Consider $p_k = -\nabla f_k + \beta_k^{FR} p_{k-1}$ and multiply from left with ∇f_k^T :

$$\nabla f_k^T P_k = -\|\nabla f_k\|^2 + \beta_k^{FR} \nabla f_k^T P_{k-1}$$

If α_{k-1} is a local minimizer of f along P_{k-1} , we have $\nabla f_k^T P_{k-1} = 0$. In this case, $\nabla f_k^T P_k = -\|\nabla f_k\|^2 < 0$ so P_k will be a descent direction. If the line search for α_{k-1} is not exact, however, then the second term might dominate the first, leading to a P_k which is an ascent direction.

To prevent this latter situation, we can use the strong Wolfe conditions in the search.

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k$$

$$|\nabla f(x_k + \alpha_k p_k)^T p_k| \leq -c_2 \nabla f_k^T p_k$$

where $0 < c_1 < c_2 < \frac{1}{2}$ (note that we impose $\frac{1}{2}$ as the bound).

Lemma 5.6 Suppose that Alg 5.4 is implemented with a step length α_k that satisfies the strong Wolfe conditions with $0 < c_1 < c_2 < \frac{1}{2}$. Then, the method generates descent directions p_k that satisfy

$$-\frac{1}{(1-c_1)} \leq \frac{\nabla f_k^T p_k}{\|\nabla f_k\|^2} \leq \frac{(2c_2-1)}{(1-c_2)} \quad \forall k=0, \dots$$

Proof: See pg 125-126 in the book.

Now consider $\cos \theta_k = \frac{-\nabla f_k^T P_k}{\|\nabla f_k\| \cdot \|P_k\|}$. Suppose that P_k is a poor search direction (i.e. P_k is a descent direction but is almost orthogonal to $-\nabla f_k$). Then $\cos \theta_k \approx 0$. Multiplying both sides of the bound in Lemma 5.6 by $\frac{\|\nabla f_k\|}{\|P_k\|}$ and using $\cos \theta_k$, we have

$$\frac{1-2c_2}{1-c_2} \frac{\|\nabla f_k\|}{\|P_k\|} \leq \cos \theta_k \leq \frac{1}{1-c_2} \frac{\|\nabla f_k\|}{\|P_k\|} \quad \forall k=0, 1, \dots$$

Clearly, $\cos \theta_k \approx 0$ iff $\|\nabla f_k\| \ll \|P_k\|$. Since P_k is almost orthogonal to $-\nabla f_k$, very likely $x_{k+1} \approx x_k$ since line search for α_k will not allow a large step.

Then, very likely we have $\nabla f_{k+1} \approx \nabla f_k$, so $\beta_{k+1}^{FR} \approx 1$.

Then, since $\|\nabla f_{k+1}\| \approx \|\nabla f_k\| \ll \|P_k\|$, we have $P_{k+1} \approx P_k$.

So if P_k is a poor direction, we could perhaps switch to using $-\nabla f_{k+1}$ for the next search direction to prevent this problem. The following variation achieves this (approximately) automatically by redefining the value of β_{k+1} .

The Polak-Ribière Method and Variants

Let $\beta_{k+1}^{PR} = \frac{\nabla f_{k+1}^T (\nabla f_{k+1} - \nabla f_k)}{\|\nabla f_k\|^2}$, Replace β_{k+1}^{FR}

in the algorithm with this selection. When the gradients are mutually orthogonal (in the quadratic case) and line search is exact, $\beta_{k+1}^{PR} = \beta_{k+1}^{FR}$. When applied to nonlinear problems with inexact line search, PR algorithm is more robust compared to FR.

If we define $\beta_{k+1}^+ = \max(\beta_{k+1}^{PR}, 0)$, then we obtain the PR+ algorithm. In this case the strong Wolfe conditions as modified in Lemma 5.6 will ensure that each search is in a descent direction.

The Hestenes-Stiefel (HS) formula leads to the HS algorithm and its performance is similar to PR.

$$\beta_{k+1}^{HS} = \frac{\nabla f_{k+1}^T (\nabla f_{k+1} - \nabla f_k)}{(\nabla f_{k+1} - \nabla f_k)^T P_k}$$

We will see that any β_k with $|\beta_k| \leq \beta_k^{FR} + k^{-2}$ will guarantee global convergence.

If this is the case, we could use

$$\beta_k = \begin{cases} -\beta_k^{FR} & \text{if } \beta_k^{PR} < -\beta_k^{FR} \\ \beta_k^{PR} & \text{if } |\beta_k^{PR}| \leq \beta_k^{FR} \\ \beta_k^{FR} & \text{if } \beta_k^{PR} > \beta_k^{FR} \end{cases}$$

This yields the FR-PR algorithm.

Alternatively $\beta_{k+1} = \frac{\|\nabla f_{k+1}\|^2}{(\nabla f_{k+1}^T \nabla f_k) P_k}$ and

$$\beta_{k+1} = \left(\hat{y}_k - \gamma_{k+1} \frac{\|\hat{y}_k\|^2}{\hat{y}_k^T P_k} \right)^T \frac{\nabla f_{k+1}}{\hat{y}_k^T P_k} \quad \text{with } \hat{y}_k = \nabla f_{k+1} - \nabla f_k$$

are good choices that guarantee descent directions P_k provided that modified strong Wolfe conditions are observed in line search.

One variation is to reset $\beta_k = 0$ after every n steps in nonlinear optimization, thus reverting to steepest descent at these iterations. For this case,

one could prove that $\|\nabla f_{k+n} - \nabla f_k\| = O(\|\nabla f_k - \nabla f^*\|^2)$, that is, it achieves n -step quadratic convergence.

Another restart strategy is to reset the algorithm when two consecutive gradients are far from orthogonal:

$$|\nabla f_k^T \nabla f_{k-1}| / \|\nabla f_k\|^2 \geq \nu \quad (\text{e.g. } \nu=0.1).$$

Global Convergence

Assumptions 5-1:

- (i) The level set $\mathcal{L} = \{x \mid f(x) \leq f(x_0)\}$ is bounded.
- (ii) In some open neighborhood N of \mathcal{L} , f is Lipschitz continuously differentiable.

These assumptions imply that $\exists \bar{\gamma} \neq 0$ s.t. $\|\nabla f(x)\| \leq \bar{\gamma} \forall x \in \mathcal{L}$. Recall Thm 3.2 (Zoutendijk's Thm). Any line search iteration $x_{k+1} = x_k + \alpha_k p_k$ where p_k is a descent direction and α_k satisfies the Wolfe conditions gives

$$\sum_{k=0}^{\infty} \cos^2 \theta_k \|\nabla f_k\|^2 < \infty$$

This can be used to prove global convergence for algorithms that periodically reset $\beta_k = 0$. If k_1, k_2, \dots are the iterations where resets occur, then $\sum_{k=k_1, k_2, \dots} \|\nabla f_k\|^2 < \infty$.

If we don't allow more than \bar{n} iterations between restarts, $\{k_j\}_{j=1}^{\infty}$ is infinite and we have $\lim_{j \rightarrow \infty} \|\nabla f_{k_j}\| = 0$. Due to the convergence of this subsequence, we have

$$\liminf_{k \rightarrow \infty} \|\nabla f_k\| = 0.$$

What about CG without restarts?

Theorem 5.7 (Al-Baali)

Suppose that Assumptions 5.1 hold and Alg 5.4(FR) is implemented with a line search satisfying Lemma 5.6. Then $\lim_{k \rightarrow \infty} \| \nabla f_k \| = 0$.

Proof: See pg 128-130 in the book.

This result can be extended to any choice of β_k satisfying $|\beta_k| \leq \beta_k^{FR}$.

means
the first positive
of stationary
point.

Theorem 5.8 Consider the PR method with an ideal line search. If a twice cont. diff. $f: \mathbb{R}^3 \rightarrow \mathbb{R}$ with a starting point of $x_0 \in \mathbb{R}^3$ such that $\{\| \nabla f_k \| \}$ is bounded away from zero.

Note that this means, it is possible to find a case where the PR method can cycle infinitely away from a local solution. While unlikely, these kinds of behavior is possible.

Chapter 6: Quasi-Newton Methods

This line of research was initiated by a ground-breaking idea, which was not accepted as a paper - a story repeated many times in peer-reviewed science.

Davidon's ^{invention} discovery of the first quasi-Newton algorithm remained as a technical report for 30 years until 1991. Quasi-Newton methods use calculated gradients to build a model of the objective function to achieve superlinear convergence - beyond steepest descent.

Now we discuss quasi-Newton methods for small- and medium-sized problems.

The BFGS Method: (Broyden, Fletcher, Goldfarb, Shanno)

Consider the following quadratic model of $f(x)$ at x_k :

$$M_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p$$

Here $B_k > 0$ is symmetric. Clearly $M_k(0) = f_k$ and $\nabla M_k(0) = \nabla f_k$. Suppose that we generated a new iterate x_{k+1} and we construct a new model

$$M_{k+1}(p) = f_{k+1} + \nabla f_{k+1}^T p + \frac{1}{2} p^T B_{k+1} p.$$

First, we require that the gradient of $M_{k+1}(p)$ matches the gradient of f at x_k and x_{k+1} . We already have $\nabla M_{k+1}(0) = \nabla f_{k+1}$, so we require

$$\nabla M_{k+1}(-\alpha_k p_k) = \nabla f_{k+1} - \alpha_k B_{k+1} p_k = \nabla f_k$$

where $x_{k+1} = x_k + \alpha_k p_k$ with $p_k = -B_k^{-1} \nabla f_k$.

From this: $B_{k+1} \alpha_k p_k = \nabla f_{k+1} - \nabla f_k$.

Let $s_k \stackrel{\Delta}{=} x_{k+1} - x_k = \alpha_k p_k$, $y_k \stackrel{\Delta}{=} \nabla f_{k+1} - \nabla f_k$

$\therefore B_{k+1} s_k = y_k$ (this is called the secant equation). If α_k is selected to satisfy the Wolfe conditions, the curvature condition implies $\nabla f_{k+1}^T s_k \geq c_2 \nabla f_k^T s_k$, therefore $y_k^T s_k \geq (c_2 - 1) \alpha_k \nabla f_k^T p_k > 0$ since $c_2 < 1$ and $\nabla f_k^T p_k < 0$.

When $s_k^T y_k > 0$ (i.e. $s_k^T B_{k+1} s_k = s_k^T y_k > 0$) the secant equation has infinitely many solutions, otherwise it does not have a solution for B_{k+1} . We choose B_{k+1} to be the closest ^{symmetric} matrix to B_k while satisfying the secant condition.

$$\min_B \|B - B_k\| \text{ s.t. } B = B^T \text{ and } B s_k = y_k$$

where $B_k > 0$ is symmetric.

Different matrix norms lead to different quasi-Newton algorithms. For the weighted-Frobenius norm

$$\|A\|_F \triangleq \|W^{1/2} A W^{1/2}\|_F$$

where $W = \bar{G}_k^{-1}$ with $\bar{G}_k = \int_0^1 \nabla^2 f(x_k + \tau x_k p_k) d\tau$

denoting the average Hessian of f on the line segment

between x_k and x_{k+1} , we have $y_k = \bar{G}_k^{-1} x_k p_k = \bar{G}_k s_k$

from Taylor's theorem (Thm 2.1). With these choices

the unique solution is (proposed by Davidon, studied by Fletcher and Powell)

$$(DFP) \quad B_{k+1} = (I - \rho_k y_k s_k^T) B_k (I - \rho_k s_k y_k^T) + \rho_k y_k y_k^T$$

$$\text{where } \rho_k = \frac{1}{y_k^T s_k}$$

Using the Sherman-Morrison-Woodbury formula, $H_k = B_k^{-1}$ \square

$$(DFP) \quad H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}$$

\therefore in DFP, the inverse-Hessian model undergoes a rank-2 update.

Now consider the alternative problem of

$$\min_H \|H - H_k\| \quad \text{s.t. } H = H^T, \quad H y_k = s_k$$

(124)

Using the same weighted Frobenius norm, we get

$$(\text{BFGS}) \quad H_{k+1} = (I - p_k s_k y_k^T) H_k (I - p_k y_k s_k^T) + p_k s_k s_k^T$$

$$\star (H_k \geq 0 \Rightarrow H_{k+1} \geq 0)$$

Setting H_0 : At the initial step, H_0 can be set to the exact or approximate Hessian-inverse at x_0 , to I or some diagonal matrix D representing the scales of the variables.

Alg. 6.1 BFGS Method

Given x_0, H_0 and convergence tolerance $\epsilon > 0$

$$k \leftarrow 0$$

while $\|\nabla f_k\| > \epsilon$

Compute $p_k = -H_k \nabla f_k$

Set $x_{k+1} = x_k + \alpha_k p_k$ where α_k satisfies Wolfe cond.

Define $s_k = x_{k+1} - x_k, y_k = \nabla f_{k+1} - \nabla f_k$

Compute H_{k+1} using the BFGS update

$$k \leftarrow k+1$$

end (while)

Each iteration cost is $\mathcal{O}(n^2)$ since solving linear systems of equations is not necessary. The B-update version is

$$(\text{BFGS}) \quad B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

The latter would require solving $B_k p_k = -\nabla f_k$ at each step, which is $\mathcal{O}(n^3)$ with brute-force but could become $\mathcal{O}(n^2)$ by updating Cholesky factors of B_k instead.

In practice, BFGS is more effective than DFP. BFGS, when used with α_k that satisfies the Wolfe conditions, self-corrects errors in H in a few iterations.

For H_0 , a successful heuristic is to use $H_0 = I$ to compute the first update, then replace it with

$$H_0 \leftarrow \frac{y_0^T s_0}{y_0^T y_0} I \quad \text{just before computing } H_1 \text{ in BFGS.}$$

This scaling allows H_0 to approximate an eigenvalue of $\nabla^2 f_0$.

SRI Method (Symmetric-rank-1 update)

In this method, we have $B_{k+1} = B_k + \sigma v v^T$ where $\sigma \in \{-1, +1\}$ and σv are chosen to make B_{k+1} satisfy the secant equation: $B_{k+1} s_k = y_k$. Then

$$y_k = B_k s_k + (\sigma v^T s_k) v$$

$\therefore v$ must be a multiple of $(y_k - B_k s_k)$, so we set $v = \delta(y_k - B_k s_k)$ for some scalar δ . Then

$$(y_k - B_k s_k) = \sigma \delta^2 [s_k^T (y_k - B_k s_k)] (y_k - B_k s_k)$$

This equation is satisfied iff $\sigma = \text{sgn}(s_k^T (y_k - B_k s_k))$ and $\delta = \pm |s_k^T (y_k - B_k s_k)|^{-1/2}$. Hence, the only symmetric-rank-1 update formula that satisfies the second equation is

$$(SR1) \quad B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}$$

Using the S-M-W formula to get $H_k = B_k^{-1}$

$$(SR1) \quad H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}$$

Note that $B_k > 0 \not\Rightarrow B_{k+1} > 0$! Consequently, SR1 might not be suitable for line search methods, but it could be still useful for trust-region methods where indefinite models can be beneficial.

In SR1, to prevent numerical instability, the update for B_{k+1} is calculated only if

$$|s_k^T (y_k - B_k s_k)| \geq r \|s_k\| \|H_k y_k - B_k s_k\|$$

for some $r \in (0, 1)$, e.g. $r = 10^{-8}$. This skipping of the update where we set $B_{k+1} \leftarrow B_k$ occurs rarely when the two vectors align.

Alg. 6-2

SR1 Trust-Region Method

Given x_0, B_0 , trust-region radius Δ_0 , tolerance $\epsilon > 0$
parameters $\eta \in (0, 10^{-3})$, $r \in (0, 1)$.

$k \leftarrow 0$

while $\|\nabla f_k\| > \epsilon$

Compute s_k by solving $\min_s \nabla f_k^T s + \frac{1}{2} s^T B_k s$ s.t. $\|s\| \leq \Delta_k$

Compute $y_k = \nabla f(x_k + s_k) - \nabla f_k$

ared = $f_k - f(x_k + s_k)$ (Actual reduction)

pred = $-\left(\nabla f_k^T s_k + \frac{1}{2} s_k^T B_k s_k\right)$ (Predicted reduction)

if $\text{ared/pred} > \eta$, $x_{k+1} = x_k + s_k$, else $x_{k+1} = x_k$, end

if $\text{ared/pred} > 0.75$

if $\|s_k\| \leq 0.8 \Delta_k$, $\Delta_{k+1} = \Delta_k$, else $\Delta_{k+1} = 2 \Delta_k$, end

else if $0.1 \leq \text{ared/pred} \leq 0.75$

$\Delta_{k+1} = \Delta_k$

else

$\Delta_{k+1} = 0.5 \Delta_k$

end (if)

if $|s_k^T (y_k - B_k s_k)| \geq r \|s_k\| \|y_k - B_k s_k\|$

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

else

$B_{k+1} \leftarrow B_k$

end (if)

end (while) $k \leftarrow k+1$

(Quadratic case & SRT)

(128)

Thm 6.1: Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is a strongly convex function $f(x) = b^T x + \frac{1}{2} x^T A x$, where $A \succ 0$, sym.

For any starting point x_0 and any starting symmetric matrix H_0 , the iterates $\{x_k\}$ generated by SRT

(using $x_{k+1} = x_k + p_k$, unit step length) converge to the minimizer in at most n steps, provided that $(s_k - H_k y_k)^T y_k \neq 0 \quad \forall k$. Moreover, if n steps are performed and the search directions p_i are linearly independent, then $H_n = A^{-1}$.

Proof. See pg 148-149 in the book.

(General nonlinear case & SRT)

Thm 6.2: Suppose that $f \in C^2$ and its Hessian is bounded and Lipschitz cont. in a neighborhood of x^* . Let $\{x_k\}$ be any sequence of iterates $\{x_k \rightarrow x^* \in \mathbb{R}^n\}$. Suppose, in addition, that the inequality $|s_k^T (y_k - B_k s_k)| \geq r \|s_k\| \|y_k - B_k s_k\|$ holds $\forall k$, for some $r \in (0, 1)$, and that the steps s_k are uniformly linearly independent (i.e. they don't fall into a subspace with dimension). Then, for SRT,

$$\lim_{k \rightarrow \infty} \|B_k - \nabla^2 f(x^*)\| = 0.$$

The Broyden Class

This family of updates is characterized by the following:

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

where $\phi_k \in \mathbb{R}$ is a scalar parameter and

$$v_k = \begin{cases} \frac{y_k}{y_k^T s_k} & - \frac{B_k s_k}{s_k^T B_k s_k} \end{cases}$$

Note that $\phi_k = 1 \Rightarrow \text{DFP}$ and $\phi_k = 0 \Rightarrow \text{BFGS}$.

In fact, we have $B_{k+1} = (1 - \phi_k) B_{k+1}^{\text{BFGS}} + \phi_k B_{k+1}^{\text{DFP}}$, which is a weighted average.

Since both B^{DFP} and B^{BFGS} satisfy the second equation, members of the Broyden class also do. Similarly, since both B^{DFP} and B^{BFGS} preserve the positive-definiteness of the matrix when $s_k^T y_k > 0$, the same property holds for the Broyden class if $0 \leq \phi_k \leq 1$, since the set of positive definite matrices is convex.

This ^{subset} of Broyden ~~class~~ algorithms (with $0 \leq \phi_k \leq 1$) is called the restricted Broyden class.

Theorem 6.3: Suppose that $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is a strongly convex quadratic function, $f(x) = \frac{1}{2} x^T A x + b^T x$, where $A > 0$, sym. Let x_0 be any starting point for $x_{k+1} = x_k + p_k$ with $p_k = -B_k^{-1} \nabla f_k$ and suppose that B_k are updated using the Broyden formula with $\phi_k \in [0, 1]$. Let $\lambda_i^k \leq \dots \leq \lambda_n^k$ be the eigenvalues of $A^{1/2} B_k^{-1} A^{1/2}$. Then, we have

$$\min\{\lambda_i^k, 1\} \leq \lambda_i^{k+1} \leq \max\{\lambda_i^k, 1\}$$

Moreover, this inequality does not hold if $\phi_k \notin [0, 1]$.

* Note that if $\lambda_i^k = 1$ for all i then $B_k = A$, which is the ideal case. The theorem tells us that the eigenvalues converge monotonically (but not strictly monotonically) to 1. This property holds even if an inexact line search based step length was used.

* Note that $\phi_k = \frac{s_k^T y_k}{s_k^T y_k - s_k^T B_k s_k} \Rightarrow \text{SR1}$, but thus ϕ_k could sometimes take values outside the interval $[0, 1]$. The theorem does not exclude such values of ϕ_k , it simply guarantees some outcomes if ϕ_k is in the restricted class.

The last term in the Broyden update is a rank-1 update. Therefore, due to the interlacing eigenvalue theorem (Thm A.1) the eigenvalues increase when $\phi_k \geq 0$. Therefore, we have $B_{k+1} > 0 \wedge \phi_k \geq 0$. If $\phi_k < 0$ then the eigenvalues decrease - eventually for a suitable negative ϕ_k , B_{k+1} becomes singular, then indefinite. Specifically,

$$\phi_k^c = \frac{1}{1-\mu_k} \Rightarrow B_{k+1} \text{ is singular}$$

$$\text{where } \mu_k \triangleq \frac{(y_k^T B_k^{-1} y_k)(s_k^T B_k s_k)}{(y_k^T s_k)^2}$$

This was proved in an earlier exercise.



From the Cauchy-Schwarz inequality, we see that $\mu_k \geq 1$, therefore $\phi_k^c \leq 0$. Hence;

- * If $B_0 > 0$, symmetric and $s_k^T y_k > 0$ and $\phi_k > \phi_k^c \ \forall k$, then $B_k > 0$, symmetric $\forall k$.

Note that when the line search is exact, all methods in the Broyden class with $\phi_k \geq \phi_k^c$ generate the same sequence of iterates. (Prove this as an exercise.)

The properties of Broyden class applied to quadratic functions with exact line search are summarized next.

Thm. 6.4: Suppose that a method in the Broyden class is applied to the strongly convex quadratic function $f(x) = b^T x + \frac{1}{2} x^T A x$, where x_0 and $B_0 > 0$, sym. are starting values. Assume that α_k is the exact step length and that $\phi_k \geq \phi_k^c \ \forall k$, where ϕ_k^c is as defined earlier. Then the following are true.

- 1) The iterates are independent of ϕ_k and converge to the solution in at most n iterations.
- 2) The secant equation is satisfied for all previous search directions, that is $B_k s_j = y_j$, $j = k-1, k-2, \dots, 1$.
- 3) If $B_0 = I$, then the iterates are identical to those generated by the conjugate gradient method. In particular, the search directions are conjugate wrt A :

$$s_i^T A s_j = 0 \text{ for } i \neq j$$
- 4) $B_n = A$.

Note that if $B_0 \neq I$, then the Broyden class method is identical to preconditioned conjugate gradient. Also, the theorem continues to hold for $\phi_k^* < \phi_k^c$ provided that the iterates do not generate a singular B_k .

Convergence Analysis

In this section, $\|\cdot\|$ refers to the Euclidean norm and $G(x) \triangleq \nabla^2 f(x)$.

Global convergence of the BFGS Method

Assumption 6.1

- 1) The objective function f is twice cont. diff., $f \in C^2$.
- 2) The level set $\mathcal{L} = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$ is convex and $\exists m, M > 0 \Rightarrow m\|z\|^2 \leq z^T G(x) z \leq M\|z\|^2$ $\forall z \in \mathbb{R}^n$ and $\forall x \in \mathcal{L}$.

~~2)~~ The second assumption implies that $G(x) \succ 0$ on \mathcal{L} and that f has a unique minimizer x^* in \mathcal{L} .

Theorem 6.5: Let B_0 be any symmetric positive definite initial matrix and x_0 be the initial point for which Assumption 6.1 is satisfied. Then $\{x_k\}$ generated by Alg. 6.1 (BFGS) using $\epsilon=0$ converges to x^* , the minimizer of f in \mathcal{L} .

Proof: See pg 154-156 in the book.

Note that this theorem holds for all of the restricted Broyden class except DFP (i.e. for $\phi_k \in \{0, 1\}$).

Superlinear Convergence of the BFGS Method

Assumption 6.2: The Hessian matrix G is Lipschitz continuous at x^* , that is $\|G(x) - G(x^*)\| \leq L\|x - x^*\|$ for all x near x^* , with $L > 0$.

Now let $G_x \triangleq G(x^*)$ and $\tilde{s}_k = G_x^{-1/2} s_k$, $\tilde{y}_k = G_x^{-1/2} y_k$ and $\tilde{B}_k = G_x^{-1/2} B_k G_x^{-1/2}$. Also define

$$\cos \tilde{\theta}_k \triangleq \frac{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k}{\|\tilde{s}_k\| \cdot \|\tilde{B}_k \tilde{s}_k\|} \quad \text{and} \quad \tilde{q}_k \triangleq \frac{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k}{\|\tilde{s}_k\|^2}$$

$$\tilde{M}_k \triangleq \frac{\|\tilde{y}_k\|^2}{\tilde{y}_k^T \tilde{s}_k} \quad \text{and} \quad \tilde{M}'_k \triangleq \frac{\tilde{y}_k^T \tilde{s}_k}{\tilde{s}_k^T \tilde{s}_k}$$

By pre- and post-multiplying the BFGS update with $G_x^{-1/2}$

$$\tilde{B}_{k+1} = \tilde{B}_k - \frac{\tilde{B}_k \tilde{s}_k \tilde{s}_k^T \tilde{B}_k}{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k} + \frac{\tilde{y}_k \tilde{y}_k^T}{\tilde{y}_k^T \tilde{s}_k}$$

This has the same form as the original BFGS update.

Recall that for the average Hessian \bar{G}_k , $y_k = \bar{G}_k s_k$, so

$$y_k - G_x s_k = (\bar{G}_k - G_x) s_k \quad \begin{matrix} \text{pre-/post-multiply} \\ \text{with } G_x^{-1/2} \end{matrix}$$

thus $\tilde{y}_k - \tilde{s}_k = G_x^{-1/2} (\bar{G}_k - G_x) G_x^{-1/2} \tilde{s}_k$. From Assumption 6.2

$$\|\tilde{y}_k - \tilde{s}_k\| \leq \|G_x^{-1/2}\|^2 \cdot \|\tilde{s}_k\| \cdot \|\bar{G}_k - G_x\| \leq \|G_x^{-1/2}\|^2 \cdot \|\tilde{s}_k\| L \epsilon_k$$

where $\epsilon_k \triangleq \max \{ \|x_{k+1} - x^*\|, \|x_k - x^*\| \}$.

Consequently

$$\frac{\|\tilde{y}_k - \tilde{s}_k\|}{\|\tilde{s}_k\|} \leq \bar{c} \epsilon_k \text{ for some } \bar{c} > 0.$$

(135)

Thm 6.6: Suppose that $f \in C^2$ and BFGS iterates $\{x_k\}$ converge to x^* , a minimizer at which Assumption 6.2 holds. Also suppose that $\sum_{k=1}^{\infty} \|x_k - x^*\| < \infty$ (i.e. $x_k \rightarrow x^*$ rapidly). Then x_k converges to x^* superlinearly.

Proof: See pg 158-160 in the book. The proof shows

that $\lim_{k \rightarrow \infty} \frac{\|(B_k - G_k) s_k\|}{\|s_k\|} = 0$

Convergence Analysis of the SR1 Method

The convergence of SR1 is not as well understood as that of BFGS.

Thm 6.7: Suppose that the iterates x_k generated by Alg. 6.2 (SR1 Trust Region method) also understand as

(i) The sequence of iterates does not terminate, but remains in a closed, bounded, convex set D , on which the function f is twice cont. diff., and in which f has a unique stationary point x^* ;

(ii) the Hessian $\nabla^2 f(x^*) > 0$ and $\nabla^2 f(x)$ is Lipschitz continuous in a neighborhood of x^* .

(iii) the sequence $\{B_k\}$ is bounded in norm

(iv) $|s_k^T(y_k - B_k s_k)| \geq r \|s_k\| \cdot \|y_k - B_k s_k\|$ holds at every iteration, where $r \in (0, 1)$.

Then $\lim_{k \rightarrow \infty} x_k = x^*$ and $\lim_{k \rightarrow \infty} \frac{\|x_{k+n} - x^*\|}{\|x_k - x^*\|} = 0$.

Note that while in SRI, B_k is not necessarily pos. defn. at every iteration, they are positive definite almost all the time asymptotically i.e. as $k \rightarrow \infty$ number of iterations where $B_k^{SRI} > 0$ (divided by k approaches 1).

The theorem above shows the conditions under which SRI will converge to x^* at an $(1+1)$ -step superlinear rate.

Exercises

6.1) Show that if f is strongly convex, $s_k^T y_k > 0$ $\forall x_k, y_k$.

$$\begin{aligned} s_k^T y_k &= (x_{k+1} - x_k)^T (\nabla f_{k+1} - \nabla f_k) = \alpha_k p_k^T (\nabla f_{k+1} - \nabla f_k) \\ &= \alpha_k \|p_k\| \left(\frac{p_k^T}{\|p_k\|} \nabla f_{k+1} - \frac{p_k^T}{\|p_k\|} \nabla f_k \right) = \alpha_k \|p_k\| (\phi'(1) - \phi'(0)) \end{aligned}$$

where $\phi'(\cdot)$ is the derivative of f along crossection p_k .

Since f is strictly convex and since p_k is a descent direction ($\phi'(0) < 0$), we have $\phi'(1) > \phi'(0)$, thus $s_k^T y_k > 0$.

6.2) Show that the second strong Wolfe condition implies $s_k^T y_k > 0$.

Similar to the reasoning above, if f is not strictly convex then $s_k^T y_k = \alpha_k p_k^T (\nabla f_{k+1} - \nabla f_k) = \alpha_k (p_k^T \nabla f_{k+1} - p_k^T \nabla f_k)$. Since

$|\nabla f_{k+1}^T p_k| \leq c_2 |\nabla f_k^T p_k|$, we have $p_k^T \nabla f_{k+1} - p_k^T \nabla f_k > 0$ and with $\alpha_k > 0$ we get $s_k^T y_k > 0$.

Chapter 7: Large-Scale Unconstrained Optimization

In this context, large-scale usually means thousands or millions of parameters to optimize. Conjugate-gradient can be applied without modification due to its low storage demands and reliance on first order derivatives only. Line search and trust regions Newton methods are not ^{usually} suitable due to the requirement to invert/decompose the Hessian $\nabla^2 f_k$. In the case of sparse matrices, this could be exploited. Quasi-Newton methods usually generate dense Hessian approximations.

Inexact Newton Methods

The basic Newton step is given by $\nabla^2 f_k p_k = -\nabla f_k$. We would like to approximate this search direction with inexpensive computations. Most rules for iteratively solving $\nabla^2 f_k p_k + \nabla f_k = 0$ terminate based on the norm of this residual, $r_k \stackrel{\Delta}{=} \nabla^2 f_k p_k + \nabla f_k$:

$$\|r_k\| \leq \gamma_k \|\nabla f_k\| \quad \text{with } 0 < \gamma_k < 1 \text{ A.k.}$$

* $\{\gamma_k\}$ is called the forcing sequence.

Thm 7.1: Suppose that $\nabla^2 f(x)$ exists and is cont. in a neighborhood of a minimizer x^* with $\nabla^2 f(x^*) > 0$.

Consider $x_{k+1} = x_k + p_k$ where p_k satisfies $\|r_k\| \leq \eta_k \|f_k\|$, and assume $\eta_k \leq \eta$ for some $\eta \in (0, 1)$. Then, if the starting point x_0 is sufficiently near x^* , $\{x_k\} \rightarrow x^*$ and satisfies

$$\|\nabla^2 f(x^*) (x_{k+1} - x^*)\| \leq \tilde{\eta} \|\nabla^2 f(x^*) (x_k - x^*)\|$$

for some $\tilde{\eta}$ with $\eta < \tilde{\eta} < 1$.

In order to see this result, note that $\nabla^2 f(x^*) > 0$ and is continuous near x^* , so $\exists L > 0$ s.t. $\|\nabla^2 f_k^{-1}\| \leq L$ $\forall x_k$ sufficiently close to x^* . Therefore, we have

$$\|p_k\| \leq L (\|\nabla f_k\| + \|r_k\|) \leq 2L \|\nabla f_k\|$$

since

$$p_k = \nabla f_k^{-1} (r_k - \nabla f_k)$$

$$\Rightarrow \|p_k\| \leq L (\|r_k\| + \|\nabla f_k\|)$$

since $\eta_k < 1$

$$\Rightarrow \|r_k\| \leq \|\nabla f_k\|$$

Using this inequality and Taylor's Thm with the continuity of $\nabla^2 f(x)$, we get

Taylor's Thm

(139)

$$\begin{aligned}
 \nabla f_{k+1} &= \nabla f_k + \nabla^2 f_k p_k + \int_0^1 \{ \nabla f(x_k + t p_k) - \nabla f(x_k) \} p_k dt \\
 &= \nabla f_k + \nabla^2 f_k p_k + o(\|p_k\|) \\
 &= \nabla f_k - (\nabla f_k - r_k) + o(\|\nabla f_k\|) \quad \begin{matrix} \text{inequalities} \\ \text{above} \end{matrix} \\
 &= r_k + o(\|\nabla f_k\|)
 \end{aligned}$$

Then $\|\nabla f_{k+1}\| \leq \gamma_k \|\nabla f_k\| + o(\|\nabla f_k\|) \leq (\gamma_k + o(1)) \|\nabla f_k\|$.

When x_k is close enough to x^* , the $o(1)$ term is bounded by $(1-\eta)/2$, so

$$(*) \quad \|\nabla f_{k+1}\| \leq (\gamma_k + (1-\eta)/2) \|\nabla f_k\| \leq \frac{1+\eta}{2} \|\nabla f_k\|$$

hence, the gradient norm decreases by a factor of $\frac{1+\eta}{2}$.

Consequently, once the iterates come to the vicinity of x^* this rate of decrease is achieved at every iteration.

Note that, under the smoothness assumption, we have

$$\nabla f_k = \nabla^2 f(x^*) (x_k^* - x^*) + o(\|x_k - x^*\|).$$

A similar expression holds for x_{k+1} as well. So, the inequality in Thm 7.1 follows from this. Also, from (*) above, we have $\frac{\|\nabla f_{k+1}\|}{\|\nabla f_k\|} \leq \gamma_k + o(1)$. If $\lim_{k \rightarrow \infty} \gamma_k = 0$,

then $\lim_{k \rightarrow \infty} \frac{\|\nabla f_{k+1}\|}{\|\nabla f_k\|} = 0$, which indicates Q-superlinear convergence of $\|\nabla f_k\| \rightarrow 0$. Consequently $x_k \rightarrow x^*$ superlinearly.

If $\nabla^2 f(x)$ is Lipschitz cont. near x^* , then one would have $\nabla f_{k+1} = r_k + \mathcal{O}(\|\nabla f_k\|^2)$; and choosing $\gamma_k = \mathcal{O}(\|\nabla f_k\|)$ we will get $\|\nabla f_{k+1}\| = \mathcal{O}(\|\nabla f_k\|^2)$, indicating quadratic convergence of $\|\nabla f_k\| \rightarrow 0$, thus $x_k \rightarrow x^*$. Formally --

Theorem 7.2: Suppose that the conditions of Thm 7.1 hold and assume that the iterates $\{x_k\}$ generated by the inexact Newton method converge to x^* . Then the rate of convergence is superlinear if $\eta_k \rightarrow 0$. If, in addition, $\nabla^2 f(x)$ is Lipschitz continuous for x near x^* and if $\eta_k = \mathcal{O}(\|\nabla f_k\|)$ then the convergence is quadratic.

Fact $\eta_k = \min(0.5, \|\nabla f_k\|^{1/2}) \Rightarrow$ superlinear convergence

$\eta_k = \min(0.5, \|\nabla f_k\|) \Rightarrow$ quadratic convergence.

The second choice forces p_k to approximate p_k^* more accurately, so it needs more iterations for p_k , but at each line search ~~step~~ iteration, it could perform better.

These results assumed that $\{x_k\}$ enters the vicinity of x^* and $\alpha_k = 1$ is used as the step length.

Can we integrate line search?

Line Search Newton-CG Method (Truncated Newton Method)

Here, we apply the CG method to $\nabla^2 f_k P_k = -\nabla f_k$. However, since CG is designed to solve linear equations with a positive definite matrix, if $\nabla^2 f_k$ is indefinite when far from a solution, CG is terminated. Now consider the general form $B_k P = -\nabla f_k$ -- where $B_k = \nabla^2 f_k$ here.

When $B_k > 0$, the inner CG iteration for z_j , in the following algorithm, converges to p_k^n . Also ϵ_k is used for $\eta_k^{(1) \text{st}}$ in the previous discussion.

Alg. 7.1 Line search Newton-CG

Given x_0

for $k=0, 1, 2, \dots$

let $\epsilon_k = \min(0.5, \|\nabla f_k\|^{1/2}) \|\nabla f_k\|$, set $z_0 = 0$, $r_0 = \nabla f_k$, $d_0 = -r_0 = -\nabla f_k$

for $j = 0, 1, 2, \dots$ (negative curvature test ensures)
 P_k is a descent direction.)

if $d_j^T B_k d_j \leq 0$

if $j=0$, return $P_k = -\nabla f_k$, else return $P_k = z_j$, end(if)

Set $\alpha_j = (r_j^T r_j) / (d_j^T B_k d_j)$, $z_{j+1} = z_j + \alpha_j d_j$, $r_{j+1} = r_j + \alpha_j B_k d_j$

if $\|r_{j+1}\| < \epsilon_k$, return $P_k = z_{j+1}$, end(if)

Set $\beta_{j+1} = (r_{j+1}^T r_{j+1}) / (r_j^T r_j)$, $d_{j+1} = -r_{j+1} + \beta_{j+1} d_j$

end(end(j))

end(for)

end(for) Set $x_{k+1} = x_k + \alpha_k P_k$, where α_k satisfies the Wolfe conditions. Use $\alpha_k = 1$ if possible.

This is
the inner
CG loop
that allows
early stopping
at $\leq \epsilon_k$.
error.

- * Note that if $\nabla^2 f_k$ is nearly singular, this algorithm might require many function evaluations in line search and still lead to poor decrease in ~~perf~~ objective. The trust-region Newton-CG method (later) deals with this situation more efficiently, and is preferable.
- * The line search Newton-CG method only requires Hessian-vector products of the form $\nabla^2 f_k \cdot d$ for a given vector d . These products can be approximated using finite differencing, for instance (Chapter 8) ...

$$\nabla^2 f_k \cdot d \approx \frac{\nabla f(x_k + h d) - \nabla f(x_k)}{h}$$

This would lead to a Hessian-free Newton method. The approximation requires one new gradient calculation per CG iteration.

Trust-Region Newton-CG Method

Here, we describe a modified CG algorithm for solving the trust-region subproblem to determine a step p_k (given some tolerance ϵ_k) for Alg. 4.1. (Trust-Region).

Consider $\min_{p \in \mathbb{R}^n} M_k(p) \stackrel{\Delta}{=} f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p$ s.t. $\|p\| \leq \Delta_k$

where $B_k = \nabla^2 f_k$.

Alg. 7.2 CG-Steihaug (developed by Steihaug)

Given $\epsilon_k > 0$ tolerance, set $z_0 = 0, r_0 = \nabla f_{k_0}, d_0 = -r_0 = -\nabla f_{k_0}$.

if $\|r_0\| < \epsilon_k$, return $p_k = z_0 = 0$, end (if)

for $j=0, 1, 2, \dots$

(*) if $d_j^T B_k d_j \leq 0$ (Terminate if B_k indefinite)

Find $\tau \geq 0$ s.t. $p_k = z_j + \tau d_j$ minimizes $m_k(p)$ with $\|p_k\| = \Delta_k$.

return p_k

end (if)

Set $\alpha_j = (r_j^T r_j) / (d_j^T B_k d_j); z_{j+1} = z_j + \alpha_j d_j;$

(*) if $\|z_{j+1}\| \geq \Delta_k$ (Terminate if current CG iterate violates trust-region)

find $\tau \geq 0$ s.t. $p_k = z_j + \tau d_j$ satisfies $\|p_k\| = \Delta_k$.

return p_k

end (if)

Set $\beta_{j+1} = (r_{j+1}^T r_{j+1}) / (r_j^T r_j); d_{j+1} = -r_{j+1} + \beta_{j+1} d_j;$

end (for)

In both checks marked with (*), if a violation occurs, the current CG search direction is followed until the trust-region boundary. Here, $\{d_j\}$ is the CG search directions and $\{z_j\}$ is the CG iterates. Here, ϵ_k can be selected in a manner similar to Alg. 7.1.

Note that if $\|r_{k+1}\| \geq \epsilon_k$, Alg. 7.2 terminates at a point p_k for which $m_k(p_k) \leq m_k^c(p_k^c)$. (144) Caching point

To see this, consider

Case 1: if $d_0^T B_k d_0 = \|f_k\|^T B_k \|f_k\| \leq 0$, then the first IF condition is satisfied and $p = -\frac{\Delta_k}{\|f_k\|} \|f_k\| = p_k^c$ is returned.

Case 2: otherwise, $z_i = \alpha_i d_0 = \frac{r_0^T r_0}{d_0^T B_k d_0} d_0 = -\frac{\|f_k\|^T \|f_k\|}{\|f_k\|^T B_k \|f_k\|} \|f_k\|$

case 2.1) if $\|z_i\| \leq \Delta_k$, then $z_i = p_k^c$. Subsequent steps ensure $m_k(p_k) \leq m_k(z_i)$.

case 2.2) if $\|z_i\| \geq \Delta_k$, the second IF statement is activated and the algorithm returns p_k^c again.

Hence, the algorithm is globally convergent.

Theorem 7.3: The sequence $\{z_j\}$ generated by Alg 7.2

satisfies $0 = \|z_0\|_2 \leq \dots \leq \|z_j\|_2 \leq \|z_{j+1}\|_2 \leq \dots \leq \|p_k\|_2 \leq \Delta_k$
 (Consequently once $\|z_{j+1}\| \geq \Delta_k$ is observed, algorithm can terminate.)

Proof: Note that ~~$z_j = z_0 + \sum_{i=0}^{j-1} \alpha_i d_i$~~ $z_j = z_0 + \sum_{i=0}^{j-1} \alpha_i d_i = \sum_{i=0}^{j-1} \alpha_i d_i$.

Multiplying by r_j and applying the expanding subspace property of conjugate gradients...

$$z_j^T r_j = \sum_{i=0}^{j-1} \alpha_i d_i^T r_j = 0 \quad (\text{Due to Thm 5.2, we have } d_i^T r_j = 0 \text{ for } i=0, \dots, (j-1).)$$

$$\therefore z_j^T r_j = 0 \text{ for } j \geq 0.$$

$$\text{Now consider } z_j^T d_i = (\alpha_0 d_0)^T (-r, r\beta, d_0) \\ = \alpha_0 \beta, d_0^T d_0 > 0$$

(145)

Then assume $z_j^T d_j > 0$ (start induction) and check

$$\begin{aligned} z_{j+1}^T d_{j+1} &= z_{j+1}^T (-r_{j+1} + \beta_{j+1} d_j) \\ &= \beta_{j+1} z_{j+1}^T d_j = \beta_{j+1} (z_j + \alpha_j d_j)^T d_j \\ &= \underbrace{\beta_{j+1}}_{>0} \underbrace{z_j^T d_j}_{>0} + \underbrace{\alpha_j \beta_{j+1}}_{>0} \underbrace{d_j^T d_j}_{\geq 0} > 0 \text{ since } \alpha_j, \beta_{j+1} > 0. \\ \therefore z_j^T d_j &> 0 \end{aligned}$$

Finally, if Alg 7.2 terminates because $d_j^T B_k d_j \leq 0$
or $\|z_{j+1}\| \geq D_k$, then $\|p_k\| = D_k$ (the largest possible length).

Otherwise, we must show that $\|z_j\| < \|z_{j+1}\|$ when

$z_{j+1} = z_j + \alpha_j d_j$ and $j \geq 1$. Observe that

$$\begin{aligned} \|z_{j+1}\|^2 &= (z_j + \alpha_j d_j)^T (z_j + \alpha_j d_j) \\ &= \|z_j\|^2 + 2\alpha_j z_j^T d_j + \underbrace{\alpha_j^2 \|d_j\|^2}_{>0} \\ \therefore \|z_{j+1}\| &> \|z_j\| \end{aligned}$$

B

Note that this algorithm (Alg 7.2) behaves similar to the dogleg method: it starts off with $-D_k$ and progresses towards p_k^* , increasing the step-magnitude, until D_k intervenes.

Precconditioning the Trust-Region Newton-CG Method

Precconditioning can accelerate the CG iterations. The idea is to use a nonsingular matrix D such that the eigenvalues of $D^{-T} \nabla^2 f_k D^{-1}$ have a more favorable distribution.

We ~~can~~ redefine the trust-region subproblem as:

$$\min_{p \in \mathbb{R}^n} M_k(p) \stackrel{\Delta}{=} f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p \quad \text{s.t. } \|Dp\| \leq \Delta_k$$

Changing variables: $\hat{p} = Dp \Rightarrow \hat{g}_k = D^{-T} \nabla f_k, \hat{B}_k = D^{-T} \nabla^2 f_k D^{-1}$.

Then we get $\min_{\hat{p} \in \mathbb{R}^n} f_k + \hat{g}_k^T \hat{p} + \frac{1}{2} \hat{p}^T \hat{B}_k \hat{p} \quad \text{s.t. } \|\hat{p}\| \leq \Delta_k$.

Now, Alg. 7-2 can be applied without any modifications. The following algorithm provides a modified incomplete Cholesky factorization as a preconditioner for the trust-region Newton-CG approach where $D = L^T$ and $B = L^T R L$ (where R is the residual error after incomplete factorization).

Alg. 7-3 Inexact Modified Cholesky

Compute $T = \text{diag}(\|B_1\|, \dots, \|B_n\|)$ where e_i is the i^{th} coordinate vector.
Set $\bar{B} \leftarrow T^{-1/2} B T^{-1/2}$ and $\beta \leftarrow \|\bar{B}\|$.

if $\min_i b_{ii} > 0$, $x_0 \leftarrow 0$, else, $x_0 \leftarrow \beta/2$, end(if)

for $k = 0, 1, 2, \dots$

Attempt to apply incomplete Cholesky algorithm to obtain $L L^T = \bar{B} + \alpha_k I$.
if factorization is successful, return L , else $x_{kn} \leftarrow \max(2x_0, \beta/2)$, end(if)

end(for)

Trust-Region Newton-Lanczos Method

(147)

Lanczos method is a generalization of the CG procedure to solve the linear system $B_k p = -\nabla f_k$ even when B_k is indefinite. After j steps, the Lanczos method generates an $n \times j$ matrix Q_j with orthogonal columns that span the appropriate Krylov space. The matrix satisfies $Q_j^T B Q_j = T_j$ where T_j is tridiagonal (only the principal diagonal and its upper and lower adjacent diagonals can have nonzero entries). Then we solve the trust-region subproblem:

$$\min_{w \in \mathbb{R}^j} f_k + e_1^T Q_j \Delta f_k e_1^T w + \frac{1}{2} w^T T_j w \quad \text{s.t. } \|w\| \leq \Delta_k$$

where e_1 is the first coordinate vector. The solution is an approximation to the original trust-region subproblem in the ~~range~~ of Q_j (span of its columns). Then $p_k = Q_j w$. Since T_j is tridiagonal, the problem above can be solved using the nearly exact approach we discussed at the end of Chapter 4.

Limited-Memory Quasi-Newton Methods

These are useful for solving large scale problems where Hessians cannot be computed or stored because they are not sparse. The main idea is to use curvature information from only recent iterations, since earlier curvatures are less likely to be relevant.

Limited Memory BFGS

Consider the original BFGS method:

$$x_{k+1} = x_k - \alpha_k H_k \nabla f_k$$

$$V_k = I P_k y_k s_k^T$$

$$\text{where } H_{k+1} = V_k^T H_k V_k + P_k S_k S_k^T \text{ with } P_k = (y_k^T S_k)^{-1}.$$

~~Here~~ $S_k \triangleq x_{k+1} - x_k, \quad y_k \triangleq \nabla f_{k+1} - \nabla f_k.$

Since, ultimately we want $H_k \nabla f_k$ (and H_k might be dense and large), suppose we store m vector pairs $\{s_i, y_i\}$ ~~at~~ where $m < n$ and at each iteration, discard the oldest pair and insert the new pair.

At iteration k , we have the m most recent vectors $\{s_i, y_i\}$ for $i = k-m, \dots, k-1$ and the current iterate x_k . Given an initial Hessian approximation H_k^0 , we repeatedly utilize the H -update formula above:

$$\begin{aligned} H_k &= \{V_{k-1}^T \dots V_{k-m}^T\} H_k^0 \{V_{k-m} \dots V_{k-1}\} \\ &\quad + P_{k-m} \{V_{k-1}^T \dots V_{k-m+1}^T\} S_{k-m} S_{k-m}^T \{V_{k-m+1} \dots V_{k-1}\} \\ &\quad + P_{k-m+1} \{V_{k-1}^T \dots V_{k-m+2}^T\} S_{k-m+1} S_{k-m+1}^T \{V_{k-m+2} \dots V_{k-1}\} \\ &\quad + \dots \\ &\quad + P_{k-1} S_{k-1} S_{k-1}^T \end{aligned}$$

$$\text{e.g. } H_k^1 = V_{k-m}^T H_k^0 V_{k-m} + P_{k-m} S_{k-m} S_{k-m}^T$$

iterate m times: $H_k^m \triangleq H_k$

$$H_k^2 = V_{k-m+1}^T V_{k-m}^T H_k^0 V_{k-m} V_{k-m+1} + P_{k-m} V_{k-m}^T S_{k-m} S_{k-m}^T V_{k-m+1} + P_{k-m+1} S_{k-m+1} S_{k-m+1}^T$$

This procedure is given in the following algorithm.

Alg. 7.4 L-BFGS two-loop recursion

$$q \leftarrow \nabla f_k$$

for $i = k-1, k-2, \dots, k-m$

$$\alpha_i \leftarrow p_i s_i^T q ; q \leftarrow q - \alpha_i y_i$$

end (for)

$$r \leftarrow H_k^0 q$$

for $i = k-m, k-m+1, \dots, k-1$

$$\beta \leftarrow p_i y_i^T r ; r \leftarrow r + s_i (\alpha_i - \beta)$$

end (for)

stop with result $H_k \nabla f_k = r \leftarrow \begin{array}{l} \text{(use } r \text{ as your)} \\ \text{search direction} \end{array}$

In practice, we could set $H_k^0 = \gamma_k I$ where $\gamma_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}$.

Alg. 7.5 L-BFGS

Choose x_0 and $m \in \mathbb{Z}, m > 0$. Set $k \leftarrow 0$.

repeat

Choose H_k^0 ; Compute $p_k \leftarrow -H_k \nabla f_k$ from Alg. 7.4;

Compute $x_{k+1} \leftarrow x_k + \alpha_k p_k$ where α_k satisfies Wolfe conditions.

if $k > m$ end(if).

Discard $\{s_{k-m}, y_{k-m}\}$; Compute and save s_k and y_k .

~~repeat~~ $k \leftarrow k + 1$

until convergence

Relationship with Conjugate Gradient Methods

Consider the Hestenes-Stiefel nonlinear conjugate gradient method. With $s_k = \alpha_k p_k$, we have

$$p_{k+1} = -\nabla f_{k+1} + \frac{\nabla f_k^T y_k}{y_k^T p_k} p_k = -\left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) \nabla f_k \triangleq -\hat{H}_{k+1}^{-1} \nabla f_k$$

In this expression, \hat{H}_{k+1} is neither symmetric nor pos. defn. It also does not satisfy the secant eqn: $\hat{H}_{k+1} y_k \neq s_k$.

If we apply a single BFGS update ($H_{k+1} = V_k^T H_k V_k + P_k S_k S_k^T$) to the identity matrix, we get

$$H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T s_k}$$

This matrix is symmetric, positive definite, and it satisfies the secant eqn. We can consider this as a memoryless BFGS method ($H_k^0 = I$ and $m=1$ in L-BFGS).

If in this case we use $p_{k+1} = -H_{k+1} \nabla f_{k+1}$ and select α_k with an exact line search (i.e. $\nabla f_{k+1}^T p_{k+1} = 0$), then

$$p_{k+1} = -H_{k+1} \nabla f_{k+1} = -\nabla f_{k+1} + \frac{\nabla f_k^T y_k}{y_k^T p_k} p_k$$

which is the Hestenes-Stiefel step. In this case, this is also equal to the Polak-Ribière formula.

Compact Representation of BFGS Updating

We will consider the BFGS approximation B_{k+1} to $\nabla^2 f_k$.

Theorem 7.4: Let B_0 be symmetric and pos. definite. Assume that the k vector pairs $\{s_i, y_i\}_{i=0}^{k-1}$ satisfy $s_i^T y_i > 0$. Let B_k be obtained by applying k BFGS updates with these vector pairs to B_0 using $B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$.

Then we have

$$B_k = B_0 - \left\{ B_0 s_k \quad Y_k \right\} \begin{bmatrix} s_k^T B_0 s_k & L_k \\ L_k^T & -D_k \end{bmatrix}^{-1} \begin{bmatrix} s_k^T B_0 \\ Y_k^T \end{bmatrix}$$

where S_k and Y_k are the $k \times k$ matrices given by

$$(L_k)_{ij} = \begin{cases} s_i^T y_{j-1}, & \text{if } i > j \\ 0, & \text{o.w.} \end{cases}$$

$$D_k = \text{diag} \{ s_0^T y_0, \dots, s_{k-1}^T y_{k-1} \}$$

Proof: Can be done by induction. See pg 182-183 in the book.

* This theorem shows that B_k has the following form:

$$B_k = S_k I + \boxed{\quad} \boxed{\quad} \quad (\text{an outer product})$$

2 $m \times d(m^3)$ computations representation
needed for this update

* Using k SR1-updates to B_0 (symmetric), one would get

$$B_k = B_0 + (Y_k - B_0 S_k) (D_k + L_k + L_k^T - S_k^T B_0 S_k)^{-1} (Y_k - B_0 S_k)^T$$

$$\text{where } S_k = [s_{k-m} \dots s_{k-1}], \quad Y_k = [y_{k-m} \dots y_{k-1}]^T.$$

Unrolling the Update (skip)

The obvious implementation of L-BFGS updating can be more expensive than the one based on compact representations.

Consider the BFGS formula, which is of the form:

$$B_{k+1} = B_k - \alpha_k a_k a_k^T + b_k b_k^T \quad (*)$$

where $\alpha_k = \frac{B_k s_k}{(s_k^T B_k s_k)^{1/2}}$, $b_k = \frac{y_k}{(y_k^T s_k)^{1/2}}$.

We could continue to save the pairs $\{s_i, y_i\}$, but use (*) above to perform matrix-vector multiplications (same trick is used in adaptive filtering on-line).

The L-BFGS method will define B_k^0 and then update with $B_k = B_k^0 + \sum_{i=k-m}^{k-1} \{b_i b_i^T - \alpha_i a_i a_i^T\}$. The vector pairs $\{\alpha_i, b_i\}$, $i=k-m, \dots, k-1$ can be obtained from $\{s_i, y_i\}$ using:

Procedure 7.6 Unrolling the BFGS formula

for $i=k-m, \dots, k-1$

$$b_i \leftarrow \frac{y_i}{(y_i^T s_i)^{1/2}} ; \quad a_i \leftarrow B_k^0 s_i + \sum_{j=k-m}^{k-1} [(b_j^T s_i) b_j - (\alpha_j^T s_i) a_j]$$

$$\alpha_i \leftarrow \frac{a_i}{(s_i^T a_i)^{1/2}}$$

end(for)

Note: $\{\alpha_i\}$ must be recomputed at each iteration, but $\{b_i\}$ can be saved and recovered from the previous iteration. Similarly $b_i^T s_i$ could be saved and reused.

This procedure, assuming $B_k^0 = I$, requires $\frac{3}{2}n^2n$ operations to determine the limited-memory matrix. The computation of $B_m v$ for any $v \in \mathbb{R}^n$ requires $4mn$ multiplications. So this approach is less efficient than the compact form.

- Updating the limited memory matrix costs $2mn$ mult.

in compact form, while it costs $\frac{3}{2}n^2n$ mult. in BFGS update.

Sparse Quasi-Newton Updates

We would like to have quasi-Newton approximations B_k that exhibit similar sparsity patterns as $\nabla^2 f_k$.

Suppose that at ~~some~~ given point, we know which entries of the Hessian may be nonzero.

$$\mathcal{S} \stackrel{\Delta}{=} \{(i,j) \mid [\nabla^2 f(x)]_{ij} \neq 0 \text{ for some } x \in \text{Domain}(f)\}$$

Suppose that B_k mirrors the nonzero structure of $\nabla^2 f_k$, i.e. $(B_k)_{ij} = 0$ for $(i,j) \notin \mathcal{S}$. Try to find B_{k+1} that satisfies the second condition, has the same sparsity pattern, and is close to B_k :

$$\min_B \|B - B_k\|_F^2 = \sum_{(i,j) \in \mathcal{S}} (B_{ij} - (B_k)_{ij})^2$$

subject to $B s_k = y_k$, $B = B^T$, $B_{ij} = 0$ for $(i,j) \notin \mathcal{S}$.

The solution to this is obtained by solving an 154 $n \times n$ linear system of equations whose sparsity pattern is Ω . B_{k+1} is not guaranteed to be positive definite. The updating process is not scale invariant under linear variable transformations, and its practical performance is not good.

Alternatively, we could relax the secant equation condition and try to make sure it approximately holds for some recent steps. Define S_k and Υ_k containing the last m s_k, y_k vectors as before:

$$\min_B \|BS_k - \Upsilon_k\|_F^2 \quad \text{s.t. } B = B^T, B_{ij} = 0 \text{ for } (i,j) \notin \Omega.$$

This convex optimization is hard to solve numerically and it can produce poorly conditioned Hessian models.

Algorithms for Partially Separable functions

If our objective can be decomposed into a sum of simpler functions that can be optimized independently, that would be great. For instance,

$$f(x) = f_1(x_1, x_3) + f_2(x_2, x_4, x_6) + f_3(x_5)$$

can be minimized by solving $\min_{\bar{x}_i} f_i(\bar{x}_i), i=1,2,3$.

155

Even if f is not separable, if it can be written as the sum of simpler functions (element functions) it would be helpful.

Defn: f is partially separable iff each of its element functions are unaffected when we move along a large number of linearly independent directions.

Fact: $\nabla^2 f$ is sparse $\Leftrightarrow f$ is partially separable.

Consider $f(x) = \sum_{i=1}^{n_e} f_i(x)$ where each f_i only depends on a few components of x . Then ∇f_i and $\nabla^2 f_i$ all contain a few non-zero elements \Rightarrow

$$\nabla f(x) = \sum_{i=1}^{n_e} \nabla f_i(x) \quad \nabla^2 f(x) = \sum_{i=1}^{n_e} \nabla^2 f_i(x)$$

Example: $f(x) = (x_1 - x_3)^2 + (x_2 - x_6)^2 + (x_3 - x_2)^2 + (x_6 - x_1)^2$
 $= f_1(x) + f_2(x) + f_3(x) + f_4(x)$

$$x_{(1)} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = u_1 x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} x. \text{ Let } \phi_i(z_i, z_j) = (z_i - z_j)^2,$$

then $f_1(x) = \phi_1(u_1 x)$, so

$$\nabla f_1(x) = u_1^\top \nabla \phi_1(u_1 x)$$

$$\nabla^2 f_1(x) = u_1^\top \nabla^2 \phi_1(u_1 x) u_1$$

$$\text{We have } \nabla^2 \phi_1(u, x) = \begin{bmatrix} 2 & -4x_3 \\ -4x_3 & 12x_3^2 - 4x_1 \end{bmatrix}$$

$$\nabla^2 f(x) = \begin{bmatrix} 2 & 0 & -4x_3 & 0 \\ 0 & 0 & 0 & 0 \\ -4x_3 & 0 & 12x_3^2 - 4x_1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

U_i is called the compactifying matrix. In the algorithm, instead of maintaining a quasi-Newton approximation to $\nabla^2 f_1$, we maintain a quasi-Newton approximation to $\nabla^2 \phi_1$: $B_{[i]} \approx \nabla^2 \phi_1(u, x) = \nabla^2 \phi_1(x_{[i]})$.

Using $s_{[i]} = x_{[i]}^+ - x_{[i]}$, $y_{[i]} = \nabla \phi_1(x_{[i]}^+) - \nabla \phi_1(x_{[i]})$, where we take a step from $x_{[i]}$ to $x_{[i]}^+$ at some iteration. Then $\nabla^2 f_1(x) \approx U_i^T B_{[i]} U_i$ and

$$B = \sum_{i=1}^n U_i^T B_{[i]} U_i \approx \nabla^2 f(x)$$

$$\text{and } f(x) = \sum_{i=1}^n \phi_i(U_i x)$$

To determine the search direction, we need to solve $B_k p_k = -\nabla f_k$, but this could be done computationally efficiently since B_k is sparse and it consists of blocks. Note: Although $B_k > 0$, we might have some $B_{[i]}$ indefinite. These element Hessian models could be updated using SR1.

Chapter 8: Calculating Derivatives

We will discuss numerical approximations of derivatives. Finite-difference approximations will form the basis.

Approximating the Gradient

Consider the popular forward-difference formula

$$\frac{\partial f}{\partial x_i}(x) \approx \frac{f(x + \epsilon e_i) - f(x)}{\epsilon}, \quad i=1, \dots, n$$

This formula comes from the Taylor theorem (Thm 2.1).

For an f that is twice cont. differentiable:

$$f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x+t p) p$$

for some $t \in (0,1)$. If, in the region of interest, we have $\|\nabla^2 f(\cdot)\| \leq L$, then $\|f(x+p) - f(x) - \nabla f(x)^T p\| \leq \frac{L}{2} \|p\|^2$.

Now let $p = \epsilon e_i$, so that $\nabla f(x)^T p = \nabla f(x)^T e_i = \frac{\partial f}{\partial x_i}$:

$$\frac{\partial f}{\partial x_i}(x) = \frac{f(x + \epsilon e_i) - f(x)}{\epsilon} + \delta_\epsilon \text{ where } |\delta_\epsilon| \leq \frac{L}{2} \epsilon.$$

When f is computed using finite precision computers, we need to take into account the unit roundoff, u , of the computer (e.g. $u \approx 1.1 \times 10^{-16}$ in double precision IEEE floating point machines).

Let $\text{comp}(f(x))$ be the finite precision value for $f(x)$. Assume that $|f(\cdot)| \leq L_f$ in the region of interest. Then

$$|\text{comp}(f(x)) - f(x)| \leq u L_f$$

$$|\text{comp}(f(x+ee_i)) - f(x+ee_i)| \leq u L_f$$

Consequently, we get ^{upper} bound for $|\text{comp}(\frac{\partial f}{\partial x_i}(x)) - \frac{\partial f}{\partial x_i}(x)|$ as $(L/2) \epsilon + 2uL_f/\epsilon$. The value of ϵ that minimizes this error is $\epsilon_*^2 = \frac{4L_f u}{L} \Rightarrow \epsilon_* = 2\sqrt{\frac{L_f}{L}}\sqrt{u}$.

\therefore The optimal choice of ϵ in the forward-diff. approximation is on the order of \sqrt{u} .

Central Difference Formula: $\frac{\partial f}{\partial x_i}(x) \approx \frac{f(x+ee_i) - f(x-ee_i)}{2\epsilon}$

This approximation is twice as expensive compared to forward differencing, but it is more accurate.

Going back to Taylor's Thm and assuming that $\nabla^2 f$ is

Lipschitz continuous, we have

$$\begin{aligned} f(x+p) &= f(x) + \nabla f^T(x)p + \frac{1}{2} p^T \nabla^2 f(x+t p) p \quad \text{for some } t \in (0,1) \\ &= f(x) + \nabla f^T(x)p + \frac{1}{2} p^T \nabla^2 f(x)p + \mathcal{O}(\|p\|^3). \end{aligned}$$

Letting $p = ee_i$ and $p = -ee_i$ respectively:

$$f(x+ee_i) = f(x) + \epsilon \frac{\partial f}{\partial x_i} + \frac{1}{2} \epsilon^2 \frac{\partial^2 f}{\partial x_i^2} + \mathcal{O}(\epsilon^3)$$

$$f(x-ee_i) = f(x) - \epsilon \frac{\partial f}{\partial x_i} + \frac{1}{2} \epsilon^2 \frac{\partial^2 f}{\partial x_i^2} + \mathcal{O}(\epsilon^3)$$

Subtracting and dividing by 2ϵ :

$$\frac{\partial f}{\partial x_i}(x) = \frac{f(x+\epsilon e_i) - f(x-\epsilon e_i)}{2\epsilon} + \mathcal{O}(\epsilon^2)$$

Although the error here is $\mathcal{O}(\epsilon^2)$, as opposed to $\mathcal{O}(\epsilon)$ in forward-difference method, when we incorporate finite precision effects, this difference in accuracy is less than promised here. Specifically, an error on the order of $u^{2/3}$ is achieved and the optimal step is about $\epsilon_* \approx u^{1/3}$.

Approximating a Sparse Jacobian: Consider $r: \mathbb{R}^n \rightarrow \mathbb{R}^m$, a vector-valued function. The Jacobian matrix is

$$J_r(x) \triangleq \left[\frac{\partial r_j}{\partial x_i} \right]_{\substack{j=1 \dots m \\ i=1 \dots n}} \quad \begin{matrix} \text{where } j \text{ is the row index} \\ i \text{ is the column index} \end{matrix}$$

$$\text{Then } J_r(x) = \begin{bmatrix} \nabla r_1^\top(x) \\ \vdots \\ \nabla r_m^\top(x) \end{bmatrix} \quad (\text{precisely why I like my gradients as row!})$$

Each row of the Jacobian can be computed using the approximations mentioned above. When r is twice cont. diff. and J is Lipschitz cont. with coefficient L :

$$\|r(x+\rho) - r(x) - J(x)\rho\| \leq \frac{L}{2} \|\rho\|^2$$

$$\Rightarrow J(x)\rho \approx \frac{r(x+\epsilon\rho) - r(x)}{\epsilon} + \mathcal{O}(\epsilon)$$

i^{th} column of the Jacobian can then be approximated as

$$J(x) e_i = \frac{\partial r}{\partial x_i}(x) \approx \frac{r(x + \epsilon e_i) - r(x)}{\epsilon}$$

This can be used to estimate the full Jacobian by using $(n+1)$ evaluations of r . If the Jacobian is sparse, then the computations can be reduced.

Example

$$r(x) = \begin{cases} 2(x_1^3 - x_1^2) & (x_1, x_2) \\ 3(x_2^3 - x_1^2) + 2(x_3^3 - x_2^2) & (x_1, x_2, x_3) \\ 3(x_3^3 - x_2^2) + 2(x_4^3 - x_3^2) \\ \vdots \\ 3(x_n^3 - x_{n-1}^2) & (x_2, x_3, x_4) \\ \vdots \\ (x_{n-1}, x_n) \end{cases}$$

In this case, $J_r(x)$ is tridiagonal:

$$J(x) = \begin{bmatrix} 0 & x & 0 & 0 & 0 & 0 \\ x & 0 & x & 0 & 0 & 0 \\ 0 & x & 0 & x & 0 & 0 \\ 0 & 0 & x & 0 & x & 0 \\ 0 & 0 & 0 & x & 0 & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \quad p = \epsilon(e_1 + e_n) \text{ can be used simultaneously in order to calculate the marked 5 entries of the Jacobian.}$$

$$r(x+p)_{1,2} = r(x + \epsilon(e_1 + e_n))_{1,2} = r(x + \epsilon e_1)_{1,2}$$

$$r(x+p)_{3,4,5} = r(x + \epsilon(e_1 + e_n))_{3,4,5} = r(x + \epsilon e_n)_{3,4,5}$$

We can use $p = \epsilon(e_2 + e_5)$ to estimate 2nd and 5th columns in a similar manner, while $p = \epsilon(e_3 + e_6)$ will give 3rd & 6th.
Fact: In the above example, for any n , 3 evaluations of r at perturbed values is sufficient to get the entire Jacobian.

Approximating the Hessian

The techniques discussed above for approximating the Jacobian matrix could be applied to the analytical gradient expression in order to get a nonsymmetric Hessian approximation. Such an estimate could be made symmetric by averaging it with its transpose.

Methods that take the symmetry of $\nabla^2 f(x)$ into account also exist. Some important algorithms, such as the Newton-CG methods discussed in Chapter 7 in the context of large-scale problems, require Hessian-vector products: $\nabla^2 f(x) p$. When $\nabla^2 f(x)$ exists and is Lipschitz cont. near x , we can use Taylor's thm:

$$\nabla f(x + \epsilon p) = \nabla f(x) + \epsilon \nabla^2 f(x)p + O(\epsilon^2)$$

so that $\nabla^2 f(x)p \approx \frac{\nabla f(x + \epsilon p) - \nabla f(x)}{\epsilon}$ with an error of $O(\epsilon)$ and a cost of one additional gradient. A central-difference version could also be derived. If gradients are not available, then

$$\frac{\partial^2 f}{\partial x_i \partial x_j}(x) = \frac{f(x + \epsilon e_i + \epsilon e_j) - f(x + \epsilon e_i) - f(x + \epsilon e_j) + f(x)}{\epsilon^2} + O(\epsilon)$$

Approximating a Sparse Hessian

The numerical approximation can be skipped for those (i, j) entries of the Hessian, which we know is zero.

Example: $f(x) = x_1 \sum_{i=1}^n i^2 x_i^2$ has a Hessian that is non-zero on the 1st column, 1st row, and the main diagonal.

We could use $p = \epsilon e_1$ to estimate the first column of the Hessian. Due to symmetry, these should help with the first row, as well.

Also, $p = \epsilon(e_2 + \dots + e_n)$ helps us identify the diagonal entries except $(1,1)$. With two additional gradient evaluations, the Hessian can be estimated.

Automatic Differentiation

This refers to the use of the computational representation of a function to produce analytic values for derivatives. Variations of this approach exhibit themselves in neural network literature and linear system sensitivity analysis (e.g. dual systems).

Complicated functions are usually composed of successively applied simpler functions that have a few arguments.

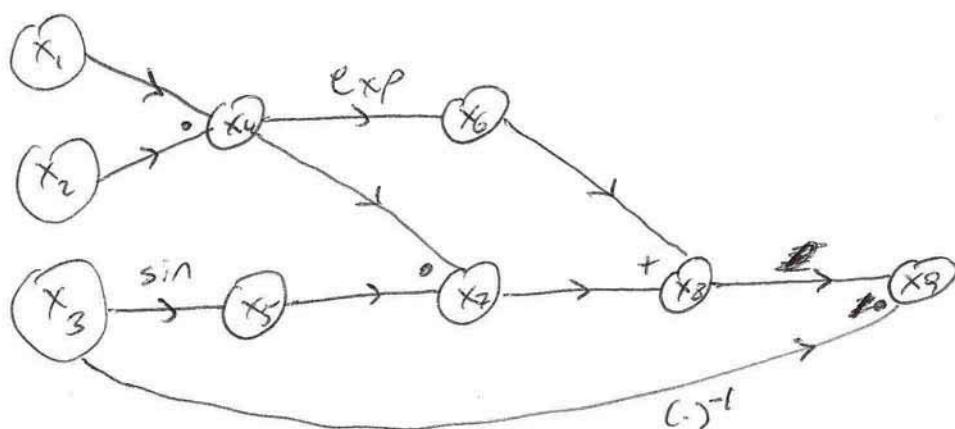
The chain rule can be exploited to compute such derivatives. Specifically, chain rule states:

$$\nabla_x h(y(x)) = \sum_{i=1}^m \frac{\partial h}{\partial y_i}(y(x)) \nabla y_i(x) \\ = J_h(y(x)) J_y(x)$$

Example Consider a function of three variables:

$$f(x) = (x_1 x_2 \sin x_3 + e^{x_1 x_2}) / x_3$$

This function can be represented by the following sequence of elementary operations and the corresponding graph.



$$\begin{aligned} x_4 &= x_1 x_2 \\ x_5 &= \sin x_3 \\ x_6 &= e^{x_4} \\ x_7 &= x_6 x_5 \\ x_8 &= x_6 + x_7 \\ x_9 &= x_8 / x_3 \end{aligned}$$

When x_1, x_2, x_3 are all known, this graph of computations flow following the arrows (generating values for child nodes from their respective parent nodes), ending up with a value $\stackrel{\text{for}}{=} x_9 = f(x)$.

The Forward Mode

We evaluate and carry forward a directional derivative of each intermediate variable x_i in a given direction $p \in \mathbb{R}^n$. The value of x_i is also carried forward.

$$D_p x_i \stackrel{\Delta}{=} (\nabla x_i)^T p = \sum_{j=1}^n \frac{\partial x_i}{\partial x_j} p_j \quad i = 1, 2, \dots, N_{\text{nodes}}$$

$N_{\text{nodes}} = 9$ (in the example)
 $n = 3$

▷ above denotes the gradient w.r.t. $(x_1, \dots, x_n)^T$, the free variables of the function. The goal is to evaluate

$$D_{\underbrace{p}_{N_{\text{nodes}}}} x_{N_{\text{nodes}}} = \nabla f(x)^T p. \text{ Let } N = N_{\text{nodes}} \text{ below...}$$

Given the values and directional derivatives at all parent nodes of a child, using the chain rule the value and directional derivative of the child node can be calculated. For example, above we had

$$x_7 = x_4 x_5 \quad \overbrace{\qquad\qquad\qquad}^{\longrightarrow}$$

$$\nabla x_7 = \frac{\partial x_7}{\partial x_4} \nabla x_4 + \frac{\partial x_7}{\partial x_5} \nabla x_5 = x_5 \nabla x_4 + x_4 \nabla x_5$$

$$\Rightarrow D_p x_7 = x_5 D_p x_4 + x_4 D_p x_5$$

$$\text{Also consider } x_9 = x_8 x_3^{-1}$$

$$\Rightarrow \nabla x_9 = \frac{\partial x_9}{\partial x_8} \nabla x_8 + \frac{\partial x_9}{\partial x_3} \nabla x_3 = x_3^{-1} \nabla x_8 - x_8 x_3^{-2} \nabla x_3$$

$$\Rightarrow D_p x_9 = x_3^{-1} D_p x_8 - x_8 x_3^{-2} D_p x_3$$

The Reverse Mode

In this approach, once $f^{(x)}$ is obtained after a forward sweep of the graph, all partial derivatives are obtained in a backward sweep, from which the gradient is constructed.

The computational graph consists of N nodes of which the first n (nodes $1-n$) are the indep. variables and x_N is the function value. For any node i , the partial derivative $\frac{\partial f}{\partial x_i}$ can be built from $\frac{\partial f}{\partial x_j}$ where x_j 's are child nodes

$$\frac{\partial f}{\partial x_i} = \sum_{j \in \text{Childs}_i} \frac{\frac{\partial f}{\partial x_j}}{\frac{\partial x_j}{\partial x_i}}$$

Let \bar{x}_i be a scalar associated with node x_i . The variable \bar{x}_i is called the adjoint of x_i . We initialize $\bar{x}_i = 0$ $\forall i$ except $\bar{x}_N = \frac{\partial f}{\partial x_N} = 1$. In the reverse sweep, as soon as the child derivative contribution $\frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i}$ becomes known, we increment $\bar{x}_i = \bar{x}_i + \frac{\partial f}{\partial x_j} \frac{\partial x_j}{\partial x_i}$. When all children contribute, we set $\frac{\partial f}{\partial x_i} = \bar{x}_i$ and declare \bar{x}_i as finalized.

In order to perform the backward sweep, we need $\frac{\partial x_j}{\partial x_i}$ and these are computed in the forward sweep and stored in order to be used later in the backward sweep.

The reverse mode trades-off storage for reduced arithmetic operations compared to the forward mode.

Vector Functions and Partial Separability

Consider functions of the form $r: \mathbb{R}^n \rightarrow \mathbb{R}^m$. In this case the computational graph consists of m nodes without any children.

Notice that if a scalar function $f(x)$ is partially separable: $f(x) = \sum_{i=1}^{n_e} f_i(x)$, then we can define $r(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_{n_e}(x) \end{bmatrix}$ and once $J_r(x)$ is determined, we can obtain $\nabla f(x) = J(x)^T \mathbf{1}$, where $\mathbf{1}$ is a vector of ones with n_e entries. Due to the sparsity of $J(x)$ which arises from the partially sep. nature of $f(x)$, we can calculate $J(x)$ and $\nabla f(x)$ in an efficient manner.

Similarly, in constrained optimization problems, the objective and the constraints can be concatenated in order to exploit common nodes/parties in their computational graphs: $r(x) = \begin{bmatrix} f(x) \\ c_i(x) \end{bmatrix}$

Calculating the Jacobians

The forward mode calculation using the computational graph is the same for vector functions. Once all directional derivatives at nodes without children (w.l.o.g. assume these are nodes $N-m+1, \dots, N$), we can construct $J(x) = \begin{bmatrix} D_p \times_{N-m+1} \\ \vdots \\ D_p \times_N \end{bmatrix}$. Seed director vectors p can be selected as $e_i, i=1, \dots, n$ or their linear combinations as discussed in the finite-difference methods for sparse Jacobians.

For reverse mode, consider $\nabla(r^T(x)q) = J_r^T(x)q$. For a given seed vector q , one can apply the reverse mode algorithm to the scalar function $r^T(x)q$ to get $J_r^T(x)q$. The seeds can be e_1, \dots, e_m to get J row-by-row, or q can be linear combinations for sparse J , as discussed before.

Calculating the Hessian

Forward Model: For a given pair of vectors p & q , define $D_{pq} x_i \triangleq p^T \nabla^2 x_i q$. For nodes x_i , $i=1, \dots, n$ corresponding to independent variables, $D_{pq} x_i = 0$ initially. (think of this like $\nabla^2 \triangleq \nabla \nabla^T$)

Examples

$$\begin{aligned} x_i &= x_j + x_k \\ \left(\begin{array}{l} \nabla x_i = \nabla x_j + \nabla x_k \\ \nabla^2 x_i = \nabla^2 x_j + \nabla^2 x_k \end{array} \right) \\ \Rightarrow D_p x_i &= D_p x_j + D_p x_k \\ D_{pq} x_i &= D_{pq} x_j + D_{pq} x_k \end{aligned}$$

$$\begin{aligned} x_i &= x_j x_k \\ \nabla x_i &= \nabla x_j x_k + x_j \nabla x_k \\ \nabla^2 x_i &= \nabla^2 x_j x_k + \nabla x_j \nabla x_k^T \\ &\quad + \nabla x_k \nabla x_j^T + x_j \nabla^2 x_k \\ \Rightarrow D_p x_i &= x_k D_p x_j + x_j D_p x_k \\ D_{pq} x_i &= x_k D_{pq} x_j + D_p x_j D_q x_k \\ &\quad + D_p x_k D_q x_j + x_j D_{pq} x_k \end{aligned}$$

$$x_i = h(x_j) \quad h: \mathbb{R} \rightarrow \mathbb{R}$$

$$\nabla x_i = h'(x_j) \nabla x_j \text{ and } \nabla^2 x_i = h''(x_j) \nabla x_j \nabla x_j^T + h'(x_j) \nabla^2 x_j$$

$$\Rightarrow D_p x_i = h'(x_j) D_p x_j \text{ and } D_{pq} x_i = h''(x_j) D_p x_j D_q x_j + h'(x_j) D_{pq} x_j$$

\therefore Since $D_{pq} x_i$ relies on $D_p x_j$, $D_q x_j$ and $D_{pq} x_j$ (besides $h'(x_j)$ & $h''(x_j)$) in the forward sweep all of these quantities must be computed and stored.

To evaluate the whole Hessian matrix, we need to set the pair $\{p, q\}$ to all possible pairs $\{e_i, e_j\}$, of which we have $\binom{n}{2} = \frac{n(n+1)}{2}$. For sparse Hessians, we only consider the pairs for nonzero entries.

If we need to evaluate $N_2(\nabla^2 f)$ entries of the Hessian, then for each of these, $x_i, D_{e_j} x_i$ ($j=1, \dots, n$) and $D_{e_j e_k} x_i$ need to be evaluated. The total increase factor for the number of arithmetic operations compared to the baseline of computing f alone is $\alpha(1 + n N_2(\nabla^2 f))$, where α is a small (>1) factor that depends on how many additional operations updating terms like $D_p x_i$ and $D_{pq} x_i$ require.

One storage location per node of the graph is needed to store all of these quantities. If we only need a Hessian-vector product $\nabla^2 f(x) q$, then we can simply consider the vector pairs $\{e_j, q\}$ $j=1, \dots, n$:

$$e_j^T \nabla^2 f(x) q = [\nabla^2 f(x) q]_j, \quad j=1, \dots, n$$

This, of course, requires fewer operations overall.

An alternative approach to the forward sweep evaluation of the Hessian is to note the following identity, and exploit it:

$$\begin{aligned} [\nabla^2 f(x)]_{ij} &= e_i^T \nabla^2 f(x) e_j \\ &= \frac{1}{2} \left[(e_i + e_j)^T \nabla^2 f(x) (e_i + e_j) - e_i^T \nabla^2 f(x) e_i - e_j^T \nabla^2 f(x) e_j \right] \end{aligned}$$

This expression allows us to avoid any cross terms and by only considering $p = e_i$, $p = e_j$ and $p = e_i + e_j$ for $i, j \in \{1, \dots, n\}$ (possibly $i=j$), we can obtain

$D_{pp} x_k$ for all nodes x_k . Each $D_{pp} x_k$ is a function of $x_k, D_p x_k, D_{pp} x_k$ for all parent nodes of x_k .

Also, for a univariate function $h: \mathbb{R} \rightarrow \mathbb{R}$, we have $h(t) = f(x + tp) \Rightarrow D_p f = p^T \nabla f(x) = h'(t)|_{t=0}$ and $D_{pp} f = p^T \nabla^2 f(x)p = h''(t)|_{t=0}$.

Reverse Model: Notice that $[\nabla^2 f(x)_q]_i = \frac{\partial}{\partial x_i} [\nabla f(x)^T q]$

Therefore, using the forward sweep we can evaluate $\frac{\partial}{\partial x_i} [\nabla f(x)^T q]$ for $i=1, \dots, n$. Both f and $\nabla f(x)^T q$ in order to accumulate x_i and $D_q x_i$. Then, we apply the reverse sweep to $\nabla f(x)^T q$ as usual

to obtain $\frac{\partial}{\partial x_i} [\nabla f(x)^T q]$ at the indep. variable nodes

$i=1, \dots, n$. If the entire Hessian $\nabla^2 f(x)$ is needed, we employ this procedure for $q = e_1, e_2, \dots, e_n$. In this case, the computational load is increased by at most a factor of $12n$.

Chapter 9: Derivative-Free Optimization

If one needs to optimize a function whose derivatives are not available for some reason, the obvious idea to try is to use finite difference approximations as described above. However, if function evaluations are not accurate (due to finite precision or algorithm noise) then these approaches may not be robust.

Methods that use models constructed from samples, which are then optimized within a trust region can be contemplated. Other methods that use simplex-reflection ideas (such as the Nelder-Mead method that Matlab's "fminsearch" employs) also exist.

We will restrict our discussion to the unconstrained case $\min_{x \in \mathbb{R}^n} f(x)$. Problems where derivatives are not available arise frequently. One case is, for instance, regularization coefficient optimization using cross-validation, in machine learning.

Finite Differences and Noise

Noise in the evaluation of $f(x)$ could arise due to a number of reasons : (i) $f(x)$ is evaluated using a stochastic simulation, (ii) iterative numerical solvers for integrals, differential equations are used to obtain $f(x)$ with nonzero error tolerance.

Suppose $f(x) = h(x) + \phi(x)$ where h is smooth and ϕ represents noise, which could depend on x or not.

Consider the centered finite difference approximation of the gradient :

$$\nabla_{\epsilon} f(x) \triangleq \left[\frac{f(x + \epsilon e_i) - f(x - \epsilon e_i)}{2\epsilon} \right]_{i=1,\dots,n}.$$

The gradient we wish to approximate is $\nabla h(x)$.

Define $\eta(x; \epsilon) \triangleq \sup_{\|z-x\|_\infty \leq \epsilon} |\phi(z)|$ as the largest value of noise in a box of length 2ϵ centered at x .

Lemma 9.1: Suppose that $\nabla^2 h$ is Lipschitz continuous in a neighborhood of the box $\{z \mid \|z-x\|_\infty \leq \epsilon\}$ with Lipschitz constant L_h . Then we have

$$\|\nabla_{\epsilon} f(x) - \nabla h(x)\|_\infty \leq L_h \epsilon^2 + \frac{\eta(x; \epsilon)}{\epsilon}$$

finite-difference
error ↑
noise error ↑

If the noise dominates the difference interval, then the vector $\nabla_{\epsilon} f(x)$ could be oriented in an arbitrary direction and for it to yield a descent direction w.r.t. $-\nabla h(x)$ will be dependent on pure luck.

Model-based Methods

We will construct a quadratic model of f using samples around the current iterate. Suppose that at iterate x_k we obtain a set of sample points $Y = \{y^1, \dots, y^q\}$ where $y^i \in \mathbb{R}^n$, $i=1, \dots, q$. Assume that $x_k \in Y$ and that $f(x_k) \leq f(y^i) \quad \forall i \in \{1, \dots, q\}$. Consider a quadratic model of the form $m_k(x_k + p) = c + g^T p + \frac{1}{2} p^T G p$. The model coefficients c, g, G are obtained by curve-fitting.
(Interpolation).

$$m_k(y^l) = f(y^l) \quad l=1, 2, \dots, q$$

Since G is symmetric, there are $1+q+\frac{q(q+1)}{2} = \frac{1}{2}(q+1)(q+2)$ coefficients in the model. Therefore we need $q = \frac{1}{2}(n+1)(n+2)$ sample points to uniquely determine these coefficients. The samples must be selected to make the linear system of eqns. nonsingular.

Once m_k is formed, we solve the trust-region^{sub} problem:

$$\min_{p} m_k(x_k + p) \text{ s.t. } \|p\|_2 \leq \Delta$$

If $x_k + p$ gives a sufficient reduction in the objective function, the new iterate is defined as $x_{k+1} = x_k + p$, and the trust-region is updated by changing Δ as appropriate.

By iteratively updating the model, we could reduce the computations needed relative to computing m_k from scratch. As usual, for a trial point x_k^+ , the ratio of actual-to-predicted reduction will be used:

$$\rho_k = \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}$$

In the following let $q = \frac{1}{2}(n+1)(n+2)$ be the number of model coefficients as described above.

Alg 9.1: Model-based Derivative-free Method

Choose $Y = \{y_1, \dots, y_q\}$ such that $m_k(y^l) = f(y^l)$ $l=1, \dots, q$ provides a nonsingular linear system of equations. Select $x_0 \in Y$ & $f(x_0) \leq f(y^l) \forall l \in \{1, \dots, q\}$. Choose Δ_0 and $\eta \in (0, 1)$. set $k \leftarrow 0$.

Repeat until convergence

Form the quadratic model $m_k(x_k + p)$ that satisfies the interpolation conditions. Compute p by approximately solving the trust-region subproblem.

Define $x_k^+ = x_k + p$ as the trial point.

Compute ρ , the actual-to-predicted reduction ratio

if $\rho \geq \eta$

Replace one element of \mathbf{Y} by x_k^+ ;

Choose $\Delta_{k+1} \geq \Delta_k$;

Set $x_{k+1} \leftarrow x_k^+$;

Set $k \leftarrow k+1$ and go to the next iteration;

else if the set \mathbf{Y} need not be improved (i.e. cond. number of matrix is fine)

Choose $\Delta_{k+1} < \Delta_k$;

Set $x_{k+1} \leftarrow x_k$;

Set $k \leftarrow k+1$ and go to the next iteration;

end(if)

Replace at least one point in \mathbf{Y} such that the condition number of the linear system matrix in the interpolation problem is improved.

Set $\Delta_{k+1} \leftarrow \Delta_k$

Choose \hat{x} as the element of \mathbf{Y} with lowest function value.

Set $x_k^+ \leftarrow \hat{x}$ and recompute ρ ;

if $\rho \geq \eta$, set $x_{k+1} \leftarrow x_k^+$; else set $x_{k+1} \leftarrow x_k$; end(if)

Set $k \leftarrow k+1$;

end (repeat)

In this algorithm if sufficient reduction ρ is obtained in the first trial point x_0^+ after the first ~~region~~^{opt} solution is obtained, then we accept it and proceed.

When $p < n$, we look at possible causes.

- (i) inadequacy of the interpolation set \mathcal{Y} ,
- (ii) a trust region that is too large.

* The first condition can arise when the samples are restricted to a lower dimensional subspace of \mathbb{R}^n . To avoid this, we monitor the conditioning of the system of linear eqns.

* If \mathcal{Y} seems to be adequately conditioned, then we decrease the trust-region radius.

* A good choice for \mathcal{Y} is the vertices and midpoints of edges and surfaces of a simplex in \mathbb{R}^n .

* The algorithm needs $\mathcal{O}(n^2)$ function evaluations just to initialize the quadratic model to get started. This limits the application to small-scale problems. Further, updating the model is at least $\mathcal{O}(n^3)$ in complexity.

One could set $G=0$, and switch to using a linear model which simplifies the trust-region subproblem as well. (Cauchy step) Then the cost of each iteration is $\mathcal{O}(n^3)$. This prevents rapid convergence!

Interpolation and Polynomial Bases

Consider a linear model: $m_k(x_k + p) = f(x_k) + g^T p$.

Let $s^l = y^l - x_k$, $l=1, 2, \dots, n$. Then to determine g :

$$(s^l)^T g = f(y^l) - f(x_k) \quad l=1, \dots, n$$

If $\{s^l\}_{l=1}^n$ is a linearly independent set of vectors, then the simplex whose vertices are $\{x_k, y^1, \dots, y^n\}$ is called non-degenerate. In this case, g is uniquely determined.

Now consider a quadratic model:

$$\begin{aligned} m_k(x_k + p) &= f(x_k) + g^T p + \sum_{i < j} G_{ij} p_i p_j + \frac{1}{2} \sum_i G_{ii} p_i^2 \\ &\stackrel{\Delta}{=} f(x_k) + \hat{g}^T \hat{p} \end{aligned}$$

where \hat{g} is the vector consisting of g and unique entries of G , while \hat{p} is the vector containing their multipliers in the quadratic model.

$$\hat{g} \stackrel{\Delta}{=} \left[\begin{array}{c} \overline{g} \\ \hline \frac{1}{2} G_{ij} \\ \hline \frac{1}{2} G_{ii} \end{array} \right], \quad \hat{p} \stackrel{\Delta}{=} \left[\begin{array}{c} p \\ \hline p_i p_j \\ \hline \frac{1}{2} p_i^2 \end{array} \right]$$

This representation of the multivariate quadratic uses the so-called monomial-basis. There, structure in the Hessian (e.g. sparsity) can be easily imposed by setting appropriate elements of G to zero. However, singularity of the system may not be easily avoided / controlled. Consider an alternative basis:

$$m_k(x) = \sum_{i=1}^q \alpha_i \phi_i(x)$$

where $\{\phi_i(x)\}_{i=1}^q$ is a basis for the linear space of n -dimensional quadratic polynomials. The interpolation set $\Upsilon = \{y^1, \dots, y^q\}$ determines $\{\alpha_i\}_{i=1}^q$ uniquely iff

$$S(\Upsilon) \triangleq \det \begin{bmatrix} \phi_1(y^1) & \cdots & \phi_1(y^q) \\ \vdots & & \vdots \\ \phi_q(y^1) & \cdots & \phi_q(y^q) \end{bmatrix} \neq 0.$$

During the iterations, this matrix must be well-conditioned.

Updating the Interpolation Set

Whenever a trial point does not provide sufficient decrease in f , we may invoke a geometry-improving procedure for Υ . The goal is to replace one sample to make $|S(\Upsilon)|$ larger.

For $y \in Y$, define the Lagrangian function $L(\cdot, y)$ to be a polynomial of degree at most 2 such that $L(y, y) = 1$ and $L(\hat{y}, y) = 0$ for $\hat{y} \neq y$, $\hat{y} \in Y$. Suppose that Y is updated by removing y_- and inserting y_+ to get Y^+ .

Then, we have $|s(Y^+)| \leq |L(y_+, y_-)| \cdot |s(Y)|$.

Alg. 9.1 (Model-based Derivative-Free Method) can make use of this inequality.

* Consider the case where x^+ provides $p \geq n$. We include x^+ in Y and remove another point y_- . We could choose $y_- = \operatorname{argmax}_{y \in Y} |L(x^+, y)|$

* Now consider the case when $p < n$. First we check if Y should be improved.

Y is adequate if $\forall y_i \in Y \nexists \|x_k - y_i\| \leq \Delta$, $s(Y)$ cannot be increased (e.g. doubled) by replacing one of these y_i with any point y inside the trust region.

If Y is adequate, then we decrease the trust-region radius.

If γ is inadequate, $f(y^i + \gamma)$, we determine a potential replacement $y_r^i = \arg \max_{\|y - x_k\| \leq \Delta} |L(y, y^i)|$. Then we replace the point for which the replacement improves this measure maximally; the index ^{of the point} to be replaced is

$$i^* = \arg \max_i |L(y_r^i, y^i)|$$

and y^{i^*} is replaced with $y_r^{i^*}$.

* Note that efficient implementation is not trivial.

Minimum-Change Updating of the Model

Inspired by the quasi-Newton methods, an $O(n^3)$ procedure (compared to $O(n^4)$ above) can be developed. As we take a step from x_k to x_{k+1} , the coefficients $f_{k+1}, g_{k+1}, G_{k+1}$ of the model m_{k+1} must be obtained.

Consider $\min_{f, g, G} \|G - G_k\|_F^2$ s.t. G is symmetric
 $m(g^l) = f(g^l) \quad l=1, \dots, p$

Notice that $\hat{q} > (n+1) \Rightarrow G_{k+1} \neq G_k$ (previous linear model cannot be maintained)

One could use, for instance, $\hat{q} = 2n+1$. if that is the case).

This is an equality-constrained quadratic problem.
 (more later).

Coordinate and Pattern Search Methods

Rather than constructing a model of f based on function values, these methods search for a lower function value along a specified direction. If such a point is found, it is accepted and the process is repeated for the next search direction.

Coordinate Search Method

This method cycles through search directions e_1, \dots, e_n and performs the search along each direction to find an acceptable new iterate. Note that at each iteration only one parameter is changed.

In practice, this method could be very inefficient. It can enter into situations (e.g. limit cycles) that prevents it from ever approaching a point where the gradient vanishes.

Fact: Any cyclic search along any set of linearly independent ~~#~~ directions could suffer from the same issue - these do not guarantee global convergence.

The issue also could arise when $-\nabla f_k$ becomes (increasingly) orthogonal to the search direction. In general, coordinate descent, if it converges to a solution, converges much more slowly than steepest descent.

A variant of coordinate descent that allows for global convergence is the back-and-forth approach in which the search direction sequence is $\{e_1, \dots, e_n, -e_1\}$. Another variant performs one search along the direction $p = x_n - x_i$ after a sequence of updates along e_{n+1}, \dots, e_n . This makes use of some local structure in e_1, \dots, e_n . This makes use of some local structure in the function.

Pattern Search Methods

These generalize coordinate search by allowing the use of a richer set of search directions. At iterate x_k assume that there is a set of search directions D_k and the step length is δ_k . The frame of search vectors/points consist of $x_k + \delta_k p_k \quad \forall p_k \in D_k$.

If one of these frame points yield a significant decrease in f , we take that step and increase δ_k . If none provide a significantly better f , we reduce δ_k and repeat the search. In either case, the frame D_k can be modified (e.g. directions changed).

Alg. 9.2 Pattern Search

Given tolerance δ_{tol} , contraction parameter θ_{max} , sufficient decrease function $\rho : [0, \infty) \rightarrow \mathbb{R}$ with $\rho(t)$ increasing w.r.t. t , $\rho(t)/t \xrightarrow[t \rightarrow 0]{} 0$.

Choose x_0 , $\delta_0 > \delta_{tol}$, D_0 ;

for $k=1, 2, \dots$

if $\delta_k \leq \delta_{tol}$

stop;

if $f(x_k + \delta_k p_k) < f(x_k) - \rho(\delta_k)$ for some $p_k \in D_k$

Set $x_{k+1} \leftarrow x_k + \delta_k p_k$ for some such p_k

Set $\delta_{k+1} \leftarrow \phi_k \delta_k$ for some $\phi_k \geq 1$;

else

Set $x_{k+1} \leftarrow x_k$

Set $\delta_{k+1} = \theta_k \delta_k$, where $0 < \theta_k \leq \theta_{max} < 1$;

end (if)

end (for)

(184)

The set D_k should be selected such that at least one direction in the set is a descent direction, whenever $\nabla f(x_k) \neq 0$. Recall the angle between $-\nabla f_k$ & p :

$$\cos \theta = \frac{-\nabla f_k^T p}{\|\nabla f_k\| \cdot \|p\|}$$

Thm 3.2 guarantees the global convergence of a line search method if the search directions at each x_k satisfies $\cos \theta_k \geq \delta > 0$, and if the step length s satisfies $\cos \theta_k \geq \delta > 0$, certain conditions. For the construction of the frame D_k we ensure that the following condition holds:

$$K(D_k) \stackrel{\Delta}{=} \min_{v \in \mathbb{R}^n} \max_{p \in D_k} \frac{v^T p}{\|v\| \cdot \|p\|} \geq \delta.$$

This ensures that we have at least one descent direction. The vectors $p \in D_k$ must also roughly have equal length so that β_k captures the radius of the frame. Thus, we impose that, for some $0 < \beta_{\min} < \beta_{\max}$

$$\beta_{\min} \leq \|p\| \leq \beta_{\max} \quad \forall p \in D_k$$

If these two conditions hold, for any k , we have

$-\nabla f_k^T p \geq K(D_k) \|\nabla f_k\| \cdot \|p\| \geq \delta \beta_{\min} \|\nabla f_k\|$ for some $p \in D_k$ i.e. that p is a descent direction.

Examples $D = \{e_1, e_2, \dots, e_n, -e_1, \dots, -e_n\}$

$$\text{or } D = \left\{ P_i \mid P_{n+1} = \frac{\mathbf{1}}{2^n}, P_i = \frac{\mathbf{1}}{2^n} - e_i \text{ } i=1, \dots, n \right\}$$

where $\mathbf{1}$ is the vector of ones. In the second set, the vectors go from the origin to the vertices of a simplex in n -dimensional space.

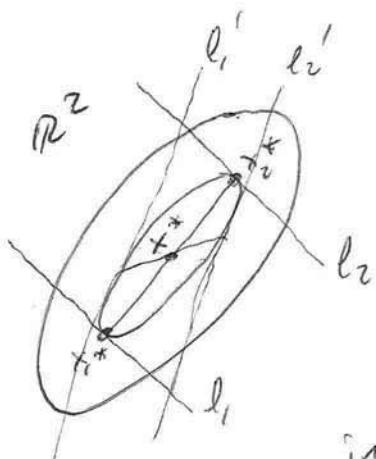
An example for the sufficient decrease function is $p(t) = M t^{3/2}$ where $M > 0$ so that the function enforces a larger decrease when a larger step is attempted.

Conjugate-Direction Method

Recall that the minimizer of a strictly convex quadratic function $f(x) = \frac{1}{2} x^T A x + b^T x$ can be located by performing one-dimensional optimization in n conjugate directions.

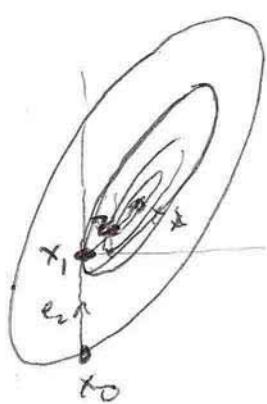
Parallel Subspace Property: Consider two parallel

lines $l_1(\alpha) = x_1 + \alpha p$ and $l_2(\alpha) = x_2 + \alpha p$, where x_1, x_2 , and p are in \mathbb{R}^n and $\alpha \in \mathbb{R}$.



If x_1^* and x_2^* denote the minimizers of f on l_1 and l_2 , then the solution x^* lies on the line segment joining x_1^* to x_2^* as shown in the illustration. The vector $(x_2^* - x_1^*)$ is conjugate to p , w.r.t. A .

This points to the following approach in \mathbb{R}^2 : For any x_0 , select two linearly independent directions (e.g. e_1 and e_2 for simplicity).



Starting at x_0 , minimize f along e_2 to get x_1 . Then, perform optimizations along e_1 and then e_2 to obtain z . Since x_1 and z are minimizers along two parallel lines, $(z - x_1)$ is conjugate to e_2 . Then from x_1 , a minimization along $(z - x_1)$ yields x^* .

Generalized Parallel Subspace Minimization: Let $x_1, x_2 \in \mathbb{R}^n$ be distinct points and $\{p_1, p_2, \dots, p_l\}$ is a set of linearly independent directions in \mathbb{R}^n . Define

$$S_1 \stackrel{\Delta}{=} \left\{ x_1 + \sum_{i=1}^l \alpha_i p_i \mid \alpha_i \in \mathbb{R}, i=1, \dots, l \right\}$$

$$S_2 \stackrel{\Delta}{=} \left\{ x_2 + \sum_{i=1}^l \alpha_i p_i \mid \alpha_i \in \mathbb{R}, i=1, \dots, l \right\}$$

Fact If x_1^* and x_2^* are the minimizers of f on S_1 and S_2 , respectively, then $x_2^* - x_1^*$ is conjugate to P_1, P_2, \dots, P_l .

Proof: Note that at x_i^* we have

$$\left. \frac{\partial f(x_i^* + \alpha_i P_i)}{\partial \alpha_i} \right|_{\alpha_i=0} = \nabla f(x_i^*)^T P_i = 0 \quad i=1, \dots, l$$

A similar expression holds for x_2^* . Therefore,

$$\begin{aligned} 0 &= (\nabla f(x_1^*) - \nabla f(x_2^*))^T P_i \\ &= (A x_1^* - b - A x_2^* + b)^T P_i \\ &= (x_1^* - x_2^*)^T A P_i \quad i=1, \dots, l \end{aligned}$$

Example: Consider \mathbb{R}^3 . Select e_1, e_2, e_3 as the lin. indep. directions. Starting at any x_0 , minimize f along e_3 to obtain x_1 . Then perform successive minimizations along e_1, e_2 , and e_3 starting from x_1 . The resulting point is. Clearly $P_1 = z - x_1$ is conjugate to e_3 and minimizing along P_1 starting from x_1 yields x_2 . Note that x_2 is the minimizer of f on $S_1 = \{y + \alpha_1 e_3 + \alpha_2 P_1 \mid \alpha_1, \alpha_2 \in \mathbb{R}\}$, where y is the point obtained after minimizing along e_1, e_2 .

We discard e_1 and use e_2, e_3, p_1 as the new set of lin. indep. directions. Starting from x_2 and sequentially minimizing along e_2, e_3, p_1 , we get \hat{z} .

Note that $p_2 = \hat{z} - x_2$ is conjugate to both e_3 and p_1 ; also \hat{z} is the minimizer of f on $S_2 = \{ \hat{y} + \alpha e_3 + \alpha_2 p_1 \mid \alpha, \alpha_2 \in \mathbb{R} \}$. We then minimize f along p_2 to obtain x_3 ; x_3 is the minimizer of f . The three conjugate directions used are e_3, p_1, p_2 .

Alg. 9.3 DFO Method of Conjugate Directions

Choose an initial x_0 ; set $p_i = e_i$ for $i=1, \dots, n$.

Compute x_1 as the minimizer of f along $x_0 + \alpha p_1$.

Set $k \leftarrow 1$;

repeat until convergence

 Set $z_i \leftarrow x_k$;

 Generates for $j=1, 2, \dots, n$, calculate $\alpha_j \geq f(z_j + \alpha_j p_j)$ is minimized;
 a new conjugate direction. Set $z_{j+1} \leftarrow z_j + \alpha_j p_j$; end(for)

 Set $p_j \leftarrow p_{j+1}$ for $j=1, 2, \dots, (n-1)$ and $p_n \leftarrow z_{n+1} - z_1$;

 Calculate $\alpha_n \geq f(z_{n+1} + \alpha_n p_n)$ is minimized;

 Set $x_{k+1} \leftarrow z_{n+1} + \alpha_n p_n$; set $k \leftarrow k+1$;

end (repeat)

Any suitable line search strategy can be used.

If f is strongly convex quadratic, then the quadratic polynomial fitting approach will yield exact line search.

The algorithm will converge in $(n-1)$ iterations and will perform $\mathcal{O}(n^2)$ function evaluations.

* When applied to nonlinear problems, the generated conjugate directions may approach towards linear dependency. If we define $\hat{P}_i \triangleq \frac{P_i}{(P_i^T A P_i)^{1/2}} \quad i=1, \dots, n$,

then $|\det(\hat{P}_1, \dots, \hat{P}_n)|$ is maximized iff $\{P_i\}$ are conjugate w.r.t. A . Consequently, if the most recently generated direction reduces this quantity, it should not replace any of the P_i 's, as that would prevent conjugacy. The following procedure can be invoked after the inner for-loop in Alg. P-3 to prevent this undesirable situation from occurring.

This procedure can be applied to general objective functions by implementing next line search process.

Procedure 9.4 : Updating of the set of Directions

Find $m \in \{1, 2, \dots, m\}$ s.t. $\psi_m = f(x_{m+1}) - f(x_m)$ is max.

Let $f_1 = f(z_1)$, $f_2 = f(z_{n+1})$, $f_3 = f(z_{n+1} - z_1)$.

If $f_3 \geq f_1$ or $(f_1 - 2f_2 + f_3)(f_1 - f_2 - \psi_m) \geq \frac{1}{2} \psi_m (f_1 - f_3)^2$

Keep the set P_1, P_2, \dots, P_n unchanged; Set $x_{k+1} \leftarrow z_{n+1}$

else

Set $\hat{p} \leftarrow z_{n+1} - z_1$ and calculate $\hat{z} = \arg \min_z f(z_{n+1} + \lambda \hat{p})$;

Set $x_{k+1} \leftarrow z_{n+1} + \lambda \hat{p}$;

Remove P_m from the set and add \hat{p} to the set.

end(if)

Nelder-Mead Method (Simplex-reflection)

At any step, we have $(n+1)$ points in \mathbb{R}^n and their convex hull forms a simplex (generalized/hyper-triangle). Given a simplex S with vertices $\{z_i, -z_{n+1}\}$ we define the matrix $V(S) \stackrel{\Delta}{=} [z_2 - z_1, z_3 - z_1, \dots, z_{n+1} - z_1]$.

Defn: The simplex S is nondegenerate or nonsingular

iff V is a nonsingular matrix.

(This means the hyper-volume of the simplex on \mathbb{R}^n is nonzero.)

In the algorithm, the worst vertex is replaced with a better one using a line search on the ray connecting this vertex to the centroid of the others. If a better value cannot be found, the best vertex is retained and others are moved towards this vertex to shrink the simplex volume.

Suppose that the $(n+1)$ vertices $\{x_1, x_2, \dots, x_{n+1}\}$ are indexed such that $f(x_1) \leq f(x_2) \leq \dots \leq f(x_{n+1})$. The centroid of the best n points is: $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$. Points along the line passing through x_{n+1} and \bar{x} are

$$\bar{x}(t) = \bar{x} + t(x_{n+1} - \bar{x})$$

Procedure 9.5 One Step of Nelder-Mead Simplex

Compute $\bar{x}(-1)$ and $f_{-1} = f(\bar{x}(-1))$.

If $f(x_1) \leq f_{-1} < f(x_n)$ % reflected point is moderate
replace x_{n+1} by $\bar{x}(-1)$ and go to next iteration.

else if $f_{-1} < f(x_1)$ % reflected point is best; try to go further.

Compute $\bar{x}(-2)$ and $f_{-2} = f(\bar{x}(-2))$

if $f_{-2} < f_{-1}$, replace x_{n+1} by \bar{x}_{-2} and go to next iteration.
else replace x_{n+1} by x_{-1} and go to next iteration; endif()

else if $f_{-1} \geq f(x_1)$ % reflected point not good enough, contract

if $f(x_1) \leq f_{-1} \leq f(x_{n+1})$

Evaluate $f_{-1/2} = \bar{x}(-1/2)$ % outside contraction

If $f_{-1/2} \leq f_{-1}$, Replace x_{n+1} by $x_{-1/2}$, go to next iter; end if

else

Evaluate $f_{1/2} = \bar{x}(1/2)$ % inside contraction

If $f_{1/2} \leq f_{n+1}$, replace x_{n+1} by $x_{1/2}$, go to next iter; end if

end (if)

Replace $x_i \leftarrow \frac{1}{2}(x_i + x_{n+1})$ for $i=2, 3, \dots, n+1$

% if alg did not go to next iter, shrink simplex towards best vertex.

end (if).

Fact: All steps except the final shrinkage (last resort) are guaranteed to ~~not increase~~ decrease the average function value

$$\bar{f} = \frac{1}{n+1} \sum_{i=1}^{n+1} f(x_i).$$

If f is convex, then even the shrinkage step is guaranteed to ~~not increase~~ -

Implicit Filtering

This is a variant of steepest descent using coarse finite difference estimates.

Alg. 9.6 Implicit Filtering

Choose a sequence $\{\epsilon_k\} \downarrow 0$, parameters $c, \rho \in (0, 1)$, α_{\max}

Set $k \leftarrow 1$, choose $x = x_0$.

repeat

increment_k \leftarrow false

 ↓ repeat

 Compute $f(x)$ and $\nabla_{\epsilon_k} f(x)$;

 if $\|\nabla_{\epsilon_k} f(x)\| \leq \epsilon_k$

 increment_k \leftarrow true

 else

 Find the smallest $m \in [0, \alpha_{\max}]$, $m \in \mathbb{Z}$ s.t.

$f(x - \rho^m \nabla_{\epsilon_k} f(x)) \leq f(x) - c \rho^m \|\nabla_{\epsilon_k} f(x)\|_2^2$;

 if no such m exists

 increment_k \leftarrow true

 else

$x \leftarrow x - \rho^m \nabla_{\epsilon_k} f(x)$

 end (if)

 end (if)

 ↓ until increment_k

$x_k \leftarrow x$; $k \leftarrow k+1$

until a termination test is satisfied

Thm 8.2 Suppose that $\nabla^2 h$ is Lipschitz cont,

that Alg 8.6 generates an infinite sequence of iterates $\{x_k\}$, and that $\lim_{k \rightarrow \infty} \epsilon_k^2 + \frac{\gamma(x_k; \epsilon_k)}{\epsilon_k} = 0$.

Also suppose that all but a finite number of inner loops in Alg 8.6 terminate with $\|\nabla_{\epsilon_k} f(x_k)\| \leq \epsilon_k$.

Then all limit points of $\{x_k\}$ are stationary.

Proof: Using $\{\epsilon_k\} \downarrow 0$, and the assumption on inner loop termination, we have $\nabla_{\epsilon_k} f(x_k) \rightarrow 0$. From

Lemma 8.1, we have $\|\nabla_{\epsilon_k} f(x) - \nabla_h(x)\|_\infty \leq L_h \epsilon^2 + \frac{\gamma(x; \epsilon)}{\epsilon}$.

Noting that the right hand side is approaching zero as assumed by the theorem, we conclude $\nabla_h(x_k) \rightarrow 0$. \square .

Q.1) Prove Lemma Q.1.

This lemma assumes that $\nabla^2 h$ is Lipschitz continuous in a neighborhood of the box $\{x \mid \|x - x\|_\infty \leq \epsilon\}$ with L_h . This means $\|\nabla^2 h(x+p) - \nabla^2 h(x)\| \leq L_h \|p\|$ for all p with $\|p\|_\infty \leq \epsilon$. Consider Taylor's theorem applied to $h(x+ee_i)$.

$$h(x+ee_i) = h(x) + e \nabla h(x)^T e_i + \frac{1}{2} \epsilon^2 e_i^T \nabla^2 h(x+t_i ee_i) e_i$$

$$h(x-ee_i) = h(x) - e \nabla h(x)^T e_i + \frac{1}{2} \epsilon^2 e_i^T \nabla^2 h(x-t_i ee_i) e_i$$

for some t_- and $t_+ \in (0, 1)$.

$$h(x+ee_i) - h(x-ee_i) = 2e \frac{\partial h}{\partial x_i}(x) + \frac{1}{2} \epsilon^2 e_i^T [\nabla^2 h(x+t_+ ee_i) - \nabla^2 h(x-t_- ee_i)] e_i$$

$$\text{and } \left| \frac{h(x+ee_i) - h(x-ee_i)}{2\epsilon} - \frac{\partial h}{\partial x_i}(x) \right| = \left| \frac{1}{2} \epsilon^2 e_i^T \{ \nabla^2 h(x+t_i ee_i) e_i \} \right| \leq \frac{1}{2} \epsilon^2 \| \nabla^2 h(x) \| \frac{1}{2\epsilon} \\ \leq \frac{1}{2} \epsilon^2 L_h \frac{2\epsilon}{2\epsilon} = \frac{1}{2} L_h \epsilon^2 \leq L_h \epsilon^2 (*)$$

Now consider

$$\left| \frac{f(x+ee_i) - f(x-ee_i)}{2\epsilon} - \frac{\partial f}{\partial x_i}(x) \right| = \left| \frac{h(x+ee_i) - h(x-ee_i)}{2\epsilon} - \frac{\partial h}{\partial x_i}(x) + \frac{\phi(x+ee_i) - \phi(x-ee_i)}{2\epsilon} \right|$$

$$\leq \left| \frac{h(x+ee_i) - h(x-ee_i)}{2\epsilon} - \frac{\partial h}{\partial x_i}(x) \right| + \frac{1}{2\epsilon} [|\phi(x+ee_i)| + |\phi(x-ee_i)|]$$

$$\leq L_h \epsilon^2 + \frac{\gamma(x; t)}{\epsilon} \quad \forall i=1, \dots, n$$

Hence the lemma is proved. \square

D_k where

Q.10) Prove that the set $P_{n+1} = \frac{1}{\|v\|}$, $P_i = P_{n+1} - e_i$, $i=1, 2, \dots, n$
 where 1 is a vector of ones, satisfies.

$$\min_{v \in \mathbb{R}^n} \max_{P \in D_k} \frac{\sqrt{v^T P}}{\|v\| \cdot \|P\|} \geq s \text{ for some } s > 0$$

for any given P let $f(v) = \frac{\sqrt{v^T P}}{\|v\| \cdot \|P\|}$. The minimizing v must satisfy

$$\nabla f(v) = \frac{P \|v\| \cdot \|P\| - (v^T P) \|P\| \frac{1}{2} (v^T v)^{-1/2} v}{\|v\|^2 \cdot \|P\|^2} = 0$$

~~$$\Rightarrow P \|v\| = (v^T P) \frac{v}{\|v\|} \Leftrightarrow v_* = \left(\frac{v_*^T v_*}{v_*^T P} \right) P. (*)$$~~

Notice that for any v_* that satisfies $(*)$, $-v_*$ also satisfies it.

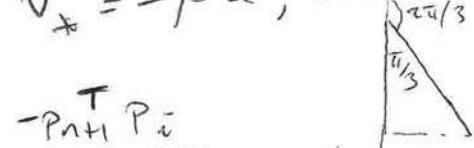
(i.e. one of v_* is the minimizer and the other is a maximizer).
 Then $f(v_*) = \frac{(v_*^T v_*) (P^T P)}{(v_*^T P) - \|v_*\| \cdot \|P\|} = \frac{\|v_*\| \cdot \|P\|}{(v_*^T P)} = \frac{1}{f(-v_*)} \Rightarrow f^2(v_*) = 1$

$\therefore f(v_*) = \pm 1$ (i.e. if $f(v_*) = +1$, then $f(-v_*) = -1$ or
 vice versa ...) Select the sign that yields $f(v_*) = +1$.

This means, for a given P , $v_* = -P\alpha$, where $\alpha > 0$ is
 some scalar.

For P_{n+1} , $v_* = -P_{n+1}$ and $\frac{-P_{n+1}^T P_i}{\|P_{n+1}\| \cdot \|P_i\|} = \cos(\pi/3)$. The same
 angle $\pi/3$ is the case for other choices of $P \in D_k$.

$$\therefore s = \cos\left(\frac{\pi}{3}\right).$$



Chapter 10: Least-Squares Problems

In LS problems, the objective has the following form:

$$f(x) = \frac{1}{2} \sum_{j=1}^m r_j^2(x)$$

where r_j is a smooth function from \mathbb{R}^n to \mathbb{R} .

Each r_j is a residual and we assume $m \geq n$.

Define the residual vector $r: \mathbb{R}^n \rightarrow \mathbb{R}^m$ as follows:

$$r(x) \triangleq \begin{bmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{bmatrix}$$

With this notation, we have $f(x) = \frac{1}{2} \|r(x)\|_2^2$. Using

the Jacobian of $r(x)$:

$$J(x) = \left[\frac{\partial r_j}{\partial x_i} \right]_{\substack{j=1 \dots m \\ i=1 \dots n}} = \begin{bmatrix} \nabla r_1(x)^T \\ \vdots \\ \nabla r_m(x)^T \end{bmatrix}$$

$$\text{we get } \nabla f(x) = \sum_{j=1}^m r_j(x) \nabla r_j(x) = J(x)^T r(x)$$

$$\nabla^2 f(x) = \sum_{j=1}^m \nabla r_j(x) \nabla r_j(x)^T + \sum_{j=1}^m r_j(x) \nabla^2 r_j(x)$$

$$= J(x)^T J(x) + \sum_{j=1}^m r_j(x) \nabla^2 r_j(x)$$

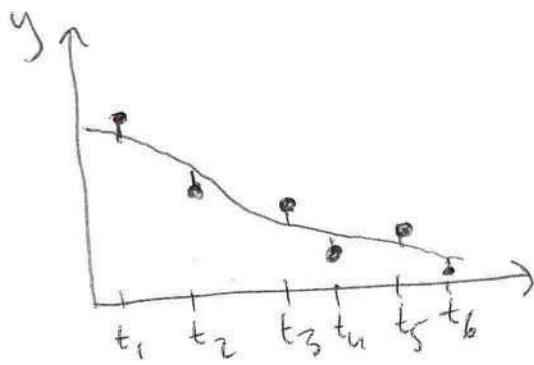
\therefore The evaluation of $J(x)$ gives us $\nabla f(x)$ and the first term in $\nabla^2 f(x)$.

Example: Based on some prior experience/knowledge we know that $\phi(x; t) = x_1 + tx_2 + t^2x_3 + x_4 e^{-x_5 t}$ is a good model of some measured variable as a function of time t (e.g. medication concentration in blood following drug intake, where time is measured in minutes).

Given m measurements of the variable y of interest, we can consider the following least-squares function as a measure of model mismatch:

$$\frac{1}{2} \sum_{j=1}^m [\phi(x; t_j) - y_j]^2$$

Here, the residuals are $r_j(x) = \phi(x; t_j) - y_j$.



This approach (called the fixed-regressor model) assumes that the time instances $\{t_i\}$ are known or measured with very high accuracy and the observations $\{y_i\}$ have random errors due to measurement noise (equipment or human errors).

In general, the ordinate t could be vector valued (e.g. smooth interpolation of image intensity over space). The target variable could also be a vector (e.g. interpolating a color image).

Instead of $\|r(x)\|_2$, one could use other algebraic norms such as $f(x) = \|r(x)\|_\infty$ or $f(x) = \|r(x)\|_1$. These options are discussed later (chapter 17).

Statistical Motivation for $\|\cdot\|_2$: Let $\epsilon_j \stackrel{\Delta}{=} \phi(x; t_j) - y_j$. Assume that $\{\epsilon_j\}_{j=1}^m$ are independent and identically distributed (iid) random variables with variance σ^2 and probability density function (pdf) $g_\sigma(\cdot)$. Under these assumptions, the likelihood is

$$p(y; x, \sigma) = \prod_{j=1}^m g_\sigma(\epsilon_j) = \prod_{j=1}^m g_\sigma(\phi(x; t_j) - y_j)$$

Given the observation vector $y = [y_1 \dots y_m]^T$, the most likely value of x is obtained by maximizing $p(y; x, \sigma)$ with respect to x (maximum likelihood estimation). Now assume that $g_\sigma(\cdot)$ is the 0-mean, σ^2 -variance Gaussian distribution:

$$g_\sigma(\epsilon) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{\epsilon^2}{2\sigma^2}\right)$$

Then

$$p(y; x, \sigma) = (2\pi\sigma^2)^{-m/2} \exp\left(-\frac{1}{2\sigma^2} \sum_{j=1}^m [\phi(x; t_j) - y_j]^2\right).$$

For any fixed value of σ^2 , the likelihood is maximized when the sum-of-squared-errors is minimized.

Fact If measurement errors are Gaussian distributed with zero-mean and σ^2 -variance, where σ is known, then the maximum likelihood estimate of x is the same as the least-squares estimate of x .

Note that if the measurement errors are zero-mean Gaussian with covariance W^{-1} (i.e. errors ^{are} uncorrelated), then we get a weighted-LS problem: $r(x)^T W r(x)$.

Linear LS Problems: $f(x) = \frac{1}{2} \|Jx - y\|_2^2$, $r(x) = Jx - y$.
notice that $r(0) = -y$. We have

$$\nabla f(x) = J^T(Jx - y), \quad \nabla^2 f(x) = J^T J$$

Clearly $f(x)$ is convex (it is a quadratic function with pos. semi-definite Hessian)
so at the global minimizer of f we have

$$J^T J x^* = J^T y$$

These are called the normal equations. If we assume that J has full column rank, then $J^T J$ is full-rank, i.e. it is invertible, hence positive definite.

Cholesky Factorization

(201)

Obvious algorithm: solve $J^T J x^* = J^T y$ as a system of linear equations.

1) Compute $J^T J$ and $J^T y$.

2) Find the Cholesky factorization of $J^T J$.

3) Perform two triangular substitutions to get x^* .

Fact - The Cholesky factorization $J^T J = \bar{R}^T \bar{R}$

where \bar{R} is an $n \times n$ upper triangular positive definite matrix is guaranteed to exist when $m \leq n$ and J has rank n (as discussed above).

Problem: The condition number of $J^T J$ is the square of the condition number of J . If $J^T J$ is poorly conditioned, the Cholesky factorization may be inaccurate.

QR Decomposition

Consider the QR decomposition $J \Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = \{Q_1, Q_2\} \begin{bmatrix} R \\ 0 \end{bmatrix}$

where Π is an $n \times n$ permutation (orthogonal) $= Q_1 R$

Q is $n \times n$ orthogonal

Q_1 is the first n columns of Q

R is $n \times n$ upper triangular, positive definite.

Also note that if Q is orthogonal, then $\|Jx-y\| = \|\alpha^T(Jx-y)\|$.

$$\begin{aligned}
 \text{Then } \|Jx-y\|_2^2 &= \left\| \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} (J\pi\pi^T x - y) \right\|_2^2 \quad (\text{since } \pi\pi^T = I \\
 &= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} (\pi^T x) - \begin{bmatrix} Q_1^T y \\ Q_2^T y \end{bmatrix} \right\|_2^2 \\
 &= \|R\pi^T x - Q_1^T y\|_2^2 + \|Q_2^T y\|_2^2
 \end{aligned}$$

The second term is unaffected by the choice of x .

$$\begin{aligned}
 \text{So } x^* = \arg \min_x \|Jx-y\|_2^2 &= \arg \min_x \|R\pi^T x - Q_1^T y\|_2^2 \dots \\
 x^* &= \pi R^{-1} Q_1^T y
 \end{aligned}$$

Notice that R^{-1} can be obtained using triangular substitution and multiplication with π is a simple permutation of vector entries.

The condition number of J and R^{-1} are comparable so this process does not negatively affect the numerical conditioning of the problem.

SVD Approach

$$\begin{aligned}
 \text{The SVD of } J \text{ is given by } J &= U \begin{bmatrix} S \\ 0 \end{bmatrix} V^T \\
 &= [U_1, U_2] \begin{bmatrix} S \\ 0 \end{bmatrix} V^T \\
 &= U_1 S V^T
 \end{aligned}$$

where U is $m \times m$ orthogonal

U_1 contains the first n columns of U

V is $n \times n$ orthogonal

S is $n \times n$ diagonal with diagonal entries $s_1 \geq s_2 \geq \dots \geq s_n > 0$.

Note that $J^T J = V S^2 V^T$.

$$\|Jx-y\|^2 = \left\| \begin{bmatrix} S \\ 0 \end{bmatrix}(V^T x) - \begin{bmatrix} u_1^T \\ u_2^T \end{bmatrix} y \right\|^2$$

$$= \|SV^T x - u_1^T y\|^2 + \|u_2^T y\|^2$$

The minimizer is found as $x^* = VS^{-1}U_1^T y$.

Let $U = [u_1 \dots u_m]$ and $V = [v_1 \dots v_n]$. Then

$$x^* = \sum_{i=1}^n \frac{u_i^T y}{\sigma_i} v_i$$

When σ_i is small, x^* is sensitive to perturbations in $(u_i^T y)$. This is important to know when J is nearly rank deficient ($\sigma_n \ll \sigma_1$). The terms with small σ_i could be omitted.

When J is rank deficient (e.g. $\sigma_n = 0$), then

any vector of the form $x^* = \sum_{\sigma_i \neq 0} \frac{u_i^T y}{\sigma_i} v_i + \sum_{\sigma_i = 0} \tau_i v_i$

for arbitrary τ_i is a minimizer. The solution with minimum norm can be found by setting $\tau_i = 0$.

Nondlinear LS Problems

The Gauss-Newton Method: This is a modified Newton method with line search. Instead of solving $\nabla^2 f_k p = -\nabla f_k$,

we solve $J_k^T J_k p_k^{GN} = -J_k^T r_k$.

Here we used $\nabla^2 f_k \approx J_k^T J_k$, so we avoid computing $\nabla^2 r_j$, $j=1, 2, \dots, m$. Only first order derivatives are needed. This approximation of the Hessian is accurate when the second order term is relatively negligible (i.e. when $|r_j(x)| \cdot \|\nabla^2 r_j(x)\|$ is small). This happens near x^* if $r_j(x)$ are small. Consequently, the GN method has fast convergence.

When J_k has full rank, then $\nabla f_k = +J_k^T r_k$ is nonzero and p_k^{GN} is a descent direction for f .

$$(p_k^{GN})^T \nabla f_k = (p_k^{GN})^T J_k r_k = - (p_k^{GN})^T J_k^T J_k p_k^{GN} = - \|J_k p_k^{GN}\|^2 \leq 0$$

Here the final equality is strict unless $J_k p_k^{GN} = 0$. This latter condition only occurs when $J_k r_k = \nabla f_k = 0$ if J_k is full rank.

Clearly, p_k^{GN} is the solution to $\min_p \frac{1}{2} \|J_k p + r_k\|^2$. The QR and SVD methods discussed above could be used to calculate p_k^{GN} . If m is large, then $J^T J$ could be recursively computed: $J^T J = \sum_{i=1}^m \nabla r_i \nabla r_i^T$. Also $J^T r = \sum_{i=1}^m r_i \nabla r_i$.

Convergence of the Gauss-Newton Method

Thm 10.1 Suppose each r_j is Lipschitz continuously differentiable in a neighborhood N of the bounded level set $\mathcal{L} = \{x \mid f(x) \leq f(\bar{x})\}$, and that the Jacobians $J(x)$ satisfy the uniform full-rank condition (i.e. $\|J(x)\|_2 \geq \delta \|x\|$ for some $\delta > 0 \forall x \in N$).^{*}

Then if the iterates x_k are generated by the GN method with step lengths α_k that satisfy the Wolfe conditions, we have $\lim_{k \rightarrow \infty} J_k^T r_k = 0$.

* This means, the singular values of $J(x)$ do not get close to zero (they are uniformly bounded away from zero).

Proof: Note that we can select N of the bounded set \mathcal{L} small enough such that for some $L > 0$ and $\beta > 0$ we have $|r_j(x)| \leq \beta$ and $\|\nabla r_j(x)\| \leq \beta$ $\forall x \in N$ $\forall j = 1, \dots, m$.
 $|r_j(x) - r_j(\bar{x})| \leq L\|x - \bar{x}\|$ and $\|\nabla r_j(x) - \nabla r_j(\bar{x})\| \leq L\|x - \bar{x}\|$ due to the assumption that r_j is Lipschitz continuously differentiable here.

Then $\exists \bar{\beta} > 0 \Rightarrow \|\nabla J(x)^T\| = \|\nabla J(x)\| \leq \bar{\beta} \quad \forall x \in \mathbb{Z}$.

(This is a consequence of Taylor's theorem as we have seen in the previous chapters.)

Recall from the Appendix that products and sums of Lipschitz continuous functions are also Lipschitz continuous. Then $\nabla f(x) = \sum_{j=1}^m r_j(x) \nabla r_j(x)$ is Lipschitz continuous. This means, the conditions of Thm 3.2 (Zoutendijk's Thm) are satisfied.

We checked before that $-\nabla f_k^T P_k^{GN} \geq 0$; we

$$\begin{aligned} \text{have } \cos \theta_k &= \frac{-\nabla f_k^T P_k^{GN}}{\|P_k^{GN}\| \cdot \|\nabla f\|} = \frac{\|\nabla J_k P_k^{GN}\|^2}{\|P_k^{GN}\| \cdot \|\nabla J_k^T J_k P_k^{GN}\|} \\ &\geq \frac{\gamma^2 \|P_k^{GN}\|^2}{\bar{\beta}^2 \|P_k^{GN}\|^2} = \frac{\gamma^2}{\bar{\beta}^2} > 0 \end{aligned}$$

Consequently, from Thm 3.2, we get $\nabla f(x_k) \rightarrow 0$. \square

Now let's investigate the convergence speed. For a unit-step-length GN update we have

$$\begin{aligned} x_{k+1} - x^* &= x_k - x^* - (\nabla J_k^T \nabla J_k)^{-1} \nabla f_k \\ &= (\nabla J_k^T \nabla J_k)^{-1} \{ \nabla J_k^T \nabla J_k (x_k - x^*) + \nabla f(x^*) - \nabla f_k \} \end{aligned}$$

this is zero.

Taylor says:

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla J(x^* + t(x_k - x^*)) (x_k - x^*) dt$$

$$\left(\text{where } H(x) \triangleq \sum_{j=1}^m r_j(x) \nabla^2 r_j(x) \right) \Rightarrow + \int_0^1 H(x^* + t(x_k - x^*)) (x_k - x^*) dt$$

Assuming $H(\cdot)$ is Lipschitz cont. near x^* :

$$\|x_k + p_k^{GN} - x^*\| \leq \int_0^1 \|[\mathbf{J}^T \mathbf{J}(x_k)]^{-1} H(x^* + t(x_k - x^*))\| \cdot \|x_k - x^*\| dt \\ + \delta(\|x_k - x^*\|^2) \\ \approx \|[\mathbf{J}^T \mathbf{J}(x^*)]^{-1} H(x^*)\| \cdot \|x_k - x^*\| + \delta(\|x_k - x^*\|^2)$$

\therefore If $\|[\mathbf{J}^T \mathbf{J}(x^*)]^{-1} H(x^*)\| \ll 1$, we expect superlinear convergence from p^{GN} . If $H(x^*) = 0$ (e.g. in the linear case), then the convergence rate is quadratic.

Levenberg-Marcquardt Method

We use the same Hessian approximation as the GN method but use a trust-region approach instead of line search. This way, the case where $\mathbf{J}(x)$ is rank-deficient or nearly so is not a problem anymore.

For a spherical trust region, the trust-region subproblem is given as (for $\Delta_k > 0$)

$$\min_p \frac{1}{2} \|J_k p + r_k\|^2 \quad \text{s.t. } \|p\| \leq \Delta_k$$

$$\text{So our model is } M_k(p) = \frac{1}{2} \|r_k\|^2 + p^T J_k^T r_k + \frac{1}{2} p^T J_k^T J_k p$$

Case 1: When $\|p^{GN}\| < \Delta$, then p^{GN} solves the subproblem.

Case 2: Otherwise, $\exists \lambda > 0$ s.t. the solution $p = p^{LM}$ satisfies

$$\|p\| = \Delta \text{ and } (J^T J + \lambda I)p = -J^T r.$$

Formally, we have the following result.

Lemma 10.2: The vector p^{LM} is a solution of the trust-region subproblem $\min_P \frac{1}{2} \|Jp + r\|^2$ s.t. $\|p\| \leq \Delta$

If p^{LM} is feasible and $\lambda > 0 \Rightarrow$

$$(J^T J + \lambda I) p^{LM} = -J^T r$$

$$\lambda (\Delta - \|p^{LM}\|) = 0$$

Proof: The semidefiniteness condition in Thm 4.1 is satisfied automatically since $J^T J \geq 0$ and $\lambda > 0$.

The two conditions in Lemma 10.2 then follow directly from Thm 4.1. \square

From the first condition in the lemma, we see that the solution is given by the normal eqns

for $\min_P \frac{1}{2} \left\| \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix} P + \begin{bmatrix} r \\ 0 \end{bmatrix} \right\|^2$ so we can develop an algorithm that does not require the computation of $J^T J$ and its Cholesky factorization.

Implementation of the LM Method

We can use the root-finding algorithm of chapter 4 to find the λ that matches the given Δ . Since the approximate Hessian $B = J^T J$ is positive semidefinite, whenever the current estimate $\lambda^{(l)}$ is positive, the Cholesky factor R is guaranteed to exist:

$$B + \lambda^{(l)} I = R^T R \quad (\text{see Alg. 4.3})$$

Due to the special form of B , Cholesky factorization need not be computed from scratch at every iteration.

If we find the QR decomposition of $\begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix}$:

$$\begin{bmatrix} R_\lambda \\ 0 \end{bmatrix} = Q_\lambda^T \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix} \quad \begin{array}{l} Q_\lambda \text{ orthogonal} \\ R_\lambda \text{ upper triangular} \end{array}$$

then $(J^T J + \lambda I) = R_\lambda^T R_\lambda$. To see this multiply both sides with their own transposes:

$$\begin{bmatrix} R_\lambda^T & 0 \end{bmatrix} \begin{bmatrix} R_\lambda \\ 0 \end{bmatrix} = \underbrace{\begin{bmatrix} J & \sqrt{\lambda} I \end{bmatrix} Q_\lambda}_{I} Q_\lambda^T \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix} \Leftrightarrow R_\lambda^T R_\lambda = (J^T J + \lambda I)$$

Let $J = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$ be the QR-decomposition of J .

$$\text{Then } \begin{bmatrix} R \\ 0 \end{bmatrix} = \begin{bmatrix} Q^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} J \\ \sqrt{\lambda} I \end{bmatrix}$$

The matrix $\begin{bmatrix} R \\ 0 \\ \sqrt{\lambda} I \end{bmatrix}$ is upper triangular

except for the n nonzero terms of $\sqrt{\lambda} I$. There is a rotation matrix \bar{Q}_λ^T such that

$$\bar{Q}_\lambda^T \begin{bmatrix} R \\ 0 \\ \sqrt{\lambda} I \end{bmatrix} = \begin{bmatrix} R_\lambda \\ 0 \\ 0 \end{bmatrix} \Rightarrow Q_\lambda = \begin{bmatrix} Q & 0 \\ 0 & I \end{bmatrix} \bar{Q}_\lambda.$$

(see the book for the iterative procedure to find \bar{Q}_λ ...)

- * In the root-finding algorithm for λ , for each new iteration only \bar{Q}_λ needs to be recalculated.

This saves a lot of computations in the case of $m \gg n$.

After an initial $O(mn^2)$ cost to find Q , we need $O(n^3)$ operations to update \bar{Q}_λ and R_λ .

- * LS problems are usually poorly scaled. To alleviate this problem, we can employ elliptical trust regions.

$$\min_p \left\| J_k p + r_k \right\|^2 \text{ s.t. } \|D_k p\| \leq \Delta_k$$

Here D_k is diagonal with positive entries. The solution

satisfies $(J_k^T J_k + \lambda D_k^2) p_k^{LM} = -J_k^T r_k$ which

solves $\min_p \left\| \begin{bmatrix} J_k \\ \sqrt{\lambda} D_k \end{bmatrix} p + \begin{bmatrix} r_k \\ 0 \end{bmatrix} \right\|^2$.

- * If D_k^2 is selected as $D_k^2 = \text{diag}(J_k^T J_k)$, then the algorithm is invariant under diagonal scaling of x . This is analogous to scaling by the diagonal elements of the Hessian.
- * If $J(x)$ is sparse, the trust-region subproblem could be solved (approximately up to some tolerance) using Alg. 7-2 (CG-Stehaag) which is a Newton-CG method for trust-region subproblems. In this case, we replace the exact Hessian $\nabla^2 f_k$ with $J_k^T J_k$.

Convergence of the LM Method

Thm 10.3: Let $m \in (0, 1/n)$ in Alg 4.1 (Trust-Region Method) and suppose that the level set $\mathcal{L} = \{x \mid f(x) \leq f(x_0)\}$ is bounded and that $r_j(\cdot)$, $j=1, \dots, m$ are Lipschitz cont. differentiable in a neighborhood N of \mathcal{L} . Assume that for each k , the approximate solution p_k of the $\overset{\text{LM}}{\text{trust-region}}$ subproblem satisfies

$$m_k(0) - m_k(p_k) \geq c_1 \|J_k^T r_k\| \min(\Delta_k, \frac{\|J_k^T r_k\|}{\|J_k^T J_k\|})$$

for some constant $c_1 > 0$; and $\|p_k\| \leq \gamma \Delta_k$ for some $\gamma \geq 1$. Then $\lim_{k \rightarrow \infty} \nabla f_k = \lim_{k \rightarrow \infty} J_k^T r_k = 0$.

Proof: Due to the smoothness of $r_j(\cdot)$, we can choose $\mu > 0 \Rightarrow \|J_k^T J_k\| \leq M \quad \forall k$. The objective f is bounded below by zero. Hence, the assumptions of Thm 4.6 are satisfied. \square

* If we simply ensure that the decrease at each step

meets or exceeds that of the Cauchy step, then we get global convergence. Alg. 7.2 satisfies this automatically for $c_i = 4/2$.

* The local convergence of LM is similar to GM.

Near a solution x^* at which $J_x^T J_x$ dominates $\nabla^2 f_x$, the trust region become inactive and the algorithm takes GN steps towards the minimizer.

* In large residual problems GN and LM converge linearly, so hybrid algorithms that start as Newton or quasi-Newton and behave like GN or LM if residuals are small could be developed.

One method is to maintain a sequence S_k that approximates $\sum_j r_j(x_k) \nabla^2 r_j(x_k)$ and use $B_k = J_k^T J_k + S_k$ as the Hessian approximation.

Ideally, S_{k+1} should be a close approximation to the exact second-order term at x_{k+1} :

$$S_{k+1} \approx \sum_j r_j(x_{k+1}) \nabla^2 r_j(x_{k+1})$$

We approximate $\nabla^2 r_j(x_{k+1}) \approx B_{j,k+1}$ and impose

$$B_{j,k+1}(x_{k+1} - x_k) = \nabla r_j(x_{k+1}) - \nabla r_j(x_k) = J_j(x_{k+1})^T - J_j(x_k)^T$$

$\uparrow j$ th row

This yields a secant equation:

$$\begin{aligned} S_{k+1}(x_{k+1} - x_k) &= \sum_j r_j(x_{k+1}) B_{j,k+1}(x_{k+1} - x_k) \\ &\dots = J_{k+1}^T r_{k+1} - J_k^T r_k \end{aligned}$$

We also require that S_{k+1} is symmetric and $S_{k+1} - S_k$ is small.

$$S_{k+1} = S_k + \frac{(z - S_k s) y^T + y(z - S_k s)^T}{y^T s} - \frac{(z - S_k s)^T s}{(y^T s)^2} yy^T$$

where $s \triangleq x_{k+1} - x_k$; $y \triangleq J_{k+1}^T r_{k+1} - J_k^T r_k$; $z \triangleq J_{k+1}^T r_{k+1} - J_k^T r_{k+1}$.

At each iteration, we scale it to $\tau_k S_k$ where

$$\tau_k = \min\left(1, \frac{|S^T z|}{|S^T S_k s|}\right)$$

to ensure that S_k vanishes at a zero-

residual solution. Finally, if the resulting GN model

produces a satisfactory step, we set $\tau_k = 0$ and use the GN update.

Orthogonal Distance Regression

So far we assumed that the ordinate t is measured very accurately compared to the other variable y . In some applications this is not the case (e.g. system identification from noisy input-output data). Errors-in-variables and total least squares (in the linear case) are well known fields that address this issue.

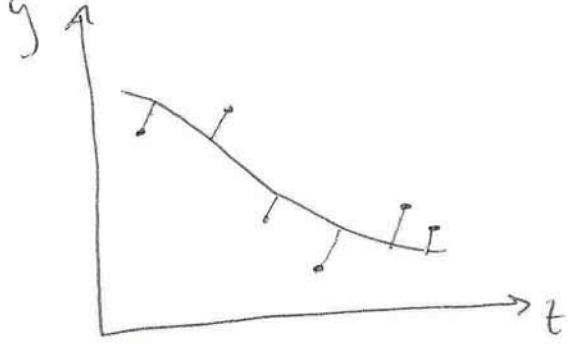
One approach (that assumes error-squares in all variables can be somehow weighted suitably) is to use orthogonal distance regression (also called nonlinear principal component analysis in neural network literature, and is related to principal surfaces and manifold learning). We assume

$$y_j = \phi(x; t_j + \delta_j) + \epsilon_j \quad j=1, \dots, m$$

and define the problem as

$$\min_{x, \delta_j, t_j} \frac{1}{2} \sum_{j=1}^m w_j^2 t_j^2 + d_j^2 \delta_j^2 = \frac{1}{2} \sum_{j=1}^m (\epsilon_j^2 + \delta_j^2) \begin{pmatrix} w_j^2 & 0 \\ 0 & d_j^2 \end{pmatrix} \begin{pmatrix} t_j \\ \delta_j \end{pmatrix}$$

where w_j and d_j are weights that determine the relative significance of the two error components.



at 215

On the curve, the points with shortest distance to a sample will be orthogonal to tangents the direction to the sample.

In general, each sample may have one or more equally distant, shortest distance, orthogonal projections.

Substituting our model for y_j into the optimization problem to eliminate the ϵ_j variables:

$$\begin{aligned} \min_{x, \delta} F(x, \delta) &= \frac{1}{2} \sum_{j=1}^m w_j^2 [y_j - \phi(x; t_j + \delta_j)]^2 + d_j^2 \delta_j^2 \\ &\stackrel{\Delta}{=} \frac{1}{2} \sum_{j=1}^{2m} r_j^2(x, \delta) \end{aligned}$$

where $\delta = [\delta_1, \dots, \delta_m]^T$ and

$$r_j(x, \delta) = \begin{cases} w_j [\phi(x; t_j + \delta_j) - y_j], & j=1, \dots, m \\ d_{j-m} \delta_{j-m}, & j=m+1, \dots, 2m \end{cases}$$

Now we have a standard least squares problem with $(m+n)$ unknowns and $2m$ residuals. The Jacobian of this problem has a special structure that can be exploited in implementation of GN and LM methods.

$$\frac{\partial r_j}{\partial \delta_i} = \frac{\partial \{ \phi(x_j; t_j + \delta_j) - y_j \}}{\partial \delta_i} = 0 \quad i, j = 1, 2, \dots, m \\ i \neq j$$

$$\frac{\partial r_j}{\partial x_i} = 0 \quad j = m+1, \dots, 2m, \quad i = 1, \dots, n$$

$$\frac{\partial r_{m+j}}{\partial \delta_i} = \begin{cases} d_i & \text{if } i=j \\ 0 & \text{o.w.} \end{cases}$$

$\therefore J(x, \delta) = \begin{bmatrix} \hat{J} & V \\ 0 & D \end{bmatrix}$ where \hat{J} is the $m \times n$ derivatives
 $\nabla w_j \phi(x_j; t_j + \delta_j)$ w.r.t. x for $j=1, \dots, m$; V and
 D are diagonal matrices. If we partition the step
vector and the residual vector accordingly,

$$p = \begin{bmatrix} p_x \\ p_s \end{bmatrix} \quad r = \begin{bmatrix} \hat{r}_1 \\ \hat{r}_2 \end{bmatrix}$$

the normal equations become

$$\begin{bmatrix} (\hat{J}^T \hat{J} + \lambda I) & \hat{J}^T V \\ V^T \hat{J} & (V^2 + D^2 + \lambda I) \end{bmatrix} \begin{bmatrix} p_x \\ p_s \end{bmatrix} = - \begin{bmatrix} \hat{J}^T \hat{r}_1 \\ (V \hat{r}_1 + D \hat{r}_2) \end{bmatrix}$$

Since $(V^2 + D^2 + \lambda I)$ is diagonal, p_s can be easily eliminated
and the remaining problem of size $n \times n$ can be solved
for p_x alone, resulting in computational efficiency.

10.1) $J \in \mathbb{R}^{m \times n}$ with $m \geq n$. Show J is full column rank iff $J^T J$ is positive definite (which implies nonsingular).

(\Leftarrow) Consider $y \in \mathbb{R}^n, y \neq 0$. Assume $J^T J > 0$. Then $x^T J^T J x > 0 \forall x$.
 $\Rightarrow y = Jx$ has $\|y\| > 0$ for all x . Suppose that J does not have full column rank. Then $\exists \alpha \neq 0 \text{ s.t. } J\alpha = 0$, hence $\alpha^T J^T J \alpha = 0$, which contradicts with the condition above.
So J must have full column rank.

(\Rightarrow) Let J have full column rank, so $\nexists \alpha \neq 0 \text{ s.t. } J\alpha = 0$.

Assume $J^T J$ is not positive definite. Then (1) $\exists \alpha \neq 0 \text{ s.t. } \alpha^T J^T J \alpha = 0$
or (2) $\exists \alpha \neq 0 \text{ s.t. } \alpha^T J^T J \alpha < 0$

(1) If $\alpha^T J^T J \alpha = 0$, then $\|J\alpha\| = 0$

then $J\alpha = 0$, but this contradicts our initial assumption.

so $\alpha^T J^T J \alpha = 0$ can not occur for $\alpha \neq 0$.
(i.e. $J^T J$ nonsingular)

(2) $\alpha^T J^T J \alpha = \|J\alpha\|^2 \leq 0$ cannot occur for any α .

\therefore Due to these contradictions, we must have $J^T J > 0$. \square

10.2) Show that $f(x) = \frac{1}{2} \|Jx - y\|^2$ is convex.

Clearly, $f(x) = \frac{1}{2} (Jx - y)^T (Jx - y) = \frac{1}{2} x^T J^T J x - y^T J^T x + \frac{1}{2} y^T y$
is a quadratic expression in the form $f(x) = x^T A x + b^T x + c$
where $A = \frac{1}{2} J^T J$, $b = -J^T y$, and $c = \frac{1}{2} y^T y$. Above, we showed
that $J^T J$ can not have a negative eigenvalue ($\Rightarrow (2)$). In general
we have $J^T J \geq 0$ (also easy to prove directly). A quadratic
with a positive semidefinite A ($A \geq 0$) is convex; if $A > 0$, then
the quadratic is strictly convex. Details left to you as practice.

10.3) a) Show that if \mathbf{Q} is orthogonal $\|\mathbf{x}_\perp\|_2^2 = \|\mathbf{Q}\mathbf{x}\|_2^2$.

$$\|\mathbf{Q}\mathbf{x}\|_2^2 = \mathbf{x}^\top \mathbf{Q}^\top \mathbf{Q} \mathbf{x} = \mathbf{x}^\top \mathbf{x} = \|\mathbf{x}\|_2^2$$

b) Given $\mathbf{J}^\top \mathbf{J} = \bar{\mathbf{R}}^\top \bar{\mathbf{R}}$ and $\mathbf{J}\mathbf{U} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ 0 \end{pmatrix} = \{\mathbf{Q}_1, \mathbf{Q}_2\} \begin{pmatrix} \mathbf{R} \\ 0 \end{pmatrix} = \mathbf{Q}, \mathbf{R}$,

Also we are given that \mathbf{J} is full column rank ($\mathbf{J}^\top \mathbf{J} > 0$).

Show that if $\mathbf{U} = \mathbf{I}$, then $\bar{\mathbf{R}} = \mathbf{R}$.

$$\mathbf{U} = \mathbf{I} \Rightarrow \mathbf{J} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ 0 \end{pmatrix} = \mathbf{Q}, \mathbf{R} \Rightarrow \mathbf{J}^\top \mathbf{J} = [\mathbf{I}^\top \mathbf{R}^\top \mathbf{Q}^\top] \begin{pmatrix} \mathbf{R} \\ 0 \end{pmatrix}$$

Since both \mathbf{R} and $\bar{\mathbf{R}}$ are ~~full rank~~ $= \begin{pmatrix} \mathbf{R}^\top \mathbf{R} & 0 \\ 0 & 0 \end{pmatrix} = \bar{\mathbf{R}}^\top \bar{\mathbf{R}}$,

due to $\mathbf{J}^\top \mathbf{J} > 0$, we cannot have a row/column of zeros in $\bar{\mathbf{R}}^\top \bar{\mathbf{R}}$, so the size of $\mathbf{R}^\top \mathbf{R}$ must match that of $\bar{\mathbf{R}}^\top \bar{\mathbf{R}}$.

$\therefore \mathbf{R}^\top \mathbf{R} = \bar{\mathbf{R}}^\top \bar{\mathbf{R}} > 0$. Both $\bar{\mathbf{R}}$ and \mathbf{R} are upper triangular nonsingular, so $\mathbf{R} = \bar{\mathbf{R}}$.

10-h) a) Show that $\mathbf{x}^* = \sum_{\sigma_i \neq 0} \frac{u_i^\top \mathbf{y}}{\sigma_i} \mathbf{v}_i + \sum_{\sigma_i = 0} \tau_i \mathbf{v}_i$ is a minimizer of $f(\mathbf{x}) = \frac{1}{2} \|\mathbf{J}\mathbf{x} - \mathbf{y}\|_2^2$.

b) Show that $\|\mathbf{x}^*\|$ is minimized when $\tau_i = 0 \ \forall i$ with $\sigma_i = 0$.

a) $\nabla f(\mathbf{x}) = 0 \Leftrightarrow \mathbf{J}^\top \mathbf{J} \mathbf{x}^* = \mathbf{J}^\top \mathbf{y}$. Substitute the candidate \mathbf{x}^* here. Note $\mathbf{J}^\top \mathbf{J} = \mathbf{V} \mathbf{S}^\top \mathbf{V}$

$$\mathbf{J}^\top \mathbf{J} \mathbf{x}^* = \mathbf{J}^\top \mathbf{J} \underbrace{\sum_i \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} + \tau_i \right) \mathbf{v}_i}_{\mathbf{v}_i^\top \mathbf{v}_i} = \sum_i \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} + \tau_i \right) \mathbf{J}^\top \mathbf{J} \mathbf{v}_i = \sum_i \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} + \tau_i \right) \sigma_i^2 \mathbf{v}_i$$

Here
 $\sum_i \frac{u_i^\top \mathbf{y}}{\sigma_i} = \sum_{\sigma_i \neq 0} \sigma_i \mathbf{v}_i^\top \mathbf{y} + \sum_{\sigma_i = 0} \tau_i \sigma_i^2 \mathbf{v}_i = \sum_{\sigma_i \neq 0} \sigma_i \mathbf{v}_i^\top \mathbf{y} = \mathbf{J}^\top \mathbf{y}$ ✓

$$\text{b) } \|\mathbf{x}^*\|_2^2 = \sum_i \sum_j \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} + \tau_i \right) \left(\frac{u_j^\top \mathbf{y}}{\sigma_j} + \tau_j \right) \mathbf{v}_j^\top \mathbf{v}_i = \sum_i \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} + \tau_i \right)^2 \|\mathbf{v}_i\|_2^2$$

$\therefore \tau_i = 0 \text{ if } \sigma_i \neq 0$	$\sum_i \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} \right)^2 \ \mathbf{v}_i\ _2^2 + \sum_{\sigma_i = 0} \tau_i^2 \ \mathbf{v}_i\ _2^2$
$\tau_i = 0 \text{ if } \sigma_i \neq 0$	$= \sum_{\sigma_i = 0} \left(\frac{u_i^\top \mathbf{y}}{\sigma_i} \right)^2 \ \mathbf{v}_i\ _2^2 + \sum_{\sigma_i = 0} \tau_i^2 \ \mathbf{v}_i\ _2^2$
$\{\text{free o.w.}\}$	$\text{Clearly, this is minimized if all } \tau_i = 0 \text{ for } \sigma_i = 0.$

(Introducing this so I can start with a single summation.)

Chapter 11: Nonlinear Equations

In many applications we might need to find the values of variables that satisfy a number of given relationships / equations. If we have n unknown variables and n equations, such as $r(x) = 0$, then we have a system of nonlinear equations.

$$r: \mathbb{R}^n \rightarrow \mathbb{R}^n \quad r(x) = \begin{bmatrix} r_1(x) \\ \vdots \\ r_n(x) \end{bmatrix} \quad \text{Assume } r_i(x) \text{ is smooth.}$$

The solution is a vector $x^* \in \mathbb{R}^n$ (also called a root).

Example $r(x) = \begin{bmatrix} x_1^2 - 1 \\ \sin x_1 - x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

This has infinitely many solutions. Two are

$$x^* = \begin{bmatrix} 3\pi/2 \\ -1 \end{bmatrix}, \quad x^* = \begin{bmatrix} \pi/2 \\ 1 \end{bmatrix}.$$

In some cases, there may be no solution, one or more solutions, including infinitely many.

Newton's Method for Nonlinear Equations

Thm 11.1 Suppose that $r: \mathbb{R}^n \rightarrow \mathbb{R}^n$ is cont. diff. in some convex open set D and that x and $x+\rho$ are vectors in D . We then have that

(220)

$$r(x+p) = r(x) + \int_0^1 J(x+tp)p dt.$$

We can define a linear model $M_p(p)$ of $r(x_k+p)$ by approximating the second term by $J(x)p$:

$$M_k(p) \triangleq r(x_k) + J(x_k)p$$

(hence the phrase Taylor approximation.)

Newton's method chooses a step p_k for which

$$M_k(p_k) = 0; \text{ i.e. } p_k = -J(x_k)^T r(x_k).$$

Alg 11.1 Newton's Method for Nonlinear Equations

Choose x_0 ;

for $k=0, 1, 2, \dots$

Calculate a solution p_k to the Newton eqns

$$J(x_k)p_k = -r(x_k);$$

$$x_{k+1} \leftarrow x_k + p_k;$$

end (for)

* Compare this to $\min_x r^T(x) r(x)$. Let $f(x) \triangleq r^T(x) r(x)$. Then

$$\nabla f(x) = 2 J^T(x) r(x) \quad \text{and} \quad \nabla^2 f(x) = 2 \sum_i r_i^2(x) I + 2 J^T(x) J.$$

This introduces many possible local minima that are not zeros of $r(x)$.

In general for some $f(x)$, we have $\min_x f(x)$ occur at a point where $\nabla f(x) = 0$. Let $r(x) = f(x)$ and $J(x) = \nabla^2 f(x)$. Then the Newton methods that solves $\min_x f(x)$ and $\nabla f(x) = 0$ are identical.

Defn. If $J(x^*)$ is singular, then x^* is called a degenerate root.

Ex $r(x) = x^2$ has a single root at $x^* = 0$.

$$J(x) = 2x. \text{ Then } P_k = -\frac{1}{2x_k} x_k^2 = -\frac{x_k}{2}.$$

$$x_{k+1} = x_k - \frac{x_k}{2} = \frac{1}{2}x_k \quad \therefore \quad r_k = \frac{1}{2^k} x_0.$$

$x_{k+1} = \frac{1}{2^{k+1}} x_0 = \frac{1}{2} x_k \Rightarrow |x_{k+1}| = \frac{1}{2} |x_k|$ so the convergence rate is linear.

Other possible problems:

- * When x_0 is far from an x^* , the algorithm may behave erratically. If $|J(x_k)| = 0$, the Newton step is undefined.

- * $J(x)$ may be difficult to obtain and/or expensive to compute so for large n P_k is expensive.

Thm 11.2 Suppose that r is cont. diff. in a convex open set $D \subset \mathbb{R}^n$. Let $x^* \in D$ be a nondegenerate solution of $r(x) = 0$ and let $\{x_k\}$ be the sequence of iterates generated by Alg. 11.1. Then when $x_k \in D$ is sufficiently close to x^* , we have

$$x_{k+1} - x^* = \mathcal{O}(||x_k - x^*||),$$

indicating Q-superlinear convergence. When r is Lipschitz continuously differentiable near x^* , we have

If x_k sufficiently close to x^* that

$$x_{k+1} - x^* = \mathcal{O}(||x_k - x^*||^2)$$

indicating Q-quadratic convergence.

Proof. See pg 276-277 in the book.

Inexact Newton Methods

Instead of solving $J(x_k)^T p_k = -r(x_k)$, we seek an approximate solution that satisfies $||r_k + J_{k,0} p_k|| \leq \gamma_k ||r_k||$ where $\gamma \in \{0, 1\}$ is a constant. for some $\gamma_k \in [0, 1]$.

Defn $\{\gamma_k\}$ is called the forcing sequence.

Framework 11.2 Inexact Newton for Nonlinear Eqs

Given $\eta \in \{0, 1\}$; choose x_0 ;

for $k=0, 1, 2, \dots$

choose forcing parameter $\gamma_k \in \{0, \eta\}$;

Find P_k that satisfies $\|r_{k+1} + J_k P_k\| \leq \gamma_k \|r_k\|$.

$$x_{k+1} \leftarrow x_k + P_k;$$

end (for)

* Iterative techniques based on Krylov spaces, such as conjugate gradient, could be used to solve $J_P = -r$ until the inequality is satisfied.

(after multiplying by J^T from left to make it symmetric positive definite).

Thm 11.3 Suppose that r is cont. diff. in a convex open set $D \subset \mathbb{R}^n$. Let $x^* \in D$ be a nondegenerate solution of $r(x) = 0$, and let $\{x_k\}$ be the sequence of iterates generated by the framework 11.2. Then when $x_k \in D$ is sufficiently close to x^* , the

Following are true:

- (i) If γ is sufficiently small, the convergence of $\{x_k\}$ to x^* is α -linear.
- (ii) If $r_k \rightarrow 0$, the convergence is α -super-linear.
- (iii) If in addition $J(\cdot)$ is Lipschitz cont. in a neighbourhood of x^* and $\gamma_k = O(1/r_k)$,
the convergence is α -quadratic. \nearrow
 $r_k \rightarrow 0$ at least
as fast as $1/r_k$.

Proof: See pg 278-279 in the book.

Broyden's Method

Secant methods do not require the calculation of $J(x)$. Let B_k be an approximation of $J(x_k)$. Then the local linear model is $M_k(p) = r(x_k) + B_k p$.

$$\Rightarrow P_k = -B_k^{-1} r(x_k)$$

Let $s_k \stackrel{\Delta}{=} x_{k+1} - x_k$ and $y_k \stackrel{\Delta}{=} r(x_{k+1}) - r(x_k)$ as we did earlier in the quasi-Newton methods.

We have (Thm 11.1): $y_k = \int_0^1 J(x_k + ts_k) s_k dt \approx J(x_{k+1}) s_k + o(1/s_k)$.

We require $y_k = B_{k+1} s_k$ due to this property of the Jacobian $J(x_{k+1})$. \Rightarrow This is the secant equation. This specifies how B_{k+1} should behave along s_k , but nothing about directions orthogonal to s_k . With n^2 unknowns and n eqns, we need more constraints.

$$\min_{B_{k+1}} \|B_k - B_{k+1}\|_2 \text{ yields}$$

$$(\text{Broyden}) \quad B_{k+1} = B_k + \frac{(y_k - B_k s_k) s_k^T}{s_k^T s_k}$$

Lemma 11.4 Among all matrices B satisfying

$B s_k = y_k$, Broyden's B_{k+1} given above minimizes $\|B - B_k\|_2$

Proof: Consider $\|\cdot\| = \|\cdot\|_2$. We have $\left\| \frac{s s^T}{s^T s} \right\| = 1 \quad \forall s \in \mathbb{R}^n$.

$$\begin{aligned} \|B_{k+1} - B_k\| &= \left\| \frac{(y_k - B_k s_k) s_k^T}{s_k^T s_k} \right\| = \left\| \frac{(B - B_k) s_k s_k^T}{s_k^T s_k} \right\| \\ &\leq \|B - B_k\| \cdot \left\| \frac{s_k s_k^T}{s_k^T s_k} \right\| = \|B - B_k\| \end{aligned}$$

$\forall B \Rightarrow B s_k = y_k \quad \square$

Alg. 11.3 Broyden

Choose x_0 and a nonsingular B_0 ;

for $k=0, 1, 2, \dots$

Calculate a solution p_k to $B_k p_k = -r(x_k)$.

Choose α_k by performing a line search along p_k :

$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$

$$s_k \leftarrow x_{k+1} - x_k; \quad y_k \leftarrow r(x_{k+1}) - r(x_k);$$

$$\text{Obtain } B_{k+1} \text{ from } B_{k+1} = B_k + \frac{(y_k - B_k s_k) s_k^T}{s_k^T s_k};$$

End (for).

* Under certain assumptions, Broyden's method converges superlinearly: $\|x_{k+1} - x^*\| = o(\|x_k - x^*\|)$.

Thm 11.5 Suppose that the assumptions of Thm 11.2 hold. Then $\exists \epsilon > 0$ and $\delta > 0 \ni$ if x_0 and B_0 satisfy

$$\|x_0 - x^*\| \leq \delta \text{ and } \|B - J(x^*)\| \leq \epsilon$$

the sequence $\{x_k\}$ generated by Broyden's method is well-defined and converges α -superlinearly to x^* .

Tensor Methods

The model $\hat{M}_k(p)$ is augmented to capture some of the nonlinear behavior of r .

$$\hat{M}_k(p) = r(x_k) + J(x_k)p + \frac{1}{2} T_k p p$$

where T_k is a tensor defined by n^3 elements (it is an order-3 $n \times n \times n$ tensor). Here:

$$(T_k uv)_i = \sum_{j=1}^n \sum_{\ell=1}^n (T_k)_{ij\ell} u_j v_\ell$$

T_k could be built from the second derivatives of r

$$\text{at the point } x_k : (T_k)_{ij\ell} = [\nabla^2 r_i(x_k)]_{j\ell}.$$

Ex $r(x) = \begin{cases} (x_1+3)(x_2^3-7) + 18 \\ \sin(x_2 e^{x_1-1}) \end{cases}$

$$(T(x)uv)_1 = u^\top \nabla^2 r_1(x)v = u^\top \begin{bmatrix} 0 & 3x_2^2 \\ 3x_2^2 & 6x_2(x_1+3) \end{bmatrix} v$$

* Storage of $n^3/2$ entries might be problematic for larger n .

* There may not be a step that gives $\hat{M}_k(p) = 0$.

We can choose T_k such that $\hat{M}_k(p)$ interpolates $r(x_{k+p})$ at some previous iterates.

$$\hat{M}_k(x_{k-j} - x_k) = r(x_{k-j}), \quad j=1, 2, \dots, q.$$

$$\therefore \frac{1}{2} T_k s_{jk} s_{jk}^T = r(x_{k-j}) - r(x_k) - J(x_k) s_{jk}$$

where $s_{jk} \stackrel{\Delta}{=} x_{k-j} - x_k, \quad j=1, 2, \dots, q.$

This yields: $T_k u v = \sum_{j=1}^q \alpha_j (s_{jk}^T u) (s_{jk}^T v)$

where $\alpha_j \in \mathbb{R}^n$ are suitable vectors. Usually $q \leq \sqrt{n}$.

Here T_k can be stored in $2n^2$ locations.

We can use $\min_p M_k(p)$ to get a step.

Practical Methods

Merit Functions: Newton and Broyden methods are not guaranteed to converge to $r(x) = 0$ if they are not initialized close enough to such a solution. A merit function could be defined so that line search and trust region methods can be employed.

Most widely, we see $f(x) = \frac{1}{2} \|r(x)\|_2^2 = \frac{1}{2} \sum_{i=1}^n r_i^2(x).$

Clearly $r(x) = 0 \Leftrightarrow f(x) = 0$.

However, local minimizers of f are not necessarily roots of r (if $J(x)$ is singular at such a point).

$\nabla f(x^*) = J(x^*)^T r(x^*) = 0$ can be obtained if $r(x^*) = 0$ or $J(x^*)$ is singular and $r(x^*)$ is in the corresponding null space.

Other merit functions are possible.

$$\text{e.g. } f_1(x) = \|r(x)\|_1 = \sum_{i=1}^m |r_i(x)|$$

Line Search Methods

Consider $f(x) = \frac{1}{2} \|r(x)\|_2^2$ and its Newton step $J(x_k) P_k = -r(x_k)$. Here P_k is a descent direction for $f(\cdot)$ whenever $r_k \neq 0$ since $P_k^T \nabla f_k = -P_k^T J_k r_k = -\|r_k\|^2 < 0$.

Step lengths for $x_{k+1} = x_k + \alpha_k P_k$ can be chosen as discussed before, satisfying the Wolfe conditions.

Then we have the following result which follows directly from Thm 3.2.

Thm 11.6 Suppose that $J(x)$ is Lipschitz cont. in a neighborhood \mathcal{D} of $\bar{x} = \{x : f(x) \leq f(\bar{x})\}$ and that $\|J(x)\|$ and $\|\nabla(x)\|$ are bounded above on \mathcal{D} . Suppose that a line-search algorithm is applied to f where $P_k^T \nabla f_k < 0$ and α_k satisfy the Wolfe conditions. Then we have that the Zoutendijk condition holds:

$$\sum_{k \geq 0} \cos^2 \theta_k \|J_k^T r_k\|^2 < \infty$$

$$\text{where } \cos \theta_k = \frac{-P_k^T \nabla f_k}{\|P_k\| \cdot \|\nabla f_k\|}.$$

Proof Sketch: Verify that Jf is Lipschitz cont on \mathcal{D} and f is bounded below by s on \mathcal{D} . Then apply Thm 3.2.

Fact: Provided that $\cos \theta_k \geq s$ for some $s \in (0, 1)$ and λ_k sufficiently large, Thm 11.6 guarantees that $J_k^T r_k \rightarrow 0$, meaning we converge to a stationary point of f .

If we know that $\|J(x_k)\|^{-1}$ is bounded, then we must have $r_k \rightarrow 0$ (since J_k cannot be singular).

Fact: It can be shown that (pg 288 in the book) for exact and inexact Newton methods, if $R(J_k) < \infty$

$$\text{then } \cos \theta_k = -\frac{\mathbf{p}_k^T J_k}{\|\mathbf{p}_k\| \cdot \|J_k\|} \geq \frac{1-\eta^2}{2\|J_k\| \cdot \|J_k^{-1}\| (1+\eta)} \geq \frac{1-\eta}{2R(J_k)}$$

(substitute $\eta = -1$ for exact, $\eta \in (0, 1)$ for inexact).

* When $R(J_k)$ is large, the Newton direction may yield poor performance. It may even cause the algorithm to fail.

Specifically, the algorithm may converge to a point where the Jacobian is singular. To prevent this, consider $P_k = -(J_k^T J_k + \lambda_k I)^{-1} J_k^T r_k$.

for any $\lambda_k > 0$, the matrix can be inverted, and if λ_k is bounded away from zero, $\cos \theta_k \geq s$ can be satisfied for some $s > 0$.

This is analogous to the Levenberg-Margardt alg.

Selecting λ_k too large can prevent fast Newton convergence. Selecting λ_k too small can cause the algorithm to behave inefficiently near singular J .

Alg. 11.4 Line-Search Newton-like Method

Given c_1, c_2 with $0 < c_1 < c_2 < \frac{1}{2}$; choose x_0 ;
for $k=0, 1, 2, \dots$

Calculate a Newton-like step from $J_k P_k = -r_k$
or from $(J_k^T J_k + \lambda_k I) P_k = J_k^T r_k$ if J_k is
near singular (or use an exact method).

if $\alpha=1$ satisfies the Wolfe conditions

set $\alpha_k = 1$

else

Find $\alpha_k > 0$ that satisfies Wolfe conditions.
end (if)

$x_{k+1} \leftarrow x_k + \alpha_k P_k$

end (for).

Trust-Region Methods

We can apply the standard trust-region method to the merit function $f(x) = \frac{1}{2} \|r(x)\|_2^2$ using $B_k = J_k^T J_k$ and ($B_k \geq 0$ in this case)

$$m_k(p) = \frac{1}{2} \|r_k + J_k p\|_2^2$$

giving the step $p_k = \underset{p}{\operatorname{arg\min}} m_k(p)$ s.t. $\|p\| \leq \Delta_k$.

Here, we have $p_k = \frac{\|r(x_k)\|^2 - \|r(x_k + p_k)\|^2}{\|r(x_k)\|^2 - \|r(x_k) + J(x_k)p_k\|^2}$

* Alg. 11.5 Trust-Region Method for Nonlinear Eqs.
(see pg 281 in the book).

The dogleg method is a special case approximate solution. The Cauchy point is

$$p_k^C = -c_k \left(\Delta_k / \|J_k^T r_k\| \right) J_k^T r_k$$

where $c_k = \min \left\{ 1, \|J_k^T r_k\|^3 / (\Delta_k r_k^T J_k (J_k^T J_k) J_k^T r_k) \right\}$;

The Newton step is $p_k^N = -(J_k^T J_k)^{-1} J_k^T r_k = -J_k^{-1} r_k$.

Then the dogleg method becomes ...

Procedure 11.6 (Dogleg)

Calculate p_k^c ;

if $\|p_k^c\| = \Delta_k$

$p_k \leftarrow p_k^c$;

else

Calculate p_k^J ;

$p_k \leftarrow p_k^c + \tau (p_k^J - p_k^c)$ where τ is largest
 $\tau \in \{0, 1\} \quad \text{if } \|p_k\| \leq \Delta_k;$

end (if).

Thm 11.7 Suppose that $J(x)$ is Lipschitz cont. and that
 $\|J(x)\|$ is bounded above in a neighborhood D of the
level set $\mathcal{L} = \{x : f(x) \leq f(x_0)\}$. Suppose that all
approximate solutions of the trust-region subproblem
satisfy

$$m_k(0) - m_k(p_k) \geq c, \quad \|(J_k^T r_k)\| \min \left(\Delta_k, \frac{\|(J_k^T r_k)\|}{\|(J_k^T)\|} \right)$$

and $\|p_k\| \leq \delta \Delta_k$ for some $\delta < 1$. Then if $\eta = 0$
in Alg. 11.5, we have $\liminf_{k \rightarrow \infty} \|J_k^T r_k\| = 0$ while
if $\eta \in (0, 1)$ we have $\lim_{k \rightarrow \infty} \|J_k^T r_k\| = 0$.

Thm 11.8 Suppose that $\{x_k\}$ generated by Alg 11.5 converges to a nondegenerate solution x^* of $r(x)=0$.

Suppose that $J(x)$ is Lipschitz cont. in an open neighbourhood D of x^* and that the trust-region subproblem is solved exactly & sufficiently large k .

Then $\{x_k\} \rightarrow x^*$ quadratically.

Proof See pg 284-286 in the book.

Continuation/Homotopy Methods

The points where $J(x)$ become singular could create local minima for f that are not solutions of $r(x)=0$. Define the homotopy map :

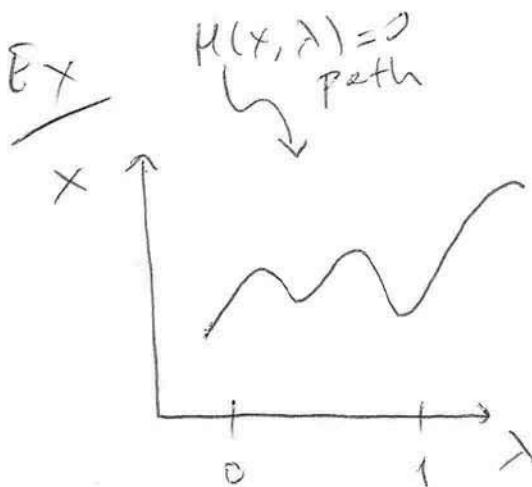
$$H(x, \lambda) = \lambda r(x) + (1-\lambda)(x-a)$$

where $\lambda \in \mathbb{R}$ and $a \in \mathbb{R}^n$. When $\lambda=0$, $H(x, 0) = (x-a)$.

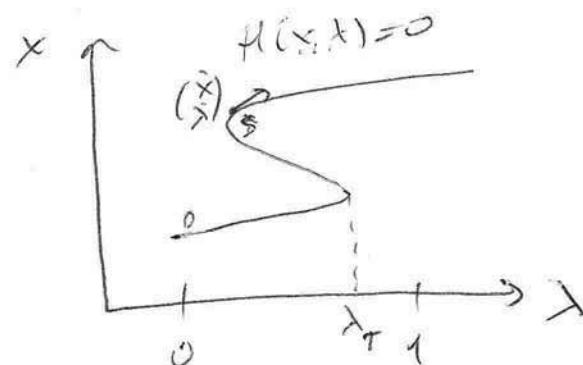
The solution is clearly $x=a$ (for $H(x, 0) = 0$).

When $\lambda=1$, we get $H(x, 1) = r(x) = 0$.

Set $\lambda=0$ and $x=a$. Then increment $\lambda \rightarrow 1$ while continuously solving $H(x, \lambda) = 0$ for x .



Proposition would work.



Proposition would not work

Here λ_+ is called a turning point.

Let s be a parameter (arc length) that we can use as a coordinate along the path $H(x, \lambda) = 0$.

Let $(x(s), \lambda(s)) = (\alpha, 0)$ for $s=0$ and $(x(s), \lambda(s))$ for ~~a~~^{any} point that is s units along the path away from $(\alpha, 0)$. We have $H(x(s), \lambda(s)) = 0 \forall s \geq 0$.

$$\frac{\partial H(x, \lambda)}{\partial x} \dot{x} + \frac{\partial H(x, \lambda)}{\partial \lambda} \dot{\lambda} = 0 \text{ where } (\dot{x}, \dot{\lambda}) = \left(\frac{dx}{ds}, \frac{d\lambda}{ds} \right)$$

Here $(\dot{x}(s), \dot{\lambda}(s))$ is the tangent vector to the path.

$$\text{We have } \left[\frac{\partial H(x, \lambda)}{\partial x}, \frac{\partial H(x, \lambda)}{\partial \lambda} \right] \begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = 0$$

If this matrix is full rank (it is $n \times (n+1)$) its null space is 1-dimensional.

We impose the condition $\|\dot{x}(s)\|^2 + |\dot{\lambda}(s)|^2 = 1$ to normalize the tangent vector to unit length.

The sign of the vector (direction to or away from $s=0$) can be selected (heuristically) such that the current tangent makes an angle $2\pi/2$ with the previous tangent.

Procedure 11.7 Tangent vector calculation

Compute a vector in the null space of $\begin{bmatrix} \nabla_x H & \nabla_\lambda H \end{bmatrix}$ by performing QR factorization with column pivoting

$$Q^T \begin{bmatrix} \nabla_x H & \nabla_\lambda H \end{bmatrix} \Pi = \begin{bmatrix} R & w \end{bmatrix}$$

where Q is $n \times n$ orthogonal

R is $n \times 1$ upper triangular

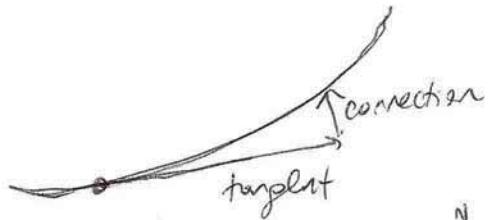
Π is an $(n+1) \times (n+1)$ permutation

$$w \in \mathbb{R}^n$$

$$\text{Set } v = \Pi \begin{bmatrix} R^{-1} w \\ -1 \end{bmatrix}$$

$$\text{Set } (\dot{x}, \dot{\lambda}) = \mp v / \|v\|_2.$$

- * The tangent vector calculation procedure can be used with a standard ordinary differential equation solver (e.g. Euler or Runge-Kutta). This will cause errors to accumulate.



Following the tangent interpretation, a correction step (root finding) can be applied.

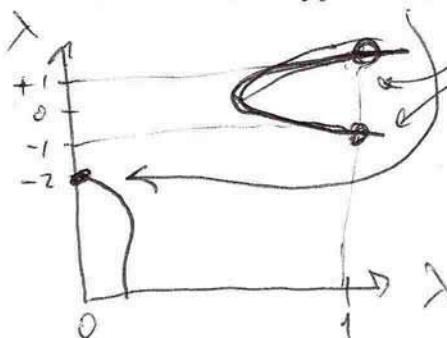
These methods assume that $\begin{bmatrix} \nabla_{x+} & \nabla_{\lambda+} \end{bmatrix}$ is full rank so that the tangent is well-defined.

Theorem 11.8 Suppose that r is twice cont. diff. Then for almost all vectors $a \in \mathbb{R}^n$, \exists a zero path emanating from $(0, a)$ along which the $n \times (n+1)$ matrix $\begin{bmatrix} \nabla_{x+} & \nabla_{\lambda+} \end{bmatrix}$ has full rank. If the path is bounded for $\lambda \in [0, 1]$, then it has an accumulation point $(\bar{x}, 1) \Rightarrow r(\bar{x}) = 0$.

Furthermore, if $J(\bar{x})$ is nonsingular, the zero path between $(a, 0)$ and $(\bar{x}, 1)$ has finite arc length.

~~E.g.~~ $r(x) = x^2 - 1$; nondegenerate solutions are ± 1 and -1 .

Choose $a = -2$. Here $H(x, \lambda) = \lambda(x^2 - 1) + (1 - \lambda)(x + 2)$



solutions.
 $= \lambda x^2 + (1 - \lambda)x + (2 - 3\lambda)$
 There is no path that connects $(-2, 0)$ to $(\pm 1, 1)$ and starting at $(-2, 0)$ we trace an unbounded path without reaching a solution.

11.1) Show that $\forall s \in \mathbb{R}^n$, $\left\| \frac{ss^T}{s^T s} \right\|_2 = 1$.

Trace identity for matrices: $\text{trace}(A_1 A_2 \dots A_m) = \text{trace}(A_2 \dots A_m A_1)$.

$$\left\| \frac{ss^T}{s^T s} \right\|_2 = \frac{\|ss^T\|_2}{(s^T s)} = \frac{\max_d \sigma_d(ss^T)}{s^T s} = \frac{\text{trace}(ss^T)}{s^T s} = \frac{\text{trace}(s^T s)}{s^T s} = 1$$

since $s^T s$
 is rank-1
 only one eigenvalue
 is nonzero

linearly
of norm.

11.5) Show that if $J^T r \neq 0$, then $\phi(\lambda) \triangleq \| (J^T J + \lambda I)^{-1} J^T r \|_2$ is monotonically decreasing.

Clearly $J^T r = 0 \Rightarrow \phi(\lambda) = 0$ regardless of λ . Now consider the case where $J^T r \neq 0$ and $J = USV^T$ is the SVD. Then $J^T J = V S^2 V^T$ and $J^T = V S U^T$ (since $S^T = S$). Substituting,

$$\begin{aligned} \phi(\lambda) &= \| (V S^2 V^T + \lambda I)^{-1} V S U^T r \|_2 \quad \text{let } \tilde{r} \triangleq U^T r. \\ &= \| V (S^2 + \lambda I)^{-1} V S \tilde{r} \|_2 = \| (S^2 + \lambda I)^{-1} S \tilde{r} \|_2 \quad \text{since } V \text{ is orthogonal.} \\ &= \| (S + \lambda S^{-1})^{-1} \tilde{r} \|_2 = \left(\sum_i \frac{\tilde{r}_i^2}{d_i} \right)^{1/2} \quad \text{where } d_i = s_i + \frac{\lambda}{s_i} \\ &\qquad\qquad\qquad = \frac{s_i^2 + \lambda}{s_i} \end{aligned}$$

Consider

$$\begin{aligned} \frac{\partial \phi(\lambda)}{\partial \lambda} &= \frac{1}{2} \left(\sum_i \frac{\tilde{r}_i^2}{d_i} \right)^{-1/2} \left(\sum_i -\frac{s_i \tilde{r}_i^2}{(s_i^2 + \lambda)^2} \right) \\ &= -\frac{1}{2} \left(\sum_i \frac{\tilde{r}_i^2}{d_i} \right)^{-1/2} \left(\sum_i \frac{s_i \tilde{r}_i^2}{(s_i^2 + \lambda)^2} \right) \end{aligned}$$

≤ 0 (since the parentheses are nonnegative
 - in fact they are positive in this case
 $\therefore \phi(\lambda)$ is monotonically decreasing. since $S > 0$).

Chapter 12: Theory of Constrained Optimization

In many optimization problems, the parameters are not allowed to take all possible values in \mathbb{R}^n .

$$\min_{x \in \mathbb{R}^n} f(x) \text{ subject to } \begin{array}{l} c_i(x) = 0 \quad i \in \mathcal{E} \\ c_i(x) \geq 0 \quad i \in \mathcal{I} \end{array}$$

where f and c_i are smooth, real-valued functions on a subset of \mathbb{R}^n , and \mathcal{E} and \mathcal{I} are two finite sets of indices, is the form of a typical constrained optimization problem, consisting of both equality and inequality constraints.

Defn For a constrained optimization problem such as the one given above, the feasible set \mathcal{S} is defined as follows:

$$\mathcal{S} = \{x \mid c_i(x) = 0, \forall i \in \mathcal{E}; c_i(x) \geq 0 \quad \forall i \in \mathcal{I}\}.$$

With this definition, we could rewrite the optimization problem as

$$\min_{x \in \mathcal{S}} f(x) \quad (\text{constrained opt. problem})$$

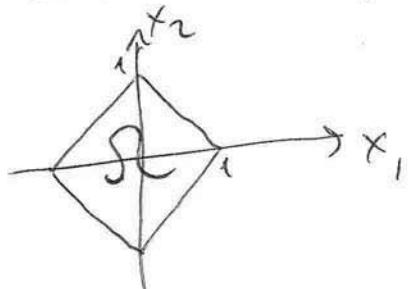
Defn: A vector x^* is a local solution of the constrained optimization problem iff $x^* \in S$ and $\exists N$, a neighborhood of x^* s.t. $f(x) \geq f(x^*) \quad \forall x \in N \cap S$.

Defn: x^* is a strict local solution iff $x^* \in S$ and $f(x) > f(x^*) \quad \forall x \in N \cap S$ with $x \neq x^*$.

Defn: x^* is an isolated local solution iff $x^* \in S$ and x^* is the only local solution in $N \cap S$.

Smoothness: Nonsmooth feasible set boundaries could be written as piecewise smooth constraints.

* Consider $\|x\|_1 \leq 1$ is the same as



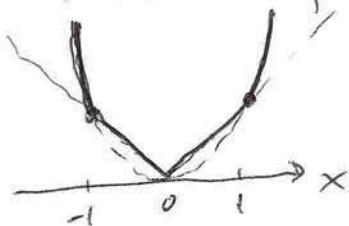
$$x_1 + x_2 \leq 1$$

$$x_1 - x_2 \leq 1$$

$$-x_1 + x_2 \leq 1$$

$$-x_1 - x_2 \leq 1$$

* Consider $f(x) = \max(x^2, x)$ is the same as ($\min_x f(x)$)



$$\begin{array}{ll} \min_t & \text{s.t. } t \geq x \\ t & t \geq x^2 \end{array}$$

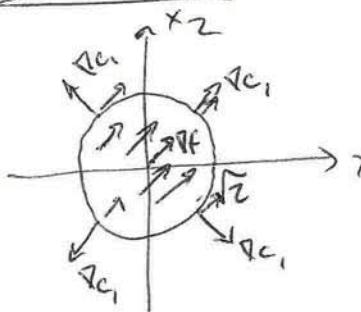
Defn: The active set $A(x)$ at any $x \in \mathbb{R}$ consists of the equality constraint indices from E together with the indices of the inequality constraints i for which $c_i(x) = 0$.

$$A(x) = E \cup \{i \in I \mid c_i(x) = 0\}$$

At a feasible point x , the inequality constraint $i \in I$ is said to be active if $c_i(x) = 0$ and inactive if $c_i(x) > 0$.

Example

$$\min_x x_1 + x_2 \quad \text{s.t. } x_1^2 + x_2^2 - 2 = 0$$



The feasible set is the circle with $r = \sqrt{2}$.

Clearly, the solution is $x^* = (-1, -1)^T$ on the circle.

At any other point, the directional derivative is not zero, or if 0, then the second directional derivative is not positive.

At the optimal solution $\nabla c_1(x^*)$ is parallel to $\nabla f(x^*)$.

$$\therefore \exists \lambda^* \quad (= 1/2 \text{ in this case}) \quad \Rightarrow \quad \nabla f(x^*) = \lambda^* \nabla c_1(x^*)$$

Let's consider a first-order Taylor approximation:

$$0 = c_1(x+s) \approx c_1(x) + \nabla c_1(x)^T s = \nabla c_1(x)^T s$$

For a small step s along the feasible set, we have both $c_1(x) = 0$ and $c_1(x+s) = 0$, hence we must have $\nabla c_1(x)^T s \approx 0$.

Similarly $0 > f(x+s) - f(x) \approx \nabla f(x)^T s$ for 242
 an s that produces a decrease in f .

$$\therefore \nabla f(x)^T s < 0$$

Therefore, we conclude that if $\exists d$ where $\|d\| \approx 1$, s.t.

$$\nabla c_1(x)^T d = 0 \text{ and } \nabla f(x)^T d < 0$$

then s such that $d \approx s/\|s\|$ will satisfy our criteria above. If $\nexists d$ satisfying these conditions, it is likely that we cannot find a suitable small s . In this latter case, x^* would be a local minimizer.

Notice that $\nexists d$ s.t. $\nabla c_1(x)^T d = 0 \wedge \nabla f(x)^T d < 0$ iff $\nabla c_1(x)$ and $\nabla f(x)$ are parallel, i.e. $\nabla f(x) = \lambda_1 \nabla c_1(x)$ for some λ_1 .

Fact * If $\nabla f(x)$ and $\nabla c_1(x)$ are not parallel, then

$$d = - \left(I - \frac{\nabla c_1(x) \nabla c_1(x)^T}{\|\nabla c_1(x)\|^2} \right) \nabla f(x); \quad d = \frac{-\nabla f(x)}{\|\nabla f(x)\|}$$

yields a d that satisfies the two conditions above.

The Lagrangian Function: $L(x, \lambda) \stackrel{\Delta}{=} f(x) - \lambda_1 c_1(x)$

Note that $\nabla_x L(x, \lambda) = \nabla f(x) - \lambda_1 \nabla c_1(x)$. At the solution $x^*, \exists \lambda_1^*$ s.t. $\nabla_x L(x^*, \lambda_1^*) = 0$ (i.e. $\nabla f(x^*) \parallel \nabla c_1(x^*)$).

∴ We can seek the solution of an equality-constrained problem by searching only among the stationary points of the Lagrangian function.

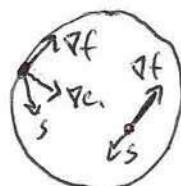
The parameter λ_1 is called the Lagrange multiplier for $c_1(x)$. Notice that $\nabla_x L(x^*, \lambda^*) = 0$ is necessary for x^* to be a solution, but it is not sufficient.

In the example above both $\begin{pmatrix} -1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ satisfy this condition. The latter is a local max.

Example. Now consider an inequality constraint

$$\min_{\mathbf{x}} x_1 + x_2 \quad \text{s.t. } 2 - x_1^2 - x_2^2 \geq 0$$

Here, S_2 is the ^{closed} disk centered at (0) with $r = \sqrt{2}$.



Here is an illustration of possible descent directions at two feasible points; one is on the boundary, the other is inside.

We notice that $x^* = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$ with $\lambda_1^* = \frac{1}{2}$ is still the solution to this problem. The sign of the Lagrange multiplier will become significant.

* Consider a feasible point x that is not optimal. Let's try to find (assume) a small step s that both retains feasibility and decreases f to the first order. If $\nabla f(x)^T s \leq 0$, the objective decreases to 1st order.

$(x+s)$ retains feasibility if $0 \leq c_i(x+s) \approx c_i(x) + \nabla c_i(x)^T s$.

∴ we need $c_i(x) + \nabla c_i(x)^T s \geq 0$.

CASE 1: Assume x lies strictly inside the circle. Then we have $c_i(x) > 0$ and any step s satisfies the sufficiently small

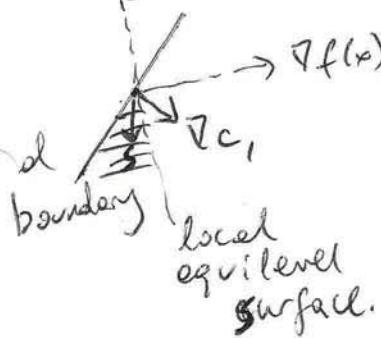
feasibility condition $c_i(x) + \nabla c_i(x)^T s \geq 0$. Whenever $\nabla f(x) \neq 0$, we can choose $s = -\alpha \nabla f(x)$ and adjust $\alpha > 0$ such that $c_i(x) - \alpha \nabla c_i(x)^T \nabla f(x) \geq 0$.

CASE 2: Assume x lies on the boundary of the circle.

Then we have $c_i(x) = 0$ and our conditions become

$$\nabla f(x)^T s < 0, \quad \nabla c_i(x)^T s \geq 0$$

These conditions define an open half-space and a closed half space, respectively.



The shaded area consists of all good search directions. The shaded area is a cone.

The optimality conditions for both cases can be summarized as follows--

Fact: When no first order feasible descent direction exists at some point x^* , we have that

$$\nabla_x L(x^*, \lambda^*) = 0 \text{ for some } \lambda^* \geq 0$$

$$(\text{complementarity condition}) \lambda_i^* c_i(x^*) = 0$$

Notice that in CASE 1, $c_i(x^*) > 0$ and $\lambda_i^* = 0$
(as if there are no constraints) $\Rightarrow \nabla f(x^*) = 0$

in CASE 2, $\lambda_i^* > 0 \Rightarrow \nabla f(x^*) = \lambda_i^* \nabla c_i(x^*)$
(as if we have an equality constraint)

Two Inequality Constraints

Example $\min_x x_1 + x_2 \text{ s.t. } 2 - x_1^2 - x_2^2 \geq 0$

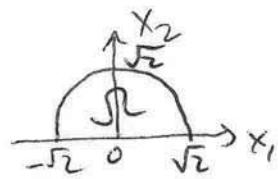
The feasible set S_2 is a half-disk. $x_2 \geq 0$

Clearly, the solution is $x^* = \begin{pmatrix} -\sqrt{2} \\ 0 \end{pmatrix}$ and here both inequality constraints are active.

A first order feasible descent direction d would satisfy

$$\nabla c_i(x)^T d \geq 0 \quad i=1,2 \quad \text{and} \quad \nabla f(x)^T d < 0$$

However, for $\begin{pmatrix} -\sqrt{2} \\ 0 \end{pmatrix}$, we cannot find d that satisfies all of these conditions.



Consider the Lagrangian:

$$\mathcal{L}(x, \lambda) = f(x) - \lambda_1 c_1(x) - \lambda_2 c_2(x)$$

elementwise
 $\lambda_1^* \geq 0$
 $\lambda_2^* \geq 0$

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0 \quad \text{for some } \lambda^* \geq 0$$

The complementarity condition: $\lambda_1^* c_1(x^*) = 0$
 $\lambda_2^* c_2(x^*) = 0$

* For $x^* = \begin{pmatrix} -\sqrt{2} \\ 0 \end{pmatrix}$: $\nabla f(x^*) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $\nabla c_1(x^*) = \begin{bmatrix} 2\sqrt{2} \\ 0 \end{bmatrix}$, $\nabla c_2(x^*) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
so $\lambda^* = \begin{bmatrix} 1/(2\sqrt{2}) \\ 1 \end{bmatrix} \Rightarrow \nabla_x \mathcal{L}(x^*, \lambda^*) = 0$.

Since $\lambda^* \geq 0$, the Lagrange constraint is satisfied.

* Consider a point that is not the solution.

Let $x = \begin{pmatrix} \sqrt{2} \\ 0 \end{pmatrix}$. Clearly $d = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$ is a feasible descent direction. For this x , we have $\nabla_x \mathcal{L}(x, \lambda) = 0$ only when $\lambda = \begin{bmatrix} -1/(2\sqrt{2}) \\ 1 \end{bmatrix}^T$, but here $\lambda_1 < 0$ so this point is not a ~~local~~ feasible local minimum.

* Consider $x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ at which only c_2 is active.

Any ^{small} step s will continue to satisfy $c_1(x+s) > 0$ (inside the whole circle) so we only need to consider c_2 and f .

$$\nabla c_2(x)^T d \geq 0 \quad \nabla f(x)^T d < 0$$

We have $\nabla f(x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $\nabla c_2(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $d = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$ satisfies these two conditions, hence is a feasible descent direction (among others).

Since $c_1(x) > 0$, we must have $\lambda_1 = 0$. So to satisfy $\nabla_x L(x, \lambda)$ while $\lambda_1 = 0$, we search for λ_2 such that $\nabla f(x) - \lambda_2 \nabla c_2(x) = 0$, but we cannot find such λ_2 ; i.e. $\forall \lambda_2 [1] - \lambda_2 [1] \neq [0]$. Hence $x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ is not a local optimum.

Tangent Cone and Constraint Qualifications

We will define the following:

- 1) $T_{\mathcal{R}}(x^*)$: the tangent cone to the convex set \mathcal{R} at $x^* \in \mathcal{R}$
- 2) $F(x^*)$: the set of first-order feasible directions at x^* .

We will also introduce constraint qualifications; these are assumptions that ensure ensure the similarity of the constraint set \mathcal{R} and its local linear approximation in a neighborhood of x^* .

Defn: Given a feasible point x , we call $\{z_k\}$ a feasible sequence approaching x iff $z_k \in \mathcal{R} \ \forall k$ sufficiently large and $z_k \rightarrow x$.

Defn. The vector d is said to be a tangent to \mathcal{R} at a point x if there exists a feasible sequence $\{z_k\}$ approaching x and a sequence of positive scalars $\{t_k\}$ with $t_k \rightarrow 0$ such that $\lim_{k \rightarrow \infty} \frac{z_k - x}{t_k} = d$

The set of all tangents to \mathcal{R} at x^* is called the tangent cone and is denoted by $T_{\mathcal{R}}(x^*)$.

Recall the definition of a cone:

A set F is a cone iff $x \in F \Rightarrow \alpha x \in F \forall \alpha > 0$.

Cone of tangent vectors d could be obtained by replacing $\{t_k\}$ with $\{\alpha^{-1} t_k\}$ for $\alpha > 0$.

Note that setting $\{t_k\} = x$ yields $0 \in T_{\mathcal{R}}(x)$.

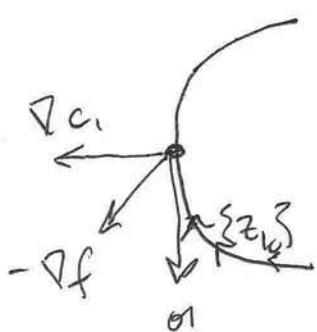
Defn Given a feasible point x and the active constraint set $A(x)$, the set of linearized feasible directions is

$$F(x) = \left\{ d \mid \begin{array}{l} d^T \nabla c_i(x) = 0 \quad \forall i \in E \\ d^T \nabla c_i(x) \geq 0 \quad \forall i \in A(x) \setminus E \end{array} \right\}$$

Note that $F(x)$ is also a cone.

Example Consider again $\min_{\mathbf{x}} \mathbf{x}_1 + \mathbf{x}_2$ s.t. $\mathbf{x}_1^2 + \mathbf{x}_2^2 - 2 = 0$

Near the nonoptimal point $\mathbf{x} = \begin{pmatrix} -5 \\ 0 \end{pmatrix}$ we have



Let the sequences be

$$\mathbf{z}_k = \begin{cases} -\sqrt{2-1/k^2} \\ -1/k \end{cases} \Rightarrow \mathbf{d} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$t_k = \|\mathbf{z}_k - \mathbf{x}\|$$

For this sequence, we have $f(\mathbf{z}_{k+1}) > f(\mathbf{z}_k)$.

Also we have $f(\mathbf{z}_k) < f(\mathbf{x})$. Clearly, \mathbf{x} cannot be a solution of the problem.

Consider $\mathbf{z}_k = \begin{cases} -\sqrt{2-1/k^2} \\ 1/k \end{cases}$ which approaches \mathbf{x} from the top-right side. Then $f(\mathbf{z}_{k+1}) > f(\mathbf{z}_k)$ and $f(\mathbf{x}) < f(\mathbf{z}_k)$. This time $\mathbf{d} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

A vector \mathbf{d} is in $\mathcal{F}(\mathbf{x})$ if

$$0 = \nabla e_1(\mathbf{x})^T \mathbf{d} = \{2x_1, 2x_2\} \begin{bmatrix} \frac{\mathbf{d}_1}{\mathbf{d}_2} \end{bmatrix} = -2\sqrt{2} \mathbf{d},$$

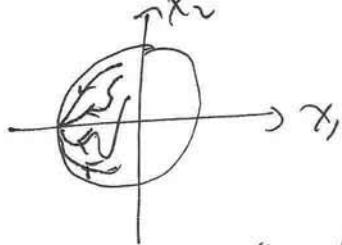
So we obtain $\mathcal{F}(\mathbf{x}) = \left\{ \begin{pmatrix} 0 \\ \mathbf{d}_2 \end{pmatrix} \mid \mathbf{d}_2 \in \mathbb{R} \right\}$. Similarly we had, $\mathcal{T}_{S2}(\mathbf{x}) = \left\{ \begin{pmatrix} 0 \\ \mathbf{d}_2 \end{pmatrix} \mid \mathbf{d}_2 \in \mathbb{R} \right\}$. Here $\mathcal{F}(\mathbf{x}) = \mathcal{T}_{S2}(\mathbf{x})$.

Suppose that the feasible set was algebraically expressed as $S_2 = \{x \mid c_1(x) = 0\}$ & $c_1(x) = (x_1^2 + x_2^2 - 2)^2 = 0$. This S_2 is the same as before but its expression is different. Then

$$Q = \nabla c_1(x)^\top d = \begin{bmatrix} 4(x_1^2 + x_2^2 - 2)x_1 \\ 4(x_1^2 + x_2^2 - 2)x_2 \end{bmatrix}^\top \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}^\top \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

$\Leftrightarrow F(x) = \mathbb{R}^2$ (since this is true for all d). In this case $F(x) \neq T_{S_2}(x)$.

Example $\min_x x_1 + x_2 \text{ s.t. } 2 - x_1^2 - x_2^2 \geq 0$



for point $x = \begin{pmatrix} -\sqrt{2} \\ 0 \end{pmatrix}$ there are infinitely many feasible sequences. These include straight lines connecting to x following

a linear trajectory: $z_k = \begin{pmatrix} -\sqrt{2} \\ 0 \end{pmatrix} + \frac{1}{k} w$

where w is any vector with $w_i > 0$ and $\|w\| \leq \sqrt{2}$.

~~eg.~~ The tangent cone is $T_{S_2}(x) = \left\{ \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \mid w_i \geq 0 \right\}$

The feasible set is: $0 \leq \nabla c_1(x)^\top d = [-2x_1 - 2x_2] \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = 2\sqrt{2} d_1$

Hence $F(x) = T_{S_2}(x)$ for this algebraic specification of S_2 .

Defn (LICQ)

Given the point x and the active set $A(x)$, we say that the linear independence constraint qualification (LICQ) holds if the set of active constraint gradients $\{\nabla c_i(x), i \in A(x)\}$ is linearly independent.

Fact: If LICQ holds, none of the active constraint gradients can be zero.

First-order optimality conditions

Let the Lagrangian be defined as

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathbb{E} \cup \mathbb{I}} \lambda_i c_i(x)$$

Thm 12.1 First-order Necessary Conditions

Suppose that x^* is a local solution of

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \begin{aligned} c_{\mathbb{E}}(x) &= 0, \quad i \in \mathbb{E} \\ c_{\mathbb{I}}(x) &\geq 0, \quad i \in \mathbb{I} \end{aligned}$$

that f and c_i are cont. diff., and that the LICQ holds at x^* . Then $\exists \lambda^*$, a Lagrange multiplier vector such that the following conditions are satisfied at (x^*, λ^*) :

$$\begin{array}{lll}
 \nabla_x L(x^*, \lambda^*) = 0 & \text{Thm -} \\
 c_i(x^*) = 0 \quad \forall i \in E & (12.1a) \\
 c_i(x^*) \geq 0 \quad \forall i \in I & (12.1b) \\
 \lambda_i^* \geq 0 \quad \forall i \in \bar{I} & (12.1c) \\
 \lambda_i^* c_i(x^*) = 0 \quad \forall i \in E \cup I & (12.1d) \\
 (\text{complementarity}) \quad \lambda_i^* c_i(x^*) = 0 & (12.1e)
 \end{array}$$

KKT α -P

These conditions are called the Karush-Kuhn-Tucker conditions (KKT).

The complementarity condition $\lambda_i^* c_i(x^*) = 0$ implies that $c_i(x^*)$ is active or $\lambda_i^* = 0$ (logical or). If we omit $i \notin A(x^*)$, then the Lagrangian condition becomes $0 = \nabla_x L(x^*, \lambda^*) = \nabla f(x^*) - \sum_{i \in A(x^*)} \lambda_i^* \nabla c_i(x^*)$

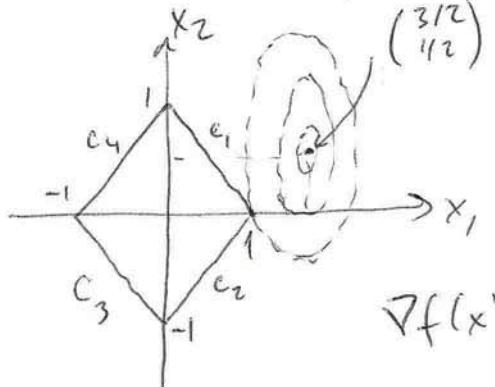
Defn: Strict Complementarity

Given a local solution x^* and a vector λ^* satisfying the KKT conditions, we say that the strict complementarity condition holds iff exactly one of λ_i^* and $c_i(x^*)$ is zero (i.e. $\lambda_i^* = 0$ XOR $c_i(x^*) = 0$) for each $i \in \bar{I}$. In other words we have $\lambda_i^* > 0 \quad \forall i \in I \cap A(x^*)$.

* Ideally, strict complementarity conditions will result in faster convergence of algorithms since the active set $A(x^*)$ can be determined easily.

* For a given problem, a solution x^* might have many λ^* such that KKT cond. are satisfied. If LICQ holds, then λ^* is unique.

Example $\min_x (x_1 - \frac{3}{2})^2 + (x_2 - \frac{1}{2})^4$ s.t. $\begin{cases} 1-x_1-x_2 \\ 1-x_1+x_2 \\ 1+x_1-x_2 \\ 1+x_1+x_2 \end{cases} \geq 0$



The solution is $x^* = \begin{pmatrix} 3/4 \\ 1/4 \end{pmatrix}$.

The first and second constraints are active at x^* .

$$\nabla f(x^*) = \begin{bmatrix} -1 \\ -1/2 \end{bmatrix} \quad \nabla c_1(x^*) = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad \nabla c_2(x^*) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

and $\lambda^* = \begin{bmatrix} 3/4 \\ 1/4 \\ 0 \\ 0 \end{bmatrix}$ satisfies the KKT conditions.

Relating the Tangent Cone and the First-Order Feasible Set

We define the matrix $A(x^*)$ whose rows are the active constraint gradients at the optimal point.

$$A(x^*)^T \stackrel{\Delta}{=} [\dots \nabla c_i(x^*) \dots]_{i \in A(x^*)}$$

Lemme 12-2 Let x^* be a feasible point. The following two statements are true

$$(i) T_{\mathcal{R}}(x^*) \subset \mathcal{T}(x^*)$$

(ii) If the LICQ condition is satisfied at x^* then $\mathcal{T}(x^*) = T_{\mathcal{R}}(x^*)$

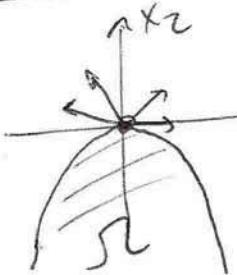
Proof See pages 323-325 in the book.

Thm 12-3 If x^* is a local solution, then we have $\nabla f(x^*)^T d \geq 0 \quad \forall d \in T_{\mathcal{R}}(x^*)$.

Proof See pages 325-326 in the book.

* Note that the converse of Thm 12-3 is NOT true. We could have a point where $\nabla f(x)^T d \geq 0 \quad \forall d \in T_{\mathcal{R}}(x)$ but that x may not be a local solution.

Example $\min x_2$ s.t. $x_2 \geq -x_1^2$



At $x = (0, 0)$ all limiting directions must have $d_2 \geq 0$ (where $d = (d_1, d_2)$) and d_1 is free.

so $\nabla f(x)^T d = d_2 \geq 0$. However, clearly $x = (0, 0)$ is not a local minimizer (it is a maximizer).

Farka's Lemma

This Lemma considers a cone K defined as:

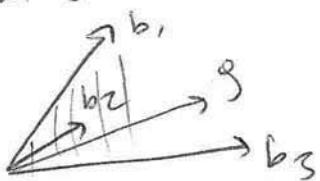
$$K = \{By + Cw \mid y \geq 0\}$$

where B and C are matrices in $\mathbb{R}^{n \times m}$ and $\mathbb{R}^{1 \times p}$; y and w are vectors in \mathbb{R}^m and \mathbb{R}^p , respectively.

Given a vector $g \in \mathbb{R}^n$, the lemma states that $C^T d = 0$

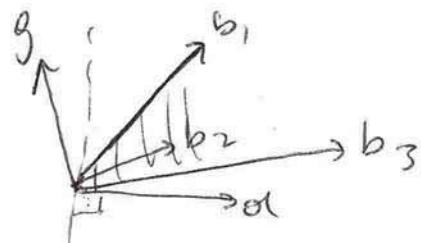
$g \in K \Leftrightarrow \exists d \in \mathbb{R}^p \ni g^T d \geq 0, B^T d \geq 0, C^T d = 0$ as

illustrated below.



$g \in K$

\Leftrightarrow



There is a separating hyperplane defined by d .

Lemma 12.4 (Farka's Lemma)

Let the cone K be defined as above. Given any $g \in \mathbb{R}^n$, we have either that $g \in K$ or that $\exists d \in \mathbb{R}^p$ satisfying the three equalities and inequalities given above, but not both.

Proof. See pages 327-328 in the book.

We can now prove Thm 12.1 (KKT).

Proof of Thm 12-1:

Suppose that $x \in \mathbb{R}^n$ is a feasible point at which the LICQ holds. The theorem claims that if x^* is a local solution then $\exists x^* \in \mathbb{R}^m$ that satisfies KKT.

Thm 12.3 tells that $d^T \nabla f(x^*) \geq 0 \quad \forall d \in T_{\mathcal{S}}(x^*)$.

Since LICQ holds, from Lemma 12-2, we have $F(x^*) = T_{\mathcal{S}}(x^*)$.

Consequently $d^T \nabla f(x^*) \geq 0 \quad \forall d \in F(x^*)$. Hence,

from Fermat's Lemma (Lemma 12-4) $\exists \lambda$ for which the following hold:

$$\nabla f(x^*) = \sum_{i \in A(x^*)} \lambda_i \nabla c_i(x^*) = A(x^*)^\top \lambda^* \quad (\star)$$

$$\lambda_i \geq 0 \text{ for } i \in A(x^*) \cap I,$$

(This is obtained by observing that for the cone N :

$$N = \left\{ \sum_{i \in A(x^*)} \lambda_i \nabla c_i(x^*) \mid \lambda_i \geq 0 \text{ for } i \in A(x^*) \cap I \right\}$$

and $g = \nabla f(x^*)$, either the expression above is true or $\exists d \in \mathbb{R}^m \setminus \{0\} \mid d^T \nabla f(x^*) < 0$.)

Now define $\lambda_i^* = \begin{cases} \lambda_i, & i \in A(x^*) \\ 0, & i \in I \setminus A(x^*) \end{cases}$

for this (x^*, λ^*) pair, we see that

- * KKT a follows immediately from the expression (*) above and the definition of the Lagrangian.
- * Since x^* is feasible KKT b and KKT c are satisfied.
- * From (*) above $\lambda_i^* \geq 0$ for $i \in A(x^*) \cap I$, while $\lambda_i^* = 0$ for $i \in I \setminus A(x^*)$. Hence $\lambda_i^* \geq 0$ for $i \in I$ so KKT d holds.
- * For $i \in A(x^*) \cap I$, $c_i(x^*) = 0$, while for $i \in I \setminus A(x^*)$, $\lambda_i^* = 0$. Hence $\lambda_i^* c_i(x^*) = 0$ for $i \in I$, so KKT e is satisfied. \square

Second-Order Conditions

The KKT conditions tell us how the first-order derivatives of f and c_i relate to each other at a local solution. This means, at x^* , for any $w \in F(x^*)$, we have $w^T \nabla f(x^*) \geq 0$ at a first-order approximation.

The second order derivatives will be useful as a tie-breaker when $w^T \nabla f(x^*) = 0$ for various $w \in F(x^*)$. For this purpose, assume that f and c_i are all twice continuously differentiable.

Defn. Given $\mathcal{F}(x^*)$ and λ^* satisfying the KKT cond.
we define the critical cone $C(x^*, \lambda^*)$ as

$$C(x^*, \lambda^*) = \left\{ \omega \in \mathcal{F}(x^*) \mid \nabla c_i(x^*)^\top \omega = 0, \text{ all } i \in A(x^*) \cap I \right. \\ \left. \text{with } \lambda_i^* > 0 \right\}$$

Equivalently

$$\omega \in C(x^*, \lambda^*) \Leftrightarrow \begin{cases} \nabla c_i(x^*)^\top \omega = 0 & \forall i \in E \\ \nabla c_i(x^*)^\top \omega = 0 & \forall i \in A(x^*) \cap I, \lambda_i^* > 0 \\ \nabla c_i(x^*)^\top \omega \geq 0 & \forall i \in A(x^*) \cap I, \lambda_i^* = 0 \end{cases}$$

Since $\lambda_i^* = 0$ for all inactive components $i \in I \setminus A(x^*)$,

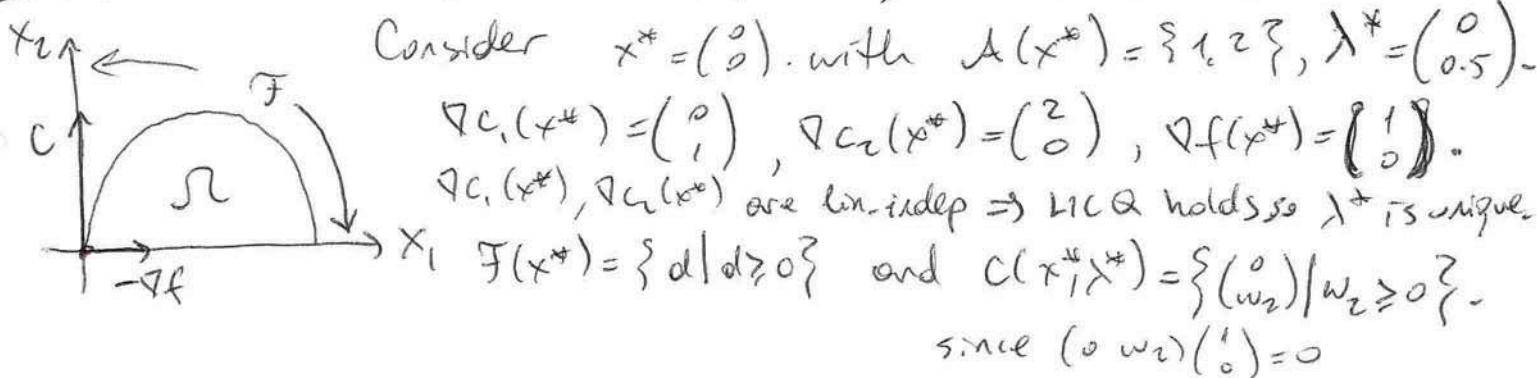
we have $\omega \in C(x^*, \lambda^*) \Rightarrow \lambda_i^* \nabla c_i(x^*)^\top \omega = 0 \quad \forall i \in E \cup I$.

Hence from KKT_a we have

$$\omega \in C(x^*, \lambda^*) \Rightarrow \omega^\top \nabla f(x^*) = \sum_{i \in E \cup I} \lambda_i^* \omega^\top \nabla c_i(x^*) = 0.$$

Here, the critical cone contains directions from $\mathcal{F}(x^*)$ for which $\omega^\top \nabla f(x^*) = 0$, so it is not clear from first-order information whether f will increase or decrease.

Example $\min x_1$ s.t. $x_2 \geq 0$, $1 - (x_1 - 1)^2 - x_2^2 \geq 0$



Theorem 12.5 Second-Order Necessary Conditions

Suppose that x^* is a local solution of the constrained optimization problem and that the LIOQ condition is satisfied. Let λ^* be the Lagrange multiplier vector for which KKT cond. hold. Then

$$w^T \nabla_{xx}^2 L(x^*, \lambda^*) w \geq 0 \quad \forall w \in C(x^*, \lambda^*).$$

Proof: See pages 332-333 in the book.

Theorem 12.6 Second-Order Sufficient Conditions

Suppose that for some feasible point $x^* \in \mathbb{R}^n$, there is a Lagrange multiplier vector λ^* such that the KKT cond. hold. Also suppose that

$$w^T \nabla_{xx}^2 L(x^*, \lambda^*) w > 0 \quad \forall w \in C(x^*, \lambda^*), w \neq 0.$$

Then x^* is a strict local solution.

Proof: See pages 333-335 in the book.
Question: Why is the inequality required to be strict?

Example $f(x) = x_1 + x_2$ $c_i(x) = 2 - x_1^2 - x_2^2$ $\mathcal{E} = \emptyset$ $\mathcal{I} = \{1\}$

$$\mathcal{L}(x, \lambda) = (x_1 + x_2) - \lambda_1(2 - x_1^2 - x_2^2)$$

We see that the KKT cond. are satisfied for

$x^* = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$ with $\lambda_1^* = \frac{1}{2}$. We have

$$\nabla_{xx}^2 \mathcal{L}(x^*, \lambda^*) = \begin{bmatrix} 2\lambda_1^* & 0 \\ 0 & 2\lambda_1^* \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} > 0$$

∴ The conditions of Thm 12.6 are satisfied and we conclude that $x^* = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$ is a strict local min.

Example $\min -0.1(x_1 - 4)^2 + x_2^2$ s.t. $x_1^2 + x_2^2 - 1 \geq 0$.

This is a nonconvex function minimized on the exterior of a circle, hence the objective is unbounded from below. Consider the feasible sequence $z_k = \begin{pmatrix} 10k \\ 0 \end{pmatrix}$.

Clearly $f(z_k) \rightarrow -\infty$ so no global solution exists.

Let's see if there is a strict local solution on the constraint boundary.

$$\nabla_x \mathcal{L}(x, \lambda) = \begin{bmatrix} -0.2(x_1 - 4) - 2\lambda_1 x_1 \\ 2x_2 - 2\lambda_1 x_2 \end{bmatrix}$$

$$\nabla_{xx}^2 \mathcal{L}(x, \lambda) = \begin{bmatrix} -0.2 - 2\lambda_1 & 0 \\ 0 & 2 - 2\lambda_1 \end{bmatrix}$$

At $x^* = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ the KKT cond. hold with $\lambda^* = 0.3$ and $A(x^*) = \{1\}$. We note that $\nabla c_i(x^*) = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$

so $C(x^*, \lambda^*) = \left\{ \begin{pmatrix} 0 \\ w_2 \end{pmatrix} \mid w_2 \in \mathbb{R} \right\}$. For any $w \in C(x^*, \lambda^*)$ with $w \neq 0$ (i.e. $w_2 \neq 0$), we have

$$w^T \nabla_{xx}^2 L(x^*, \lambda^*) w = [0 \ w_2] \begin{bmatrix} -0.4 & 0 \\ 0 & 1.4 \end{bmatrix} \begin{bmatrix} 0 \\ w_2 \end{bmatrix} = 1.4w_2^2 > 0.$$

Hence $x^* = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ is a strict local solution.

Second-Order Conditions and Projected Hessians

We are actually only concerned with the portion of the Lagrangian Hessian $\nabla_{xx}^2 L(x^*, \lambda^*)$ that ~~is~~ related to $C(x^*, \lambda^*)$.

* If λ^* that satisfies the KKT conditions is unique and if strict complementarity holds, $C(x^*, \lambda^*)$ reduces to

$$C(x^*, \lambda^*) = \text{Null} \left[\nabla c_i(x^*)^\top \right]_{i \in A(x^*)} = \text{Null } A(x^*).$$

Let Z be a full^{rank} column matrix whose columns span the space $C(x^*, \lambda^*)$; $C(x^*, \lambda^*) = \{z u \mid u \in \mathbb{R}^{1 \times |A(x^*)|}\}$.

Hence for Thm 12.5 we only need that

$$Z^T \nabla_{xx}^2 L(x^*, \lambda^*) Z \geq 0$$

or for Thm 12.6 we need $Z^T \nabla_{xx}^2 L(x^*, \lambda^*) Z > 0$.

* \mathbf{z} can be computed via QR factorization:

$$\mathbf{A}(\mathbf{x}^*)^T = \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ 0 \end{bmatrix} = [\mathbf{Q}_1 \quad \mathbf{Q}_2] \begin{bmatrix} \mathbf{R} \\ 0 \end{bmatrix} = \mathbf{Q}_1 \mathbf{R}$$

where $\mathbf{Q} \sim n \times n$ orthogonal, $\mathbf{R} \sim$ square upper triangular.

$$\mathbf{R} \sim \text{nonsingular} \Rightarrow \mathbf{z} \stackrel{\Delta}{=} \mathbf{Q}_2$$

$\mathbf{R} \sim \text{singular} \Rightarrow$ Use column-pivoting during QR to find \mathbf{z} .

* \mathbf{R} is singular iff the active constraint gradients are linearly dependent.

Other Constraint Qualifications

Lemma 12.7 Suppose that at some $\mathbf{x}^* \in \mathcal{S}$, all active constraints $c_i(\cdot)$, $i \in A(\mathbf{x}^*)$, are linear functions. Then $\mathbf{F}(\mathbf{x}^*) = T_{\mathcal{S}}(\mathbf{x}^*)$.

Proof: See pages 338-339 in the book.

Defn (MFCQ) Mangasarian-Fromovitz constraint qualification
We say that the MFCQ holds iff $\exists w \in \mathbb{R}^n \neq$

$$\nabla c_i(\mathbf{x}^*)^T w > 0 \quad \forall i \in A(\mathbf{x}^*) \cap I$$

$$\nabla c_i(\mathbf{x}^*)^T w = 0 \quad \forall i \in E$$

and $\{\nabla c_i(\mathbf{x}^*) \mid i \in E\}$ is linearly independent.

Fact If LICQ holds, then MFCQ holds.

Note that the converse is not true in general.

Fact A theorem that extends KKT conditions to the case where MFCQ is used in Thm 12.1 in place of LICQ can be proven.

Fact Constraint qualifications are SUFFICIENT conditions for the linear approximation to be adequate, not necessary conditions.

Example Consider the constraints $x_2 \geq -x_1^2$ and $x_2 \leq x_1^2$ with feasible point $x^* = (\frac{1}{2}, \frac{1}{4})$. At x^* , none of the qualifications are satisfied but $\nabla f(x^*) = \left\{ \begin{pmatrix} w_i \\ 0 \end{pmatrix} \mid w_i \in \mathbb{R} \right\}$ accurately reflects the local linear geometry of S_2 near x^* .

A Geometric View Point

The first-order optimality conditions above were dependent on the algebraic description of the constraints. Here we develop a geometric condition based on the geometry of S_2 .

The problem can be stated as $\min_{x \in \mathcal{R}} f(x)$.

Defn The normal cone to the set \mathcal{R} at $x \in \mathcal{R}$ is defined as $N_{\mathcal{R}}(x) = \{v | v^T w \leq 0 \text{ } \forall w \in T_{\mathcal{R}}(x)\}$.

Each vector $v \in N_{\mathcal{R}}(x)$ is said to be a normal vector.

Fact: Geometrically, the angle between any $v \in N_{\mathcal{R}}(x)$ and $w \in T_{\mathcal{R}}(x)$ is at least $\pi/2$. To see this notice that both $N_{\mathcal{R}}(x)$ and $T_{\mathcal{R}}(x)$ are cones, so we can consider only the case $\|v\|_2 = 1 = \|w\|_2$. Then the condition $v^T w \leq 0 \Leftrightarrow \cos \theta_{vw} \leq 0$.

Thm 12.8 Suppose that x^* is a local minimizer of f in \mathcal{R} . Then $-\nabla f(x^*) \in N_{\mathcal{R}}(x^*)$.

Proof: Given any $d \in T_{\mathcal{R}}(x^*)$, we have for sequences $\{t_k\}, \{z_k\}$ that $z_k \in \mathcal{R}, z_k = x^* + t_k d + o(t_k) \forall k$. Since x^* is a local solution, we must have $f(z_k) \geq f(x^*)$ for all k sufficiently large. Since f is continuously differentiable, we have from Taylor's theorem that

$$f(z_k) - f(x^*) = t_k \nabla f(x^*)^T d + o(t_k) \geq 0.$$

Dividing by t_k and taking the limit as $k \rightarrow \infty$, we have

$$\nabla f(x^*)^T d \geq 0$$

Recall that $d \in T_{\mathcal{R}}(x^*)$ is arbitrary so, $-\nabla f(x^*)^T d \leq 0 \forall d \in T_{\mathcal{R}}(x^*)$. Consequently, $-\nabla f(x^*) \in N_{\mathcal{R}}(x^*)$. \square

Lemma 12.9 Suppose that the LICQ assumption holds at x^* . Then the normal cone $N_{\mathcal{R}}(x^*)$ is simply $-N$ where

$$N \triangleq \left\{ \sum_{i \in A(x^*)} \lambda_i \nabla c_i(x^*) \mid \lambda_i \geq 0 \right\}$$

Proof: This follows from Farkas' lemma. We have

$$g \in N \Rightarrow g^\top d \geq 0 \quad \forall d \in \mathcal{F}(x^*) \text{ from Lemma 12.4 (Farkas).}$$

Since $\mathcal{F}(x^*) = T_{\mathcal{R}}(x^*)$ from Lemma 12.2 (since LICQ holds),

letting $g \in -N \Rightarrow g^\top d \leq 0 \quad \forall d \in T_{\mathcal{R}}(x^*)$.

$$\therefore N_{\mathcal{R}}(x^*) = -N$$

* Lemma 12.15. The set N is closed. (See pages 350-351 for proof).
 * This result shows the close relationship between the normal cone $N_{\mathcal{R}}(x^*)$ and the conic combination of active constraint gradients N given LICQ holds.

Lagrange Multipliers and Sensitivity

Each Lagrange multiplier λ_i^* tells us something about the sensitivity of the optimal value $f(x^*)$ to the presence of the constraint c_i ; i.e. it shows how hard f is pushing or pulling x^* against c_i .

From the ^{LICQ} conditions, we observe that for inactive constraints, we have $c_i(x^*) > 0 \Rightarrow \lambda_i^* = 0$ and x^* is indifferent to the presence of c_i .

Now consider an active constraint c_i and instead of requiring $c_i(x) \geq 0$, require that $c_i(x) \geq -\epsilon \|\nabla c_i(x^*)\|$, which is a small perturbation in the constraint boundary.

Suppose that ϵ is small enough such that $x^*(\epsilon)$ has the same set of active constraints and that the Lagrange multipliers are not affected much. Then

$$-\epsilon \|\nabla c_i(x^*)\| = c_i(x^*(\epsilon)) - c_i(x^*) \approx (x^*(\epsilon) - x^*)^T \nabla c_i(x^*)$$

$$0 = c_j(x^*(\epsilon)) - c_j(x^*) \approx (x^*(\epsilon) - x^*)^T \nabla c_j(x^*)$$

Notice that here we used a forward difference approximation $\nabla_j c_i(x^*)$, $j \neq i$.

The value of $f(x^*(\epsilon))$ is

$$f(x^*(\epsilon)) - f(x^*) \approx (x^*(\epsilon) - x^*)^T \nabla f(x^*)$$

$$= \sum_{j \in A(x^*)} \lambda_j^* (x^*(\epsilon) - x^*)^T \nabla c_j(x^*)$$

$$\approx -\epsilon \|\nabla c_i(x^*)\| \lambda_i^*.$$

Taking the limit as $\epsilon \rightarrow 0$, we get

$$\frac{df(x^*(\epsilon))}{d\epsilon} = -\lambda_i^* \|\nabla c_i(x^*)\|$$

If $\lambda_i^* = 0$ for an active constraint (e.g. if $\nabla f(x^*) = 0$ and x^* is on the boundary of S), then to first-order, the placement of c_i does not affect the optimal f .

- Defn: Let x^* be a solution of the constrained opt. problem and suppose that the KKT cond. hold. We say that an inequality constraint c_i , if
- (i) is strongly active (binding) if $i \in A(x^*)$ and $\lambda_i^* > 0$ for some λ^* satisfying KKT cond.
 - (ii) is weakly active if $i \in A(x^*)$ and $\lambda_i^* = 0$ for all λ^* satisfying KKT conditions.

Duality

Consider a primal problem with no equality constraints.

$$\min_{x \in \mathbb{R}^n} f(x) \text{ s.t. } c_i(x) \geq 0 \quad i = 1, 2, \dots, m.$$

Define the constraint vector $c(x) \triangleq \begin{bmatrix} c_1(x) \\ c_2(x) \\ \vdots \\ c_m(x) \end{bmatrix}$,

then $\min_{x \in \mathbb{R}^n} f(x)$ s.t. $c(x) \geq 0$; for which the

Lagrangian is $L(x, \lambda) = f(x) - \lambda^T c(x)$.

The dual objective function $g: \mathbb{R}^m \rightarrow \mathbb{R}$ is

$$g(\lambda) \triangleq \inf_x L(x, \lambda).$$

Many times the infimum is $-\infty$ for some λ .

The domain of g is defined as the set of λ for which g is finite:

$$D = \{ \lambda | g(\lambda) > -\infty \}$$

In general, finding the global minimizer over x may not be possible. If f and $-c$ are convex functions and $\lambda \geq 0$, $L(\cdot, \lambda)$ is convex and all local minimizers are global minimizers.

The dual problem is

$$\max_{\lambda \in \mathbb{R}^n} g(\lambda) \text{ s.t. } \lambda \geq 0$$

Example: $\min_{(x_1, x_2)} 0.5(x_1^2 + x_2^2)$ s.t. $x_1 - 1 \geq 0$

$$\text{Then } L(x_1, x_2, \lambda_1) = 0.5(x_1^2 + x_2^2) - \lambda_1(x_1 - 1).$$

$$\begin{aligned} \frac{\partial L(x_1, x_2, \lambda_1)}{\partial x_1} &= x_1 - \lambda_1 = 0 & g(\lambda_1) &= \inf_x L(x, \lambda_1) \\ \frac{\partial L(x_1, x_2, \lambda_1)}{\partial x_2} &= x_2 = 0 & &= 0.5(\lambda_1^2 + 0^2) - \lambda_1(\lambda_1 - 1) \\ & & &= -0.5\lambda_1^2 + \lambda_1 \end{aligned}$$

The dual problem is $\max_{\lambda_1} (-0.5\lambda_1^2 + \lambda_1)$ s.t. $\lambda_1 \geq 0$.

Clearly, $\lambda_1^* = 1$ is the solution. $\lambda_1 = 1$ is stationary and satisfies $\lambda_1 \geq 0$.

Now we study the relationship between the primal and dual problems.

Thm 12.10 The function g in the dual problem is concave and its domain D is convex.

Proof: For any $\lambda^0, \lambda^1 \in \mathbb{R}^m$, any $x \in \mathbb{R}^n$, any $\alpha \in [0, 1]$,

$$l(x, (\alpha\lambda^0 + (1-\alpha)\lambda^1)) = \alpha l(x, \lambda^0) + (1-\alpha)l(x, \lambda^1).$$

Taking the \inf_x of both sides and noticing that the infimum of a sum is greater than or equal to the sum of infimums; we get

$$g((\alpha\lambda^0 + (1-\alpha)\lambda^1)) \geq \alpha g(\lambda^0) + (1-\alpha)g(\lambda^1)$$

$\therefore g$ is concave. If both λ^0 and λ^1 belongs to D

$$\text{then } g((\alpha\lambda^0 + (1-\alpha)\lambda^1)) \geq \alpha g(\lambda^0) + (1-\alpha)g(\lambda^1) \geq -\infty$$

$\therefore (\alpha\lambda^0 + (1-\alpha)\lambda^1) \in D \Rightarrow D$ is convex. \square

Thm 12.11 weak Duality $(\bar{x} \in S)$

For any \bar{x} feasible for the primal problem and any $\bar{\lambda}$ feasible for the dual problem ($\bar{\lambda} \geq 0$), we have

$$g(\bar{\lambda}) \leq f(\bar{x}).$$

$$\begin{aligned}
 \text{Proof: } q(\bar{x}) &= \inf_x f(x) - \bar{\lambda}^T c(x) \quad \xrightarrow{\text{by definition}} \\
 &\leq f(\bar{x}) - \bar{\lambda}^T c(\bar{x}) \quad \xrightarrow{\text{of infimum}} \\
 &\leq f(\bar{x}) \quad \xrightarrow{\text{since } \bar{\lambda} \geq 0, c(\bar{x}) \geq 0} \quad \square
 \end{aligned}$$

* Fact: Clearly $q(\lambda^*) \leq f(x^*)$ if x^* is a local solution to the primal problem. The optimum objective value of the dual problem gives a lower bound on the optimal objective value for the primal problem.

* For the primal problem $\min_{x \in \mathbb{R}^n} f(x)$ s.t. $c(x) \geq 0$,

the KKT conditions become

$$\begin{aligned}
 \nabla f(\bar{x}) - \nabla c(\bar{x}) \bar{\lambda} &= 0 \\
 c(\bar{x}) &\geq 0 \\
 \bar{\lambda} &\geq 0 \\
 \bar{\lambda}_i c_i(\bar{x}) &= 0 \quad i=1, 2, \dots, m
 \end{aligned}$$

where $\nabla c(x) = [\nabla c_1(x) \ \nabla c_2(x) \ \dots \ \nabla c_m(x)]$.

Next we show that the optimal Lagrange multipliers for the primal problem are solutions of the dual problem under some conditions.

Thm 12.12 Suppose that \bar{x} is a solution of the primal problem and that f and $-c_i$, $i=1, \dots, m$ are convex functions on \mathbb{R}^n that are differentiable at \bar{x} . Then any $\bar{\lambda}$, which $(\bar{x}, \bar{\lambda})$ satisfies the KKT conditions is a solution of the dual problem.

Proof: Suppose that $(\bar{x}, \bar{\lambda})$ satisfies the KKT cond.
 Then we have $\bar{\lambda} \geq 0 \Rightarrow L(\cdot, \bar{\lambda})$ is convex & differentiable.
 Hence $L(x, \bar{\lambda}) \geq L(\bar{x}, \bar{\lambda}) + \nabla_x L(\bar{x}, \bar{\lambda})^T(x - \bar{x}) = L(\bar{x}, \bar{\lambda})$
 where inequality is due to L being convex and equality
 is due to KKT a , $\nabla_x L(\bar{x}, \bar{\lambda}) = 0$. Therefore

$$q(\bar{\lambda}) = \inf_x L(x, \bar{\lambda}) = L(\bar{x}, \bar{\lambda}) = f(\bar{x}) - \bar{\lambda}^T c(\bar{x}) = f(\bar{x}).$$

The last step is due to $\bar{\lambda}_i c_i(\bar{x}) = 0 \forall i$. From Thm 12.11, we have $q(\bar{\lambda}) \leq f(\bar{x}) \forall \bar{\lambda} \geq 0$, clearly from $q(\bar{\lambda}) = f(\bar{x})$ we conclude that $\bar{\lambda}$ is a solution of the dual problem. \square

Fact: From Thm 12.1, if the functions are cont-diff. and LICQ (or other const. qualifications) hold at \bar{x} , then an optimal Lagrange multiplier $\bar{\lambda}$ exists.

A partial converse of Thm 12.12 which shows that the solutions of the dual problem can sometimes be used to derive solutions to the primal problem.

Thm 12.13 Suppose that f and $-c_i$, $i=1,\dots,n$ are convex and cont. diff. on \mathbb{R}^n . Suppose that \bar{x} is a solution of the primal problem at which LICQ holds. Suppose that $\hat{\lambda}$ solves the dual problem and that the infimum $\inf_x L(x, \hat{\lambda})$ is attained at \hat{x} . Further, assume that $L(\cdot, \hat{\lambda})$ is a strictly convex function. Then $\hat{x} = \bar{x}$; that is \hat{x} is the unique solution of the primal problem and $f(\bar{x}) = L(\bar{x}, \hat{\lambda})$.

Proof: Assume for contradiction that $\bar{x} \neq \hat{x}$. From Thm 12.12 and the LICQ assumption, $\exists \tilde{\lambda}$ satisfying the KKT cond. Hence, from Thm 12.12, $\tilde{\lambda}$ solves the dual problem, so

$$L(\bar{x}, \tilde{\lambda}) = g(\tilde{\lambda}) = g(\hat{\lambda}) = L(\hat{x}, \hat{\lambda}).$$

Since $\hat{x} = \operatorname{arg\,min}_x L(x, \hat{\lambda})$, from Thm 2.2 $\nabla_x L(\hat{x}, \hat{\lambda}) = 0$. From the strict convexity of $L(\cdot, \hat{\lambda})$, we have

$$L(\bar{x}, \hat{\lambda}) - L(\bar{x}, \bar{\lambda}) > \nabla_x L(\bar{x}, \hat{\lambda})^T (\bar{x} - \hat{x}) = 0$$

$$\Rightarrow L(\bar{x}, \hat{\lambda}) > L(\bar{x}, \bar{\lambda}) = L(\bar{x}, \bar{\lambda})$$

$$\text{so } -\hat{\lambda}^T c(\bar{x}) > -\bar{\lambda}^T c(\bar{x}) = 0$$

where the last equality is due to $\bar{\lambda}_i c_i(\bar{x}) = 0$ for all i .

Since $\hat{\lambda} \geq 0$ and $c(\bar{x}) \geq 0$ and from the last result above $\hat{\lambda}^T c(\bar{x}) < 0$, there is a contradiction.

Hence our initial assumption that $\bar{x} \neq \hat{x}$ must be wrong. $\therefore \bar{x} = \hat{x}$. \square

Defn: The Wolfe dual for the primal problem

$\max_{x \in \mathbb{R}^n} f(x)$, ~~it~~ is defined as

$$\max_{x, \lambda} L(x, \lambda) \quad \text{s.t. } \nabla_x L(x, \lambda) = 0 \\ \lambda \geq 0$$

The Wolfe dual is convenient for computations.

Thm 12.14 Suppose that f and $-c_i$, $i=1, \dots, m$ are convex and cont. diff. on \mathbb{R}^n . Suppose that $(\bar{x}, \bar{\lambda})$ is a solution pair of the primal problem at which LICQ holds. Then $(\bar{x}, \bar{\lambda})$ solves the Wolfe dual problem.

Proof: From the KKT cond. we have that

$$\nabla_x \mathcal{L}(\bar{x}, \bar{\lambda}) = 0, \bar{\lambda} \geq 0, \text{ and } \mathcal{L}(\bar{x}, \bar{\lambda}) = f(\bar{x});$$

i.e. for any pair (x, λ) that satisfies $\nabla_x \mathcal{L}(x, \lambda) = 0$ and $\lambda \geq 0$, we have

$$\mathcal{L}(\bar{x}, \bar{\lambda}) = f(\bar{x})$$

$$\text{KKT} \Leftrightarrow f \geq f(\bar{x}) - \lambda^T c(\bar{x})$$

$$\text{convexity} \Leftrightarrow f = \mathcal{L}(x, \lambda)$$

$$\geq \mathcal{L}(x, \lambda) + \nabla_x \mathcal{L}(x, \lambda)^T (\bar{x} - x)$$

$$= \mathcal{L}(x, \lambda)$$

□

Example Linear Programming

Consider $\min_x c^T x$ s.t. $Ax - b \geq 0$. The dual is

$$g(\lambda) = \inf_x [c^T x - \lambda^T (Ax - b)] = \inf_x [(c - A^T \lambda)^T x + b^T \lambda].$$

Clearly, $c - A^T \lambda \neq 0 \Rightarrow g(\lambda) = -\infty$ (by setting $x = -\alpha(c - A^T \lambda)$ with $\alpha > 0$ large).

If $c - A^T \lambda = 0$, then $g(\lambda) = b^T \lambda$ and the dual problem becomes $\max_{\lambda} b^T \lambda$ s.t. $\lambda \geq 0$

$$\therefore \max_{\lambda} b^T \lambda \quad \text{s.t. } \lambda \geq 0$$

$$\quad \quad \quad A^T \lambda = c$$

The Wolfe ~~b^T~~-dual is

$$\max_{\lambda} c^T x - \lambda^T (Ax - b) \quad \text{s.t. } A^T \lambda = c, \lambda \geq 0.$$

(gives rise to the penalty methods.)

Example Convex Quadratic Programming

Consider $\min \frac{1}{2} x^T G x + c^T x$ s.t. $Ax - b \geq 0$ where $G > 0$ is symmetric. The dual objective is

$$g(\lambda) = \inf_x \mathcal{L}(x, \lambda) = \inf_x \frac{1}{2} x^T G x + c^T x - \lambda^T (Ax - b).$$

Since $G > 0$ and $\mathcal{L}(x, \lambda)$ is strictly convex, there is a unique minimizer with $\nabla_x \mathcal{L}(x, \lambda) = 0 : Gx + c - A^T \lambda = 0$.

$$\Rightarrow g(\lambda) = -\frac{1}{2} (A^T \lambda - c)^T G^{-1} (A^T \lambda - c) + b^T \lambda$$

The Wolfe dual is given by

$$\max_{x, \lambda} \frac{1}{2} x^T G x + c^T x - \lambda^T (Ax - b)$$

$$\text{s.t. } Gx + c - A^T \lambda = 0$$

$$\lambda \geq 0$$

From the equality constraint, we have $(c - A^T \lambda)^T x = -x^T G x$. Then the Wolfe dual objective can be expressed as

$$\max_{x, \lambda} -\frac{1}{2} x^T G x + \lambda^T b \quad \text{s.t. } Gx + c - A^T \lambda = 0 \\ \lambda \geq 0.$$

\therefore The Wolfe objective is clearly concave.

12.5) Let $v: \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a smooth vector function and consider the unconstrained optimization problems of minimizing $f(x)$ where

$$f(x) = \|v(x)\|_\infty, \quad f(x) = \max_{i=1, \dots, m} v_i(x)$$

Reformulate these as smooth constrained optimization problems.

$$\min_x \|v(x)\|_\infty \equiv \min_x \max_i |v_i(x)|$$

$$\equiv \min_t t \text{ s.t. } t \geq v_i(x), \quad t \geq -v_i(x) \\ i = 1, \dots, m$$

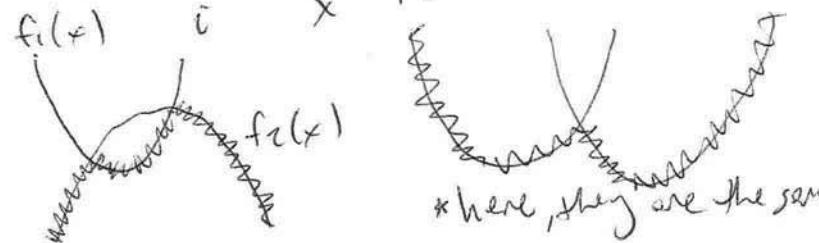
$$\min_x \max_i v_i(x) \equiv \min_t t \text{ s.t. } \cancel{t \geq v_i(x)}, \quad i = 1, \dots, m.$$

12.6) Can you perform a smooth reformulation of the problem when $f(x) = \min_{i=1, \dots, m} f_i(x)$?

Consider $\min_x \min_i f_i(x)$. Notice that this is not

the same as $\min_{f_i(x)} \min_i f_i(x)$!

Example



Notice that $\min_i f_i(x)$ could take $-\infty$ as the value for some $x \in \mathbb{R}^n$. Consequently, we cannot always find a smooth constrained formulation.

12.16) Solve $\min_x x_1 + x_2$ s.t. $x_1^2 + x_2^2 = 1$.

From the constraint, we have $x_2 = \pm \sqrt{1 - x_1^2}$

* For $x_2 = +\sqrt{1 - x_1^2}$, $f(x) = x_1 + \sqrt{1 - x_1^2} = f(x_1)$

$$\frac{d}{dx_1} f(x_1) = 1 + \frac{1}{2}(1 - x_1^2)^{-1/2}(-2x_1) = 0$$

$$\Leftrightarrow 1 - \frac{x_1}{(1 - x_1^2)^{1/2}} = 0 \Leftrightarrow x_1 = (1 - x_1^2)^{1/2}$$

$$x_1^2 = 1 - x_1^2 \Leftrightarrow 2x_1^2 = 1 \Leftrightarrow x_1^2 = \frac{1}{2} \Leftrightarrow x_1 = \pm \frac{1}{\sqrt{2}}$$

The solution candidate is $x_1 = +\frac{1}{\sqrt{2}}$ (since $\frac{1}{\sqrt{2}} = (1 - (\frac{1}{\sqrt{2}})^2)^{1/2}$).

Then $f(x_1) = \frac{1}{\sqrt{2}} + \sqrt{1 - \frac{1}{2}} = \frac{1}{\sqrt{2}} + \sqrt{\frac{1}{2}} = \frac{2}{\sqrt{2}} = \sqrt{2}$.

* For $x_2 = -\sqrt{1 - x_1^2}$, $f(x_1) = x_1 - \sqrt{1 - x_1^2}$.

$$\frac{d}{dx_1} f(x_1) = 1 + \frac{1}{2}(1 - x_1^2)^{-1/2}(2x_1) = 0$$

$$\Leftrightarrow 1 + \frac{x_1}{(1 - x_1^2)^{1/2}} = 0 \Leftrightarrow x_1 = -\sqrt{1 - x_1^2}$$

$$x_1^2 = 1 - x_1^2 \Leftrightarrow 2x_1^2 = 1 \Leftrightarrow x_1^2 = \frac{1}{2} \Leftrightarrow x_1 = \mp \frac{1}{\sqrt{2}}$$

The solution candidate is $x_1 = -\frac{1}{\sqrt{2}}$ ($-\frac{1}{\sqrt{2}} = -\sqrt{1 - (-\frac{1}{\sqrt{2}})^2}$).

Then $f(x_1) = -\frac{1}{\sqrt{2}} - \sqrt{1 - (-\frac{1}{\sqrt{2}})^2} = -\frac{1}{\sqrt{2}} - \sqrt{\frac{1}{2}} = -\sqrt{2}$.

Comparing the candidate $(\bar{x}, f(\bar{x}))$ pairs: $(x_1 = \frac{1}{\sqrt{2}}, f(x) = \sqrt{2})$

we select $(x_1 = \frac{1}{\sqrt{2}}, x_2 = \frac{1}{\sqrt{2}})$ as the solution.

Chapter 13: Linear Programming & The Simplex Method

The objective and constraint functions are linear:

$$\min_x c^T x \quad \text{s.t.} \quad \begin{matrix} \cancel{Ax=b} \\ \cancel{x \geq 0} \end{matrix}$$

Here, $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$ and any linear program can be transformed into this form.

Example $\min c^T x$ s.t. $Ax \leq b$ is the same as $\min c^T x$ s.t. $Ax + z = b$, $z \geq 0$ where z is the vector of slack variables. This is still not in the standard form.

$$\text{Let } x = x^+ - x^- \text{ where } x^+ \stackrel{\Delta}{=} \max(x, 0) \geq 0 \\ x^- \stackrel{\Delta}{=} \max(-x, 0) \geq 0$$

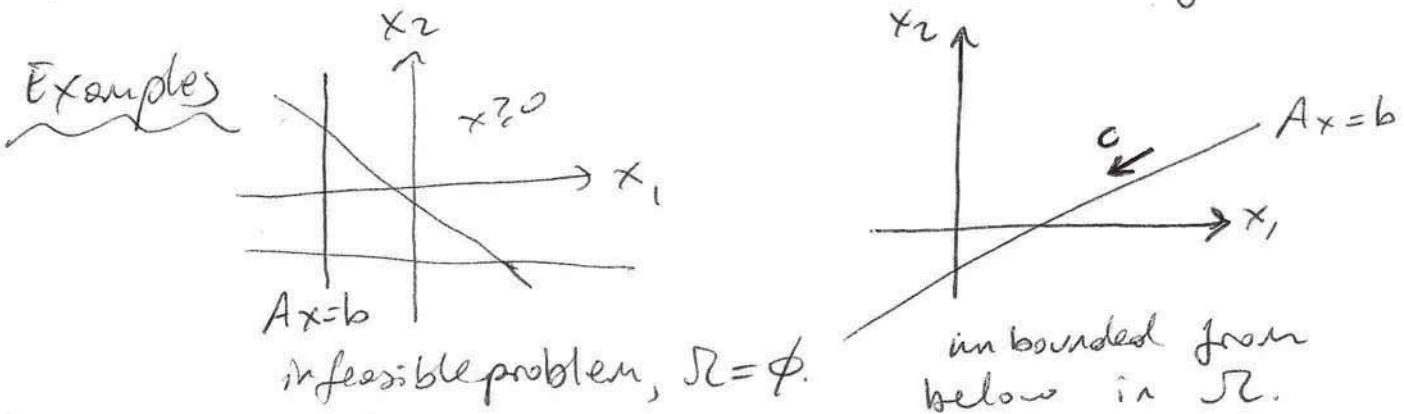
$$\text{Then } c^T x = [c^T \quad -c^T] \begin{bmatrix} x^+ \\ x^- \end{bmatrix}$$

$$\{A \quad -A \quad I\} \begin{bmatrix} x^+ \\ x^- \end{bmatrix} = b, \begin{bmatrix} x^+ \\ x^- \end{bmatrix} \geq 0, \text{ which}$$

yields the standard form.

- * $x \leq u \Leftrightarrow x + w = u, w \geq 0$ { where w and y
- $Ax \geq b \Leftrightarrow Ax - y = b, y \geq 0$ { are slack variables.

Defn: The linear program is infeasible iff the feasible set is empty. The linear programming problem is unbounded if the objective function is unbounded below on the feasible region.



Optimality and Duality

We partition the Lagrange multipliers as λ and s for equality and inequality constraints, respectively.

$$\mathcal{L}(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x$$

From the KKT conditions (Thm 12.1) we have that for a solution (x^*, λ^*, s^*) :

$$A^T \lambda^* + s^* = c \quad \text{Here } \lambda \in \mathbb{R}^m$$

$$Ax^* = b \quad s \in \mathbb{R}^n.$$

$$x^* \geq 0$$

$$s^* \geq 0$$

$$(x^* \geq 0) \quad x_i^* s_i^* = 0 \quad i=1 \dots n$$

since $x^* \geq 0, s^* \geq 0$.

equivalently

from the three equalities:

$$c^T x^* = (A^T \lambda^* + s^*)^T x^* = (A x^*)^T \lambda^* = b^T \lambda^*$$

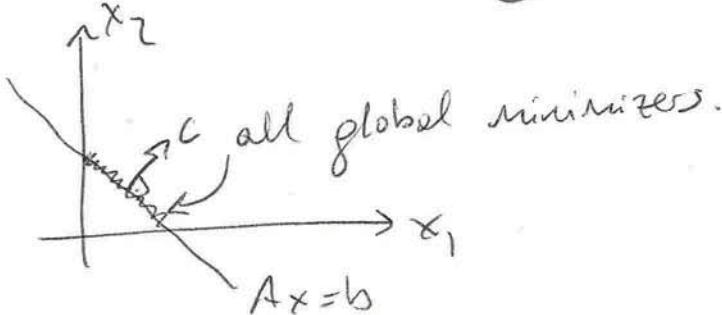
We will see that $b^T \lambda$ is the ^{dual} objective.

Let $\bar{x} \in \mathbb{R}$ so $A\bar{x} = b$, $\bar{x} \geq 0$. Then

$$c^T \bar{x} = (A \lambda^* + s^*)^T \bar{x} = b^T \lambda^* + \bar{x}^T s^* \geq b^T \lambda^* = c^T x^*$$

$\therefore x^*$ is a feasible solution with the smallest possible objective value (not necessarily a unique solution).

Example



Furthermore, \bar{x} is optimal iff $\bar{x}^T s^* = 0$; i.e. for $s_i^* > 0$, we must have $\bar{x}_i = 0$ for all $\bar{x} \in \mathbb{R}$ with minimal objective value.

The Dual Problem: The dual problem is defined as

$$\max_{\lambda} b^T \lambda \quad \text{s.t. } A^T \lambda \leq c$$

for the primal $\min_x c^T x$ s.t. $Ax = b$, $x \geq 0$.

Introduce dual slack variables s :

$$\max_{\lambda} b^T \lambda \quad \text{s.t. } A^T \lambda + s = c, \quad s \geq 0.$$

The variables (λ, s) are jointly referred to as the dual variables. Convert the dual problem into standard form: $\min -b^T \lambda$ s.t. $c - A^T \lambda \geq 0$ and obtain the KKT conditions for $\bar{L}(\lambda, x) = -b^T \lambda - x^T(c - A^T \lambda)$

$$\begin{aligned} \therefore \quad & Ax = b \\ & A^T \lambda \leq c \\ & x \geq 0 \end{aligned}$$

$$x_i(c - A^T \lambda)_i = 0 \quad i=1, 2, \dots, n \quad (x_i s_i = 0)$$

$$\begin{aligned} \text{Let } s_i &\stackrel{\Delta}{=} (c - A^T \lambda)_i \\ &\Downarrow s = c - A^T \lambda. \end{aligned}$$

* The KKT conditions for the primal and dual problems are identical. The parameters x and Lagrange multipliers λ have changed roles in the two problems, but their solutions are identical.

* You can see that the dual of the dual problem gives the primal problem. (Prove this as exercise).

Thm 13.1 (Strong Duality)

Primal: $\max c^T x$ s.t. $Ax = b, x \geq 0$

Dual: $\max b^T \lambda$ s.t. $A^T \lambda \leq c$

- (i) If either the primal or the dual has a (finite) solution, then so does the other, and the objective values are equal.
- (ii) If either the primal or the dual problem is unbounded, then the other is infeasible.

Proof: See page 361.

Sensitivity Analysis

How does the optimal objective change if b is perturbed?

Suppose a perturbation Δb causes perturbations Δs and Δx . From complementarity:

$$0 = x^T s = x^T \Delta s = (\Delta x)^T s = (\Delta x)^T \Delta s.$$

From Thm 13.1, the primal and dual objectives are equal:

$$c^T x = b^T \lambda, \quad c^T(x + \Delta x) = (b + \Delta b)^T(\lambda + \Delta \lambda).$$

Since perturbed solutions must be feasible:

$$A(x + \Delta x) = b + \Delta b, \quad A^T \Delta \lambda = -\Delta s$$

∴ with some algebra (page 362 in the book), we get

$$c^T \Delta x = (\Delta b)^T \lambda$$

Consequently $\Delta b = \epsilon e_j \Rightarrow c^T \Delta x = \epsilon \lambda_j$, similar to the sensitivity results we obtained before.

Geometry of the Feasible Set

From this point on, we assume that A has full row rank. This means, the equality constraints does not contain redundancies or conflicts.

Defn A point x is a basic feasible point iff there exists a subset B of $\{1, \dots, n\}$ such that

- (i) B contains exactly m indices ($|B|=m$)
- (ii) $i \notin B \Rightarrow x_i = 0$ (i.e. $x_i \geq 0$ is inactive only if $i \notin B$)
- (iii) The $n \times m$ matrix B defined by $B \triangleq [A_{:,i}]_{i \in B}$, where $A_{:,i}$ is the i^{th} column of A , is nonsingular.

Defn: A set B satisfying the properties above is called a basis for the primal linear problem. The corresponding matrix B is called the basis matrix.

Theorem 13.2 (Fundamental Theorem of Linear Programming)

- (i) If the primal problem has a nonempty feasible region, then there is at least one basic feasible point.
- (ii) If the primal linear prog. problem has solutions, then at least one such solution is a basic optimal point.
- (iii) If the primal lin. prog. problem is feasible and bounded, then it has an optimal solution.

Proof: See pages 363-364 in the book.

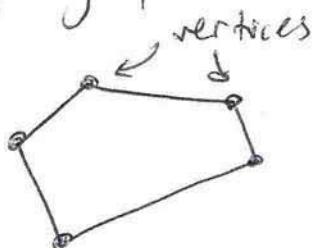
Vertices of the Feasible Polytope

The feasible set defined by the linear constraints is a polytope. The vertices of this polytope are the points on the

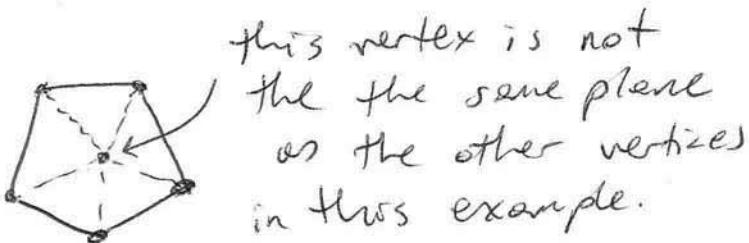
boundary which do not lie ~~on~~ a straight line segment.

Example

Polytope in 2D



Polytope attempt in 3D.

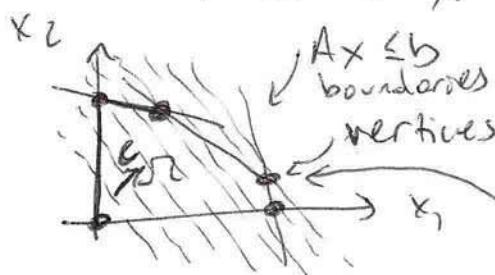


Theorem 13.3 All basic feasible points for the primal lin. progr. problem are vertices of the feasible polytope $\{x \mid Ax = b, x \geq 0\}$ and vice versa.

Proof See pages 365-366 in the book.

Example Consider the problem (nonstandard)

$$\max c^T x \text{ s.t. } A^T x \leq b, x \geq 0 \quad (\text{all inequalities in 2D.})$$



vertices of the polytope S_2 are the basic feasible points. optimal point (unique in this case).

Defn (Degeneracy)

A basis B is said to be degenerate if $x_i = 0$ for some $i \in B$, where x is the basic feasible solution corresponding to B . A linear program is said to be degenerate if it has at least one degenerate basis.

The Simplex Method

All iterates of the simplex method are basic feasible points. Therefore, the algorithm traverses the vertices of the feasible polytope. Most steps consist of changing the basis B in one component (choose a different vertex) and move to the neighboring vertex on the polytope.

The objective $c^T x$ does not decrease at all steps.

The problem is, which index in B should be replaced and how?

Let $N = \{1, \dots, n\} \setminus B$ be the complement of B and call it the nonbasic index set. Let $N = \{A_{:,i}\}_{i \in N}$ be the corresponding matrix of columns of A . Define

$$x_B = \{x_i\}_{i \in B} \quad x_N = \{x_i\}_{i \in N}$$

$$s_B = \{s_i\}_{i \in B} \quad s_N = \{s_i\}_{i \in N}$$

$$c_B = \{c_i\}_{i \in B} \quad c_N = \{c_i\}_{i \in N}$$

From the KKT conditions, we have

$$Ax = Bx_B + Nx_N = b$$

The primal variable x for this basis B is then

$$x_B = B^{-1}b, \quad x_N = 0.$$

By construction B is nonsingular and $x_B \geq 0$, so this choice satisfies the equality constraints and the nonnegativity constraint for x ($Ax=b$, $x \geq 0$).

To satisfy the complementarity condition, we must set $s_B = 0$. Then $A^T \lambda + s = c$ yields $\begin{bmatrix} B^T \\ N^T \end{bmatrix} \lambda + \begin{bmatrix} s_B \\ s_N \end{bmatrix} = \begin{bmatrix} c_B \\ c_N \end{bmatrix}$
 $\Rightarrow B^T \lambda = c_B$ and $N^T \lambda + s_N = c_N$.

From the first equation, $\lambda = B^{-T} c_B$. From the second equation, $s_N = c_N - N^T \lambda = c_N - (B^{-1} N)^T c_B$ ~~satisfy~~.

The basic components $s_B \geq 0$ since we set $s_B = 0$. If the calculations above yield $s_N \geq 0$, then we have found an optimal solution, so we can stop.

If there are entries in N with $s_{Ni} < 0$, $i \in N$, then we select an entering and a leaving index for B .

- * For this, we basically select x_q for which $s_q < 0$.
 - * We allow x_q to increase from zero while keeping all other components of x_N at zero. While trying to satisfy $Ax=b$, we find the entry of x_B that becomes zero (say x_p).
 - * Then we replace in B index p with q .
- For details see pages 368-370 in the book.

Theorem 13.4 Provided that the linear program is nondegenerate and bounded, the simplex method terminates at a basic optimal point.

Proof: See page 370 in the book.

Procedure 13.1 (One step of Simplex)

Given $B, N, x_B = B^{-1}b, x_N = 0$

Solve $B^T \lambda = c_B$ for λ .

Compute $s_N = c_N - N^T \lambda$

if $s_N \geq 0$, STOP (optimal found).

Select $q \in N$ with $s_q < 0$ as entering index

Solve $Bd = A_q$ for d

if $d \leq 0$, STOP (problem is unbounded)

Calculate $x_q^+ = \min_{i/d_i > 0} (x_B)_i/d_i$, set p to best i .

Update $x_B^+ = x_B - d x_q^+$, $x_N^+ = (0, \dots, 0, x_q^+, 0, \dots, 0)^T$.

Change B by adding q and removing p .

For practical issues including the initialization, entering index selection and handling of degenerate steps, please see the pages 371-382 in the book.

The Dual Simplex Method

The simplex method above starts with a feasible x and a corresponding dual iterate (λ, s) where $s_B = 0$ but s_N is not necessarily nonnegative. It exchanges columns between B and N until $s_N \geq 0$.

The dual simplex method solves the dual problem by starting with a point (λ, s) which is feasible for the dual problem (so $s_B = 0$ and $s_N \geq 0$) and a corresponding feasible point x for which $x_N = 0$ but x_B is not necessarily nonnegative. Interchanging columns between B and N it reaches a feasible primal point x . You can find the details of a single step of this algorithm on pages 383-385 in the book.

Presolving

Some preprocessing steps can be carried out to reduce the size of the user-defined linear programming problem. Some straightforward preprocessing steps are as follows:-

Consider $\min c^T x$ s.t. $Ax=b$, $l \leq x \leq u$

where l and u are lower and upper bound vectors possibly with some $-\infty$ and $+\infty$ in their entries.

Row singletor: An equality constraint contains only one variable.

e.g. the k th constraint is $a_{kj} x_j = b_k$ and $a_{ki}=0$, i ≠ j.

Then clearly $x_j = b_k/a_{kj}$ and if $x_j \notin [l, u]$, then the problem is infeasible.

Column singletor: The variable x_j appears in only one of the equality constraints. If $l_j = -\infty$ and $u_j = +\infty$, then clearly x_j is free (unconstrained by the inequalities). Then

we can set $x_j = \frac{b_k - \sum_{p \neq j} A_{kp} x_p}{A_{kj}}$ where constraint

k satisfies $A_{kj} \neq 0$ and $A_{lk} = 0$ for all $l \neq k$.

The cost vector c must be updated accordingly:

$$c_p \leftarrow c_p - c_j A_{kp}/A_{kj} \text{ for all } p \neq j.$$

Since x_j is a free variable, there is no dual slack variable for it and the j th dual constraint becomes

$$\sum_{l=1}^m A_{lj} \lambda_l = c_j \Rightarrow A_{kj} \lambda_k = c_j \Rightarrow \lambda_k = \frac{c_j}{A_{kj}}.$$

* All-zero rows: If $A_{kj} = 0$ & $b_k = 0$, then the k th constraint can be removed and λ_k can take an arbitrary value.

All-zero columns: If $A_{kj} = 0 \forall j$, then x_j does not appear in any equality constraints so we can check c and set x_j as follows:

$$c_j \leq 0 \Rightarrow x_j = u_j; \quad c_j > 0 \Rightarrow x_j = l_j$$

$u_j \geq \infty \quad l_j < -\infty$

If the respective bounds are $\pm\infty$, then the problem is unbounded.

Forcing Constraints: Consider $5x_1 - x_4 + 2x_5 = 10$

$$0 \leq x_1 \leq 1, \quad -1 \leq x_4 \leq 5, \quad 0 \leq x_5 \leq 2$$

$x_1 = 1, x_4 = -1, x_5 = 2$ is the only selection that satisfies all constraints, so these parameters can be set and the constraints can be dropped.

Dominated Constraints: Consider $2x_2 + x_6 - 3x_7 = 8$

$$-10 \leq x_2 \leq 10, \quad 0 \leq x_6 \leq 1, \quad 0 \leq x_7 \leq 2$$

From this, we see that $x_2 = 4 - \frac{x_6}{2} + \frac{3x_7}{2} \leq 4 - 0 + \frac{3}{2} \cdot 2 = 7$
 $(x_6=0) \quad (x_7=2)$

Using the opposite bounds: $x_2 \geq 7/2$ so $\frac{7}{2} \leq x_2 \leq 7$ and the bounds $-10 \leq x_2 \leq 10$ are redundant.

Recursive application of preprocessing: Consider $3x_2 = 6, x_2 + 4x_5 = 10$.

Once we set $x_2 = 2$, then the second equality yields $x_5 = 2$. So the preprocessing steps can be applied recursively to discover simplifications that are not originally obvious.

(3.2) Verify that the dual of the dual linear program is the primal linear program.

The dual problem is $\min_{\lambda} -b^T \lambda$ s.t. $c - A^T \lambda \geq 0$.

The dual of the dual objective is

$$\begin{aligned} q(x) &= \inf_{\lambda} f(\lambda, x) = \inf_{\lambda} -b^T \lambda - x^T(c - A^T \lambda) \\ &= \inf_{\lambda} (Ax - b)^T \lambda - c^T x \quad \text{if finite.} \end{aligned}$$

The $q(x)$ value is finite $\Leftrightarrow Ax - b = 0$. So the dual of the dual is

$$\max_x -c^T x \quad \text{s.t. } Ax = b, x \geq 0$$

$$\Leftrightarrow \min_x c^T x \quad \text{s.t. } Ax = b, x \geq 0.$$

Chapter 14: Linear Programming - Interior-Point Methods

The simplex method explicitly estimates the active and inactive constraints at each iteration and works its way around the feasible polytope boundary.

Interior-point methods follow trajectories that are satisfying the inequalities strictly; hence remaining at the interior of the polytope.

Primal-Dual Methods

These have been shown to be practically efficient and strong competitors to the simplex method on large problems. We consider the standard linear programming problem

$$\min c^T x \quad \text{s.t. } Ax = b, \quad x \geq 0.$$

Here $A \in \mathbb{R}^{m \times n}$, $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$. A has full row rank (i.e. $m \leq n$ necessarily). The dual problem is

$$\max b^T \lambda \quad \text{s.t. } A^T \lambda + s = c, \quad s \geq 0$$

where $\lambda \in \mathbb{R}^m$, $s \in \mathbb{R}^n$. The KKT conditions of both primal and dual problems are identical as we have seen in Chapter 13.

$$\begin{array}{l} \text{KKT conditions} \\ A^T \lambda + s = c \\ Ax = b \\ s_i x_i = 0 \quad \forall i \\ x \geq 0, s \geq 0 \end{array}$$

Primal-dual methods find solutions (x^*, λ^*, s^*) by applying variants of Newton's method to the three inequalities and modifying

search directions and step lengths so that the inequalities are strictly satisfied at every iteration.

We restate the KKT conditions as follows:

$$F(x, \lambda, s) = \begin{bmatrix} A^T \lambda + s - c \\ Ax - b \\ XSe \\ x \geq 0, s \geq 0 \end{bmatrix} = 0$$

here

$$X \triangleq \text{diag}(x_1, \dots, x_n)$$

$$S \triangleq \text{diag}(s_1, \dots, s_n)$$

$$e \triangleq (1, \dots, 1)^T$$

The two basic components are:

- (i) a procedure for determining the step
- (ii) a measure of the desirability of each point

The average value of $x_i s_i$, $i = 1, \dots, n$ is an important component of (ii) above. Notice that $x \geq 0, s \geq 0$.

The desirability measure is $\mu \stackrel{\Delta}{=} \frac{1}{n} \sum_{i=1}^n x_i s_i = \frac{x^T s}{n}$.

Let $J(x, \lambda, s)$ be the Jacobian of $F(x, \lambda, s)$.

Then the Newton method needs to solve

$$J(x, \lambda, s) \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = -F(x, \lambda, s)$$

Let $r_b = Ax - b$, $r_c = A^T \lambda + s - c$. Then

The Newton equations are

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -XSe \end{bmatrix}$$

A full step along the Newton direction might violate the bounds $x \geq 0$, $s \geq 0$, so a line search along the Newton direction can be performed; $\alpha \in (0, 1]$

$$(x, \lambda, s) \leftarrow (x, \lambda, s) + \alpha (\Delta x, \Delta \lambda, \Delta s)$$

In practice, typically $\alpha \ll 1$, otherwise $x \geq 0, s \geq 0$ are violated; so the Newton step with its affine scaling direction does not allow much progress.

- * Instead of aiming for a point where $\mu = 0$, we can seek a direction that is less aggressive but still tries to reduce μ to a low value.
- * If the current duality measure is μ , and $\sigma \in [0, 1]$ is the desired reduction factor, we take a Newton step towards a point for which $x_i s_i = \sigma \mu$.

The modified step equation is then

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -x^T s + \sigma \mu \end{bmatrix}$$

Here, σ is called the centering parameter.

Framework 14.1 (Primal-Dual Path Following)

Given (x^0, λ^0, s^0) with $(x^0, s^0) > 0$

for $k = 0, 1, 2, \dots$

* Choose $\alpha_k \in \{0, 1\}$ and solve

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -x^k s^k + \sigma_k \mu_k \end{bmatrix}$$

where $\mu_k = (x^k)^T s^k / n$.

* set $(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k, \lambda^k, s^k) + \alpha_k (\Delta x^k, \Delta \lambda^k, \Delta s^k)$

choosing α_k such that $(x^{k+1}, s^{k+1}) > 0$.

end (for)

The Central Path The primal-dual feasible set \mathcal{F} and strictly feasible set \mathcal{F}^0 are defined as follows.

$$\mathcal{F} = \{(x, \lambda, s) \mid Ax = b, A^T \lambda + s = c, (x, s) \geq 0\}$$

$$\mathcal{F}^0 = \{(x, \lambda, s) \mid Ax = b, A^T \lambda + s = c, (x, s) > 0\}.$$

Defn. - The central path C is an arc of strictly feasible points that is parametrized by $\tau > 0$ and $(x_\tau, \lambda_\tau, s_\tau) \in C$ satisfies the conditions

$$\begin{array}{l} A^T \lambda + s = c \\ Ax = b \\ x_i s_i = \tau \quad i=1, \dots, n \\ (x, s) > 0 \end{array}$$

instead of requiring
 $x_i s_i = 0$, we require
 $x_i s_i = \tau$.

The central path is $C = \{(x_\tau, \lambda_\tau, s_\tau) \mid \tau > 0\}$.

Fact: $(x_\tau, \lambda_\tau, s_\tau)$ is defined uniquely for $\tau > 0$ iff \mathcal{F}° is nonempty.

We introduce the log-barrier approach for the nonnegativity constraints. With barrier parameter $\tau > 0$, we get

$$\max c^T x - \tau \sum_{i=1}^n \ln x_i \quad \text{s.t. } Ax = b.$$

The KKT conditions for this problem are

$$c_i - \frac{\tau}{x_i} - A_{i\cdot}^T \lambda = 0 \quad i=1, \dots, n \quad \text{and } Ax = b.$$

This problem has a strictly convex objective so these conditions are both necessary and sufficient for optimality.

* This is called the log-barrier formulation of the linear program.

If we let $s_i \stackrel{\Delta}{=} \tau/x_i$, $i=1,\dots,n$ then we recover the conditions for C the central path:

$$A^T \lambda + s = c, \quad Ax = b, \quad x_i s_i = \tau \quad \forall i, \quad (x, s) \geq 0.$$

Most primal-dual algorithms take Newton steps toward points on C for which $\tau > 0$. Since these steps are pointing towards the interior of the nonnegative orthant $(x, s) \geq 0$, one can take larger steps without violating the constraints.

Central Path Neighborhoods

Path following algorithms restrict the iterates to a neighborhood of the central path C and follow C to a solution of the linear program. This ensures that large steps can be taken while forcing the duality measure μ_k to 0 as $k \rightarrow \infty$. This ensures (x^k, λ^k, s^k) comes closer to satisfying the KKT conditions asymptotically.

We introduce two neighborhoods of C .

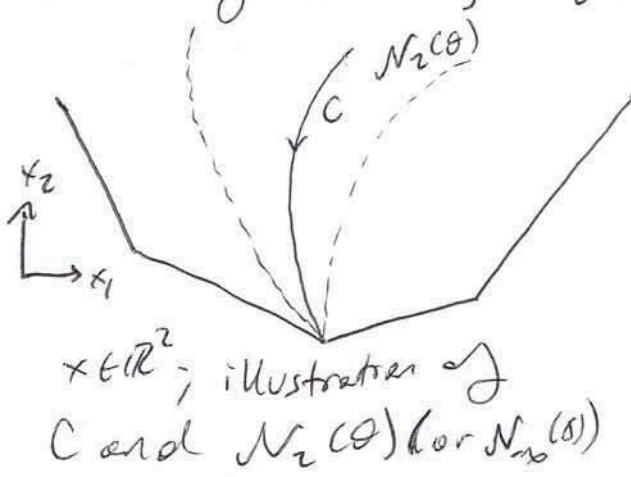
$$\mathcal{N}_2(\theta) = \{(x, \lambda, s) \in \mathcal{F}^0 \mid \|x - \text{Mell}_2\|_2 \leq \theta \mu\}, \quad \theta \in [0, 1]$$

$$\mathcal{N}_{-\infty}(\gamma) = \{(x, \lambda, s) \in \mathcal{F}^0 \mid x_i s_i \geq \gamma \mu \quad \forall i \in \{1, \dots, n\}\}, \quad \gamma \in [0, 1]$$

For instance one can set $\theta = 1/2$ and $\gamma = 10^{-3}$.

As γ approaches 0, $\mathcal{N}_{-\infty}(\gamma)$ approaches \mathcal{F}^0 (and \mathcal{F}).

Even if $\theta = 1$, $\mathcal{N}_2(\theta)$ is not close to \mathcal{F}^0 or \mathcal{F} .



Path following methods are similar to homotopy methods in principle. In homotopy methods, the neighborhood is tubular around the trajectory while here it is like a curved cone & narrows towards the solution.

The neighborhood narrows as $\mu \rightarrow 0$ since $(x, s) > 0$.

Notation: $(x^{k(\alpha)}, \lambda^{k(\alpha)}, s^{k(\alpha)}) \triangleq (x^k, \lambda^k, s^k) + \alpha(\Delta x^k, \Delta \lambda^k, \Delta s^k)$

$$\mu_k(\alpha) \triangleq x^{k(\alpha)}^T s^{k(\alpha)} / n$$

Algorithm 14.2 (Long-step Path-Following)

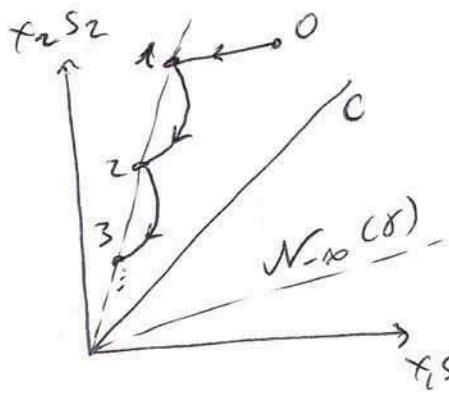
Given $\gamma, \sigma_{\min}, \sigma_{\max}$ with $\gamma \in (0, 1)$, $0 < \sigma_{\min} \leq \sigma_{\max} \leq 1$, $(x^0, \lambda^0, s^0) \in \mathcal{N}_{-\infty}(\gamma)$ for $k = 0, 1, 2, \dots$

Choose $\theta_k \in [\sigma_{\min}, \sigma_{\max}]$; Solve $\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ s^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_{x^k} \\ -r_{\lambda^k} \\ -X^k s^k e + \sigma_k \mu_k e \end{bmatrix}$.

Set α_k to largest value in $\{0, 1\}$ such that $(x^k, \lambda^k, s^k) + \alpha_k(\Delta x^k, \Delta \lambda^k, \Delta s^k) \in \mathcal{N}_{-\infty}(\gamma)$.

Set $(x^{k+1}, \lambda^{k+1}, s^{k+1})$ to $(x^k(\alpha_k), \lambda^k(\alpha_k), s^k(\alpha_k)) \in \mathcal{N}_{-\infty}(\gamma)$.

end (for).



This is an illustration of the process. A line search along $\alpha \in [0,1]$ yields curved trajectories in the $x_i^S_i$ space. The selected iterate remains on the boundary.

The parameter $\sigma_{\min} > 0$ ensures that each search direction first moves away from the boundary of $N_\infty(\epsilon)$.

Fact: For some tolerance $\epsilon > 0$, the algorithm needs $O(n \log \epsilon^{-1})$ iterations to reduce μ by a factor of ϵ ($\mu_k \leq \epsilon \mu_0$) where $k \sim O(n \log \epsilon^{-1})$.

Now we see a sequence of results that prove this result.

Lemma 14.1 Let u and v be any two vectors in \mathbb{R}^n with $u^T v \geq 0$. Then $\|uv\|_2 \leq 2^{-3/2} \|u+v\|_2$ where $U \triangleq \text{diag}(u, \dots, u_n)$, $V \triangleq \text{diag}(v, \dots, v_n)$.

Proof: See pages 402-403 in the book.

Notice that $UVe = \begin{bmatrix} u^T v & 0 \\ 0 & u^T v n \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} u^T v \\ u^T v \\ \vdots \\ u^T v \end{bmatrix}$ and $\|uv\|_2$ is $(u^T v_1)^2 + \dots + (u^T v_n)^2)^{1/2}$.

Now let $\Delta X = \text{diag}(\Delta x_1, \dots, \Delta x_n)$, $\Delta S = \text{diag}(\Delta s_1, \dots, \Delta s_n)$.
be the ~~iteration~~^{update} vectors metrisized at any iteration k .

Lemma 14.2 If $(x, \lambda, s) \in N_{\gamma, \delta}(x)$, then $\|\Delta x \Delta s\| \leq 2^{-3/2} \frac{1}{(1+\gamma)} \gamma \mu$. 300

Proof: See pages 403-404 in the book.

Theorem 14.3 Given the parameters γ , σ_{\min} , and σ_{\max} in

Alg. 14.2, $\exists \delta$ independent of n $\nexists \mu_{k+1} \leq (1 - \frac{\delta}{n}) \mu_k \forall k \geq 0$.

Proof: See pages 404-406 in the book. This is a constructive proof and we get:

$$\delta = 2^{3/2} \times \frac{1-\gamma}{1+\gamma} \min \left\{ \sigma_{\min}(1-\sigma_{\min}), \sigma_{\max}(1-\sigma_{\max}) \right\}.$$

Now the sought result. . .

Theorem 14.4 Given $\epsilon \in (0, 1)$ and $\gamma \in (0, 1)$, suppose the starting point in Alg. 14.2 satisfies $(x^0, \lambda^0, s^0) \in N_{\gamma, \delta}(x)$. Then $\exists K = O(n \log \frac{1}{\epsilon}) \nexists \mu_k \leq \epsilon M_0 \forall k \geq K$.

Proof: see page 406 in the book. The particular K

$$K = \frac{1}{\delta} \log \frac{1}{\epsilon} = \frac{n}{\delta} |\log \epsilon|.$$

Practical Primal-Dual Algorithms

We now consider practical issues in implementing this approach.

Corrector and Centering steps

Consider the Newton direction $(\Delta x, \Delta \lambda, \Delta s)$ given by

$$J \rightarrow \begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^{off} \\ \Delta \lambda^{off} \\ \Delta s^{off} \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -XSe \end{bmatrix}$$

- $x_i s_i$ in each entry.
 $r_b = Ax - b$
 $r_c = A^T \lambda + S - c$

Recall that this is based on the linearization of $F(x, \lambda, s)$ with its Jacobian J at the current point (x, λ, s) .

If we take a full step in this direction, we get

$$(x_i + \Delta x_i^{off})(s_i + \Delta s_i^{off}) = x_i s_i + x_i \Delta s_i^{off} + s_i \Delta x_i^{off} + \Delta x_i^{off} \Delta s_i^{off}$$

$$= \Delta x_i^{off} \Delta s_i^{off}$$

Note that this follows from the i th row of the 3rd block-row in the expression above.

$$S \Delta x^{off} + X \Delta s^{off} = -XSe \Rightarrow s_i \Delta x_i^{off} + x_i \Delta s_i^{off} = -x_i s_i.$$

While the linearized model aims to achieve $(x_i + \Delta x_i)(s_i + \Delta s_i) =$
 the nonlinear model yields $\Delta x_i \Delta s_i$ after the update

We can perform a correction step that attempts to fix this deviation from ideal.

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^{cor} \\ \Delta \lambda^{cor} \\ \Delta s^{cor} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\Delta x^{off} \Delta s^{off} e \end{bmatrix}.$$

The combined step $(\Delta x^{\text{off}}, \Delta \lambda^{\text{off}}, \Delta s^{\text{off}}) + (\Delta x^{\text{cor}}, \Delta \lambda^{\text{cor}}, \Delta s^{\text{cor}})$ does a better job of reducing the duality measure μ .

Selecting α_k : The maximum allowable step lengths along the affine scaling direction are

$$\alpha_{\text{aff}}^{\text{pri}} \triangleq \min\left(1, \min_{i: \Delta x_i^{\text{off},<0}} -\frac{x_i}{\Delta x_i^{\text{off}}}\right)$$

$$\alpha_{\text{aff}}^{\text{dual}} \triangleq \min\left(1, \min_{i: \Delta s_i^{\text{off},<0}} -\frac{s_i}{\Delta s_i^{\text{off}}}\right)$$

Defining $M_{\text{aff}} \triangleq (x + \alpha_{\text{aff}}^{\text{pri}} \Delta x^{\text{off}})^T (s + \alpha_{\text{aff}}^{\text{dual}} \Delta s^{\text{off}})/\mu$, a proportion that works well in practice is obtained:

$$\sigma = \left(\frac{M_{\text{aff}}}{\mu}\right)^3$$

For this σ value, we solve the prediction-centering step

from (predictor-
corrector
step)

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_a \\ -r_b \\ -x_S e - \Delta x^{\text{off}} \Delta s^{\text{off},e} + \sigma \mu e \end{bmatrix}$$

as discussed above in the context of correction.

Selecting Δ_k : Once the predictor-correction step direction is determined from the procedure above for a given σ , we need to select the step length.

Given (x^k, λ^k, s^k) with $(x^k, s^k) > 0$ and $(\Delta x^k, \Delta \lambda^k, \Delta s^k)$ we find the maximum allowed step lengths as

$$\alpha_{k,\max}^{\text{pri}} \stackrel{\Delta}{=} \min_{i: \Delta x_i^k < 0} \frac{-x_i^k}{\Delta x_i^k} ; \quad \alpha_{k,\max}^{\text{dual}} \stackrel{\Delta}{=} \min_{i: \Delta s_i^k < 0} -\frac{s_i^k}{\Delta s_i^k}$$

These are the largest values of α for which $x^k + \alpha \Delta x^k \geq 0$ and $s^k + \alpha \Delta s^k \geq 0$. In practice we use

$$\alpha_k^{\text{pri}} \in (0, \alpha_{k,\max}^{\text{pri}}) \quad \text{and} \quad \alpha_k^{\text{dual}} \in (0, \alpha_{k,\max}^{\text{dual}})$$

however, the step lengths must not be larger than 1, so

$$\alpha_k^{\text{pri}} = \min(1, \eta_k \alpha_{k,\max}^{\text{pri}}) ; \quad \alpha_k^{\text{dual}} = \min(1, \eta_k \alpha_{k,\max}^{\text{dual}})$$

where $\eta_k \in [0.9, 1.0]$ with $\eta_k \rightarrow 1$ as $k \rightarrow \infty$ to accelerate asymptotic convergence. Then

$$x^{k+1} = x^k + \alpha_k^{\text{pri}} \Delta x^k \quad \text{and} \quad (\lambda^{k+1}, s^{k+1}) = (\lambda^k, s^k) + \alpha_k^{\text{dual}} (\Delta \lambda^k, \Delta s^k).$$

Starting Point: A poor choice of (x^0, λ^0, s^0) may negatively affect convergence. We can find a reasonably good initial point by solving the following problems

$$\min_x \frac{1}{2} x^T x \quad \text{s.t.} \quad Ax = b$$

$$\min_{(\lambda, s)} \frac{1}{2} s^T s \quad \text{s.t.} \quad A^T \lambda + s = c$$

The solutions are $\tilde{x} = A^T (A A^T)^{-1} b$ and $\tilde{\lambda} = (A A^T)^{-1} A c$, $\tilde{s} = c - A^T \tilde{\lambda}$. These \tilde{x} and \tilde{s} do not satisfy the nonnegativity constraints in general.

Let $\delta_x = \max\left(-\frac{3}{2} \min_i \tilde{x}_i, 0\right)$; $\delta_s = \max\left(-\frac{3}{2} \min_i \tilde{s}_i, 0\right)$ and

$\hat{x} = \tilde{x} + \delta_x e$, $\hat{s} = \tilde{s} + \delta_s e$ so that $\hat{x} \geq 0$ and $\hat{s} \geq 0$.

To ensure that x^0 and s^0 are not too close to zero and not too dissimilar, introduce

$$\hat{\delta}_x = \frac{1}{2} \frac{\hat{x}^T \hat{s}}{e^T \hat{s}} \quad \text{and} \quad \hat{\delta}_s = \frac{1}{2} \frac{\hat{x}^T s}{e^T \hat{x}}$$

entries of

where $\hat{\delta}_x$ is a measure of the average size of \hat{x} relative to that of \hat{s} and $\hat{\delta}_s$ is similar. Then we let

$$x^0 = \hat{x} + \hat{\delta}_x e \quad \text{and} \quad \lambda^0 = \hat{\lambda}, \quad s^0 = \hat{s} + \hat{\delta}_s e.$$

The computational cost of this procedure is comparable to one step of the primal-dual algorithm.

Alg. 14.3 (Predictor-Corrector Algorithm)

Initialize (x^0, λ^0, s^0) as described above.

for $k=0, 1, 2, \dots$

Set $(x, \lambda, s) = (x^k, \lambda^k, s^k)$ and solve for $(\Delta x^{eff}, \Delta \lambda^{eff}, \Delta s^{eff})$.

Calculate α_k^{pri} , α_k^{dual} , μ_{eff} and set $\sigma = (\mu_{eff}/n)^3$.

Solve for $(\Delta x, \Delta \lambda, \Delta s)$ as the (*) predictor-corrector step above.

Calculate α_k^{pri} and α_k^{dual} .

Set $x^{k+1} = x^k + \alpha_k^{pri} \Delta x$, $(\lambda^{k+1}, s^{k+1}) = (\lambda^k, s^k) + \alpha_k^{dual} (\Delta \lambda, \Delta s)$.

end (for)

This algorithm is not guaranteed to converge but divergence incidents are rare. Safeguards can be added or rules can be implemented.

Solving the Linear Systems

Most of the computational effort in primal-dual methods is spent on solving linear systems of equations. The matrix is usually large and sparse since typically A is sparse.

Consider $\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -r_{xs} \end{bmatrix}, (x, s) > 0.$
 $\Leftrightarrow x > 0, s > 0.$

Multiplying the third egn with $-X^{-1}$ and adding to the first one:

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_c + X^{-1} r_{xs} \\ -r_b \end{bmatrix}$$

where $D \stackrel{\triangle}{=} S^{-1/2} X^{1/2}$. Then adding AD^2 times the first egn to the second one in this augmented system:

$$\left. \begin{aligned} AD^2 A^T \Delta \lambda &= -r_b - AXS^{-1} r_c + AS^{-1} r_{xs} \\ \Delta s &= -r_c - A^T \Delta \lambda \end{aligned} \right\} \begin{array}{l} \text{Normal} \\ \text{Equations} \end{array}$$

$$\Delta x = -S^{-1} r_{xs} - XS^{-1} \Delta s$$

where $\Delta s = -X^{-1} r_{xs} - X^{-1} S \Delta x$ from the first augmentation is also used. Cholesky decomposition of $AD^2 A^T$ can yield $\Delta \lambda$ and then Δs and Δx can be sequentially obtained.

- 14.2) Show that (i) $\theta_1 < \theta_2 \Rightarrow N_2(\theta_1) \subset N_2(\theta_2)$ ($0 < \theta_1, \theta_2 < 1$)
(ii) $\gamma_2 \leq \gamma_1 \Rightarrow N_{-\infty}(\gamma_1) \subset N_{-\infty}(\gamma_2)$ ($0 < \gamma_2 \leq \gamma_1 \leq 1$)
(iii) $\gamma \leq 1 - \theta \Rightarrow N_2(\theta) \subset N_{-\infty}(\gamma)$

(i) $\left(\begin{matrix} x \\ s \end{matrix} \right) \in N_2(\theta_1) \Rightarrow \|xSe\text{-Mell}\|_2 \leq \theta_1 \mu < \theta_2 \mu \Rightarrow \left(\begin{matrix} x \\ s \end{matrix} \right) \in N_2(\theta_2)$
 $(\theta_1 < \theta_2)$

(ii) $\left(\begin{matrix} x \\ s \end{matrix} \right) \in N_{-\infty}(\gamma_1) \Rightarrow x_i s_i \geq \gamma_1 \mu \quad \forall i \Rightarrow x_i s_i \geq \gamma_2 \mu \Rightarrow \left(\begin{matrix} x \\ s \end{matrix} \right) \in N_{-\infty}(\gamma_2)$
 $(\gamma_1 \geq \gamma_2)$

(iii) $\gamma \leq 1 - \theta \Leftrightarrow \theta \leq 1 - \gamma$

$$\left(\begin{matrix} x \\ s \end{matrix} \right) \in N_2(\theta) \Rightarrow \|xSe\text{-Mell}\|_2 \leq \theta \mu \leq (1 - \gamma) \mu$$

$$\therefore \|xSe\text{-Mell}\|_\infty \leq \|xSe\text{-Mell}\|_2 \leq (1 - \gamma) \mu$$

$$\Rightarrow |x_i s_i - \mu| \leq (1 - \gamma) \mu \quad \forall i$$

$$\text{if } x_i s_i - \mu \geq 0 \text{ then } x_i s_i - \mu \leq (1 - \gamma) \mu \Rightarrow x_i s_i \leq \mu + (1 - \gamma) \mu$$

$$\text{if } x_i s_i - \mu < 0 \text{ then } \mu - x_i s_i \leq (1 - \gamma) \mu \Rightarrow x_i s_i \geq \gamma \mu.$$

If we sort $x_i s_i$ such that $x_{(1)} s_{(1)} \leq \dots \leq x_{(n)} s_{(n)}$, since

$\mu = \bar{x}s/n$, we have $x_{(1)} s_{(1)} \leq \mu \leq x_{(n)} s_{(n)}$. From the argument above $x_{(1)} s_{(1)} \geq \gamma \mu$ so clearly $x_{(1)} s_{(1)} \geq \gamma \mu$.

$$\therefore \left(\begin{matrix} x \\ s \end{matrix} \right) \in N_{-\infty}(\gamma).$$

- 14.5) Show that $N_{-\infty}(1) = N_2(0) = C$.

$$N_{-\infty}(1) = \left\{ \left(\begin{matrix} x \\ s \end{matrix} \right) \mid x_i s_i \geq \mu \quad \forall i \right\} = \left\{ \left(\begin{matrix} x \\ s \end{matrix} \right) \mid x_i s_i = \dots = x_n s_n = \mu \right\} \stackrel{\mu = \bar{x}}{=} C$$

$$N_2(0) = \left\{ \left(\begin{matrix} x \\ s \end{matrix} \right) \mid \|xSe\text{-Mell}\|_2 \leq 0 \right\} = \left\{ \left(\begin{matrix} x \\ s \end{matrix} \right) \mid xSe = \mu e \right\} = \left\{ \left(\begin{matrix} x \\ s \end{matrix} \right) \mid x_i s_i = \dots = x_n s_n = \mu \right\} \stackrel{\mu = \bar{x}}{=} C.$$

Chapter 15: Fundamentals of Algorithms for Nonlinear Constrained Optimization

We now consider the general constrained optimization problem:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \begin{aligned} c_i(x) &= 0, \quad i \in E \\ c_i(x) &\geq 0, \quad i \in I \end{aligned}$$

where f and $c_i, i \in E \cup I$ are all smooth, real-valued functions on a subset of \mathbb{R}^n and E and I are finite index sets.

In Chapter 16, we will discuss algorithms for solving quadratic programming problems. We will consider active set, interior-point and gradient projection methods.

In Chapter 17, we will discuss penalty and augmented Lagrangian methods.

In Chapter 18, we describe sequential quadratic programming methods.

In Chapter 19, the interior-point methods for nonlinear programming is considered.

Combinatorial Difficulty of Inequality-Constrained Problems

Deciding which inequality constraints are active at the solution is a major problem. In active set methods, a working set W will be selected and updated as an estimate of the optimal active set A^* . At each iteration the problem with equality constraints $E \cup W$ is solved and $I \setminus W$ are ignored. If the solution obtained from this subproblem satisfies the original KKT conditions then we can stop. Otherwise, we update W and proceed as described again.

The number of choices for W is $2^{|I|!}$ and it may be very large. In practice, considering all possible choices of W is most likely hopeless. By considering the functions $f_i, \{c_i\}$ and their derivatives, we will try to make smart choices for W .

Interior-point (barrier) methods will try to stay away from the boundary of the feasible set but the estimate will be allowed to become increasingly accurate near the solution as in the case with linear prog. problems.

Elimination of Variables

Example: $\min_{x \in \mathbb{R}^4} f(x)$ s.t.

$$x_1 + x_2 - x_3 - x_4 = 0$$

$$-x_2 + x_4 + x_3 = 0$$

Without any risk, we can set $x_1 = x_3 x_4 - x_3^2$
 $x_2 = x_4 + x_3^2$

Substituting these in $f(x)$:

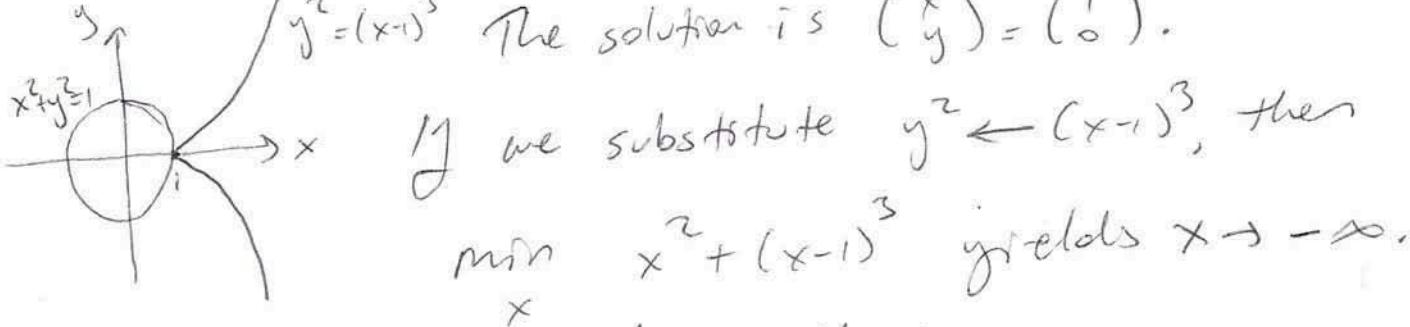
$$h(x_3, x_4) = f(x_3 x_4 - x_3^2, x_4 + x_3^2, x_3, x_4)$$

We can then solve the unconstrained problem

$$\min_{(x_3, x_4)} h(x_3, x_4).$$

Example: $\min x^2 + y^2$ s.t. $(x-1)^3 = y^2$

$y^2 = (x-1)^3$ The solution is $(\begin{matrix} x \\ y \end{matrix}) = (\begin{matrix} 1 \\ 0 \end{matrix})$.



If we substitute $y^2 \leftarrow (x-1)^3$, then
 $\min_x x^2 + (x-1)^3$ yields $x \rightarrow -\infty$.

The mistake here was the following.

$y^2 \geq 0$ by construction so we must have $(x-1)^3 \geq 0$

which implies $x \geq 1$. The correct equivalent problem
 is still constrained: $\min_x x^2 + (x-1)^3$ s.t. $x \geq 1$.

Elimination using linear constraints

Consider $\min f(x)$ s.t. $Ax=b$ where $A \in \mathbb{R}^{m \times n}$. Assume that A has full row rank. Then we can find m linearly independent columns of A . Gathering these columns into a matrix B can be achieved by finding an $n \times n$ permutation P such that $AP = [B|N]$.

Similar to the simplex method, we define sub-variables and B is the basis matrix. Since $PP^T = I$, we have $\begin{bmatrix} x_B \\ x_N \end{bmatrix} = P^T x$ where x_B is the vector of basic variables.

$$b = Ax = AP P^T x = Bx_B + Nx_N.$$

Then $x_B = B^{-1}b - B^{-1}N x_N$. Consequently, for every feasible point x , the component x_N uniquely determines it. The following substitution yields an unconstrained problem:

$$\min_{x_N} h(x_N) \triangleq f\left(P \begin{bmatrix} B^{-1}b - B^{-1}N x_N \\ x_N \end{bmatrix}\right)$$

This is simple elimination of variables.

Now we show that the simple elimination technique expresses feasible points as the sum of a particular solution of $Ax=b$ plus a displacement along the null space of the constraints.

Suppose that the variables in x are indexed from 1 to n such that $A = [B|N]$ (i.e. $P=I$) already has a nonsingular $m \times n$ matrix B in the first m columns of A . Then

$$\begin{bmatrix} x_B \\ x_N \end{bmatrix} = x = Yb + Zx_N$$

$$\text{where } Y = \begin{bmatrix} B^{-1} \\ 0 \end{bmatrix} \text{ and } Z = \begin{bmatrix} -B^{-1}N \\ I \end{bmatrix}.$$

Since $Ax = AYb + AZx_N = b$ $\forall x_N$, we have $AZ=0$.

Therefore columns of Z form a basis for the null space of A . Also $AYb=b$ so $x=Yb$ is a particular solution.

General Reduction Strategies for Linear Constraints

Suppose we choose $Y \in \mathbb{R}^{n \times m}$ and $Z \in \mathbb{R}^{n \times (n-m)}$ such that $[Y|Z] \in \mathbb{R}^{n \times n}$ is nonsingular, $AZ=0$.

Since A has full row rank, $A\{Y|Z\} = \{AY|0\}$ also has full row rank. Consequently AY is nonsingular. Let $x = Yx_Y + Zx_Z$. Then

$$Ax = AYx_Y + \cancel{AZx_Z} = b$$

so $x_Y = (AY)^{-1}b$. Substituting this above:

$$x = Y(AY)^{-1}b + Zx_Z.$$

We now have the following equivalent unconstrained problem:

$$\min_{x_Z} f(Y(AY)^{-1}b + Zx_Z)$$

Given the QR factorization of A^T : $A^T\pi = \{Q_1 Q_2\}^T R$

where $\{Q_1 Q_2\}$ is orthogonal and R is $m \times n$ upper triangular and nonsingular (with π , an $m \times m$ permutation),

we can define $Y = Q_1$, $Z = Q_2$ so that the columns of Y and Z are orthonormal.

Then $AY = \pi R^T$ and $AZ = 0$, and the condition number of AY is the same as that of R , which is the same of that of A . Also

$$x = Q_1 R^{-T} \pi^T b + Q_2 x_Z \text{ for some } x_Z,$$

$$\text{where } Q_1 R^{-T} \pi^T b = A^T (A A^T)^{-1} b.$$

Notice that $A^T(AA^T)^{-1}b$ is the solution to

$$\min \|Ax\|^2 \text{ s.t. } Ax=b.$$

i.e., the minimum norm solution to $Ax=b$.

Effect of Inequality Constraints

If inequality ~~or~~ constraints are present with equality constraints, elimination of variables may complicate the inequality constraints; e.g. simple constraints like $x \geq 0$ may be transformed to $c(x) \geq 0$ with some complicated C.L.

Merit Functions

In unconstrained optimization problems, f was the natural merit function and we required the algorithms to reduce f at each step. In nonlinear programming we could define the ℓ_1 ~~penalty~~ ^{merit} function as follows:

$$\phi_1(x; \mu) = f(x) + \mu \left[\sum_{i \in E} |c_i(x)| + \sum_{i \in I} [c_i(x)]^- \right]$$

where $\{z\}^- = \max(0, -z)$ and μ is the penalty parameter.

The ℓ_1 merit function ϕ_1 is not differentiable but it is exact.

Defn (Exact Merit Function)

A merit function $\phi(x; \mu)$ is exact iff $\exists \mu^* > 0$ $\forall \mu > \mu^*$ any local solution of the nonlinear programming problem is a local minimizer of $\phi(x; \mu)$.

The ℓ_2 merit function for equality constrained problems is $\phi_2(x; \mu) = f(x) + \mu \|c(x)\|_2$. This is also not differentiable since at x where $c(x) = 0$, the penalty is nondifferentiable.

For equality constrained problems, Fletcher's augmented Lagrangian is given by

$$\phi_F(x; \mu) = f(x) - \lambda^T(x)c(x) + \frac{1}{2}\mu \sum_{i \in E} c_i(x)^2$$

where $\mu > 0$ and $\lambda(x) = [A(x)A(x)^T]^{-1} A(x) \nabla f(x)$ with $A(x)$ denoting the Jacobian of $c(x)$. This function is both smooth and exact. However, solving for $\lambda(x)$ imposes practical limitations if computationally costly.

For equality constrained problems, the standard augmented Lagrangian is

$$\mathcal{L}_A(x, \lambda; \mu) = f(x) - \lambda^T c(x) + \frac{1}{2} \mu \|c(x)\|_2^2.$$

A trial point (x^t, λ^t) is assessed by comparing the value of $\mathcal{L}_A(x^t, \lambda^t; \mu)$ to that of the current iterate.

A solution (x^*, λ^*) of the nonlinear optimization problem is not a minimizer of \mathcal{L}_A but is a stationary point. Consequently, it is not a merit function in general.

Given a merit function $\phi(x, \mu)$ we can apply line search or trust region methods as discussed before in the unconstrained case.

Filters

Let's define a measure of infeasibility as

$$h(x) = \sum_{i \in E} |c_i(x)| + \sum_{i \in I} [c_i(x)]^+$$

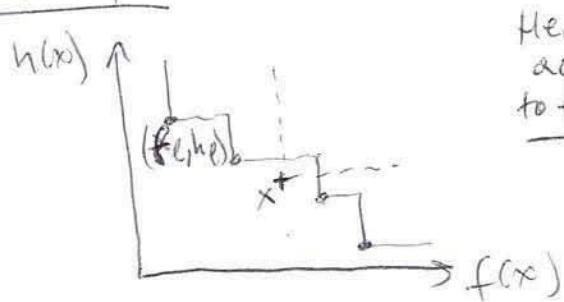
as in the l_1 merit function. The goal of constrained optimization is to solve for $\min_x f(x)$ and $\min_x h(x)$.

The filter methods keep the two goals separate
 → in multi-objective optimization. A total point x^+
 is accepted if the pair $(f(x^+), h(x^+))$ is not
dominated by a previous pair $(f(x_e), h(x_e))$.

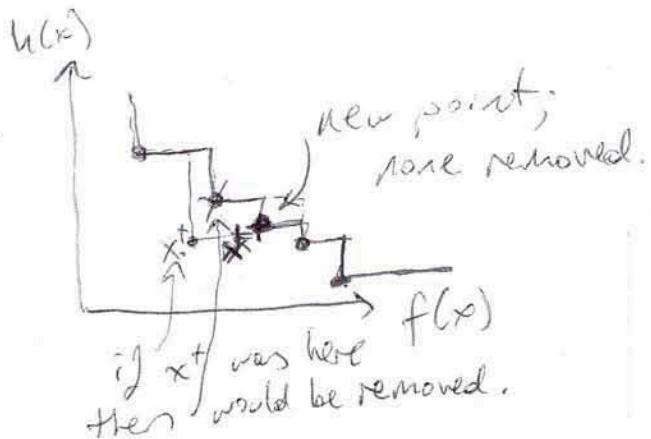
- Defn:
- A pair (f_k, h_k) is said to ~~be~~ dominate another pair (f_ℓ, h_ℓ) if both $f_k \leq f_\ell$ and $h_k \leq h_\ell$.
 - A filter is a list of pairs (f_e, h_e) if no pair dominates any other.
 - An iterate x_k is said to be acceptable to the filter if (f_k, h_k) is not dominated by any pair in the filter.

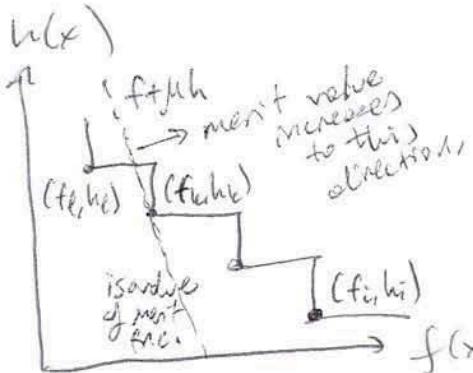
When (f_k, h_k) is acceptable to the filter, we include it in the filter and remove any pairs that are dominated by it from the filter.

Example



Here x^+ is acceptable to the filter.





Consider the merit function

(3.7)

$f + \mu h$. At x_k , the dashed line passing through (f_k, h_k) shows the $f(x)$ isovalue curve (line) of the merit func.

Clearly (f_k, h_k) has a smaller merit value than (f_{k+1}, h_{k+1}) for this specific choice of μ . However if μ is selected differently, one could find the opposite situation.

A line search or trust region method applied to the merit function will yield a candidate x^+ at which (f^+, h^+) is obtained. If this is acceptable to the filter we keep it; if not backtracking line search for a smaller step length or a smaller trust region radius is used for a new trial.

A sufficient decrease condition must be applied to ensure global convergence. A trial x^+ is acceptable to the filter if \forall pairs (f_j, h_j) in the filter

$$f(x^+) \leq f_j - \beta h_j \text{ or } h(x^+) \leq h_j - \beta h_j$$

for $\beta \in (0, 1)$.

If the step length or the trust region radius become too small in order to generate acceptable points, the filter algorithm switches to a feasibility restoration phase and solves

$$\min h(x)$$

$\underset{x}{\star}$ to exclusively reduce the constraint violation.

Alg. 15.1 General Filter Method using Trust Regions.

Choose x_0 and Δ_0 ; Set $k \leftarrow 0$;

repeat until some convergence test is satisfied

if the step-generation subproblem is infeasible

Compute x_{k+1} using the feasibility restoration phase.

else

Compute $x^+ = x_k + p_k$

if (f^+, h^+) is acceptable to the filter

Set $x_{k+1} = x^+$ and add (f_{k+1}, h_{k+1}) to the filter;

Choose $\Delta_{k+1} \geq \Delta_k \geq \Delta_0$

Remove all pairs from the filter, dominated by (f_{k+1}, h_{k+1})

else

Reject the step; set $x_{k+1} = x_k$; Choose $\Delta_{k+1} < \Delta_k$;

end (if)

end (if)

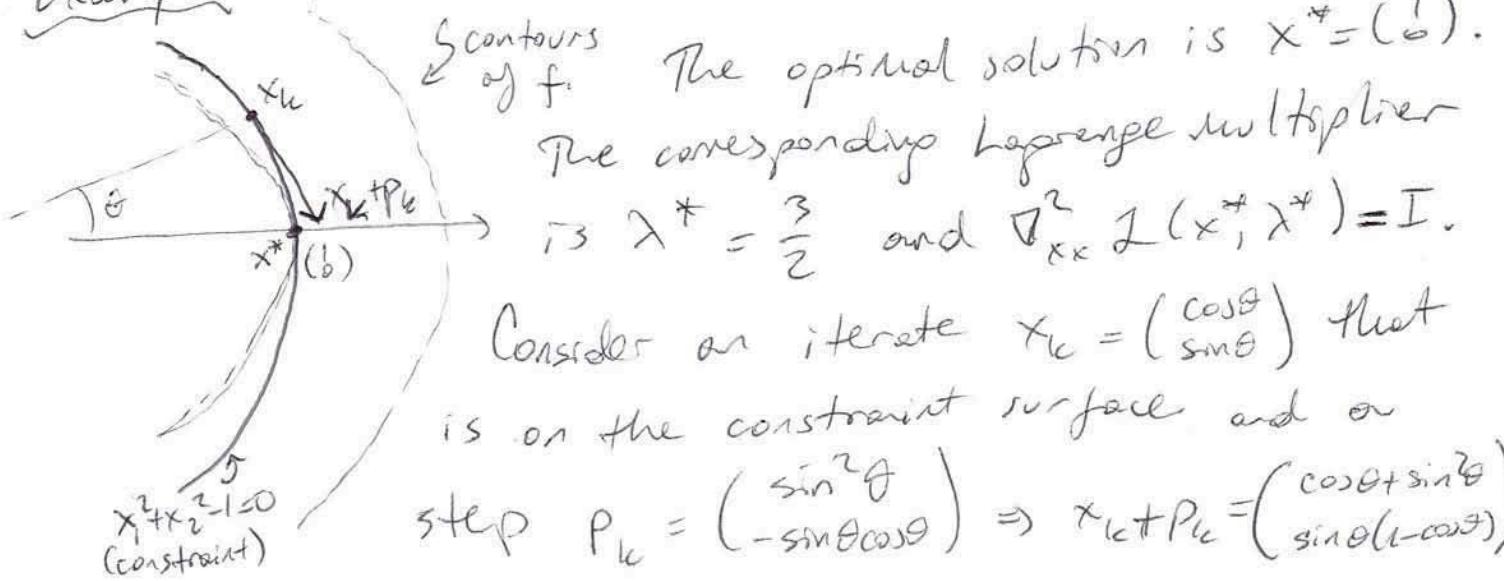
end ($k \leq k+1$)

end (repeat)

The Maratos Effect

Some algorithms based on merit function or filter methods^{may} fail to converge rapidly because they reject steps that make good progress toward a solution; as first observed by Maratos.

Example $\min f(x_1, x_2) = 2(x_1^2 + x_2^2 - 1) - x_1$, s.t. $x_1^2 + x_2^2 - 1 = 0$



$$\text{We observe that } \|x_k + p_k - x^*\|_2 = 2 \sin^2(\theta/2)$$

$$\|x_k - x^*\|_2 = 2 |\sin(\theta/2)|$$

$$\text{and therefore } \frac{\|x_k + p_k - x^*\|_2}{\|x_k - x^*\|_2} = \frac{1}{2}.$$

While the iterate gets closer to the minimizer, both f and c increase: $f(x_k + p_k) = \sin^2\theta - \cos\theta > -\cos\theta = f(x_k)$
 $c(x_k + p_k) = \sin^2\theta > c(x_k) = 0$
so this step will be rejected by both merit ($f + \mu h$) and filter approaches.

Some remedies to the Maratos effect.

- 1) Use a merit function that does not suffer from the Maratos effect (e.g. Fletcher's augmented Lagrangian).
- 2) Use a second order correction term \hat{p}_k computed at $c(x_k + p_k)$, where \hat{p}_k decreases constraint violation.
- 3) Allow the merit function to increase on certain iterations (nonmonotone strategy).

Second-Order Correction

Consider an equality-constraint problem with $c(x)=0$. Given a step p_k , the correction step is

$$\hat{p}_k = -A_k^T (A_k A_k^T)^{-1} c(x_k + p_k)$$

where $A_k \triangleq A(x_k)$ is the Jacobian of c at x_k . Note that \hat{p}_k is the minimum norm solution of

$$A_k \hat{p}_k + c(x_k + p_k) = 0$$

and a step along \hat{p}_k reduces $\|c(x)\|$ to the order of $\|x_k - x^*\|^3$ provided that p_k satisfies $A_k p_k + c(x_k) \neq 0$. This comes at the cost of calculating \hat{p}_k .

The following algorithm assumes that p_k and μ_k are computed so that p_k is a descent direction for the merit function: $D(\phi(x_k; \mu); p_k) < 0$.

If $\alpha_k = 1$ does not produce satisfactory descent in the merit function, the second order correction step is applied before backtracking along p_k .
 * This procedure is effective in practice.

Alg. 15.2 Generic Algorithm with Second-Order Correction

Choose parameters $\eta \in (0, 0.5)$ and τ_1, τ_2 with $\tau_1 < \tau_2 < 1$

Choose initial x_0 and set $k \leftarrow 0$.

repeat until a convergence test is satisfied

 Compute a search direction p_k ;

 Set $\alpha_k \leftarrow 1$, newpoint \leftarrow false;

 while newpoint = false

 if $\phi(x_k + \alpha_k p_k; \mu) \leq \phi(x_k; \mu) + \eta \alpha_k D(\phi(x_k; \mu); p_k)$

 Set $x_{k+1} \leftarrow x_k + \alpha_k p_k$, newpoint \leftarrow true;

 else if $\alpha_k = 1$

 Compute \hat{p}_k ;

 if $\phi(x_k + p_k + \hat{p}_k; \mu) \leq \phi(x_k; \mu) + \eta D(\phi(x_k; \mu); p_k)$

 Set $x_{k+1} \leftarrow x_k + p_k + \hat{p}_k$; newpoint \leftarrow true;

 else

 choose new $\alpha_k \in [\tau_1, \alpha_k, \tau_2 \alpha_k]$

 end

 else choose new $\alpha_k \in [\tau_1 \alpha_k, \tau_2 \alpha_k]$, end (if)

 end (while); end (repeat)

Non-monotone (watchdog) Strategy

In relaxed steps, one could allow the merit function to increase. If a sufficient reduction of the merit function is not obtained within a certain number of iterates of the relaxed step, then we return to the iterate before the relaxed step and perform a normal monotone iteration.

The following algorithm sets the μ ^{number of post-relaxed} steps to $t=1$ and assumes that μ does not change until a successful cycle is completed.

Alg. 15.3 (watchdog)

Choose $\eta \in (0, 0.5)$, x_0 ; set $k \leftarrow 0$, $S \leftarrow \{x_0\}$;
repeat until termination

Compute p_k , set $x_{k+1} \leftarrow x_k + p_k$;

if $\phi(x_{k+1}) \leq \phi(x_k) + \eta D(\phi(x_k); p_k)$, $k \leftarrow k+1$, $S \leftarrow S \cup \{x_k\}$;

else Compute p_{k+1} from x_{k+1} , find α_{k+1} s.t. $\phi(x_{k+1}) \leq \phi(x_{k+2})$

Set $x_{k+2} \leftarrow x_{k+1} + \alpha_{k+1} p_{k+1}$

if $\phi(x_{k+2}) > \phi(x_k)$, (return to x_k , search along p_k)

Find α_k s.t. $\phi(x_{k+2}) \leq \phi(x_k) + \eta \alpha_k D(\phi(x_k); p_k)$

Compute $x_{k+3} = x_k + \alpha_k p_k$, set $k \leftarrow k+3$, $S \leftarrow S \cup \{x_k\}$

else Compute p_{k+2} from x_{k+2} ; find α_{k+2} s.t. $\phi(x_{k+2}) \leq \phi(x_{k+3}) + \eta \alpha_{k+2} D(\phi(x_{k+2}); p_{k+2})$

set $x_{k+3} \leftarrow x_{k+2} + \alpha_{k+2} p_{k+2}$

end end $k \leftarrow k+3$; $S \leftarrow S \cup \{x_k\}$

The set S keeps track of the iterates for which a sufficient decrease was obtained. At least a third of the iterates have their indices in S . This allows global convergence. Also for large k , $\alpha_k=1$ and convergence rate is superlinear.

In practice t^1 may be larger (e.g. 5 or 8). In practice, performance of the watchdog strategy is good and in the best case the algorithm rarely returns to a relaxed iteration. The number of constraint evaluations may be significantly smaller compared to second-order correction.

324

15-4) Show that $\min x^2 + y^2$ s.t. $(x-1)^3 = y^2$ could be converted to an unconstrained problem.

We saw that eliminating y still maintains x^2 as a constraint. Instead consider $(x-1) = y^{2/3}$ since $(\cdot)^{2/3}$ is an invertible (one-to-one) function; then $x = y^{2/3} + 1 \Rightarrow x^2 = (y^{2/3} + 1)^2$. Substituting:

$$\Rightarrow \min_y (y^{2/3} + 1)^2 + y^2 ; y^* = 0, x^* = 1.$$

15-6) Show that for $Ax = b$ with $x = Q_1 R^{-T} \Pi^T b$ where $A^T \Pi = \{Q_1 \ Q_2\} \begin{pmatrix} R \\ 0 \end{pmatrix}$, $x = A^T (A A^T)^{-1} b$ as well.

$$\begin{aligned}
 A^T &= [Q_1 \ Q_2] \begin{pmatrix} R \\ 0 \end{pmatrix} \Pi^T \Leftrightarrow A = \Pi \begin{pmatrix} R^T & 0 \end{pmatrix} \begin{pmatrix} Q_1^T \\ Q_2^T \end{pmatrix} \\
 A A^T &= \Pi \begin{pmatrix} R^T & 0 \end{pmatrix} \underbrace{\begin{pmatrix} Q_1^T \\ Q_2^T \end{pmatrix} \{Q_1 \ Q_2\}}_I \begin{pmatrix} R \\ 0 \end{pmatrix} \Pi^T = \Pi \begin{pmatrix} R^T & 0 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} \Pi^T \\
 &= \Pi (R^T R) \Pi^T \quad \Rightarrow (A A^T)^{-1} = \Pi (R^T R)^{-1} \Pi^T \\
 x &= A^T (A A^T)^{-1} b = \{Q_1 \ Q_2\} \underbrace{\begin{pmatrix} R \\ 0 \end{pmatrix} \Pi^T}_{\Pi} (R^T R)^{-1} \Pi^T b \\
 &= Q_1 R (R^T R)^{-1} \Pi^T b = Q_1 \underbrace{R^T}_{I} R^{-1} R^{-T} \Pi^T b \\
 &= Q_1 R^{-T} \Pi^T b.
 \end{aligned}$$

Chapter 16: Quadratic Programming

The general quadratic program (QP) can be stated as

$$\min_x q(x) = \frac{1}{2} x^T G x + x^T c$$

$$\text{subject to } a_i^T x = b_i \quad i \in \mathcal{E}$$

$$a_i^T x \geq b_i \quad i \in \mathcal{I}$$

where $G \in \mathbb{R}^{n \times n}$ is symmetric; \mathcal{E} and \mathcal{I} are finite sets of indices; $c, x, \{a_i\} \in \mathbb{R}^n$.

QP can always be solved or shown to be infeasible in a finite amount of computation. If $G \geq 0$, the QP is strictly convex; if $G \geq 0$, the QP is convex; if G is indefinite, then the QP is nonconvex. A nonconvex QP may have multiple stationary points.

Equality Constrained QP

We consider $\min_x q(x) = \frac{1}{2} x^T G x + x^T c$ s.t. $Ax = b$ where $A \in \mathbb{R}^{m \times n}$ with $m < n$. We assume that A has full row rank so that the equality constraints are consistent. The Lagrangian is

$$\mathcal{L}(x, \lambda) = \frac{1}{2} x^T G x + x^T c - \lambda^T (Ax - b)$$

Then we must have

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = Gx^* + c - A^T\lambda^* = 0$$

$$\nabla_\lambda \mathcal{L}(x^*, \lambda^*) = -(Ax^* - b) = 0$$

$$\Leftrightarrow \begin{bmatrix} G & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -c \\ b \end{bmatrix} \text{ at the solution.}$$

If x^* is the solution and x is some estimate, then $x^* = x + p$ for some step p , which satisfies (substituting in the equations above)

$$\xrightarrow{\text{KKT matrix}} \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} -p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix}$$

$$\text{where } h = Ax - b, \quad g = c + Gx, \quad p = x^* - x.$$

Lemma 16.1 Let A have full row rank and assume that the reduced-Hessian matrix $\nabla^2 \mathcal{L}(x^*) \geq 0$. Then the KKT matrix $K = \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix}$ is nonsingular, and hence, $\exists! (\begin{smallmatrix} x^* \\ \lambda^* \end{smallmatrix})$ satisfying the equations above.

Proof: Suppose $\exists (\begin{smallmatrix} w \\ v \end{smallmatrix}) \neq \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} = 0$. Since $Aw = 0$, we must have

$$0 = [w^T \ v^T] \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} = w^T G w$$

Let $Az = 0$ with $\mathbb{z} \in \mathbb{R}^{1 \times (n-m)}$ having full column rank.

The columns of \mathbf{z} span $\text{Null}(A)$, and $w \in \text{Null}(A)$
 so $\mathbf{f}^T u^m w = zu$. Therefore

$$0 = w^T G w = u^T \mathbf{z}^T G \mathbf{z} u$$

Since $\mathbf{z}^T G \mathbf{z} > 0$, we must have $u = 0 \Rightarrow w = 0$

$\Rightarrow A^T v = 0$ (from $Gw + A^T v = 0$). Since A is full row rank, A^T is full column rank and $v = 0$.

\therefore We must have $w = 0, v = 0$ so K is nonsingular.

Thm 16.2: Let A have full row rank and assume that the reduced-Hessian matrix $\mathbf{z}^T G \mathbf{z} > 0$. Then the vector x^* satisfying $\begin{bmatrix} G & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -c \\ b \end{bmatrix}$ is the unique global solution of the equality constrained QP.

Proof: Let x be any other feasible point ($Ax = b$).

Let $p = x^* - x$. $Ax^* = Ax = b \Rightarrow Ap = 0$. Then

$$q(x) = \frac{1}{2} (x^* - p)^T G (x^* - p) + c^T (x^* - p)$$

$$= \frac{1}{2} p^T G p - p^T G x^* - c^T p + q(x^*)$$

From the KKT conditions $Gx^* - A^T \lambda^* = -c$, we have $Gx^* = -c + A^T \lambda^*$ and since $Ap = 0$

$$p^T G x^* = p^T (-c + A^T \lambda^*) = -p^T c.$$

Substituting in $q(x)$ above: $q(x) = \frac{1}{2} p^T G p + q(x^*)$.

Also $f \in C^1(\mathbb{R}^{n-m})$ & $P = ZU$, so

$$q(x) = \frac{1}{2} u^T Z^T G Z u + q(x^*).$$

Since $Z^T G Z \geq 0$, we must have $q(x) > q(x^*)$, except when $u=0$; that is $x=x^*$. $\therefore x^*$ is the unique global solution. \square

As a direct consequence of this proof, we also see that (i) $Z^T G Z \geq 0 \Rightarrow x^*$ is a local min (not strict). (ii) $Z^T G Z$ has negative eigenvalues $\Rightarrow x^*$ is a stationary point.

Direct Solution of the KKT system

Defn: The inertia of a symmetric matrix K is

$$\text{inertia}(K) = (n_+, n_-, n_0)$$

where n_+ is the number of positive eigenvalues, n_- is the number of negative eigenvalues, and n_0 is the number of 0 eigenvalues.

Thm 16.3 Let $K = \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix}$ be the KKT matrix.

Suppose that A has rank m . Then

$$\text{inertia}(K) = \text{inertia}(Z^T G Z) + (m, m, 0).$$

Therefore, $Z^T G Z \geq 0 \Rightarrow \text{inertia}(K) = (n, m, 0)$.

Fact: If $m > 0$, then the KKT matrix is indefinite.

Factoring the Full KKT System

For a symmetric matrix K , the symmetric indefinite factorization is of the form $P^T K P = L B L^T$ where P is a permutation matrix, L is unit lower triangular and B is block-diagonal with 1×1 or 2×2 blocks.

Fact $\text{inertia}(K) = \text{inertia}(B)$.

Fact The 2×2 blocks in B always have 1 negative and 1 positive eigenvalue.

We need to solve $K \begin{bmatrix} -P \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix}$ so

$$PL \underbrace{B \begin{bmatrix} \hat{z} \\ L^T P^T \end{bmatrix}}_z \begin{bmatrix} -P \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix}$$

Procedure: Solve $Lz = P^T \begin{bmatrix} g \\ h \end{bmatrix}$ for z ;

Solve $B\hat{z} = z$ for \hat{z} ;

Solve $L^T \bar{z} = \hat{z}$ for \bar{z} ;

Set $\begin{bmatrix} -P \\ \lambda^* \end{bmatrix} = P\bar{z}$.

Schur-Complement Method

Assume $G > 0$. Then $AG^{-1}(-G_p + A^T \lambda^*) = AG^{-1}g$ and since $-Ap = h$, we have $(AG^{-1}A^T)\lambda^* = AG^{-1}g - h$.

Solving for λ^* : $\lambda^* = (AG^{-1}A^T)^{-1}(AG^{-1}p - h)$.

Then substituting to solve for p : $Gp = A^T\lambda^* - g$
we can obtain the solution. This could be found also
as follows...

Let $\begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix}^{-1} = \begin{bmatrix} C & E \\ E^T & F \end{bmatrix}$. Then

$$C = G^{-1} - G^{-1}A^T(AG^{-1}A^T)^{-1}AG^{-1}$$

$$E = G^{-1}A^T(AG^{-1}A^T)^{-1}$$

$$F = -(AG^{-1}A^T)^{-1}$$

The solution is $\begin{bmatrix} -p_* \\ \lambda_* \end{bmatrix} = \begin{bmatrix} C & E \\ E^T & F \end{bmatrix} \begin{bmatrix} g \\ h \end{bmatrix}$.

$$\begin{bmatrix} C & E \\ E^T & F \end{bmatrix} \begin{bmatrix} g \\ h \end{bmatrix} = \begin{bmatrix} Cg + Eh \\ E^Tg + Fh \end{bmatrix}$$

$Cg + Eh$ and $E^Tg + Fh$ simplify to the expressions
found above.

Null-Space Method

Let $P = Yp_Y + Zp_Z$ where $AZ=0$ and $\{Y|Z\}$ is
nonsingular. These were discussed in the previous chapter.

Substituting P into ~~$A^T(Yp_Y + Zp_Z) = h$~~ , we get

$$(AY)p_Y = -h \text{ since } AZ=0.$$

From $-Gp + A^T\lambda^* = g$, we get $-GYp_Y - GZp_Z + A^T\lambda^* = g$

$$\text{Multiplying by } Z^T: (Z^TGZ)p_Z = -Z^TGYp_Y - Z^Tg$$

Using the Cholesky factorization for $Z^T G Z$, we get
 P_Z ; then we obtain $\rho = \gamma P_Y + \tau P_Z$. The
Lagrange multiplier is the solution to

$$(A\gamma)^T \lambda^* = \gamma^T (g + G\rho)$$

from the first block of equations.

Iterative Solution of the KKT System

Let $x^* = \gamma x_Y + \tau x_Z$. From $Ax^* = b$, we get
 $A\gamma x_Y = b$ which determines x_Y . Substituting
this in the equality constrained QP, we get an
unconstrained problem:

$$\min_{x_Z} \frac{1}{2} x_Z^T Z^T G Z x_Z + x_Z^T c_Z$$

where $c_Z = Z^T G Y x_Y + Z^T c$. The solution for x_Z is
given by $Z^T G Z x_Z = -c_Z$.

Since $Z^T G Z > 0$, we can apply the conjugate
gradient (CG) method.

See the book for Alg. 16.1 and Alg 16.2 that
detail two particular CG implementations for this
problem. (pg 460-463).

Inequality-Constrained Problems

The Lagrangian is in the following form:

$$L(x, \lambda) = \frac{1}{2} x^T G x + c^T x - \sum_{i \in \mathcal{U} \setminus \mathcal{I}} \lambda_i (a_i^T x - b_i)$$

The active set $A(x^*) = \{i \in \mathcal{U} \setminus \mathcal{I} \mid a_i^T x^* = b_i\}$ at x^* yields the KKT conditions:

$$G x^* + c - \sum_{i \in A(x^*)} \lambda_i^* a_i = 0$$

(KKT conditions
for QP)

$$a_i^T x^* = b_i \quad \forall i \in A(x^*)$$

$$a_i^T x^* \geq b_i \quad \forall i \in \mathcal{I} \setminus A(x^*)$$

a result of

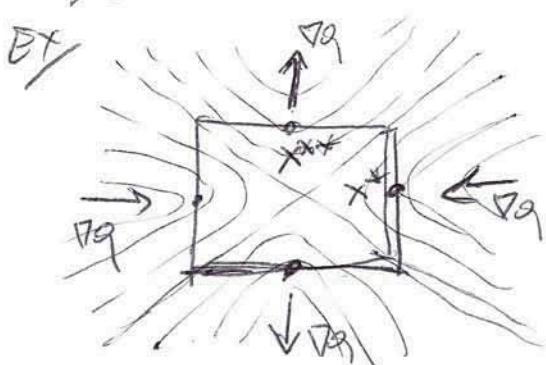
$$\lambda_i^* \geq 0 \quad \forall i \in \mathcal{I} \cap A(x^*)$$

Hence Thm 12.1 holds since the constraints are linear.
In this case, we do not require LICQ to hold (i.e.
the active constraints need not be linearly independent).

Thm 16.4 If x^* satisfies the KKT conditions for the QP for some λ_i^* , $i \in A(x^*)$, and $G \geq 0$, then x^* is a global solution of the QP.

Proof: See pg 465 in the book.

Fact: In the theorem, if $G > 0$, then x^* is the unique global solution.

Nonconvex QP

One local maxst x^{**} and one local minst x^* .

Concave $q(x)$:

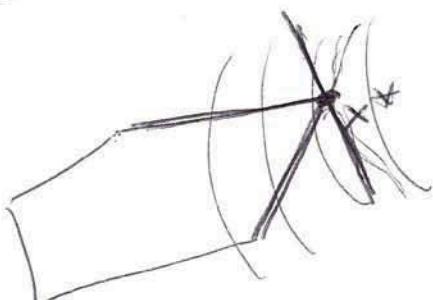
Two local minima x_* , x_{**} are shown.

Degeneracy

- (i) The active constraint gradients α_i , $i \in A(x^*)$ are linearly dependent at x^*
- (ii) The strict complementarity conditions do not hold; i.e. $\exists i \in A(x^*) \text{ s.t. } \text{all } \alpha_i^* = 0$ (i.e. active constraints are weakly active)

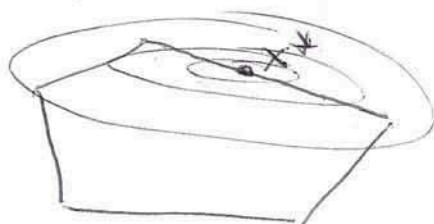
These cause numerical difficulties in algorithms (e.g. rank-deficient matrices).

Ex (i) There is a redundant inequality constraint at x^* ,



3 inequalities are active at $x^* \in \mathbb{R}^2$, so their gradients (3 vectors in \mathbb{R}^2) are not lin. indep.

Ex (ii) Unconstrained minimum appears on the boundary of the feasible set.



Active Set Methods for Convex QP

If $A(x^*)$ is known, then we could solve

$$\min_x q(x) \text{ s.t. } a_i^T x = b_i, i \in A(x^*) \text{ using one}$$

of the methods discussed earlier.

In linear programming, the simplex method repeatedly updated the active set until convergence. In practice, we have an estimate of $A(x^*)$ called the working set W_k .

Given an iterate x_k in W_k .

- 1) we check if x_k minimizes $q(x)$ subject to eq. const in W_k .
- 2) If not we compute p by solving this eq. const. QP (ECQP).

$$p = x - x_k, g_k = Gx_k + c$$

$$\Rightarrow q(x) = q(x_k + p) = \frac{1}{2} p^T G p + g_k^T p + f_k$$

where $f_k = \frac{1}{2} x_k^T G x_k + c^T x_k$. The ECQP is

$$\min_p \frac{1}{2} p^T G p + g_k^T p$$

$$\text{s.t. } a_i^T p = 0 \quad i \in W_k$$

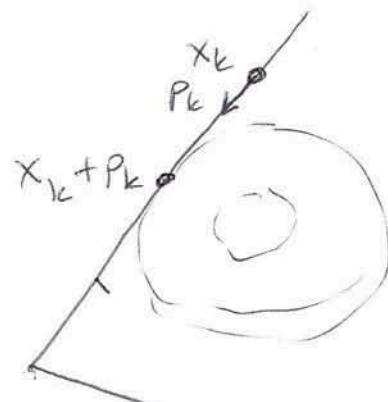
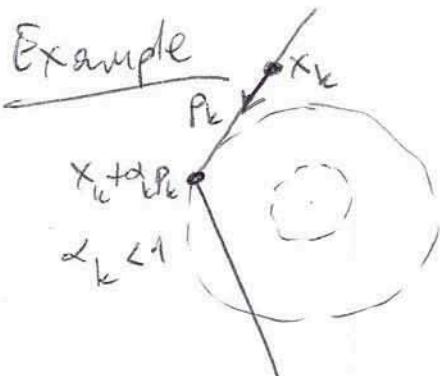
Let p_k be the solution. Note that any $x_k + \alpha p_k$ satisfies the equality constraints.

Since $G > 0$, there is a unique solution to p_k .

3) If $x_k + p_k$ is feasible w.r.t. all constraints we set $x_{k+1} = x_k + p_k$; otherwise we set $x_{k+1} = x_k + \alpha_k p_k$ where $\alpha_k \in \{0, 1\}$ is the largest value for which all constraints are satisfied:

$$\alpha_k \stackrel{\Delta}{=} \min(1, \min_{i \notin W_k, a_i^T p_k < 0} \frac{b_i - a_i^T x_k}{a_i^T p_k})$$

If $\alpha_k < 1$, then the constraints $i \notin W_k$ for which the minimum is achieved are called the blocking constraints; if $\alpha_k = 1$, then there are no new constraints active at $x_k + \alpha_k p_k$.



- 4) If $\alpha_k < 1$, then W_{k+1} is obtained by adding one of the blocking constraints to W_k .
- 5) If $p=0$, then \hat{x} solves the ECQP for its \hat{W} set so that \hat{x} is the solution we seek.

When we obtain $\rho = 0$, since we are at a solution that satisfies the KKT conditions, we have

$$\sum_{i \in \hat{W}} \hat{\lambda}_i a_i = g = G\hat{x} + c$$

for some $\hat{\lambda}_i, i \in \hat{W}$. These $(\hat{x}, \hat{\lambda})$ values are the solution that satisfies KKT conditions if we show that these $\hat{\lambda}$ are all nonnegative.

If one or more $\hat{\lambda}_j, j \in \hat{W} \cap I$ is negative, then KKT conditions are not satisfied and one of these constraints can be dropped from \hat{W} .

Prop 16.5 Suppose that \hat{x} satisfies first-order cond. for the equality-constrained subproblem with $i \in \hat{W}_j$ (i.e. $a_i^T \hat{x} = b_i$). Suppose also that the constraint gradients $a_i, i \in \hat{W}$ are linearly independent and that $j \notin \hat{W} \cap I \hat{\lambda}_j < 0$. Let p be the solution obtained by dropping the constraint j and solving the subproblem $\min_p \frac{1}{2} p^T G p + (G\hat{x} + c)^T p$ s.t. $a_i^T p = 0$ $\forall i \in \hat{W}, i \neq j$.

Then p is a feasible direction for constraint j , that is $a_j^T p \geq 0$. Moreover, if p satisfies

second-order sufficient conditions, then $\mathbf{Q}_j^T \mathbf{P} > 0$
 and \mathbf{p} is a descent direction for $g(\cdot)$.

Proof: See pages 470-471 in the book.

While any index j with $\hat{\lambda}_j < 0$ will yield a feasible descent direction, the one with the largest $|\hat{\lambda}_j|$ is ^{often} selected in practice, since this choice gives the largest decrease based on the sensitivity analysis

for the subproblem

Thm 16.6 Suppose that the solution \mathbf{p}_k is non-zero and satisfies the second-order sufficient conditions for optimality. Then the function $g(\cdot)$ is strictly decreasing along \mathbf{p}_k .

Proof: See pages 471-472 in the book.

Fact When $G > 0$, the second-order conditions are satisfied for all feasible subproblems. Consequently $\mathbf{p}_k = 0$ is only possible at the solution and this can be used as a termination criterion.

Algorithm 16.3

Active-Set Method for Convex QP

Compute a feasible x_0 ;

Set w_0 to be a subset of the active constraints at x_0 ;

for $k=0, 1, 2, \dots$

Solve $P_k = \underset{p}{\operatorname{argmin}} \frac{1}{2} p^T G p + g_k^T p$ s.t. $\alpha_i^T p = 0, i \in w_k$;

if $P_k = 0$

Compute $\hat{\lambda}_i \neq \sum_{i \in \hat{w}} \alpha_i \hat{\lambda}_i = G x_k + c$ with $\hat{w} = w_k$

if $\hat{\lambda}_i > 0 \forall i \in w_k \cap \mathbb{Z}$

stop and set $x^* = x_k$

else

$j \leftarrow \underset{j \in w_k \cap \mathbb{Z}}{\operatorname{argmin}} \hat{\lambda}_j$; $x_{k+1} \leftarrow x_k$; $w_{k+1} \leftarrow w_k \setminus \{j\}$

end (if)

else (i.e. $P_k \neq 0$)

Compute $\alpha_k = \min(1, \min_{i \in w_k, \alpha_i^T P_k < 0} (b_i - \alpha_i^T x_k) / \alpha_i^T P_k)$

if there are blocking constraints

obtain w_{k+1} by adding one blocking const. to w_k

else

end (if); $w_{k+1} \leftarrow w_k$

end (if)

end (for)

* The algorithm can be initialized using the same method as in linear programming. Alternatively,

* Given \tilde{x} , we can solve

$$\min_{(x, z)} e^T z \text{ s.t. } \begin{aligned} & \alpha_i^T x + \gamma_i z_i = b_i, \quad i \in E \\ & \alpha_i^T x + \gamma_i z_i \geq b_i, \quad i \in I \end{aligned}$$

where $e = (1, \dots, 1)^T$, $\gamma_i = -\text{sign}(\alpha_i^T \tilde{x} - b_i)$ for $i \in E$ and $\gamma_i = 1$ for $i \in I$. This is called the feasibility linear program.

$$x = \tilde{x}, \quad z_i = |\alpha_i^T \tilde{x} - b_i|, \quad i \in E, \quad z_i = \max(b_i - \alpha_i^T \tilde{x}, 0) \quad i \in I$$

is a feasible initial point.

Any solution of the feasibility linear program has cost zero, and is a feasible point of the original problem.

* Alternatively, we could introduce a measure η of constraint violation and solve (for some large $M > 0$)

$$\begin{array}{ll} (\text{penalty}) \quad \min_{(x, u)} & \frac{1}{2} x^T G x + x^T C + M u \\ \text{(x-norm)} & \text{s.t. } (\alpha_i^T x - b_i) \leq \eta \quad i \in E \\ \text{penalty} & \quad \quad \quad -(\alpha_i^T x - b_i) \leq \eta \quad i \in I \end{array}$$

For sufficiently large M , the solution will have $\eta=0$, so x will coincide with the desired original problem solution.

$$b_i - \alpha_i^T x \leq \eta \quad i \in I$$

$$0 \leq \eta$$

This problem can be initialized by setting x to any value and then selecting a sufficiently large η so all constraints are satisfied.

Alternatively we could use an ℓ_1 -norm penalty:

$$\min_{(x,s,t,v)} \frac{1}{2} x^T G x + x^T c + M e_E^T (s+t) + M e_Z^T v$$

$$\text{s.t. } \alpha_i^T x - b_i + s_i - t_i = 0, \quad i \in E$$

$$\alpha_i^T x - b_i + v_i \geq 0, \quad i \in I$$

$$s \geq 0, \quad t \geq 0, \quad v \geq 0$$

where $e_E = (1, \dots, 1)^T$ with $|E|$ entries and $e_Z = (1, \dots, 1)^T$ with $|I|^T$ entries. For an arbitrary x , the slack variables s, t, v ensure that feasibility can be satisfied easily.

Fact Under certain assumptions (e.g. if each $\alpha_k \neq 0$) the active-set QP algorithm terminates in a finite number of iterations.

The assumption that $\alpha_k \neq 0$ ensures that the algorithm does not enter a cycle; since the number of possible working sets is finite, we are certain that the algorithm will stop at a solution in finite time.

Updating Factorizations

Each step computation in the active set method requires the solution of the equality constrained subproblem. This involves solving the KKT system of equations. Since the working set can change by only one index, the KKT matrix changes in one row and one column. Recall that $K = \begin{bmatrix} G & A^T \\ A & S \end{bmatrix}$. The row of A and column of A^T corresponding to the working set change is modified.

Let the QR factorization of A^T be

$$A^T \Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = \{Q_1 \ Q_2\} \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where Π is a permutation, R is square upper triangular and Q is orthogonal. If one constraint is added to the working set, then $\bar{A}^T = \{A^T \ \alpha\}$ where α is a column vector such that \bar{A}^T is still full column rank. Note that $QQ^T = Q_1 Q_1^T + Q_2 Q_2^T = I$. So

$$\bar{A}^T \begin{bmatrix} \Pi & 0 \\ 0 & 1 \end{bmatrix} = \{A^T \Pi \ \alpha\} \xrightarrow{\downarrow} Q \begin{bmatrix} R & Q_1^T \alpha \\ 0 & Q_2^T \alpha \end{bmatrix}$$

Since $\begin{bmatrix} R & Q_1^T \alpha \\ 0 & Q_2^T \alpha \end{bmatrix}$ must be upper triangular, we need to have .

$$\bar{A}^T \begin{bmatrix} \bar{\pi} & 0 \\ 0 & 1 \end{bmatrix} = Q \begin{bmatrix} R & Q_1^T \alpha \\ 0 & \hat{Q}^T \begin{bmatrix} \gamma \\ 0 \end{bmatrix} \end{bmatrix} = Q \begin{bmatrix} I & 0 \\ 0 & \hat{Q}^T \end{bmatrix} \begin{bmatrix} R & Q_1^T \alpha \\ 0 & \frac{\gamma}{\alpha} \end{bmatrix}$$

where $\hat{Q}^T Q_2^T \alpha = \begin{bmatrix} \gamma \\ 0 \end{bmatrix}$ for some scalar γ .

The new factorization is $\bar{A}^T \bar{\pi} = \bar{Q} \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix}$ where

$$\bar{\pi} = \begin{bmatrix} \pi & 0 \\ 0 & 1 \end{bmatrix}, \bar{Q} = Q \begin{bmatrix} I & 0 \\ 0 & \hat{Q}^T \end{bmatrix} = \{Q_1 \ Q_2 \hat{Q}^T\}, \bar{R} = \begin{bmatrix} R & Q_1^T \alpha \\ 0 & \gamma \end{bmatrix}.$$

∴ We only need to find \hat{Q} & $\hat{Q}(Q_2^T \alpha) = \begin{bmatrix} \gamma \\ 0 \end{bmatrix}$.

* As an exercise, determine a procedure for finding such \hat{Q} . One possibility is to use Gram-Schmidt orthogonalization.

Interior-Point Methods

Consider $\min_x q(x) = \frac{1}{2} x^T G x + x^T C$ s.t. $Ax \geq b$,

where $G \succ 0$ is symmetric and $b \in \mathbb{R}^m$ with $m \leq n$.

Equality constraints, if any exists, can be eliminated by reparametrizing the problem as discussed earlier.

The KKT conditions are

$$Gx - A^T \lambda + c = 0$$

$$Ax - b \geq 0$$

$$(Ax - b)_i \lambda_i = 0 \quad i=1, \dots, m$$

$$\lambda \geq 0$$

By introducing the slack vector $y \geq 0$, we rewrite as

$$Gx - A^T \lambda + c = 0$$

$$Ax - y - b = 0$$

$$y_i \lambda_i = 0 \quad i=1, \dots, m$$

$$(y, \lambda) \geq 0$$

which brings the problem into standard form.

Since $G \geq 0$, these conditions are necessary and sufficient.

Given a current (x, y, λ) with $(y, \lambda) > 0$ and using the complementarity measure $\mu = \frac{y^T \lambda}{m}$, we get

$$F(x, y, \lambda; \sigma\mu) = \begin{bmatrix} Gx - A^T \lambda + c \\ Ax - y - b \\ y \Lambda e - \sigma y \mu e \end{bmatrix} = 0$$

where $\Lambda = \text{diag}(y_1, \dots, y_m)$, $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_m)$, $e = (1, \dots, 1)^T$, and $\sigma \in \{0, 1\}$. The solutions of this system of equations for each σ define the central path. As $\mu \rightarrow 0$, the central path tends to the solution.

For a fixed μ , the Newton method yields:

$$\begin{bmatrix} G & 0 & -A^T \\ A & -I & 0 \\ 0 & \Lambda & Y \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -\Lambda y + \sigma y \mu e \end{bmatrix}$$

where $r_d = Gx - A^T \lambda + c$, $r_p = Ax - y - b$.

Using a step length of α , we update the parameters:

$$(x^+, y^+, \lambda^+) = (x, y, \lambda) + \alpha (\Delta x, \Delta y, \Delta \lambda)$$

where α ensures that $(y^+, \lambda^+) > 0$.

Solving the Primal-Dual System

From $A\Delta x - \Delta y = -r_p$, we get $\Delta y = A\Delta x + r_p$.

Substituting this into $\lambda \Delta y + \gamma \Delta \lambda = -\lambda^* y + \gamma^* \lambda$:

$$\lambda A \Delta x + \gamma \Delta \lambda = -\lambda^* y - \lambda^* r_p + \gamma^* \lambda$$

Multiplying this with λ^* from left, we get the augmented system:

$$\begin{bmatrix} G & -A^T \\ A & \lambda^* y \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p + (-y + \gamma^* \lambda^*) \end{bmatrix}$$

From the second equation here, we get

$$\Delta \lambda = \gamma^* \lambda \left\{ -A \Delta x - r_p + \underbrace{(m)}_{\text{in}} \right\}$$

Substituting in the first equation:

$$G \Delta x + A^T y^* \lambda \Delta x = -r_d + A^T y^* \lambda \left\{ -r_p + (m) \right\}$$

which yields the normal equations:

$$\{G + A^T y^* \lambda A\} \Delta x = \{-r_d + A^T y^* \lambda (-r_p + \gamma^* \lambda^*)\}$$

This can be solved by employing a modified Cholesky factorization

We can also use the projected CG method. By rewriting the equations as follows

$$\begin{bmatrix} G & 0 & -A^T \\ 0 & y^T & I \\ A & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -\lambda e + y^T e \\ -r_p \end{bmatrix}$$

we notice that these are the optimality conditions for an equality constrained convex quadratic program where the variable is $(\Delta x, \Delta y)$. Then, we can use Alg. 16.2, the projected CG algorithm to solve this system of equations.

Step length selection

Since using separate step lengths α^{pri} and α^{dual} for the primal and dual problems may help interior point methods' performance, we will do so.

$$(x^+, y^+) = (x, y) + \alpha^{pri} (\Delta x, \Delta y); \quad \lambda^+ = \lambda + \alpha^{dual} \Delta \lambda$$

where α^{pri} and α^{dual} ensure $(y^+, \lambda^+) \geq 0$. The new residuals satisfy (from $r_d = Gx - A^T \lambda + c$; $r_p = Ax - y - b$)

$$r_p^+ = (1 - \alpha^{pri}) r_p$$

$$r_d^+ = (1 - \alpha^{dual}) r_d + (\alpha^{pri} - \alpha^{dual}) G \Delta x$$

If $\alpha^{\text{pri}} = \alpha^{\text{dual}} = \alpha$, then both residuals decrease linearly if $\alpha \in (0, 1)$. For different step lengths, α^{pri} may increase for some choices of α^{pri} and α^{dual} .

* For equal step lengths, we could use $\alpha = \min(\alpha_{\tau}^{\text{pri}}, \alpha_{\tau}^{\text{dual}})$

where $\alpha_{\tau}^{\text{pri}} = \max\{\alpha \in [0, 1] : y + \alpha \Delta y \geq (1-\tau)y\}$

$$\alpha_{\tau}^{\text{dual}} = \max\{\alpha \in [0, 1] : \lambda + \alpha \Delta \lambda \geq (1-\tau)\lambda\}$$

and $\tau \in (0, 1)$ controls how far back we are from the maximum step for which $y^+ \geq 0$ and $\lambda^+ \geq 0$ are satisfied.

* For faster convergence, we could use different step lengths such that the optimality measure is approximately minimized:

$$\min_{\{\alpha^{\text{pri}}, \alpha^{\text{dual}}\}} \|Gx^+ - A^T \lambda^+ + c\|_2^2 + \|(Ax^+ - y^+ - b)\|_2^2 + (y^+)^T z^+$$

subject to $0 \leq \alpha^{\text{pri}} \leq \alpha_{\tau}^{\text{pri}}$ and $0 \leq \alpha^{\text{dual}} \leq \alpha_{\tau}^{\text{dual}}$.

A Practical Primal-Dual Method

First an affine scaling step $(\Delta x^{\text{aff}}, \Delta y^{\text{aff}}, \Delta \lambda^{\text{aff}})$ is computed using $\sigma = 0$ in the Newton equations. Then a corrector step is implemented as before.

Then the centering parameter σ is computed as before.
The total step obtained is given by

$$(*) \quad \begin{bmatrix} G & 0 & -A^T \\ A & -I & 0 \\ 0 & I & \gamma \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -\gamma y e^{-\Delta \lambda^{eff}} \Delta y + \gamma y \end{bmatrix}$$

Alg-16.4 Predictor-Corrector Algorithm for QP

Compute (x_0, y_0, λ_0) with $(y_0, \lambda_0) > 0$;

for $k=0, 1, 2, \dots$

Set $(x, y, \lambda) = (x_k, y_k, \lambda_k)$ and solve the Newton eqns
with $\sigma=0$ for $(\Delta x^{eff}, \Delta y^{eff}, \Delta \lambda^{eff})$;

Calculate $\mu = y^T \lambda / m$;

Calculate $\bar{\alpha}_{eff} = \max \{ \alpha \in (0, 1) | (y, \lambda) + \alpha (\Delta y^{eff}, \Delta \lambda^{eff}) \geq 0 \}$

Calculate $M_{eff} = (y + \bar{\alpha}_{eff} \Delta y^{eff})^T (\lambda + \bar{\alpha}_{eff} \Delta \lambda^{eff}) / m$;

Set centering parameter to $\sigma = (M_{eff} / \mu)^3$;

Solve (*) above for $(\Delta x, \Delta y, \Delta \lambda)$;

Choose $\tau_k \in (0, 1)$ and set $\hat{\alpha} = \min(\alpha_{\tau_k}^{pri}, \alpha_{\tau_k}^{dual})$;

Set $(x_{k+1}, y_{k+1}, \lambda_{k+1}) = (x_k, y_k, \lambda_k) + \hat{\alpha} (\Delta x, \Delta y, \Delta \lambda)$;

end (for)

* To accelerate convergence let $\tau_k \rightarrow 1$ as iterates approach the solution.

The Gradient Projection Method

In the active set methods discussed above, the working set changes one constraint at a time. The gradient projection method will allow more indices to change and will potentially decrease the number of iterations. The gradient projection method is most effective for the following problem:

$$\min_x q(x) = \frac{1}{2} x^T G x + x^T c \quad \text{s.t. } l \leq x \leq u$$

where G is symmetric.

The feasible set in this problem is a "box". Some components may have $l_i = -\infty$ or $u_j = +\infty$.

Cauchy Point Computation

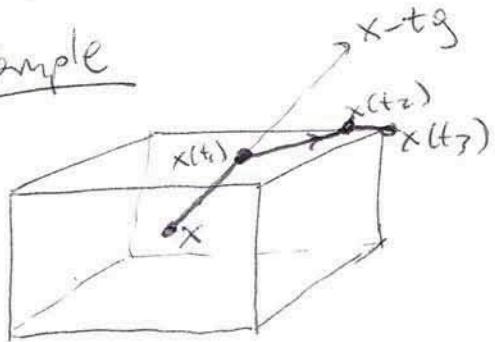
The projection of an arbitrary point x onto the feasible region is defined as follows:

$$P(x, l, u)_i = \begin{cases} l_i & \text{if } x_i < l_i \\ x_i & \text{if } x_i \in [l_i, u_i] \\ u_i & \text{if } x_i > u_i \end{cases}$$

(This basically employs a two-sided saturation function).

Let $g = Gx + c$ be the gradient of the objective.
 Then the path along the steepest descent direction
 is projected to $x(t) = P(x - tg, l, u)$.

Example



Defn: The Cauchy point x^c is the first local minimizer of the univariate, piecewise quadratic function $q_t(x)$ for $t \geq 0$.

The values of t for which each component reaches the boundary along $-g$ are

$$\bar{t}_i = \begin{cases} (x_i - u_i)/g_i & \text{if } g_i < 0 \text{ and } u_i < +\infty \\ (x_i - l_i)/g_i & \text{if } g_i > 0 \text{ and } l_i > -\infty \\ \infty & \text{otherwise} \end{cases}$$

Then, the components of $x(t)$ are $x_i(t) = \begin{cases} x_i - t g_i, & t \leq \bar{t}_i \\ x - \bar{t}_i g, & \text{O.w.} \end{cases}$

After eliminating any zeros and duplicate values from the set $\{\bar{t}_1, \dots, \bar{t}_n\}$ and sorting to get $\{t_1, \dots, t_n\}$ where $0 \leq t_1 < t_2 < \dots < t_n$, we examine the intervals $[t_i, t_{i+1}]$.

Suppose that up to t_{j-1} we have not found a local minimizer. For $[t_{j-1}, t_j]$ we have

$$x(t) = x(t_{j-1}) + \Delta t \rho^{j-1}$$

where $\Delta t = t - t_{j-1} \in [0, t_j - t_{j-1}]$,

$$\text{and } p_i^{j-1} = \begin{cases} -g_i & \text{if } t_{i-1} < \bar{t}_i \\ 0 & \text{otherwise} \end{cases}$$

(350)

The quadratic on the line segment $\{x(t_{j-1}), x(t_j)\}$ is

$$\begin{aligned} q_t(x(t)) &= c^T (x(t_{j-1}) + (\Delta t)p^{j-1}) + \frac{1}{2} (x(t_{j-1}) + (\Delta t)p^{j-1})^T G \\ &\quad (x(t_{j-1}) + (\Delta t)p^{j-1}) \\ &= f_{j-1} + f'_{j-1} \Delta t + \frac{1}{2} f''_{j-1} (\Delta t)^2, \quad \Delta t \in [0, t_j - t_{j-1}] \end{aligned}$$

where $f_{j-1}, f'_{j-1}, f''_{j-1}$ are defined appropriately. The unconstrained ~~stationary point~~ is at $\Delta t^* = -f'_{j-1} / f''_{j-1}$.

(i) if $f'_{j-1} > 0$, t_{j-1} is a local minimizer

(ii) else if $\Delta t^* \in [0, t_j - t_{j-1}]$, $t_{j-1} + \Delta t^*$ is a minimizer.

(iii) else we move to $[t_j, t_{j+1}]$.

Noting that p^j typically differs from p^{j-1} in one component, computational savings are possible.

Subspace Minimization

Once the Cauchy point x^c is computed, the components that are at their respective bounds define the active set.

$$A(x^c) = \{i \mid x_i^c = l_i \text{ or } x_i^c = u_i\}$$

We approximately solve the following QP:

$$\min_x q_t(x) = \frac{1}{2} x^T G x + x^T c \quad \text{s.t. } x_i = x_i^c, i \in A(x^c) \\ l_i \leq x_i \leq u_i, i \notin A(x^c)$$

We only need an approximate solution x^+ that is feasible w.r.t. the original problem and $q(x^+) \leq q(x^c)$. Under these conditions, the gradient projection procedure will converge globally.

Alg. 16.5 Gradient Projector Method for QP

Compute a feasible x^0 ;

for $k = 0, 1, 2, \dots$

if x^k satisfies the KKT conditions

stop; $x^+ = x^k$;

Set $x = x^k$ and find the Cauchy point x^c ;

Find an approximate solution x^+ as described above, such that $q(x^+) \leq q(x^c)$ and x^+ feasible.

$x^{k+1} \leftarrow x^+$

end (for)

* Recall that the dual problem has $x \geq 0$ as its constraints so the gradient-projector method could be used to solve the dual problem.

Chapter 17: Penalty and Augmented Lagrangian Methods

The Quadratic Penalty Method: we will replace the constrained optimization problem by a single function that is the original objective plus a "quadratic" additive term for each constraint that becomes zero when the constraint is satisfied.

Consider $\min_x f(x)$ s.t. $c_i(x) = 0 \quad i \in E$. The

quadratic penalty function is (with $\mu > 0$)

$$Q(x; \mu) \stackrel{\Delta}{=} f(x) + \frac{\mu}{2} \sum_{i \in E} c_i^2(x).$$

If we consider a sequence $\mu_k \rightarrow \infty$, then we can impose increasingly severe penalty for constraint violations.

Framework 17.1 Quadratic Penalty Method

Given $\mu_0 > 0$, $\{\tau_k\} \geq 0$ with $\tau_k \rightarrow 0$ and x_0^S ; for $k=0, 1, 2, \dots$

find an approximate minimizer x_k of $Q(\cdot; \mu_k)$ starting at x_k^S , terminating when $\| \nabla_x Q(x_k; \mu_k) \| \leq \tau_k$ if final convergence test is satisfied

STOP with approximate solution x_k

end(i,j)

Choose new penalty parameter $\mu_{k+1} > \mu_k$;

Choose new starting point x_{k+1}^s ;
end (for)

We must include safeguards such as restoring the initial point in order to recover from situations in which the penalty term is diverging or is not decreasing fast enough.

Convergence of the Quadratic Penalty Method

Theorem 17.1 Suppose that each x_k is the exact global minimizer of $Q(x; \mu_k)$ defined by the quadratic penalty term as given above, and that $\mu_k \rightarrow \infty$. Then every limit point x^* of the sequence $\{x_k\}$ is a global solution of the problem with equality constraints above.

Proof: See pages 502-503 in the book.

This theorem assumes that we are able to find the global minimizer of $Q(x; \mu_k)$, which is not always possible or easy in practice.

Thm 17.2 Suppose that the tolerances and penalty parameters in framework 17.1 satisfy $\epsilon_k \rightarrow 0$ and $\mu_k \rightarrow \infty$. Then if a limit point x^* of the sequence $\{x_k\}$ is feasible and the constraint gradients $\nabla c_i(x^*)$ are linearly independent, then x^* is a KKT point for the equality-constrained problem.

For such points, we have for any infinite subsequence K such that $\lim_{k \in K} x_k = x^*$ that

$$\lim_{k \in K} -\mu_k c_i(x_k) = \lambda_i^* \quad \forall i \in E$$

where λ^* is the multiplier vector that satisfies the KKT conditions for the equality-constrained problem.

Proof: See pages 503-505 in the book.

Notice that if a limit point x^* is not feasible, then it will at least be a stationary point for $\|c(x)\|^2$. This is similar to root finding ($r(x)=0$) by minimizing $\|r(x)\|^2$. Our algorithms, being local minimizers, will get attracted to such local minima.

Ill Conditioning

Consider the following fission:

$$\nabla_{xx}^2 Q(x; \mu_k) = \nabla^2 f(x) + \sum_{i \in E} \mu_k c_i(x) \nabla^2 c_i(x) + \mu_k A(x)^T A(x)$$

where $A(x)^T \triangleq \{\nabla c_i(x)\}_{i \in E}$ is the constraint Jacobian transposed. If the conditions in Thm 17.2 are satisfied, then

$$\nabla_{xx}^2 Q(x; \mu_k) \approx \nabla_{xx}^2 L(x, \lambda^*) + \mu_k A(x)^T A(x)$$

when x is close to the minimizer of $Q(\cdot; \mu_k)$.

The first term in this approximation is independent of μ_k and the second term is of rank-1 w.r.t. μ_k and the order of μ_k . Since usually we have eigenvalues on the order of μ_k , the second term is usually singular. Then $|\mathcal{E}| < n$, the second term is usually singular. Then

the eigenvalues of $\nabla_{xx}^2 Q(x; \mu_k)$ are in two groups:

(i) those that approach a constant;

(ii) those that grow unboundedly with μ_k .

Clearly, in this case, the Newton step given by

$$\nabla_{xx}^2 Q(x; \mu_k) p = -\nabla_x Q(x; \mu_k)$$

will become increasingly inaccurate due to ill conditioning.

If we let $\zeta = \mu A(x)p$, then the vector p that satisfies the Newton equations above also satisfies

$$\begin{bmatrix} \nabla^2 f(x) + \sum_{i \in E} \mu_i c_i(x) \nabla^2 c_i(x) & A(x)^T \\ A(x) & -I/\mu_e \end{bmatrix} \begin{bmatrix} p \\ \zeta \end{bmatrix} = \begin{bmatrix} -\nabla_x Q(x, \mu_e) \\ 0 \end{bmatrix}$$

This new equation will yield p without numerical issues.

Nonsmooth Penalty Functions

The L_1 penalty function is exact and it is popular.

$$\phi_1(x; \mu) = f(x) + \mu \sum_{i \in E} |c_i(x)| + \mu \sum_{i \in I} [c_i(x)]^-$$

where $[y]^- \triangleq \max\{0, -y\}$

Thm 17.3: Suppose that x^* is a strict local solution of the nonlinear programming problem $\min_x f(x)$ s.t. $c_i(x) = 0$ if $i \in E$ and $c_i(x) \geq 0$ if $i \in I$ at which the first-order necessary conditions of Thm 12.1 are satisfied, with Lagrange multipliers λ_i^* , $i \in E \cup I$. Then x^* is a local minimizer of $\phi_1(x; \mu)$ if $\mu > \mu^*$ where

$$\mu^* = \|\lambda^*\|_\infty = \max_{i \in E \cup I} |\lambda_i^*|.$$

If, in addition, the second-order sufficient conditions of Thm 12.6 hold and $\mu > \mu^*$, then x^* is a strict local minimizer of $\phi_1(x; \mu)$.

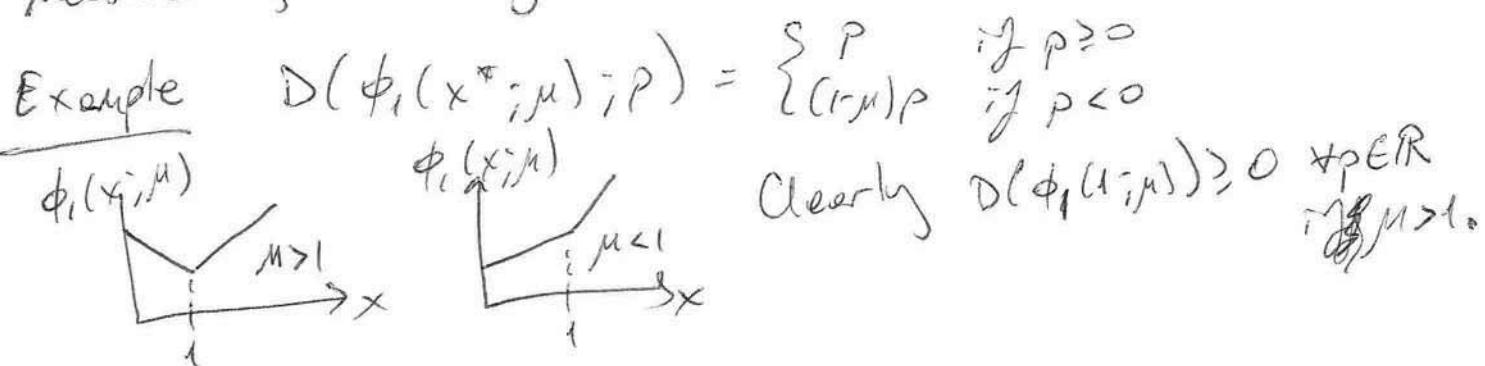
Since ϕ_1 is not differentiable, consider the alternative definition of a stationary point using directional derivatives:

Defn 17.1 A point $\vec{x} \in \mathbb{R}^n$ is a stationary point for

the penalty function $\phi_1(\vec{x}; \mu)$ if $D(\phi_1(\vec{x}; \mu); p) \geq 0$

$\forall p \in \mathbb{R}^n$. Similarly, \vec{x} is a stationary point of the measure of infeasibility $h(\vec{x}) = \sum_{i \in E} |c_i(\vec{x})| + \sum_{i \in I} c_i(\vec{x})^+$ if $D(h(\vec{x}); p) \geq 0 \quad \forall p \in \mathbb{R}^n$.

If a point is infeasible for the nonlinear programming problem, but stationary with respect to the infeasibility measure h , we say that it is an infeasible stationary point.



Thm 17.4 Suppose that \vec{x} is a stationary point of the penalty function $\phi_1(\vec{x}; \mu)$ $\forall \mu > \hat{\mu} > 0$. Then if \vec{x} is feasible for the nonlinear program with equality and inequality constraints, it satisfies the KKT conditions. If \vec{x} is not feasible for the nonlinear program, it is an infeasible stationary point.

Proof: See pages 509-510 in the book.

Framework 17.2 Classical ℓ_1 -Penalty Method

Given $M_0 > 0$, tolerance $\tau > 0$, starting point x_0^S ;

for $k = 0, 1, 2, \dots$

Find an approximate minimizer x_k of $\phi_i(x; M_k)$
starting at x_{k+1}^S ;

if $\|n(x_k)\| \leq \tau$

STOP with approximate solution x_k

end (if)

Choose new penalty parameter $M_{k+1} \geq M_k$;

Choose new starting point x_{k+1}^S ;

end (for)

Practical ℓ_1 -Penalty Method

We will linearize the constraints and model the objective with a quadratic approximation.

$$\begin{aligned} q(p; \mu) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T W p + \mu \sum_{i \in E} |c_i(x) + \nabla c_i(x)^T p| \\ + \mu \sum_{i \in I} [c_i(x) + \nabla c_i(x)^T p]. \end{aligned}$$

Introducing slack variables r_i, s_i , and t_i , we can rewrite this problem as

$$\begin{array}{ll} \min_{p, r, s, t} & f(x) + \frac{1}{2} p^T W p + \nabla f(x)^T p + \mu \sum_{i \in E} (r_i + s_i) + \mu \sum_{i \in I} t_i \\ \text{s.t.} & \nabla c_i(x)^T p + c_i(x) = r_i - s_i, \quad i \in E \\ & \nabla c_i(x)^T p + c_i(x) \geq -t_i, \quad i \in I \\ & r, s, t \geq 0 \end{array}$$

This is a standard quadratic programming problem. We can also include a trust region constraint: $\|p\|_\infty \leq \Delta$. Sequential quadratic programming (SQP) methods can be employed here (later in chapter 18).

General Nonsmooth Penalty Functions

Consider $\phi(x; \mu) = f(x) + \mu \|c_E(x)\| + \mu \|c_I(x)\|$ where $\|\cdot\|$ is any vector norm. The infeasibility measure is

$$h(x) = \|c_E(x)\| + \|c_I(x)\|$$

Any l_p norm can be used clearly (without the p -power). Then in Thm 17.3 $\mu^* = \|\lambda^*\|_D$ where $\|\cdot\|_D$ is the dual norm of $\|\cdot\|$. Then 17.4 applies as is.

Penalty functions of this type are always nonsmooth. (For a brief argument leading to this see page 513 in the book.)

Augmented Lagrangian Method: Equality Constraints

This approach is also called the method of multipliers.
We consider the equality-constrained problem:

$$\min_x f(x) \text{ s.t. } c_i(x) = 0 \quad i \in E$$

In Thm 17-2, we saw that for the quadratic penalty function $Q(x; \mu)$, the approximate minimizers x_k of $Q(x; \mu_k)$ do not satisfy the feasibility conditions $c_i(x) = 0, i \in E$. Instead, we have $c_i(x_k) \approx -\gamma_i^* / \mu_k, \forall i \in E$.

Consider the augmented Lagrangian function

$$L_A(x, \lambda; \mu) \triangleq f(x) - \sum_{i \in E} \lambda_i c_i(x) + \frac{\mu}{2} \sum_{i \in E} c_i^2(x)$$

which is a combination of the Lagrangian with the quadratic penalty concept.

The unconstrained minimizer x_k of $L_A(x, \lambda^k; \mu_k)$ for some λ^k (at iteration k of the upcoming algorithm) satisfies the first-order conditions of Thm 2-2:

$$0 \approx \nabla_x L_A(x_k, \lambda^k; \mu_k) = \nabla f(x_k) - \sum_{i \in E} \{\lambda_i^k - \mu_k c_i(x_k)\} \nabla c_i(x_k)$$

$$= \nabla f(x_k) - \sum_{i \in E} \lambda_i^* \nabla c_i(x_k)$$

where $\lambda_i^* \approx \lambda_i^k - \mu_k c_i(x_k), \forall i \in E$.

Rearranging the terms, we get $c_i(x_k) \approx -\frac{1}{\mu_k} (\lambda_i^k - \lambda_i^{*k})$ $\forall i \in E$.

The formula above also suggests a recursion to update our Lagrange multipliers:

$$\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(x_k) \quad \forall i \in E.$$

Framework 17.3 (Augmented Lagrangian Method: Equality const.)

Given $\mu_0 > 0$, tolerance $\tau_0 > 0$, starting points x_0^S, λ^0 ;

for $k=0, 1, 2, \dots$

Find an approximate minimizer x_k of $L_A(x, \lambda^k, \mu_k)$,
starting at x_k^S , and terminating when $\|\nabla_x L_A(x_k, \lambda^k, \mu_k)\| \leq \tau_k$

if a convergence test is satisfied

stop with approximate solution x_k ;

end (if)

Update Lagrange multipliers: $\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(x_k)$, $\forall i \in E$

Choose new penalty parameter $\mu_{k+1} \geq \mu_k$;

Set starting point for the next iteration: $x_{k+1}^S = x_k$;

Select tolerance τ_{k+1} ;

end (for)

The convergence of this method can be assured without increasing μ indefinitely. Consequently, ill conditioning is less of a problem.

Properties of the Augmented Lagrangian

Theorem 17.5 Let x^* be a local solution of the equality constrained problem at which LICQ is satisfied (i.e. the gradients $\nabla c_i(x^*)$ form a linearly independent set), and the second-order sufficiency conditions in Theorem 17.6 are satisfied for λ^* . Then $\exists \bar{\mu}$, a threshold, such that $\forall \mu \geq \bar{\mu}$, x^* is a strict local minimizer of $L_A(x, \lambda^*; \mu)$. If μ sufficiently large, that is $\nabla_x L_A(x^*, \lambda^*; \mu) = 0$, $\nabla_{xx}^2 L_A(x^*, \lambda^*; \mu) > 0$.

Proof: See pages 517-518 in the book.

This theorem shows that when we know λ^* , the solution x^* is a strict minimizer of $L_A(x; \lambda^*; \mu)$ for all sufficiently large μ . In practice, if we can get a reasonable estimate of λ^* , even for moderate μ , one can approximate x^* .

This justifies framework 17.3 above.

Theorem 17.6 Suppose that the assumptions of Theorem 17.5 are satisfied at x^* and λ^* , and let $\bar{\mu}$ be chosen as in that theorem. Then $\exists \delta > 0, \epsilon > 0, M > 0$ the following claims hold:

- $\forall \lambda^k$ and μ_k with $\|\lambda^k - \lambda^*\| \leq \mu_k \delta$, $\mu_k \geq \bar{\mu}$, $\min_x L_A(x, \lambda^k; \mu_k)$ s.t. $\|x - x^*\| \leq \epsilon$ has a unique solution x_k . Moreover we have $\|x_k - x^*\| \leq \frac{M}{\mu_k} \|\lambda^k - \lambda^*\|$.
- $\forall \lambda^k$ and μ_k that satisfy the condition in (a), we have $\|\lambda^{k+1} - \lambda^*\| \leq \frac{M}{\mu_k} \|\lambda^k - \lambda^*\|$, where λ^{k+1} is updated as in framework 17.3.
- $\forall \lambda^k$ and μ_k that satisfy the condition in (a), the matrix $\nabla_{xx}^2 L_A(x_k, \lambda^k; \mu_k) > 0$ and $\nabla c_i(x), i \in \mathcal{E}$ are linearly independent.

Proof: See reference given in the book.

This theorem deals with the more realistic situation $\lambda^k \neq \lambda^*$ and provides the conditions in which x_k lies close to x^* with error bounds in part (a). In part (b), we see that we can increase the accuracy of our hyperplane multiplier estimate by using a large μ_k . Part (c) shows that at iteration k , the problem has a strict solution.

Practical Augmented Lagrangian Methods

Bound Constrained Formulation: Given the general nonlinear program $\min_x f(x)$ s.t. $c_i(x) = 0, i \in E; c_i(x) \geq 0, i \in I$, we can convert it to a problem with equality constraints and bound constraints by introducing slack variables s_i :

$$c_i(x) - s_i = 0, s_i \geq 0 \quad \forall i \in I.$$

Bound constraints of the form $l \leq x \leq u$ need not be transformed. Incorporating the slack variables into x :

$$\min_{x \in \mathbb{R}^n} f(x) \text{ s.t. } c_i(x) = 0, i = 1, 2, \dots, m \quad l \leq x \leq u.$$

Some entries of l and u might be $-\infty$ and $+\infty$. Some entries of l and u might be $-\infty$ and $+\infty$. The Lagrangian augmented with the equality constraints is:

$$\mathcal{L}_A(x, \lambda; \mu) = f(x) - \sum_{i=1}^m \lambda_i c_i(x) + \frac{\mu}{2} \sum_{i=1}^m c_i^2(x).$$

Then we solve $\min_x \mathcal{L}_A(x, \lambda; \mu)$ s.t. $l \leq x \leq u$, in an iterative and recursive fashion.

The (nonlinear) gradient projection approach can be applied here. The KKT condition for the solution x is:

$$x - P(x - \gamma \nabla \mathcal{L}_A(x, \lambda; \mu), l, u) = 0.$$

Alg. 17.4 Bound-Constrained Lagrangian Method

Choose an initial point x_0 and initial multipliers λ^0 ;

Choose convergence tolerances γ_* , w_* ;

Set $M_0 = 10$, $\mu_0 = 1/M_0$, $\eta_0 = 1/\mu_0^{0.1}$;

for $k = 0, 1, 2, \dots$

Find an approximate solution x_k to $\min_x L_A(x, \lambda, \mu)$ s.t.

$$\ell \leq x \leq u,$$

 such that $\|x_k - P(x_k - \nabla_x L_A(x_k, \lambda^k, \mu_k), \ell, u)\| \leq w_k$;

if $\|c(x_k)\| \leq \eta_k$ (then test for convergence)

{ if $\|c(x_k)\| \leq \gamma_*$ and $\|x_k - P(x_k - \nabla_x L_A(x_k, \lambda^k, \mu_k), \ell, u)\| \leq w_*$
 STOP with approximate solution x_k ;

end(if)

(otherwise update multipliers and tighter tolerances)

$$\lambda^{k+1} = \lambda^k - \mu_k c(x_k); \quad \mu_{k+1} = M_k$$

$$\eta_{k+1} = \eta_k / \mu_{k+1}^{0.9}; \quad w_{k+1} = w_k / \mu_{k+1}$$

{ else (increase penalty and tighter tolerances)

$$\lambda^{k+1} = \lambda^k; \quad \mu_{k+1}^* = 10\mu_k$$

$$\eta_{k+1} = 1/\mu_{k+1}^{0.1}; \quad w_{k+1} = 1/\mu_{k+1}$$

end(if)

end(for)

when
constraint
violation
level
is
acceptable

{ otherwise

In the gradient projection method, a quadratic model with a trust region can be used:

$$\min_d \frac{1}{2} d^T [\nabla_{xx}^2 L(x_k, \lambda^k) + M_k A_k^T A_k] d + \nabla_x L(x_k, \lambda^k, M_k) d$$

s.t. $d \leq x_k + d \leq u$, $\|d\|_\infty \leq \Delta$.

(trust region)

where $A_k = A(x_k)$ is the constraint gradient matrix.

The trust region can be expressed as $-\Delta e \leq d \leq \Delta e$ where $e = (1, -1, 1)^T$. The Hessian can be approximated as in quasi-Newton methods.

Linearly Constrained Formulation: Here, we will generate a step by minimizing the augmented Lagrangian subject to linearizations of the constraints. The subproblem of each iteration takes the form

$$\min_x F_k(x) \text{ s.t. } c(x_k) + A_k(x - x_k) = 0, \quad l \leq x \leq u.$$

Here, we can set $F_k(x) = f(x) - \sum_{i=1}^m \lambda_i^k \bar{c}_i^k(x)$

where $\bar{c}_i^k(x) = c_i(x) - (c_i(x_k) + \nabla c_i(x_k)^T (x - x_k))$.

Alternatively, $F_k(x) = f(x) - \sum_{i=1}^m \lambda_i^k \bar{c}_i^k(x) + \frac{\mu}{2} \sum_{i=1}^m [\bar{c}_i^k(x)]^2$.

In practice, the latter usually works better (converges from remote starting points).

Unconstrained Formulation: Consider a nonlinear program with inequality constraints only. Let feasible set

$$F(x) \triangleq \max_{\lambda \geq 0} \left(f(x) - \sum_{i \in \mathcal{I}} \lambda_i c_i(x) \right) = \begin{cases} f(x) & \text{if } x \in \mathbb{R} \\ \infty & \text{o.w.} \end{cases}$$

Then we can solve $\min_{x \in \mathbb{R}^n} F(x)$ as an unconstrained problem. To see this consider the case when $x \notin \mathbb{R}$. Then $\exists i \ni c_i(x) < 0$, choose $\lambda_i > 0$ arbitrarily large while setting $\lambda_j = 0$ for all $j \neq i$. Then we can make $F(x) \rightarrow \infty$ by $\lambda_i \rightarrow \infty$.

If $x \in \mathbb{R}$, then $c_i(x) \geq 0 \forall i \in \mathcal{I}$, so the max is attained for $\lambda = 0$ and $F(x) = f(x)$.

$$\therefore \min_{x \in \mathbb{R}^n} F(x) \equiv \min_{x \in \mathbb{R}^n} f(x)$$

This F is not smooth but we can instead use

$$\hat{F}(x; \lambda^k; \mu_k) = \max_{\lambda \geq 0} \left(f(x) - \sum_{i \in \mathcal{I}} \lambda_i c_i(x) - \frac{1}{2\mu_k} \sum_{i \in \mathcal{I}} (\lambda_i - \lambda_i^k)^2 \right)$$

The last term $(\lambda_i - \lambda_i^k)^2$ encourages the new solution to stay proximal to the previous estimate of λ .

This problem is separable and quadratic in λ so the solution is: $\lambda_i = \begin{cases} 0 & \text{if } -c_i(x) + \lambda_i^k / \mu_k \leq 0 \\ \lambda_i^k - \mu_k c_i(x) & \text{o.w.} \end{cases}$

Substituting these, we get

$$\hat{F}(x; \lambda^k, \mu_k) = f(x) + \sum_{i \in I} \Psi(c_i(x), \lambda_i^k; \mu_k)$$

where $\Psi(t, \sigma; \mu) \triangleq \begin{cases} -\sigma t + \frac{\mu}{2}t^2, & \text{if } t - \sigma/\mu \leq 0 \\ -\sigma^2/(2\mu), & \text{o.w.} \end{cases}$

We can minimize \hat{F} w.r.t x at each step while getting λ from the expression above. Here F plays the role of L_A (in Framework 17.3) and we extend that concept to inequality constraints.

Chapter 18: Sequential Quadratic Programming

Here we consider active set methods for nonlinear programs.

Local SQP Method

Consider $\min f(x)$ s.t. $c(x) = 0$,

where $f: \mathbb{R}^n \rightarrow \mathbb{R}$ and $c: \mathbb{R}^n \rightarrow \mathbb{R}^m$ are smooth functions.

At the current iterate x_k , we will model the equality constrained problem by a quadratic programming subproblem and use its solution to get x_{k+1} .

The Lagrangian is $\mathcal{L}(x, \lambda) = f(x) - \lambda^T c(x)$. Let $A(x)$

be the Jacobian of $c(x)$: $A(x) \triangleq [\nabla c_1(x) \quad \nabla c_2(x) \cdots \nabla c_m(x)]$.

The KKT conditions can be written as

$$F(x, \lambda) = \begin{Bmatrix} \nabla f(x) - A(x)^T \lambda \\ c(x) \end{Bmatrix} = 0.$$

Any solution (x^*, λ^*) for which $A(x^*)$ has full rank satisfies these KKT conditions. We can solve these nonlinear equations by using Newton's method. The Jacobian of $F(x, \lambda)$ is

$$F'(x, \lambda) = \begin{Bmatrix} \nabla_x^2 \mathcal{L}(x, \lambda) & -A(x)^T \\ A(x) & 0 \end{Bmatrix}$$

which gives the Newton step $\begin{Bmatrix} x_{k+1} \\ \lambda_{k+1} \end{Bmatrix} = \begin{Bmatrix} x_k \\ \lambda_k \end{Bmatrix} + \begin{Bmatrix} p_k \\ \lambda_k \end{Bmatrix}$

$$\text{is } \begin{Bmatrix} \nabla_x^2 \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{Bmatrix} \begin{Bmatrix} p_k \\ \lambda_k \end{Bmatrix} = \begin{Bmatrix} -\nabla f_k + A_k^T \lambda_k \\ -c_k \end{Bmatrix}.$$

\rightarrow KKT matrix needs to be nonsingular for a well defined step.

Assumptions 18.1

- a) The constraint Jacobian $A(x)$ has full rank.
- b) The matrix $\nabla_{xx}^2 \mathcal{L}(x, \lambda)$ is positive definite on the tangent space of the constraints, that is, $d^T \nabla_{xx}^2 \mathcal{L}(x, \lambda) d > 0 \quad \forall d \neq 0 \Rightarrow A(x)d = 0$.

Assumption (a) is the linear independence constraint qualification.
 Assumption (b) is guaranteeing that close to the optimum
 the second-order sufficient condition is satisfied.

SQP Framework The Newton step given above can be viewed in an alternative framework. Suppose that at (x_k, λ_k) we model the equality constrained problem using the following quadratic program:

$$\begin{aligned} \min_P f_k + \nabla f_k^T P + \frac{1}{2} P^T \nabla_{xx}^2 \mathcal{L}_k P \\ \text{s.t. } A_k P + c_k = 0 \end{aligned}$$

If Assumptions 18.1 hold, this problem has a unique solution (P_k, λ_k) that satisfies

$$\nabla_{xx}^2 \mathcal{L}_k P_k + \nabla f_k - A_k^T \lambda_k = 0$$

$$A_k P_k + c_k = 0$$

or compactly
 (after subtracting
 $A_k^T \lambda_k$ from both sides)

$$\begin{bmatrix} \nabla_{xx}^2 \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} P_k \\ \lambda_k \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -c_k \end{bmatrix}$$

Solving this yields $\lambda_{k+1} = l_k$ and p_k is the Newton step that solves the model quadratic program.

Alg. 18.1 Local SQP Algorithm

Choose an initial pair (x_0, λ_0) ; set $k \leftarrow 0$;

Repeat until a convergence test is satisfied

Evaluate f_k , ∇f_k , $\nabla_{xx}^2 \mathcal{L}_k$, c_k , and A_k ;

Solve the model quadratic program to get p_k & λ_{k+1}

Set $x_{k+1} \leftarrow x_k + p_k$ and $\lambda_{k+1} \leftarrow l_k$;

end (repeat)

Inequality Constraints

$$\min f(x) \text{ s.t. } c_i(x) = 0 \quad i \in E \\ c_i(x) \geq 0 \quad i \in I$$

The model quadratic problem is:

$$\min_p f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p$$

$$\text{s.t. } \nabla c_i(x_k)^T p + c_i(x_k) = 0 \quad i \in E$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0 \quad i \in I$$

This is a standard quadratic program and we discussed algorithms for tackling it in chapter 16.

The new iterate is $(x_k + p_k, \lambda_{k+1})$.

Thm 18.1 Suppose that x^* is a local solution (372) of the inequality constrained nonlinear program at which the KKT conditions are satisfied for some λ^* . Suppose the LICQ conditions are satisfied (Defn 12.4), too that the linear independence constraint qualification (LICQ), (Defn 12.5) the strict complementarity condition, and the second-order sufficient conditions (Thm 12.6) hold at (x^*, λ^*) . Then if (x_k, λ_k) is sufficiently close to (x^*, λ^*) , there is a local solution of the subproblem (~~inequality~~ constrained model quadratic program) whose active set A_k is the same as the active set $A(x^*)$ of the original nonlinear program at x^* .

Handling Inconsistent Linearizations

The linearizations in the model quadratic program might lead to inconsistencies which results in an infeasible model problem. To overcome this, we formulate the nonlinear program as the l_1 penalty problem (for some μ)

$$\begin{aligned} \min_{x, v, w, t} \quad & f(x) + \mu \sum_{i \in E} (v_i + w_i) + \mu \sum_{i \in I} t_i \\ \text{s.t.} \quad & c_i^-(x) = v_i - w_i \quad i \in E \\ & c_i^-(x) \geq -t_i \quad i \in I \\ & v, w, t \geq 0 \end{aligned}$$

The quadratic subproblem associated with this is always feasible. As discussed in chapter 17, if the original nonlinear program has a solution x^* , then x^* is a solution of the subproblem above. If the original nonlinear program does not have a feasible solution, then for large enough μ , the subproblem above determines a stationary point of the infeasibility measure.

Full Quasi-Newton Approximations

In some cases (e.g. large scale problems), $\nabla_{xx}^2 \mathcal{L}(x_k)$ might be expensive to compute. We can use quasi-Newton approximations as usual. With

$$s_k \stackrel{\Delta}{=} x_{k+1} - x_k ; \quad y_k \stackrel{\Delta}{=} \nabla_x \mathcal{L}(x_{k+1}, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_k)$$

we can employ BFGS or SR1 formulas as before. This approach assumes that the objective is the Lagrangian for a fixed λ -value.

If $\nabla_{xx}^2 \mathcal{L} > 0$, the approach will work normally as in unconstrained optimization.

If $\nabla_x^2 f$ contains negative eigenvalues, then approximating it with $B > 0$ will be problematic. BFGS requires that $s_k^T y_k > 0$, but this may not hold even close to the solution. In this case, we could skip the BFGS update if $s_k^T y_k < \theta s_k^T B_k s_k$ where $\theta > 0$ (e.g. $\theta \approx 10^{-2}$). In general, this strategy might not work reliably.

Procedure 18.2 (Damped BFGS updating)

Given a symmetric and positive definite B_k ;

Define s_k and y_k as given above and set:

$$r_k = \theta_k y_k + (1 - \theta_k) B_k s_k$$

$$\text{where } \theta_k = \begin{cases} 1 & \text{if } s_k^T y_k \geq \delta s_k^T B_k s_k \\ (0.8 s_k^T B_k s_k) / (s_k^T B_k s_k - s_k^T y_k) & \text{o.w.} \end{cases}$$

$$\text{Update } B_{k+1}: B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{r_k r_k^T}{s_k^T r_k}$$

This procedure uses the BFGS update formula but replaces y_k by r_k . It guarantees that $B_{k+1} > 0$ since when $\theta_k \neq 1$ we have $s_k^T r_k = 0.2 s_k^T B_k s_k > 0$.

Also $\theta_k = 0 \Rightarrow B_{k+1} = B_k$ and $\theta_k = 1 \Rightarrow B_{k+1} = \text{usual BFGS}$ possibly indefinite. Here, r_k interpolates the line segment between B_k and B_{BFGS} using $\theta_k \in (0, 1)$.

Reduced-Hessian Quasi-Newton Approximations

Recall the KKT system for the equality constrained problem:

$$\begin{bmatrix} \nabla_{xx}^2 \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \lambda_{kn} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -c_k \end{bmatrix}$$

We have $A_k p_k = -c_k$ and the part of p_k in the range of A_k^T is completely determined by this;

$\nabla_{xx}^2 \mathcal{L}_k$ affects only the part of p_k in the orthogonal subspace. Let Y_k and Z_k be matrices whose columns span the range space of A_k^T and the null space of A_k , respectively. Let $p_k = Y_k p_Y + Z_k p_Z$. Then

$$(A_k Y_k) p_Y = -c_k$$

$$(Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k) p_Z = -Z_k^T \nabla_{xx}^2 \mathcal{L}_k Y_k p_Y - Z_k^T \nabla f_k$$

$$\text{Also, } (A_k Y_k)^T \lambda_{kn} = Y_k^T (\nabla f_k + \nabla_{xx}^2 \mathcal{L}_k p_k).$$

If we choose $Y_k = A_k^T$ (a valid choice when A_k has full row rank), we get $\hat{\lambda}_{kn} = (A_k A_k^T)^{-1} A_k \nabla f_k$. These are called the least-squares multipliers because

$$\hat{\lambda}_{kn} = \arg \min_{\lambda} \| \nabla_{xx}^2 \mathcal{L}(x_n, \lambda) \|_2^2 = \| \nabla f_k - A_k^T \lambda \|_2^2.$$

We can also use the following approximation:

$$(z_k^T \nabla_{xx}^2 L_k z_k) P_z = -z_k^T \nabla f_k$$

by removing the cross term $z_k^T \nabla_{xx}^2 L_k Y_k P_y$. This allows us to only approximate the matrix $z_k^T \nabla_{xx}^2 L_k z_k$ using quasi-Newton methods. This is justified because $P_y \rightarrow 0$ (normal component) faster than $P_z \rightarrow 0$ (tangential component) and the cross term is negligible near the solution.

Now, how do we approximate $z_k^T \nabla_{xx}^2 L_k z_k$?

Suppose we just took a step $\alpha_k P_k = x_{k+1} - x_k = \alpha_k z_k P_z + \alpha_k Y_k P_y$. By Taylor's Theorem ($\nabla_{xx}^2 L_{k+1} \stackrel{\Delta}{=} \nabla_{xx}^2 L(x_{k+1}, \lambda_{k+1})$)

$$\nabla_{xx}^2 L_{k+1} \alpha_k P_k \approx \nabla_x L(x_k + \alpha_k P_k, \lambda_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}).$$

Pre multiplying by z_k^T :

$$\begin{aligned} z_k^T \nabla_{xx}^2 L_{k+1} z_k \alpha_k P_k &\approx -z_k^T \nabla_{xx}^2 L_{k+1} Y_k \alpha_k P_y \\ &\quad + z_k^T \{ \nabla_x L(x_k + \alpha_k P_k, \lambda_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}) \} \end{aligned}$$

Dropping the cross term $z_k^T \nabla_{xx}^2 L_{k+1} Y_k \alpha_k P_y$ (as before) we get the secant eqn: $M_{k+1} S_k = Y_k$ where $S_k = \alpha_k P_z$ and $Y_k = z_k^T \{ \nabla_x L(x_k + \alpha_k P_k, \lambda_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}) \}$.

We can then apply BFGS or SR1 updates to M_k and obtain positive definite reduced-Hessian models with higher likelihood even far from the solution.

Merit Functions

In SQP, a merit function can be used to decide if a trial step should be accepted or not. For line search, this can determine the acceptable step lengths; for trust region methods it determines if the step is acceptable and it helps adjust the region radius.

We convert inequality constraints to equalities:

$$\bar{c}(x, s) = c(x) - s = 0 \quad \& \quad s \geq 0$$

using slack variables. Therefore, we can consider the case where all constraints are equalities w.l.o.g.

The ℓ_1 merit function is $\phi_i(x; \mu) = f(x) + \mu \|c(x)\|_1$. In line search, $\alpha_k p_k$ will be accepted if sufficient decrease condition holds ($\eta \in (0, 1)$):

$$\phi_i(x_k + \alpha_k p_k; \mu_k) \leq \phi_i(x_k, \mu_k) + \eta \alpha_k D(\phi_i(x_k; \mu); p_k)$$

where $D(\phi_i(x_k; \mu); p_k)$ denotes the directional derivative of ϕ_i along p_k .

Theorem 18.2 Let P_k and λ_{k+1} be generated by the SQP iteration:

$$\begin{bmatrix} \nabla_{xx}^2 L - A_k^T \\ A_k \quad 0 \end{bmatrix} \begin{bmatrix} P_k \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -c_k \end{bmatrix}.$$

Then the directional derivative of ϕ_1 along P_k satisfies

$$D(\phi_1(x_k; \mu); P_k) = \nabla f_k^T P_k - \mu \|c_k\|_1.$$

$$\text{Moreover, } D(\phi_1(x_k; \mu); P_k) \leq -P_k^T \nabla_{xx}^2 L_k P_k - (\mu - \|\lambda_{k+1}\|_\infty) \|c_k\|_1.$$

Proof: See pg 541 in the book.

This theorem shows that P_k will be a descent direction for ϕ_1 if $P_k \neq 0$, $\nabla_{xx}^2 L_k > 0$, and $\mu > \|\lambda_{k+1}\|_\infty$.

(Using arguments ~~we~~ seen before, we can show that we only need $z_k^T \nabla_{xx}^2 L_k z_k > 0$).

The penalty parameter can be increased at every iteration such that $\mu > \|\lambda_{k+1}\|_\infty$ is satisfied with some margin. In practice, this might lead to poor performance at times.

Alternatively, we can require that

$$D(\phi_1(x_k; \mu); P_k) = \nabla f_k^T P_k - \mu \|c_k\|_1 \leq -\rho \mu \|c_k\|_1,$$

for some $\rho \in (0, 1)$. This means, we need to have

$\mu \geq \frac{\nabla f_k^T p_k}{(1-\rho) \|c_{k+1}\|_1}$. Since this choice is not dependent on the Lagrange multiplier estimates, it may perform better in practice.

Yet another approach to selecting μ that is suitable for both line search and trust region methods is to consider a quadratic model

$$\varphi_\mu(p) = f_k + \nabla f_k^T p + \frac{\sigma}{2} p^T \nabla_{xx}^2 L_k p + \mu m(p)$$

where $m(p) = \|c_k + A_k p\|_1$ and $\sigma = \begin{cases} 1 & \text{if } p_k^T \nabla_{xx}^2 L_k p_k > 0 \\ 0 & \text{otherwise.} \end{cases}$

Choosing $\mu \geq \frac{\nabla f_k^T p_k + \frac{\sigma}{2} p_k^T \nabla_{xx}^2 L_k p_k}{(1-\rho) \|c_{k+1}\|_1}$ for some $p \neq 0$ ensures that $\varphi_\mu(0) - \varphi_\mu(p_k) \geq \rho(m(0) - m(p_k))$ which measures the quality of the model or checks for sufficient decrease depending on the approach.

If μ_k satisfies the inequality above, we can set $\mu_{k+1} = \mu_k$; otherwise we choose $\mu_{k+1} > \mu_k$ to satisfy this inequality.

If μ satisfies this inequality and σ is selected as specified, $D_L \phi_1(x_k; \mu); p_k \leq -\rho \mu \|c_{k+1}\|_1$, so that p_k is a descent direction for ϕ_1 .

Second-Order Correction

We may need a higher order correction term to overcome the effects of the Moretto effect. Suppose that the SQP method has computed a step \bar{p}_k . If this causes ϕ_k to increase, our linear constraint approximations might have been inaccurate. We could resolve the subproblem by replacing linearized constraints with quadratic ones:

$$\begin{array}{ll} \min_{\bar{p}} & f_k + \nabla f_k^T \bar{p} + \frac{1}{2} \bar{p}^T Q_{xx} \bar{p} \\ \text{s.t.} & c_i(x_k) + \nabla c_i(x_k)^T \bar{p} + \frac{1}{2} \bar{p}^T Q_i^2 c_i(x_k) \bar{p} = 0 \end{array} \quad i \in \mathcal{E}$$

This might be difficult to solve, so ~~we can't~~ $i \in \mathcal{I}$
evaluating $c_i(x_k + \bar{p}_k)$ $\forall i \in \mathcal{E} \cup \mathcal{I}$, we can use Taylor's theorem:

$$c_i(x_k + \bar{p}_k) \approx c_i(x_k) + \nabla c_i(x_k)^T \bar{p}_k + \frac{1}{2} \bar{p}_k^T Q_i^2 c_i(x_k) \bar{p}_k$$

and also use the approximation $\bar{p}^T Q^2 c_i(x_k) \bar{p} \approx \bar{p}_k^T Q^2 c_i(x_k) \bar{p}_k$
Then the ^{approximated} subproblem becomes

$$\begin{array}{ll} \min_{\bar{p}} & \nabla f_k^T \bar{p} + \frac{1}{2} \bar{p}^T Q_{xx} \bar{p} \\ \text{s.t.} & \nabla c_i(x_k)^T \bar{p} + d_i = 0 \end{array} \quad i \in \mathcal{E}$$

where $d_i = c_i(x_k + \bar{p}_k) - \nabla c_i(x_k)^T \bar{p}_k \quad \forall i \in \mathcal{E} \cup \mathcal{I}$.

Alg. 18.3 Line search SQP Algorithm

Choose $\eta \in (0, 0.5)$, $\tau \in (0, 1)$, (x_0, λ_0) ;

Evaluate f_0 , ∇f_0 , c_0 , A_0 ;

If using quasi-Newton approx. choose sym. $B_0 > 0$; otherwise compute $\nabla_{xx}^2 f_0$.

repeat until convergence test is satisfied

Compute P_k by solving the quadratic subproblem; let
 $\hat{\lambda}$ be the corresponding multiplier.

$$\text{Set } P_k \leftarrow \hat{\lambda} - \lambda_k; \text{ choose } \mu_k \geq \frac{\nabla f_k^T P_k + \frac{\alpha}{2} P_k^T \nabla_{xx}^2 f_k P_k}{((-\rho) \|c_k\|_1)} \quad |_{\delta=1}$$

$$\text{Set } \lambda_k \leftarrow 1;$$

$$\text{while } \phi_1(x_k + \lambda_k P_k; \mu_k) > \phi_1(x_k; \mu_k) + \eta \lambda_k D_1(\phi(x_k; \mu_k); P_k)$$

$$\text{Reset } \lambda_k \leftarrow \tau_2 \lambda_k \text{ for some } \tau_2 \in (0, \tau];$$

end (while)

$$\text{Set } x_{k+1} \leftarrow x_k + \lambda_k P_k \text{ and } \lambda_{k+1} \leftarrow \lambda_k + \lambda_k P_k;$$

$$\text{Evaluate } f_{k+1}, \nabla f_{k+1}, c_{k+1}, A_{k+1} \text{ (and } \nabla_{xx}^2 f_{k+1});$$

If a quasi-Newton approximation is used, set

$$s_k \leftarrow \lambda_k P_k, g_k \leftarrow \nabla f(x_{k+1}) - \nabla f(x_k)$$

and obtain B_{k+1} by updating B_k .

end (repeat)

- * Initialize the active set of each subproblem to the active set of the previous subproblem solution.
- * Incorporate second-order correction and use as needed.

Trust-Region SQP Methods

We can add a trust-region constraint to the subproblem:

$$\begin{aligned} \min_P f_k + \frac{1}{2} h_k^T P + \frac{1}{2} P^T Q_{xx}^{-2} L_k P \\ \text{s.t. } Q_{ci}(x_k)^T P + c_i(x_k) = 0 \quad i \in E \\ \quad \quad \quad Q_{ci}(x_k)^T P + c_i(x_k) \geq 0 \quad i \in I \\ \quad \|P\|_F \leq \Delta_k \end{aligned}$$

Notice that if the trust-region does not intersect with the linearized feasible set, this problem will become infeasible. We cannot simply increase Δ_k until the problem becomes feasible since this might require very large trust regions inappropriately.

Since linearized constraints are only approximations, we can instead try to improve the feasibility at each iteration.

A Relaxation Method for Equality-Constrained Optimization

At x_k , we compute the SQP step by solving

$$\min_P f_k + \frac{1}{2} h_k^T P + \frac{1}{2} P^T Q_{xx}^{-2} L_k P \quad \text{s.t. } A_k P + c_k = r_k \\ \|P\|_F \leq \Delta_k$$

Here, r_k is called the relaxation vector. We try to choose r_k as small as possible s.t. the subproblem is feasible for some reduced value of Δ_k .

for this purpose, we solve

$$\min_{\mathbf{v}} \|\mathbf{A}_k \mathbf{v} + \mathbf{c}_k\|_2^2 \text{ s.t. } \|\mathbf{v}\|_2 \leq 0.8 D_k$$

Using the solution \mathbf{v}_k of this, we set $r_k = \mathbf{A}_k \mathbf{v}_k + \mathbf{c}_k$.

Now using r_k , we solve the relaxed subproblem to get the step p_k and set $x_{kt+1} = x_{kt} + p_k$; get γ_{kt+1} from the least-squares formula mentioned earlier.

In practice, inexact solutions for both of these problems are sufficient and much easier to obtain.

The auxiliary subproblem can be approximately solved using the dogleg method; if $\mathbf{A}_k^T \mathbf{A}_k$ is singular, then the pseudo inverse can be used to set the Newton step as $p^B = -\mathbf{A}_k^T (\mathbf{A}_k \mathbf{A}_k^T)^{-1} \mathbf{c}_k$.

The subproblem for p_k can be approximately solved using the projected conjugate gradient method (Alg 16.2). Use the ℓ_2 -smooth function as merit to set step acceptability threshold: $p_k = \frac{\text{Act. Red.}_k}{\text{Pred. Red.}_k} = \frac{\phi_2(x_{kt+1}) - \phi_2(x_{kt} + p_k)}{g_{\mu}(0) - g_{\mu}(p_k)}$

where $\phi_x(x; \mu) = f(x) + \mu \|c(x)\|_2$ and

$$g_\mu(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 f_k p + \mu n(p),$$

$$\text{and } n(p) = \|c_k + A_p p\|_2.$$

Alg. 18.4 Byrd-Omojokun Trust-Region SQP Method

Choose constants $\epsilon > 0$ and $\eta, \gamma \in (0, 1)$;

Choose x_0 and $\Delta_0 > 0$;

for $k=0, 1, 2, \dots$
Compute $f_k, c_k, \nabla f_k, A_k$; $\hat{\lambda}_{k+1} = (A_k A_k^T)^{-1} A_k^T f_k$;

if $\|\nabla f_k - A_k^T \hat{\lambda}_k\|_\infty \leq \epsilon$ and $\|c_k\|_\infty \leq \epsilon$

stop with approx soln. x_k ;

Solve ~~the auxiliary~~ subproblem for v_k and get r_k .

Compute $\nabla_{xx}^2 f_k$ or a quasi-Newton approx.

Compute P_k by applying the projected conj. grad.

to the subproblem;

choose $\mu_k \geq \frac{\nabla f_k^T P_k + \frac{\eta}{2} P_k^T \nabla_{xx}^2 f_k P_k}{(1-\rho_k) \|c_k\|_1}$, $\beta_k = \frac{\text{ActRes}_k}{\text{PredRes}_k}$

if $P_k > \eta$

Set $x_{k+1} = x_k + P_k$; choose $\Delta_{k+1} \geq \Delta_k$;

else

set $x_{k+1} = x_k$; choose $\Delta_{k+1} \leq \gamma \|P_k\|$

end (if)

end (for)

Sequential L₁ Quadratic Programming (SLQP)

(385)

$$\min_{\rho} q_\mu(\rho) \triangleq f_k + \nabla f_k^T \rho + \frac{1}{2} \rho^T Q_{xx}^2 \Delta_k \rho + \mu \sum_{i \in E} [c_i(x_k) + \nabla c_i(x_k)^T \rho] \\ + \mu \sum_{i \in I} [c_i(x_k) + \nabla c_i(x_k)^T \rho]$$

$$\text{s.t. } \|\rho\|_\infty \leq \Delta_k.$$

for some μ . Here $Eg^j \triangleq \max\{0, -g_j\}$ as before.
Introducing slack variables v, w, t , we can rewrite as:

$$\begin{array}{ll} \min_{\rho, v, w, t} & f_k + \nabla f_k^T \rho + \frac{1}{2} \rho^T Q_{xx}^2 \Delta_k \rho + \mu \sum_{i \in E} (v_i + w_i) + \mu \sum_{i \in I} t_i \\ \text{s.t. } & \nabla c_i(x_k)^T \rho + c_i(x_k) = v_i - w_i \quad i \in E \\ & \quad \quad \quad \geq -t_i \quad i \in I \end{array}$$

$$v, w, t \geq 0$$

$$\|\rho\|_\infty \leq \Delta_k$$

The constraints of this latter problem are always consistent and it can be solved using standard QP algorithms. For step acceptance, we can use the ℓ_1 -norm: $\phi_i(x, \mu) = f(x) + \mu \sum_{i \in E} |c_i(x)| + \mu \sum_{i \in I} [c_i(x)]^-$.

Sequential Linear Quadratic Programming (SLQP)

The SLQP methods we discussed try to solve a general inequality constrained QP at each iteration. Thus it is cumbersome if the size of the problem is large.

In SLOP, the step is obtained in two stages.

Both stages handle large problems well.

- 1) A linear program (LP) to identify the working set, \mathcal{W} .
- 2) An equality constrained DP (EDP) using \mathcal{W} .

The steps are combined...

$$(LP) \quad \min_p f_k + \nabla f_k^T p \quad \text{s.t.} \quad \begin{aligned} c_i(x_k) + \nabla c_i(x_k)^T p &= 0 \quad i \in \mathcal{E} \\ c_i(x_k) + \nabla c_i(x_k)^T p &\geq 0 \quad i \in \mathcal{I} \\ \|p\|_\infty &\leq \Delta_k^{LP} \end{aligned}$$

This is solved using ℓ_1 -penalty.

$$\min_p l_\mu(p) \stackrel{\Delta}{=} f_k + \nabla f_k^T p + \mu \left[\sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T p| + \sum_{i \in \mathcal{I}} [c_i(x_k) + \nabla c_i(x_k)^T p] \right]$$

Introducing slack variables, this can be converted to a standard LP with orthant inequality constraints and equality constraints.

* Let p^{LP} be the solution of this problem, which can be obtained using the simplex method.

$$\text{Let } A_k(p^{LP}) = \{i \in \mathcal{E} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{LP} = 0\}$$

$$\cup \{i \in \mathcal{I} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{LP} = 0\}$$

be the corresponding active set.

Similarly, the set of violated constraints are

$$\mathcal{V}_k(\rho^{LP}) = \left\{ i \in \mathcal{E} \mid c_i(x_k) + \gamma_{c_i}(x_k)^T \rho^{LP} \neq 0 \right\}$$

$$\cup \left\{ i \in \mathcal{I} \mid c_i(x_k) + \gamma_{c_i}(x_k)^T \rho^{LP} < 0 \right\}.$$

Then \mathcal{W}_k is some linearly indep. subset of $\mathcal{A}_k(\rho^{LP})$.

Let $\rho^c = \alpha^{LP} \rho^{LP}$, $\alpha^{LP} \in (0, 1]$ be the Cauchy step-

for ϕ_1 . Given \mathcal{W}_k , we solve

$$\min_{\rho} f_k + \frac{1}{2} \rho^T \nabla_{xx}^2 L_k \rho + \left(\nabla f_k + M_k \sum_{i \in \mathcal{W}_k} \gamma_i \gamma_{c_i}(x_k) \right)^T \rho$$

$$\text{s.t. } c_i(x_k) + \gamma_{c_i}(x_k)^T \rho = 0 \quad i \in (\mathcal{E} \cup \mathcal{I}) \cap \mathcal{W}_k$$

$$\|\rho\|_2 \leq \Delta_k$$

where γ_i is the sign of the i th violated constraint.

* Let ρ^ϑ be the solution to this problem, for instance, obtained using the projected gradient procedure in Alg 16.2 (and using Alg. 7.2 to handle the first regular constraint).

$$\text{The total step is } \rho_k = \rho^c + \alpha^\vartheta (\rho^\vartheta - \rho^c)$$

where $\alpha^\vartheta \in \{0, 1\}$ is selected to approximately minimize the ~~prob~~ SLQP model ϕ_M .

The multiplier estimates λ_k can be obtained using the least-squares estimates using \mathcal{W}_k and setting $\lambda_i \geq 0$ for $i \in \mathcal{I}$.

Alg. 18-5 Penalty Update and Step Computation

Given x_k , $M_{k-1} > 0$, $\Delta_k > 0$, $\epsilon_1, \epsilon_2 \in (0, 1)$.

Solve SLQP (with slack variables introduced)

with $\mu = M_{k-1}$ to get $P(M_{k-1})$

if $M_k(P(\mu_{k-1})) = 0$, set $\mu^+ \leftarrow M_{k-1}$;

else, compute P_∞ ;

if $M_k(P_\infty) = 0$, find $\mu^+ > M_{k-1}$ s.t. $M_k(P(\mu^+)) = 0$

else find $\mu^+ > M_{k-1}$ s.t. $M_k(0) - M_k(P(\mu^+)) \geq \epsilon_1 \{M_k(0) - M_k(P_\infty)\}$

end (if)

end (if)

Increase μ^+ if necessary to satisfy

$$\frac{g_{\mu^+}(0)}{g_{\mu^+}} - \frac{g_{\mu^+}(P(\mu^+))}{g_{\mu^+}} \geq \epsilon_2 \mu^+ \{M_k(0) - M_k(P(\mu^+))\}$$

Set $\mu_k \leftarrow \mu^+$. and $P_k \leftarrow P(\mu^+)$;

This procedure has nice practical performance and global convergence guarantees. It tries to choose μ small enough to avoid over-emphasizing the constraints in the merit function. Here

$$M_k(\rho) = \sum_{i \in E} |c_i(x_k) + \nabla c_i(x_k)^T \rho| + \sum_{i \in I} [c_i(x_k) + \nabla c_i(x_k)^T \rho]^+$$

$$g_{\mu}(\rho) = f_k + \nabla f_k^T \rho + \frac{1}{2} \rho^T \nabla^2 L_k \rho + \mu M_k(\rho).$$

Nonlinear Gradient Projection

Consider $\min f(x)$ s.t. $l \leq x \leq u$. Let

$$g_k(x) = f_k + \nabla f_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T B_k(x - x_k)$$

where $B_k > 0$ approximates $\nabla^2 f_k$. We can use the gradient projection QP method to solve

$$\min g_k(x) \text{ s.t. } l \leq x \leq u.$$

$P_k = \hat{x} - x_k$ and $x_{k+1} = x_k + \alpha_k P_k$ where

$$f(x_k + \alpha_k P_k) \leq f(x_k) + \gamma \alpha_k \nabla f_k^T P_k$$

for some $\gamma \in (0, 1)$ as usual in the search.

$$\begin{aligned} \text{Notice that } f_k &= g_k(x_k) \geq g_k(x^c) \geq g_k(\hat{x}) \\ &= f_k + \nabla f_k^T P_k + \frac{1}{2} P_k^T B_k P_k \end{aligned}$$

so $\nabla f_k^T P_k < 0$ since $B_k > 0$. Consequently, if \hat{x} , the solution to the model problem performs better than the Cauchy point p^c , the update is a descent direction for the original objective.

Trust-regions can be incorporated. Consider

$$\min g_k(x) \text{ s.t. } l \leq x \leq u \text{ and } \|x - x_k\|_2 \leq \Delta_k$$

Equivalently, $\min g_k(x)$ s.t. $\max(l, x_k - \Delta_k e) \leq x \leq \min(u, x_k + \Delta_k e)$ where $e = (1, -1, 1)^T$. Gradient projection QP method can be used here as usual.

Convergence Analysis

Consider an SQP method that computes a search direction P_k by solving the quadratic subproblem

$$\begin{array}{ll} \min_{P} & f_k + \nabla f_k^T P + \frac{1}{2} P^T Q_k P \\ \text{s.t.} & \nabla c_i(x_k)^T P + c_i(x_k) = 0 \quad i \in \mathcal{E} \\ & \geq 0 \quad i \in \mathcal{I} \end{array}$$

and the Hessian $Q_k = \nabla^2 f_k$ is approximated by $B_k \geq 0$. We set $x_{k+1} = x_k + \lambda_k P_k$ and λ_k is obtained via backtracking line search starting from unity and

terminating when $\phi_1(x_k + \lambda_k P_k; \mu) \leq \phi_1(x_k; \mu) - \eta \lambda_k \phi_2(\mu) - \rho(\mu)$
where $\eta \in (0, 1)$ where ϕ_1 and ϕ_2 are defined as in SLQP.

We assume that μ is fixed for all k and is sufficiently large; we also assume that each quadratic subproblem is feasible and yields a bounded P_k .

Thm 18-3 Suppose that the SQP process above is used.

Suppose $\{x_k\}$ and $\{x_k + P_k\}$ are contained in a closed, bounded, convex region of \mathbb{R}^n , in which f and c_i have cont. first derivatives. Suppose B_k and λ_k are bounded and $\mu \geq \|x_k\|_\infty + p \cdot \lambda_k$ for some $p > 0$. Then all limit points of $\{x_k\}$ are KKT points of the nonlinear program $\min f(x)$ s.t. $c_i(x) \leq 0$ $i \in \mathcal{I}$.

The global convergence assurance is nice but the requirement for $\{x_k\}$ to stay in a bounded set means B_k is not allowed to become ill conditioned. Similarly the constraint Jacobians J_k cannot become ill cond.

Rate of Convergence

We will see the conditions that guarantee superlinear local convergence.

Assumptions 18-2 The point x^* is a local soln. of $\min f(x)$ s.t. $c(x)=0$ at which the following conditions hold.

- The functions f and c are twice differentiable in some $N(x^*)$ with Lipschitz continuous second derivatives.
- The linear indep. constraint gradients (Axiom 12.6) holds at x^* . This implies that the KKT conditions are satisfied for some λ^* .
- The second-order sufficient conditions (Thm 12.6) hold at (x^*, λ^*) .

Thm 18-4 Suppose that Assumptions 18-2 hold.

Then, if (x_0, λ_0) is sufficiently close to (x^*, λ^*) , the pairs (x_k, λ_k) generated by Alg. 18-1 converge quadratically to (x^*, λ^*) .

Alg. 18-1 uses Newton steps to solve a quadratic program and the proof follows from Thm 11-2 and Assumptions 18-2.

$$\text{Let } \nabla_{xx}^2 \mathcal{L}_* \stackrel{\Delta}{=} \nabla_{xx}^2 \mathcal{L}(x^*, \lambda^*).$$

Thm 18-5 Suppose that Assumptions 18-2 hold and the iterates x_k generated by Alg. 18-1 with guess λ_k approximate Hessians B_k converge to x^* . Then x_k converges superlinearly iff the Hessian approx. B_k sets for

$$\lim_{k \rightarrow \infty} \frac{\|P_k(B_k - \nabla_{xx}^2 \mathcal{L}_*)(x_{k+1} - x_k)\|}{\|x_{k+1} - x_k\|} = 0.$$

where $P_k = I - A_k^T [A_k A_k^T]^{-1} = Z_k Z_k^T$.

* So if $P_k B_k \approx P_k \nabla_{xx}^2 \mathcal{L}_*$, convergence of quasi-Newton methods discussed will be superlinear.

Thm 18.6 Suppose that Assumptions 18-2 hold. 383

Assume that $\mathbf{Q}_{xx}^2 \mathbf{L}_x$ and \mathbf{B}_0 are symmetric, pos. defn.
 If $\|\mathbf{x}_0 - \mathbf{x}^*\|$ and $\|\mathbf{B}_0 - \mathbf{Q}_{xx}^2 \mathbf{L}_x\|$ are sufficiently small,
 then $\{\mathbf{x}_k\}$ generated by Alg. 18-1 with BFGS Hessian
 approximations \mathbf{B}_k satisfy the limit in Thm 18-5;
 therefore, $\{\mathbf{x}_k\}$ converges superlinearly to \mathbf{x}^* .
 (Q-superlinear for normal, R-superlinear for damped).

Thm 18.7 Suppose that Assumption 18-2 holds and \mathbf{B}_k
 are bounded. Assume also that $\{\mathbf{x}_k\}$ generated by
 Alg. 18-1 with approx Hessians \mathbf{B}_k converges to \mathbf{x}^* ,
 and the limit in Thm 18-5 holds for \mathbf{B}_k with some
~~modification~~

modification:

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{P}_k(\mathbf{B}_k - \mathbf{Q}_{xx}^2 \mathbf{L}_x) \mathbf{P}_k(\mathbf{x}_{k+1} - \mathbf{x}_k)\|}{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|} = 0.$$

Then $\{\mathbf{x}_k\} \rightarrow \mathbf{x}^*$ two-step superlinearly:

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{x}_{k+2} - \mathbf{x}^*\|}{\|\mathbf{x}_k - \mathbf{x}^*\|} = 0.$$

Here $\mathbf{B}_k = \mathbf{Z}_k \mathbf{M}_k \mathbf{Z}_k^T$ (reduced Hessen quasi-Newton)
 where $\mathbf{B}_k \approx \mathbf{P}_k \mathbf{Q}_{xx}^2 \mathbf{L}_k \mathbf{P}_k$.

Chapter 19: Interior-Point Methods for Nonlinear Programming

These are also referred to as barrier methods. The main ideas for the algorithms will carry over directly from LP, but new challenges will be encountered: nonconvexity, updating of the barrier parameters in the presence of nonlinearities, guaranteed progress towards a solution. The problem in standard formulation is:

$$\min_{x, s} f(x) \text{ s.t. } c_E(x) = 0 \\ c_I(x) - s = 0$$

Here $f: \mathbb{R}^n \rightarrow \mathbb{R}$, $c_E: \mathbb{R}^l \rightarrow \mathbb{R}$ and $c_I: \mathbb{R}^m \rightarrow \mathbb{R}$.

Continuation interpretation: The KKT conditions for the nonlinear program given above are (with $\mu=0$):

$$\nabla f(x) - A_E^T(x) y - A_I^T(x) z = 0$$

$$S z - M e = 0$$

$$c_E(x) = 0$$

$$c_I(x) - s = 0$$

$$s \geq 0, z \geq 0$$

Here, $A_E(x)$ and $A_I(x)$ are the Jacobian matrices of the functions c_E and c_I , y and z are their Lagrange multipliers.

$$S \triangleq \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix}, Z \triangleq \begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix}, e \triangleq \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Although the solution is obtained when $\mu=0$, by setting $\mu>0$, we enforce $s>0$ and $z>0$. The continuation (homotopy) approach approximately solves the KKT conditions above for a sequence $\{\mu_k\} \rightarrow 0$, while maintaining $s, z>0$. By requiring the iterates to decrease a merit function or be acceptable to a filter, the iterations are likely to converge to a minimizer and not simply any KKT point.

The trajectory $(x(\mu), s(\mu), z(\mu), \mu)$ $\xrightarrow{\mu \rightarrow 0} (x^*, s^*, z^*)$
is called the primal-dual central path.

Barrier Interpretation: Consider the barrier problem

$$\min_{x,s} f(x) - \mu \sum_{i=1}^m \log s_i \quad \text{s.t. } c_E(x) = 0 \\ c_I(x) - s = 0$$

where $\mu>0$. Since the minimization of the barrier term $-\mu \sum_i \log s_i$ prevents s from becoming too close to zero, we do not need to enforce $s>0$ when solving the problem iteratively.

The barrier approach finds approximate solutions for the barrier problem for a positive sequence $\{\mu_k\} \rightarrow 0$. The KKT conditions for the barrier problem are:

$$\nabla f(x) - A_E^T(x)y - A_I^T(x)z = 0$$

$$-\mu s^{-1}e + z = 0$$

$$c_E(x) = 0$$

$$c_I(x) - s = 0$$

These two KKT conditions differ only in the way 2nd equation is expressed (and the missing orthant inequalities in the barrier method). Since $s > 0$, we have $s > 0$ and multiplying the 2nd eqn above by s : $Sz - \mu e = 0$, which is the same as the previous KKT conditions.

A Basic Interior Point Algorithm

Applying Newton's algorithm to the KKT conditions of the continuation interpretation, we obtain

$$\begin{bmatrix} \nabla_{xx}^2 L & 0 & -A_E^T(x) & -A_I^T(x) \\ 0 & z & 0 & S \\ A_E(x) & 0 & 0 & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_s \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \nabla f(x) - A_E^T(x)y \\ -A_I^T(x)z \\ Sz - \mu e \\ c_E(x) \\ c_I(x) - s \end{bmatrix}$$

where $L(x, s, y, z) = f(x) - y^T c_E(x) - z^T (c_I(x) - s)$.

The system of (Newton) linear equations is called the primal-dual system. After the step $p = \begin{bmatrix} p_x \\ p_z \end{bmatrix}$ is determined, we compute (x^+, y^+, z^+) :

$$x^+ = x + \alpha_{\max}^{(s)} p_x, \quad s^+ = s + \alpha_{\max}^{(s)} p_s$$

$$y^+ = y + \alpha_{\max}^{(z)} p_y, \quad z^+ = z + \alpha_{\max}^{(z)} p_z \quad \begin{array}{l} \text{Typically} \\ \alpha = 0.995 \end{array}$$

where $\alpha_{\max}^{(s)} = \max \left\{ \alpha \in (0, 1) : s + \alpha p_s \geq (1 - \alpha) s \right\} \downarrow^{(\alpha > 0)}$

$$\alpha_{\max}^{(z)} = \max \left\{ \alpha \in (0, 1) : z + \alpha p_z \geq (1 - \alpha) z \right\} \text{ with } \alpha \in (0, 1).$$

These step length selection rules are called the fraction to the boundary rules, and they prevent s and z from approaching the lower bound of 0 too quickly. This simple iteration forms the basis of modern interior-point methods.

One problem is the selection of barrier parameters in $\{\mu_k\} \rightarrow 0$. This sequence could be stepwise constant over the index k , or it could be changing every iteration. The primal-dual system matrix remains nonsingular as the iteration converges to a solution that satisfies the second-order sufficiency and strict complementarity conditions.

If x^* is a solution for which strict complementarity holds (exactly one of λ_i^* and $c_i(x^*)$ is zero $\forall i \in I \cap A(x^*)$), $\forall i$ either s_i or z_i remains bounded away from 0 as the iterates approach x^* . This ensures $\{0 \geq 0\}$, the 2nd block row in the matrix is full row rank, making the whole matrix full rank.

$$\text{Let } E(x, s, y, z; \mu) \triangleq \max \left\{ \| \nabla f(x) - A_E(x)^T y - A_I(x)^T z \|_1, \| s - y \|_1, \| c_E(x) \|_1, \| c_I(x) - z \|_1 \right\}$$

be an error function for some μ and vector norm $\| \cdot \|_1$.

Alg. 19.1 Basic Interior-Point Algorithm

Choose $x_0, s_0 > 0$ and compute $y_0, z_0 > 0$.

Select $\mu_0 > 0$ and $\sigma, \tau \in (0, 1)$; set $k \leftarrow 0$.

Repeat until a stopping test for the nonlinear program is satisfied

repeat until $E(x_k, s_k, y_k, z_k; \mu_k) \leq \mu_k$

Solve the primal-dual system to get ρ .

Compute α_s^{new} , α_z^{new} using the fraction to boundary rule.

Compute $(x_{k+1}, s_{k+1}, y_{k+1}, z_{k+1})$ using the update rule.

Set $\mu_{k+1} \leftarrow \mu_k$ and $k \leftarrow k+1$;

end

choose $\mu_k \in (0, \sigma \mu_0)$;

end

Thm 18.1 Suppose that Alg. 18.1 generates a sequence $\{x_k\}$ and that $\{\mu_k\} \rightarrow 0$ (i.e. the alg. does not loop infinitely in the inner repeat). Suppose that f and c are continuously differentiable. Then all limit points \hat{x} of $\{x_k\}$ are feasible. Furthermore, if any limit point \hat{x} satisfies LICQ, then the first order optimality (KKT) conditions of the nonlinear program hold at \hat{x} .

(skip) Proof: Only considering the case with inequality constraints.

$$\min_{x, s} f(x) \quad \text{s.t. } c(x) - s = 0, \quad s \geq 0 \quad (\text{Here } c = c_I).$$

Let \hat{x} be a limit point of $\{x_k\}$ and let $\{x_{k_\ell}\}$ be a convergent subsequence: $\{x_{k_\ell}\} \rightarrow \hat{x}$.

Since $\mu_k \rightarrow 0$, $E_k \rightarrow 0$. So $(c_{k_\ell} - s_{k_\ell}) \rightarrow 0$. By continuity of c , this fact implies that $\hat{c} \triangleq c(\hat{x}) \geq 0$; i.e. \hat{x} is feasible, and $s_{k_\ell} \rightarrow \hat{s} = \hat{c}$.

Now suppose LICQ holds at \hat{x} and consider $\lambda = \{\lambda_i : \hat{c}_i = 0\}$. For $i \notin I$ we have $\hat{c}_i > 0$ and $\hat{s}_i > 0$ and thus by the complementarity condition ($s_i - \mu_i = 0$) we have $\{z_{k_\ell}\}_i \rightarrow 0$.

From this and $\nabla f_{k_\ell} - A_{k_\ell}^T z_{k_\ell} \rightarrow 0$, we conclude that

$$\nabla f_{k_\ell} - \sum_{i \in I} \{z_{k_\ell}\}_i \nabla c_i(x_{k_\ell}) \rightarrow 0.$$

By constraint qualification, the vectors $\{\nabla \hat{c}_i : i \in A\}$ are linearly independent. Hence, by continuity of $\nabla f(\cdot)$ and $\nabla c_{ii}(\cdot)$,

$i \in A$, the positive sequence $\{z_{k,i}\} \rightarrow \vec{z} \geq 0$.

Taking the limit, we have

$$\nabla f(\vec{x}) = \sum_i \vec{z}_i \nabla c_i(\vec{x}).$$

We also have $\vec{c}^T \vec{z} = 0$. D.

Algorithmic Development

Since Thm 18.1 assures us that algorithms with approximate barrier problem solutions converge, we can detail these algorithms.

The primal-dual system can be rewritten as:

$$\begin{bmatrix} \Sigma & 0 & A_E^T(x) & A_I^T(x) \\ 0 & \Sigma & 0 & -I \\ A_E(x) & 0 & 0 & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_s \\ -p_y \\ -p_z \end{bmatrix} = \begin{bmatrix} \nabla f(x) + A_E^T(x)y \\ \Sigma - M^{-1}e \\ c_E(x) \\ c_I(x) - s \end{bmatrix}$$

where $\Sigma = S^{-1} \vec{z}$. Since here the system matrix is symmetric, we can use robust algorithms for solving the linear system.

The main computational load at each iteration of the interior-point method is the solution of this system.

Solving the Primal-Dual System

If we eliminate p_s in the Newton step system using the second equation $(\bar{z}p_s + \bar{s}p_z) = S\bar{z} - M\bar{e}$, thus system of equations reduces to

$$\begin{bmatrix} \bar{\lambda}_{xx}^2 I & A_E^\top(x) & A_I^\top(x) \\ A_E(x) & 0 & 0 \\ A_I(x) & 0 & -\Sigma^{-1} \end{bmatrix} \begin{bmatrix} p_x \\ -p_y \\ -p_e \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A_E^\top(\bar{y}) - A_I^\top(\bar{w}) \\ c_E(\bar{x}) \\ c_I(x) - \mu \bar{z}^\top \bar{e} \end{bmatrix}.$$

This can be solved using symmetric indefinite factorization:

$$PKP = LBL^T$$

where P is a permutation matrix, L is unit lower triangular, B is block diagonal with 1×1 or 2×2 blocks.

If we further reduce this system by removing p_z the matrix becomes:

$$\begin{bmatrix} (\bar{\lambda}_{xx}^2 I + A_I^\top \Sigma A_I) & A_E^\top(x) \\ A_E(x) & 0 \end{bmatrix}$$

This can be useful in reducing the system size when the number of inequality constraints is large. The draw-back is that in the $(1,1)$ block, $+A_I^\top \Sigma A_I$ might significantly reduce sparsity.

Updating the Barrier Parameter

The sequence $\{\mu_k\}$ must converge to 0 so that, in the limit, we recover the solution of the nonlinear program.

If $\mu_k \rightarrow 0$ too slowly then unnecessarily large number of iterations will be required for convergence. If $\mu_k \rightarrow 0$ too fast, some slack variables s or multipliers z may approach 0 prematurely, thus slowing progress.

Fiacco-McCormick Rule: The barrier parameter is

kept fixed until the perturbed KKT conditions are satisfied to some accuracy. Then the barrier parameter is decreased by the rule $\mu_{k+1} = \sigma_k \mu_k$

where $\sigma_k \in (0, 1)$. We can use a larger σ_k when recent iterations do not make much progress and a smaller σ_k when they progress well. Near the solution, letting $\sigma_k \rightarrow 0$ and $\tau \rightarrow 1$ in step length selection for Newton iteration, we can achieve superlinear convergence.

Adaptive Strategies: These are more successful in difficult situations. They vary μ at every iteration depending on progress. They mostly depend on the use of complementarity (see framework 14.1 in LP) and use

$$\mu_{\text{act}} = \sigma_k \frac{s_k^T z_k}{m}$$

which allows μ to reflect the scale of the problem. One popular choice for σ_k is given by

$$\sigma_k = 0.1 \min \left(0.05 \frac{1 - \xi_k}{\xi_k}, 2 \right)^3 \text{ where } \xi_k = \frac{\min_i \{ s_{ik} \} \{ z_{ik} \}}{(s_k)^T z_k / m}$$

$\{v\}_i$ denotes the i^{th} entry of vector v in the above--- when $\xi_k \approx 1$ (all individual products are near the average), the deviation from the near-complementarity is low), μ is decreased aggressively.

Alternatively, using a predictor-offer scaling direction $(\Delta x^{\text{off}}, \Delta s^{\text{off}}, \Delta y^{\text{off}}, \Delta z^{\text{off}})$ as in LP by setting $\mu=0$ and probing this search direction with α_p^{off} and α_z^{off} , largest step lengths before violating $(s, z) \geq 0$.

Then $\mu_{\text{eff}} = (s_k + \alpha_s^{\text{off}} \Delta s^{\text{off}})^T (z_k + \alpha_z^{\text{off}} \Delta z^{\text{off}}) / m$ and

$$\sigma_k = \left(\frac{\mu_{\text{eff}}}{s_k^T z_k / m} \right)^3 \text{ as in LP can be used.}$$

Handling Necessity and Singularity

The primal-dual system only seeks KKT points so they can lead to convergence towards any stationary point. For the primal-dual system, the step ρ is a descent direction of the matrix $\begin{bmatrix} \nabla_{xx}^2 L & 0 \\ 0 & \Sigma \end{bmatrix}$ is positive definite on the null space of the constraint matrix $\begin{bmatrix} A_E(x) & 0 \\ A_I(x) & -I \end{bmatrix}$. From ~~Thm.~~ 16.3, we know that this pos. definiteness holds if the inertia of the primal-dual matrix (symmetric version) is given by $(l+m, l+m, 0)$; i.e. l ($l+m$) positive, m ($l+m$) negative, and no zero eigenvalues, where l & m are the number of equality and inequality constraints.

The following modified primal-dual matrix has the required inertia (with $s > 0$ sufficiently large to make $\nabla_{xx}^2 L + sI > 0$ and $\gamma > 0$ guarding against a rank-deficient A_E):

$$K \triangleq \begin{bmatrix} \nabla_{xx}^2 L + sI & 0 & A_E(x)^T & A_I(x)^T \\ 0 & \Sigma & 0 & -I \\ A_E(x) & 0 & -\gamma I & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix}$$

An algorithm for selecting s and γ is as follows:

Alg B.1 (Appendix B: Inertia Correction & Regularization)

Given the current μ , $\eta > 0$ and $\beta < 1$ (η, β constants) and s_{old} from the previous interior-point iteration.

Factor the matrix λ above with $\delta = \theta = 0$.

if K is nonsingular and its inertia is $(1m, 1m, 0)$

Compute the primal-dual step; stop;

if K has 0 eigenvalues

$$\text{Set } \delta \leftarrow 10^{-8} \eta \mu^\beta$$

if $s_{\text{old}} = 0$

$$\text{Set } \delta \leftarrow 10^{-4}$$

else

$$\text{Set } \delta \leftarrow s_{\text{old}}/2$$

repeat

Factor the modified matrix K

if the inertia is $(1m, 1m, 0)$

$$\text{Set } s_{\text{old}} \leftarrow \delta$$

Compute the primal-dual step; stop

else

$$\text{Set } \delta \leftarrow 10\delta$$

end (repeat)

* This is a heuristic algorithm that tries to avoid excessively large δ which dominates second order derivative $\nabla^2 L$ information. All constants are arbitrary.

Step Acceptance: Merit Functions & Filters

Consider the following exact merit function:

$$\phi_v(x, s) = f(x) - \mu \sum_{i=1}^m \log s_i + v \|c_E(x)\| + v \|c_I(x) - s\|$$

where $\|\cdot\|$ is some vector norm (e.g. ℓ_1 or ℓ_2 norm).

In line search, after P is computed and maximum step lengths have been determined, we perform a backtracking line search that computes the step lengths

$$\alpha_s \in (0, \alpha_s^{\max}], \quad \alpha_r \in (0, \alpha_r^{\max}]$$

providing sufficient decrease of the merit function or ensuring ~~the~~ acceptability by the filter. Then

$$x^+ = x + \alpha_s p_x, \quad s^+ = s + \alpha_s p_s; \quad y^+ = y + \alpha_r p_y, \quad z^+ = z + \alpha_r p_z$$

In the filter, the two objectives are the barrier function

$$f(x) - \mu \sum_{i=1}^m \log s_i \quad \text{and the constraint violators } \|c_E(x)\|, \|c_I(x) - s\|.$$

Quasi-Newton Approximations

We approximate $\nabla_x^2 L$ by some quasi-Newton model B .

For updating B using BFGS or SR1 we use the conditions

$$\Delta x = x^+ - x \quad \text{and} \quad \Delta l = \nabla_x L(x^+, s^+, y^+, z^+) - \nabla_x L(x, s^+, y^+, z^+).$$

To ensure that BFGS generates a pos. definite B , the damped update may be used.

Feasible Interior-Point Methods: We might require all iterates to remain feasible. If the current iterate satisfies $c_I(x) \geq 0$, then in the primal-dual iteration compute P , let $x^+ = x + p_x$ and redefine slacks: $s^+ \leftarrow c_I(x^+)$

not

and test if (x^*, s^*) is acceptable for merit ϕ .
 If so, accept this point as the new iterate, otherwise
 reject p and use a shorter trial step. The latter
 can be done by backtracking line search or trust-region
 radius reduction. The barrier-merit function defined above
 will reject steps that are infeasible or feasible but
 too close to the boundary.

A Line Search Interior-Point Method

Let $D\phi(x, s; p)$ be the directional derivative of ϕ_v at (x, s) .

Alg. 18.2 Line Search Interior-Point Algorithm

Choose x_0 and $s_0 > 0$; compute y_0 and $z_0 > 0$. If a quasi-Newton method is used choose $n \times n$ symmetric, pos-definite B_0 .
 Select initial $\mu > 0$ and parameters $\eta, \epsilon \in (0, 1)$, tolerances $\epsilon_\mu, \epsilon_{\text{TOL}}$.
 Set $k \leftarrow 0$.

repeat until $E(x_k, s_k, y_k, z_k; 0) \leq \epsilon_{\text{TOL}}$

 repeat until $E(x_k, s_k, y_k, z_k; \mu) \leq \epsilon_\mu$

 Compute the primal-dual direction $p = (p_x, p_s, p_y, p_z)$.

 Compute $\alpha_s^{\max}, \alpha_z^{\max}$ and set $p_w = (p_x, p_s)$;

 Compute step lengths α_s, α_z below $\alpha_s^{\max}, \alpha_z^{\max}$ and satisfying

$$\phi_v(x_k + \alpha_s p_x, s_k + \alpha_s p_s) \leq \phi_v(x_k, s_k) + \gamma \alpha_s D\phi_v(x_k, s_k; p_w);$$

 Compute $(x_{k+1}, s_{k+1}, y_{k+1}, z_{k+1})$ using these step lengths and directions;

 If a quasi-Newton approach is used update B_{k+1} .

 Set $k \leftarrow k+1$

end
 set $\mu \leftarrow \eta \mu$ and update ϵ_μ ; (Here, we can choose $\epsilon_\mu = \mu$
 as in Alg. 18.1).

end

A Trust-Region Interior-Point Method

408

The algorithm below generates steps that allow faster convergence than the line search method.

First, note that the barrier problem $\min_{x, s} f(x) - \mu \sum_{i=1}^m \log s_i$ is an equality constrained problem that can be solved using a sequential quadratic programming method with trust regions. The subproblem is

$$\min_{P_x, P_s} Q_f^T P_x + \frac{1}{2} P_x^T Q_{xx}^{-2} P_x - \mu e^T S^{-1} P_s + \frac{1}{2} P_s^T \Sigma P_s$$

$$\text{subject to } A_E(x) P_x + C_E(x) = r_E$$

$$A_I(x) P_x - P_s + (C_I(x) - s) = r_I$$

$$\| (P_x, S^{-1} P_s) \|_2 \leq \Delta$$

$$P_s \geq -cs \quad \text{where } \Sigma = S^{-1} \Sigma.$$

Ideally $r = (r_E, r_I) = 0$, but this might cause the linearized constraints to become incompatible so we choose r using the auxiliary problem described earlier (fig. 18.4).

Alg. 18.3 Trust-Region Algorithm for Barrier Problems

(for fixed μ)

Input parameters $\mu > 0$, $x_0, s_0 > 0$, $\epsilon_\mu, \Delta_0 > 0$.

Compute $y_0, z_0 > 0$. Set $k \leftarrow 0$.

repeat until $E(x_{k+1}, y_{k+1}, z_{k+1}; \mu) \leq \epsilon_\mu$

 Compute $P = (P_x, P_s)$ by solving the subproblem approximately.

 if P provides sufficient decrease in merit ϕ_P

 Set $x_{k+1} \leftarrow x_k + P_x$; $s_{k+1} \leftarrow s_k + P_s$;

 Compute new $y_{k+1}, z_{k+1} > 0$ and set $\Delta_{k+1} \geq \Delta_k$

 else define $x_{k+1} \leftarrow x_k$; $s_{k+1} \leftarrow s_k$ and set $\Delta_{k+1} \leq \Delta_k$

 end

 set $k \leftarrow k+1$

end (repeat)

Step Computation

Due to the nonlinear trust-region constraint, the subproblem is difficult to solve exactly. By a change of variables, we convert the trust region to a ball: $\tilde{p} = \begin{bmatrix} p_x \\ \tilde{p}_s \end{bmatrix} = \begin{bmatrix} p_x \\ s^{-1}p_s \end{bmatrix}$. Then

$$\min_{p_x, \tilde{p}_s} \nabla f^T p_x + \frac{1}{2} p_x^T \nabla_{xx}^2 p_x - M e^T \tilde{p}_s + \frac{1}{2} \tilde{p}_s^T S \Sigma S \tilde{p}_s$$

subject to $A_E(x) p_x + c_E(x) = r_E$

$$A_I(x) p_x - S \tilde{p}_s + (c_I(x) - s) = r_I$$

$$\|(p_x, \tilde{p}_s)\|_2 \leq \Delta$$

$$\tilde{p}_s \geq -\tau e$$

To compute r_E and r_I , we use the normal subproblem (Sec. 18.5)

$$\min_v \|A_E(x)v_x + c_E(x)\|_2^2 + \|A_I(x)v_x - Sv_s + (c_I(x) - s)\|_2^2$$

s.t. $\|(v_x, v_s)\|_2 \leq 0.8 \Delta$ and $v_s \geq -(\tau/2)e$.

We can solve this problem using dogleg and ignoring the second bound constraint on v_s . Then if the solution violates the bound, we find a ~~step~~ solution by backtracking.

Now, given r_E and r_I and $r_E = A_E(x)v_x + c_E(x)$

and $r_I = A_I(x)v_x - Sv_s + (c_I(x) - s)$, we can proceed to solving the subproblem. The subproblem is tackled by solving it using the three equality constraints ~~with~~ with the projected

CG iteration (Alg. 16.2) stopped using Stekhanov's rules: we monitor the satisfaction of the trust region constraint and stop if the boundary is reached, if negative curvature is detected or if an approximate solution is obtained. If the solution given by projected CG

does not satisfy the bounds, we backtrack until they are satisfied. After (p_x, \tilde{p}_s) is computed we recover p using the inverse of the change of variables.

Lagrange Multiplier Estimates and Step Acceptance

At (x, s) , we choose (y, z) to be the least-squares multipliers corresponding to the equality constraints of the subproblem: $\begin{bmatrix} y \\ z \end{bmatrix} = (\hat{A}\hat{A}^T)^{-1}\hat{A}\begin{bmatrix} \nabla f(x) \\ -\mu e \end{bmatrix}$ where $\hat{A} = \begin{cases} A_E(x) & 0 \\ A_I(x) & S \end{cases}$. If we don't get $z \geq 0$, then we set $z_i \leftarrow \min(10^{-3}, \mu/s_i)$ where μ/s_i is referred to as the i th primal multiplier estimate. As usual, the step p is accepted if $\text{ActRed}(p) \geq \eta \text{PredRed}(p)$, where $\text{ActRed}(p) = \phi_v(x, s) - \phi_v(x + p_x, s + p_s)$ and $\eta \in (0, 1)$ ($\eta \ll 1$) and $\text{PredRed}(p) = \varphi_v(0) - \varphi_v(p)$, where $\varphi_v(p) = \nabla f^T p_x + \frac{1}{2} p_x^T \nabla_{xx}^2 p_x - \mu e^T S^{-1} p_s + \frac{1}{2} p_s^T S^2 p_s + v m(p)$ with $m(p) = \left\| \begin{pmatrix} A_E(x)p_x + C_E(x) \\ A_I(x)p_s + C_I(x) - S \end{pmatrix} \right\|_2$. We require v to be large enough so that $\text{PredRed}(p) \geq p v(m(0) - m(p))$, for some $p \in (0, 1)$ in order to ensure that the model reduction is faster than constraint violation reduction.

Alg 18-4 Trust-Region Interior-Point Algorithm

Choose a value for the parameters $\eta > 0$, $\tau \in (0, 1)$, $\alpha \in (0, 1)$, and $\beta \in (0, 1)$. Select stopping tolerances ϵ_{in} and ϵ_{tol} . If a quasi-Newton approach is used, select $B_0 > 0$. Choose $\mu > 0$, $x_0, s_0 > 0$, and $\Delta_0 > 0$. Set $k \leftarrow 0$.

repeat until $E(x_k, s_k, y_k, z_k; 0) \leq \epsilon_{\text{tol}}$

repeat until $E(x_k, s_k, y_k, z_k; \mu) \leq \epsilon_{\text{in}}$

Compute Lagrange multipliers using the least squares approach.

Compute $\nabla_{xx}^2 L(x_k, s_k, y_k, z_k)$ or B_k and define $\Sigma_k = S_k^{-1} Z_k$.

Compute the normal step $v_k = (\nu_x, \nu_s)$.

Compute \tilde{p}_k using the projected CG method on the subproblem.

Obtain the total step p_k by inverse parameter change.

Update ν_k to satisfy $\text{PredRed}(p) \geq \rho \nu (m_0 - m(p))$.

Compute $\text{PredRed}_k(p_k)$ and $\text{ActRed}_k(p_k)$.

if $\text{ActRed}_k(p_k) \geq \eta \text{PredRed}_k(p_k)$

 Set $x_{k+1} \leftarrow x_k + p_x$, $s_{k+1} \leftarrow s_k + p_s$; choose $\Delta_{k+1} \geq \Delta_k$.

else

 Set $x_{k+1} = x_k$, $s_{k+1} = s_k$; choose $\Delta_{k+1} < \Delta_k$,

end if).

Set $k \leftarrow k+1$

end

set $\mu \leftarrow \sigma \mu$ and update ϵ_{in}

end.

If good steps are rejected due to the Morozov effect,
the second-order corrector can be employed selectively.

The Primal Log-BARRIER Method

Consider the inequality constrained nonlinear problem-

$$\min_x f(x) \text{ s.t. } c_i(x) \geq 0.$$

The log-barrier function is $P(x; \mu) = f(x) - \mu \sum_{i \in I} \log c_i(x)$.

Here $\mu > 0$ and the minimizers of $P(x; \mu)$, denoted by $x(\mu)$, approach a solution of the neg.-const. nonlin. problem as $\mu \rightarrow 0$ under certain conditions. The trajectory

$C_p \equiv \{x(\mu) / \mu > 0\}$ is called the primal central path.

Since $x(\mu)$ lies strictly in the feasible set,

we can use unconstrained optimizer techniques
by making sure that the iterates/estimates

never leave the feasible set.

Using $\nabla_x P(x; \mu) = \nabla f(x) - \sum_{i \in I} \frac{\mu}{c_i(x)} \nabla c_i(x)$, when

x is close to the minimizer $x(\mu)$ and μ is small,
from Thm 12.1, the optimal Lagrange multipliers are
estimated as $z_i^* \approx \mu/c_i(x)$, $i \in I$.

Framework 18.5 Unconstrained Primal Barrier Method

Given $\mu_0 > 0$, a sequence $\{\tau_k\}$ with $\tau_k \rightarrow 0$, x_0^s .

for $k = 0, 1, 2, \dots$

Find an approx min. x_k of $P(\cdot; \mu_k)$ starting at x_{k+1}^s and terminating when $\|\nabla P(x_k; \mu_k)\| \leq \tau_k$.

Compute $z_{i,k} = \mu_k / c_i(x_k)$.

if final convergence test not satisfied
stop with approx. soln. x_k .

Choose new penalty parameter $\mu_{k+1} < \mu_k$;

Choose new starting point x_{k+1}^s ;

end (for).

PRB method has serious drawbacks compared to
SQP and primal-dual interior-point methods. Most
importantly, minimizer of $P(\cdot; \mu_k)$ becomes increasingly
difficult. Specifically, the log-function prevents local
quadratic models from being accurate over long distances,
thus preventing Newton type methods from converging
fast. The ill conditioning of the Hessians also prevent
gradient methods from rapidly converging.
(see Ex. 18.1 in the book).

Global Convergence Properties

Line search: Due to the lack of coordination between step computation and imposition of the bounds, line search methods may fail or more commonly perform extremely poorly. (See example 18-2 in the book).

Specifically, if initialized to an unlucky region, they might converge to a nonoptimal infeasible point.

Trust-Region: Alg 18.4 has favorable global convergence properties. Let B_k denote $\nabla_{xx}^2 L_k$ or a quasi-Newton approximation to it. Let $h(x) = \|[\mathbf{c}(x)]^\top\|$ for an ineq.-const. problem. Note that $\nabla[h(x)^2] = 2A(x)\mathbf{c}(x)$.

A sequence $\{x_k\}$ is asymptotically feasible if $\mathbf{c}(x_k) \rightarrow 0$.

Thm 18.2. Suppose that Alg. 18.4 is applied to the barrier problem with μ fixed and the inner repeat loop executed with $\epsilon_B=0$. Suppose that $\{f_k\}$ is bounded below and $\{\nabla f_k\}, \{\mathbf{c}_k\}, \{A_k\}, \{B_k\}$ are bounded. Then one of the following three situations occurs:

- (i) $\{x_k\}$ is not asymptotically feasible.
In this case, the iterates approach a stationarity point of $h(x)$, meaning $A_k c_k \rightarrow 0$ and the penalty parameters $v_k \rightarrow \infty$.
 - (ii) $\{x_k\}$ is asymptotically feasible, but $\{(c_k, A_k)\}$ has a limit point (\bar{c}, \bar{A}) failing LICA.
In this case, the penalty parameter $v_k \rightarrow \infty$ as well. LICA.
 - (iii) $\{x_k\}$ is asymptotically feasible and all limit points of $\{(c_k, A_k)\}$ satisfy LICA. In this case, v_k is constant and $c_k > 0$ & large k and the stationarity conditions of the barrier problem are satisfied in the limit.
- * In outcome (i), there is no direction that improves feasibility to first order in the limit.
- * In (ii), we converge to a point that does not satisfy KKT and LICA does not hold.
- * In (iii), we get the most desirable outcome.
- Now, for Alg. 18.4, the full/complete interior-point method, we see that the following outcomes occur:

(416)

- * For some μ generated by the algorithm, either
 $\|c_k - s_k\| \leq \epsilon_\mu$ is never satisfied (statement)
cond. for minimizing $h(x)$ is satisfied in the limit),
or $(c_k - s_k) \rightarrow 0$ (the seq. $\{(c_k, A_k)\}$ has a limit
point (\bar{c}, \bar{A}) failing LICQ).
- * At each outer iteration of Alg 18.4, the inner
loop stop test is satisfied. Then all limit points
of the iteration sequence are feasible. Furthermore,
if any limit point \bar{x} satisfies the ~~LICQ~~, the
first-order necessary cond hold at \bar{x} for the
mug. const. problem.
- * Under some assumptions (Assumptions 18.1 on pg 581
of the book) superlinear convergence may be
achieved using primal-dual interior-point methods.
(For some suggestions on setting parameters in
Alg 18.2 to achieve this, see pg 592 in the book.)