|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Feasible Set | | | | Solutions that satisfy the constraints: feasible. Set of soln = S = . So solve min f(**x**) s.t. **x** S. Else: infeasible | | | | | | | | | | | | |
| Convex set | A set D is convex if for any 2 pts **x** and **y** in D, the line segment joining **x** and **y** also lies in D. i.e. **x**, **y** D **x** + (1-)**y** D  Notation: line segment [**x**, **y**] := {**x** + (1-)**y** D } | | | | | | | | | | | | | | | |
| E.g. Half Space: Given **a** , b , H = {**x** |**a**T**x** ≤ b} is a convex set  **x** H **a**T**x** ≤ b. **y** H (1 - )**a**T**y** ≤ (1 - )b. **x** + (1-)**y** = **a**T(**x** + (1 - )**y**) ≤ b convex set | | | | | | | | | | | | | | | E.g. d-Ball, w center **a**, radius b > 0. B(**a**, b) = { **x** | ≤ b } is convex |
| Proposition 3.1: If C1, C2, ..., Cm are convex sets in , then C = is also convex | | | | | | | | | | | | | | | |
| E.g. Polyhedron or polyhedron set {**x** |**A**T**x** ≤ **b**} = D = {**x** ≤ b1,..., **x** ≤ bn} = (intersection of half space) D is convex | | | | | | | | | | | | | | | |
| Convex fn | Let D be a convex set. Consider a fn f: D R.  f is convex if f(**x** + (1-)**y**) ≤ f(**x**) + (1-)f(**y**), **x**, **y** D, | | | | | | | f is strictly convex if f(**x** + (1-)**y**) < f(**x**) + (1-)f(**y**), distinct **x**, **y** D, . concave just flip sign: ≤ : ≥. < : > | | | | | | | | |
| Linear fn: f(**x**) = **a**T**x** + b, **a**, **x** . f(**x** + (1-)**y**) = **a**T(**x** + (1-)**y**) + b = (**a**T**x** + b) + (1-)(**a**T**y** + b) = f(**x**) + (1-)f(**y**) f is convex & concave | | | | | | | | | | | | | | | |
| l2 norm to a pt a, f(**x**) = f is convex | | | | | | Square fn: f(**x**) = **x**2 is convex | | | | | | Max of coordinates: f(**x**) = max{x1, ..., xn} is convex | | | |
| Cauchy-Schwarz Inequality: – ≤ **x**T**y** ≤ | | | | | | | triangle inequality: | | | | | | | | |
| Corollary 3.1: Let f1, ..., fk: D R be convex fns on a convex set D . Then f(**x**) = , where ai ≥ 0 i, is also a convex fn on D  If at least 1 of f­i is strictly convex on D, then f is strictly convex on D (i.e. LC of convex fns is convex) | | | | | | | | | | | | | | | |
| R/s btw sets & fns | | | | Proposition 3.5: Suppose D is convex. If f: D R is convex, then for any a , the sublevel set Sa := {**x** D| f(**x**) ≤ a} is convex  Level set: f(**x**) = a. Superlevel set: f(**x**) ≥ a | | | | | | | | | | | | |
| Half space: H = {**x** |**a**T**x** ≤ b}. **a**T**x** is convex. H is a sublevel set, hence is convex | | | | | | | | | | d-Ball: B(**a**, b) = { **x** | ≤ b } also convex | | |
| Gradient, convexity and optimum | | Suppose f: S is differentiable at **x**, then the gradient vector of f at **x** is the col vector  If f has cts gradient on S, we say it is C1 on S  Taylor's Theorem, 1st order: Suppose f: S is C1. Suppose [**x**, **y**] S, then **w** [**x**, **y**] s.t. f(**y**) = f(**x**) + (**y** - **x**)  Suppose **d** is non-zero, = 1, **d** is a direction vector. Then **x** + is moving from **x** in dir **d** and is rate of change in f when **x** moves from **w** along dir **d**.  f(**x** + ) = f(**x**) + = f(**x**) + . So **.** Hence = (since **w** **x** when )  At **x**, the vector is a normal vector to the level set. | | | | | | | | | | | | | | |
| f(**x**) = x1 - . | | | | Linear fn: f(**x**) = **b**T**x**. **b** | | | | Quadratic fn: f(**x**) = . Suppose A is symmetric. **Ax** | | | | | | |
| Stationary pt and optimality | | Thrm 3.1: Suppose that f(**x**) is C1 on an open convex set S in . Then | | | | | | | (a) the fn f is convex iff f(**x**) + (**y** - **x**) ≤ f(**y**) **x**, **y** S  (b) the fn f is strictly convex iff f(**x**) + (**y** - **x**) < f(**y**) **x** ≠ **y** S | | | | | | | |
| Corollary (Sufficiency): Suppose f(**x**) is C1 and convex on an open convex set S in . Suppose = **0**. Then **x**\* is an optimal soln for , **x** S. Suppose in addition that f(**x**) is strictly convex on S, then **x**\* is the unique optimal soln. | | | | | | | | | | | | | | |
| Thrm (Necessity): Suppose that f(**x**) is C1 on an open set S in and **x**\* is its min or max, then = 0 (note convexity not needed) | | | | | | | | | | | | | | |
| Positive definite matrices | | If A is real symmetric, is an eigenvalue of A w eigenvector **v**i if A**v**i = **v**i | | | | | | | | | | To find eigenvalues: find roots to det(I - A) = 0 | | | | |
| Let A be a real symmetric d x d matrix.  (a) A is positive semidefinite (PSD) if **x**TA**x** ≥ 0, **x** . Notation: A 0  (b) A is positive definite (PD) if **x**TA**x** > 0, **x** ≠ 0 . Notation: A 0  (c) A is negative semidefinite if **x**TA**x** ≤ 0, **x** . i.e. -A is PSD  (d) A is negative definite if **x**TA**x** < 0, **x** ≠ 0 . i.e. -A is PD  (e) A is indefinite if A is neither positive nor negative semidefinite | | | | | | | | | | Thrm 2.3: Let A be a real symmetric d x d matrix  (a) A is PSD iff every eigenvalue of A is nonnegative ( ≥ 0)  (b) A is PD iff every eigenvalue of A is positive ( > 0)  (c) A is NSD iff every eigenvalue of A is nonpositive ( ≤ 0)  (d) A is ND iff every eigenvalue of A is negative ( < 0)  (e) A is indefinite iff there is a +ve and -ve eigenvalue of A | | | | |
| Let A . The kth principal minor of A is the determinant of the kth principal submatrix of A, i.e. = det[a11], = det ...  Thrm: Suppose A is a symmetric matrix, then A is PD iff > 0 for all k = 1, ..., d | | | | | | | | | | | | | | |
| Hessian and convexity | | The Hessian of f at **x** is the d x d matrix, . If f has cts second order derivatives, f C2, then Hessian matrix is symmetric  Thrm 2.5: Suppose f: S is C2. Suppose line segment [**x**, **y**] S, then **w** [**x**, **y**] s.t. f(**y**) = f(**x**) + (**y** - **x**) + | | | | | | | | | | | | | | |
| Thrm 3.2: Suppose f(**x**) is C2 on an open convex set D in .  - The fn f is convex on D iff the Hessian matrix is PSD **x** D  - If is PD at each **x** D, then f is strictly convex on D  - The fn D is concave on D iff the is NSD at each **x** D | | | | | | | - If is ND at each **x** D, then f is strictly concave on D  - If is indefinite at some pt **x** D, then f is not a convex nor a concave fn on D (i.e. **x** where have +ve & -ve ) | | | | | | | |
| Local Optimum | | A point **x**\* S is a local minimzer of f(**x**) if an > 0 s.t. f(**x**) ≥ f(**x**\*) **x** S  = {**x**: < }, i.e. the ball centered at **x**\* w radius | | | | | | | A point **x**\* S is a global minimizer of f(**x**) if an > 0 s.t. f(**x**) ≥ f(**x**\*) **x** S.  Stationary pt **x**\* of f which is not local min nor local max = saddle pt | | | | | | | |
| Thrm 4.2: Let X be an open subset of Rn. Suppose f: X R is C1 in X. If **x**\* X is a local minimizer of f on X, then **x**\* is a stationary pt, i.e. = **0**. In addition, if f is C2, then must be PSD | | | | | | | | | | | | | | |
| Thrm 4.3: Let X be an open subset of Rn. Suppose f: X R is C2. If **x**\* X is a stationary pt, i.e. = **0**, and is PD, then **x**\* is a local minimizer (if is ND, then **x**\* is a local max) | | | | | | | | | | | | | | |
| So given a stationary pt **x**\*, check Hessian : 1) PD/all eigenvalues > 0 **x**\* is local min. 2) ND/all eigenvalues < 0 **x**\* is local max  3) Indefinite/eigenvalues > 0 & < 0 **x**\* is a saddle pt. 4) PSD or NSD inconclusive | | | | | | | | | | | | | | |
| page18image35460176Solve .  Case 1: f is convex  - local min = global min  Case 2: f is not convex  - global min also local min  - they are stationary points w PSD Hessian  - assuming they exist, we can go through all local mins  1) Find stationary pt **x**\* of f. 2) If f is convex, **x**\* is a minimizer. 3) If 0 **x**, f is convex. 4) To check PSD: eigenvalues/principal minors | | | | | | | | | | | | | | |
| E.g. | | For any > 0, - (since ball centered at 0), f(-) = < f(0) 0 is not a local min  Also,  , f() = > f(0) 0 is not a local max 0 is neither a local min or local max 0 is a saddle pt | | | | | | | | | | | | | | |
| Iterative Algos | To find stationary pt **x**\*: (**x**\*) = **0**  Mtd 1: Generate iterates **x**1, **x**2, ..., **x**n **x**\*  Mtd 2: Generate decreasing sets C1 C2 C3 ... Cn **x**\* | | | | | | | | | Stop algo at: 1) Budget: stop at k = T iteration, T is given  2) Tolerance: stop at ≤ Tol  3) Improvement: stop if ≤ Tol or f(**x**k) - f(**x**k+1) ≤ Tol | | | | | | |
| To assess algo: 1) Complexity (how many iteration needed). 2) Implementation (easy to code). 3) Update cost (cost of running 1 step of algo) | | | | | | | | | | | | | | | |
| Bisection mtd (Mtd 2) | | Suppose f' exists and is cts. Want to find f'(x) = 0. Suppose f'(a) and f'(b) have diff signs  Input: a0, b0, f', Tol. Output: ct so f'(ct) ≤ Tol  Interval length at next iteration is half. Accuracy of using midpoint: at least half of interval  |ak - bk| = 2-k|a0 - b0|. OR log|ak - bk| = -k log 2 + log|a0 - b0|  Linear convergence to x\*: log error decreases linearly | | | | | | | | | | | | | | t = 0; while |f'(ct)| > Tol do {  ct = 0.5at + 0.5bt; at+1 = at, bt+1 = bt;  if f'(ct)f'(at) < 0 then bt+1 = ct;  else at+1 = ct;  t = t+1 } |
| Golden Ratio (Mtd 2)  Note f need not be convex | | Suppose f is unimodal on [a,b]. So f is decreasing on [a, **x**\*] and increasing on [b, **x**\*]  Choose 2 pts, < . If f() < f(): **x**\* [a1, ]. If () ≥ f(): **x**\* [, b1].  We want . So .  Solving, = 0.382. 1 - = 0.618 = golden ratio (case for f() > f(); same for other)  Intervals shrink = 0.618 of previous interval.  General steps: 1) Find left and right boundary, so f is unimodal inside  2) Find left & right intervals of proportion 3) Use interval w larger interior boundary value  4) Repeat 2 and 3 until the interval length is lower than Tol  Length of kth iteration = |ak - bk| = k|a0 - b0| = 0.618k of original len  OR log|ak - bk| = klog + log|a0 - b0|So linear convergence to x\*. | | | | | | | | | | | | | Input: a0, b0, f, Tol  Output: (at + bt)/2 as an approximate soln  t = 0;  while bt - at > TOL do  = 0.618ai + 0.382bi;  = 0.382ai + 0.618bi;  [ai+1, bi+1] = [ai, bi];  if f() > f() then bi+1 = ;  else ai+1 = ;  t = i+1  Accuracy: 0.618k+1 \* original length ≤ error  OR 0.5\*0.618k \* original length ≤ error | |
| Univariate Newton's mtd (Mtd 1) | | | Suppose f' and f'' are available. Then . Trying to solve x where f'(x) = 0  Thrm: Suppose |x0 - x\*| is sufficiently small and f is C3 with f''(x\*) ≠ 0, then for a constant M, |xk+1 - x\*| ≤ M|xk - x\*|2, k  Therefore, error of xk+1 is square of error of xk quadratic convergence (only when |x0 - x\*| is small)  And . So digit of accuracy doubles w each iteration (if error of 1st = 10-2, then error of 2nd = 10-4, 3rd = 10-8, ...) | | | | | | | | | | | | | |
| For multivariate, suppose and are both available. Then . Trying to solve | | | | | | | | | | | | | |
| To see how algo is performing, can plot . Or can plot f(**x**k) against k | | | | | | | | | | | | | |
| Line Search | | | Univariate NLP: tk = , g(t) := f(**x**k + t**v**k)  Need to find **v**k and tk (aka stepsize). Then let **x**k+1 = **x**k + tk**v**k | | | | | | | | tk can be: 1) optimal step size: t = arg min f(**x**k + t**v**k)  2) Using some prefixed tk. 3) Backtrack with Armijo rule | | | | | |
| Backtrack | | | Armijo rule uses maximises t but in practical in terms of fn value decrease  1) Start with t = . 2) Check if reduction is met: f(**x**k + t**v**k) ≤ f(**x**k) + t  OR 2) = g(t) ≤ g(0) + t | | | | | | | | | | 3) If yes: use t for soln  4) If no: replace t with (where | | | |
| Newton + Line Search | | | Newton + Line Search: 1) Use **v**k =  2) Find tk (Standard Newton: tk = 1) | | | | | | | | | | 3) Let  4) Repeat step 2-3 until | | | |
| Gradient / Steepest descent | | | Suppose we let **x**k+1 = **x**k + t**v**k w a small t and = 1  How to choose **v**k s.t. f(**x**k+1) is minimized? In practice, take **v**k =  Input: **x**0, , tk, Tol. Output: **x**k so | | | | | | | | | | k = 0; while do  **v**k = ; find tk based on some strategy;  **x**k+1 = **x**k + tk**v**k; k = k+1; | | | |
| Complexity analysis of GD | | | | | Linear convergence: , which is slower than Newton's method | | | | | | | | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Overview | Large dimensions: Coordinate Descent (Jacobi and Gauss-Seidel's mtd) | | | Large data size: SGD |
| Coordinate descend | Note if f is C1, (**x**) = **0** f(**x**) = 0, i  Fixed point: xk+1 = xk (x no change after update) | | If **x**\* is an optimal soln, then ... | |
| Jacobi's Mtd | First iterate: x0 = []. g1(x1) = f(x1, ,...,). = ... gn(xn) = f(, ,...,xn). =  x1 = []. So need to use inner loop to update each xi / weights. Outer loop update from x0 to x1 to .... | | | |
| Gauss-Seidel Mtd | First iterate: x0 = []. g1(x1) = f(x1, ,...,). = ... gn(xn) = f(, ,...,xn). =  x1 = []. Double loop also. Main idea: use most recent value to calculate next xi. | | | |
| Coordinate GD | | If can't solve minimization exactly, use . Check if xk+1 meets criteria, else find next iterate | | |
| Stochastic GD (SGD) | GD: **x**k+1 = **x**k + tk. If we have N data points **z**i, ≈ . O(N) for each update of **x** | | | |
| SGD: draw a random **z**i from . **x**k+1 = **x**k + tk . O(1) for each update of **x** | | | |
| Step size must be fixed deterministic num: tk = t0, (0.5, 1]. Where = ∞ and < ∞  Stopping rule: and have to go through all data points.  Use average-SGD: for better result, as normal SGD might have oscillation around optimal point  Mini batch SGD: **x**k+1 = **x**k - , where M << N | | | |
|  | Thrm: Suppose f is c-strongly convex, is bounded, tk = , then E[] = O(1/n). 1/n convergence. Takes decreasing step size | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Constrained Optimization | | | Minimize f(**x**) **x** s.t. gi(**x**) = 0, i = 1,..., m (equality constraint)  hj(**x**) ≤ 0, j = 1,..., p (inequality constraint) | | | f, gi, hj are all cts. Feasible set: S = | |
| Regular point | | Let **x**\* S. An inequality constraint hj(**x**) ≤ 0 is active at **x**\* if hj(**x**\*) = 0. Otherwise, it is inactive at **x**\*. Equality constraints always active | | | | | |
| (LICQ): Let **x**\* S be a feasible point of (NLP). Let J(**x**\*) = be the index set of active constraints at **x**\*. Suppose that the set of gradient vectors is linearly independent (LI). Then **x**\* is a regular point or the regularity condition holds at **x**\*. (LI constraint qualification) | | | | | |
| LICQ: If only equality constraints are active, we check to be LI  If **x**\* is in the interior of S, only equality constraints are active (i.e. hj(**x**\*) < 0, so only gi active)  If only 1 constraint is active, check c is active, check c ≠ 0. If only 2 constraints c1, c2 are active, check c1 ≠ kc2, k  If m constraints are active, check [c1, ..., cm] have rank m  i.e. If a(**x**) + b(**x**) + c(**x**) = 0 and a = b = c = 0, then gradients are LI. Solve Simultaneous eqns to get a = b = c = 0. **x** is regular pt | | | | | |
| KKT conditions | | A pt **x**\* satisfies the KKT 1st order conditions if **x**\* is a regular point. And there are , ..., and , ..., s.t. , J(**x**\*) : index set of active inequality constraints at **x**\*  **x**\* is a KKT point if it satisfies KKT 1st order condition. | | | | | |
| Lagrange multipliers: Multipliers for equality (, ..., ), Multipliers for inequality () and | | | | | |
| Complementary Slackness condition: (i.e. if constraint is slack, hj(**x**\*) < 0, cannot be slack = 0, and vice versa) | | | | | |
| KKT necessary condn. Thrm 12.1: Suppose f, gi, hj : are C1 on feasible set S. Let **x**\* S be regular. If **x**\* is local min, then **x**\* is KKT pt | | | | | |
| 2nd order KKT condition | | | A point **x**\* satisfies the KKT 2nd order conditions, if **x**\* is a KKT pt, H­L(**x**\*) is a PD matrix, where HL(**x**\*) = Hf(**x**\*) + , where , are the Lagrangian multipliers (exactly same as in 1st order condition) | | | | |
| Thrm 12.2: Let f, gi, hj : are C2 on feasible set S. Suppose **x**\* S is a KKT pt, i.e. there exist and s.t. . Let HL(**x**\*) as defined previously. Suppose HL(**x**\*) is PD, then **x**\* is a local minimizer of f on S | | | | |
| E.g. | | | If have multiple KKT points, and asked to find global min, just check f(**x**) for all KKT pts. | | | | |
| Summary | | | |  |  |  | | --- | --- | --- | | Local min | Unconstrained | Constrained | | Necessary cond | **x**\* is a local min **x**\* is a critical/stationary pt  If f is C2, then Hf(**x**\*) must be PSD | **x**\* is a local min and regular **x**\* is a KKT pt  If f, gi, hj are C2, then HL(**x**\*) must be PSD on T(**x**\*) | | Sufficient cond | **x**\* is a stationary pt and H(**x**\*) is PD **x**\* is a local min | **x**\* is a KKT pt and HL(**x**\*) PD on T(**x**\*) **x**\* is a local min | | | | | |
| Primal Problem (P) | | | (P): min f(**x**) s.t. gi(**x**) = 0, i = 1,..., m, hj(**x**) ≤ 0, j = 1,...,p, **x** X, where X  Can impose additional constraints, include some constraints to simplify discussion | | | | Directly finding KKT pts can be difficult as multipliers are hard to find |
| Penalty for inequality constraints | | | | Instead of min f(**x**) s.t. h1(**x**) ≤ 0. Consider min f(**x**) + h1(**x**). ≥ 0 is the strength of penalty.  Combine both cases, | | | |
| Penalty for equality constraint | | | | Instead of min f(**x**) s.t. g1(**x**) = 0. Consider min f(**x**) + g1(**x**). For sufficiently large ± , equivalent to g1(**x**) = 0  i.e., | | | |
| Lagrangian fn | | | (P): min f(**x**) s.t. gi(**x**) = 0, i = 1,..., m, hj(**x**) ≤ 0, j = 1,...,p, **x** X  Lagrangian fn: L(**x**,, ) = f(**x**) + | | | | Rewrite as  Can swap order of min and max. Doesn't matter |
| Lagrangian dual problem (D) | | | Lagrangian dual fn: , . (D): Dual problem: . = Lagrangian dual variables/multipliers  Prop 6.1: Suppose dual fn , is finite , w ≥ 0. Then is concave fn  Then max , can be solved by finding stationary points. But sometimes may not be C1 | | | | |
| E.g. | | | min f(**x**) = s.t. h1(**x**) = -x1 -x2 + 4 ≤ 0. X = . Lagrangian fn: L(**x**, ) = f(**x**) + h1(**x**) = -x1 -x2 + 4  Dual fn: = = = when (x1 = x2 = )  Dual problem: = = 8 when = 4. So min f(**x**) = 8, x1 = x2 = 4/2 = 2 | | | | |
| E.g. | | | Same but now X = . From dual fn above, we want x1 = x2 =  x1 = if 0 ≤ ≤ 1 or 0 ≤ ≤ 2. x2 = if 1 ≤ ≤ 5 or 2 ≤ ≤ 10. So consider cases for diff  . Dual problem: max . When , max = = 6. When, max = = 10 | | | | |
| Weak duality | Thrm 13.1: Consider primal problem (P). Let **x** be a feasible soln to (P) and , be a feasible soln to (D). Then f(**x**) ≥ ,  Corollary 6.1: If **x**\* feasible to (P), , feasible to (D) s.t. f(**x**\*) = , , then **x**\* is optimal soln to (P), (, ) is optimal soln to (D)  Duality Gap: min{f(**x**): **x**  S} – max{, : ≥ 0}. Can be used as stopping criteria for algos  For constrained NLP: solve Dual problem to get , . If given **x**\*. Check if f(**x**\*) = , **x**\* is global optimal | | | | | | |
| Linear optimization | | | min f(**x**) = **c**T**x** s.t. h­­j(**x**) = **x** – bj ≤ 0, j = 1,..., m. The feasible set: **Ax** ≤ **b** is a polygon  Proposition: Suppose the feasible set is a bounded polygon, 1 soln to the Linear Optimization problem is in the set V = , where the union is taken over all feasible combinations of ji, and d is the dimension.  Note vertices of polygon lies in V, i.e. optimizer is at one of the vertices (aka extreme points) | | | | |
| E.g. | | | min f(**x**) = x1 + 2x2 s.t. -x1 ≤ 0, -x2 ≤ 0, x1 + x2 - 2 ≤ 0 | | Consider vertices of polygon, f(0,0) = 0, f(0,2) = 4, f(2,0) = 2. So min f(**x**) = 0 | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Flow chart | | Diagram  Description automatically generated | | Diagram  Description automatically generated |
| Convex Optimization & KKT cond | | Proposition: The feasible set of (P) is convex: min f(**x**) s.t. gi(**x**) := **x** - bi = 0, i = 1,...,m; hj(**x**) ≤ 0, j = 1,...,p  Thrm 12.3: Suppose f, hj are convex C1 fns and gi(**x**) := **x** - bi. If **x**\* S is a KKT pt, then **x**\* is a global minimizer of f on S  Necessity would require all optimal soln to be regular. Note (CP) = convex optimization problem | | |
| E.g. | | Check f, hj convex -> find Hessian -> find eigenvalues of Hessian -> eigenvalues > 0 = PD = convex.  Note affine fns = linear fn + constant (e.g. x1 + x2 + 3) always convex.  Conclude (P) is a (CP). So next check x\* is a KKT point (LC of gradient of active constraint = 0) -> global min | | |
| Check (P) is CP: find eigenvalue of Hessian of f(**x**). hi are affine fns, so convex. Find KKT pt: 0 = ...  So = 1.5, = 0. Note h1(**x**\*) = -1.5 < 0. So = 0. So subbing back, = 6/5 ≥ 0, = 1/5 ≥ 0. So **x**\* is a KKT pt, and a optimal soln | | |
| Strong duality | | Thrm 13.2: Suppose in the primal problem: X is convex, f, hj are convex, gi are affine. And there is **x**0 X s.t. h(**x**0) < 0 and g(**x**0) = 0. And 0 is interior of g(X) = {g(x): x X}. Then duality gap is 0: min{f(**x**): g(**x**) = 0, h(**x**) ≤ 0, **x**  X} = max{, : ≥ 0, }.  i.e. solving primal problem equivalent to solving dual problem (note for primal, **x** X converted to hi, but for dual, just ignore **x** X) | | |
| Projection on a convex set | | Thrm 5.2: Suppose S is a closed convex set. Then for every **y**, there is a unique minimizer (**y**) of the problem min{}  To find projection: 1) Solve constrained optimization. 2) Use graphical interpretation find an ansatz, and check ansatz using KKT.  3) **x**\* = (**y**) iff ≤ 0 for all **x** S. Note if **y** S, then (**y**) = **y** | | |
| E.g. | For **y** R2, find (**y**) for S = { ≤ 1}. Want to min f(**x**) = s.t. h(**x**) = - ≤ 0  Using KKT mtd: = 0. **x** - **y** + **x** = 0. So **x** = **y**/(1 + )  Case 1) = 0. Then **x** = **y,** i.e. ≤ 1. Case 2) > 0. Then h(**x**) = 0, i.e. = , i.e. = . Then **x** = | | | |
| Half space: For **y** Rd, find (**y**) for S = {**a**T**x** + b ≤ 0}. Want to min f(**x**) = s.t. h(**x**) = **a**T**x** + b ≤ 0. 0 = = **x** - **y** + **a**. So **x** = **y** - **a**  Case 1) > 0. Then h(**x**) = **a**T**x** + b = **a**T(**y** - **a**) + b = 0. Then = . Since > 0, then **a**T**y** + b > 0, i.e. **y** S. Case 2) = 0. Then **x** = **y** if **y** S | | | |
| Projected Gradient Descent | | | 1. Let **v**k = -f(**x**k). 2. Let g(t) = f(**xk** + t**v**k) and find tk = mint ≥ 0 g(t). 3. Let **x**' = **x**k + tk**v**k. 4. Let **x**k+1 = (**x**') | |
| Thrm 5.2: For any **x** S, **y**, | |
| E.g. | | min f(**x**) = x1 + x2, s.t. ≤ 1. Start from **x**0 = **0**. Find **x**1 using project G.D. w step size t0 = 1  **x**' = **x**0 - t0f = **0** - [1;1] = [-1; -1]. **x**1 = (**x**') = = [ ; ] | | |
| min f(**x**) = , s.t. 0 ≤ x1, x2 ≤ 1. Start w **x**0 = **0**, find **x**1 using projected G.D. if A = [2,0 ; 0,1], b = [1;2]  f(**x**) = ... f(**x**) =... **v**0 = - f(**x**0) = [2; 2]. g(t) = f(x0 + tv0) = f([2t; 2t]) = 6t2 - 8t + 3. g'(t) = 12t - 8 = 0. t\* = 2/3. **x**' = [4/3; 4/3]. **x**1 = (**x**') = [1; 1] | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Penalty mtd w inequality constraints | Consider min f(**x**) s.t. hj(**x**) ≤ 0, j = 1,...,p. The strictly feasible region S- := {x : hj(x) < 0, j}. Add a penalty B, P(**x**, ) = f(**x**) + B(**x**) which penalizes soln outside of S-  A barrier fn: B(x) = , where = – log() (logarithmic barrier). (which is too strong, so need decr strength)  Let P(**x**; ) = f(**x**) + B(**x**). Find xk = arg min P(**x**; ). If is finite, xk is in S-. So xk is always feasible. As to 0, we create a central path. | | | | |
| E.g. | min x s.t. -x + 1 ≤ 0. P(x, ) = f(x) - log(-h(x)) = x - log(x - 1). P'(x, ) = 1 - /(x-1). P'' > 0. For P'(x, ) = 0, x = + 1  = arg min P(x, ) = + 1. As 0, 1 (optimal pt) | | | | |
| Penalty mtd w equality constraints | Consider min f(**x**) s.t. gi(**x**) ≤ 0, i = 1,...,m. Let Q(x; ), the quadratic penalty fn, be Q(x; ) = f(x) +  When 0, the penalty for violating gi(x) = 0 is very heavy. | | | | |
| E.g. | min = f(x) s.t. g(x) = x1 + x2 - 1 = 0. Q(x, ) = f(x) + g(x)2  = arg min Q(x, ). Solve Q(x, ) = 0 to find x1, x2. Let 0, then = [x1; x2] tends to optimal soln | | | | |
| Algo | Logarithmic barrier mtd (inequality)  Input: **x**0, , f, h, , h, Tol. Output: **x**k s.t. h(**x**k) < 0, p < Tol  k = 0;  while p > Tol do: **x** = **x**k;  while > Tol do:  | **v** = ; (which is our ­–P(x, )  | t = arg mint f(**x** + t**v**); **x** = **x** + t**v**;  **x**k+1 = **x**; = ; k = k+1; | | Quadratic penalty mtd (equality)  Input: **x**0, , f, g, , g, Tol. Output: **x**k s.t. , ≤ Tol  k = 0;  while > Tol do: **x** = **x**k;  while ≥ Tol do:  | **v** = ; (which is our ­–Q(x, )  | t = arg mint f(**x** + t**v**); **x** = **x** + t**v**;  **x**k+1 = **x**; = ; k = k+1; | | |
| Penalties mtd in ML | SVM: (**a**1, b1), ..., (**a**N, bN). bi = -1 or 1  min**x**,b s.t. bi( + c) ≥ 1, for all i = 1,...,N | Penalised version: Consider slack variable = max{1 - bi( + c), 0}  min**x**,b s.t. ≥ 0, + bi( + c) ≥ 1, for all i = 1,...,N | | | |
| Regulariza-tion for ML | Consider model b = g(**x**, **a**) w data (**a**i, bi). Let F(**x**, **z**) = l(b, F(**x**, **a**))  min f(**x**) = E[F(**x**, **z**) + R(**x**)] ≈ + R(**x**)  F(**x**, **z**i) measures data misfit for ith data point  R(**x**) measures model simplicity. R aka regularization | | | R(**x**) = (Tikhonov/ridge/L2 regularization): Smooth convex fn  R(**x**) = (Lasso/L1 regularization): cts and covex, favours sparsity  R(**x**) = = #{i: xi ≠ 0} (cardinality): non-cts and non-convex | |
| E.g. | Ridge regression: f(**x**) = =  f(**x**) = AT(A**x** - **b**) + 2**x** = 0. Then **x** = (ATA + m)-1AT**b**  If = 0, then **x**\* = (ATA)-1AT**b**. But ATA might not be invertible  If ≠ 0, ATA + m is always invertible (since all its eigenvalues ≠ 0) | | | | = 2AT(Ax - b)  Lasso regression: f(**x**) = |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sub-gradient | | Let f: (-∞, ∞) be a convex fn. A vector **v** is a subgradient of f at **x** if f(**y**) ≥ f(**x**) +  The set of all subgradients at **x** = subdifferential of f at **x**, denoted as f(**x**) | | | | |
| If f is cts and convex, then is attained at **x**\* iff **0** f(**x\***) | | f is a convex set. If f is differentiable at **x**, then f(**x**) = {f(**x**)} | | |
| The subdifferential of f + g = (f + g)(**x**) {u + v: u f(**x**), v g(**x**)} | | | If f is differentiable at **x**, then (f + g)(**x**) = {f(**x**) + v: v g(**x**)} | |
| Given a set S, C = conv(S) is the convex hull of S, if C is the smallest convex set containing S | | | | |
| If S = {v1, ..., vn}, then conv(S) = | | | | |
| Suppose f(**x**) = max{f1(**x**), ..., fm(**x**)}, where fi are all convex and C1 fns. Suppose f(**x**\*) = f1(**x**\*) = ... = fj(**x**\*). Then f(**x\***) = conv({f1(**x**\*), ..., fj(**x**\*)}). | | | | Note that |g(**x**)| = max{-g(**x**), g(**x**)}  g(**x**) > / < / = 0, |g(**x**)| = {g(**x**)} / {g(**x**)} / conv{g(**x**), g(**x**)} |
| E.g. | f(x) = |x-1|. When x < 1, f(x) = {-1}. x > 1, f(x) = {1}. At x = 1, f(x) = conv{-1, 1} = [-1,1] which contains 0. So x\* = 1. | | | | | |
| Find subgradient of f(**x**) = |x1 + x2| + |x1| = g(x) + h(x). At (1,0), (0,1), (1,-1), (0,0), which is a global min  g(x) = max{x1 + x2, -x1 - x2} = max{g1(x), -g1(x)}. h(x) = max{x1, -x1} = max{h1(x), -h1(x)}  At (1,0), g(x) = x1 + x2. g(x) = {[1;1]}. h(x) = x1. h(x) = {[1;0]}. f(x) = {[2;1]}  At (0,1), g(x) = {[1;1]}. h(x) = h1(x) = - h1(x) = 0. So h(x) = conv{[1;0], [-1;0]}. So f(x) = conv{[2;1], [0;1]}. [0;0] is not in f(x) since x2 = 1  At (1,-1), g(x) = conv{[1;1], [-1;-1]}, h(x) = {[1;0]}. f(x) = conv{[2;1], [0;-1]}. Suppose [0;0] = [2;1] + (1-)[0;-1]. Then = 0 and = 0.5 (reject)  At (0,0), ... f(x) = conv{[2;1], [0;1], [0;-1], [-2;-1]}. For [2;1] + [0;1] + [0;-1] + [-2;-1] = 0, . ,  So 0 f(x) and (0,0) is minimizer. | | | | | |
| Subgradient Descent Mtd | | | Same as SGD/GD, just use **v**k = any element of subgradient f(**x**k) | | | |