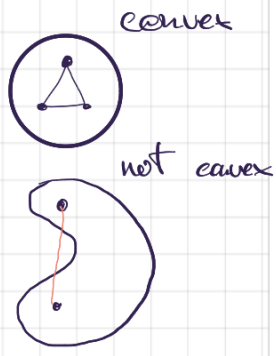
Problem definition - Convexity



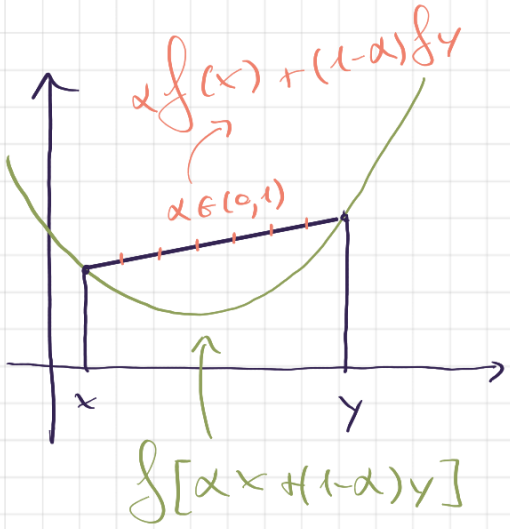
Def A set $C \subset \mathbb{R}^n$ is convex if $\forall x, y$ the segment joining the two points is contained in C

$$\alpha x + (1-\alpha) y \in C \quad \forall \alpha \in [0,1]$$

Def Given a convex set $C \subset \mathbb{R}^n$, a function $f: C \to \mathbb{R}$ is convex if

$$f(\alpha x + (1-\alpha) y) \leq \alpha f(x) + (1-\alpha) f(y)$$

$\hookrightarrow \forall x, y \in C, \forall \alpha \in [0,1]$



$f(x) \leq *$

$\hookrightarrow f$ is CONVEX

PROPRIETY

If "f" is convex and the set of feasable solutions (set of points satisfying the constrains) is a convex set, a local minimum is also a global minimum

S14

TODO

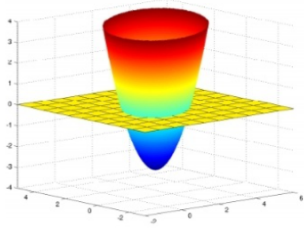
- Gradient methods - - - - - - - - - - - -

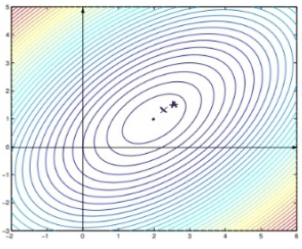
↳ if A is a symmetric positive definite (SPD) matrix, then the solution of $Ax = b$ is the UNIQUE MINIMUM of:

$$J: \mathbb{R}^n \to \mathbb{R} \qquad J(x) = \frac{1}{2} x^T A x - x^T b \qquad \left(\approx \int Ax - b\right)$$

(!) Since A is SPD and A is the Hessian of $J(x)$, then $J(x)$ is strictly convex

STEEPEST DESCENT



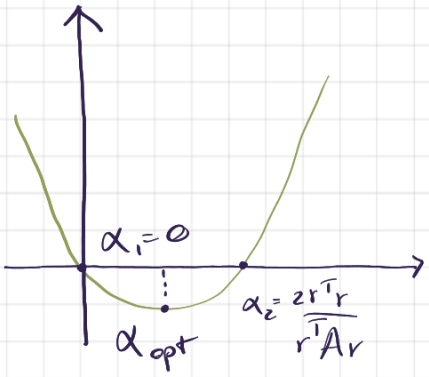


↳ IDEA: build an iterative method by moving, at each step, along the direction $-\nabla J(x_k)$

↳ $\text{direction}(x) = -\nabla J(x) = b - Ax = r(x)$

↳ $\text{step}(x) \quad = \alpha_{opt} \quad = \dfrac{r^T r}{r^T A r}$

↳ once we have a direction, knowing $J(x)$ is a paraboloid, on the plane given by $-\nabla J(x)$ will result a parabola: get the minimum for optimal step

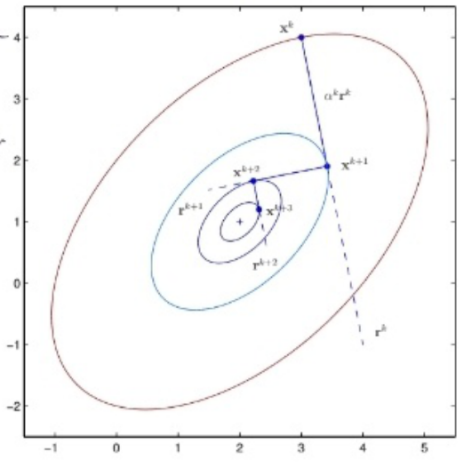


↳ IMPLEMENTATION:

matrix mult is the main cost → if A sparse it's better (n^2)

Given x_0, for $k \geq 0$ ↑

- Compute $r_k = b - Ax_k$ → direction
- Compute $\alpha_k = \frac{r_k^T r_k}{r_k^T A r_k}$ → step
- Update $x_{k+1} = x_k + \alpha_k r_k$ → update



Note that 2 matrix-vector products are performed at each iteration. We can do better:

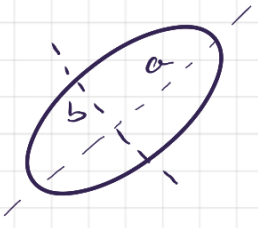
Given x_0, $r_0 = b - Ax_0$, for $k \geq 0$

- Compute $z_k = Ar_k$
- Compute $\alpha_k = \frac{r_k^T r_k}{r_k^T z_k}$
- Update $x_{k+1} = x_k + \alpha_k r_k$
- Update $r_{k+1} = r_k - \alpha_k z_k$

↳ BETTER COMPUTATIONAL COST (only 1 matrix-multiplication per iteration)

↳ CONVERGENCE PROPRIETIES:

$$K_2(A) = \frac{\max(\Lambda(A))}{\min(\Lambda(A))} \overset{\text{SPD matrix}}{=} \frac{a}{b}$$

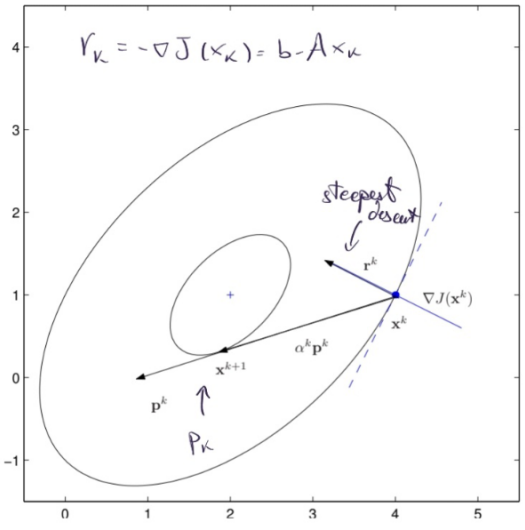


↳ bigger $K_2(A)$ = "more bounces" = harder convergence

TODO:

Energy norm?

CONJUGATE GRADIENT METHODS



Def A vector p_k is a descent vector if $\nabla J(x_k)^T p_k < 0$ for a short enough step

↳ IDEA: there may be better descent vectors to improve convergence

- In Steepest descent
  - ↳ all new directions were parallel to the ones before defined in the linear space of the new gradient

$$x_{k+1} = \arg\min J(x_k + v)$$
$$v \in \mathcal{L}(r_k)$$
$$d_k = r_k \rightarrow d_{k+1}^T d_k = 0$$

- in Conjugate descent
  - ↳ all new directions belong to the linear space of every direction took before

$$x_{k+1} = \arg\min J(x_k + v)$$
$$v \in \mathcal{L}(d_0, \dots, d_k)$$

↳ d_{k+1} and d_k are conjugate to A

$$d_k = r_k + \beta_k d_{k-1} \rightarrow d_{k+1}^T A d_k = 0$$

(!) If vectors $d_0, \dots, d_k$ form a A-conjugate set, they are also independent

↳ in order to compute $\left[d_{k+1} = -\nabla J(x_{k+1}) + \beta_k d_k \right]$ it needs to compute β_k

↳ it can be proven that β_k can be computed in such way that d_{k+1} is A-conjugate to all previous directions:

$$\beta_k = -\frac{(d_k)^T A\, r_{k+1}}{d_k^T A\, d_k}$$

↳ IMPLEMENTATION

Given x^0

Compute $r^0 = b - Ax^0$

Set $d^0 = r^0$

For $k \geq 0$

$z_k = Ad_k$

$\alpha_k = \frac{r_k^T d_k}{d_k^T z_k}$ — only computed once

$x_{k+1} = x_k + \alpha_k d_k$

$r_{k+1} = r_k - \alpha_k z_k$

$\beta_{k+1} = -\frac{(r_{k+1})^T r_{k+1}}{r_k^T r_k}$

$d_{k+1} = r_{k+1} + \beta_{k+1} d_k$

→ one line in MATLAB:

```
x = pcg ( A, b, tol,
          max_iter,
          pre_conditioner)
```

↳ CONVERGENCE PROPRIETY

- Finite Termination propriety: the method WILL stop after no more then "n" steps (as in $A \in \mathbb{R}^{n,n}$)

↳ if "n" is very large it is not usefull

↳ very depending on $K_2(A)$ → PRECONDITIONING

PRECONDITIONING

↳ look for $P \in \mathbb{R}^{n,n}$ invertible such that $K_2(P^{-1}A) << K_2(A)$

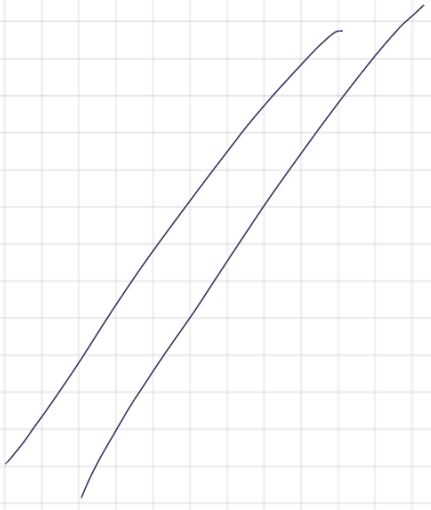
in order to: $Ax = b \longrightarrow P^{-1}Ax = P^{-1}b$

↳ best conditioner $P = A$ BUT COST TOO HIGH

↳ IDEA: build P = approx. of A

↳ if A is SPD (symmetric positive definite)

one can compute an INCOMPLETE CHOLESKY FACTORIZATION



- Unconstrained optimization (general case) - - - -

↳ $f: \mathbb{R}^n \to \mathbb{R}$ smooth (first order derivable) non linear function

we want x^* such as $f(x^*) = \min f(x)$

Def FIRST ORDER NECESSARY CONDITIONS

↳ if x^* is minimum, then $\nabla f(x^*) = 0$

Def SECOND ORDER NECESSARY CONDITIONS

↳ if x^* is minimum, then $\nabla^2 f(x^*)$ is positive semidefinite

?//

↳ x^* minimum $\Rightarrow \begin{cases} \nabla f(x^*) = 0 \\ \nabla^2 f(x^*) \text{ semidefinite positive} \end{cases}$

Def SECOND ORDER SUFFICIENT CONDITIONS

↳ $\left.\begin{array}{l} \nabla f(x^*) = 0 \\ \nabla^2 f(x^*) \text{ positive definite} \end{array}\right\} \Rightarrow x^*$ strict local minimum

(!) If f is convex any local minimum is a global minimum

If f is also differentiable, then any stationary point is a global minimum of f.

GRADIENT METHODS

STEEPEST DESCENT METHODS

↳ we want to find direction p_k and step length α

$x_{k+1} = x_k + \alpha p_k$ and let $\phi(\alpha) = f(x_k + \alpha p_k)$

↳ once you decide p_k you have only one dimension to work in

NEWTON METHOD

CHOOSING STEP LENGTH α - BACKTRACKING ARMIJO

NON-LINEAR CONJUGATE GRADIENT METHOD

FR / PR

VARIANTS OF NEWTON METHOD

↳ the computation of the Newton step may be critical in several respects:

1. assembling $\nabla^2 f(x_k)$ is costly
2. solving the linear system ($m(p) \to \nabla m(p) = 0 \to p$) is costly
3. $\nabla^2 f(x_k)$ may not be SPD ($p \to$ not a descent direction)

INEXACT NEWTON METHOD

FINITE DIFFERENCE APPROXIMATION

↳ problems:

- computing derivatives (∇f, $\nabla^2 f$) is very costly
- if you don't know f analytically you can't compute derivatives

↳ need approximation rules for f derivatives which only calls for function evaluation at some given points.

- COMPUTING DERIVATIVE $f'(\bar{x})$

↳ approximate f around $\bar{x}$ with 1st order Taylor

$$f(x) = f(\bar{x}) + f'(\bar{x})(x-\bar{x}) + \frac{1}{2} f''(\xi)(x-\bar{x})$$

← Lagrange remainder

↳ choose $h>0$ and take $x = \bar{x} + h$

$$f(\bar{x}+h) = f(\bar{x}) + f'(\bar{x})h + f''(\xi)h^2$$

↳ so $$f'(x) = \frac{f(\bar{x}+h) - f(\bar{x})}{h} - \frac{1}{2} f''(\xi) h$$

then $$\left[f'(x) \simeq \frac{f(\bar{x}+h) - f(\bar{x})}{h} \right]$$ → you can approximate it even if you don't know f analytically

with error $\frac{1}{2} f''(\xi) h \to \Theta(h)$

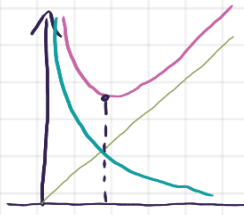
(!) Ideally h as small as possible

↳ problem with small h:

- smaller analytical error $-\frac{1}{2}f''(\xi)h$
- bigger computer error

$$tot\_error = C_1 h + C_2 \frac{1}{h}$$

analytical ε ↑ numerical error



$$h_{BEST} = |x_0| \sqrt{\varepsilon_m}$$

machine precision

↳ one can either go for

$x = \bar{x} + h$ (forward finite differences)

$x = \bar{x} - h$ (backward finite differences)

↳ or getting a (better) centred finite difference:

↳ take both FW and BW finite differences of seand order:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(\xi_+)h^3$$

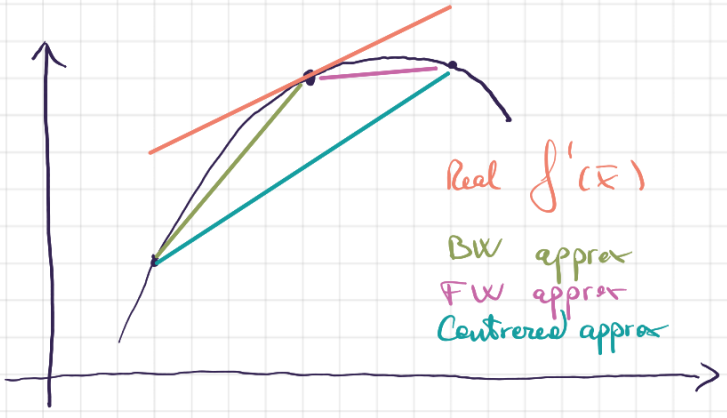
$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(\xi_-)h^3$$

↳ Subtracting: $$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} + \frac{1}{6}(f'''(\xi_+) + f'''(\xi_-))h^2$$

↑ error $\mathcal{O}(h^2)$

having $\xi_+ \in [\bar{x}, \bar{x} + h]$

$\xi_- \in [\bar{x} - h, \bar{x}]$



→ Cost: higher numbers of function evaluation (twice more)

- COMPUTING SECOND DERIVATIVE $f''(\bar{x})$

↳ approximate 3rd order Taylor expansion

$$f(x_0+h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(x_0)h^3 + \frac{1}{24}f^{(4)}(\xi_+)h^4$$

$$f(x_0-h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(x_0)h^3 + \frac{1}{24}f^{(4)}(\xi_-)h^4$$

↳ summing:

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} + \frac{1}{24}(f''''(\xi_+) + f''''(\xi_-))h^2$$

error $O(h^2)$

- COMPUTING GRADIENT $\nabla f(\bar{x})$

↳ given $f : \mathbb{R}^n \to \mathbb{R}$ $\quad f(x) \quad x = x_1 \dots x_n$

↳ knowing $\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n}\right)$

↳ you have to compute 'n' derivatives:

$$\frac{\partial f}{\partial x_i}(x) = \frac{\partial f}{\partial x_i}(x_1, \dots, x_n) \simeq \frac{f(x_1, \dots, x_i + h, \dots, x_n) - f(x_1, \dots, x_n)}{h}$$

↳ now, one may generalize:

$$\begin{aligned}(x_1, \dots, x_i + h, \dots, x_n) &= (x_1, \dots, x_n) + h(0, \dots, \underbrace{1}_{\text{pos. } i}, \dots, 0) \\ &= (x_1, \dots, x_n) + he_i = x + he_i\end{aligned}$$

where e_i is the i-th vector of the canonical basis in $\mathbb{R}^n$

↳ so in compact form: $\frac{\partial f}{\partial x_i}(x) \simeq \frac{f(x + he_i) - f(x)}{h}$

(one may use also backward or centered variants)

↳ Costs: FW/BW: $n+1$ evaluations

Centered: $2n$ evaluations

↳ choice of $h = \|\bar{x}\| \sqrt{\varepsilon_m}$

- COMPUTING JACOBIAN

↳ given $F: \mathbb{R}^n \to \mathbb{R}^n$

$$F'(x) \in \mathbb{R}^{n \times n}$$

$$\left[F'(x)\right]_{ij} = \frac{\partial F_i}{\partial x_j}(x) \to F'(x) = \begin{pmatrix} \nabla F_1(x) \\ \nabla F_2(x) \\ \vdots \\ \nabla F_n(x) \end{pmatrix}$$

↳ naive idea: repeat construction of ∇f to F_i (quite costly)

↳ we may work on columns:

$$F'(x) = \begin{pmatrix} \frac{\partial F(x)}{\partial x_1} & \vdots & \cdots & \vdots & \frac{\partial F(x)}{\partial x_n} \end{pmatrix}$$

↳ now $\frac{\partial F}{\partial x_i}(x) \simeq \frac{F(x + he_i) - F(x)}{h}$

↳ cost: $n+1$ evaluations

(only one per column because we always have the same f one one column)

↳ the cost is still high for a high 'n'

↳ we may optimize it: • matrix free implementation
• sparse Jacobians

- COMPUTING JACOBIAN: MATRIX-FREE IMPLEMENTATION

↳ if n is large: costly to build and to store

↳ generally you don't need all $F'(x)$

↳ you need to know $F'(x) \cdot d$ so you can replace given d, gives the $F'(x)$ with a function that, same result as $F'(x) \cdot d$

$$F'(x)d = \begin{pmatrix} \nabla F_1(x)^T d \\ \nabla F_2(x)^T d \\ \vdots \\ \nabla F_n(x)^T d \end{pmatrix}$$

↳ for simplicity $\nabla F_i = \nabla f_i$

↳ recall the definition of the directional derivative of f in direction "d"

$$D(f(x); d) := \lim_{h \to 0} \frac{f(x + hd) - f(x)}{h} = \nabla f(x)^T d \quad \rightarrow \quad \nabla f(x) \text{ is column vector}$$

↳ so we may approximate $F'(x)\, d \simeq \dfrac{F(x + hd) - F(x)}{h}$

↳ computational gain: ① No need for matrix allocation
② Only two function evaluations

(!) the ② is only apparent since you may need to frequently change "d" (anyway $K \ll n$)

- COMPUTE JACOBIANS: SPARSE JACOBIANS
  - ↳ consider F(x) like:

$$F(x) = \begin{pmatrix} F_1(x_1, x_2) \\ F_2(x_1, x_2, x_3) \\ F_3(x_2, x_3, x_4) \\ \vdots \\ F_n(x_{n-1}, x_n) \end{pmatrix}, \text{ then: } \quad F'(x) = \begin{pmatrix} \times & \times & & & \\ \times & \times & \times & & \\ & \times & \times & \times & \\ & & \times & \times & \times \\ & & & \times & \times \end{pmatrix}$$

tridiagonal

  - ↳ if we approximate F'(x) column-wise:

$$\frac{\partial F}{\partial x_i}(x) \simeq \frac{F(x + he_i) - F(x)}{h}$$

  - ↳ since F_3 (and so on) does NOT depend on x_1, then $F_3(x + he_1) = F_3(x)$:

$$\frac{F(x + he_1) - F(x)}{h} = \begin{pmatrix} \frac{F_1(x+he_1) - F_1(x)}{h} \\ \frac{F_2(x+he_1) - F_2(x)}{h} \\ \frac{F_3(x) - F_3(x)}{h} \\ \vdots \end{pmatrix} \simeq \begin{pmatrix} \frac{\partial F_1}{\partial x_1}(x) \\ \frac{\partial F_2}{\partial x_1}(x) \\ 0 \\ \vdots \end{pmatrix}$$

→ so it's useless to compute zero components

  - ↳ we can compute larger perturbations for another column that has different (mutually exclusive) dependencies;
  - ↳ for this particular example we need to use only 3 perturbations:

$$p_1 = h\,(e_1 + e_4 + e_7 + \dots)$$
$$p_2 = h\,(e_2 + e_5 + e_8 + \dots)$$
$$p_3 = h\,(e_3 + e_6 + e_9 + \dots)$$

→ because we have a tridiagonal F'(x)

$$\frac{F(x + h(e_1 + e_4)) - F(x)}{h} = \begin{pmatrix} \frac{F_1(x+he_1) - F_1(x)}{h} \\ \frac{F_2(x+he_1) - F_2(x)}{h} \\ \frac{F_3(x+he_4) - F_3(x)}{h} \\ \frac{F_4(x+he_4) - F_4(x)}{h} \\ \frac{F_5(x+he_4) - F_5(x)}{h} \\ 0 \\ \vdots \end{pmatrix} \simeq \begin{pmatrix} \frac{\partial F_1}{\partial x_1}(x) \\ \frac{\partial F_2}{\partial x_1}(x) \\ \frac{\partial F_3}{\partial x_4}(x) \\ \frac{\partial F_4}{\partial x_4}(x) \\ \frac{\partial F_5}{\partial x_4}(x) \\ 0 \\ \vdots \end{pmatrix}$$

$$F'(x) = \begin{pmatrix} \times & \times & & & \\ \times & \times & \times & & \\ & \times & \times & \times & \\ & & \times & \times & \times \\ & & & \times & \times \end{pmatrix}$$

↳ this can be generalized to arbitrary functions $F: \mathbb{R}^n \to \mathbb{R}^m$ by means of graphs and graph colouring

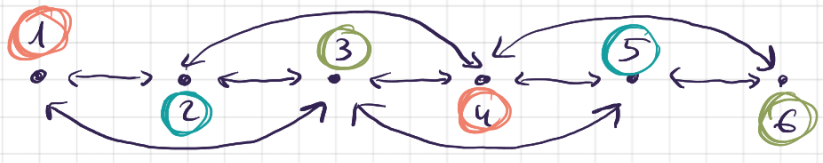
↳ build Column incidence graphs:

↳ graph with "n" nodes (number of columns)

↳ connect nodes "i" and "j" if the two columns have atleast one dependency in common

↳ assign color to the nodes: nodes not connected by arcs have the same color

example: $F_i(x_{i-1}, x_i, x_{i+1})$ $\quad n = 6$



↳ the number of colors correspond to the number of perturbations needed

↳ if i_1 i_4 have the same color we use perturbation $p = h(e_{i_1} + e_{i_2})$

- COMPUTING HESSIAN $\nabla^2 f(x)$

↳ note that Hessian of $f(x)$ is the Jacobian of $\nabla f(x)$

so:

Taylor's expansion applied to ∇f:

$$\nabla f(x+p) = \nabla f(x) + \nabla^2 f(x)p + \mathcal{O}(||p||^2)$$

Taking $p = he_i$, we get

$$\nabla^2 f(x)e_i = \frac{(\nabla f)}{\partial x_i}(x) \simeq \frac{\nabla f(x+he_i) - \nabla f(x)}{h}$$

with error $\mathcal{O}(h)$

↳ if ∇f is not available you can approximate thraugh centrered finite differences:

$$\frac{\partial f}{\partial x_i \partial x_j}(x) = \frac{f(x+he_i+he_j) - f(x+he_i) - f(x+he_j) + f(x)}{h^2} + \mathcal{O}(h)$$

Use this approximations in order to solve optimization problems with gradient based methods

SUMMARY - FINITE DIFFERENCE APPROXIMATIONS

- $f: \mathbb{R}^n \to \mathbb{R}$

	FORMULA	ERROR	TYPE
$f'(x)$	$f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h} - \frac{1}{2}f''(\xi_+)h$	$\mathcal{O}(h)$	forward
	$f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} + \frac{1}{2}f''(\xi_-)h$	$\mathcal{O}(h)$	backward
	$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} + \frac{1}{6}(f'''(\xi_+) + f'''(\xi_-))h^2$	$\mathcal{O}(h^2)$	centered
$f''(x)$	$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} + \frac{1}{24}(f''''(\xi_+) + f''''(\xi_-))h^2$	$\mathcal{O}(h^2)$	centered

$\nabla f(x)$ $\quad \frac{\partial f}{\partial x_i}(x) \simeq \frac{f(x+he_i) - f(x)}{h}$

- FW/BW cost = $n+1$ eval
- CENTERED cost = $2n$ eval

$\nabla f^2(x)$ → Jacobian of grad $\quad \nabla^2 f(x) e_i = \frac{(\nabla f)}{\partial x_i}(x) \simeq \frac{\nabla f(x+he_i) - \nabla f(x)}{h}$ (grad cost) + $n+1$ eval

↳ Second derivatives $\quad \frac{\partial f}{\partial x_i \partial x_j}(x) = \frac{f(x+he_i+he_j) - f(x+he_i) - f(x+he_j) + f(x)}{h^2}$ $\quad 3n^2+1$ eval

- $F: \mathbb{R}^n \to \mathbb{R}^m$

$$F'(x) = \left(\frac{\partial F(x)}{\partial x_1} \;\vdots\; \cdots \;\vdots\; \frac{\partial F(x)}{\partial x_n} \right) \text{ having } \frac{\partial F}{\partial x_i}(x) \simeq \frac{F(x+he_i) - F(x)}{h}$$

↳ cost = $n+1$ evaluations (1 for $F(x)$ and n $F(x+he_i)$)

↳ optimizations: • MATRIX-FREE IMPLEMENTATIONS

• SPARSE JACOBIANS

NON GRADIENT METHODS

NEDLER-MEAD METHOD

↳ direct search method ← simplex method

Def A SYMPLEX S in $\mathbb{R}^n$ is the convex hull of $n+1$ points $x_i \in \mathbb{R}^n$,

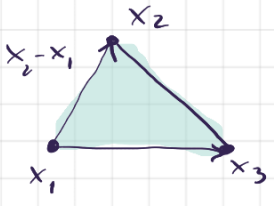
$$S = \left\{ y \in \mathbb{R}^n : y = \sum^{n+1} \lambda_i x_i \,,\ \lambda_i \geq 0,\ \sum \lambda_i = 1 \right\}$$

↳ a symplex S is NON-SINGULAR if the "n" vectors

$$\left.\begin{array}{l} x_2 - x_1 \\ x_3 - x_1 \\ \vdots \\ x_{n+1} - x_1 \end{array}\right\} \text{ are linearly independent}$$

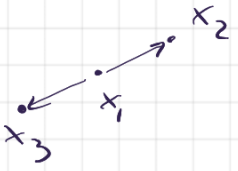
EXAMPLE

$\mathbb{R}^2 \rightarrow n = 2$



$y = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3$

non-singular



singular because $x_2 - x_1 \,/\!/\, x_3 - x_1$

- **HOW DOES NM-METHOD WORK ?**
  - ↳ based on 4 parameters
    1. ρ reflection $\rho > 0$ (typically $\rho = 1$)
    2. χ expansion $\chi > 1, \chi > \rho$ (typically $\chi = 2$)
    3. γ contraction $0 < \gamma < 1$ (typically $\gamma = 1/2$)
    4. σ shrinking $0 < \sigma < 1$ (typically $\sigma = 1/2$)

IDEA: start with a symplex and, at every step replace our points with a suitable new point that improve your situation
↳ the point chosen is the one with worse value

- **DESCRIPTION OF A SINGLE ITERATION**

S_k: given non singular symplex

$x_1^{(k)}, \dots, x_{n+1}^{(k)}$: points of S_k

↳ assume that points $x_i^{(k)}$ are ordered in such a way that $f(x_1^{(k)}) \leq \dots \leq f(x_{n+1}^{(k)})$

↳ for semplicity $f_1^{(k)} = f(x_1^{(k)})$

NUMERICAL SOLUTION OF NL EQUATIONS

↳ having $f: \mathbb{R}^n \to \mathbb{R}$
find $x^* \in \mathbb{R}$ st $f(x^*) = 0$ → NON-LINEAR EQUATIONS

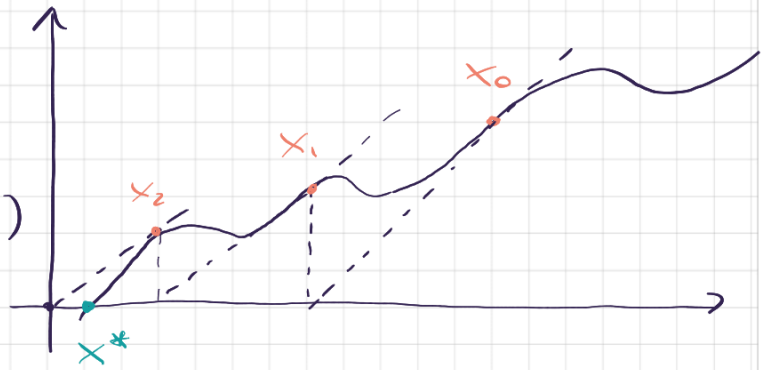
NEWTON METHOD

↳ ALGORITHM

↳ at x_k linearize $f(x_k)$
(1^{ST} order taylor polinomial)

↳ get the root of
the linearized function
and use it as x_{k+1}

↳ repeat untill you find x^*



↳ mathematically: $f(x) \simeq f(x_k) + f'(x_k)(x_{k+1} - x_k) = 0$

↳ so $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$; $f'(x_k) \neq 0$

↳ BEHAVIOR

↳ good convergence (quadratic with some assumptions)
↳ bad for robustness (the linearized function
may bring you in the opposite direction)
↳ quite problematic the choice of the starting point

(!) Typically, to improve robustness is to endow it with
a linesearch strategy applied to a merit function
(ie a function whose unconstrained minimum
corresponds to a solution of the nonlinear eq.

NUMERICAL SOLUTIONS FOR NON-LINEAR SYSTEMS OF EQUATIONS

$F \in \mathbb{R}^n \to \mathbb{R}^m$ → we want to find $F(x) = 0$

$$F(x) = \begin{pmatrix} F_1(x_1 \dots x_n) \\ \vdots \\ F_m(x_1 \dots x_n) \end{pmatrix}$$

which means $\begin{pmatrix} F_1(x) = 0 \\ F_2(x) = 0 \\ \vdots \end{pmatrix}$

↳ important for Constrained Optimization

NEWTON METHOD

↳ it works in a similar fashion as the one for single NL equations:

- take $x^{(k)}$
- linearize F at $x^{(k)}$
- replace $F(x) = 0$ by it's $\text{lin } F^A = 0$
- the solution is $x^{(k+1)}$
- repeat

↳ even in this case **convergence is good** if $F'(x)$ is non-singular

↳ the **robustness is NOT satisfying**, the method is typically enhanced by the use of linesearch applied on to the merit function:

$$f(x) = \frac{1}{2} \|F(x)\|^2$$

Jacobian

$$F(x) \simeq F(x^{(k)}) + F'(x^{(k)})(x - x^{(k)})$$

$$F(x^{(k)}) + F'(x^{(k)})(x - x^{(k)}) = 0$$

$$F'(x^{(k)}) \underbrace{x}_{\in \mathbb{R}^n} = -F(x^{(k)}) + \underbrace{F'(x^{(k)})}_{\in \mathbb{R}^{n \times n}} \cdot \underbrace{x^{(k)}}_{\in \mathbb{R}^n}$$

↑ the only unknown in a linear system $Ax = b$

$F'(x^{(k)})$ should be non-singular

Computationally better

$$F(x^{(k)}) + F'(x^{(k)}) \underbrace{(x - x^{(k)})}_{s} = 0$$

$$F'(x^{(k)}) \cdot s = -F(x^{(k)})$$

↳ then $x^{(k+1)} = x^{(k)} + s$

NON LINEAR LEAST SQUARES PROBLEM

↳ already discussed $F: \mathbb{R}^n \to \mathbb{R}^n$
finding x^* s.t. $F(x^*) = 0 \in \mathbb{R}^n$

↳ now we consider $F: \mathbb{R}^n \to \mathbb{R}^m$, $m \geq n$
$F(x) = 0 \in \mathbb{R}^m$ rappresents the overdetermined system:

$$\begin{cases} F_1(x_1 \dots x_n) = 0 \\ \vdots \\ F_m(x_1 \dots x_n) = 0 \end{cases}$$

→ can fail at having a solution (no point belongs simultaneously to ALL F_i)

↳ change the problem $(F(x)=0) \to$ ("all F_i as small as pos")
so now we want $\left[\min \frac{1}{2} \sum_i F_i^2(x^*) \right]$

↳ we call $f(x) = \frac{1}{2} \sum F_i^2(x) = \frac{1}{2} \| F(x) \|^2$
the MERIT FUNCTION

(!) In the linear case $F(x) = Ax = y$
so the problem translates in finding $\min \frac{1}{2} \| Ax - y \|_2^2$

$$\min \frac{1}{2} \| Ax - y \|_2^2 = \min \left[\frac{1}{2} x^T A^T A x - x^T A^T y + \underbrace{\frac{1}{2} y^T y}_{\text{const}} \right]$$

↳ which you can find is a convex function
(since A^TA semi pos def) so it's solvable

GAUSS-NEWTON METHOD

↳ GN is an approximation of Newton method based on the structure of the non linear least square problem:

$$\begin{cases} \nabla^2 f(x^{(k)}) p^{(k)} + \nabla f(x^{(k)}) = 0 \\ x^{(k+1)} = x^{(k)} + p^{(k)} \end{cases}$$

→ need ∇f and $\nabla^2 f$ for Newton method

$$f(x) = \frac{1}{2} \| F(x) \|_2^2 = \frac{1}{2} F^T(x) F(x) = \frac{1}{2} \sum F_i^2(x)$$

→ merit function for NLSQ problem

- ∇f ? $\rightarrow$ $\dfrac{\partial f}{\partial x_i} = F_1 \dfrac{\partial F_1}{\partial x_i} + \dots + F_m \dfrac{\partial F_m}{\partial x_i}$

↳ $$\nabla f(x) = [F'(x)]^T F(x)$$

$$F'(x) = \times \begin{bmatrix} & F & \\ & & \\ & & \end{bmatrix}$$

- $\nabla^2 f$? (I neglect dependency of x to ease notation)

↳ $$\frac{\partial f}{\partial x_i \partial x_j} = \frac{\partial F_1}{\partial x_j} \frac{\partial F_1}{\partial x_i} + F_1 \frac{\partial^2 F_1}{\partial x_i \partial x_j} + \dots + \frac{\partial F_m}{\partial x_j} \frac{\partial F_m}{\partial x_i} + F_m \frac{\partial^2 F_m}{\partial x_i \partial x_j}$$

$$= \sum_{k=1}^{m} \left(\underbrace{\frac{\partial F_k}{\partial x_i}}_{[F'(x)]_{ki}} \cdot \underbrace{\frac{\partial F_k}{\partial x_j}}_{[F'(x)]_{kj}} + \underbrace{F_k \frac{\partial^2 F_k}{\partial x_i \partial x_j}}_{F_k \cdot \nabla^2 F_k(x)} \right)$$

$$\underbrace{[F'(x)]_{ki} [F'(x)]_{kj}}_{(F'(x))^T F'(x)}$$

$$= [F'(x)]^T F'(x) + \underbrace{\sum F_k (\nabla^2 F_k(x))}_{\text{↳ it is very costly because this has to compute a Hess for every } F_i}$$

↳ so in the GN method you simply avoid it: $\nabla^2 f(x) \approx [F'(x)]^T F'(x)$

↳ so the step vector will be:

N: $p_k^{(N)}$ s.t. $\nabla^2 f(x_k)\, p_k^{(N)} = -\nabla f(x_k)$

GN: $p_k^{(GN)}$ s.t. $[F'(x_k)]^T F'(x_k)\, p_k^{(GN)} = -F'(x_k)\, F(x_k)$

explanation of why GN is linearization of NLSQ problem

25/10

Constrained Optimization

↳ given $f : \mathbb{R}^n \to \mathbb{R}$ (continously differentiable)

$\min_{x \in X} f(x)$ with $X \subset \mathbb{R}^n$ non-empty, convex, closed.

↳ if a vector $x \in X$ then is said to be a FEASIBLE POINT

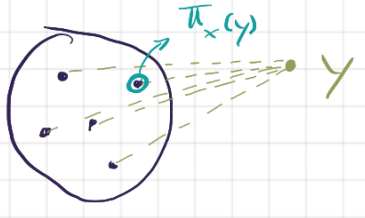
PROJECTED GRADIENT METHOD

Def PROJECTION ONTO A SET

↳ given $X \subset \mathbb{R}^n$

↳ given $y \in \mathbb{R}^n$

↳ the projection of y on X is: $\left[\Pi_X(y) = \underset{x \in X}{\operatorname{argmin}} \|y - x\|^2 \right]$



→ if $y \in X$ then $\Pi_X(y) = y$

↳ To compute the projection you need to solve a minimization problem

↳ there are some cases in which is easier to compute projections:

CASE 1 X is a sphere in $\mathbb{R}^n$

$$X = \{ x \in \mathbb{R}^n \mid \|x\|_2 \leq \rho \}$$

so $$\Pi_X(y) = \begin{cases} y & \text{if } \|y\| \leq \rho \\ \frac{\rho}{\|y\|} y & \text{if } \|y\| > \rho \end{cases}$$

CASE 2 X is a n-dim "box"

$$X = \{ x \in \mathbb{R}^n \mid L < x < U \quad \text{(componentwise)} \}$$

so $$[\Pi_X(y)]_i = \begin{cases} y_i & L_i < y_i < U_i \\ L_i & y_i < L_i \\ U_i & y_i > U_i \end{cases}$$

↳ **PROJECTED GRADIENT METHOD IDEA**

↳ given x_k compute x_{k+1} with steepest descent

↳ if $x_{k+1} \in X$ is ok

↳ if $x_{k+1} \notin X$ then use projection $\Pi_X(x_{k+1})$

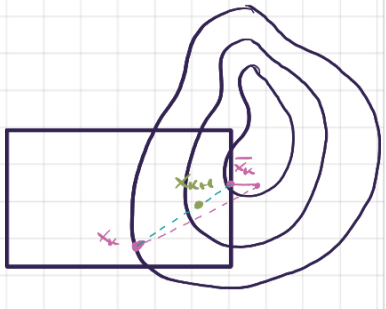
↳ so $$x_{k+1} = \Pi_X(x_k - \alpha_k \nabla f(x_k)) \qquad \alpha_k > 0$$

↳ **It is actually preferred to be used in two steps:**

① $$\bar{x}_k = \Pi_X(x_k - \gamma_k \nabla f(x_k)) \qquad \gamma > 0$$

② move along the direction $\bar{x}_k - x_k$ with suitable steplength $\alpha_k > 0$

$$x_{k+1} = x_k + \alpha_k (\bar{x}_k - x_k)$$



dir $\bar{x}_k - x_k$ → get x_{k+1} there

↳ how do we choose γ_k and α_k ?

① "Limited minimization rule"

↳ $\gamma_k = \gamma$ constant

↳ $\alpha_k = [0, 1]$ chosen to satisfy

$$f(x_k + \alpha_k (\bar{x}_k - x_k)) = \min_{\alpha \in [0,1]} f(x_k + \alpha (\bar{x}_k - x_k))$$

② Armijo rule

↳ $\gamma_k = \gamma$ constant

↳ α_k choosen to satisfy Armijo condition

LAGRANGE MULTIPLIER THEORY

↳ consider constrained opt. problems as

$$\min f(x) \qquad f: \mathbb{R}^n \to \mathbb{R}$$

$$\text{st} \quad h_i(x) = 0 \quad i = 1 \dots n \qquad h_i: \mathbb{R}^n \to \mathbb{R} \qquad \text{(equality)}$$

$$g_j(x) \leq 0 \quad j = 1 \dots p \qquad g_j: \mathbb{R}^n \to \mathbb{R} \qquad \text{(unequality)}$$

↳ written in compact form as:

$$h(x) = \begin{pmatrix} h_1(x) \\ \vdots \\ h_m(x) \end{pmatrix} \qquad g(x) = \begin{pmatrix} g_1(x) \\ \vdots \\ g_p(x) \end{pmatrix}$$

$$h(x): \mathbb{R}^n \to \mathbb{R}^m \qquad g(x): \mathbb{R}^n \to \mathbb{R}^m$$

↳ so the problem translates:

$$\min f(x)$$

$$\text{st} \quad h(x) = 0 \qquad \text{componentwise}$$

$$g(x) \leq 0 \qquad \text{componentwise}$$

• Case 1: Equality constraints only

(p) $\min f(x)$ st $h(x) = 0$

$f: \mathbb{R}^n \to \mathbb{R}$

$h: \mathbb{R}^n \to \mathbb{R}^m$

LAGRANGE MULTIPLIER THEOREM

↳ let x^* be the local minimum for (p) and assume that $\{\nabla h_i(x^*)\}_{i=1..m}$ is a set of linearly independent vectors. (⇐ IMP)

↳ there is an unique vector $\lambda \in \mathbb{R}^m$ s.t. $\boxed{\nabla f(x^*) + \sum \lambda_i \nabla h_i(x^*) = 0}$

↳ the elements of λ are called **LAGRANGE MULTIPLIERS**

LAGRANGIAN FUNCTION

↳ introduce $\boxed{\mathcal{L}(x, \lambda) = f(x) + \sum \lambda_i h_i(x) = f(x) + \lambda^T h(x)}$ called **LAGRANGIAN FUNCTION** associated to (p)

(?) What is the connection between LAGRANGE MULTIPLIERS and LAGRANGIAN FUNCTION ?

$$\boxed{\mathcal{L}(x, \lambda) = f(x) + \sum \lambda_i h_i(x)} \to \boxed{\nabla_x \mathcal{L} = \underbrace{\nabla f(x^*) + \sum \lambda_i \nabla h_i(x^*)}_{\text{Lagrange mult theorem}}}$$

but also $\boxed{\nabla_\lambda \mathcal{L}(x, \lambda) = h(x)}$

→ in a nutshell $\nabla_{x,\lambda} \mathcal{L}(x^*, \lambda^*) = 0$ is the solution to our (p)

• Case 2: Adding inequality constraints

$$\begin{aligned} \min\ & f(x) && f: \mathbb{R}^n \to \mathbb{R} \\ \text{s.t.}\ & h(x) = 0 && h: \mathbb{R}^n \to \mathbb{R}^m \\ & g(x) \le 0 && g: \mathbb{R}^n \to \mathbb{R}^p \end{aligned}$$

Def **ACTIVE SET**

↳ the active set for the inequality constraint $g(x) \le 0$ at a point $x \in \mathbb{R}^n$ is defined as:

$$A(x) = \{ j \in (1, \dots, p) \text{ s.t. } g_j(x) = 0 \}$$

↳ basically an inequality constraint $g_j(x) \le 0$ is active at x if $g_j(x) = 0$

EXAMPLE

$$\begin{aligned} \min\ & x^2 + 1 \\ \text{s.t.}\ & 1 - x \le 0 \\ & x - 2 \le 0 \end{aligned} \quad \mapsto \quad 1 \le x \le 2 \quad \to \quad X = [1, 2]$$

(feasible set)

given $x \in X \to A(x) = \{ j \in (1, 2) \mid g_j(x) = 0 \}$

$$= \begin{cases} \{1\} & \text{if } x = 1 \\ \{2\} & \text{if } x = 2 \end{cases}$$

KARUSH - KUHN - TUCKER THEOREM [KKT CONDITIONS]

↳ let us introduce the Lagrangian function for (P)

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m} \lambda_i h_i(x) + \sum_{j=1}^{p} \mu_j g_j(x)$$

↳ let x^* be a LOCAL MINIMUM point for (P) and assume x^* is REGULAR (ie. the set of vectors $\nabla h_i(x^*)\ \forall i = 1 \dots m$ & $\nabla g_j(x^*)\ \forall j \in A(x^*)$) is a set of linearly independent vectors.

↳ then there exists a UNIQUE $\lambda^* \in \mathbb{R}^m$ and a UNIQUE $\mu^* \in \mathbb{R}^p$ such that:

$$\begin{cases} \nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) = 0 \\ \nabla_\lambda \mathcal{L}(x^*, \lambda^*, \mu^*) = 0 \\ g(x^*) \leq 0 \\ \mu_j^* \geq 0 \quad \forall j = 1 \dots p \\ \mu_j^* = 0 \quad \forall j \notin A(x^*) \end{cases} \quad \begin{matrix} \text{which} \\ \text{means} \\ \rightarrow \end{matrix} \quad g(x^*) < 0$$

- REMARK: $\mu_j^* = 0$ if $g_j(x^*) < 0$; $\mu_j \geq 0$ if $g_j(x^*) = 0$ → so $\mu_j^* g_j(x^*) = 0$

↳ thus KKT CONDITIONS are EQUIVALENTLY WRITTEN:

$$\begin{cases} \nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) = 0 \\ \nabla_\lambda \mathcal{L}(x^*, \lambda^*, \mu^*) = 0 \\ g(x^*) \leq 0 \\ \mu^* \geq 0 \\ \mu_j^* g_j(x^*) = 0 \end{cases}$$

(COMPLEMENTARITY CONDITIONS)

↳ since both $g(x^*)$ and μ^* should NOT change sign $\mu_j^* g_j(x^*) = 0$ can be replaced by $(\mu^*)^T g(x^*) = 0$

(in general $v^T v = 0$ doesn't mean $v_i v_i = 0\ \forall i$)

REMARK: If $j \notin A(x^*) \rightarrow \mu_j^* = 0$

but if $j \in A(x^*) \rightarrow \mu_j^* \gtrless 0$ (so both $\begin{cases} = 0 \\ > 0 \end{cases}$ can be)

↳ if $j \in A(x^*)$ $\begin{cases} \mu_j^* = 0 & \text{NOT STRICT COMPLEMENTARITY} \\ \mu_j^* > 0 & \text{STRICT COMPLEMENTARITY} \end{cases}$

↳ strict complementarity → stronger convergence

QUADRATIC PROGRAMMING

EQUALITY CONSTRAINTS ONLY

$$(QP) \quad \min \ \frac{1}{2} x^T Q x + c^T x \quad \text{s.t.} \quad Ax = b$$

$Q \in \mathbb{R}^{n \times n}$ symmetric positive semidefinite

$c \in \mathbb{R}^n \quad x \in \mathbb{R}^n$

$A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ — $m =:$ number of constraints

↳ apply KKT:

$$\mathcal{L}(x, \lambda) = \frac{1}{2} x^T Q x + c^T x + \sum \lambda_i \cdot [Ax - b]_i$$

$$= \frac{1}{2} x^T Q x + c^T x + \lambda^T (Ax - b)$$

$$\begin{cases} \nabla_x \mathcal{L}(x, \lambda) = Qx + c + A^T \lambda \\ \nabla_\lambda \mathcal{L}(x, \lambda) = Ax - b \end{cases}$$

↳ so at minimum we have (KKT conditions for QP):

$$\begin{cases} Qx + c + A^T \lambda = 0 \\ Ax - b = 0 \end{cases}$$

↳ build $K \omega^* = d$ such that

$$K = \begin{bmatrix} Q & A^T \\ A & \varnothing \end{bmatrix} \quad \omega^* = \begin{pmatrix} x^* \\ \lambda^* \end{pmatrix} \quad d = \begin{pmatrix} -c \\ b \end{pmatrix}$$

↳ remember that $h(x) = 0 \rightarrow Ax = b$

$\nabla h(x) \rightarrow A$

and we need $h(x)$ linearly independent

which means A is full rank

THEOREM

↳ Consider $K = \begin{bmatrix} Q & A^T \\ A & \varnothing \end{bmatrix}$ → Q is symmetric positive semidef

- If A full (row) rank and given Z such that columns (Z): basis for $\ker(A)$

↳ then $Z^T Q Z$ is SPD (Q is positive definite on $\ker(A)$)

↳ then K is non-singular, hence KKT have unique solution (x^*, λ^*)
(x^* local minimizer which is also a global one)

PROPRIETIES OF K

- K is Symmetric
- Def: the INERTIA of a symmetric matrix is a triplet of numbers denoting the # of positive, negative and NULL eigenvalues $(P, N, \varnothing)$

↳ INERTIA $(K) = (n, m, 0)$ if $Z^T Q Z$ is pos. def.

dimensionality ↑ ↑ # constraints

↳ K is INDEFINITE

HOW TO SOLVE KKT CONDITIONS FOR QP

① Full system factorization

$K = LDL^T$ / LU decomposition

↳ GOOD APPROACH IF: K NOT large and NOT sparse

② Schur complement approach

$$\begin{cases} Qx + A^T\lambda = -c \\ Ax = b \end{cases} \rightarrow$$ if Q is non singular you can eliminate x from the system

$$x = -Q^{-1}A^T\lambda - Q^{-1}c$$

$$A(-Q^{-1}A^T\lambda - Q^{-1}c) = b$$

$$\underbrace{AQ^{-1}A^T}_{\text{has dim} = m}\lambda = -b - AQ^{-1}c \rightarrow$$

(from $m+n$ to m)

$\hat{Q} = AQ^{-1}A^T$ is SPD

under assumptions:
- A full rank
- Q non singular

↳ GOOD APPROACH IF: m is small and Q easy to invert

(3) Null-space method

↳ suitable also for Q singular

↳ assume we know one of $\hat{x}$ s.t. $A\hat{x} = b$

(we must have many, as $m << n$)

↳ assume also that full rank $Z \in \mathbb{R}^{n \times (n-m)}$ is given such that $AZ = \emptyset$

(note that $n - m = \dim(\text{Ker}(A))$)

↳ $\left.\begin{cases} \forall \text{ column } z_i \text{ of } Z: \; Az_j = 0 \\ z_j \in \text{Ker}(A) \\ z_j \text{ are linearly independent} \end{cases}\right\| \Rightarrow Z$: columns form a basis for $\text{Ker}(A)$

↳ we can write $\boxed{x = Zv + \hat{x}}$

where Z: generic element in $\text{Ker}(A)$

$\hat{x}$: particular solution

v: arbitrary vector in $\mathbb{R}^{n-m}$

↳ replace it in 1st equation

$$Qx - A^T\lambda = -c \rightarrow QZv + Q\hat{x} - A^T\lambda = -c$$

↳ premultiply Z^T (because $AZ = Z^TA^T = 0$)

$$Z^TQZv + Z^TQ\hat{x} + Z^TA^T\lambda = -Z^Tc$$

$$\boxed{Z^TQZv = -Z^T(c + Q\hat{x})} \rightarrow \text{being a } \mathbb{R}^{(n-m)\times(n-m)} \text{ system}$$

↳ LINSYS with Z^TQZ symmetric positive definite with dimension $n-m$

↳ you get $v \rightarrow x = Zv + \hat{x}$

REMARK: If you also need λ:

$$Qx + A^T\lambda = -c$$

$$A^T\lambda = -c - Qx$$

$$AA^T\lambda = -Ac - AQx \quad \text{and solve for } \lambda$$

REMARK: How to obtain Z?

↳ if A is full rank

$$A = \underbrace{\left[\begin{array}{c|c} A_1 & A_2 \end{array}\right]}_{\substack{m \\ n-m}} \Big\} m \rightarrow \begin{array}{ll} A_1 & \text{square } \mathbb{R}^{m \times m} \\ A_2 & \mathbb{R}^{n-m \times m} \end{array}$$

↳ A_1 square NON singular

(if the first m columns are NOT linearly independent we need to reorder)

$$\hookrightarrow Z = \underbrace{\left[\begin{array}{c} -A_1^{-1}A_2 \\ \hline I \end{array}\right]}_{n-m} \Big\} \begin{array}{l} m \\ n-m \end{array} \rightarrow AZ = \left[A_1 \,|\, A_2\right]\left[\begin{array}{c} -A_1^{-1}A_2 \\ \hline I \end{array}\right]$$

$$= -A_1^{-1}A_1A_2 + A_2$$

$$= \emptyset$$

REMARK: How to obtain $\hat{x}$?

$$A\hat{x} = b \rightarrow \left[A_1 \,|\, A_2\right]\left[\begin{array}{c} \hat{x}_1 \\ \hline \hat{x}_2 \end{array}\right] = b \rightarrow A_1\hat{x}_1 + A_2\hat{x}_2 = b$$

take $\hat{x}_2 = 0$, then $\hat{x}_1 = A_1^{-1}b \rightarrow \hat{x} = \left[\begin{array}{c} A_1^{-1}b \\ \hline 0 \end{array}\right]$

Duality theory

↳ problems with inequality constraints only

$$\left[\begin{array}{lll} (P) & \min f(x) & f: \mathbb{R}^n \to \mathbb{R} \\ & \text{st } g(x) \leq 0 & g: \mathbb{R}^n \to \mathbb{R}^m \end{array}\right]$$

↳ Lagrangian function: $\mathcal{L}(x,\lambda) := f(x) + \sum \lambda_i g_i(x)$

$\mathcal{L}: \mathbb{R}^{n+m} \to \mathbb{R}$

↳ Define $q(\lambda) := \inf_x \mathcal{L}(x,\lambda)$

(!) What is the infimum of a set?

↳ given $A \subset \mathbb{R}$, $\inf A$ is a number c such that:

① $c \leq x \quad \forall x \in A$

② $\nexists\, b$ st $b \leq x, \forall x \in \mathbb{R}, c < b$

Example 1: $A = \left\{\frac{1}{n}\right\}_{n\in\mathbb{N}} = \left\{1, \frac{1}{2}, \frac{1}{3}\right\} \longrightarrow \inf A = 0$

↳ MINIMUM vs INFIMUM? → MIN $\in A$ always

INF $\in A$ not always

↳ we introduce the set

$$\mathcal{D} = \{\lambda \text{ st } q(\lambda) > -\infty\} \quad \text{dual problem for } (P)$$

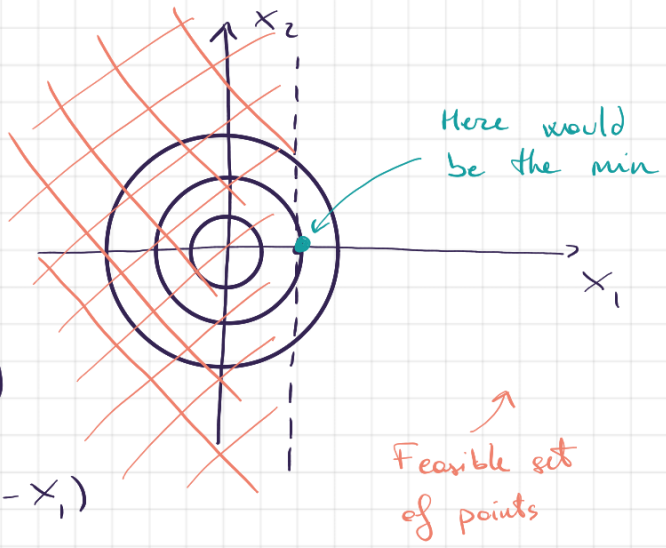
$$\begin{array}{lll} (D) & \max q(\lambda) & \longleftrightarrow \quad (P) \quad \min f(x) \\ & \text{st } \lambda \geq 0 & \qquad\qquad \text{st } g(x) \leq 0 \end{array}$$

EXAMPLE

$$\min \quad \frac{1}{2}(x_1^2 + x_2^2)$$

$$\text{st} \quad 1 - x_1 \leq 0$$

↳ we can see $x^* = (1, 0)$



$$\mathcal{L}(x, \lambda) = \frac{1}{2}(x_1^2 + x_2^2) + \lambda(1 - x_1)$$

↳ for a fixed λ, what is $\inf \mathcal{L}(x, \lambda)$?

↳ let us compute $\nabla_x \mathcal{L}(x, \lambda)$ to minimize $\mathcal{L}(x, \lambda)$ for fixed λ:

$$\nabla_x \mathcal{L}(x, \lambda) = \begin{pmatrix} x_1 - \lambda \\ x_2 \end{pmatrix} = 0 \quad \rightarrow \quad \begin{cases} x_1 = \lambda \\ x_2 = 0 \end{cases}$$

$$\mathcal{L}(x, \lambda)\Big|_{x=(\lambda,0)} = \frac{1}{2}\lambda^2 + \lambda(1 - \lambda) = -\frac{1}{2}\lambda^2 + \lambda$$

↳ we found the dual problem:

$$(D) \quad \max \; -\frac{1}{2}\lambda^2 + \lambda$$
$$\text{st} \quad \lambda \geq 0$$

(*)

THEOREM

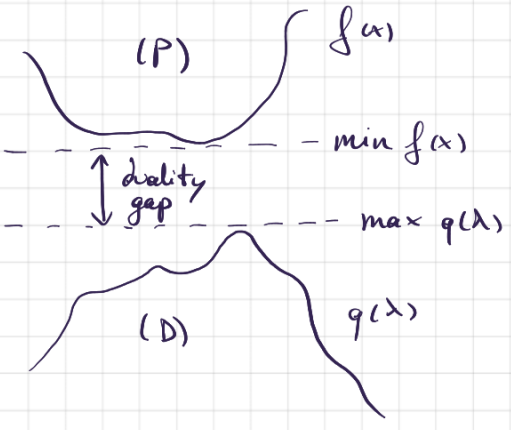
The function $q(\lambda)$ is concave and $\mathcal{D}$ is convex

WEAK DUALITY THEOREM

↳ Relationship between (P) and (D)

↳ for any feasible x for (P) and any feasible λ for (D)

$$q(\lambda) \leq f(\lambda)$$



THEOREM → (P) - (D) pair KKT conditions ?

↳ let x^* be a solution to (P) with f, g convex functions smooth at x^*

↳ if (x^*, λ^*) satisfies KKT conditions for (P) then λ^* is a solution for (D)

↳ the Lagrange multiplier of (P) is the solution to (D)

BACK TO EXAMPLE (*)

$$\min \ \frac{1}{2}(x_1^2 + x_2^2) \quad \text{st} \quad 1 - x_1 \leq 0 \quad \longmapsto \quad \mathcal{L}(x,\lambda) = \frac{1}{2}(x_1^2 + x_2^2) + \lambda(1-x_1)$$

KKT CONDITIONS

$$\begin{cases} x_1 - \lambda = 0 \\ x_2 = 0 \\ 1 - x_1 \leq 0 \\ \lambda \geq 0 \\ \lambda(1-x_1) = 0 \end{cases}$$

$\left.\begin{cases} x_1 - \lambda = 0 \\ x_2 = 0 \end{cases}\right\} \nabla_x \mathcal{L}(x,\lambda) = 0$

$g(x) \leq 0$

non negativity of $\mathcal{L}$ multipliers

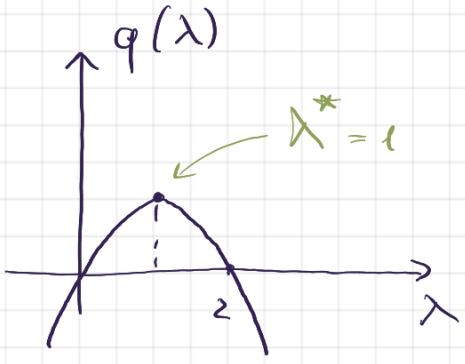
complementarity condition

↳ we find $\begin{pmatrix} x_1 = \lambda \\ x_2 = 0 \end{pmatrix}$, $\lambda(1-\lambda) = 0$ ↗ $\lambda = 0$ ✗ (doesn't satisfy $1 - x_1 \leq 0$) / ↘ $\lambda = 1$ ✓

so $w^* = (\underbrace{1 \ 0}_{x^*} \ \underbrace{1}_{\lambda^*})'$

↳ consider now the dual

$$\max \lambda(1-\lambda) \quad \text{st} \quad \lambda \geq 0$$



DT - LINEAR PROGRAMMING

$$\left[\begin{array}{llll} (LP) & \min\ c^T x & x \in \mathbb{R}^n & c \in \mathbb{R}^n \\ & \text{st}\ \ b - Ax \leq 0 & b \in \mathbb{R}^m & A \in \mathbb{R}^{m \times n} \end{array}\right]$$

↳ dual objective : $q(\lambda) = \inf_x \mathcal{L}(x, \lambda)$

$$\begin{aligned} \mathcal{L}(x, \lambda) &= c^T x + \lambda^T (b - Ax) \\ &= (c - A^T \lambda)^T x - \lambda^T b \end{aligned}$$

- **CASE 1 :** $c - A^T \lambda \neq 0$

  then $\inf_x \mathcal{L}(x, \lambda) = \inf_x \underbrace{(c - A^T \lambda)^T x}_{\text{unbounded from below (it's a simple straight line)}} + \lambda^T b = -\infty$

  → Cannot define $q(\lambda)$

- **CASE 2 :** $c - A^T \lambda = 0$

  then $\inf_x \mathcal{L}(x, \lambda) = \inf_x \lambda^T b = \lambda^T b$

  $q(\lambda) = \lambda^T b$

INTERIOR POINT METHODS

LINEAR PROGRAMMING

↳ consider the following

$$(LP) \quad \min \; c^T x \qquad x \in \mathbb{R}^n, \; c \in \mathbb{R}^n$$
$$(2) \quad \text{st} \; Ax = b \qquad b \in \mathbb{R}^m, \; A \in \mathbb{R}^{m \times n} \text{ full rank}$$
$$x \geq 0$$

(!) This way of writing the equivalence is the "canonical form" and it's equivalent to $b - Ax \leq 0$ written before BUT with different meanings for c, x, A, b :

↳ start from (1) → $b - Ax \leq 0$ form

↳ introduce $s = Ax - b$ (slack variable)

so rewrite (1) as
$$\begin{cases} \min \; c^T x \\ \text{st} \quad s \geq 0 \\ \qquad s = Ax - b \end{cases}$$

↳ introduce
$$\begin{cases} x^+ = \max(x, 0) \\ x^- = \max(-x, 0) \end{cases} \longrightarrow x = \begin{pmatrix} 7 \\ -3 \\ 5 \end{pmatrix} \; x^+ = \begin{pmatrix} 7 \\ 0 \\ 5 \end{pmatrix} \; x^- = \begin{pmatrix} 0 \\ 3 \\ 0 \end{pmatrix}$$

so $x = x^+ - x^-$

↳ so write (1) as
$$\begin{aligned} \min \quad & c^T(x^+ - x^-) = c^T x^+ - c^T x^- + 0s \\ \text{st} \quad & s = Ax^+ - Ax^- - b \\ & s \geq 0 \quad x^+ \geq 0 \quad x^- \geq 0 \end{aligned}$$

↳ if we introduce $\hat{c} = \begin{pmatrix} c \\ -c \\ 0 \end{pmatrix} \; \hat{x} = \begin{pmatrix} x^+ \\ x^- \\ s \end{pmatrix} \; \hat{A} = \begin{bmatrix} A & -A & -I \end{bmatrix}$

↳ we get
$$\begin{aligned} \min \quad & \hat{c}^T \hat{x} \\ (2) \quad & \hat{A}\hat{x} = b \\ & \hat{x} \geq 0 \end{aligned}$$

↳ let's write the dual for (LP) ②

(P-LP) $\min\ c^T x$ — A full rank

$\text{st}\ \ Ax = b$ — $\lambda \to$ equality constraints $\quad \lambda \in \mathbb{R}^m$

$x \geq 0$ — $s \to$ inequality constraints $\quad s \in \mathbb{R}^n$

(because we need $\leq$ ineq. const.)

$$\mathcal{L}(x, \lambda, s) = c^T x + s^T(-x) + \lambda^T(b - Ax)$$

$$= (c - s - A^T\lambda)^T x + \lambda^T b$$

↳ $q(\lambda, s) = \min_x \mathcal{L}(x, \lambda, s)$ → for fixed λ, s

$$\nabla_x \mathcal{L}(x, \lambda, s) = c - s - A^T\lambda = 0$$

↳ replace $c - s - A^T\lambda = 0$ in $\mathcal{L}(x, \lambda, s)$ and get $q(\lambda, s) = \lambda^T b$

(D-LP) $\max\ \lambda^T b$

$\text{st}\ \ c - s - A^T\lambda = 0$

$s \geq 0,\ \lambda \geq 0$

x : Primal var.

λ, s : Dual var

! Remember Weak Duality holds:

$\lambda^T b \leq c^T x \quad \forall\, x \mid x$ is primal feasible

$\forall\, \lambda \mid \lambda$ is dual feasible

$$X = \{ x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0 \} \to \text{primal feasible}$$

$$Y = \{ (\lambda, s) \in \mathbb{R}^{m+n} \mid A^T\lambda + s = c,\ s \geq 0 \} \to \text{dual feasible}$$

STRONG DUALITY THEOREM (FOR LP)

↳ if either $X \neq \emptyset$ or $Y \neq \emptyset$
and either P-LP or D-LP is solvable
then: $\min_x c^T x = \max_\lambda b^T \lambda$

↳ thus the DUALITY GAP is vanishing

KKT CONDITIONS FOR LP

(P-LP) $\min c^T x$
st $Ax = b$
$x \geq 0$

$\mathcal{L}(x, s, \lambda) = c^T x - s^T x + \lambda^T (b - Ax)$

KKT: $\begin{cases} \nabla_x \mathcal{L} = 0 \\ \nabla_\lambda \mathcal{L} = 0 \\ x \geq 0 \\ s \geq 0 \\ x^T s = 0 \quad \text{(complementarity condition)} \end{cases}$

KKT: $\begin{cases} c - s - A^T \lambda = 0 \\ b - Ax = 0 \\ x \geq 0 \\ s \geq 0 \\ x^T s = 0 \end{cases}$

(D-LP) $\max b^T \lambda$
st $c - s - A^T \lambda = 0$
$s \geq 0$

<< can be re-written as: >>

$\max b^T \lambda$
st $c - A^T \lambda \geq 0$

$\mathcal{L}(\lambda, x) = -b^T \lambda + x^T (A^T \lambda - c)$ ← multiplier for ineq. constr.

KKT: $\begin{cases} Ax - b = 0 \\ c - A^T \lambda \geq 0 \\ x \geq 0 \\ x^T (c - A^T \lambda) = 0 \end{cases}$

KKT: $\begin{cases} Ax - b = 0 \\ s \geq 0 \\ x \geq 0 \\ x^T s = 0 \\ s = c - A^T \lambda \end{cases}$ ← reintroduced

(!) We can observe that we have the same KKT-conditions for both problems
↳ indeed they are called **KKT conditions for the primal-dual pair**

INTERIOR POINT METHOD IDEA (FOR LP)

↳ it tries to find a vector (x, λ, s) satisfying:

$$\text{KKT} \begin{cases} c - s - A^T\lambda = 0 \\ b - Ax = 0 \\ x \geq 0 \\ s \geq 0 \\ x^T s = 0 \end{cases} \Bigg| \rightarrow \begin{array}{l} \text{constrained} \\ \text{non linear} \\ \text{equations} \end{array}$$

↳ thus we can define:

$$F(x, \lambda, s) = \begin{pmatrix} c - s - A^T\lambda \\ Ax - b \\ x^T s \end{pmatrix} \begin{matrix} \rightarrow \mathbb{R}^n \\ \rightarrow 0 \in \mathbb{R}^m \\ \rightarrow \mathbb{R} \end{matrix}$$

$x \in \mathbb{R}^n$, $\lambda \in \mathbb{R}^m$, $s \in \mathbb{R}^n$

↳ PROBLEM: $F: \mathbb{R}^{2n+m} \rightarrow \mathbb{R}^{n+m+1}$ not squared

↳ TRICK: replace $x^T s = \sum x_i s_i = x_1 s_1 + \dots + x_n s_n = 0$ with XSe

where $X = \begin{pmatrix} x_1 & & \\ & x_2 & \\ & & \ddots \\ & & & x_n \end{pmatrix}$, $S = \begin{pmatrix} s_1 & & \\ & s_2 & \\ & & \ddots \\ & & & s_n \end{pmatrix}$, $e = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

so $XSe = \begin{pmatrix} x_1 s_1 \\ x_2 s_2 \\ \vdots \\ x_n s_n \end{pmatrix} \in \mathbb{R}^n$

↳ we get $F(x, \lambda, s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe \end{bmatrix}$

having $F: \mathbb{R}^{2n+m} \to \mathbb{R}^{2n+n}$

↳ so we need to find (x, λ, s) st $\begin{cases} F(x, \lambda, s) = 0 \\ x, s \geq 0 \end{cases}$

INTERIOR POINT METHOD ALGORITHM FOR LP

↳ iteratively choose $(x_k \ \lambda_k \ s_k)$ satisfying $x_k > 0$, $s_k > 0$

↳ use perturbed Newton method

↳ ALGORITHM:

- Given $(x_0 \ \lambda_0 \ s_0)$
- For $k > 0$
  - Compute $(\Delta x_k \ \Delta\lambda_k \ \Delta s_k)$ with Newton method

$$F'(x_k \ \lambda_k \ s_k) \begin{pmatrix} \Delta x_k \\ \Delta\lambda_k \\ \Delta s_k \end{pmatrix} = -F(x_k \ \lambda_k \ s_k)$$

$$\| \quad (x \quad \lambda \quad s)$$

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix}$$ ← sparse!

  - Update $\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \\ s_k \end{bmatrix} + \alpha_k \begin{bmatrix} x_k \\ \lambda_k \\ s_k \end{bmatrix}$

α_k given by line search in order to satisfy $x_{k+1} > 0$ and $s_{k+1} > 0$

IMPROVEMENTS FOR IPM FOR LP

(1) Consider a less aggressive choice

$$F(x,\lambda,s) = 0 \quad \longrightarrow \quad F(x,\lambda,s) \leq tol$$

↳ in particular we focus on XSe
at each iteration XSe is "sufficiently small"
and approaches zero towards the solution

↳ we need a measure for the distance from the solution:

$$\mu = \frac{\sum x_i s_i}{n} \xrightarrow{k\to\infty} 0 \quad \longmapsto \quad n\mu = x^T s =: \text{duality gap}$$

↳ we can prove $x^T s$ is the duality gap actually:

$$\begin{aligned} \text{Duality Gap} &=: c^T x - b^T \lambda = \\ &= (A^T\lambda + s)^T x - b^T\lambda \quad \rightarrow \text{use } c = A^T\lambda + s \\ &= \dots = \\ &= \lambda^T (Ax - b) + x^T s = x^T s \end{aligned}$$

($Ax - b = 0$)

↳ thus by the strong duality theorem we know that $x^T s$ vanishes at the solution

↳ so we can impose

$$XSe = 0 \quad \rightarrow \quad XSe - \sigma\mu e = 0$$

↳ μ will be updated at each step
↳ $\sigma \in (0,1)$ called centrality parameter

(2) A different rule of choosing α_k

↳ for simplicity we ignore σ for now

Def ANALYTIC CENTER

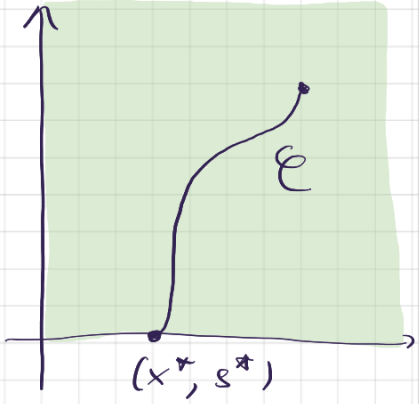
↳ for any fixed $\mu > 0$, solution to $F(x, \lambda, s; \mu) =$ is called analytic center or μ-center and it's denoted by $(x(\mu), \lambda(\mu), s(\mu))$

Def CENTRAL PATH

↳ the set

$$\mathcal{C} = \{(x(\mu)\ \lambda(\mu)\ s(\mu)) : \mu > 0\}$$

is called primal-dual central path

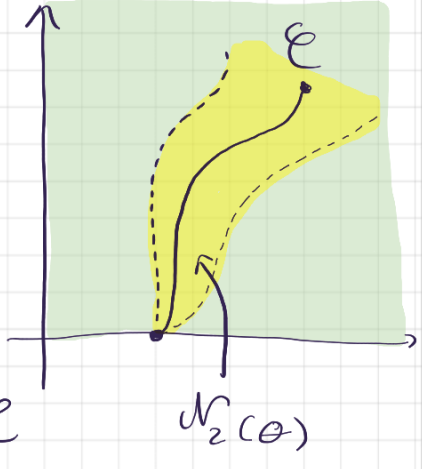


↳ we now want to follow the central path but NOT exactly, just stay close

↳ we want to be more strict as we get close to $\mu = 0$

$$XSe \overset{\times}{=} \mu e \quad \rightarrow \quad XSe \overset{\checkmark}{\approx} \mu e$$

↳ we introduce the neighborhood of $\mathcal{C}$



$$\mathcal{N}_2(\theta) = \{(x, \lambda, s) \text{ strictly feasible} : \|XSe - \mu e\|_2 < \theta\mu\}$$

$$(x, \lambda, s) \in \mathcal{F}^\circ = \{(x, \lambda, s) \in \mathbb{R}^{2n+m} : \begin{array}{l} Ax = b \\ c = s + A^T\lambda \\ x, s > 0 \end{array}\}$$

$\mathcal{F}^\circ$: strictly; $x, s > 0$: strictly

SHORT STEP PATH FOLLOWING FEASIBLE IPM

$(\Delta x_k, \Delta \lambda_k, \Delta s_k)$ is actually short — follows $\mathcal{C}$ — all iterates are feasible

- start with $(x_0, \lambda_0, s_0) \in \mathcal{N}_2(\theta)$
- for $k > 0$
  - Compute Newton step ensuring $(x_{k+1}, \lambda_{k+1}, s_{k+1})$ stay feasible:

$$F'(x_k \lambda_k s_k \mu_k) \begin{pmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta s_k \end{pmatrix} = -F(x_k \lambda_k s_k ; \mu_k)$$

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta s_k \end{bmatrix} = \begin{bmatrix} c - s_k - A^T \lambda_k > 0 \\ b - A x_k > 0 \\ -XSe + \sigma \mu_k e \end{bmatrix}$$

since is feasible it satisfies first two conditions

! How do we know we are and stay feasible?

↳ it can be proven that by choosing

$$\theta = 0{,}1 \qquad \sigma = 1 - \frac{0{,}1}{\sqrt{n}}$$

all iterates belong to $\mathcal{N}_2(\theta)$

↳ we use σ to reduce μ: $\mu_{k+1} = \sigma \mu_k$

! DRAWBACK: Since $\mathcal{N}_2(\theta)$ is very narrow we make little progress at each iteration

LONG STEP PATH FOLLOWING FEASIBLE IPM

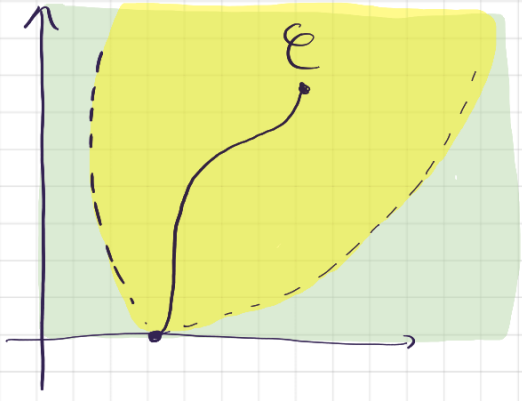
↳ we can enlarge $\mathcal{N}_2(\theta)$ to $\mathcal{N}_\infty(\gamma)$

$$\mathcal{N}_\infty(\gamma) = \left\{ (x, \Lambda, s) \in \mathcal{F}^\circ : x_i s_i \geq \gamma \mu \ \ \forall i \right\}$$

$\gamma \approx 10^{-3}$

↳ basically we ask that no pair $x_i s_i$ that approaches zero faster than the others

↳ this way we can take generally longer steps



THEOREM

↳ given $(x_0, \Lambda_0, s_0) \in \mathcal{N}_\infty(\gamma)$

given $\varepsilon > 0$

↳ asymptotically you get

$\mu_K \leq \varepsilon \mu_0$ within $\mathcal{O}\left(n \log\left(\frac{1}{\varepsilon}\right)\right)$ iterations

PREDICTOR - CORRECTOR IPM IMPLEMENTATION

↳ consider the original KKT (P)-(D) pair

$$\begin{cases} c - s - A^T\lambda = 0 \\ Ax - b = 0 \\ XSe = 0 \\ x, s \geq 0 \end{cases} \leftarrow \text{without } -\sigma\mu e$$

↳ also consider the Newton step:

↳ called **affine scaling system**

$$\textcircled{\star} \quad \begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x_k^{AFF} \\ \Delta \lambda_k^{AFF} \\ \Delta s_k^{AFF} \end{bmatrix} = \begin{bmatrix} c - s_k - A^T\lambda_k \\ b - Ax_k \\ -XSe \end{bmatrix}$$

↳ let's consider what happens for $x_k + \Delta x_k^{AFF}$ and $s_k + \Delta s_k^{AFF}$
(we are seeking x, s satisfying $XSe = 0$ having $x, s \geq 0$)

↳ $XSe = 0$ at solution,
so in general $X_k S_k e \neq 0$

$= \text{diag}(\Delta x_k)$ $= \text{diag}(\Delta s_k)$

$$x_{k+1} = x_k + \Delta x_k^{AFF} \; ; \; s_{k+1} = s_k + \Delta s_k^{AFF}$$

$$(XSe)_{k+1} = (x_k + \Delta x_k^{AFF})(s_k + \Delta s_k^{AFF})$$

$$= \underbrace{x_k s_k + x_k \Delta s_k^A + s_k \Delta x_k^A}_{=0 \rightarrow \text{according to } \textcircled{\star} \text{ (3rd block eq)}} + \Delta x_k^A \Delta s_k^A$$

$$x_k \Delta s_k + s_k \Delta x_k = -x_k s_k e$$

$$= \Delta x_k^A \Delta s_k^A$$

(!) The situation improves if $\Delta x_k^A \Delta s_k^A \approx 0$

↳ **Predictor-Corrector IDEA**:

- Compute the Newton step $(\Delta x_k^A \quad \Delta\lambda_k^A \quad \Delta s_k^A)$ → PREDICTOR

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x_k^{AFF} \\ \Delta\lambda_k^{AFF} \\ \Delta s_k^{AFF} \end{bmatrix} = \begin{bmatrix} c - s_k - A^T\lambda_k \\ b - Ax_k \\ -XSe \end{bmatrix}$$

- Compute a new direction $(\Delta x_k^C \quad \Delta\lambda_k^C \quad \Delta s_k^C)$ that pushes towards $\Delta x_k^A \Delta s_k^A = 0$

  i.e. solve

$$\begin{bmatrix} 0 & A^T & I_n \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x_k^{Corr} \\ \Delta\lambda_k^{Corr} \\ \Delta s_k^{Corr} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\Delta X_k^A \Delta S_k^A \end{bmatrix}$$

I assume to have a feasible iterate

Implementation

18/12

100% ∞

I love you pusone ♡

I love you more pusona ♡

PUS1

- Summarizing Predictor-Corrector IPM

(1) Solve the affine scaling system

(2) Compute α st $x_{k+1} = x_k + \alpha \Delta x_k^{A} > 0$

$s_{k+1} = s_k + \alpha \Delta s_k^{A} > 0$

(3) Build μ_k^{Aff}, σ_k

(4) Solve for $(\Delta x_k^{New} \; \Delta \lambda_k^{New} \; \Delta s_k^{New})$ the NEW system merging predictor & corrector + perturbation

(5) Compute α st $x_{k+1} = x_k + \alpha \Delta x_k^{N}$

$s_{k+1} = s_k + \alpha \Delta s_k^{N}$

(6) Compute $(x_{k+1} \; \lambda_{k+1} \; s_{k+1})$

INTERIOR POINT METHODS FOR QP

↳ also having inequality constraints

(QP) $\min \ \frac{1}{2} x^T Q x + c^T x$ st. $Ax \geq b$ $\rightarrow$ (KKT)

$$\begin{cases} Qx + c - A^T\lambda = 0 \\ \lambda \geq 0 \\ b - Ax \leq 0 \\ \lambda^T (b - Ax) = 0 \end{cases}$$

↳ introduce $y = Ax - b$ $\rightarrow$ (KKT)

$$\begin{cases} Qx + c - A^T\lambda = 0 \\ \lambda \geq 0 \\ y \geq 0 \\ \lambda^T y = 0 \rightarrow \Lambda Y e = 0 \\ y = Ax - b \end{cases}$$

(Λ = diag(λ), Y = diag(y))

↳ now solve $\begin{cases} F(x, y, \lambda) = 0 \\ \Lambda, Y \geq 0 \end{cases}$ being $F(x,y,\lambda) = \begin{bmatrix} Qx + c - A^T\lambda \\ Ax - b - y \\ \Lambda Y e \end{bmatrix}$

↳ better having perturbed $F(x, y, \lambda; \mu, \theta) = \begin{bmatrix} Qx + c - A^T\lambda \\ Ax - b - y \\ \Lambda Y e - \theta \mu e \end{bmatrix}$

PENALTY METHODS

↳ consider a purely non-linear program

$$\begin{aligned} &\min f(x) && f: \mathbb{R}^n \to \mathbb{R}^n \\ &\text{st } h(x) = 0 && h: \mathbb{R}^m \to \mathbb{R}^m \\ &\quad g(x) \leq 0 && g: \mathbb{R}^n \to \mathbb{R}^p \end{aligned}$$

IDEA: plug the constraints in the cost function so we have an unconstrained problem

QUADRATIC PENALTY METHODS

↳ first analyze equality constraints

$$(P_1)\ \min f(x) \ \text{ st } h(x) = 0 \quad \longmapsto \quad Q(x;\mu) := f(x) + \frac{\mu}{2}\|h(x)\|_2^2$$

(Penalty Parameter)

(!) μ parameter may make the problem unbounded from below, so we have to be careful in choosing it

↳ what about inequality constraints?

↳ we introduce $g^+(x) = \max(g(x), 0)$

$$g = \begin{pmatrix} 7 \\ -5 \\ x-1 \end{pmatrix} \to g^+ = \begin{pmatrix} 7 \\ 0 \\ \max(x-1, 0) \end{pmatrix}$$

and replace $g(x) \leq 0$ with $g^+(x) = 0$

$$Q(x;\mu) = f(x) + \frac{\mu}{2}\|h(x)\|^2 + \frac{\mu}{2}\|g^+(x)\|$$

QUADRATIC PENALTY METHOD ALGORITHM

- Given $x_0^{(0)}$, $\mu_0 > 0$
- For $k > 0$ ← outer iters
  - Build $Q(x; \mu_k)$ and find an APPROXIMATE sol
    $$\min Q(x, \mu_k) \quad \text{starting from } x_k^{(0)}$$
  - Update μ_{k+1}, choose $x_{k+1}^{(0)}$ for next step

  Until convergence

NOTE: $\mu_{k+1} = \rho \mu_k$, $\rho = \begin{cases} 10 \rightarrow \text{if inner iterations small} \\ 1{,}5 \rightarrow \text{if inner iterations large} \end{cases}$

PROBLEMS WITH Q. PENALTY METHODS

↳ in general $x_k \approx \operatorname{argmin} Q(x, \mu_k)$

but $x_k \neq \operatorname{argmin} f(x)$ st $h(x) = 0$

↳ you reach $x_k \approx \operatorname{argmin} f(x)$ only at CONVERGENCE

↳ furthermore $h(x_k) \neq 0$

↳ it can be PROVEN that:

$$h_i(x) \approx -\frac{\lambda_i^*}{\mu_k}$$

→ i-th Lagrange Multiplier associated to $h_i(x) = 0$ in KKT theorem

↳ penalty parameter at iteration k

↳ remember $\mu_k \xrightarrow{k \to \infty} \infty$ so $h_i(x) \to 0$

METHOD OF MULTIPLIERS
(AUGMENTED LAGRANGIAN METHOD)

$$\mathcal{L}_A(x;\lambda,\mu) := f(x) - \lambda^T h(x) + \frac{\mu}{2}\|h(x)\|^2$$

↳ which is composed by $\begin{cases} \mathcal{L}(x,\lambda) = f(x) - \lambda^T h(x) \\ Q(x,\mu) = f(x) + \frac{\mu}{2}\|h(x)\|^2 \end{cases}$

↳ the basic IDEA is:

- fix $\mu_k > 0$, $\lambda_k \in \mathbb{R}^n$ and approximately solve $\min \mathcal{L}_A(x_k, \lambda_k; \mu_k) \to x_k$
- update μ_k, λ_k and repeat until convergence

↳ how do you update λ_k?

↳ the optimality condition for $\min_x \mathcal{L}_A(x, \lambda_k; \mu_k)$ (P2) is $\nabla_x \mathcal{L}_A = 0$

- $\nabla_x \mathcal{L}_A = \nabla f(x_k) - \sum_{i=1}^{m} \left((\lambda_k)_i - \mu_k h_i(x_k) \right) \nabla h_i(x_k) \approx 0$

↳ while the optimality conditions for [$\min f(x)$ (P1) st $h(x) = 0$]

$$\mathcal{L}(x,\lambda) = f(x) - \lambda^T h(x)$$

$$\text{KKT}: \begin{cases} \nabla_x \mathcal{L} = 0 \\ \nabla_\lambda \mathcal{L} = 0 \end{cases} \to \begin{cases} \nabla f(x) - \sum_{i}^{m} \lambda_i \nabla h_i(x) \\ -h(x) = 0 \end{cases}$$

↳ let $x^* = \operatorname{argmin} f(x)$ and λ^* sat. KKT conditions

then • $\nabla f(x^*) - \sum \lambda_i^* \nabla h_i(x) = 0$

↳ let's look closely at both optimality conditions

- $\nabla f(x^*) - \sum \lambda_i^* \nabla h_i(x) = 0$

- $\nabla f(x_k) - \sum_{i=1}^{m} \left((\lambda_k)_i - \mu_k h_i(x_k) \right) \nabla h_i(x_k) \approx 0$

↳ we want that $\lambda_i^* \approx (\lambda_k)_i - \mu_k h_i(x_k)$ atleast asymptotically

$$h_i(x_k) \approx \frac{(\lambda_k)_i - \lambda_i^*}{\mu_k} \rightarrow \begin{array}{l} \text{the more } (\lambda_k)_i \rightarrow \lambda_i^* \\ \text{the more } h_i(x_k) = 0 \end{array}$$

↳ this suggests the following rule:

$$\lambda_{k+1} = \lambda_k - \mu_k h(x_k)$$

METHOD OF MULTIPLIERS ALGORITHM

- Given $\mu_0 > 0$, $x_0^{(0)} \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$
- For $k > 0$
  - Build $\mathcal{L}_A(x, \lambda_k; \mu_k)$ and find

$$x_k \approx \operatorname{argmin} \mathcal{L}_A(x; \lambda_k, \mu_k)$$

  - Update $\lambda_{k+1} = \lambda_k - \mu_k h_i(x)$

$\mu_{k+1} = \rho \mu_k$

choose $x_{k+1}^{(0)}$

ALTERNATE DIRECTION METHOD OF MULTIPLIERS (ADMM)

↳ augmented Lagrangian for separable problems.

$$\min \; f_1(x) + f_2(z) \quad \text{st} \quad Ax + Bz = c$$

↳ can rewrite constraints as $D = [A \mid B]$, $w = \begin{bmatrix} x \\ --- \\ z \end{bmatrix}$

thus $Dw = c$

↳ exploit problem structure to simplify iterations

ADMM uses the Augmented Lagrangian Method in 3 steps:

① $\min \mathcal{L}_A(x, z; \lambda, \mu)$ wrt x

② $\min \mathcal{L}_A(x, z; \lambda, \mu)$ wrt z

③ update λ

$$\mathcal{L}_A(x, z; \lambda, \mu) = f_1(x) + f_2(z) - \lambda^T(Ax + Bz - c) + \frac{\mu}{2}\|Ax + Bz - c\|^2$$

ADMM ALGORITHM

- Given $x_0, z_0, \lambda_0, \mu_0 > 0$
- For $k > 0$
  - Find $x_{k+1} = \operatorname{argmin} \mathcal{L}_A(x_k, z_k; \lambda_k, \mu_k)$
    $= \operatorname{argmin} f_1(x) + f_2(z) - \lambda^T(Ax + Bz - c) + \frac{\mu}{2}\|Ax + Bz - c\|^2$
  - Find $z_{k+1} = \operatorname{argmin} \mathcal{L}_A(x_k, z_k; \lambda_k, \mu_k)$
    $= \operatorname{argmin} f_1(x) + f_2(z) - \lambda^T(Ax + Bz - c) + \frac{\mu}{2}\|Ax + Bz - c\|^2$
  - Update $\lambda_{k+1} = \lambda_k - \mu_k(Ax_{k+1} + Bz_{k+1} - c)$

    $\mu_{k+1} = \rho \mu_k$

    Choose x_{k+1}, z_{k+1}

ADMM CONVERGENCE PROPRIETIES

↳ under mild assumptions, for $k \to \infty$

- $Ax + Bz - c \to 0$
- $f_1(x_{k+1}) + f_2(z_{k+1}) \to f_1(x^*) + f_2(z^*)$
- $\lambda_{k+1} \to \lambda^*$

↳ stronger assumptions are needed for $x_{k+1} \to x^*$ and $z_{k+1} \to z^*$

↳ rate of convergence: ADMM behaves like a 1^{st} order method

ADMM IN SCALED FORM

↳ using $\rho = \frac{\lambda}{\mu}$ you can replace:

$$\begin{cases} x_{k+1} = \operatorname{argmin} f_1(x) - \lambda^T (Ax + Bz - c) + \frac{\mu}{2} \|Ax + Bz - c\|^2 \\ \text{with} \\ x_{k+1} = \operatorname{argmin} f_1(x) + \frac{\mu_k}{2} \|Ax + Bz - c - \rho_k\|^2 + \frac{1}{2\mu_k} \|\lambda_k\|^2 \\ \qquad = \operatorname{argmin} f_1(x) + \frac{\mu_k}{2} \|Ax + Bz - c - \rho_k\|^2 \end{cases}$$

not dep on x

(and similarly for z_{k+1})

↳ then only update $\rho_{k+1} = \rho_k - (Ax_{k+1} + Bz_{k+1} - c)$

ADMM APPLICABLE TO LASSO

(LEAST ABSOLUTE SHRINKING AND SELECTION OPERATOR)

$$\min \frac{1}{2} \|Ex - b\|_2^2 + \tau \|x\|_1$$

$$\downarrow$$

$$\min \frac{1}{2} \|Ex - b\|_2^2 + \tau \|z\|_1$$

$$\text{s.t.} \quad x - z = 0$$