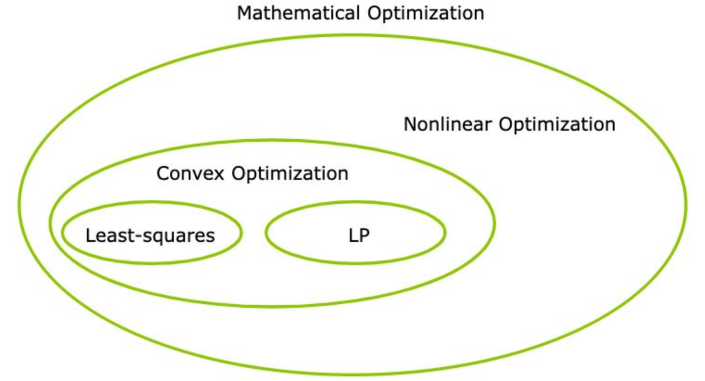


Optimisation (Basic)

V2020.11.25

(Dr Yan Xu)

Basic Definitions

1. Convex/Concave/Affine

affine	$f(\theta x+(1-\theta)y)=\theta f(x)+(1-\theta)f(y),\ \forall x,\ y,\ \theta\in[0,1]$
convex	$f(\theta x+(1-\theta)y)\le\theta f(x)+(1-\theta)f(y),\ \forall x,\ y,\ \theta\in[0,1]$
concave	$f(\theta x+(1-\theta)y)\ge\theta f(x)+(1-\theta)f(y),\ \forall x,\ y,\ \theta\in[0,1]$

2. Standard Convex Optimization (CP)

$$\begin{array}{ll}\text{minimize} & f_0(x)\\ \text{subject to} & f_i(x)\le 0,\quad i=1,\dots,m\\ & Ax=b\end{array}$$

3. Least-squares

$$\text{minimize}\quad \|Ax-b\|_2^2$$

4. Linear programming (LP)

$$\begin{array}{ll}\text{minimize} & c^Tx\\ \text{subject to} & a_i^Tx\le b_i,\quad i=1,\dots,m\end{array}$$

5. Quadratic program (QP)

$$\begin{array}{ll}\text{minimize} & (1/2)x^TPx+q^Tx+r\\ \text{subject to} & Gx\preceq h\\ & Ax=b\end{array}$$

- $P\in\mathbf{S}_+^n$, so objective is convex quadratic

6. Quadratically constrained quadratic program (QCQP)

Only difference from QP is convex quadratic inequality constraints

7. Feasibility problem (Optimisation View)

$$\begin{array}{ll}\text{minimize} & 0\\ \text{subject to} & f_i(x)\le 0,\quad i=1,\dots,m\\ & h_i(x)=0,\quad i=1,\dots,p\end{array}$$

8. Nonlinear programming
   - Nonconvex problems
   - Local optimization vs global optimization

Convex Sets

1. Affine set
2. Convex set
3. Convex combination

convex combination of $x_1,\dots,x_k$: any point x of the form

$$x=\theta_1x_1+\theta_2x_2+\cdots+\theta_kx_k$$

with $\theta_1+\cdots+\theta_k=1$, $\theta_i\ge 0$

4. Convex hull (or Conv(S))

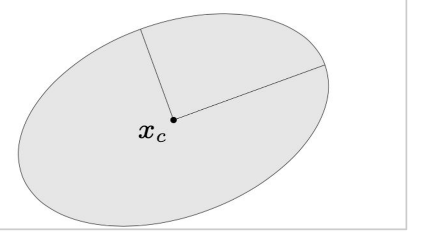
The smallest convex set to contain all convex combinations of points in S

5. Convex cone
6. Hyperplane & Halfspace
7. Ellipsoid

ellipsoid: set of the form

$$\{x\mid (x-x_c)^TP^{-1}(x-x_c)\le 1\}$$

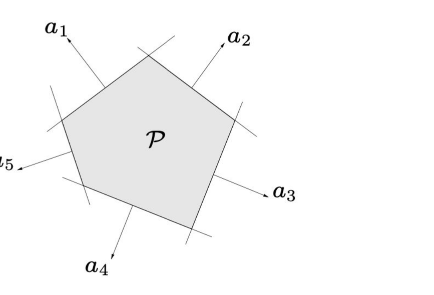
with $P\in\mathbf{S}_{++}^n$ (*i.e.*, P symmetric positive definite)



8. Norm balls

norm ball with center x_c and radius r: $\{x\mid \|x-x_c\|\le r\}$

9. Polyhedra



polyhedron is intersection of finite number of halfspaces and hyperplanes

Set Operations (Preserve Convexity)

1. The intersection of convex sets
2. Affine function
   - scaling, translation, projection
3. Perspective function

Theorem for Convex Sets:
1: Separating Hyperplane Theorem (for 2 disjoint convex sets)
2: Supporting Hyperplane Theorem (for every boundary point)

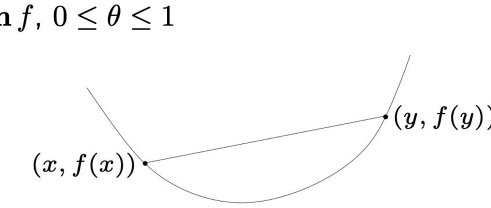
Convex Functions

1. Definition

$f:\mathbf{R}^n\to\mathbf{R}$ is convex if $\mathbf{dom}\,f$ is a convex set and

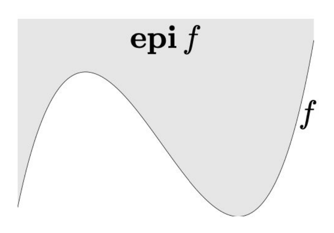
$$f(\theta x+(1-\theta)y)\le\theta f(x)+(1-\theta)f(y)$$

for all $x,y\in\mathbf{dom}\,f$, $0\le\theta\le 1$



2. Common Convex Functions
   - exp(x)
   - logistic(x)
   - max(X)
   - norm(X, k=1,2,...)
   - others ...
3. Common Concave Functions
   - log(x)
   - entropy(x)
   - min(X)
   - others ...
4. Epigraph

$$\mathbf{epi}\,f=\{(x,t)\in\mathbf{R}^{n+1}\mid x\in\mathbf{dom}\,f,\ f(x)\le t\}$$



f is convex if and only if $\mathbf{epi}\,f$ is a convex set

Rules for Convex Functions:
1: Any locally optimal point is globally optimal
2: first-order approximation of f(x) is global underestimator for all x
2: $\nabla^2f(x)\succeq 0$ for all $x\in\mathbf{dom}\,f$
3: −f(x) is concave if f(x) is convex

Function Operations (Preserve Convexity)

1. Nonnegative weighted sum
2. Composition with affine function

$f(Ax+b)$ is convex if f is convex

3. Pointwise maximum

if $f_1,\dots,f_m$ are convex, then $f(x)=\max\{f_1(x),\dots,f_m(x)\}$ is convex

4. Pointwise supremum (or least upper bound)
5. Examples (convex): $f(x)=-\sum_{i=1}^m\log(b_i-a_i^Tx)$,

Convex Functions (Composition Rules)

$f(\text{expr}_1,\text{expr}_2,\dots,\text{expr}_n)$ is convex if f is a convex function and for each expr_i one of the following conditions holds:

- f is increasing in argument i and expr_i is convex.
- f is decreasing in argument i and expr_i is concave.
- expr_i is affine or constant.

$f(\text{expr}_1,\text{expr}_2,\dots,\text{expr}_n)$ is concave if f is a concave function and for each expr_i one of the following conditions holds:

- f is increasing in argument i and expr_i is concave.
- f is decreasing in argument i and expr_i is convex.
- expr_i is affine or constant.

Log-concave

a positive function f is log-concave if $\log f$ is concave:

$$f(\theta x+(1-\theta)y)\ge f(x)^\theta f(y)^{1-\theta}\quad\text{for } 0\le\theta\le 1$$

- Most common probability densities are log-concave.

Some Interesting Convex Problems

1. Regularized approximation (e.g. L2 norm)

$$\text{minimize}\quad \|Ax-b\|_2^2+\delta\|x\|_2^2$$

can be solved as a least-squares problem

$$\text{minimize}\quad \left\|\begin{bmatrix}A\\ \sqrt{\delta}I\end{bmatrix}x-\begin{bmatrix}b\\ 0\end{bmatrix}\right\|_2^2$$

solution $x^\star=(A^TA+\delta I)^{-1}A^Tb$

2. Penalty function approximation

$$\begin{array}{ll}\text{minimize} & \phi(r_1)+\cdots+\phi(r_m)\\ \text{subject to} & r=Ax-b\end{array}$$

($A\in\mathbf{R}^{m\times n}$, $\phi:\mathbf{R}\to\mathbf{R}$ is a convex penalty function)

Example penalty: Square(), Norm(), log-barrier(), Huber()

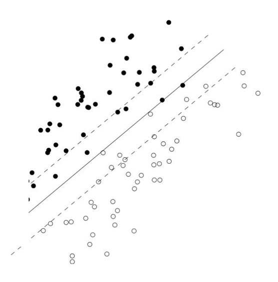
3. Maximum likelihood estimation (MLE)

$$\text{maximize (over } x)\quad \log p_x(y)$$

- y is observed value
- $l(x)=\log p_x(y)$ is called log-likelihood function

4. Support vector classifier

$$\begin{array}{ll}\text{minimize} & \|a\|_2+\gamma(\mathbf{1}^Tu+\mathbf{1}^Tv)\\ \text{subject to} & a^Tx_i+b\ge 1-u_i,\quad i=1,\dots,N\\ & a^Ty_i+b\le -1+v_i,\quad i=1,\dots,M\\ & u\succeq 0,\quad v\succeq 0\end{array}$$



- trade-off curve between inverse of margin 2/ // a //$_2$ and classification error

Duality

1. The Primal Problem

standard form problem (not necessarily convex)

$$\begin{array}{ll}\text{minimize} & f_0(x)\\ \text{subject to} & f_i(x)\le 0,\quad i=1,\dots,m\\ & h_i(x)=0,\quad i=1,\dots,p\end{array}$$

variable $x\in\mathbf{R}^n$, domain $\mathcal{D}$, optimal value $p^\star$

2. Lagrangian

Lagrangian: $L:\mathbf{R}^n\times\mathbf{R}^m\times\mathbf{R}^p\to\mathbf{R}$, with $\mathbf{dom}\,L=\mathcal{D}\times\mathbf{R}^m\times\mathbf{R}^p$,

$$L(x,\lambda,\nu)=f_0(x)+\sum_{i=1}^m\lambda_if_i(x)+\sum_{i=1}^p\nu_ih_i(x)$$

- weighted sum of objective and constraint functions
- λ_i is Lagrange multiplier associated with $f_i(x)\le 0$
- ν_i is Lagrange multiplier associated with $h_i(x)=0$

3. Dual Function (or greatest lower bound)

Lagrange dual function: $g:\mathbf{R}^m\times\mathbf{R}^p\to\mathbf{R}$,

$$\begin{aligned}g(\lambda,\nu)&=\inf_{x\in\mathcal{D}}L(x,\lambda,\nu)\\ &=\inf_{x\in\mathcal{D}}\left(f_0(x)+\sum_{i=1}^m\lambda_if_i(x)+\sum_{i=1}^p\nu_ih_i(x)\right)\end{aligned}$$

g is concave, can be $-\infty$ for some λ, ν

4. Dual Lower Bound

lower bound property: if $\lambda\succeq 0$, then $g(\lambda,\nu)\le p^\star$

5. Lagrange dual problem (or The Dual Problem)

$$\begin{array}{ll}\text{maximize} & g(\lambda,\nu)\\ \text{subject to} & \lambda\succeq 0\end{array}$$

- finds best lower bound on $p^\star$, obtained from Lagrange dual function
- a convex optimization problem; optimal value denoted $d^\star$

6. Duality
   - weak duality (d* ≤ p*)
   - strong duality (d* = p*)
     - usually holds for convex problems
7. An example

primal problem

$$\begin{array}{ll}\text{minimize} & c^Tx\\ \text{subject to} & Ax\preceq b\end{array}$$

dual function

$$g(\lambda)=\inf_x\left((c+A^T\lambda)^Tx-b^T\lambda\right)=\begin{cases}-b^T\lambda & A^T\lambda+c=0\\ -\infty & \text{otherwise}\end{cases}$$

dual problem

$$\begin{array}{ll}\text{maximize} & -b^T\lambda\\ \text{subject to} & A^T\lambda+c=0,\quad \lambda\succeq 0\end{array}$$

Reformulating the primal problem can be useful when the dual is difficult to derive, or uninteresting. (e.g. new variables / transform objectives)

Optimal Verification (KKT conditions)

the following four conditions are called KKT conditions (for a problem with differentiable f_i, h_i):

1. primal constraints: $f_i(x)\le 0$, $i=1,\dots,m$, $h_i(x)=0$, $i=1,\dots,p$
2. dual constraints: $\lambda\succeq 0$
3. complementary slackness: $\lambda_if_i(x)=0$, $i=1,\dots,m$
4. gradient of Lagrangian with respect to x vanishes:

$$\nabla f_0(x)+\sum_{i=1}^m\lambda_i\nabla f_i(x)+\sum_{i=1}^p\nu_i\nabla h_i(x)=0$$

If The Primal Problem is Convex, x, λ, ν which satisfy KKT are optimal:

Unconstrained Minimization Solvers

1. Descent Methods (or Gradient Descent)

given a starting point $x\in\mathbf{dom}\,f$.
repeat
1. $\Delta x:=-\nabla f(x)$.
2. *Line search.* Choose step size t via exact or backtracking line search.
3. *Update.* $x:=x+t\Delta x$.

until stopping criterion is satisfied.

   - Δx is the step, or search direction; t is the step size
   - stopping criterion usually of the form // ∇f(x) //$_2$ ≤ ε
   - very simple, but could be very slow

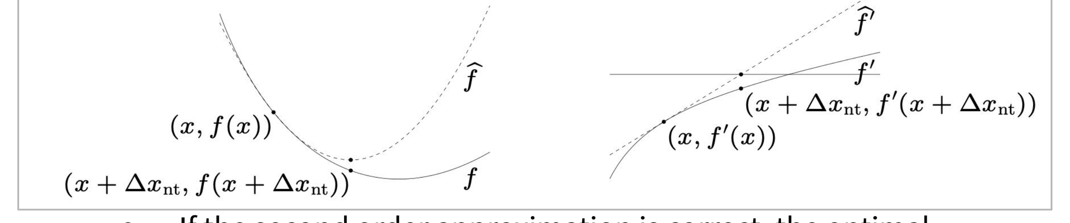
2. Newton's Method

given a starting point $x\in\mathbf{dom}\,f$, tolerance $\epsilon>0$.
repeat
1. *Compute the Newton step and decrement.*
$\Delta x_{\mathrm{nt}}:=-\nabla^2f(x)^{-1}\nabla f(x)$; $\quad\lambda^2:=\nabla f(x)^T\nabla^2f(x)^{-1}\nabla f(x)$.
2. *Stopping criterion.* **quit** if $\lambda^2/2\le\epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x:=x+t\Delta x_{\mathrm{nt}}$.

3. Why Newton's Method is Fast?

- $x+\Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v)\approx\nabla\widehat{f}(x+v)=\nabla f(x)+\nabla^2f(x)v=0$$



   - If the second order approximation is correct, the optimal solution could be found in 1 step with 1 as the step size.
   - In most cases, converges in 5-20 steps.

4. Hessian Matrix Inverses
   - The mostly costly operation in each step and typically follow a Cholesky Factorization approach (complexity: n^3)

Equality Constrained Minimization Solvers

1. The Problem

$$\begin{array}{ll}\text{minimize} & f(x)\\ \text{subject to} & Ax=b\end{array}$$

- f convex, twice continuously differentiable

2. Newton's Method (with a feasible starting point)

given starting point $x\in\mathbf{dom}\,f$ with $Ax=b$, tolerance $\epsilon>0$.
repeat
1. Compute the Newton step and decrement Δx_{nt}, $\lambda(x)$.
2. *Stopping criterion.* **quit** if $\lambda^2/2\le\epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x:=x+t\Delta x_{\mathrm{nt}}$.

3. Newton's Method (with an infeasible starting point)

given starting point $x\in\mathbf{dom}\,f$, ν, tolerance $\epsilon>0$, $\alpha\in(0,1/2)$, $\beta\in(0,1)$.
repeat
1. Compute primal and dual Newton steps Δx_{nt}, $\Delta\nu_{\mathrm{nt}}$.
2. *Backtracking line search on* $\|r\|_2$.
$t:=1$.
while $\|r(x+t\Delta x_{\mathrm{nt}},\nu+t\Delta\nu_{\mathrm{nt}})\|_2>(1-\alpha t)\|r(x,\nu)\|_2$, $\quad t:=\beta t$.
3. *Update.* $x:=x+t\Delta x_{\mathrm{nt}}$, $\nu:=\nu+t\Delta\nu_{\mathrm{nt}}$.

until $Ax=b$ and $\|r(x,\nu)\|_2\le\epsilon$.

   - not a descent method, f(x) may increase in the next step

4. How to calculate Δx, Δv?

$$\begin{bmatrix}\nabla^2f(x) & A^T\\ A & 0\end{bmatrix}\begin{bmatrix}\Delta x_{\mathrm{nt}}\\ \Delta\nu_{\mathrm{nt}}\end{bmatrix}=-\begin{bmatrix}\nabla f(x)+A^T\nu\\ Ax-b\end{bmatrix}$$

Inequality Constrained Minimization Solvers

1. The Problem

$$\begin{array}{ll}\text{minimize} & f_0(x)\\ \text{subject to} & f_i(x)\le 0,\quad i=1,\dots,m\\ & Ax=b\end{array}$$

- f_i convex, twice continuously differentiable

2. Approximation via logarithmic barrier

$$\begin{array}{ll}\text{minimize} & f_0(x)-(1/t)\sum_{i=1}^m\log(-f_i(x))\\ \text{subject to} & Ax=b\end{array}$$

approximation improves as t → ∞

3. Logarithmic barrier function
4. Barrier method (need to tune t^0 and u)

given strictly feasible x, $t:=t^{(0)}>0$, $\mu>1$, tolerance $\epsilon>0$.
repeat
1. *Centering step.* Compute $x^\star(t)$ by minimizing $tf_0+\phi$, subject to $Ax=b$.
2. *Update.* $x:=x^\star(t)$.
3. *Stopping criterion.* **quit** if $m/t<\epsilon$.
4. *Increase* t. $t:=\mu t$.

5. Primal-dual interior-point methods

Library 1: scipy.optimize

1. Minimize

minimize(fun, x0, args=(), method=None, jac=None, hess=None, hessp=None, bounds=None, constraints=(), tol=None, callback=None, options=None)

- fun(): the objective function
- X0: initial point
- Method (popular):
  - BFGS-B
  - L-BFGS-B
  - SLSQP
- jac: Method for computing the gradient vector
- hess: Method for computing the Hessian matrix
- bounds: Bounds on variables
- constraints: Array of constraint objects (linear & nonlinear)
  - mostly for SLSQP
- options: maxiter & disp

Note: parameters are Method dependent.

2. Least Squares

least_squares(fun, x0, jac='2-point', bounds=- inf, inf, method='trf', ftol=1e-08, xtol=1e-08, gtol=1e-08, x_scale=1.0, loss='linear', f_scale=1.0, diff_step=None, tr_solver=None, tr_options={}, jac_sparsity=None, max_nfev=None, verbose=0, args=(), kwargs={})

- fun(): the residual function
- loss: the loss function (to reduce the influence of outliers)
- X0: initial point

3. Curve Fit (or general parameter estimation)

*curve_fit(fun, xdata, ydata, p0=None, sigma=None, absolute_sigma=False, check_finite=True, bounds=- inf, inf, method=None, jac=None, **kwargs)*

- fun(): the target function
(the independent variable as the first argument and the parameters to fit as separate remaining arguments)
- xdata: input of an (k,M)-shaped array
- ydata: output of length M array
- Bounds: parameter bounds

Library 2: CVXPY (focusing convex optimization)

1. Basic Concepts (Objects)
- cvxpy.Parameter
- cvxpy.Variable
- Cost Function & Constraints
- cvxpy.Problem & cvx.Minimize
- prob.solve()

2. Code Example (TensorFlow style)

```
import cvxpy as cvx
# Create two scalar optimization variables (CVXPY Variable)
x = cvx.Variable()
y = cvx.Variable()
# Create two constraints (Python list)
constraints = [x + y == 1, x - y >= 1]
# Form objective
obj = cvx.Minimize(cvx.square(x - y))
# Form and solve problem
prob = cvx.Problem(obj, constraints)
prob.solve()  # Returns the optimal value.
print("status:", prob.status)
print("optimal value", prob.value)
print("optimal var", x.value, y.value)
```