2. Existence of optimal solutions and optimality conditions for unconstrained problems

Th. Weierstrass - If the objective function f is continuous and the feasible region X is closed and bounded, then (at least) a global optimum exists

> **Corollary 2** - If the objective function f is continuous, the feasible region X is closed and there exists k € R such that the k-sublevel set **S_k is nonempty and bounded**, then (at least) a global optimum exists.
>
> **Corollary 3** - If the objective function f is continuous and **coercive** (lim f(x) → inf) and the feasible region X ≠ 0 is closed, then (at least) a global optimum exists.

Existence in the presence of convexity assumptions

Theorem 1 - Assume that f is convex on the convex set X. Then any local optimum of (P) is a **global optimum.**

> **Proposition 1** - f strictly convex on the convex set X and (P) admits a global optimum x*. Then x* is the unique optimal solution of (P).

Theorem 2 - If f is strongly convex on R^n and X is closed, then there exists a global optimum.

> **Corollary 1** - If f is strongly convex on R^n and X is closed and convex, then there exists a **unique** global optimum.

Optimality conditions for unconstrained problems

Theorem 3 (necessary optimality condition) - Assume that X is an open set and let f be differentiable at x* € X. If x* is a **local optimum** of (P) then ∇f(x*) = 0.

Theorem 4 (second order necessary optimality condition) - X open set and x* € X is a **local optimum** for (P). Then these conditions hold:

- ∇f(x*)=0
- the Hessian matrix ∇^2f(x*)=0 is positive **semidefinite**

Theorem 5 (second order sufficient optimality condition)- Let X be an open set, x* € X and assume the following conditions hold:

- ∇f(x*)=0
- the Hessian matrix ∇^2f(x*)=0 is positive **definite**

→ x* is a **local optimum** for (P)

Theorem 6 (optimality condition for convex problems) - Let f be a differentiable convex function on the open convex set X, then x* € X is a **global optimum** for (P) if and only if ∇f(x*)=0.

Theorem 7 - Let f be a differentiable strictly convex function on the open convex set X, then x* € X is a **unique global optimum** for (P) if and only if ∇f(x*)=0.

Corollary 2 - There exists a **global optimum** for (P) if and only if:

- Qx* + c = 0; (i)
- Q is positive semidefinite

Remark - We observed that if Q is positive definite then (P) admits a unique global optimum. Indeed, in such a case Q is nonsingular and the system in (i) admits a unique solution x* = -Q^{-1}*c.

3. Unconstrained optimization methods

Gradient Method (exact line search) -

1) Choose $x^0 \in R^n$, set k = 0
2) If $\nabla f(x^k)=0$, STOP, otherwise go to step 3
3) Let $d^k = -\nabla f(x^k)$ (search direction). Compute an optimal solution t_k of the problem min[t>0] $f(x^k + td^k)$

   Set $x^{k+1} = x^k + t_k d^k$

   k = k + 1

   Go to step 2)

Theorem - If f is **coercive** , then for any starting point x^0 the generated sequence $\{x^k\}$ is bounded and any of its cluster points is a **stationary point** of f.

> **Corollary** - if f is coercive and **convex**, then for any starting point x^0 the generated sequence $\{x^k\}$ is bounded and any of its cluster points is a **global minimum** of f
>
> **Corollary** - if f is **strongly convex**, then for any starting point x^0 the generated sequence $\{x^k\}$ converges to the **unique global minimum** of f.

Exercise - Implement the gradient method for solving the problem $\{\min 1/2x^TQx+c^Tx\}$. This is a quadratic function, so we can use the exact line search method.

```
%% Problema definition
Q=[6 0 -4 0;0 6 0 -4;-4 0  6 0;0 -4 0 6]
c = [ 1 -1 2 -3]';
disp('eigenvalues of  Q:')
eig(Q)
%% Parameters
x0 = [0 0 0 0]';
tolerance = 10^(-6);
%% Gradient method with exact line search
% starting point
x = x0 ;
X=[Inf,Inf,Inf,Inf,Inf,Inf,Inf];
for ITER=1:1000
   v = 0.5*x'*Q*x + c'*x;
   g = Q*x + c ;
   X=[X;ITER,x',v,norm(g)];
   % stopping criterion
   if norm(g) < tolerance
       break
   end

   %   search direction
   d = -g;
   %   exact line search
   t = norm(g)^2/(d'*Q*d) ;
   %   new point
   x = x + t*d ;
end
disp('optimal solution')
x
disp('optimal value')
v
disp('gradient norm at the solution')
norm(g)
```

Gradient Method (Armijo inexact line search) -

1) Choose $x^0 \in R^n$, set k = 0
2) If $\nabla f(x^k)=0$, STOP, otherwise go to step 3
3) Let $d^k = -\nabla f(x^k)$ (search direction), t_k = t_bar

   while $f(x^k + td^k) > f(x^k) + \alpha t_k (d^k)^T \nabla f(x^k)$ do

   $t_k = \gamma * t_k$

   end

   Set $x^{k+1} = x^k + t_k d^k$

   k = k + 1

   Go to step 2)

Exercise - When f is not a quadratic function, the exact line search may be computationally expensive. We use the Armijo inexact line search.

```
% min f(x(1),x(2))= 2*x(1)^4 + 3*x(2)^4 +
2*x(1)^2 + 4*x(2)^2 + x(1)*x(2) - 3*x(1) -
2*x(2)
alpha = 0.1;
gamma = 0.9;
tbar = 1;
x0 = [0;0];
tolerance = 10^(-3) ;
%% method
%disp('Gradient method with Armijo inexact
line search');
x = x0 ;
for ITER=0:100
   [v, g] = f(x);

   % stopping criterion
   if norm(g) < tolerance
       break
   end

   % search direction
   d = -g;

   % Armijo inexact line search
   t = tbar ;
   while f(x+t*d) > v + alpha*g'*d*t
       t = gamma*t ;
   end

   % new point
   x = x + t*d;

end
x
v
norm(g)
function [v, g] = f(x)
v = 2*x(1)^4 + 3*x(2)^4 + 2*x(1)^2 + 4*x(2)^2
+ x(1)*x(2) - 3*x(1) - 2*x(2)  ;
g = [ 8*x(1)^3 + 4*x(1) + x(2) - 3;
    12*x(2)^3 + 8*x(2) + x(1) - 2];
end
```

Conjugate gradient method

1. Choose $x^0 \in \mathbb{R}^n$, set $g^0 = Q\,x^0 + c$, $k := 0$; go to Step 2.
2. Let $g^k = \nabla f(x^k)$. **If** $g^k = 0$ **then** STOP, **else** go to Step 3.
3. **If** $k = 0$ **then** $d^k = -g^k$

   else $\beta_k = \dfrac{(g^k)^\top Q\,d^{k-1}}{(d^{k-1})^\top Q\,d^{k-1}}, \quad d^k = -g^k + \beta_k\,d^{k-1}$

   $$t_k = -\frac{(g^k)^\top d^k}{(d^k)^\top Q\,d^k}$$

   $$x^{k+1} = x^k + t_k\,d^k,\ g^{k+1} = Q\,x^{k+1} + c,\ k = k+1$$

   Go to Step 2.

Theorem (Convergence)- The CG method finds the global minimum in at most n iterations. If Q has r distinct eigenvalues, then CG method finds the global minimum in at most r iterations.

Exercise -

```
% format short e
%% Quadratic Problem
% Problem definition
Q = [6 0 -4 0;0 6 0 -4;-4 0 6 0;0 -4 0 6]
c = [ 1 -1 2 -3]';
disp('Eigenvalues of Q:')
eig(Q)
%% Parameters
x0 = [0,0,0,0]';
tolerance = 10^(-6);
%% Conjugate Gradient method
% starting point
x = x0;
X=[Inf,Inf,Inf,Inf,Inf,Inf,Inf];
for ITER=1:10
   v = 0.5*x'*Q*x + c'*x;
   g = Q*x + c ;
   X=[X;ITER,x',v,norm(g)];
   % stopping criterion
   if norm(g) < tolerance
       break
   end
   %   search direction
   if ITER == 1
       d = -g;
   else
                                               beta     =
(g'*Q*d_prev)/(d_prev'*Q*d_prev);
       d = -g + beta*d_prev;
   end

   %   step size
   t = (-g'*d)/(d'*Q*d);

   %   new point

   x = x + t*d;
   d_prev = d ;
end
x
v
norm(g)
ITER
```

Newton Method (basic version) -

1. Let $x^0 \in \mathbb{R}^n$, set $k = 0$. Go to Step 2.
2. **If** $\nabla f(x^k) = 0$ then STOP **else** go to Step 3.
3. Let d^k be the solution of the linear system $\nabla^2 f(x^k)d = -\nabla f(x^k)$.
   Set $x^{k+1} = x^k + d^k$, $k = k + 1$ and go to Step 2.

Newtown Method (inexact line search) - If f is strongly convex, then we have global convergence because dk is a descent direction. If f is **strongly convex**, then for any starting point x0 ∈ Rn the sequence {xk} converges to the **global minimum** of f . Moreover, if α ∈ (0, 1/2) and tbar = 1 then the convergence is quadratic.

1. Let $\alpha, \gamma \in (0,1)$, $\bar{t} > 0$, $x^0 \in \mathbb{R}^n$, set $k = 0$. Go to Step 2.
2. **If** $\nabla f(x^k) = 0$ then STOP **else** go to Step 3.
3. Let d^k be the solution of the linear system $\nabla^2 f(x^k)d = -\nabla f(x^k)$.
   Set $t_k = \bar{t}$
   while $f(x^k + t_k\, d^k) > f(x^k) + \alpha\, t_k\, (d^k)^T \nabla f(x^k)$ **do**
   $t_k = \gamma t_k$
   end
   Set $x^{k+1} = x^k + t_k\, d^k$, $k = k + 1$
   Go to Step 2.

Exercise -

```
%% data
alpha=0.1;
gamma=0.9;
tbar =1;
x0 = [0;0];
tolerance = 10^(-3) ;
x = x0 ;
%X=[Inf,Inf,Inf,Inf,Inf];
for ITER=0:100
   [v, g, H] = f(x);
%    X=[X;ITER,x',v,norm(g)];
   % stopping criterion
   if norm(g) < tolerance
       break
   end

   % search direction
   d = -inv(H)*g;
   t=tbar;
   while (f(x+t*d) > f(x)+alpha*t*d'*g)
       t=gamma*t;
   end
   % new point
   x = x + t*d;

end
x
v
norm(g)
function [v, g, H] = f(x)
v = 2*x(1)^4 + 3*x(2)^4 + 2*x(1)^2 + 4*x(2)^2 + x(1)*x(2) - 3*x(1) - 2*x(2)  ;
g = [ 8*x(1)^3 + 4*x(1) + x(2) - 3
     12*x(2)^3 + 8*x(2) + x(1) - 2];
H = [ 24*x(1)^2+4    1
          1        36*x(2)^2+8];
end
```

4. KKT optimality conditions and Lagrangian duality

Theorem 1 (Sufficient conditions for ACQ) -

a) (Affine constraints) If g_j and h_k are **affine** for all j = 1,...,m and k = 1,...,p, then ACQ holds at any x ∈ X.
b) (Slater condition for convex problems) If g_j are **convex** for all j = 1,...,m, h_k are **affine** for all k = 1,...,p and there exists xbar € X s.t. g(xbar)< 0 and h(xbar)=0, then ACQ holds at any x € X.
c) (Linear independence of the gradients of active constraints) If x* ∈ X and the vectors
   i) $\nabla g_j(x^*)$ for j ∈ $A(x^*)$,
   ii) $\nabla h_k(x^*)$ for k = 1,...,p
   are linearly independent, then ACQ holds at x^*.

Theorem 2 (KKT)

If x^* is a local minimum and ACQ holds at x^*, then there exist $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^p$ s.t. (x^*, λ^*, μ^*) satisfies the KKT system:

$$\begin{cases} \nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \mu_j^* \nabla h_j(x^*) = 0 \\ \lambda_i^* g_i(x^*) = 0 \qquad \forall\, i = 1, \dots, m \\ \lambda^* \geq 0 \\ g(x^*) \leq 0 \\ h(x^*) = 0 \end{cases}$$

Note that ACQ assumption is crucial in the KKT Theorem, in fact KKT Theorem gives **necessary** optimality conditions, but not sufficient ones!

Theorem 3 - If the optimization problem is **convex** and (x*,λ*,μ*) solves KKT system, then x* is a **global optimum.**

Lagrangian relaxation -

Given $\lambda \geq 0$ and $\mu \in \mathbb{R}^p$, the problem

$$\begin{cases} \inf \ L(x, \lambda, \mu) \\ x \in \mathbb{R}^n \end{cases}$$

is called Lagrangian relaxation of (P) and
$\varphi(\lambda, \mu) = \inf_{x \in \mathbb{R}^n} L(x, \lambda, \mu)$ is the Lagrangian dual function.

The dual function φ:
- is concave because inf of affine functions w.r.t (λ, μ)
- may be equal to -∞ at some point
- may be not differentiable at some point

Lagrangian dual problem -

The problem

$$\begin{cases} \max \ \varphi(\lambda, \mu) \\ \lambda \geq 0 \end{cases}$$

is called Lagrangian dual problem of (P) [and (P) is called primal problem].

The dual problem (D) consists in finding the best lower bound of v(P). (D) is always equivalent to a convex problem, even if (P) is a non-convex problem, indeed, it is a maximization of a concave function on a convex set.

Theorem 5 -

Suppose f, g, h **continuously differentiable**, the primal problem (P) is **convex**, there exists a global optimum c* and **ACQ holds** at x*. Then:

- **Strong duality** holds ($v(D) = v(P)$ and (D) admits an optimal solution
- (λ^*, μ^*) is **optimal for (D)** if and only if (λ^*, μ^*) is a KKT multipliers vector associated with x^*.

Theorem 6 (characterization of strong duality) -

(x^*, λ^*, μ^*) is a saddle point of L, i.e.

$$L(x^*, \lambda, \mu) \leq L(x^*, \lambda^*, \mu^*) \leq L(x, \lambda^*, \mu^*) \qquad \forall\, x \in \mathbb{R}^n,\ \lambda \in \mathbb{R}^m_+,\ \mu \in \mathbb{R}^p,$$

if and only if x^* is optimum of (P), (λ^*, μ^*) is optimum of (D) and $v(P) = v(D)$.

5. Support Vector Machines for supervised classification problems

We are given a set of vectors of data (objects) partitioned in several classes with known labels, we want to **assign to a suitable class** a new object with **unknown label.**

Margin of separation -

If H is a separating hyperplane, then the margin of separation of H is defined as the minimum distance between H and $A \cup B$, i.e.

$$\rho(H) = \min_{x \in A \cup B} \frac{|w^\mathsf{T} x + b|}{\|w\|}.$$

Theorem - Finding the separating hyperplane with the maximum margin of separation is equivalent to solve the following convex quadratic programming problem:

$$\begin{cases} \min\limits_{w,b} \ \frac{1}{2}\|w\|^2 \\ w^\mathsf{T} x^i + b \geq 1 \qquad \forall\, x^i \in A \\ w^\mathsf{T} x^j + b \leq -1 \qquad \forall\, x^j \in B \end{cases}$$

Exercise - Find the separating hyperplane with maximum margin for the data set (A and B sets provided). Since the problem is quadratic, it is defined by

$$\begin{cases} \min\limits_{w,b} \ \frac{1}{2}(w,b)^T C \begin{pmatrix} w \\ b \end{pmatrix} \\ D \begin{pmatrix} w \\ b \end{pmatrix} \leq d \end{cases}$$

where, assuming $n = 2$, $w \in \mathbb{R}^2$, $b \in \mathbb{R}$,

$$C = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad D = \begin{pmatrix} -A & -e_m \\ B & e_p \end{pmatrix} \quad d = \begin{pmatrix} -e_m \\ -e_p \end{pmatrix}$$

$$-e_m = (-1, -1, ..., -1)^T \in \mathbb{R}^m, \quad -e_p = (-1, -1, ..., -1)^T \in \mathbb{R}^p$$

```
nA = size(A,1);
nB = size(B,1);
% training points
T = [A ; B];
%% Linear SVM - primal model
% define the optimization problem
Q = [ 1 0 0 ;
      0 1 0 ;
      0 0 0 ];
 D = [-A -ones(nA,1);
      B ones(nB,1) ] ;
 d = -ones(nA+nB,1) ;
% solve the problem
sol = quadprog(Q,zeros(3,1),D,d);
w = sol(1:2)
b = sol(3)
% plot the solution
xx = 0:0.1:10 ;
uu = (-w(1)/w(2)).*xx - b/w(2);
vv = (-w(1)/w(2)).*xx + (1-b)/w(2);
vvv = (-w(1)/w(2)).*xx + (-1-b)/w(2);
plot(A(:,1),A(:,2),'bo',B(:,1),B(:,2),'ro',xx,
uu,'k-',xx,vv,'b-',xx,vvv,'r-','Linewidth',1.5
)
axis([0 10 0 10])
```

Linear SVM -

$$\begin{cases} \min\limits_{w,b} \ \dfrac{1}{2}\|w\|^2 \\ 1 - y^i(w^\mathsf{T} x^i + b) \leq 0 \qquad \forall\, i = 1, \dots, \ell \end{cases}$$

Dual formulation -

$$\begin{cases} \max\limits_{\lambda} \ -\dfrac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell} y^i y^j (x^i)^\mathsf{T} x^j \lambda_i \lambda_j + \sum\limits_{i=1}^{\ell} \lambda_i \\ \sum\limits_{i=1}^{\ell} \lambda_i y^i = 0 \\ \lambda \geq 0 \end{cases}$$

Since X^TX is always positive semidefinite then the dual problem is convex quadratic programming problem; a KKT multiplier λ^* associated to the primal optimum (w^*,b^*) is a **dual optimum**; if $\lambda_i^* > 0$, then x^i is said **support vector**;

If λ^* is a dual optimum, then, by (9), we have:

$$w^* = \sum_{i=1}^{\ell} \lambda_i^* y^i x^i;$$

b^* is obtained using the complementarity conditions:

$$\lambda_i^* \left[1 - y^i((w^*)^\mathsf{T} x^i + b^*)\right] = 0;$$

in fact, if i is such that $\lambda_i^* > 0$, then $b^* = \frac{1}{y^i} - (w^*)^\mathsf{T} x^i$.

This allows us to find the separating hyperplane $(w^*)^\mathsf{T} x + b^* = 0$ and the decision function

$$f(x) = \operatorname{sign}((w^*)^\mathsf{T} x + b^*).$$

Exercise - Find the separating hyperplane with maximum margin for the data set by solving the dual problem:

$$\begin{cases} -\min\limits_{\lambda} \dfrac{1}{2} \sum\limits_{i=1}^{\ell} \sum\limits_{j=1}^{\ell} y^i y^j (x^i)^\mathsf{T} x^j \lambda_i \lambda_j - \sum\limits_{i=1}^{\ell} \lambda_i \\ \sum\limits_{i=1}^{\ell} \lambda_i y^i = 0 \\ \lambda \geq 0 \end{cases}$$

where the generic component q_{ij} of the hessian matrix Q is given by $q_{ij} = y^i y^j (x^i)^T x^j$

```
nA = size(A,1);
nB = size(B,1);
% training points
T = [A ; B];
%% Linear SVM - dual model
% define the problem
y = [ones(nA,1) ; -ones(nB,1)]; % labels
l = length(y);
Q = zeros(l,l);
for i = 1 : l
   for j = 1 : l
       Q(i,j) = y(i)*y(j)*(T(i,:))*T(j,:)' ;
   end
end
% solve the problem
la                                              =
quadprog(Q,-ones(l,1),[],[],y',0,zeros(l,1),[]
);
% compute vector w
wD = zeros(2,1);
for i = 1 : l
   wD = wD + la(i)*y(i)*T(i,:)';
end
wD
% compute scalar b
ind = find(la > 1e-3) ;
i = ind(1) ;
bD = 1/y(i) - wD'*T(i,:)'
% plot the solution
xx = 0:0.1:10 ;
uuD = (-wD(1)/wD(2)).*xx - bD/wD(2);
vvD = (-wD(1)/wD(2)).*xx + (1-bD)/wD(2);
vvvD = (-wD(1)/wD(2)).*xx + (-1-bD)/wD(2);
figure
plot(A(:,1),A(:,2),'bo',B(:,1),B(:,2),'ro',...

xx,uuD,'k-',xx,vvD,'b-',xx,vvvD,'r-','Linewidt
h',1.5)
axis([0 10 0 10])
title('Optimal  separating  hyperplane  (dual
model)')
```

Linear SVM (soft margin) -

If sets A and B are not linearly separable we introduce slack variables and consider the relaxed system:

$$\begin{aligned} &1 - y^i(w^\mathsf{T} x^i + b) \leq \xi_i && i = 1, \dots, \ell \\ &\xi_i \geq 0 && i = 1, \dots, \ell \end{aligned}$$

So the linear SVM with soft margin model will be defined like this:

$$\begin{cases} \min\limits_{w,b,\xi} \dfrac{1}{2} \|w\|^2 + C \sum\limits_{i=1}^{\ell} \xi_i \\ 1 - y^i(w^\mathsf{T} x^i + b) \leq \xi_i & \forall\, i = 1, \dots, \ell \\ \xi_i \geq 0 & \forall\, i = 1, \dots, \ell \end{cases}$$

Dual formulation -

$$\begin{cases} \max\limits_{\lambda} \ -\frac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell} y^i y^j (x^i)^\mathsf{T} x^j \lambda_i \lambda_j + \sum\limits_{i=1}^{\ell} \lambda_i \\ \sum\limits_{i=1}^{\ell} \lambda_i y^i = 0 \\ 0 \le \lambda_i \le C \qquad i = 1,\dots,\ell \end{cases}$$

If λ^* is a dual optimum, then $w^* = \sum\limits_{i=1}^{\ell} \lambda_i^* y^i x^i$. We can find b^* by choosing i s.t. $0 < \lambda_i^* < C$ and using the complementarity conditions (…), thus: $b^* = \frac{1}{y^i} - (w^*)^\mathsf{T} x^i$.

Exercise - Find the separating hyperplane with soft margin for the following data set by solving the dual problem with C = 10. Compute the vector ξ of the errors.

```
nA = size(A,1);
nB = size(B,1);
% training points
T = [A ; B];
%% Linear SVM - dual model (soft margin) - Exercise 5.4
% define the problem
C = 10 ;
y = [ones(nA,1) ; -ones(nB,1)]; % labels
l = length(y);
Q = zeros(l,l);
for i = 1 : l
   for j = 1 : l
       Q(i,j) = y(i)*y(j)*(T(i,:))*T(j,:)' ;
   end
end
% solve the problem
la = quadprog(Q,-ones(l,1),[],[],y',0,zeros(l,1),C*ones(l,1),[]);
% compute vector w
wD = zeros(2,1);
for i = 1 : l
  wD = wD + la(i)*y(i)*T(i,:)';
end
% compute scalar b
indpos = find(la > 10^(-3));
ind = find(la(indpos) < C - 10^(-3));
i = indpos(ind(1));
bD = 1/y(i) - wD'*T(i,:)';
%% plot the solution
xx = 0:0.1:10;
uuD = (-wD(1)/wD(2)).*xx - bD/wD(2);
vvD = (-wD(1)/wD(2)).*xx + (1-bD)/wD(2);
vvvD = (-wD(1)/wD(2)).*xx + (-1-bD)/wD(2);
plot(A(:,1),A(:,2),'bo',B(:,1),B(:,2),'r*',...
xx,uuD,'k-',xx,vvD,'b-',xx,vvvD,'r-','Linewidth',1)
axis([0 10 0 10])
title('Optimal separating hyperplane with soft margin')
% Compute the support vectors
supp = find(la > 10^(-3));
suppA = supp(supp <= nA);
suppB = supp(supp > nA);
% Compute the errors xi
for i=1:nA+nB
   if la(i) >0.001
       xi(i)= 1 - y(i)*(T(i,:)*wD +bD);
   else xi(i)=0;
   end
end
```

Nonlinear SVM (Primal problem) -

$$\begin{cases} \min\limits_{w,b,\xi} \ \frac{1}{2}\|w\|^2 + C\sum\limits_{i=1}^{\ell}\xi_i \\ 1 - y^i(w^\mathsf{T}\phi(x^i) + b) \le \xi_i \qquad \forall\ i = 1,\dots,\ell \\ \xi_i \ge 0 \qquad \forall\ i = 1,\dots,\ell \end{cases}$$

Dual formulation -

$$\begin{cases} \max\limits_{\lambda} \ -\frac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell} y^i y^j \phi(x^i)^\mathsf{T}\phi(x^j) \lambda_i \lambda_j + \sum\limits_{i=1}^{\ell} \lambda_i \\ \sum\limits_{i=1}^{\ell} \lambda_i y^i = 0 \\ 0 \le \lambda_i \le C \qquad \forall\ i = 1,\dots,\ell \end{cases}$$

Let λ^* be a solution of the dual problem, then $w^* = \sum\limits_{i=1}^{\ell} \lambda_i^* y^i \phi(x^i)$. and

$b^* = \frac{1}{y^i} - \sum_{j=1}^{\ell} \lambda_j^* y^j \phi(x^j)^T \phi(x^i)$, and the decision function will be given by

$$f(x) = \text{sign}((w^*)^T \phi(x) + b^*) = \text{sign}\left(\sum_{i=1}^{\ell} \lambda_i^* y^i \phi(x^i)^T \phi(x) + b^*\right)$$

Kernel function -

A function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is called kernel if there exists a map $\phi : \mathbb{R}^n \to \mathcal{H}$ such that

$$k(x,y) = \langle \phi(x), \phi(y) \rangle,$$

where $\langle \cdot, \cdot \rangle$ is a scalar product in the features space $\mathcal{H}$.

Theorem -

If $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a kernel and $x^1, \dots, x^\ell \in \mathbb{R}^n$, then the matrix K defined as follows

$$K_{ij} = k(x^i, x^j)$$

is positive semidefinite.

Dual formulation (with kernel function) -

$$\begin{cases} \max_{\lambda} \ -\frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} y^i y^j k(x^i, x^j) \lambda_i \lambda_j + \sum_{i=1}^{\ell} \lambda_i \\ \sum_{i=1}^{\ell} \lambda_i y^i = 0 \\ 0 \le \lambda_i \le C \qquad i = 1, \dots, \ell \end{cases}$$

Method -

- choose a kernel k
- find an optimal solution λ^* of the dual
- choose i s.t. $0 < \lambda_i^* < C$ and find b*: $b^* = \frac{1}{y^i} - \sum_{j=1}^{\ell} \lambda_j^* y^j k(x^i, x^j)$
- decision function $f(x) = \text{sign}\left(\sum_{i=1}^{\ell} \lambda_i^* y^i k(x^i, x) + b^*\right)$

Exercise - Find optimal separating surface for the following data set using a Gaussian kernel with parameters C = 1 and γ = 1.

```
nA = size(A,1);
nB = size(B,1);
% training points
T = [A ; B];
y = [ones(nA,1) ; -ones(nB,1)]; % labels
l = length(y);
%% Nonlinear SVM
% parameter
C = 1 ;
% Gaussian kernel
gamma = 1 ;
K = zeros(l,l);
for i = 1 : l
   for j = 1 : l
                                            K(i,j)     =
exp(-gamma*norm(T(i,:)-T(j,:))^2);
   end
end
% define the problem
Q = zeros(l,l);
for i = 1 : l
   for j = 1 : l
       Q(i,j) = y(i)*y(j)*K(i,j) ;
   end
end
% solve the problem
[la,ov]                                               =
quadprog(Q,-ones(l,1),[],[],y',0,zeros(l,1),C*
ones(l,1));
```

```
% compute b
ind = find((la > 1e-3) & (la < C-1e-3));
i = ind(1);
b = 1/y(i) ;
for j = 1 : l
   b = b - la(j)*y(j)*K(i,j);
end
%% plot the surface f(x)=0
for xx = -2 : 0.01 : 2
   for yy = -2 : 0.01 : 2
        s = 0;
        for i = 1 : l

s=s+la(i)*y(i)*exp(-gamma*norm(T(i,:)-[xx
yy])^2);
        end
        s = s + b;
        if (abs(s)< 10^(-2))
             plot(xx,yy,'g.');
        hold on

        end
    end
end
plot(A(:,1),A(:,2),'bo',B(:,1),B(:,2),'ro','Li
newidth',5)
```

6. Regression problems

We want to find coefficients z := ($z_1, z_2 \dots z_n$) of polynomial p such that |r| is minimum, which amount to solve the following unconstrained problem

$$\begin{cases} \min \|Az - y\| \\ z \in \mathbb{R}^n \end{cases}$$

$$A = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_\ell & x_\ell^2 & \dots & x_\ell^{n-1} \end{pmatrix} \in \mathbb{R}^{\ell \times n} \qquad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_\ell \end{pmatrix}$$

For any norm, the objective function f(z) = ||Az - y|| is **convex.**

Polynomial regression with $|.|_2$ (least squares approximation) -

$$\begin{cases} \min \frac{1}{2}\|Az - y\|_2^2 = \frac{1}{2}(Az - y)^\mathsf{T}(Az - y) = \frac{1}{2}z^\mathsf{T}A^\mathsf{T}Az - z^\mathsf{T}A^\mathsf{T}y + \frac{1}{2}y^\mathsf{T}y \\ z \in \mathbb{R}^n \end{cases}$$

It is an **unconstrained quadratic programming problem.**
It can be proved that rank(A) = n, thus $A^T A$ is positive definite.
→ the unique optimal solution is the stationary point of the objective function, the solution of the system of linear equations: $A^T Az = A^T y$

Polynomial regression with $|.|_1$ -

$$\begin{cases} \min \|Az - y\|_1 = \min \sum_{i=1}^{\ell} |A_i z - y_i| \\ z \in \mathbb{R}^n \end{cases}$$

It is a **linear programming problem.** Which is equal to these formulations:

$$\begin{cases} \min\limits_{z,u} \sum_{i=1}^{\ell} u_i \\ u_i \geq A_i z - y_i & \forall\, i = 1, \dots, \ell \\ u_i \geq y_i - A_i z & \forall\, i = 1, \dots, \ell \end{cases}$$

and in the matrix form:

Set

$$D = \begin{pmatrix} A & -I_\ell \\ -A & -I_\ell \end{pmatrix} \quad d = \begin{pmatrix} y \\ -y \end{pmatrix}$$

where I_ℓ is the identity matrix of order ℓ, then we obtain

$$\begin{cases} \min\limits_{z,u} (0_n^T, e_\ell^T) \begin{pmatrix} z \\ u \end{pmatrix} \\ D \begin{pmatrix} z \\ u \end{pmatrix} \leq d \end{cases}$$

Polynomial regression with $|.|_{inf}$ -

$$\begin{cases} \min \|Az - y\|_\infty = \min \max\limits_{i=1,\dots,\ell} |A_i z - y_i| \\ z \in \mathbb{R}^n \end{cases}$$

It is a **linear programming problem.**
In the matrix form it will be expressed as:

Set

$$D = \begin{pmatrix} A & -e_\ell \\ -A & -e_\ell \end{pmatrix} \quad d = \begin{pmatrix} y \\ -y \end{pmatrix}$$

where $e_\ell = (1, \dots, 1) \in \mathbb{R}^\ell$, in matrix form (3) becomes:

$$\begin{cases} \min\limits_{z,u} (0, 0, \dots, 0, 1) \begin{pmatrix} z \\ u \end{pmatrix} \\ D \begin{pmatrix} z \\ u \end{pmatrix} \leq d \end{cases}$$

Exercise - Find the best approximating polynomials of degree 3 with respect to the norms $|.|_2$, $|.|_1$, $|.|_{inf}$

```
x = data(:,1) ;
y = data(:,2) ;
l = length(x) ;
n = 4 ; % number of coefficients of polynomial
% Vandermonde matrix
A = [ ones(l,1) x x.^2 x.^3 ];
%% 2-norm problem
z2 = inv(A'*A)*(A'*y)
p2 = A*z2; % regression values at the data
%% 1-norm problem
% define the problem
c = [ zeros(n,1); ones(l,1) ];
D = [ A -eye(l); -A -eye(l) ];
d = [ y; -y ];
% solve the problem
sol1 = linprog(c,D,d) ;
z1 = sol1(1:n)
p1 = A*z1;
%% inf-norm problem
% define the problem
c = [ zeros(n,1); 1 ];
D = [ A -ones(l,1); -A -ones(l,1) ];
% solve the problem
solinf = linprog(c,D,d) ;
zinf = solinf(1:n)
pinf = A*zinf;
%% plot the solutions
plot(x,y,'b.',x,p2,'r-',x,p1,'k-',x,pinf,'g-')
legend('Data','2-norm','1-norm','inf-norm',...
   'Location','NorthWest');
```

Linear ε-SV regression

In general, in ε-SV regression we aim at finding a function f that $|f(x_i)-y_i|\leq\varepsilon$, i =1,..,l.

In a linear regression we consider an **affine** function $f(x)=w^Tx+b$ and set a tolerance parameter ε > 0. If we want f to be **flat** we must seek for a small w, which leads us to solve the convex quadratic optimization problem:

$$\begin{cases} \min\limits_{w,b} \ \dfrac{1}{2}\|w\|^2 \\ y_i \leq w^T x_i + b + \varepsilon \qquad \forall\ i = 1,\dots,\ell \\ y_i \geq w^T x_i + b - \varepsilon \qquad \forall\ i = 1,\dots,\ell \end{cases}$$

which in the matrix form is

$$\begin{cases} \min\limits_{w,b} \ \frac{1}{2}(w^T, b) Q \begin{pmatrix} w \\ b \end{pmatrix} \\ D \begin{pmatrix} w \\ b \end{pmatrix} \leq d \end{cases}$$

where

$$Q = \begin{pmatrix} I_\ell & 0 \\ 0_\ell^T & 0 \end{pmatrix} \quad D = \begin{pmatrix} -x & -e_\ell \\ x & e_\ell \end{pmatrix} \quad d = \begin{pmatrix} \varepsilon e_\ell - y \\ \varepsilon e_\ell + y \end{pmatrix}$$

Exercise - Apply the linear Ɛ-Sv regression model with Ɛ = 0.5 to the following training data.

```
x = data(:,1) ;
y = data(:,2) ;
l = length(x) ; % number of points
%%  linear  regression  -  primal  problem
-Exercise 6.2
% parameter
epsilon = 0.5 ;
% define the problem
Q = [ 1 0
     0 0 ];
 c = [0;0];
D = [-x -ones(l,1)
     x  ones(l,1)];
 d = epsilon*ones(2*l,1) + [-y;y];
% solve the problem
sol = quadprog(Q,c,D,d);
% compute w
w = sol(1);
% compute b
b = sol(2);
% find regression and epsilon-tube
z = w.*x + b ;
zp = w.*x + b + epsilon ;
zm = w.*x + b - epsilon ;
%% plot the solution
plot(x,y,'b.',x,z,'k-',x,zp,'r-',x,zm,'r-');
legend('Data','regression','\epsilon-tube',...
   'Location','NorthWest')
```

Linear ε-SV regression with slack variables -

$$\begin{cases} \min\limits_{w,b,\xi^+,\xi^-} \ \dfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-) \\ y_i \leq w^T x_i + b + \varepsilon + \xi_i^+ \qquad \forall\ i = 1,\dots,\ell \\ y_i \geq w^T x_i + b - \varepsilon - \xi_i^- \qquad \forall\ i = 1,\dots,\ell \\ \xi^+ \geq 0 \\ \xi^- \geq 0 \end{cases}$$

Where the parameter C gives the trade-off between the flatness of f and tolerance to deviations larger than ε.

Exercise - Apply the linear regression with slack variables (set xi = 0.2 and C=10) to the training data given.

```
x = data(:,1) ;
y = data(:,2) ;
l = length(x) ; % number of points
%% linear regression - primal problem with slack variables
% parameters
epsilon = 0.2 ;
C = 10 ;
% define the problem
Q = [ 1                zeros(1,2*l+1)
     zeros(2*l+1,1) zeros(2*l+1) ];
 c = [ 0 ; 0 ; C*ones(2*l,1)];
D = [-x -ones(l,1) -eye(l)   zeros(l)
     x  ones(l,1)  zeros(l) -eye(l)];
 d = epsilon*ones(2*l,1) + [-y;y];
% solve the problem
sol = quadprog(Q,c,D,d,[],[],[-inf;-inf;zeros(2*l,1)],[]);
% compute w
w = sol(1);
% compute b
b = sol(2);
% compute slack variables xi+ and xi-
xip = sol(3:2+l);
xim = sol(3+l:2+2*l);
% find regression and epsilon-tube
z = w.*x + b ;
zp = w.*x + b + epsilon ;
zm = w.*x + b - epsilon ;
%% plot the solution
plot(x,y,'b.',x,z,'k-',x,zp,'r-',x,zm,'r-');
legend('Data','regression',...
'\epsilon-tube','Location','NorthWest')
```

Dual formulation -

$$\begin{cases} \max\limits_{\lambda^+,\lambda^-} \; -\dfrac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)(x_i)^\mathsf{T} x_j \\ \qquad\qquad -\varepsilon\sum\limits_{i=1}^{\ell}(\lambda_i^+ + \lambda_i^-) + \sum\limits_{i=1}^{\ell} y_i(\lambda_i^+ - \lambda_i^-) \\ \sum\limits_{i=1}^{\ell}(\lambda_i^+ - \lambda_i^-) = 0 \\ \lambda_i^+ \in [0, C], \;\; i = 1,..,\ell \\ \lambda_i^- \in [0, C], \;\; i = 1,..,\ell \end{cases}$$

Is a convex quadratic programming problem, dual constraints are simpler than primal, if $\lambda_i^+ > 0$ or $\lambda_i^- > 0$, then x_i is said **support vector**.
If $(\lambda_i^+, \lambda_i^-)$ is a dual optimum then

$$w = \sum_{i=1}^{\ell}(\lambda_i^+ - \lambda_i^-)x_i,$$

b is obtained using the complementarity conditions, hence,if there is some i s.t. $0< \lambda_i^+ < C$, then $b = y_i - w^T x_i - \varepsilon$; if there is some i s.t. $0<\lambda_i^- < C$,then $b= y_i - w^T x_i + \varepsilon$.

Exercise -

```
x = data(:,1) ;
y = data(:,2) ;
l = length(x) ; % number of points
%% linear regression - dual problem
% parameters
epsilon = 0.2 ;
C = 10;
% define the problem
X = zeros(l,l);
for i = 1 : l
   for j = 1 : l
       X(i,j) = x(i)*x(j);
   end
end
Q = [ X -X ; -X X ];
c = epsilon*ones(2*l,1) + [-y;y];
% solve the problem
sol = quadprog(Q,c,[],[],[ones(1,l)
-ones(1,l)],0,zeros(2*l,1),C*ones(2*l,1
));
lap = sol(1:l);
lam = sol(l+1:2*l);
% compute w
w = (lap-lam)'*x ;
% compute b
ind = find(lap > 10^(-3) & lap <
C-10^(-3));
if isempty(ind)==0 %~isempty(ind)
   i = ind(1);
   b = y(i) - w*x(i) - epsilon ;
else
    ind = find(lam > 10^(-3) & lam <
C-10^(-3));
   i = ind(1);
   b = y(i) - w*x(i) + epsilon ;
end
% find regression and epsilon-tube
z = w.*x + b ;
zp = w.*x + b + epsilon ;
zm = w.*x + b - epsilon ;
%% plot the solution
% find support vectors
sv = [find(lap > 1e-3);find(lam >
1e-3)];
sv = sort(sv);
plot(x,y,'b.',x(sv),y(sv),...
   'ro',x,z,'k-',x,zp,'r-',x,zm,'r-');
legend('Data','Support vectors',...
   'regression','\epsilon-tube',...
   'Location','NorthWest')
```

Nonlinear ε-SV regression
Primal problem

$$\begin{cases} \min\ \dfrac{1}{2}\|w\|^2 + C\displaystyle\sum_{i=1}^{\ell}(\xi_i^+ + \xi_i^-) \\ y_i \le w^T\phi(x_i) + b + \varepsilon + \xi_i^+ & \forall\ i = 1,\dots,\ell \\ y_i \ge w^T\phi(x_i) + b - \varepsilon - \xi_i^- & \forall\ i = 1,\dots,\ell \end{cases}$$

Dual problem

$$\begin{cases} \displaystyle\max_{(\lambda^+,\lambda^-)}\ -\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}(\lambda_i^+ - \lambda_i^-)(\lambda_j^+ - \lambda_j^-)k(x_i,x_j) \\ \qquad\qquad -\varepsilon\displaystyle\sum_{i=1}^{\ell}(\lambda_i^+ + \lambda_i^-) + \sum_{i=1}^{\ell} y_i(\lambda_i^+ - \lambda_i^-) \\ \displaystyle\sum_{i=1}^{\ell}(\lambda_i^+ - \lambda_i^-) = 0 \\ \lambda_i^+, \lambda_i^- \in [0, C] \end{cases}$$

Method

- choose a kernel k
- solve the dual → find (λ^+, λ^-)
- find b:

$$b = y_i - \varepsilon - \sum_{j=1}^{\ell}(\lambda_j^+ - \lambda_j^-)k(x_i,x_j), \qquad \text{for some } i \text{ s.t. } 0 < \lambda_i^+ < C$$

  or

$$b = y_i + \varepsilon - \sum_{j=1}^{\ell}(\lambda_j^+ - \lambda_j^-)k(x_i,x_j), \qquad \text{for some } i \text{ s.t. } 0 < \lambda_i^- < C$$

- Regression function is: $\displaystyle f(x) = \sum_{i=1}^{\ell}(\lambda_i^+ - \lambda_i^-)k(x_i,x) + b$

Exercise - Consider the training data given. Apply the nonlinear ε-SV regression using a polynomial kernel with degree p = 3 and parameters ε = 10, C = 10. Moreover, find the support vectors.

```
x = data(:,1) ;
y = data(:,2) ;
l = length(x) ; % number of points
%% nonlinear regression - dual problem
epsilon = 10 ;
C = 10;
% define the problem
X = zeros(l,l);
for i = 1 : l
   for j = 1 : l
       X(i,j) = kernel(x(i),x(j)) ;
   end
end
Q = [ X -X ; -X X ];
c = epsilon*ones(2*l,1) + [-y;y];
% solve the problem
sol = quadprog(Q,c,[],[],...
   [ones(1,l) -ones(1,l)],0,...
   zeros(2*l,1),C*ones(2*l,1));
lap = sol(1:l);
lam = sol(l+1:2*l);
% compute b
ind = find(lap > 1e-3 & lap < C-1e-3);
if isempty(ind)==0
   i = ind(1);
   b = y(i) - epsilon;
   for j = 1 : l
        b = b - (lap(j)-lam(j))*kernel(x(i),x(j));
   end
else
   ind = find(lam > 1e-3 & lam < C-1e-3);
   i = ind(1);
   b = y(i) + epsilon ;
   for j = 1 : l
        b = b - (lap(j)-lam(j))*kernel(x(i),x(j));
   end
end
% find regression and epsilon-tube
z = zeros(l,1);
for i = 1 : l
   z(i) = b ;
   for j = 1 : l
        z(i) = z(i) + (lap(j)-lam(j))*kernel(x(i),x(j));
   end
end
zp = z + epsilon ;
zm = z - epsilon ;
%% plot the solution
% find support vectors
sv = [find(lap > 1e-3);find(lam > 1e-3)];
sv = sort(sv);
plot(x,y,'b.',x(sv),y(sv),...
   'ro',x,z,'k-',x,zp,'r-',x,zm,'r-');
legend('Data','Support vectors',...
   'regression','\epsilon-tube',...
   'Location','NorthWest')
%% kernel function
function v = kernel(x,y)
p = 4 ;
v = (x'*y + 1)^p;
end
```

7. Clustering problems

A clustering consists in finding a partition of S in k subsets $S_1 \ldots S_k$ (clusters) that are homogeneous and well separated.
Clustering problem is of interest in **unsupervised machine learning**.

Patterns are vectors $p_1 \ldots p_l$. Consider a distance d. For each cluster S_j we introduce a **centroid** x_j (unknown).
Define clusters so that each pattern is associated to the closest centroid.
We aim to find k centroids in order to **minimize the sum of the distances** between each pattern and the closest centroid.

$$\begin{cases} \min \sum_{i=1}^{\ell} \min_{j=1,\ldots,k} d(p_i, x_j) \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\ldots,k \end{cases}$$

Optimization model with $|.|_2$

$$\begin{cases} \min \sum_{i=1}^{\ell} \min_{j=1,\ldots,k} \|p_i - x_j\|_2^2 \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\ldots,k \end{cases}$$

If k = 1 we have one cluster, then it is a **convex quadratic programming problem** without constraints.

$$\begin{cases} \min \sum_{i=1}^{\ell} \|p_i - x\|_2^2 = \min \sum_{i=1}^{\ell} (x - p_i)^T (x - p_i) \\ x \in \mathbb{R}^n \end{cases} \qquad (1)$$

The global optimum is the stationary point:

$$x = \frac{\sum_{i=1}^{\ell} p_i}{\ell} \quad \text{(mean or baricenter)}$$

If k > 1 then the problem is **nonconvex and nondifferentiable**

$$\min_{j=1,\ldots,k} \|p_i - x_j\|_2^2 = \begin{cases} \min \sum_{j=1}^{k} \alpha_{ij} \|p_i - x_j\|_2^2 \\ \sum_{j=1}^{k} \alpha_{ij} = 1 \\ \alpha_{ij} \geq 0 \qquad \forall\; j = 1,\ldots,k \end{cases} \quad \textbf{(3)}$$

An optimal solution of (3) is given by

$$\alpha_{ij}^* = \begin{cases} 1 & \text{if } \|p_i - x_j\|_2 = \min_{h=1,\ldots,k} \|p_i - x_h\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

Remark

Observe that $\alpha_{ij}^* = 1$ if pattern i is assigned to cluster j.

Theorem -
The initial model is equivalent to the following **nonconvex differentiable** problem:

$$\begin{cases} \min_{x,\alpha} f(x,\alpha) := \sum_{i=1}^{\ell} \sum_{j=1}^{k} \alpha_{ij} \|p_i - x_j\|_2^2 \\ \sum_{j=1}^{k} \alpha_{ij} = 1 \quad \forall\, i = 1,\ldots,\ell \\ \alpha_{ij} \geq 0 \qquad \forall\, i = 1,\ldots,\ell,\; j = 1,\ldots,k \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\ldots,k. \end{cases}$$

K-means algorithm -
Based on the properties of the problem above:

- if x_j are fixed, then the problem is decomposable into l simple LP problems of the form of the problem (3) above

$$\alpha^*_{ij} = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x_j\|_2 = \min_{h=1,\dots,k} \|p_i - x_h\|_2 \\ & \quad (x_j \text{ is the first closest centroid to } p_i), \\ 0 & \text{otherwise.} \end{cases}$$

- if A_{ij} are fixed, then is decomposable into k **convex QP problems** similar to (1) above

$$\begin{cases} \min \sum_{i=1}^{\ell} \alpha_{ij} \|p_i - x_j\|_2^2 = \min \sum_{i=1}^{\ell} \alpha_{ij} (x_j - p_i)^\mathsf{T} (x_j - p_i) \\ x_j \in \mathbb{R}^n \end{cases}$$

For any j = 1…k the optimal solution is $x^*_j = \dfrac{\sum_{i=1}^{\ell} \alpha_{ij} p_i}{\sum_{i=1}^{\ell} \alpha_{ij}}$ (mean of patterns).

k-means algorithm -

0. (Inizialization) Set $t = 0$, choose centroids $x^0_1, \dots, x^0_k \in \mathbb{R}^n$ and assign patterns to clusters: for any $i = 1, \dots, \ell$

$$\alpha^0_{ij} = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x^0_j\|_2 = \min_{h=1,\dots,k} \|p_i - x^0_h\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

1. (Update centroids) For each $j = 1, \dots, k$ compute the mean

$$x^{t+1}_j = \left(\sum_{i=1}^{\ell} \alpha^t_{ij} p_i \right) / \left(\sum_{i=1}^{\ell} \alpha^t_{ij} \right).$$

2. (Update clusters) For any $i = 1, \dots, \ell$ compute

$$\alpha^{t+1}_{ij} = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x^{t+1}_j\|_2 = \min_{h=1,\dots,k} \|p_i - x^{t+1}_h\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

3. (Stopping criterion) If $f(x^{t+1}, \alpha^{t+1}) = f(x^t, \alpha^t)$ then STOP
else $t = t + 1$, go to Step 1.

Theorem -

The k-means algorithm stops after a finite number of iterations at a solution (x^*, α^*) of the KKT system of problem (5) such that

$$\begin{aligned} & f(x^*, \alpha^*) \le f(x^*, \alpha), && \forall\, \alpha \ge 0 \text{ s.t. } \sum_{j=1}^{k} \alpha_{ij} = 1 \quad \forall\, i = 1, \dots, \ell, \\ & f(x^*, \alpha^*) \le f(x, \alpha^*), && \forall\, x \in \mathbb{R}^{kn}. \end{aligned}$$

Remark. The k-means algorithm does not guarantee to find a global optimum.

Exercise - Consider the k-means algorithm with k=3 for the following set of patterns.

```
l = size(data,1); % number of patterns
k=3;
InitialCentroids=[5,7;6,3;4,4];
[x,cluster,v] = kmeans1(data,k,InitialCentroids)
% plot centroids
plot(x(1,1),x(1,2),'b*',x(2,1),x(2,2),'r*',x(3,1),x(3,2),'g*');
hold on
% plot cluster
c1 = data(cluster==1,:);
c2 = data(cluster==2,:);
c3 = data(cluster==3,:);
plot(c1(:,1),c1(:,2),'bo',c2(:,1),c2(:,2),'ro',c3(:,1),c3(:,2),'go');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,cluster,v] = kmeans1(data,k,InitialCentroids)
l = size(data,1); % number of patterns
% initialize centroids
x = InitialCentroids;
% initialize clusters
cluster = zeros(l,1);
for i = 1 : l
   d = inf;
   for j = 1 : k
       if norm(data(i,:)-x(j,:)) < d
           d = norm(data(i,:)-x(j,:));
           cluster(i) = j;
       end
   end
end
% compute the objective function value
vold = 0;
for i = 1 : l
    vold = vold + norm(data(i,:)-x(cluster(i),:))^2 ;
end
while true

   % update centroids
   for j = 1 : k
       ind = find(cluster == j);
       if isempty(ind)==0
           x(j,:) = mean(data(ind,:),1);
       end
   end

   % update clusters
   for i = 1 : l
       d = inf;
       for j = 1 : k
           if norm(data(i,:)-x(j,:)) < d
               d = norm(data(i,:)-x(j,:));
               cluster(i) = j;
           end
       end
   end
   % update objective function
   v = 0;
   for i = 1 : l
       v = v + norm(data(i,:)-x(cluster(i),:))^2 ;
   end

   % stopping criterion
   if vold - v < 1e-5
       break
   else
       vold = v;
   end
end
end
```

Optimization model with $|.|_1$

$$\begin{cases} \min \sum_{i=1}^{\ell} \min_{j=1,\dots,k} \|p_i - x_j\|_1 \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\dots,k \end{cases}$$

If k=1 it is a **convex** problem decomposable into n convex of one variable:

$$\begin{cases} \min \sum_{i=1}^{\ell} \|p_i - x\|_1 = \min \sum_{i=1}^{\ell}\sum_{h=1}^{n} |x_h - (p_i)_h| = \min \sum_{h=1}^{n} \underbrace{\sum_{i=1}^{\ell} |x_h - (p_i)_h|}_{f_h(x_h)} \\ x \in \mathbb{R}^n \end{cases} \tag{7}$$

Given l real numbers $a_1 < a_2 < \dots < a_l$ what is the optimal solution of

$$\begin{cases} \min \sum_{i=1}^{\ell} |x - a_i| = f(x) \\ x \in \mathbb{R} \end{cases}$$

?

The global optimum is median(a1, …, al) =

$$\begin{cases} a_{(\ell+1)/2} & \text{if } \ell \text{ is odd,} \\ \dfrac{a_{\ell/2} + a_{1+\ell/2}}{2} & \text{if } \ell \text{ is even.} \end{cases}$$

If k > 1 then the problem is **nonconvex and nonsmooth**, and it is equivalent to the following problem:

$$\begin{cases} \min\limits_{x,\alpha} \sum\limits_{i=1}^{\ell} \sum\limits_{j=1}^{k} \alpha_{ij} \|p_i - x_j\|_1 \\ \sum\limits_{j=1}^{k} \alpha_{ij} = 1 \quad \forall\, i = 1,\dots,\ell \\ \alpha_{ij} \ge 0 \qquad \forall\, i = 1,\dots,\ell,\ j = 1,\dots,k \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\dots,k. \end{cases}$$

which is equivalent to the **nonxconvex differentiable (bilinear)** problem:

$$\begin{cases} \min\limits_{x,\alpha,u} \sum\limits_{i=1}^{\ell} \sum\limits_{j=1}^{k} \sum\limits_{h=1}^{n} \alpha_{ij} u_{ijh} \\ u_{ijh} \ge (p_i)_h - (x_j)_h \qquad \forall\, i = 1,\dots,\ell,\ j = 1,\dots,k,\ h = 1,\dots,n \\ u_{ijh} \ge (x_j)_h - (p_i)_h \qquad \forall\, i = 1,\dots,\ell,\ j = 1,\dots,k,\ h = 1,\dots,n \\ \sum\limits_{j=1}^{k} \alpha_{ij} = 1 \qquad \forall\, i = 1,\dots,\ell \\ \alpha_{ij} \ge 0 \qquad \forall\, i = 1,\dots,\ell,\ j = 1,\dots,k \\ x_j \in \mathbb{R}^n \qquad \forall\, j = 1,\dots,k. \end{cases}$$

- if x_j are fixed, then is decomposable into l simple LP problems: for any i = 1…l, the optimal solution is

$$\alpha^*_{ij} = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x_j\|_1 = \min\limits_{h=1,\dots,k} \|p_i - x_h\|_1 \\ & \quad (x_j \text{ is the first closest centroid to } p_i), \\ 0 & \text{otherwise.} \end{cases}$$

- If $a_{ij} \in \{0,1\}$ are fixed, then is decomposable into k simple convex problems similar to (7)

$$\begin{cases} \min \sum\limits_{i=1}^{\ell} \alpha_{ij} \|p_i - x_j\|_1 = \min \sum\limits_{i=1}^{\ell} \sum\limits_{h=1}^{n} \alpha_{ij} |(x_j)_h - (p_i)_h| \\ x_j \in \mathbb{R}^n \end{cases}$$

  For any j = 1…k the optimal solution is x_j^* = median(p_i : a_{ij} = 1)

k-median algorithm

0. (Inizialization) Set $t = 0$, choose centroids $x_1^0, \dots, x_k^0 \in \mathbb{R}^n$ and assign patterns to clusters: for any $i = 1,\dots,\ell$

$$\alpha_{ij}^0 = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x_j^0\|_1 = \min\limits_{h=1,\dots,k} \|p_i - x_h^0\|_1 \\ 0 & \text{otherwise.} \end{cases}$$

1. (Update centroids) For each $j = 1,\dots,k$ compute

$$x_j^{t+1} = \text{median}(p_i :\ \alpha_{ij}^t = 1).$$

2. (Update clusters) For any $i = 1,\dots,\ell$ compute

$$\alpha_{ij}^{t+1} = \begin{cases} 1 & \text{if } j \text{ is the first index s.t. } \|p_i - x_j^{t+1}\|_1 = \min\limits_{h=1,\dots,k} \|p_i - x_h^{t+1}\|_1 \\ 0 & \text{otherwise.} \end{cases}$$

3. (Stopping criterion) If $f(x^{t+1}, \alpha^{t+1}) = f(x^t, \alpha^t)$ then STOP
   else $t = t + 1$, go to Step 1.

Theorem -

The k-median algorithm stops after a finite number of iterations at a stationary point (x^*, α^*) of problem (8) such that

$$f(x^*, \alpha^*) \le f(x^*, \alpha), \qquad \forall\, \alpha \ge 0 \ \text{s.t.} \ \sum_{j=1}^{k} \alpha_{ij} = 1 \quad \forall\, i = 1,\dots,\ell,$$
$$f(x^*, \alpha^*) \le f(x, \alpha^*), \qquad \forall\, x \in \mathbb{R}^{kn}.$$

Remark - The k-median algorithm **does not guarantee** to find a **global optimum.**

Exercise -

```
l = size(data,1); % number of patterns
k=3;
InitialCentroids=[5,7;6,3;4,3];
%InitialCentroids= 10*rand(k,2)
[x,cluster,v]                                    =
kmedian2(data,k,InitialCentroids)
% plot centroids
plot(x(1,1),x(1,2),'b*',x(2,1),x(2,2),'r*',x(3
,1),x(3,2),'g*');
hold on
% plot cluster
c1 = data(cluster==1,:);
c2 = data(cluster==2,:);
c3 = data(cluster==3,:);
plot(c1(:,1),c1(:,2),'bo',c2(:,1),c2(:,2),'ro'
,c3(:,1),c3(:,2),'go');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
function             [x,cluster,v]            =
kmedian2(data,k,InitialCentroids)
l = size(data,1); % number of patterns
% initialize centroids
x = InitialCentroids;
% initialize clusters
cluster = zeros(l,1);
for i = 1 : l
   d = inf;
   for j = 1 : k
       if norm(data(i,:)-x(j,:),1) < d
           d = norm(data(i,:)-x(j,:),1);
           cluster(i) = j;
       end
   end
end
% compute the objective function value
vold = 0;
for i = 1 : l
                  vold      =      vold      +
norm(data(i,:)-x(cluster(i),:),1) ;
end
while true

   % update centroids
   for j = 1 : k
       ind = find(cluster == j);
       if isempty(ind)==0
           x(j,:) = median(data(ind,:),1);
       end
   end

   % update clusters
   for i = 1 : l
       d = inf;
       for j = 1 : k
           if norm(data(i,:)-x(j,:),1) < d
               d = norm(data(i,:)-x(j,:),1);
               cluster(i) = j;
           end
       end
   end
   % update objective function
   v = 0;
   for i = 1 : l
                              v     =     v     +
norm(data(i,:)-x(cluster(i),:),1) ;
   end

   % stopping criterion
   if vold - v < 1e-5
       break
   else
       vold = v;
   end
end
end
```

8. Constrained optimization problems

Penalty method -
Consider a constrained optimization problem

$$\begin{cases} \min\ f(x) \\ g_i(x) \leq 0 \qquad \forall\ i = 1, \dots, m \end{cases}$$

Define the quadratic **penalty function**

$$p(x) = \sum_{i=1}^{m} (\max\{0, g_i(x)\})^2$$

and consider the **unconstrained** penalized problem

$$\begin{cases} \min\ f(x) + \frac{1}{\varepsilon} p(x) := p_\varepsilon(x) \\ x \in \mathbb{R}^n \end{cases}$$

$$p_\varepsilon(x) \begin{cases} = f(x) & \text{if } x \in X \\ > f(x) & \text{if } x \notin X \end{cases}$$

Proposition 8.1

1. If f, g_i are continuously differentiable, then p_ε is continuously differentiable and $\nabla p_\varepsilon(x) = \nabla f(x) + \frac{2}{\varepsilon} \sum_{i=1}^{m} \max\{0, g_i(x)\} \nabla g_i(x)$
2. If f and g_i are convex, then p_ε is convex
3. Any (P_ε) is a relaxation of (P), i.e., $v(P_\varepsilon) \leq v(P)$ for any $\varepsilon > 0$
4. If x_ε^* solves (P_ε) and $x_\varepsilon^* \in X$, then x_ε^* is optimal also for (P)
5. If $0 < \varepsilon_2 < \varepsilon_1$, then $v(P_{\varepsilon_1}) \leq v(P_{\varepsilon_2})$

Penalty method

0. Set $\varepsilon_0 > 0$, $\tau \in (0, 1)$, $k = 0$
1. Find an optimal solution x^k of the penalized problem (P_{ε_k})
2. **If** $x^k \in X$ **then** STOP
   else $\varepsilon_{k+1} = \tau \varepsilon_k$, $k = k + 1$ and go to step 1.

Theorem 8.2 -

- If f is coercive, then the sequence $\{x^k\}$ is bounded and any of its cluster points is an optimal solution of (P).
- If $\{x^k\}$ converges to x^*, then x^* is an optimal solution of (P).
- If $\{x^k\}$ converges to x^* and the gradients of active constraints at x^* are linear independent, then x^* is an optimal solution of (P) and the sequence of vectors $\{\lambda^k\}$ defined as

$$\lambda_i^k := \frac{2}{\varepsilon_k} \max\{0, g_i(x^k)\}, \qquad i = 1, \dots, m$$

converges to a vector λ^* of KKT multipliers associated to x^*.

Exercise - Implement the penalty method for solving the quadratic programming constrained problem, with Q positive definite matrix. (use max(Ax - b) < 10^{-6} as stopping criterion)

$$\begin{cases} \min \ \frac{1}{2}x^T Qx + c^T x \\ Ax \le b \end{cases} \qquad \begin{cases} \min \ \frac{1}{2}(x_1-3)^2 + (x_2-2)^2 \\ -2x_1 + x_2 \le 0 \\ x_1 + x_2 \le 4 \\ -x_2 \le 0 \end{cases}$$

```
global Q c A b eps;
%% data
Q = [ 1 0 ; 0 2 ] ;
c = [ -3 ; -4 ] ;
A = [-2 1 ; 1 1 ; 0 -1 ];
b = [ 0 ; 4 ; 0 ];
tau = 0.1 ;
eps0 = 5 ;
tolerance = 1e-6 ;
%% method
eps = eps0;
x = [0;0];
iter = 0;
SOL=[];
while true
    [x,pval] = fminunc(@p_eps,x);
    infeas = max(A*x-b);

    SOL=[SOL;iter,eps,x',infeas,pval];
    if infeas < tolerance
        break
    else
        eps = tau*eps;
        iter = iter + 1 ;
    end
end
fprintf('\t iter \t eps \t x(1) \t x(2) \t
max(Ax-b) \t pval \n');
SOL
%% penalized function
function v= p_eps(x)
   global Q c A b eps;
   v = 0.5*x'*Q*x + c'*x ;
    for i = 1 : size(A,1)
        v = v +
(1/eps)*(max(0,A(i,:)*x-b(i)))^2 ;
   end
end
```

Exact penalty method

Consider a convex constrained problem and define the linear penalty function

$$\tilde{p}(x) = \sum_{i=1}^{m} \max\{0, g_i(x)\}.$$

and consider the penalized problem **unconstrained, convex and nonsmooth**

$$\begin{cases} \min \ \tilde{p}_\varepsilon(x) := f(x) + \frac{1}{\varepsilon}\tilde{p}(x) \\ x \in \mathbb{R}^n \end{cases}$$

For such penalized problem we do not need a sequence εk → 0 to approximate an optimal solution of (P) (which avoid numerical issues), in fact there exists a suitable ε such that the minimum of (Pε) coincides with the minimum of (P).

Exact penalty method

0. Set $\varepsilon_0 > 0$, $\tau \in (0,1)$, $k = 0$
1. Find an optimal solution x^k of the penalized problem $(\widetilde{P}_{\varepsilon_k})$
2. **If** $x^k \in X$ **then** STOP
   else $\varepsilon_{k+1} = \tau\varepsilon_k$, $k = k+1$ and go to step 1.

The method stops after a **finite number** of iterations at an **optimal solution of (P)**. Notice that penalty methods generate a sequence of **unfeasible points** that approximate an optimal solution of (P).

Exercise - Run the exact penalty method with τ = 0.5 and ε0 = 4 for solving the problem

```
global Q c A b eps;
%% data
Q = [ 1 0 ; 0 2 ] ;
c = [ -3 ; -4 ] ;
A = [-2 1 ; 1 1 ; 0 -1 ];
b = [ 0 ; 4 ; 0 ];
tau = 0.5 ;
eps0 = 4 ;
tolerance = 1e-6 ;
%% exact penalty method
eps = eps0;
x0 = [0;0];
iter = 0;
SOL=[];
while true
    [x,pval] = fminunc(@p_eps,x0);
    infeas = max(A*x-b);

    SOL=[SOL;iter,eps,x',infeas,pval];
    if infeas < tolerance
        break
    else
        eps = tau*eps;
        iter = iter + 1 ;
    end
end
fprintf('\t iter \t eps \t x(1) \t x(2)
\t max(Ax-b) \t pval \n');
SOL
%% penalized function
function v= p_eps(x)
   global Q c A b eps;
   v = 0.5*x'*Q*x + c'*x ;
    for i = 1 : size(A,1)
        v = v +
(1/eps)*(max(0,A(i,:)*x-b(i))) ;
   end
end
```

Barrier methods -

Unlike penalty methods, barrier methods generate a sequence of **feasible** points that approximate an optimal solution of (P).

Consider

$$\begin{cases} \min\ f(x) \\ g_i(x) \leq 0 \quad i = 1, .., m \end{cases} \qquad (P)$$

under the following assumptions:

- f, g_i convex and twice continuously differentiable (on an open set containing X)
- there exists an optimal solution (e.g. f is coercive or X is bounded)
- Slater constraint qualification holds: there exists $\bar{x}$ such that

$$g_i(\bar{x}) < 0,\ \forall\ i = 1, \ldots, m$$

Hence strong duality holds.

Logarithmic barrier -

$$\begin{cases} \min\ f(x) - \varepsilon \sum_{i=1}^{m} log(-g_i(x)) \\ x \in \text{int}(X) \end{cases} \qquad B(x) := -\sum_{i=1}^{m} log(-g_i(x))$$

B(x) is called **logarithmic barrier function.**

The function B(x) has the following properties:

- dom(B) = int(X)
- B is convex
- B is smooth with

$$\nabla B(x) = -\sum_{i=1}^{m} \frac{1}{g_i(x)} \nabla g_i(x)$$

$$\nabla^2 B(x) = \sum_{i=1}^{m} \frac{1}{g_i(x)^2} \nabla g_i(x) \nabla g_i(x)^{\mathsf{T}} + \sum_{i=1}^{m} \frac{1}{-g_i(x)} \nabla^2 g_i(x)$$

Logarithmic barrier method

0. Set tolerance $\delta > 0$, $\tau \in (0,1)$ and $\varepsilon_1 > 0$. Choose $x^0 \in \text{int}(X)$, set $k = 1$
1. Find the optimal solution x^k of

$$\begin{cases} \min \; f(x) - \varepsilon_k \sum_{i=1}^{m} log(-g_i(x)) \\ x \in \text{int}(X) \end{cases}$$

   using x^{k-1} as starting point
2. **If** $m\,\varepsilon_k < \delta$ **then** STOP
   else $\varepsilon_{k+1} = \tau\varepsilon_k$, $k = k+1$ and go to step 1

Choice of starting point

In order to find an initial point $x^0 \in \text{int}(X)$ we can consider the auxiliary problem

$$\begin{cases} \min\limits_{x,s} \; s \\ g_i(x) \leq s, \quad i = 1, .., m \end{cases}$$

- Take any $\tilde{x} \in \mathbb{R}^n$, find $\tilde{s} > \max\limits_{i=1,\dots,m} g_i(\tilde{x})$
  [$(\tilde{x}, \tilde{s})$ is in the interior of the feasible region of the auxiliary problem]
- Find an optimal solution (x^*, s^*) of the auxiliary problem using a barrier method starting from $(\tilde{x}, \tilde{s})$
- If $s^* < 0$ then $x^* \in \text{int}(X)$
  else $\text{int}(X) = \emptyset$

Exercise - Run the logarithmic barrier method with $\delta = 10^{-3}$, $\tau = 0.5$, $\varepsilon_1 = 1$ and $x^0 = (1, 1)$ for solving the problem

$$\begin{cases} \min \; \frac{1}{2}(x_1 - 3)^2 + (x_2 - 2)^2 \\ -2x_1 + x_2 \leq 0 \\ x_1 + x_2 \leq 4 \\ -x_2 \leq 0 \end{cases}$$

```
%% data
global Q c A b eps;
Q = [ 1 0 ; 0 2 ] ;
c = [ -3 ; -4 ] ;
A = [-2 1 ; 1 1 ; 0 -1 ];
b = [ 0 ; 4 ; 0 ];
delta = 1e-3 ;
tau = 0.5 ;
eps1 = 1 ;
x0 = [ 1 ; 1 ];
%% barrier method
x = x0;
eps = eps1 ;
m = size(A,1) ;
SOL=[]
while true
   [x,pval] = fminunc(@logbar,x);
   gap = m*eps;
   SOL=[SOL;eps,x',gap,pval];
   if gap < delta
       break
   else
       eps = eps*tau;
   end
end
fprintf('\t eps \t x(1) \t x(2) \t gap 
\t  pval \n\n');
SOL
%% logarithmic barrier function
function v = logbar(x)
   global Q c A b eps
   v = 0.5*x'*Q*x + c'*x ;

   for i = 1 : length(b)
       v = v - eps*log(b(i)-A(i,:)*x) ;
   end
end
```

9. Multiobjective optimization

Minimum points for a set of vectors

Given a subset $A \subseteq \mathbb{R}^s$, we say that

- $\bar{x} \in A$ is a Pareto ideal minimum (or ideal efficient point) of A if $y \geq \bar{x}$ for any $y \in A$.
- $\bar{x} \in A$ is a Pareto minimum (or efficient point) of A if there is no $y \in A$, $y \neq \bar{x}$, such that $\bar{x} \geq y$ (or, equivalently, there is no $y \in A$ such that $\bar{x} \geq y$ and $\bar{x}_j > y_j$, for some $j \in \{1, .., s\}$).
- $\bar{x} \in A$ is a Pareto weak minimum (or weakly efficient point) of A if there is no $y \in A$ such that $\bar{x} > y$, i.e., $\bar{x}_i > y_i$ for any $i = 1, \ldots, s$.

$IMin(A)$, $Min(A)$ and $WMin(A)$ denote the set of ideal minima, minima, weak minima of A, respectively.

Given a multiobjective optimization problem

$$\begin{cases} \min \ f(x) = (f_1(x), f_2(x), \ldots, f_s(x)) \\ x \in X \end{cases} \tag{P}$$

- $x^* \in X$ is a Pareto ideal minimum of (P) if $f(x^*)$ is a Pareto ideal minimum of $f(X)$, i.e., $f(x) \geq f(x^*)$ for any $x \in X$.
- $x^* \in X$ is a Pareto minimum of (P) if $f(x^*)$ is a Pareto minimum of $f(X)$, i.e., if there is no $x \in X$ such that

$$\begin{array}{ll} f_i(x^*) \geq f_i(x) & \text{for any } i = 1, \ldots, s, \\ f_j(x^*) > f_j(x) & \text{for some } j \in \{1, \ldots, s\}. \end{array}$$

- $x^* \in X$ is a Pareto weak minimum of (P) if $f(x^*)$ is a Pareto weak minimum of $f(X)$, i.e., if there is no $x \in X$ such that

$$f_i(x^*) > f_i(x) \qquad \text{for any } i = 1, \ldots, s.$$

Theorem 2 - If f_i is **continuous** for any i = 1…s, and X is **compact** then there exists a minimum of (P).

Theorem 3 - If f_i is continuous for any i=1…s , X is **closed** and there exist v € R and j € {1…s} such that the sublevel set {x € X: $f_j(x)$ <= v} is **nonempty and bounded**, then there exists a minimum of (P).

Corollary - If f_i is continuous for any i=1…s , X is **closed** and f_j is **coercive** for some j € {1…s}, then there exists a minimum of (P).

Theorem 4 -

$x^* \in X$ is a minimum of (P) if and only if the auxiliary optimization problem

$$\begin{cases} \max \ \sum_{i=1}^{s} \varepsilon_i \\ f_i(x) + \varepsilon_i \leq f_i(x^*) \qquad \forall \ i = 1, \ldots, s \\ x \in X \\ \varepsilon \geq 0 \end{cases}$$

has optimal value equal to 0.

To solve the auxiliary problem in MATLAB, the structure is this:

$$\begin{cases} -\min \ -\varepsilon_1 - \varepsilon_2 - \varepsilon_3 \\ A \begin{pmatrix} x \\ \varepsilon \end{pmatrix} \leq b \\ x \geq 0, \ \varepsilon \geq 0 \end{cases}$$

Theorem 5 -

$x^* \in X$ is a weak minimum of (P) if and only if the auxiliary optimization problem

$$\begin{cases} \max\ v \\ v \leq \varepsilon_i & \forall\ i = 1, \ldots, s \\ f_i(x) + \varepsilon_i \leq f_i(x^*) & \forall\ i = 1, \ldots, s \\ x \in X \\ \varepsilon \geq 0 \end{cases}$$

has optimal value equal to 0.

To solve the auxiliary problem in MATLAB, the structure is this:

$$\begin{cases} -\min\ -v \\ A \begin{pmatrix} x \\ \varepsilon \\ v \end{pmatrix} \leq b \\ x \geq 0,\ \varepsilon \geq 0 \end{cases}$$

First-order optimality conditions: unconstrained problems

Consider an unconstrained multiobjective problem where f_i is continuously differentiable for any i = 1…s

Necessary optimality condition -

If x^* is a weak minimum of (P_u), then there exists $\theta^* \in \mathbb{R}^s$ such that (x^*, θ^*) is a solution of the system

$$\begin{cases} \sum_{i=1}^{s} \theta_i \nabla f_i(x) = 0 \\ \theta \geq 0, \quad \sum_{i=1}^{s} \theta_i = 1, \\ x \in \mathbb{R}^n \end{cases} \qquad \text{(S)}$$

Sufficient optimality condition -

Assume that the problem (P_u) is convex, i.e., f_i is convex for any $i = 1, \ldots, s$, and (x^*, θ^*) is a solution of the system (S). Then:

- x^* is a weak minimum of (P_u).
- If, additionally, $\theta^* > 0$, then x^* is a minimum of (P_u).

First order optimality conditions: constrained problems

Consider an unconstrained multiobjective problem where f_i, g_j, h_k are continuously differentiable for any i, j, k.

Abadie constraint qualification (ACQ) -

We say that the Abadie constraint qualification (ACQ) holds at a point $x^* \in X$, if $T_X(x^*) = D(x^*)$. Where Tx is the Tangent cone at x^* and D is the first order feasible direction cone at x^*.

Sufficient conditions for ACQ -

a) (*Affine constraints*)
If g_j and h_k are affine for all $j = 1, \ldots, m$ and $k = 1, \ldots, p$, then ACQ holds at any $x \in X$.

b) (*Slater condition for convex problems*)
If g_j are convex for all $j = 1, \ldots, m$, h_k are affine for all $k = 1, \ldots, p$ and there exists $\bar{x} \in X$ s.t. $g(\bar{x}) < 0$ and $h(\bar{x}) = 0$, then ACQ holds at any $x \in X$.

c) (*Linear independence of the gradients of active constraints*)
If $x^* \in X$ and the vectors

$$\begin{cases} \nabla g_j(x^*) & \text{for } j \in \mathcal{A}(x^*), \\ \nabla h_k(x^*) & \text{for } k = 1, \ldots, p \end{cases}$$

are linearly independent, then ACQ holds at x^*.

Necessary optimality conditions (KKT) -

If x^* is a weak minimum of (P) and ACQ holds at x^*, then there exist $\theta^* \in \mathbb{R}^s$, $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^p$ such that $(x^*, \theta^*, \lambda^*, \mu^*)$ solves the KKT system

$$\begin{cases} \sum_{i=1}^{s} \theta_i \nabla f_i(x) + \sum_{j=1}^{m} \lambda_j \nabla g_j(x) + \sum_{k=1}^{p} \mu_k \nabla h_k(x) = 0 \\ \theta \geq 0, \quad \sum_{i=1}^{s} \theta_i = 1 \\ \lambda \geq 0 \\ \lambda_j g_j(x) = 0 \qquad \forall j = 1, \dots, m \\ g(x) \leq 0, \ h(x) = 0 \end{cases} \tag{4}$$

Necessary conditions
Theorem -

If x^* is a weak minimum of (P), then the system

$$\begin{cases} \nabla f_i(x^*)^T d < 0, i = 1, ..., s \\ d \in T_X(x^*). \end{cases}$$

has no solutions.

Corollary -

If x^* is a weak minimum of (P) and ACQ holds at x^*, then the system

$$\begin{cases} v^T \nabla f_i(x^*) < 0, i = 1, ..., s \\ v^T \nabla g_j(x^*) \leq 0, \ j \in \mathcal{A}(x^*), \\ v^T \nabla h_k(x^*) = 0, \ k = 1, \dots, p, \\ v \in \mathbb{R}^n \end{cases}$$

has no solutions.

Sufficient condition
Theorem -

Assume that f_i and g_j are convex, $i = 1, .., s$, $j = 1, .., m$, h_k are affine $k = 1, .., p$.

- If $(x^*, \theta^*, \lambda^*, \mu^*)$ solves the KKT system, then x^* is a weak minimum of (P).
- If $(x^*, \theta^*, \lambda^*, \mu^*)$ solves the KKT system with $\theta^* > 0$, then x^* is a minimum of (P).

Proposition -

If x^* is the unique global minimum of the function f_k on the set X for some $k \in \{1, ..., s\}$, then x^* is a minimum of (P).

Scalarization method

We associate with (P) vector optimization problem the following scalar optimization problem:

$$\begin{cases} \min \sum_{i=1}^{s} \alpha_i f_i(x) \\ x \in X \end{cases}$$

where Sa be the set of optimal solutions of (Pa)

Theorem -

- $\bigcup_{\alpha \geq 0} S_\alpha \subseteq \{\text{weak minima of (P)}\}$
- $\bigcup_{\alpha > 0} S_\alpha \subseteq \{\text{minima of (P)}\}$

Theorem (convex case) -

Assume that X is a convex set and that f_i are convex on X for $i = 1, .., s$.
Then $\{\text{weak minima of (P)}\} = \bigcup_{\alpha \geq 0} S_\alpha$

Theorem (linear case) -

Let (P) be linear, i.e., f_i are linear for $i = 1, .., s$ and X is a polyhedron. Then,

- $\{\text{weak minima of (P)}\} = \bigcup_{\alpha \geq 0} S_\alpha$;
- $\{\text{minima of (P)}\} = \bigcup_{\alpha > 0} S_\alpha$.

Exercise - Find the set of minima and weak minima by means of the scalarization method, considering the linear multiobjective problem:

$$\begin{cases} \min\ (x_1 - x_2\ ,\ x_1 + x_2) \\ -2\,x_1 + x_2 \leq 0 \\ -x_1 - x_2 \leq 0 \\ 5\,x_1 - x_2 \leq 6 \end{cases}$$

```
%% Problem data
% min Cx
% Ax <= b
C = [ 1  -1
      1  1] ;
 A =[ -2    1
   -1      -1
    5     -1];
b = [ 0
      0
      6] ;
% % solve the scalarized problem with 0 < alfa
< 1
MINIMA=[ ];
LAMBDA=[ ];
for alfa = 0.01 : 0.01 : 0.99
    [x,fval,exitflag,output,lambda] =
linprog(alfa*C(1,:)+(1-alfa)*C(2,:),A,b) ;
    plot(x(1),x(2),'g*');
    MINIMA=[MINIMA;alfa, x'];
    LAMBDA=[LAMBDA;alfa,lambda.ineqlin'];
    hold on
    grid on
end
%
% % solve the scalarized problem with alfa = 0
%
figure;
alfa = 0;
[xalfa0,f0,exitflag,output,lambda0] =
linprog(alfa*C(1,:)+(1-alfa)*C(2,:),A,b) ;
plot(xalfa0(1),xalfa0(2),'r*');
hold on
grid on
%
% % solve the scalarized problem with alfa = 1
%
alfa = 1;
[xalfa1,f1,exitflag,output,lambda1] =
linprog(alfa*C(1,:)+(1-alfa)*C(2,:),A,b) ;
plot(xalfa1(1),xalfa1(2),'r*');
hold on
grid on
%
```

If x* is the unique global minimum of Pα for some α, then x* is a minimum of (P).

Exercise - Consider the nonlinear multiobjective problem (P). Find the set of minima and weak minima by means of the scalarization method.

$$\begin{cases} \min\ (x_1^2 + x_2^2 + 2x_1 - 4x_2\ ,\ x_1^2 + x_2^2 - 6x_1 - 4x_2) \\ -x_2 \leq 0 \\ -2x_1 + x_2 \leq 0 \\ 2x_1 + x_2 \leq 4 \end{cases}$$

The scalarized problem P_α is

$$\begin{cases} \min\ (\alpha_1(x_1^2 + x_2^2 + 2x_1 - 4x_2) + \alpha_2(x_1^2 + x_2^2 - 6x_1 - 4x_2)) \\ -x_2 \le 0 \\ -2x_1 + x_2 \le 0 \\ 2x_1 + x_2 \le 4 \end{cases}$$

We note that the feasible set X is convex and the objective function of P_α is strongly convex for any $\alpha = (\alpha_1, \alpha_2) \in \mathbb{R}^2_+$ with $\alpha_1 + \alpha_2 = 1$ so that the set of minima and weak minima coincide.

The scalarized problem P_α becomes:

$$\begin{cases} \min\ (\frac{1}{2}x^T(\alpha_1 Q_1 + \alpha_2 Q_2)x + (\alpha_1 c1^T + \alpha_2 c2^T)x) \\ Ax \le b \end{cases}$$

which can be solved by the Matlab function "quadprog".

```
Q1 = [2 0; 0 2] ;
Q2 = [2 0; 0 2] ;
c1=[2 -4]';
c2=[-6 -4]';
A =[ 0 -1; -2 1; 2 1 ];
b = [0 0 4]';
% solve the scalarized problem with  alfa1 in [0,1]
MINIMA=[ ]; % First column: value of alfa1
LAMBDA=[ ]; % First column: value of alfa1
for alfa1 = 0 : 0.01 : 1
[x,fval,exitflag,output,lambda] =
quadprog(alfa1*Q1+(1-alfa1)*Q2,alfa1*c1+(1-alfa1)*c2,A,b) ;
MINIMA=[MINIMA; alfa1 x'];
LAMBDA=[LAMBDA;alfa1,lambda.ineqlin'];
end
 plot(MINIMA(:,2),MINIMA(:,3), 'r*')
```

fmincon -

The function fmincon solves a problem of the form:

$$\begin{cases} \min\ f(x) \\ Ax \le b \\ Dx = e \\ l \le x \le u \\ c(x) \le 0 \\ ceq(x) = 0 \end{cases}$$

where x, b, e, l, u are vectors, A, D are matrices, c and ceq are functions that return vectors and f is a scalar function.

```
% solve the scalarized problem with 0 =< alfa1 <= 1
MINIMA=[ ]; % First column: value of alfa1
LAMBDA=[ ];
for alfa1 = 0 : 0.01 : 1

   FUN=@(x) (2*alfa1-1)*x(1)+x(2);

   NONLINCON= @(x) const(x);
   [x,fval,exitflag,output,lambda] = fmincon(FUN,[0;0],[],[],[],[],[],[],NONLINCON) ;
MINIMA=[MINIMA; alfa1, x'];
LAMBDA=[LAMBDA; alfa1, lambda.ineqnonlin];
end
plot(MINIMA(:,2),MINIMA(:,3))
function [C,Ceq]=const(x)
C=x(1)^2 +x(2)^2 -1;
Ceq=[];
end
```

10. Non-cooperative game theory

Definition -

A non-cooperative game (in normal form) is defined by a set of N players, where each player i has a set X_i of strategies and a cost function $f_i : X_1 \times \cdots \times X_N \to \mathbb{R}$.
The aim of each player i consists in solving the optimization problem

$$\begin{cases} \min\ f_i(x^1, x^2, \dots, x^{i-1}, x^i, x^{i+1}, \dots, x^N) \\ x^i \in X_i \end{cases}$$

Definition (Nash Equilibrium) -

In a two-person non-cooperative game, a pair of strategies $(\bar{x}, \bar{y})$ is a **Nash equilibrium** if

$$f_1(\bar{x}, \bar{y}) = \min_{x \in X} f_1(x, \bar{y}), \qquad f_2(\bar{x}, \bar{y}) = \min_{y \in Y} f_2(\bar{x}, y).$$

In other words, $(\bar{x}, \bar{y})$ is a **Nash equilibrium** if and only if

- $\bar{x}$ is the best response of player 1 to strategy $\bar{y}$ of player 2
- $\bar{y}$ is the best response of player 2 to strategy $\bar{x}$ of player 1

Matrix Game -

A matrix game is a two person non-cooperative game where:

- X and Y are finite sets: X = {1…m}, Y = {1…n}
- $f_2 = -f_1$ (*zero-sum game*)

It can be represented by a m x n matrix C, where $f_1(i, j) = c_{ij}$ is the amount of money player 1 pays to player 2 if player 1 chooses strategy i and player 2 chooses strategy j.

Strictly dominated strategies -

Given a two-persons non-cooperative game, a strategy $x \in X$ is strictly dominated by $\tilde{x} \in X$ if

$$f_1(x, y) > f_1(\tilde{x}, y) \qquad \forall\ y \in Y.$$

Similarly, a strategy $y \in Y$ is strictly dominated by $\tilde{y} \in Y$ if

$$f_2(x, y) > f_2(x, \tilde{y}) \qquad \forall\ x \in X.$$

Strictly dominated strategies can be deleted from the game.

Mixed strategy -

If C is a $m \times n$ matrix game, then a mixed strategy for player 1 is a m-vector of probabilities and we consider $X = \{x \in \mathbb{R}^m : x \geq 0, \ \sum_{i=1}^m x_i = 1\}$ the set of mixed strategies of player 1.
The vertices of X, i.e., $e_i = (0, \dots, 0, 1, 0, \dots, 0)$ are pure strategies of player 1.

Similarly, a mixed strategy for player 2 is a n-vector of probabilities and $Y = \{y \in \mathbb{R}^n : y \geq 0, \ \sum_{j=1}^n y_j = 1\}$ is the set of mixed strategies of player 2.

The expected costs are $f_1(x, y) = x^T C y$ (player 1), $f_2(x, y) = -x^T C y$ (player 2).

with

$$x^T C y = \sum_{i=1}^m \sum_{j=1}^n x_i c_{ij} y_j.$$

Mixed strategy Nash equilibria -

If C is a $m \times n$ matrix game, then $(\bar{x}, \bar{y}) \in X \times Y$ is a mixed strategies Nash equilibrium if

$$\max_{y \in Y} \bar{x}^{\mathsf{T}} C y = \bar{x}^{\mathsf{T}} C \bar{y} = \min_{x \in X} x^{\mathsf{T}} C \bar{y},$$

or, equivalently,

$$\bar{x}^{\mathsf{T}} C y \leq \bar{x}^{\mathsf{T}} C \bar{y} \leq x^{\mathsf{T}} C \bar{y}, \quad \forall (x, y) \in X \times Y,$$

i.e., $(\bar{x}, \bar{y})$ is a saddle point of the function $f_1(x, y) = x^{\mathsf{T}} C y$ on $X \times Y$.

Corollary - Any matrix game has at least a mixed strategies Nash equilibrium

$(\bar{x}, \bar{y})$ is a mixed strategies Nash equilibrium if and only if

$$\begin{cases} \bar{x} \text{ is an optimal solution of } \min\limits_{x \in X} \max\limits_{y \in Y} x^{\mathsf{T}} C y \\ \bar{y} \text{ is an optimal solution of } \max\limits_{y \in Y} \min\limits_{x \in X} x^{\mathsf{T}} C y \end{cases}$$

with optimal values both equal to $\bar{x}^T C \bar{y}$.

Theorem -

1. The problem $\min\limits_{x \in X} \max\limits_{y \in Y} x^{\mathsf{T}} C y$ is equivalent to the linear programming problem

$$\begin{cases} \min \ v \\ v \geq \sum_{i=1}^{m} c_{ij} x_i \quad \forall\, j = 1, \dots, n \\ x \geq 0, \quad \sum_{i=1}^{m} x_i = 1 \end{cases} \qquad (P_1)$$

2. The problem $\max\limits_{y \in Y} \min\limits_{x \in X} x^{\mathsf{T}} C y$ is equivalent to the linear programming problem

$$\begin{cases} \max \ w \\ w \leq \sum_{j=1}^{n} c_{ij} y_j \quad \forall\, i = 1, \dots, m \\ y \geq 0, \quad \sum_{j=1}^{n} y_j = 1 \end{cases} \qquad (P_2)$$

Exercise -

```
%% Exercise 2 - matrix game - mixed strategies Nash equilibrium
clear all
C=[7,15,2,3;4 2 3 10; 5 3 4 12]
m = size(C,1);
n = size(C,2);
c=[zeros(m,1);1];
A= [C', -ones(n,1)]; b=zeros(n,1); Aeq=[ones(1,m),0]; beq=1;
lb= [zeros(m,1);-inf]; ub=[ ];
[sol,Val,exitflag,output,lambda] = linprog(c, A,b, Aeq, beq, lb, ub);
x = sol(1:m)
y = lambda.ineqlin
```

Bimatrix game -

A **bimatrix game** is a two-person non-cooperative game where:

- the sets of pure strategies are finite, hence the sets of mixed strategies are $X = \{x \in \mathbb{R}^m : \ x \geq 0, \quad \sum_{i=1}^m x_i = 1\}$ and $Y = \{y \in \mathbb{R}^n : \ y \geq 0, \quad \sum_{j=1}^n y_j = 1\}$;
- $f_2 \neq -f_1$ (non-zero-sum game), the cost functions are $f_1(x, y) = x^{\mathsf{T}} C_1 y$ and $f_2(x, y) = x^{\mathsf{T}} C_2 y$, where C_1 and C_2 are $m \times n$ matrices.

Exercise - Solve a KKT system associated with a bimatrix game

```
C1=[3,3;4 1;6 0];
C2=[3 4;4 0;3 5];
[m,n] = size(C1);
H=[zeros(m,m),C1+C2,ones(m,1), zeros(m,1); C1'+C2',zeros(n,n),zeros(n,1),ones(n,1); ones(1,m),
zeros(1,n+2); zeros(1,m),ones(1,n),0,0];
%X0=[0,1,0,0,1,1,1]'; % m+n+2 vector
X0=[rand(5,1);10-20*rand(2,1)]
%X0=[0,0,1,1,0,10-20*rand(1,2)]';
Ain=[-C2', zeros(n,n),zeros(n,1),-ones(n,1);zeros(m,m), -C1,-ones(m,1),zeros(m,1)];
bin=zeros(n+m,1);
Aeq=[ones(1,m),zeros(1,n+2);zeros(1,m),ones(1,n),0,0];
beq=[1;1]; LB=[zeros(m+n,1);-Inf;-Inf];
UB=[ones(m+n,1);Inf;Inf];
[sol,fval,exitflag,output]=fmincon(@(X) 0.5*X'*H*X, X0, Ain,bin, Aeq,beq,LB,UB)
x = sol(1:m)
y = sol(m+1:m+n)
```

Convex games -

We consider a general two-persons non-cooperative game where f1, g1, f2, and g2 are continuously differentiable. The game is said convex if the optimization problem of each player is convex.

Theorem -

If the feasible regions $X = \{x \in \mathbb{R}^m : \; g_i^1(x) \leq 0 \;\; i = 1, \dots, p\}$ and $Y = \{y \in \mathbb{R}^n : \; g_j^2(y) \leq 0 \;\; j = 1, \dots, q\}$ are closed, convex and bounded, the cost function $f_1(\cdot, y)$ is quasiconvex for any $y \in Y$ and $f_2(x, \cdot)$ is quasiconvex for any $x \in X$, then there exists at least a Nash equilibrium.

The quasiconvexity of the cost function is crucial.

Theorem (KKT conditions) -

- If $(\bar{x}, \bar{y})$ is a Nash equilibrium and the Abadie constraints qualification holds both in $\bar{x}$ and $\bar{y}$, then there exist $\lambda^1 \in \mathbb{R}^p$, $\lambda^2 \in \mathbb{R}^q$ such that

$$\begin{cases} \nabla_x f_1(\bar{x}, \bar{y}) + \sum_{i=1}^{p} \lambda_i^1 \nabla g_i^1(\bar{x}) = 0 \\ \lambda^1 \geq 0, \qquad g^1(\bar{x}) \leq 0 \\ \lambda_i^1 g_i^1(\bar{x}) = 0, \quad i = 1, \dots, p \\ \nabla_y f_2(\bar{x}, \bar{y}) + \sum_{j=1}^{q} \lambda_j^2 \nabla g_j^2(\bar{y}) = 0 \\ \lambda^2 \geq 0, \qquad g^2(\bar{y}) \leq 0 \\ \lambda_j^2 g_j^2(\bar{y}) = 0, \quad j = 1, \dots, q \end{cases}$$

- If $(\bar{x}, \bar{y}, \lambda^1, \lambda^2)$ solves the above system and the game is convex, then $(\bar{x}, \bar{y})$ is a Nash equilibrium.