Programmazione Matematica: Introduzione

Alessandro Hill

Basato sul materiale di

Daniele Vigo (D.E.I.) & Marco Boschetti (D.M.).

rev. 1.1(AH) – 2024

Preliminari

Notazione

- R : insieme dei numeri reali (R^n : spazio vettoriale a n dimensioni)
- Z : insieme dei numeri interi ($Z+$: numeri interi positivi)
- $[a, b] = \{x \in R : a \leq x \leq b\}$ (intervallo chiuso)
- $(a, b) = \{x \in R : a < x < b\}$ (intervallo aperto)

norma 2 (sommatoria delle componenti di un vettore, con componenti alla seconda, tutto sotto radice)

- $\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$: norma euclidea; $|z|$: valore assoluto dello scalare $z \in R$
- $Q = \{q_1, \dots q_n\}$: insieme degli n elementi $q_1, \dots, q_n$
- $Q = \{x \in R^n : P(x)\}$: insieme dei punti di R^n che soddisfano le condizioni P
- $|Q|$: cardinalità dell'insieme Q
- $\text{argmin}\{f(i) : i \in I\}$: $i^* \in I$ tale che $f(i^*) = \min\{f(i) : i \in I\}$

i* è il valore di i che minimizza la funzione f, e f(i*) è il valore minimo che f assume su I

- $\lfloor z \rfloor = \max\{i \in Z : i \leq z\}$; $\lceil z \rceil = \min\{i \in Z : i \geq z\}$

Significato: il Floor è il più grande numero intero i che è minore o uguale a z.
Esempio: Se z=3.7, [z]=3 perché 3 è il più grande intero che è minore o uguale a 3.7.

Significato: il Ceil è il più piccolo numero intero i che è maggiore o uguale a z.
Esempio: Se z=3.7, [z]=4 perché 4 è il più piccolo intero che è maggiore o uguale a 3.7.

Notazione

▷ $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$: vettore colonna n dimensionale ($x \in R^n$)

▷ $c^T = [c_1, \ldots, c_n]$: vettore riga n-dimensionale

▷ $c^T x = \sum_{j=1}^{n} c_j x_j$: prodotto scalare (o anche cx) (vettore riga per vettore colonna)

▷ $A = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \ldots & a_{mn} \end{bmatrix}$: matrice m × n

riga

colonna

▷ $A = \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} = \begin{bmatrix} a^1 \\ \vdots \\ a^m \end{bmatrix} = [A_1, \ldots, A_n] = [a_1, \ldots, a_n]$

riga matrice per vettore colonna

▷ $Ax = \begin{bmatrix} \sum_{j=1}^{n} a_{1j} x_j \\ \vdots \\ \sum_{j=1}^{n} a_{mj} x_j \end{bmatrix} = \begin{bmatrix} a_1^T x \\ \vdots \\ a_m^T x \end{bmatrix} = \begin{bmatrix} a^1 x \\ \vdots \\ a^m x \end{bmatrix}$

riga matrice per vettore colonna = bi

▷ $Ax = b \longrightarrow \begin{cases} a_1^T x = b_1 \\ \vdots \\ a_m^T x = b_m \end{cases} \equiv \begin{cases} a^1 x = b_1 \\ \vdots \\ a^m x = b_m \end{cases}$

▷ rango(A): rango di A

Il rango di una matrice è il numero massimo di righe (o colonne) linearmente indipendenti.

▷ det(A): determinante di A

▷ A^{-1}: matrice inversa di A

Combinazione Convessa

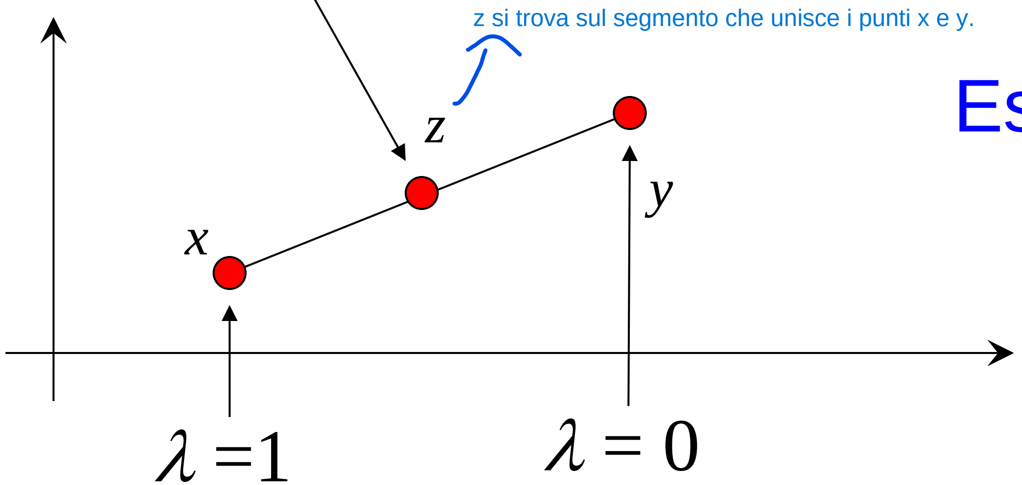
Un vettore | due vettori

Def.: z, è **combinazione convessa** di x,y se

uno scalare

$\exists\ \lambda \in [0,1]$ tale che $z = \lambda x + (1-\lambda)\, y$

z si trova sul segmento che unisce i punti x e y.



Es. $x,y \in R^2$

Combinazione Convessa (2)

Def.: Combinazione convessa di K punti $p_1, p_2, \ldots, p_K \in R^n$

$$z = \sum_{i=1,K} \lambda_i p_i \quad con\ \lambda_i \geq 0 \quad e \quad \sum_{i=1,K} \lambda_i = 1$$

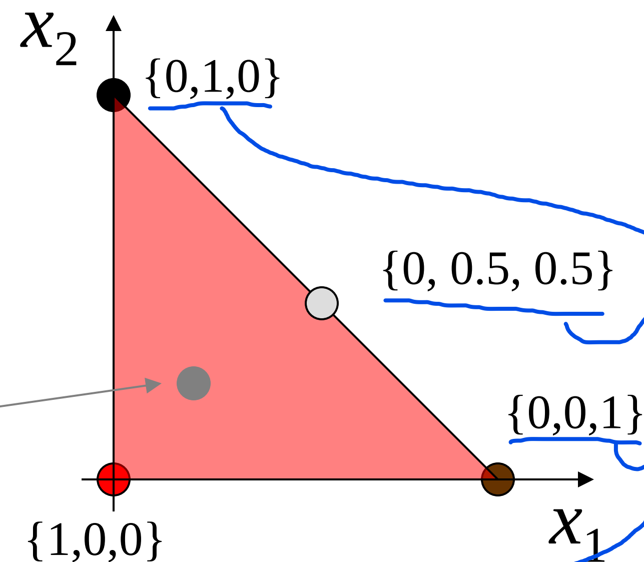
Es. $z = \lambda x + (1-\lambda) y, \quad \lambda_1 = \lambda > 0,\ \lambda_2 = 1-\lambda,\ \lambda_1 + \lambda_2 = 1$

In questo caso, z appartiene all'inviluppo convesso dei vettori x1,…,xn; cioè alla regione delimitata dai punti x1,…,xn.

$$p_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad p_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad p_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\lambda_i = \{0.5, 0.2, 0.3\} \quad z = \begin{bmatrix} 0.2 \\ 0.3 \end{bmatrix}$$

zx = lambda1 * p1x + lambda2 * p2x + lambda3 * p3x
zy = lambda1 * p1y + lambda2 * p2y + lambda3 * p3y



sono i lambda in riferimento ai punti

Insiemi Convessi

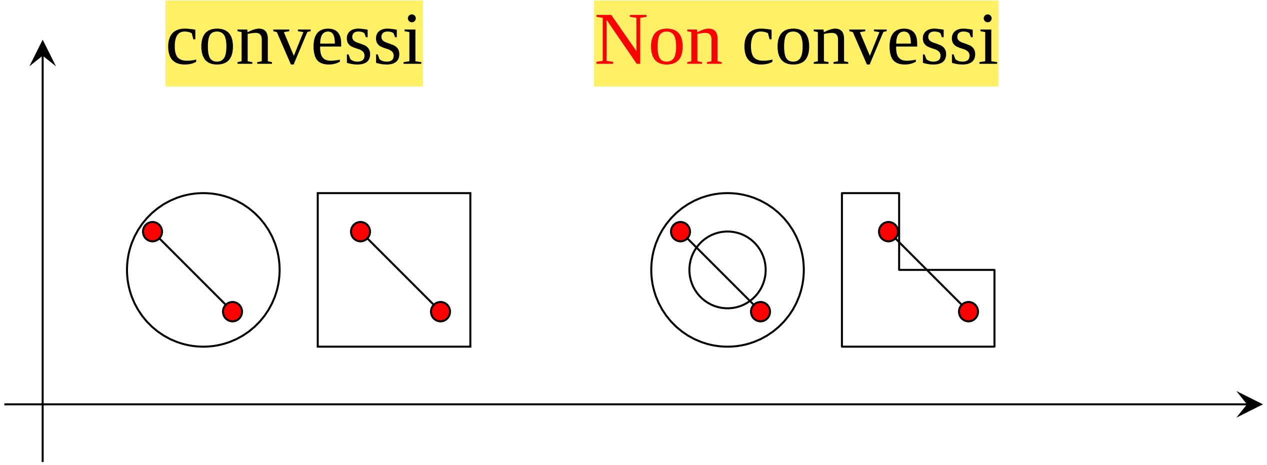
Dati x e y vettori appartenenti a F, anche z, combinazione convessa e vettore, appartiene a F, allora F insieme convesso

Def.: $F \subseteq \mathbb{R}^n$ è un **insieme convesso** se

$$\forall\, x, y \in F \ \text{ e } \forall\, \lambda \in [0,1],$$

$$z = \lambda x + (1-\lambda)\, y \in F$$

Es. $x, y \in \mathbb{R}^2$



La convessità richiede che per ogni coppia di punti nell'insieme, ogni punto sulla retta che li congiunge appartenga all'insieme stesso.

Proprietà degli Insiemi Convessi

Proprietà 0: R^n è convesso (ovvio)

Proprietà 1: Dati F_i convessi $\Rightarrow$

$$F = \cap F_i \text{ è convesso}$$

l'intersezione di insiemi convessi è anch'essa un insieme convesso.

DIM.:

$$x,y \in F \Rightarrow x,y \in F_i \; \forall i$$

Poiché ogni Fi è convesso, z appartiene a ciascun Fi

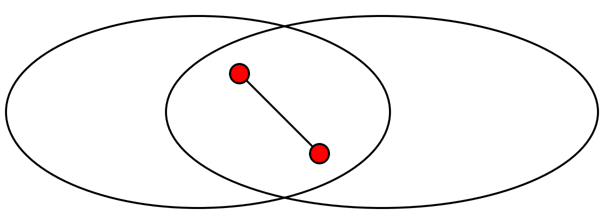
$$\Rightarrow z = \lambda x + (1- \lambda)\, y \in F_i \; \forall i, \; \forall \lambda$$

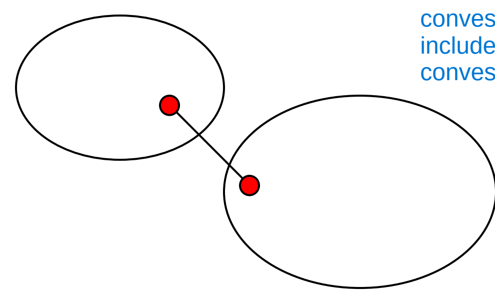
$$\Rightarrow z \in \cap F_i$$

• Pertanto, intersezione di Fi è convesso

L'unione di insiemi convessi in generale non è convessa. Questo accade perché l'unione può includere punti tra i quali una combinazione convessa non appartiene all'unione stessa.

$\cap F_i$





$\cup F_i$

Funzioni Convesse

Def.: Dato $F \subseteq \mathrm{R}^n$ convesso, $\varphi : F \rightarrow \mathrm{R}$ è **convessa** in F

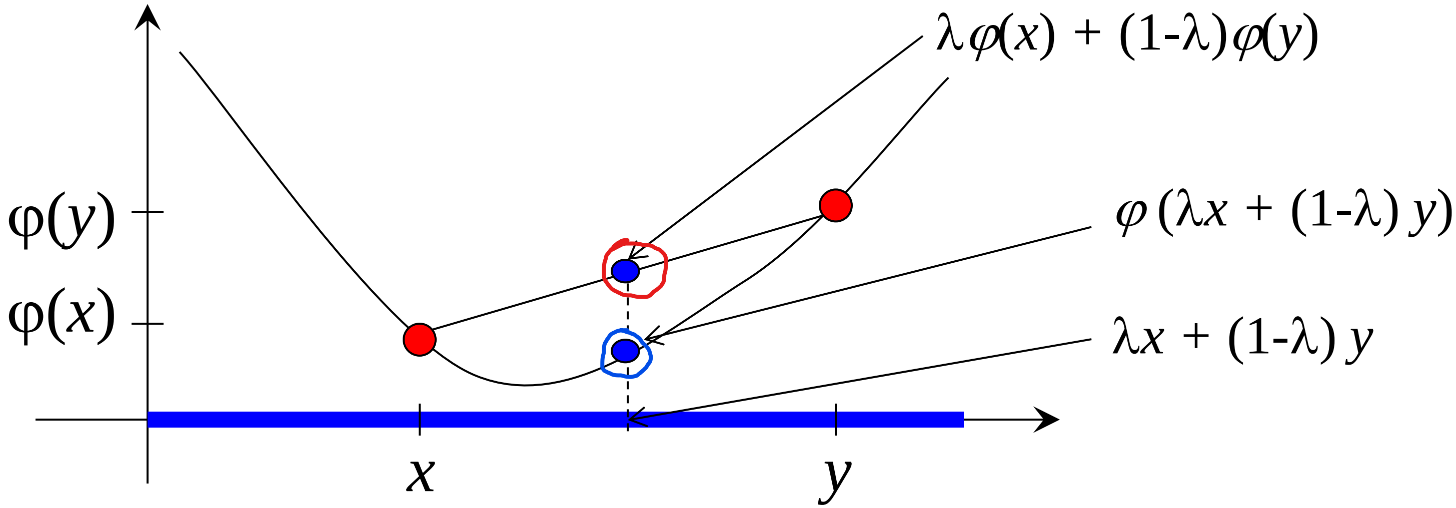
se $\forall\, x,y \in F,\ \forall\, \lambda \in [0,1]$, si ha

$$\varphi(\lambda x + (1 - \lambda)\, y) \leq \lambda\, \varphi(x) + (1 - \lambda)\varphi(y)$$

il punto del segmento tra x e y, ma sulla funzione

punto sul segmento tra x e y

Es. $F = [0,1] \subset \mathrm{R}$



Problemi di ottimizzazione

Problemi di Ottimizzazione

- $x = (x_1, \ldots, x_n) \in \mathrm{R}^n$: vettore di variabili decisionali
  - prodotti da realizzare, istanti in cui produrli ...
  - merci o materie prime da stoccare: quanto, quando …
  - luogo in cui realizzare una infrastruttura ...
  - tratti di strada da scegliere in un percorso ...
- $F \subseteq \mathrm{R}^n$: insieme delle soluzioni ammissibili (regione ammissibile)
- $\varphi : F \to \mathrm{R}$: funzione obiettivo (f. costo)

$$\boxed{(P) \quad \min_{x \in F} \varphi(x)}$$

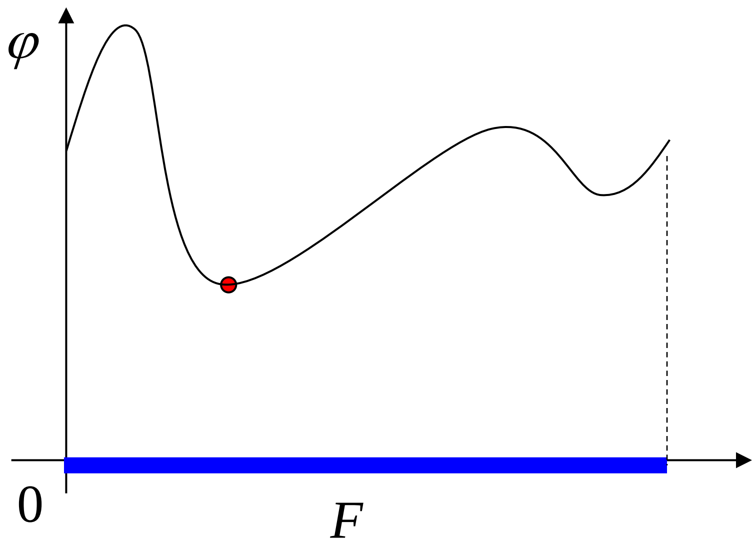
Il problema P va a definire se dobbiamo Massimizzare o Minimizzare la Funzione Obiettivo

Problemi di Ottimizzazione (2)

$$(P) \qquad \min_{x \in F} \varphi(x)$$

ovvero determinare $x^* \in F$ (ottimo globale) tale che:

$$\varphi(x^*) \le \varphi(x) \qquad \forall\, x \in F$$



In generale φ ed F sono qualsiasi

storicamente detti problemi di programmazione

Regione Ammissibile

La regione ammissibile *F* può essere definita:

- esplicitamente: specificando le proprietà di $x \in F$

  Es. $[0,1]^2$; *x* intere nell'ipercubo di lato 1

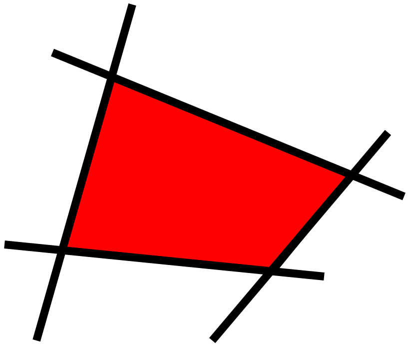
- implicitamente: equazioni e disequazioni

Vincoli di Disuguaglianza

$$\text{Es. } F := \{x \in \mathbb{R}^n : g_i(x) \leq 0,\ (i=1,\ \ldots,\ m)$$

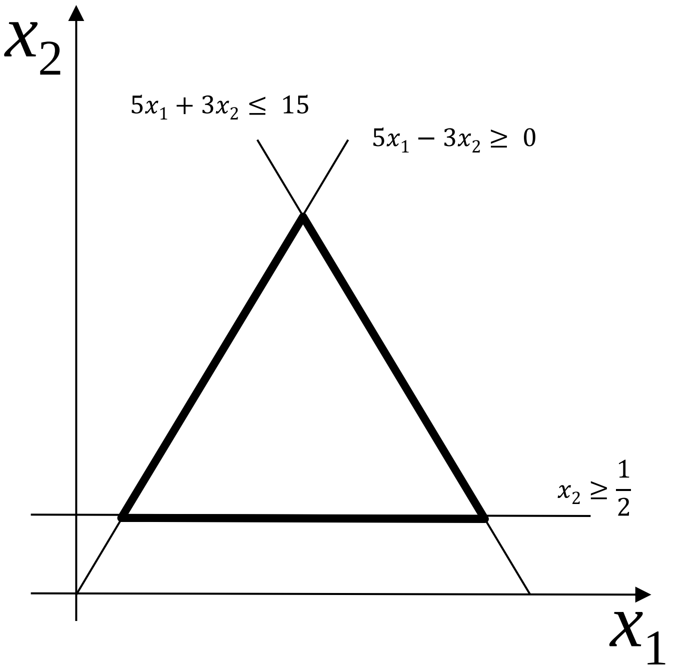
Vincoli di Uguaglianza

$$h_j(x) = 0,\ (j=1,\ \ldots,\ p)\}$$



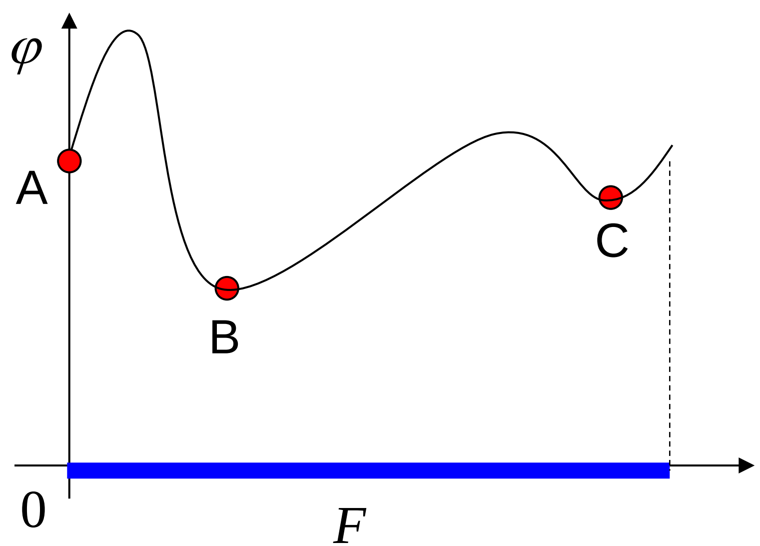
Esempio di regione ammissibile

$$F = \{x \in \mathbb{R}^2 : 5x_1 + 3x_2 \leq 15;\ 5x_1 - 3x_2 \geq 0;\ x_2 \geq \frac{1}{2},\ x_1, x_2 \geq 0\}$$



Minimi Locali e Globali

- non è detto che x^* esista ($F = \emptyset$) o che sia unica
- possono esistere ottimi (minimi) locali e globali



(P) richiede di trovare almeno un ottimo globale

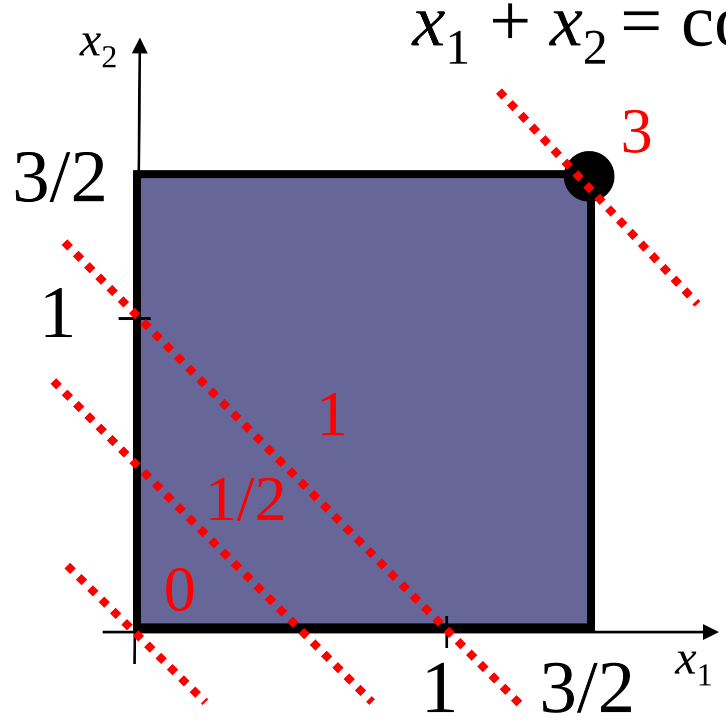
Il Problema

Esempio 1

problema continuo: $F = [0,3/2]^2 \subset R^2$

$\max \varphi(x) = x_1 + x_2$

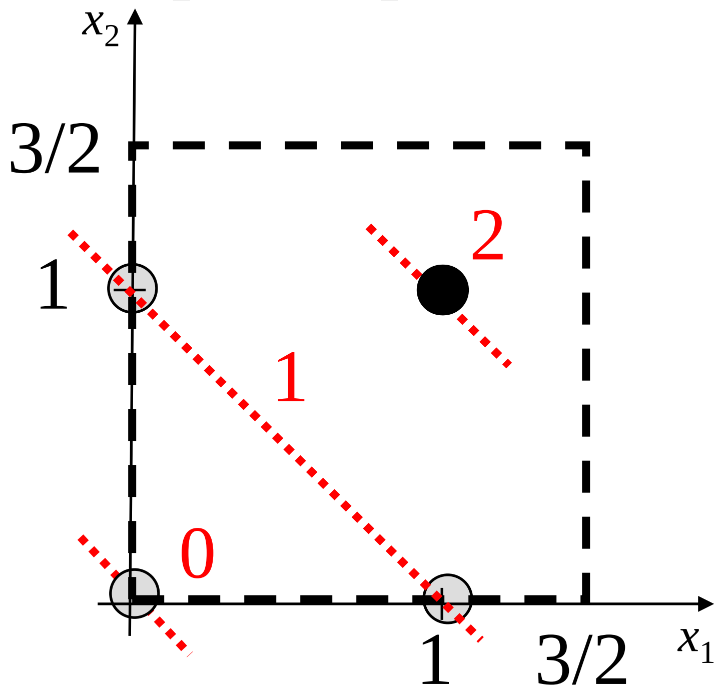
$x_1 + x_2 =$ costante (livello di «costo»)



Esempio 2

- problema discreto:

$$F = [0,3/2]^2 \cap Z^2 \qquad \max \varphi(x) = x_1 + x_2$$



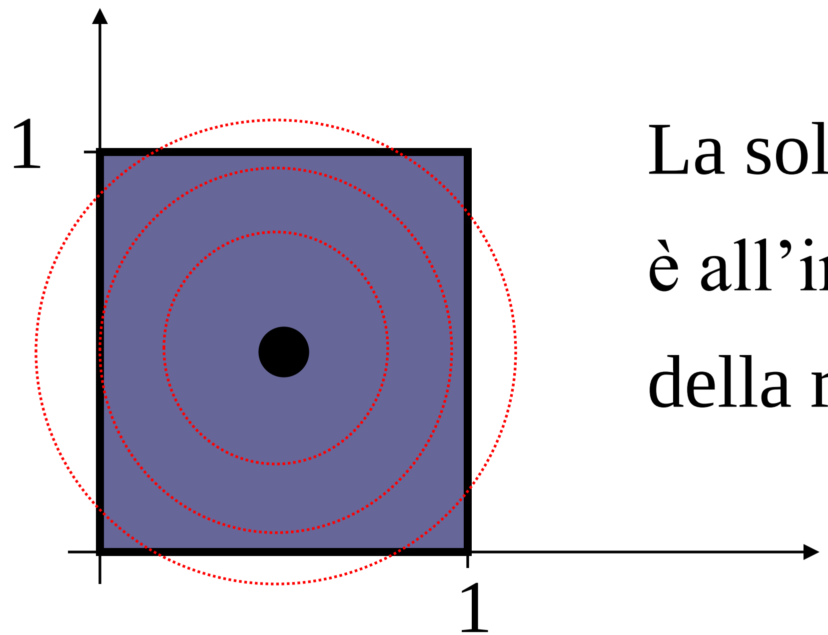
si può valutare $\varphi(x)$
in ciascun vertice
se F=$[0,1]^{100} \cap Z^{100}$
$\Rightarrow 2^{100} \sim 10^{30}$ valutazioni

Esempio 3

- problema continuo:

$$F = [0,1]^2 \subset \mathrm{R}^2$$

$$\min \varphi(x) = (x_1 - 1/2)^2 + (x_2 - 1/2)^2$$

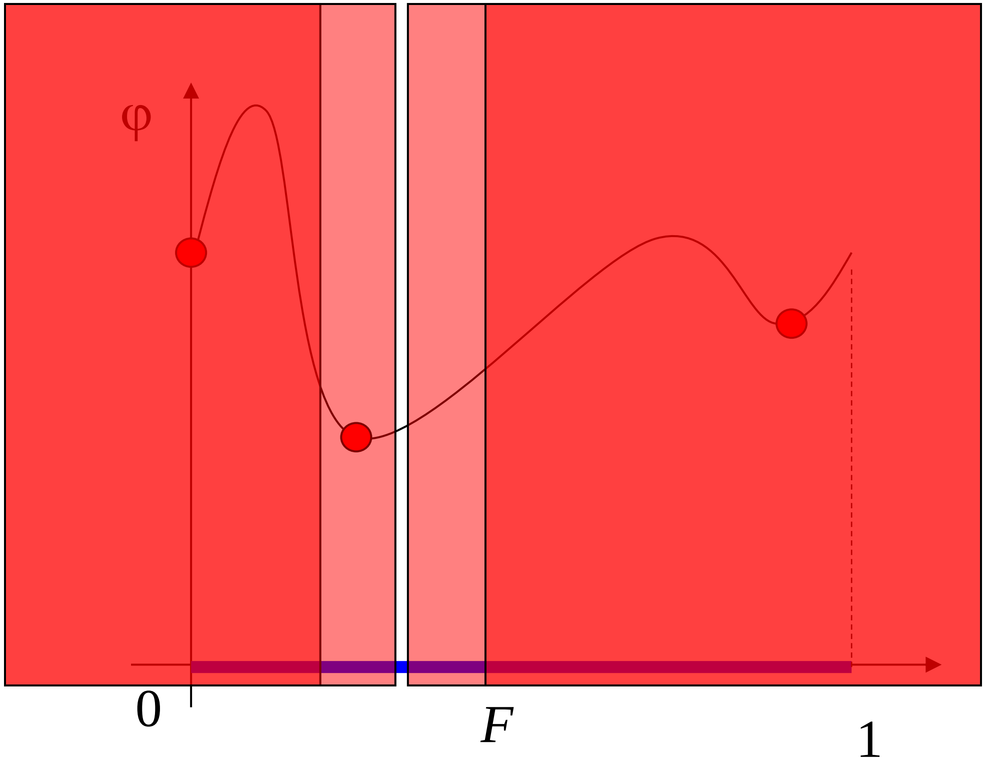


La soluzione ottima
è all'interno
della regione ammissibile

Algoritmi Numerici

- Un algoritmo non ha visione completa di φ ed F
- valuta $\varphi(x)$ in una sequenza di punti $x \in F$

l'algoritmo prende decisioni basate su una visione parziale della funzione obiettivo e della regione ammissibile, adattando la sua ricerca in base ai risultati ottenuti in ogni iterazione.



Gli algoritmi numerici di ottimizzazione sono generalmente eseguiti su un intervallo ristretto rispetto all'intera regione ammissibile F.

Algoritmi Numerici (2)

Gli algoritmi per i problemi di ottimizzazione sono generalmente di tipo iterativo:

1. Sia x_0 ($\in F$) una soluzione iniziale;
   $k := 0$;
2. repeat
   2.1 verifica l'ottimalità (locale) di x_k
   2.2 se x_k non ottima genera x_{k+1} ($\in F$) tale che $\varphi(x_{k+1}) \leq \varphi(x_k)$ e poni $k := k + 1$;

until (x_k ottima) o (*condizione di terminazione*)

Algoritmi Numerici (3)

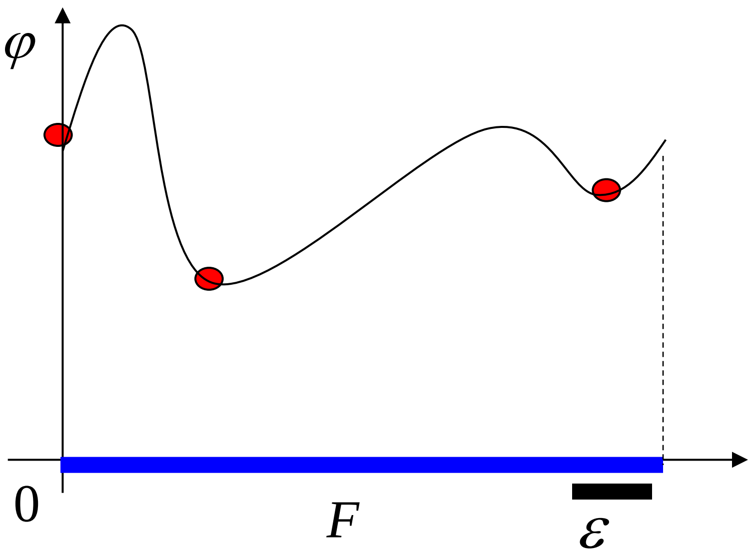
- La sequenza $x_0, x_1, \ldots, x_k$ converge alla soluzione ottima x^* (o ad un ottimo locale)
- Nel caso generale (φ ed F qualsiasi) la convergenza è ad un ottimo locale ed il numero di iterazioni è molto elevato
- Nel caso continuo si termina quando si è raggiunta l'approssimazione desiderata (piccole variazioni tra iterazioni successive)

Ottimi Locali ed Intorni

Def.: $y \in F$ è un ottimo locale se $\exists$ un intorno $N \subseteq F$

tale che $\varphi(y) \leq \varphi(x) \quad \forall\, x \in N$

Es. $N_\varepsilon(y) := \{x \in F : \| y - x \| \leq \varepsilon ,\ \varepsilon > 0\}$ (intorno euclideo)



N è esatto se un ottimo locale rispetto ad N è ottimo globale

(Es. N_1)

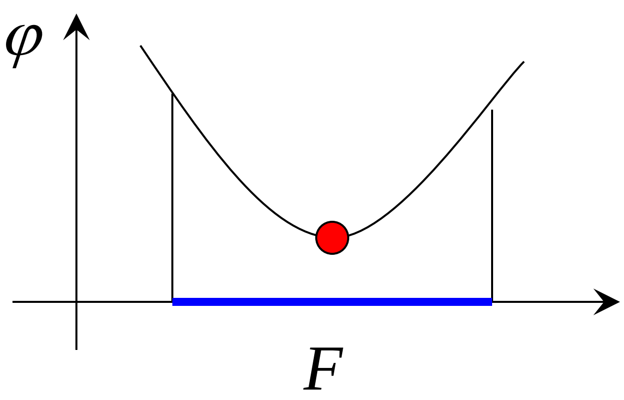
Convessità ed Intorno Euclideo

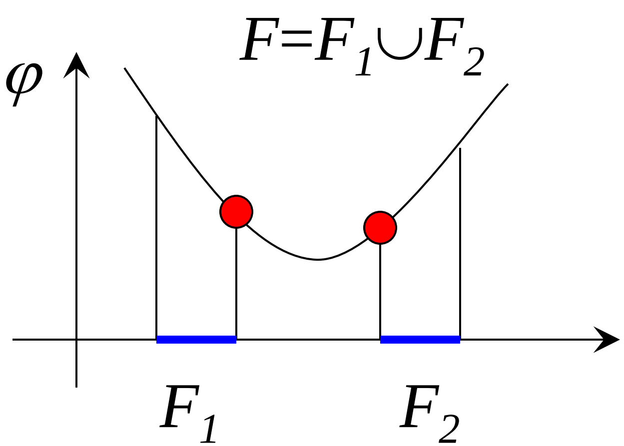
per le funzioni convesse su insiemi convessi, ottimo locale e ottimo globale coincidono.

Th.: Dato (F,φ) con $F \subseteq R^n$ convesso e φ convessa, $N_\varepsilon(x) = \{ y \in F : \| x- y \| \le \varepsilon \}$ è esatto $\forall \varepsilon > 0$

intorno euclideo di raggio epsilon centrato in un punto x appartenente ad F.

$\Rightarrow$ (ottimo locale $\equiv$ ottimo globale)





Classificazione

(I vincoli di disuguaglianza gi(x)<=0 restringono lo spazio delle soluzioni ammissibili F)

φ, g_i, h_j qualunque $\Rightarrow$ Progr. Non Lineare (PNL, NLP)

I vincoli di uguaglianza hj(x)=0 restringono ulteriormente lo spazio delle soluzioni ammissibili F.

Non esistono algoritmi generali di ottimizzazione

$\exists$ metodi che convergono a ottimi locali

φ, g_i, convesse $\Rightarrow$ Programmazione Convessa (PC,CP)

h_j lineari

ottimo locale $\equiv$ ottimo globale

$\exists$ algoritmi ma non efficienti

Classificazione (2)

φ, g_i, h_j lineari ⇒ Programmazione Lineare (PL, LP)

ottimo locale ≡ ottimo globale

∃ algoritmi efficienti (Simplesso, Interior Point, …)

PL con variabili intere ⇒ Progr. Lineare Intera (PLI, ILP)

Problema difficile (∝ PNL)

∃ algoritmi generali (Branch-and-Bound, …)