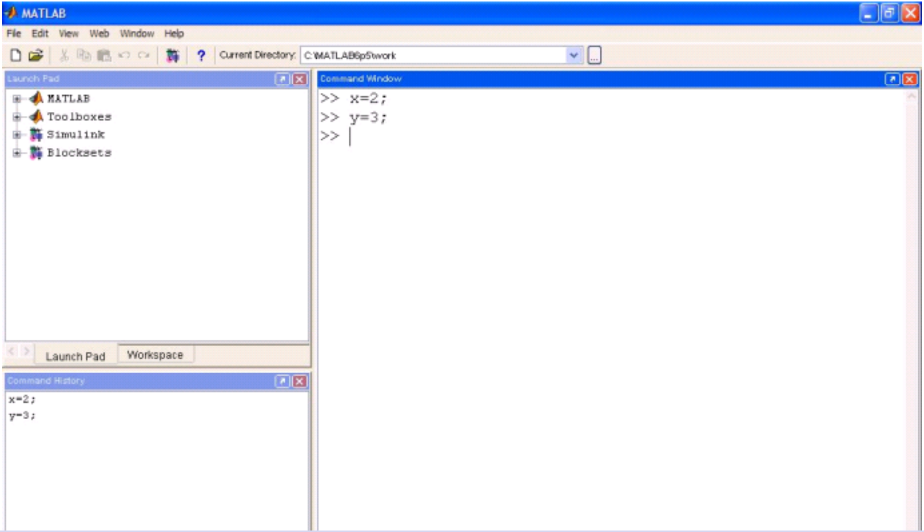
Ricerca Operativa

Massimo Pappalardo
Dipartimento di Informatica
Largo B. Pontecorvo 3, Pisa
massimo.pappalardo@unipi.it

Laurea in Ingegneria Informatica
Universitá di Pisa
A.A. 2022/'23

Introduzione all'Optimization Toolbox di MATLAB

Per avviare MATLAB é sufficiente fare doppio clic sull'icona MATLAB



Barra dei menu

In essa si possono distinguere vari parti:

- la *barra dei menu* É la riga in alto contenente e comprende i nomi di 6 menu, ognuno contenente i comandi per fornire istruzioni al programma;

  File *Edit* *View* *Web* *Window* *Help*

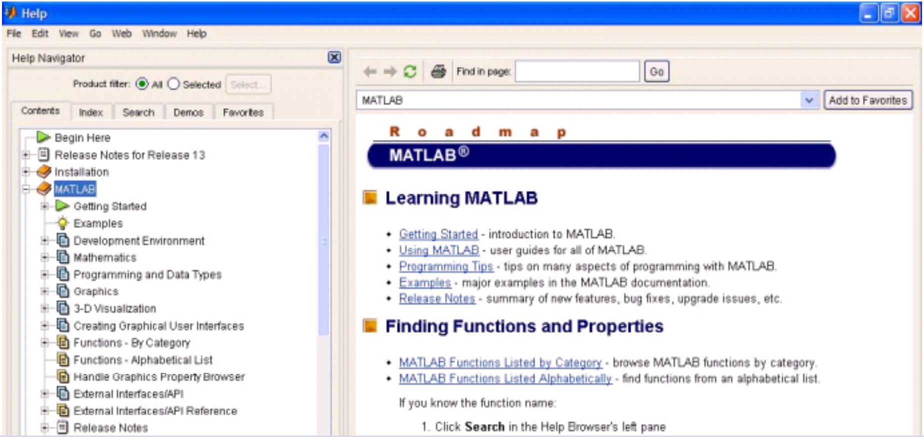
- la *barra degli strumenti* rappresenta operazioni di MATLAB. Fare clic equivale ad aprire il menu e selezionare la corrispondente opzione. Le prime sette icone da sinistra corrispondono alle opzioni *New File, Open File, Cut, Copy, Paste, Undo e Redo*. L'ottava icona avvia Simulink; la nona icona (quella con il punto interrogativo) permette di accedere alla guida di MATLAB. Il riquadro a destra della barra degli strumenti indica la cartella di lavoro corrente (*Current Directory*).
- *Command Window*(la finestra dei comandi) in cui si digitano i nomi dei comandi o le istruzioni da eseguire.

Barre

- *Command History* (la cronologia dei comandi) visualizza tutti i comandi che sono stati digitati durante la sessione di lavoro corrente.
- *Launch Pad*(strumenti di MATLAB) contiene un grafo ad albero i cui nodi rappresentano tutte le cartelle e i file di MATLAB. Se vicino ad un nodo c'é un $+$, significa che esso contiene cartelle e file che non sono visualizzati. Se si fa clic sul $+$ vengono visualizzate le cartelle ed i file contenuti, e il $+$ viene trasformato in - (facendo clic sul meno si torna alla situazione precedente).
- *Workspace* mostra i nomi e i valori di tutte le variabili utilizzate nella sessione di lavoro corrente.

Help

Per avere la documentazione di MATLAB c'é la guida interna.
Per accedere alla guida é sufficiente fare clic sull'icona con il punto interrogativo, o selezionare l'opzione *Help* dal menu *View*.
Sullo schermo apparirá il browser illustrato nella figura.



Quando nella finestra dei comandi compare il prompt di MATLAB ($>>$), il programma é pronto a ricevere nuove istruzioni.
Se il cursore non si trova dopo il prompt, si puó utilizzare il mouse per spostarlo.
Quando invece sono in corso operazioni il prompt scompare.
Per annullare una operazione, premere contemporaneamente i tasti `Ctrl` e `c`.
Digitando `1/700` dopo il prompt e premendo *Invio* si ottiene

```
ans =
    0.0014
```

MATLAB assegna la risposta ad una variabile temporanea chiamata ans.
Per default il risultato viene visualizzato con 4 cifre decimali.
Il comando `format` permette di modificare il formato di uscita secondo la seguente tabella:

Comando	**Descrizione**
`format short`	4 cifre decimali
`format long`	14 cifre decimali
`format short e`	4 cifre decimali piú l'esponente
`format long e`	15 cifre decimali piú l'esponente
`format rat`	approssimazione razionale

Table: Formati numerici

Formati

Se digitiamo

```
>> format long
>> 1/700
```

otteniamo

```
ans =
    0.00142857142857
```

In modo analogo

```
>> format short e
>> 1/700
```

restituisce

```
ans =
    1.4286e-003
```

Formati

```
>> format long e
>> 1/700
ans =
    1.428571428571429e-003
```

Mentre,

```
>> format rat
>> 1/700
```

visualizza

```
ans =
    1/700
```

Esempio

MATLAB puó essere usato come calcolatrice come descritto nella seguente tabella:

Simbolo	Operazione	Formato di MATLAB
^	elevazione a potenza	a^b
*	Moltiplicazione	a*b
/	Divisione	a/b
+	Addizione	a+b
-	Sottrazione	a-b

Table: Operazione aritmetiche

Esempio

Se si commette un errore durante la digitazione , si ritorna ai comandi digitati in precedenza con la freccia in alto ($\uparrow$) e si utilizza la freccia in basso ($\downarrow$) per far scorrere al contrario la lista dei comandi.

Per spostare il cursore a sinistra o a destra all'interno della riga corrente é sufficiente premere i tasti ($\leftarrow$) o ($\rightarrow$)

Quando si trova il comando in cui si é commesso un errore, si modifica utilizzando il tasto *Canc* per cancellare il carattere che si trova davanti al cursore o quello *Backspace* per cancellare il carattere che si trova dietro il cursore.

Il punto e virgola posto alla fine di un comando indica a MATLAB di non visualizzare i risultati dell'istruzione sullo schermo.

Variabili e Costanti

I programmi in genere registrano dati in memoria.

Le zone di memoria in cui vengono registrati i dati si chiamano *variabili*.
Per assegnare il valore ad una variabile MATLAB usa il segno uguale (=).

Per esempio il comando $x = 2$, permette di registrare nella variabile x il valore 2.

Per registrare nella variabile x il risultato dell'addizione tra 3 e il contenuto della variabile y si scrive:

```
>> x=y+3
```

Esempio

Il comando $x = 5$ é diverso dal comando $5 = x$ che genera un messaggio di errore.

Il nome di una variabile deve iniziare con una lettera, che puó essere seguita da una qualunque combinazione di lettere e cifre, ma non piú lungo di 32 caratteri.

MATLAB distingue tra caratteri maiuscoli e minuscoli: `A` e `a` identificano variabili diverse.

Per conoscere il valore corrente di una variabile é sufficiente digitare il suo nome e premere *Invio*.

Esempio

Per sapere se la variabile x é giá stata definita, basta digitare

```
>>exist('x')
```

Se MATLAB restituisce il valore 1, la variabile esiste; se restituisce il valore 0, la variabile non esiste.
I nomi ed i valori di tutte le variabili utilizzate si trovano nel *Workspace*.
In alternativa,il comando

```
>> who
```

elenca i nomi di tutte le variabili, ma non indica i loro valori.

MATLAB conserva l'ultimo valore di una variabile finché la sessione di lavoro corrente é aperta o finché la variabile non é eliminata espressamente.

Esempio

Il comando

$$>> \texttt{clear}$$

rimuove tutte la variabili dall'area di lavoro. Il comando

$$>> \texttt{clear var1 var2 ...}$$

consente di eliminare le variabili var1 var2 ....
Prima di uscire da MATLAB é possibile salvare la sessione di lavoro con il comando

$$>> \texttt{save}$$

Questo comando salva tutte le variabili nel file matlab.mat.

Esempio

Se si desidera salvare la sessione di lavoro con un nome diverso si digita

```
>> save nomefile
```

Tutte le variabili saranno salvate nel file nomefile.mat. Il comando

```
>> load nomefile
```

ricarica tutte le variabili memorizzate in nomefile.mat mantenendo inalterato il nome con cui erano state memorizzate.
Il comando

```
>> clc
```

cancella il contenuto della finestra dei comandi, lasciando inalterate le variabili.

Vettori e matrici

Per creare un vettore riga basta digitare gli elementi all'interno di una coppia di parentesi quadre separandoli con uno spazio o una virgola.

Ad esempio, il comando per creare il vettore $a = (2, 4, 10)$ é:

$>>$ `a=[2 4 10]`

oppure

$>>$ `a=[2, 4, 10]`

Per creare un vettore colonna si possono digitare gli elementi separati dal punto e virgola.

Ad esempio, il vettore $b = \begin{pmatrix} 1 \\ 5 \\ 7 \end{pmatrix}$ é definito dal comando:

$>>$ `b=[1; 5; 7]`

Vettori

E' possibile creare un vettore riga o colonna “accodando” due vettori:

```
>> x=[2, 4, 20];
>> y=[9, 6, 3];
```

allora `z=[x, y]` da come risultato:

```
z=
    2     4     20     9     6     3
```

Per selezionare le componenti di indici $a, b, c, \ldots$ del vettore x si scrive `x([a, b, c, ...])`, ad esempio:

```
>> x=[4, 6, 1, 9, 3];
>> z=x([1, 2, 4])
z=
    4     6     9
```

Vettori

Per generare un vettore x con elementi intervallati regolarmente da a a b con incremento pari a q si scrive `x=[a:q:b]`:

```
>> x=[0:2:8]
x=
    0     2     4     6     8
```

Se viene omesso l'incremento q, allora MATLAB lo pone uguale a 1:

```
>> x=4:6
x=
    4     5     6
```

Sono permessi anche incrementi negativi, ad esempio:

```
>> x=7:-2:3
x=
    7     5     3
```

Esempio

Per avere un vettore x con m componenti intervallate regolarmente da a a b si usa il comando `x=linspace(a,b,m)`, ad esempio:

```
>> x=linspace(1,7,4)
x=
    1     3     5     7
```

Il comando `length(x)` fornisce il numero di componenti del vettore x.
La somma e la sottrazione di due vettori riga (o colonna) della stessa lunghezza si effettuano con il + ed il −, ad esempio:

```
>> x=[3, 4, 8, 1];
>> y=[-2, 9, 1, 0];
>> x+y
ans=
    1    13     9     1
>> x-y
ans=
    5    -5     7     1
```

Prodotto

Il prodotto tra uno scalare ed un vettore si effettua con il $*$:

```
>> x=[3, -4, 8, 1];
>> 2*x
ans=
    6     -8     16     2
```

Anche il prodotto scalare tra due vettori si effettua con il $*$:

```
>> x=[3, 4, 8, 1];
>> y=[-2; 9; 1; 0];
>> x*y
ans=
    38
```

Per creare una matrice si inseriscono gli elementi per riga separati da spazi o virgole e per passare alla riga successiva si usa il punto e virgola.

Per inserire la matrice $A = \begin{pmatrix} 1 & 2 & 3 & 6 \\ 3 & -5 & -8 & 12 \\ 2 & 0 & 1 & 9 \end{pmatrix}$, si digita:

$>>$ `A=[1, 2, 3, 6; 3, -5, -8, 12; 2, 0, 1, 9]`

Per creare una matrice ottenuta dalla matrice A aggiungendo il vettore colonna $b = \begin{pmatrix} 3 \\ -5 \\ 0 \end{pmatrix}$ si scrive `[A, b]`, mentre per aggiungere la riga $c = (6, 5, 8, 3)$ si digita `[A;c]`.

Matrici

Il comando `eye(n)` genera la matrice identitá di ordine n:

```
>> eye(3)
ans=
    1     0     0
    0     1     0
    0     0     1
```

Il comando `ones(m,n)` genera una matrice di ordine $m \times n$ i cui elementi sono tutti uguali a 1:

```
>> ones(3,2)
ans=
    1     1
    1     1
    1     1
```

`zeros(m,n)` genera una matrice nulla $m \times n$:

```
>> zeros(3,4)
ans=
    0     0     0     0
    0     0     0     0
    0     0     0     0
```

Il comando `diag` ha due modi di funzionamento: se l'argomento é una matrice quadrata, fornisce gli elementi sulla diagonale; se l'argomento é un vettore, genera una matrice diagonale i cui elementi diagonali sono gli elementi del vettore.

Matrici

Ad esempio:

```
>> diag([6, 4, 9; 5, 8, 2; 7, 5, 1])
ans=
    6
    8
    1
```

e

```
>> diag([3, 5, 1])
ans=
    3    0    0
    0    5    0
    0    0    1
```

Per sapere la dimensione di una matrice si usa il comando `size`:

```
>> A=[3, 5, 1; 6, 2, 8];
>> size(A)
ans=
    2    3
```

Matrici

La somma e la differenze tra matrici si effettuano rispettivamente con il segno + ed il segno − come tra due vettori:

```
>> A=[-7, 16; 4, 9];
>> B=[6, -5; 12, -2];
>> A+B
ans=
     -1  11
     16   7
```

Per fare il prodotto tra uno scalare ed una matrice basta usare il segno ∗, ad esempio:

```
>> A=[-7, 16; 4, 9];
>> 2*A
ans=
     -14  32
       8  18
```

Matrici

Il prodotto riga per colonna tra una matrice A di ordine $m \times n$ e d una matrice B di ordine $n \times p$ si effettua anch'esso con il segno $*$, ad esempio:

```
>> A=[-7, 16; 4, 9];
>> B=[6, -5; 12, -2];
>> A*B
ans=
     150     3
     132   -38
```

Operatori relazionali

MATLAB dispone di 6 operatori relazionali che consentono di confrontare variabili e vettori:

Simbolo	Descrizione
==	uguale a
~=	diverso da
<	minore di
<=	minore o uguale a
>	maggiore di
>=	maggiore o uguale a

Table: Operatori relazionali

Esempio

Il risultato di un confronto puó essere 0 (falso) oppure 1 (vero).
Tale risultato puó essere assegnato ad una variabile. Ad esempio, se $x = 2$ e $y = 3$, allora il comando `z=x<y` assegna alla variabile z il risultato del confronto $x < y$. In questo caso si ottiene:

```
z=1
```

Gli operatori relazionali permettono anche di confrontare, elemento per elemento, due vettori aventi lo stesso numero di componenti. Ad esempio, supponiamo che:

```
>> x=[6, 3, 9, 11];
>> y=[14, 2, 9, 13];
```

il comando `z=x<y` da come risultato

```
z=
    1    0    0    1
```

mentre il comando `z=x<=y` da come risultato

```
z=
    1    0    1    1
```

La funzione `linprog`

Nell'*Optimization toolbox* di MATLAB, la funzione `linprog` risolve un problema di PL della forma:

$$\begin{cases} \min\ c^{\mathsf{T}} x \\ A\,x \leq b \\ D\,x = e \\ l \leq x \leq u \end{cases} \tag{1}$$

dove c, x, b, e, l, u sono vettori e A, D sono matrici. Se, ad esempio, non ci sono vincoli di uguaglianza, si pongono `D=[ ]` ed `e=[ ]`.

Esempio

La sintassi della funzione é la seguente:

```
[x, v] = linprog(c, A, b, D, e, l, u)
```

dove gli input

```
c, A, b, D, e, l, u
```

definiscono il problema da risolvere, mentre gli output sono:

- `x` É una soluzione ottima del problema (1);
- `v` É il valore ottimo del problema (1).

Esempio

Supponiamo di dover risolvere il problema di PL:

$$\begin{cases} \max \; x_1 + 2\,x_2 + 3\,x_3 \\ 3\,x_1 + 4\,x_3 \leq 5 \\ 5\,x_1 + x_2 + 6\,x_3 = 7 \\ 8\,x_1 + 9\,x_3 \geq 2 \\ 0 \leq x_1 \leq 5 \\ 0 \leq x_2 \leq 4 \\ x_3 \geq 0 \end{cases} \tag{2}$$

Lo trasformiamo nella forma (1):

$$\begin{cases} -\min \; -x_1 - 2\,x_2 - 3\,x_3 \\ 3\,x_1 + 4\,x_3 \leq 5 \\ -8\,x_1 - 9\,x_3 \leq -2 \\ 5\,x_1 + x_2 + 6\,x_3 = 7 \\ 0 \leq x_1 \leq 5 \\ 0 \leq x_2 \leq 4 \\ 0 \leq x_3 \leq +\infty \end{cases}$$

Esempio

e scriviamo:

```
>> c = [-1; -2; -3];
>> A = [3, 0, 4; -8, 0, -9];
>> b = [5; -2];
>> D = [5, 1, 6];
>> e = 7;
>> l = [0;0;0];
>> u = [5;4;inf];

>> [x, v] = linprog(c, A, b, D, e, l, u)
```

si ottengono:

```
x=
    0.0000
    4.0000
    0.5000

v=
    -9.5000
```

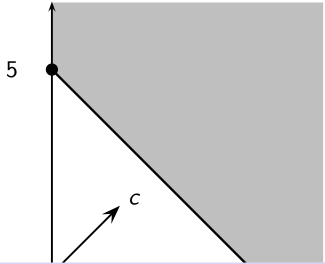
Esempio

Per risolvere problemi di PL la funzione `linprog` utilizza, di *default*, un metodo a punti interni invece del simplesso.

Quindi potrebbe non fornire come soluzione ottima un vertice. Ad esempio il problema

$$\begin{cases} \min\ x_1 + x_2 \\ x_1 + x_2 \geq 5 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases} \tag{3}$$

ammette infinite soluzioni ottime (il segmento di estremi $(0, 5)$ e $(5, 0)$).



Esempio

Se poniamo

```
>> c = [1; 1];
>> A = [-1, -1];
>> b = -5;
>> l = [0;0];
```

il comando

```
>> x = linprog(c, A, b, [ ], [ ], l, [ ])
```

fornisce la soluzione ottima

```
x=
    2.5000
    2.5000
```

che non é un vertice del poliedro del problema (3).

Opzione

Per risolvere il problema con l'algoritmo del simplesso é necessario cambiare le opzioni della funzione `linprog` con il seguente comando:

```
>> options = optimoptions('linprog','Algorithm','dual-simplex');
```

Ora il comando

```
>> x = linprog(c, A, b, [ ], [ ], l, [ ], [ ], options)
```

fornisce la soluzione ottima

```
x=
    5
    0
```

che é un vertice del problema (3).

Una fabbrica di detersivi produce due tipi di saponi che passano attraverso 4 fasi di lavorazione: le ore necessarie per ogni fase di lavorazione per quintale di prodotto sono riportate nella tabella che segue, in cui compaiono anche le ore mensili a disposizione per ciascuna fase.

	Fase a	Fase b	Fase c	Fase d
Sapone A	1.5	1.5	3	2.5
Sapone B	2.5	2	3	4
Ore mensili disponibili	155	200	240	400

Il guadagno netto é di 2100 euro per quintale di sapone A e 3400 euro per quintale di sapone B. Quanti quintali dei saponi A e B bisogna produrre per massimizzare il guadagno?

Esempio

Se indichiamo con x_1 il numero di quintali prodotti del sapone A e con x_2 quelli del sapone B, il problema si puó formulare come segue:

$$\begin{cases} \max\ 2100\ x_1 + 3400\ x_2 \\ 1.5\ x_1 + 2.5\ x_2 \leq 155 \\ 1.5\ x_1 + 2\ x_2 \leq 200 \\ 3\ x_1 + 3\ x_2 \leq 240 \\ 2.5\ x_1 + 4\ x_2 \leq 400 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

Per risolverlo con la funzione `linprog` dobbiamo trasformarlo:

$$\begin{cases} -\min\ -2100\ x_1 - 3400\ x_2 \\ 1.5\ x_1 + 2.5\ x_2 \leq 155 \\ 1.5\ x_1 + 2\ x_2 \leq 200 \\ 3\ x_1 + 3\ x_2 \leq 240 \\ 2.5\ x_1 + 4\ x_2 \leq 400 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

Esempio

COMANDI DI MATLAB

funzione obiettivo	`c=[-2100;-3400]`

vincoli	`A=[1.5, 2.5; 1.5, 2; 3, 3; 2.5, 4]`
	`b=[155; 200; 240; 400]`
	`lb=[0; 0]`

Comando risolutivo	`[x,fval]=linprog(c,A,b,[ ],[ ],lb,[ ])`

SOLUZIONI

Soluzione ottima	(45, 35)
Valore ottimo	213500

ANALISI DI SENSIBILITÀ

Supponendo di portare a 168 le ore mensili a disposizione per la fase a, determinare la nuova strategia di produzione ottima, il nuovo guadagno ed il costo per ogni ora lavorativa aggiuntiva per la fase a affinché la nuova strategia sia conveniente rispetto a quella vecchia.

Nuova soluzione ottima	(32,48)
Nuovo valore ottimo	230400
Costo per ogni ora aggiuntiva	< 1300

Esempio

Un'azienda deve produrre almeno 500 litri di un cocktail utilizzando tre tipi di succhi di frutta S_1, S_2 e S_3. La disponibilitá ed il costo dei diversi tipi di succhi di frutta sono indicati nella seguente tabella:

Tipo di succo	Disponibilitá max in litri	Costo in euro per litro
S_1	560	1.5
S_2	260	1
S_3	950	4

La dose di miscelatura per il cocktail é: non piú del 25 % di S_1, non meno del 50 % di S_2 e non meno del 40 % di S_3.

Si vuole determinare la combinazione dei tre tipi di succhi di frutta che minimizzi la spesa.

Esempio

Indicando con x_1 il numero di litri del succo S_1, con x_2 quelli di S_2 e con x_3 quelli di S_3, si ha il seguente modello:

$$\begin{cases} \min \ 1.5\ x_1 + x_2 + 4\ x_3 \\ x_1 \leq 0.25(x_1 + x_2 + x_3) \\ x_2 \geq 0.5(x_1 + x_2 + x_3) \\ x_3 \geq 0.4(x_1 + x_2 + x_3) \\ x_1 + x_2 + x_3 \geq 500 \\ 0 \leq x_1 \leq 560 \\ 0 \leq x_2 \leq 260 \\ 0 \leq x_3 \leq 950 \end{cases}$$

COMANDI DI MATLAB

funzione obiettivo	`c=[1.5; 1; 4]`

vincoli	`A=[3, -1, -1; 1, -1, 1; 2, 2, -3; -1, -1, -1]`
	`b=[0; 0; 0; -500]`
	`lb=[0; 0; 0]`
	`ub=[560; 260; 950]`

Comando risolutivo	`[x,fval]=linprog(c,A,b,[ ],[ ],lb,ub)`

SOLUZIONI

Soluzione ottima	(40, 260, 200)
Valore ottimo	1120

ANALISI DI SENSIBILITÀ

Supponendo che la massima disponibilitá del secondo succo di frutta aumenti di 40 litri, diventando quindi di 300 litri, determinare la nuova soluzione ottima e la nuova spesa minima.

Nuova soluzione ottima	(0,300,200)
Nuova spesa minima	1100

Esempio

Un'impresa produce un bene in 2 stabilimenti, situati a Pontedera e a Rosignano. La produzione viene immagazzinata in 2 depositi a Pisa e a Livorno e poi distribuita alla rete di vendita al dettaglio.
I dati riguardano il costo unitario di trasporto, la capacitá produttiva massima settimanale dei 2 stabilimenti e le statistiche di vendita settimanale di ognuno dei 2 depositi.

	Pisa	Livorno	Capacitá produttiva massima
Pontedera	10	30	105
Rosignano	35	6	80
Vendita	110	46	

Esempio

Indichiamo con x_1 la quantitá di merce spedita da Pontedera a Pisa, con x_2 quella spedita da Pontedera a Livorno, con x_3 quella da Rosignano a Pisa e con x_4 quella da Rosignano a Livorno. Il modello é:

$$\begin{cases} \min\ 10\ x_1 + 30\ x_2 + 35\ x_3 + 6\ x_4 \\ x_1 + x_2 \leq 105 \\ x_3 + x_4 \leq 80 \\ x_1 + x_3 = 110 \\ x_2 + x_4 = 46 \\ x \geq 0 \end{cases}$$

Esempio

COMANDI DI MATLAB

funzione obiettivo	`c=[10; 30; 35; 6]`

vincoli	`A=[1, 1, 0, 0; 0, 0, 1, 1]`
	`b=[105; 80]`
	`Aeq=[1, 0, 1, 0; 0, 1, 0, 1]`
	`beq=[110; 46]`
	`lb=[0; 0; 0; 0]`

Comando risolutivo	`[x,fval]=linprog(c,A,b,Aeq,beq,lb,[ ])`

Esempio

SOLUZIONI

Soluzione ottima	(105, 0, 5, 46)
Valore ottimo	1501

ANALISI DI SENSIBILITÀ

Supponendo che la capacitá produttiva dello stabilimento di Pontedera diventi 110, determinare la nuova soluzione ottima e la nuova spesa minima.

Nuova soluzione ottima	(110, 0, 0, 46)
Nuova spesa minima	1376

Esempio

Un'industria siderurgica ha tre stabilimenti che necessitano di 50, 70 e 60 tonnellate di acciaio a settimana.
L'acciaio puó essere acquistato da due fornitori. Il primo puó fornire al massimo 30 tonnellate a settimana a ciascun stabilimento, mentre il secondo puó fornire al massimo 40 tonnellate a settimana a ciascun stabilimento.
Il primo fornitore non puó fornire piú di 100 tonnellate a settimana e deve fornire non meno di 25 tonnellate a settimana al terzo stabilimento. La seguente tabella indica i costi unitari di trasporto (euro/ton) dai fornitori agli stabilimenti.

Fornitori	Stabilimenti		
	1	2	3
1	2	3	5
2	3	3.6	3.2

Determinare come si deve rifornire l'industria per minimizzare il costo di trasporto.

Esempio

Indichiamo con x_1, x_2, x_3 le tonnellate di acciaio spedite rispettivamente dal primo fornitore ai tre stabilimenti e con x_4, x_5, x_6 le tonnellate di acciaio spedite rispettivamente dal secondo fornitore ai tre stabilimenti. Il modello é:

$$\begin{cases} \min\ 2\ x_1 + 3\ x_2 + 5\ x_3 + 3\ x_4 + 3.6\ x_5 + 3.2\ x_6 \\ x_1 + x_4 = 50 \\ x_2 + x_5 = 70 \\ x_3 + x_6 = 60 \\ x_1 + x_2 + x_3 \leq 100 \\ 0 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 30 \\ 25 \leq x_3 \leq 30 \\ 0 \leq x_4 \leq 40 \\ 0 \leq x_5 \leq 40 \\ 0 \leq x_6 \leq 40 \end{cases}$$

COMANDI DI MATLAB

funzione obiettivo	c=[2; 3; 5; 3; 3.6; 3.2]
vincoli	A=[1, 1, 1, 0, 0, 0] b=100 Aeq=[1, 0, 0, 1, 0, 0; 0, 1, 0, 0, 1, 0; 0, 0, 1, 0, beq=[50; 70; 60] lb=[0; 0; 25; 0; 0; 0] ub=[30; 30; 30; 40; 40; 40]
Comando risolutivo	[x,fval]=linprog(c,A,b,Aeq,beq,lb,ub)

SOLUZIONI

Soluzione ottima	(30, 30, 25, 20, 40, 35)
Valore ottimo	591

ANALISI DI SENSIBILITÀ

Supponendo che il primo fornitore possa spedire al massimo 35 tonnellate a settimana ai tre stabilimenti, determinare la nuova soluzione ottima e la nuova spesa minima.

Nuova soluzione ottima	(35, 35, 25, 15, 35, 35)
Nuova spesa minima	583

Esempio

Uno stabilimento produce tre diversi tipi di pitture per l'edilizia: una economica, una normale ed una di extra qualitá. Ogni pittura viene lavorata da tre linee di produzione A, B e C. I tempi (in minuti) necessari alla lavorazione di ogni quintale, la disponibilitá delle linee di produzione ed i profitti dei tre tipi di pittura sono indicate in tabella:

	Economica	Normale	Extra	Disponibilitá
A	20	30	62	480
B	31	42	51	480
C	16	81	10	300
Profitto	100	150	220	

La quantitá della pittura extra deve essere non piú del 20% del totale, mentre quella economica deve essere non meno del 40% del totale. Determinare le quantitá dei tre diversi tipi di pittura in modo da massimizzare il profitto.

Esempio

Sia con x_1 la quantitá prodotta di pittura economica, con x_2 quella di pittura normale e con x_3 quella di pittura extra. Il modello di PL é il seguente:

$$\begin{cases} \max\ 100\ x_1 + 150\ x_2 + 220\ x_3 \\ 20\ x_1 + 30\ x_2 + 62\ x_3 \leq 480 \\ 31\ x_1 + 42\ x_2 + 51\ x_3 \leq 480 \\ 16\ x_1 + 81\ x_2 + 10\ x_3 \leq 300 \\ x_3 \leq 0.2(x_1 + x_2 + x_3) \\ x_1 \geq 0.4(x_1 + x_2 + x_3) \\ x \geq 0 \end{cases}$$

che equivale a:

$$\begin{cases} -\min\ -100\ x_1 - 150\ x_2 - 220\ x_3 \\ 20\ x_1 + 30\ x_2 + 62\ x_3 \leq 480 \\ 31\ x_1 + 42\ x_2 + 51\ x_3 \leq 480 \\ 16\ x_1 + 81\ x_2 + 10\ x_3 \leq 300 \\ -0.2\ x_1 - 0.2\ x_2 + 0.8\ x_3 \leq 0 \\ -0.6\ x_1 + 0.4\ x_2 + 0.4\ x_3 \leq 0 \\ x \geq 0 \end{cases}$$

COMANDI DI MATLAB

funzione obiettivo	`c=[-100; -150; -220]`

vincoli	`A=[20,30,62;31,42,51;16,81,10; -0.2,-0.2,0.8;-0.6,0.4,0.4]`
	`b=[480; 480; 300; 0; 0]`
	`lb=[0; 0; 0]`

Comando risolutivo	`[x,fval]=linprog(c,A,b,[ ],[ ],lb,[ ])`

SOLUZIONI

Soluzione ottima	(8.95, 1.60, 2.64)
Valore ottimo	1718

ANALISI DI SENSIBILITÀ

Supponendo che i minuti a disposizione della linea di produzione C diventino 360, determinare la nuova soluzione ottima ed il nuovo profitto ottimo.

Nuova soluzione ottima	(7.71, 2.60, 2.57)
Nuovo profitto ottimo	1729