Dipartimento di Impresa e Management

Cattedra di Matematica Finanziaria

Il Metodo del Simplesso in Python: un agile strumento di Programmazione Lineare per la risoluzione di problemi complessi

RELATORE:
Prof. Gennaro Olivieri

CANDIDATO:
Simone Maria Caranci

MATRICOLA:
223261

Anno Accademico 2019/2020

Indice

Introduzione

L'economia globale contemporanea ruota intorno ad un elemento fondamentale: i dati.
La loro importanza ha assunto un ruolo centrale nelle scelte strategiche delle imprese, che siano grandi player multinazionali o piccole società, tanto da rappresentare un valore economico a sé stante, di primaria importanza.
Semmai, la novità che caratterizza questa fase è data dalla enorme quantità di informazioni disponibili, il cui "governo" richiede metodologie di trattamento che permettano di sfruttarle adeguatamente al fine di ottimizzare le risorse e massimizzare i profitti.
Molti sono i software che utilizzano algoritmi, più o meno complessi, per tali fini.
Lo scopo di questa tesi è di analizzare uno degli algoritmi più potenti mai creati nell'ambito dall'ottimizzazione matematica e di studiarne il funzionamento, attraverso diverse applicazioni in molteplici campi dell'economia e non solo: il Metodo del Simplesso.

La motivazione che mi ha spinto a trattare di tale tematica è duplice: da un lato, vi è l'intenzione di approfondire uno degli aspetti più caratterizzanti dell'economia e della finanza al giorno d'oggi che, come sopra indicato, è il trattamento e l'elaborazione dei dati; dall'altro la volontà di combinare una mia passione, la programmazione, con un argomento tanto attuale quanto preponderante in una delle materie che più mi hanno affascinato nel percorso di questa laurea triennale: la Matematica Finanziaria.

L'obiettivo è dunque quello di trovare un connubio tra queste due sfere analizzando i vari passaggi dell'algoritmo del Simplesso e scrivendo, nel linguaggio di programmazione Python, un programma che riesca, attraverso il metodo, a risolvere problemi di Programmazione Lineare.

La tesi è articolata in quattro capitoli: nel primo viene fornita una panoramica sulla Programmazione Matematica attraverso un rapido *excursus* sulla storia di questa branca e la presentazione, anche attraverso esempi numerici, di una sua sotto-categoria: la Programmazione Lineare; nel secondo viene esposto e discusso il Metodo del Simplesso nelle sue tre forme (Standard, Tabellare e Matriciale); nel terzo capitolo si analizzano due mezzi informatici per l'utilizzo dell'algoritmo: il Risolutore di Excel e un programma in Python (scritto da me, attraverso l'utilizzo della libreria Pulp); infine, il quarto capitolo tratta di un'applicazione pratica del metodo in una particolare categoria di problemi, ovvero quelli di Flusso a Costo Minimo.

1. La Programmazione Matematica

La Programmazione Matematica (o ottimizzazione) è una branca della matematica applicata che studia le teorie e i metodi per la ricerca di punti di massimo e minimo di una funzione matematica[1]. Essa è un sottoinsieme della Ricerca Operativa, anche nota come Teoria delle Decisioni o Operational Research. Una particolare tipologia di problemi affrontati in PM è rappresentata dalla Programmazione Lineare. Tali problemi possono essere risolti mediante un metodo numerico conosciuto come Metodo del Simplesso (o algoritmo del Simplesso).

1.1. Storia della Programmazione Matematica

Il termine Programmazione Matematica fu introdotto per la prima volta nel 1949 da Robert Dorfman per indicare problemi il cui scopo è trovare i valori ottimali di una funzione obiettivo avente le variabili indipendenti soggette a dei vincoli[2].

In realtà, lo studio di questa parte della matematica ha posto le proprie radici più indietro nel tempo: nel XVIII secolo il matematico svizzero Leonhard Euler (1707-1783), conosciuto nella nostra lingua col nome di Eulero, con il suo "*Methodus inveniendi lineas curvas maximi minimive proprietate gaudentes, sive solutio problematis isoperimetrici latissimo sensu accepti*" affermò che: "*Qualunque sia il paradigma umano che si manifesta, di solito riflette il comportamento di massimizzazione o minimizzazione. Pertanto, non vi è alcun dubbio sul fatto che i fenomeni naturali possano essere spiegati mediante i metodi di massimizzazione o minimizzazione*"[3].

Eulero, introducendo la *variational analysis*, ha definito per la prima volta le condizioni di ottimalità dei modelli di programmazione matematica.

Il suo lavoro fu poi ripreso dall'italiano Joseph Louis Lagrange (1736-1813) che giunse all'equazione conosciuta come equazione di Eulero-Lagrange, la quale, semplificando la condizione di ottimalità di Eulero, rappresenta una delle condizioni necessarie per l'esistenza di una soluzione ottima.

Nell'800 Jean Baptiste Joseph de Fourier (1768-1830), successore di Lagrange alla cattedra di analisi e meccanica all'Ecole Polytechnique di Parigi, continuò gli studi del suo predecessore.

Tra la seconda metà dell'800 e i primi del '900 Julius Farkas (1847-1930) studiò i teoremi delle alternative e introdusse un metodo per la risoluzione di sistemi di diseguaglianze lineari che, circa quaranta anni dopo, fu fondamentale per lo sviluppo della Teoria della Dualità in Programmazione Lineare.

Un matematico russo, Pafnuti Lvovich Chebyshev (1821-1894), apportò un importante contributo allo studio della programmazione matematica introducendo la cosiddetta approssimazione di Chebyshev. Egli formulò un polinomio capace di risolvere problemi semi-infiniti di minimizzazione convessa non regolare (non lineare).

In maniera analoga a Eulero egli ha affermato che: "*In tutte le attività umane pratiche troviamo lo stesso problema: Come allocare le nostre risorse in modo tale da ottenere il massimo profitto possibile?*"[4].

Due suoi studenti, Markov e Lyaounov, diedero ulteriore impulso ai suoi studi.

Andrey Andreyevich Markov (1856–1922), oltre ad essere famoso per il suo processo stocastico (conosciuto, appunto, come processo stocastico di Markov), ha studiato i cosiddetti *moment problems*, finalizzati ad ottimizzare i momenti di una funzione di distribuzione o di variabili stocastiche.

Aleksandr Mikhailovich Lyapunov (1857–1918) formulò, invece, un metodo per studiare la convergenza e la stabilità dei metodi numerici in ottimizzazione.

Il francese Gaspard Monge (1746-1818) e l'olandese Charles-Jean de La Vallée Poussin (1866-1962), considerati i "pionieri" della Programmazione/ottimizzazione Lineare, pubblicarono studi che hanno posto le basi della PL. Molti matematici del '900, come Dantzig, Koopmans e Kantorovich, prenderanno da loro ispirazione per lo sviluppo dei propri lavori.

Monge fu il matematico che formulò il "Metodo della doppia proiezione ortogonale"[5] apportando un grandissimo contributo alla creazione della "geometria descrittiva". Alla sua attività è dovuto anche il lavoro sul trasporto di massa continuo, che ha costituito un fondamentale contributo iniziale al problema del trasporto lineare (caso particolare di PL).

Vallée Poussin proseguì gli studi di Chebyshev: con il suo articolo *Sure la methode de l'approximation minimum*, pubblicato sull' *Annales de la Societe Scientifique de Bruxelles*, è considerato uno dei fondatori dell'ottimizzazione lineare.

Il primo però, a risolvere un problema di PL fu Leonid Vitalevich Kantorovich (1912–1986); egli si occupò della modellizzazione matematica della produzione in alcune aziende del legno e sviluppò un metodo che in seguito fu riconosciuto come equivalente al metodo del simplesso per i problemi duali.

Nel 1939 pubblicò *A Mathematical Methods for Production Organization and Planning*, *Leningrad* e nel 1969 *Economical Calculation of the Best Utilization of Resources* nel quale si possono trovare due appendici sulla teoria matematica della PL e i suoi metodi numerici. I metodi analitici di Kantorovich nell'ottimizzazione sono stati utilizzati per lo sviluppo della "teoria dell'analisi convessa".

Gli scritti di Kantorovich furono tradotti, approfonditi e pubblicati dal matematico olandese Tjalling Charles Koopmans (1910–1985).

I due, nel 1975, furono insigniti del Premio Nobel per l'economia. Nel corso della cerimonia Koopmans tenne la sua *Memorial Lecture* dal titolo *Concepts of optimality and their uses*[6].

Fondamentale, ancora, il lavoro di George Bernard Dantzig (1914-2005), matematico statunitense noto per aver sviluppato e pubblicato nel 1947, con l'articolo intitolato *Programming in a Linear Structure,* un metodo numerico per risolvere problemi di programmazione lineare talmente potente da essere citato dalla rivista statunitense *Computing in Science and Engineering* come uno dei dieci migliori algoritmi del secolo: il Metodo (o Algoritmo) del Simplesso[7].

Due anni dopo la pubblicazione dell'articolo, e precisamente nel 1949, Koopmans organizzò a Chicago un importante convegno sulla PL i cui atti furono pubblicati nel 1951 sotto il nome di *Activity analysis of production and allocation*[8].

Fu la prima conferenza sulla programmazione lineare e fu considerata la "conferenza-zero" di una lunga serie di convegni sulla Programmazione Matematica che arriva fino ad oggi.

Proprio durante questo congresso Robert Dorfman coniò il termine "Programmazione Matematica"[9]

1.2. *La Programmazione Lineare*

1.2.1. Introduzione

Negli ultimi cinquant'anni i problemi di PL sono stati intensamente studiati e sono così divenuti uno strumento fondamentale in campi di studio quali l'economia e la finanza[10].

Un problema generale di Programmazione Lineare (PL) ha come scopo l'allocazione ottimale di risorse soggette a vincoli, al fine di soddisfare una data funzione obiettivo.

La funzione obiettivo e i suoi vincoli sono espressi sotto forma di funzioni lineari, ovvero funzioni definite da un polinomio di primo grado, rappresentabili come rette sul piano cartesiano e che devono rispettare le seguenti condizioni:

$$f(\alpha \mathbf{x}) = \alpha f(\mathbf{x});$$

$$f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y}).$$

Potremo dunque scrivere un problema generale di massimo condizionato con vincoli di diseguaglianza:

$$\begin{cases} \max f(\mathbf{x}) \\ g(\mathbf{x}) \leq 0 \\ \mathbf{x} \geq 0 \end{cases}$$

dove la funzione obiettivo si presenta come:

$$f(\mathbf{x}) = c_1x_1 + c_2x_2 + \dots + c_nx_n = \sum_{j=1}^{n} c_jx_j$$

nel quale le x_j sono le incognite del problema e le c_j i termini noti;
i vincoli rappresentano le restrizioni al problema di ottimizzazione. Si dividono in vincoli strutturali e vincoli di non negatività: i primi limitano l'utilizzo delle risorse; i secondi impongono che le soluzioni ottime devono essere positive. Possono essere espressi sotto forma di disequazioni o equazioni:

$$g(\mathbf{x}) \begin{cases} a_{11}x_1 + a_{12}x_2 + \dots\dots a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots\dots a_{2n}x_n \le b_2 \\ a_{m1}x_1 + a_{m2}x_2 + \dots\dots a_{mn}x_n \le b_m \end{cases}$$
$$x_1 \ge 0,\ x_2 \ge \dots\ x_n \ge 0$$

dove a_{ij} rappresenta la quantità di risorse utilizzate da ogni attività x_j e b_i la quantità disponibile di ogni risorsa.
Il problema così presentato viene detto espresso in forma *canonica*.
Se $f(\mathbf{x})$ è una funzione definita nel dominio **A** , si ha

$$\max_{\mathbf{x}\in\mathbf{A}} f(x) = -\min_{\mathbf{x}\in\mathbf{A}}[-f(x)]$$

dunque, senza perdita di generalità, si considerano solo problemi LP di massimo.
Sapendo, inoltre, che:

$$\sum_{j=1}^{n} a_{ij}x_j \le b_i \qquad i = 1,2, \dots, m$$

è equivalente a

$$-\sum_{j=1}^{n} a_{ij}x_j \ge -b_i \qquad i = 1,2, \dots, m$$

e che un vincolo di diseguaglianza può diventare un vincolo di uguaglianza aggiungendo una variabile detta variabile di scarto (o variabile slack*)* x_s non negativa, dunque:

$$\sum_{j=1}^{n} a_{ij}x_j \ \le \ b_i \qquad i = 1,2, \dots , m$$

è equivalente a

$$\sum_{j=1}^{n} a_{ij}x_j + x_s \ = \ b_i \qquad i = 1,2, \dots , m$$

possiamo senza perdita di generalità, riscrivere il problema dalla forma canonica nella cosiddetta forma *standard:*

$$\begin{cases} \max \ \sum_{j=1}^{n} c_j x_j \\ \sum_{j=1}^{n} a_{ij}x_j \ = \ b_i \qquad i = 1,2, \dots , m \\ x_j \geq 0 \quad j = 1, 2, \dots , n \end{cases}$$

che vedremo essere utilizzata nel metodo del Simplesso.
Qualunque sia la forma in cui è presentato un problema di PL questo si fonda sulle seguenti ipotesi:

– *Proporzionalità*
– *Additività*
– *Continuità*

dove per *proporzionalità* si intende che il contributo di una variabile di decisione alla funzione obiettivo e ai vincoli è proporzionale secondo una costante moltiplicativa alla quantità rappresentata dalla variabile stessa; per *additività* si intende che il contributo delle variabili di decisione alla funzione obiettivo e ai vincoli è dato dalla somma dei contributi di ogni singola variabile; infine, per *continuità* si intende che ogni variabile di decisione può assumere tutti i valori reali nell'intervallo di ammissibilità, e quindi le variabili possono assumere valori frazionari.
I modelli di Programmazione Lineare hanno attratto l'attenzione di molti studiosi poiché caratterizzati dalla presenza di molti vantaggi, quali:

- *generalità e flessibilità:* i modelli di PL possono descrivere un numero elevatissimo di situazioni reali, in un vastissimo numero di campi (anche eterogenei) e la loro generalizzazione costituisce un'ottima base anche per problemi che non rispettano l'ipotesi di linearità;
- *semplicità:* essendo espressi attraverso il linguaggio dell'algebra lineare sono di facile comprensione;
- *efficienza degli algoritmi risolutivi:* esistono molteplici algoritmi e metodi di risoluzione capaci di risolvere problemi aventi migliaia di vincoli e centinaia di migliaia di variabili;

- *possibilità di analisi quantitative:* i modelli di PL consentono di ottenere delle informazioni aggiuntive sulla dipendenza tra la soluzione e gli eventuali parametri presenti, che possono avere significative interpretazioni economiche[11].

1.2.2. Soluzioni ammissibili e soluzioni ottime

L'insieme dei vettori $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_n \end{bmatrix} \in \mathbb{R}^n$ che soddisfano i vincoli del problema è detta Regione Ammissibile (R.A.) . Un vettore $\mathbf{x} \in$ R.A. è detto soluzione ammissibile. Qualora un vettore

$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_n \end{bmatrix} \in \mathbb{R}^n$ non rispettasse anche solo un vincolo questo è detto soluzione inammissibile.

Una soluzione ammissibile $\mathbf{x^*}$ tale che

$$f(\mathbf{x^*}) \geq f(\mathbf{x}) \qquad \forall\, \mathbf{x} \in \text{R.A.}$$

è detta soluzione ottima.

È possibile però che il problema non ammetta alcuna soluzione ottima. In tal caso ci si potrebbe trovare davanti a una delle seguenti situazioni:

- il problema è inammissibile: vi è un'incompatibilità tra i vincoli strutturali e quelli di non negatività, la R.A. risulta vuota e dunque non sono presenti soluzioni ottime;
- il problema è illimitato: la R.A. è superiormente o inferiormente illimitata, presentando quindi infinite soluzioni ammissibili che non consentono di trovare una soluzione ottima.

1.2.3 Geometria della Programmazione Lineare

Analizzando dal punto di vista geometrico un problema di PL e la sua R.A., ogni equazione individua un iperpiano e ogni disequazione individua un semispazio chiuso.

La R.A. deriva dall'intersezione degli iperpiani e dei semispazi chiusi.

Definizione 1 (Poliedro). Un insieme $P \subseteq \mathbb{R}^n$ è un poliedro se è ottenuto dall'intersezione di un numero finito di semispazi chiusi e iperpiani in $\mathbb{R}^n$.

Dunque una soluzione ammissibile è un punto nello spazio *n*-dimensionale e la R.A. è un poliedro nello stesso spazio (Figura 1).

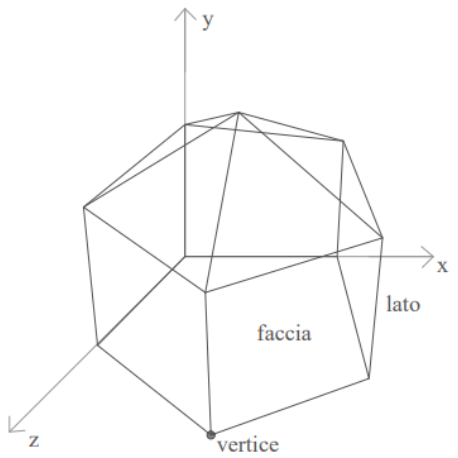


Figura 1: *Poliedro in* $\mathbb{R}^3$

Un poliedro può essere limitato o illimitato. Per la risoluzione di un problema di PL è necessario che sia limitato. Si dice limitato un insieme $S \subset \mathbb{R}^n$ se esiste una costante M tale che il valore assoluto di ogni componente di x, per ogni $x \in S$, è minore o uguale a M. Se il poliedro è limitato è detto politopo (Figura 2).

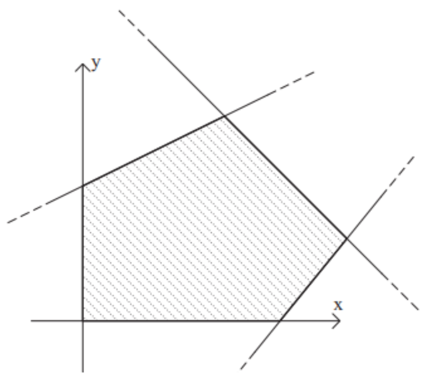


Figura 2: *Poliedro limitato in* $\mathbb{R}^2$

Oltre ad essere limitato un poliedro di un problema di PL deve essere convesso. Un insieme $S \subset \mathbb{R}^n$ è convesso se, per ogni coppia di vettori $x, y \in S$, e per ogni $\lambda \in [0,1]$, la combinazione lineare $\lambda x + (1 - \lambda)y \in S$, il che comporta che tutti i punti appartenenti al segmento che congiunge x e y sono contenuti nell'insieme S (Figura 3).

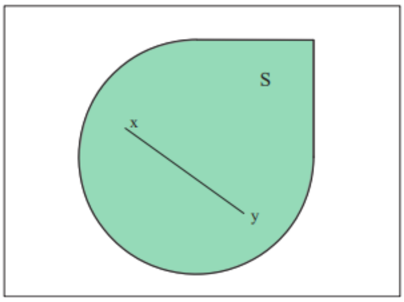


Figura 3: *Figura convessa*

Definito il poliedro, che deve quindi essere un politopo convesso, bisogna definire i vertici, i quali hanno enorme importanza per la risoluzione dei problemi di PL.

Definizione 2 (Vertice di un poliedro). Dato un poliedro P e un punto del poliedro v ∈ P, v è vertice di P se non può essere espresso come combinazione convessa stretta di due punti distinti dello stesso poliedro: $\nexists\, x, y \in P, \lambda \in (0, 1) : x \neq y, v = \lambda x + (1 - \lambda)y$.

Essendo chiuso il poliedro è composto da un numero finito di vertici.
Sulla base di queste premesse e date le condizioni possiamo, dunque, enunciare il teorema di Minkowski-Weyl:

Teorema 1 (Rappresentazione dei poliedri - caso limitato): Dato un poliedro limitato $P \subset \mathbb{R}^n$, e indicando con $v^1, v^2, \dots, v^k$ ($v^i \in \mathbb{R}^n$) i vertici di P, si ha $x \in P \Rightarrow x = \sum_{i=1}^{k} \lambda_i v^i$ con $\lambda_i \geq 0 \ \forall i = 1,\dots,k$ e $\sum_{i=1}^{k} \lambda_i = 1$. In altre parole, ogni punto di P si può ottenere come combinazione convessa dei suoi vertici.

il quale ci consente di enunciare il Teorema Fondamentale della Programmazione Lineare:

Teorema 2: (Teorema fondamentale della PL): Si consideri il problema di Programmazione Lineare.

Supponiamo che il poliedro P = $\{x \in \mathbb{R}^n \mid Ax \geq b\}$ non contenga rette. Allora una e una sola delle seguenti tre affermazioni è vera:

1. Il problema è inammissibile, ovvero il poliedro P è vuoto;
2. Il problema è illimitato inferiormente;
3. Il problema ammette soluzioni ottime e almeno una di queste soluzioni è un vertice del poliedro P[11].

1.2.4 Problemi di Programmazione Lineare in forma matriciale. Soluzioni di Base

È possibile formulare un problema di PL in forma matriciale considerando i vettori $\mathbf{c}, \mathbf{x} \in \mathbb{R}^n$, il vettore $\mathbf{b} \in \mathbb{R}^m$ e la matrice $\mathbf{A}(m,n)$ con $m < n$ e rango r($\mathbf{A}$) = m così da poter scrivere:

$$\begin{cases} \max \mathbf{c}\,\mathbf{x} \\ \mathbf{A}\mathbf{x} = \mathbf{b} \\ \mathbf{x} \geq 0 \end{cases}$$

con $\mathbf{Ax} = \mathbf{b}$ non ridondante.

Sia a_j la generica j-ma colonna di A; se, riordinando le colonne, i vettori $a_1, a_2, \dots, a_m$ sono linearmente indipendenti, essi costituiscono una base di $\mathbb{R}^m$e il sistema Ax = b si può riscrivere

$$\mathbf{x}_h = d_h + \sum_{k=1}^{n-m} d_{hk} + \boldsymbol{x}_{m+k} \qquad h = 1,2, \dots, m.$$

Ponendo

$$\mathbf{x}_{m+1} = \mathbf{x}_{m+2} = \dots = \mathbf{x}_{m+n} = 0$$

il sistema ha una unica soluzione:

$$(d_1, d_2, \dots, d_m, 0, 0, \dots, 0)$$

la quale, se $d_h \geq 0 \;\; \forall\, h$, è una soluzione ammissibile ed è detta soluzione di base ammissibile (SBA) dove le prime m componenti non negative sono dette variabili di base, e le successive n-m componenti nulle sono dette variabili fuori base.

Possiamo, pertanto, fare le seguenti osservazioni:

- una SBA ha almeno *n-m* variabili nulle e al più *m* variabili positive;
- una combinazione lineare convessa non banale di due o più SBA è una soluzione ammissibile non di base;
- se una o più variabili di base è nulla la SBA è detta degenere;
- una soluzione ammissibile con *m* variabili positive e *n-m* variabili nulle non è necessariamente una SBA.

Definita una SBA possiamo analizzare nello specifico la sua importanza all'interno di un problema di PL al fine di trovare una sua soluzione ottima.

Presupponendo ancora che la matrice A abbia dimensione (m, n), con $m < n$ e $r(A) = m$

Teorema 3 dato un problema di PL in forma standard

$$\begin{cases} \max \mathbf{c}\,\mathbf{x} \\ \mathbf{A}\mathbf{x} = \mathbf{b} \\ \mathbf{x} \geq 0 \end{cases}$$

se esiste una soluzione ammissibile, esiste una SBA.

Infatti, detta $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ una soluzione ammissibile, supponiamo che, riordinando le variabili, sia

$$\mathbf{x}_1 > 0;\ \mathbf{x}_2 > 0;\ \dots;\ \mathbf{x}_p > 0;\ \mathbf{x}_{p+1} = 0;\ \dots;\ \mathbf{x}_n = 0$$

e quindi

$$a_1\mathbf{x}_1 + a_2\mathbf{x}_2 + \dots + a_p\mathbf{x}_p = b;$$

a) se $a_1, \dots, a_p$, sono vettori linearmente indipendenti allora $p \leq m$;
   se $p = m$ la soluzione è una base;
   se, invece, $p < m$, dato che $r(A) = m$, è possibile scegliere $m - p$ colonne di A , fra $[a_{p+1}, a_{p+2} \dots, a_n]$ in modo da costituire, con $[a_1, \dots, a_p]$, un insieme di m colonne linearmente indipendenti. Assegnando il valore zero alle corrispondenti variabili, si ottiene una SBA degenere.
b) Se, invece, $[a_1, \dots, a_p]$ sono vettori linearmente dipendenti, è possibile esprimere uno di essi, poniamo a_p, come combinazione lineare di $a_1, a_2, \dots, a_{p-1}$. Di conseguenza, si ricava una nuova soluzione ammissibile $\bar{x} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$ tale che

$$a_1\bar{x}_1 + a_2\bar{x}_2 + \dots + a_{p-1}\bar{x}_{p-1} = b; \qquad \overline{x_1} = \overline{x_2} = \dots = \overline{x_n} = 0$$

   Applicando iterativamente la procedura si ottiene un insieme di vettori linearmente indipendenti, cui è applicabile quanto indicato in a).

Si ricordi che, dato un insieme convesso $S \subset \mathbb{R}^n$, un punto estremo di S è un punto di S che non può essere espresso come combinazione lineare convessa di altri punti di S.
Dato il problema LP in forma standard, si suppone che, come nel caso del Teorema 3 , sia A una matrice m, n con $m < n$ e $r(A) = m$.

Vale il seguente

Teorema 4: Un vettore **x** è un punto estremo della R.A. se e solo se **x** è una SBA

Infatti

a) se $\mathbf{x} = (\mathbf{x}_1 , \dots , \mathbf{x}_m, 0, 0, \dots , 0)$, è una SBA, sarà uguale ad $a_1\mathbf{x}_1 + \dots + a_m\mathbf{x}_m = b$, con $a_1, a_2, \dots , a_m$ vettori linearmente indipendenti.
Supponendo, per assurdo, che **x** sia esprimibile come combinazione lineare convessa di due altre soluzioni ammissibili y e z :

$$\mathbf{x} = \lambda y + (1 - \lambda)z, \ (0 < \lambda < 1), \ y \neq z$$

dato che $y_i \geq 0$ e $z_i \geq 0$ da $\mathbf{x}_{m+h}$ segue

$$y_{m+h} = 0 \text{ e } z_{m+h} = 0, \ (h = 1 , 2, \dots , n-m);$$

perciò

$$y_1a_1 + y_2a_2 + \dots + y_ma_m = b \text{ e}$$

$$z_1a_1 + z_2a_2 + \dots + z_ma_m = b.$$

Dato che $(a_1, \dots , a_m)$ sono linearmente indipendenti e che $r(A) = m$ si ha, in conclusione, che $y = z = \mathbf{x}$.

b) se **x** è un punto estremo della R.A., siano $\mathbf{x}_1, \mathbf{x}_2 , \dots , \mathbf{x}_k$ le componenti positive e $\mathbf{x}_{k+1}, \mathbf{x}_{k+2}, \dots , \mathbf{x}_n$ le componenti nulle di **x** . Supponendo, per assurdo, che **x** non sia una base e che perciò $a_1 , \dots , a_k$ siano linearmente dipendenti, esiste un insieme di coefficienti, non tutti nulli, $\{y_1, \dots , y_k\}$ tale che

$$y_1a_1 + \dots + y_ka_k = 0$$

Dato il vettore $y = (y_1, \dots , y_k, 0, \dots , 0)$, esiste $\varepsilon > 0$ tale che

$$(\mathbf{x} + \varepsilon y) \text{ e } (\mathbf{x} - \varepsilon y)$$

sono soluzioni ammissibili e che

$$\mathbf{x} = 1/2(\mathbf{x} + \varepsilon y) + 1/2 (\mathbf{x} - \varepsilon y).$$

È quindi possibile esprimere **x** come combinazione lineare convessa di due soluzioni ammissibili, il che contrasta con l'ipotesi che **x** sia un punto estremo della R.A. .

Pertanto a_1 , ... , a_k sono linearmente indipendenti e **x** è una SBA[10].

Considerando, quindi, per quest'ultimo teorema che un punto **x** è un estremo della R.A. se e solo se è una SBA, e che per il Teorema fondamentale della PL almeno una soluzione ottima si trova su un vertice del poliedro che iscrive la R.A. stessa possiamo affermare la seguente

Proprietà Fondamentale: l'insieme delle soluzioni ottime, se non vuoto, contiene almeno una SBA.

Analizziamo ora due metodi di risoluzione dei problemi di PL dimostrando che una soluzione ottima coincide con un vertice del poliedro convesso rappresentante la R.A.: il Metodo Grafico e il Metodo del Simplesso.

1.2.5 Metodo grafico per la risoluzione di problemi in Programmazione Lineare

È possibile risolvere i problemi di PL attraverso il metodo grafico, vale a dire attraverso l'iscrizione dei vincoli in forma di rette su un piano cartesiano, l'individuazione della R.A. e, per il Teorema Fondamentale della PL, la ricerca delle soluzioni ottime sui vertici del politopo.
Questo metodo risulta molto efficace e di facile applicazione se circoscritto a casi che presentano due variabili, all'aumentare delle quali la sua efficacia diminuisce. In tal caso diventa più complesso riuscire a costruire il modello e individuarne i vertici.
Vediamone un esempio:
La SughiPronti S.r.l. è un'azienda artigianale che lavora nel campo della produzione e vendita di salse e sughi preconfezionati, ed ha deciso di aggiungere al catalogo due nuovi prodotti: un pesto alla genovese e un pesto rosso.
L'azienda vuole calcolare quale sia la quantità e la combinazione dei barattoli di pesto da produrre al fine di massimizzare il proprio profitto sapendo che:

- Gli ingredienti per singolo barattolo (80g) delle due salse sono elencati nella tabella che segue:

INGREDIENTI	**PESTO alla GENOVESE (grammi)**	**PESTO ROSSO (grammi)**
Basilico	38	3,9
Olio	29	29
Pinoli	7,4	6,8
Parmigiano Reggiano	4,6	4
Aglio	1	0,3
Concentrato	0	36

- La disponibilità giornaliera degli ingredienti è illustrata nella tabella che segue:

INGREDIENTI	KG
Basilico	280
Olio	380
Pinoli	55
Parmigiano Reggiano	55
Aglio	8
Concentrato	250

- I due prodotti sono venduti rispettivamente a €5 e €7

Per calcolare la combinazione più profittevole dei due prodotti si formula il modello matematico di PL per il problema, sapendo che:

Z = profitto totale;
x_1 = quantità barattoli di pesto alla genovese;
x_2 = quantità barattoli di pesto rosso;

La funzione obiettivo sarà quindi: $Z = 5x_1 + 7x_2$

Inserendo i vincoli si avrà:

$$\begin{cases} Z = 5x_1 + 7x_2 \\ 38x_1 + 3{,}9x_2 \leq 280.000 \\ 29x_1 + 29x_2 \leq 380.000. \\ 7{,}4x_1 + 6{,}8x_2 \leq 55.000 \\ 4{,}6x_1 + 4x_2 \leq 55.000 \\ 1x_1 + 0{,}3x_2 \leq 8.000 \\ 36x_2 \leq 250.000 \\ x_1 \geq 0 \text{ e } x_2 \geq 0 \end{cases}$$

È possibile riscrivere i vincoli sotto forma di equazioni implicite di rette del tipo $ax_1 + bx_2 = c$ e tracciarle su un piano cartesiano (Figura 4):

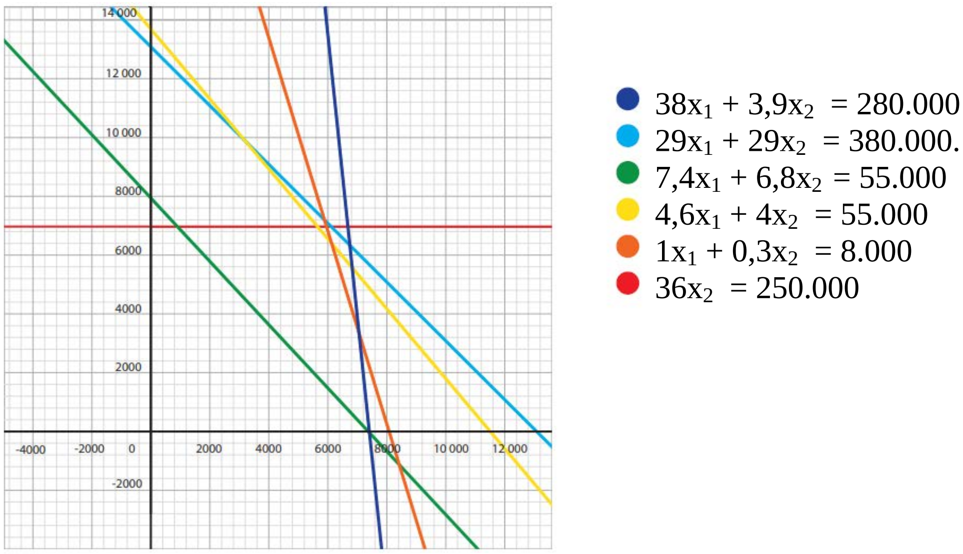


Figura 4: *Vincoli in forma di retta*

Avendo posto i vincoli di non negatività, consideriamo le aree sottostanti alle rette, circoscritte solamente al primo quadrante (Figura 5).

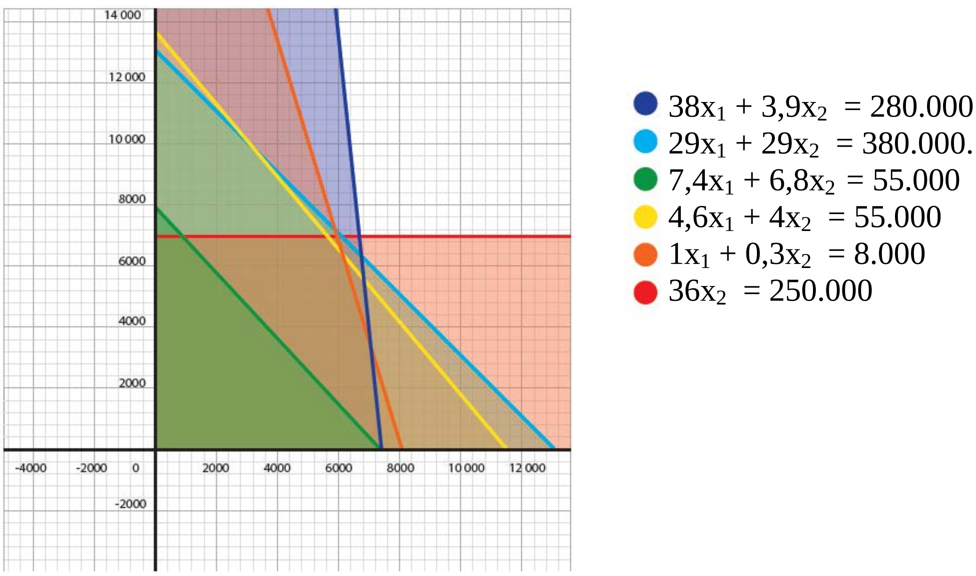


Figura 5: *Aree sottostanti alle rette dei vincoli*

In tal modo possiamo individuare la R.A., ovvero il politopo convesso (nel nostro caso bidimensionale) che contiene tutte le soluzioni ammissibili, tra cui la soluzione ottima, avente come lati le rette e come vertici le loro intersezioni(Figura 6).

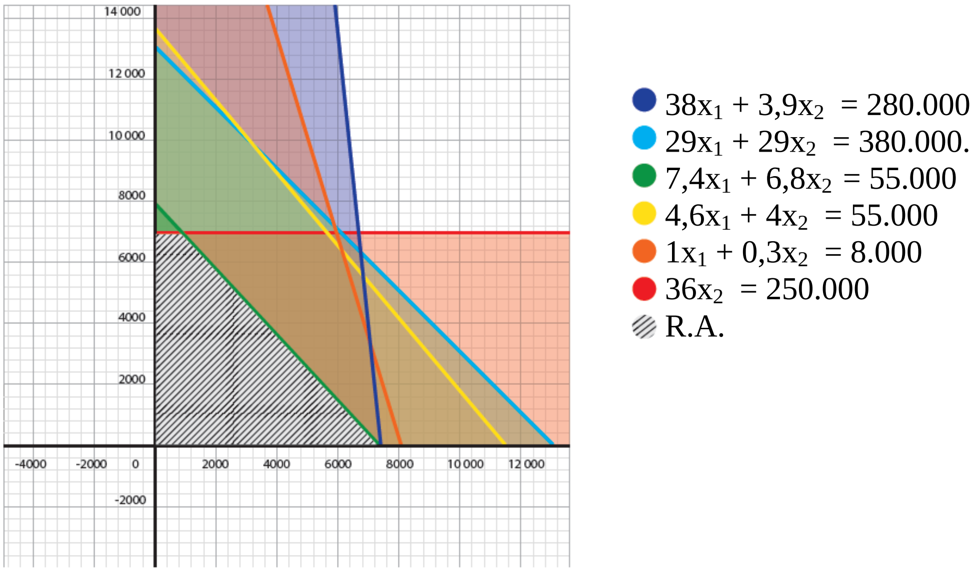


Figura 6: *Regione Ammissibile*

Infine tracciamo le rette parallele alla retta passante per l'origine ed avente pendenza $-\frac{5}{7}$ (che ricaviamo dall'equazione della funzione obiettivo) e passanti per ogni vertice del politopo (Figura 7).

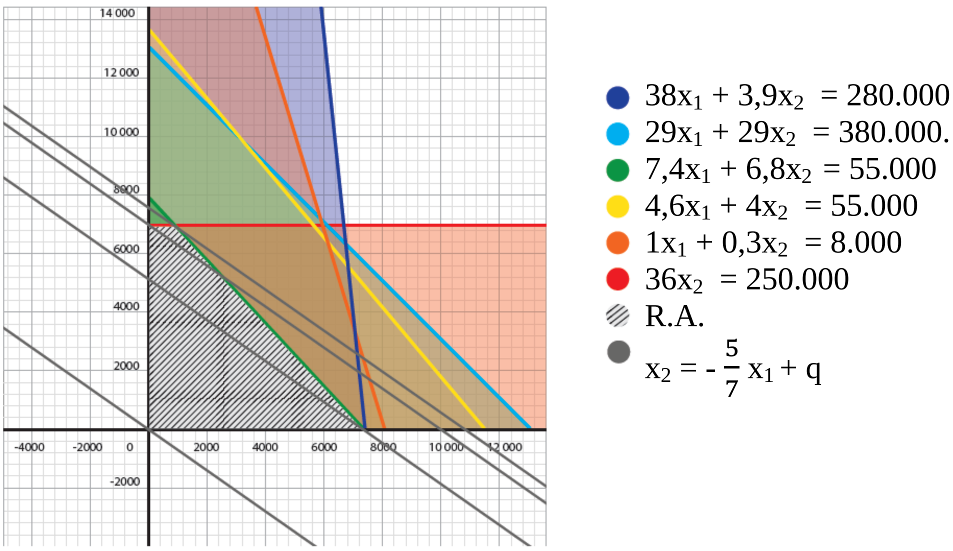


Figura 7: *Ricerca della soluzione ottima*

Per il Teorema Fondamentale della PL, almeno una soluzione ottima del problema coincide con un vertice del poliedro. Si può allora concludere che tale soluzione coincide con il vertice nel quale passa la retta con termine noto più elevato tra quelle parallele alla retta passante per l'origine e derivante dalla funzione obiettivo, che nel nostro caso è il punto (1051 ; 6944).

Dunque la SughiPronti s.r.l. massimizzerà il proprio profitto producendo 1051 barattoli di pesto alla genovese e 6944 di pesto rosso.

Come detto in precedenza questo metodo è molto utile e di facile applicazione in problemi a due variabili. Quando il numero di incognite inizia a crescere diventa sempre più difficile applicarlo ed è più ragionevole utilizzare il Metodo del Simplesso.

2. Il Metodo del Simplesso

In questo capitolo viene trattato l'algoritmo del Simplesso, metodo matematico iterativo per la risoluzione dei problemi di Programmazione Lineare sviluppato nel 1947 da George Dantzig.

2.1. L'algoritmo del Simplesso

Per poter essere risolto con il Metodo del Simplesso il problema di PL deve essere presentato in forma *standard*, dove, come abbiamo visto nel precedente capitolo:

- la funzione obiettivo deve essere massimizzata;
- i vincoli strutturali sono espressi in forma di equazioni grazie all'aggiunta di variabili *slack*;
- le variabili sono soggette a vincoli di non negatività.

Il numero delle variabili *slack* coincide con il numero dei vincoli di disuguaglianza *m*.
Quindi, scritto in questa forma, il problema presenta $n+m$ variabili.
Prendendo ad esempio il seguente problema di PL

$$\max Z = 3x_1 + 5x_2$$

$$\begin{aligned} x_1 + x_2 &\leq 12 \\ x_1 &\leq 10 \\ x_2 &\leq 6 \\ x_1 + 2x_2 &\leq 16 \end{aligned}$$

possiamo riscriverlo in forma standard:

$$\max Z = 3x_1 + 5x_2$$

$$\begin{aligned} x_1 + x_2 + x_3 &= 12 \\ x_1 + x_4 &= 10 \\ x_2 + x_5 &= 6 \\ x_1 + 2x_2 + x_6 &= 16 \end{aligned}$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0$$

Come visto precedentemente, è importante ricordare la differenza tra soluzioni ammissibili, non ammissibili e ottime, dove le prime fanno parte della R.A., le seconde sono fuori da essa e le terze massimizzano la funzione obiettivo e sono situate sul perimetro della R.A., più precisamente (per il Teorema Fondamentale della PL) sui vertici, così come il concetto di soluzione di base ammissibile (SBA), ovvero quella soluzione di base avente almeno $n - m$ elementi nulli e al più m elementi positivi e, per il **Teorema 4**, che i punti estremi della R.A. sono delle SBA.
Gli elementi caratterizzanti di questo metodo sono:

- la capacità di selezionare in maniera efficiente i vertici che visita;
- il fatto di passare da un vertice ad un altro senza richiedere inversioni di matrici o soluzioni di sistemi di equazioni;
- l'uso di semplici criteri che permettono di individuare il vertice ottimo o di concludere che il problema di Programmazione Lineare non ammette soluzioni in quanto è illimitato.

Il metodo del Simplesso è composto da 2 fasi:

- Nella **Fase I** viene controllata l'ammissibilità del problema da risolvere; vengono individuati ed eliminati i vincoli di uguaglianza linearmente dipendenti dagli altri fino a ottenere un sistema di vincoli di uguaglianza descritto da una matrice a rango massimo e viene identificata una SBA.
- Nella **Fase II** viene risolto il problema di programmazione lineare.
  Tale risultato è ottenuto partendo dalla SBA calcolata nella Fase I ed eseguendo i seguenti passaggi:
  - si controlla se la SBA soddisfa il criterio di ottimalità;
  - si controlla se il problema soddisfa il criterio di illimitatezza;
  - se nessuno dei due criteri è soddisfatto, viene determinata una nuova SBA[11].

Iterazione 1

Tornando all'esempio, avendo aggiunto le variabili *slack*, il sistema dei vincoli è esprimibile come una matrice con m = 4 e n = 6, dunque la SBA avrà 4 elementi in base e 2 fuori base, ai quali attribuiremo il valore 0. Partiamo dalle variabili x_1 e x_2 come variabili fuori base. I vincoli saranno:

$$x_3 = 12$$

$$x_4 = 10$$

$$x_5 = 6$$

$$x_6 = 16$$

Quindi la SBA è (0, 0, 12, 10, 6, 16).
In questo caso la funzione obiettivo sarà:

$$Z = 3x_1 + 5x_2 = 0$$

in quanto non compaiono nella funzione le variabili di base. Poiché i coefficienti delle variabili non di base indicano i tassi di miglioramento della funzione obiettivo (per ogni unità di x_1 Z aumenta di 3 e per ogni unità di x_2 di 5) risulterà opportuno valutare l'ipotesi di far entrare x_1 o x_2 in base. Volendo far entrare x_1 come variabile di base, è necessario determinarne il valore e decidere quale variabile far uscire dalla base.
Si dovrà, inoltre, rispettare i vincoli di non negatività, evitando che le variabili di base assumano valori ≤ 0.
Se $x_2 = 0$, allora:

$$x_3 = 12 - x_1 \geq 0 \quad \Rightarrow \quad x_1 \leq 12$$
$$x_4 = 10 - x_1 \geq 0 \quad \Rightarrow \quad x_1 \leq 10 \Rightarrow \text{Valore minimo}$$
$$x_5 = 6 \Rightarrow \text{Non dipende da } x_1$$
$$x_6 = 16 - x_1 \geq 0 \quad \Rightarrow \quad x_1 \leq 16$$

Utilizzando il cosiddetto test del minimo rapporto possiamo rilevare che x_4 è la variabile che, all'aumentare dell'entrante, raggiunge per prima il valore di 0. Sarà dunque x_4 la variabile uscente.

Iterazione 2
Possiamo, quindi, riscrivere il problema facendo entrare x_1 in base nella prima riga e facendo uscire x_4. Per farlo, usiamo l'equazione $x_1 + x_4 = 10$ per esplicitare x_1 in funzione di x_2 e x_4 (che sono le variabili della nuova non-base) e lo sostituiamo nelle equazioni[12].
Otterremo:

$$x_1 + x_4 = 10$$
$$x_2 + x_3 - x_4 = 2$$
$$x_2 + x_5 = 6$$
$$2x_2 + x_6 - x_4 = 6$$

$$Z = 3 * (10 - x_4) + 5x_2 = 30 - 3x_4 + 5x_2$$

La nuova SBA sarà dunque (10, 0, 2, 0, 6, 6).

La funzione obiettivo assume il valore di 30. Tuttavia la funzione può essere migliorata in quanto il coefficiente di x_2 è pari a 5, il che significa che ad ogni unità aggiuntiva di questa variabile, la funzione obiettivo aumenterà di 5 unità. Mentre, all'aumentare di un'unità di x_4, avendo la variabile un coefficiente negativo, Z diminuirebbe e quindi non converrà aumentarla.

Sarà pertanto opportuno cercare una nuova SBA, adiacente a questa, facendo entrare x_2 in base attraverso il test del minimo rapporto.

Supponendo $x_4 = 0$, allora:

$$x_3 = 2 - x_2 \geq 0 \quad \Rightarrow \quad x_2 \leq 2 \Rightarrow \text{ Valore minimo}$$
$$x_1 = 10 \Rightarrow \text{Non dipende da } x_2$$
$$x_5 = 6 - x_2 \geq 0 \quad \Rightarrow \quad x_2 \leq 6$$
$$x_6 = 6 - 2x_2 \geq 0 \quad \Rightarrow \quad x_2 \leq 3$$

all'aumentare dell'entrante x_3 sarà il primo a raggiungere 0 e sarà quindi la variabile uscente.

Iterazione 3

Riscriviamo, allora, il problema facendo entrare x_2 in base nella prima riga e facendo uscire x_3. Per farlo, usiamo le equazioni $x_1 + x_2 + x_3 = 12$ e $x_1 + x_4 = 10$ per esplicitare x_2 in funzione di x_3 e x_4 (che sono le variabili della nuova non-base) e lo sostituiamo nelle equazioni, ottenendo:

$$x_2 + x_3 - x_4 = 2$$
$$x_1 + x_4 = 10$$
$$- x_3 + x_4 + x_5 = 4$$
$$- 2x_3 + x_4 + x_6 = 2$$

$$Z = 3 * (10 - x_4) + 5 * (2 - x_3 + x_4) = 30 - 3x_4 + 10 - 5x_3 + 5x_4 = 40 - 5x_3 + 2x_4$$

La nuova SBA sarà dunque (10, 2, 0, 0, 4, 2).

La funzione obiettivo assume il valore di 40.

Tuttavia, la funzione può ulteriormente essere migliorata, in quanto il coefficiente di x_4 è pari a 2, il che significa che ad ogni unità aggiuntiva di questa variabile, la funzione obiettivo aumenterà di 2 unità. Mentre, all'aumentare di un'unità di x_3, avendo la variabile un coefficiente negativo, Z diminuirebbe: pertanto non converrà aumentarla.

Posto ciò, sarà opportuno cercare una nuova SBA, adiacente ad questa, facendo entrare x_4 in base attraverso il test del minimo rapporto.

Supponendo $x_3 = 0$, allora:

$$x_1 = 10 - x_4 \geq 0 \quad \Rightarrow \quad x_4 \leq 10$$
$$x_2 = 2 + x_4 \geq 0 \quad \Rightarrow \quad x_4 \geq -2$$
$$x_5 = 4 - x_4 \geq 0 \quad \Rightarrow \quad x_4 \leq 4$$
$$x_6 = 2 - x_4 \geq 0 \quad \Rightarrow \quad x_4 \leq 2 \Rightarrow \text{ Valore minimo}$$

Dal test del minimo rapporto si evidenzia come sia x_6 la variabile che, all'aumentare dell'entrante, raggiunge per prima il valore di 0; conseguentemente essa sarà la variabile uscente.

Iterazione 4

Riscrivendo il problema facendo entrare x_4 e uscire x_6 e esplicitando le equazioni per le variabili fuori base, avremo:

$$x_1 = 8 - 2x_3 + x_6$$
$$x_2 = 4 + x_3 - x_6$$
$$x_4 = 2 + 2x_3 - x_6$$
$$x_5 = 2 - x_3 + x_6$$

$$Z = 3 * (8 - 2x_3 + x_6) + 5 * (4 + x_3 - x_6) = 24 - 6x_3 + 3x_6 + 20 + 5x_3 - 5x_6 = 44 - x_3 - 2x_6$$

La nuova SBA sarà (8, 4, 0, 2, 2, 0).

La funzione obiettivo assume il valore di 44. Considerando che i coefficienti delle due variabili x_3 e x_6 sono entrambi negativi, la funzione obiettivo non risulterà ulteriormente migliorabile, il che ci porta a concludere che la SBA individuata sia soluzione ottima al problema.

2.2. Il Metodo del Simplesso in forma tabellare

Un altro modo per risolvere i problemi di PL con il metodo del Simplesso è l'utilizzo della forma tabellare. Per la creazione del *tableau* sono sufficienti come informazioni le variabili, i coefficienti e i termini noti del problema.

Con riferimento al problema del precedente paragrafo, il *tableau* che si ottiene sarà:

Iterazione	**Variabili di base**	**Coefficienti**							**Termini noti**
		Z	**x_1**	**x_2**	**x_3**	**x_4**	**x_5**	**x_6**	
1	Z	1	-3	-5	0	0	0	0	0
	x_3	0	1	1	1	0	0	0	12
	x_4	0	1	0	0	1	0	0	10
	x_5	0	0	1	0	0	1	0	6
	x_6	0	1	2	0	0	0	1	16

Nella seconda colonna si inseriscono le variabili che consideriamo in base, quindi non compaiono x_1 e x_2, ovvero le variabili fuori base alle quali assegniamo valore 0.

I coefficienti di Z hanno segno negativo in quanto l'equazione va riscritta e inserita come

$$Z - 3x_1 - 5x_2 = 0.$$

Nel *tableau* sono state già inserite le variabili *slack* e quindi per la prima iterazione la SBA è (0, 0, 12, 10, 6, 16) e la funzione obiettivo da come risultato 0, non essendo presente in essa nessuna variabile di base. Con la forma tabellare si giunge alla conclusione che si è trovata la soluzione ottima quando nella riga di Z non compaiono coefficienti negativi.

In questo caso sia x_1 che x_2 hanno segno negativo; si dovrà, dunque, cercare una SBA adiacente a questa e verificarne l'ottimalità.

Si renderà necessario, quindi, inserire x_1 nella tabella è far uscire una variabile. Per far ciò dovremo individuare la variabile uscente con il test del minimo rapporto attraverso tre passaggi:

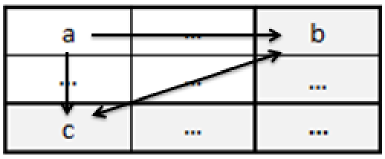
1. si individua la colonna *pivot*, ovvero la colonna della variabile entrante;
2. si calcola il rapporto tra i termini noti e i coefficienti positivi presenti nella colonna pivot;

3. si individua la variabile uscente, ovvero quella a cui corrisponde il rapporto più basso, nel nostro caso x_4. La riga che corrisponde alla variabile uscente è detta riga *pivot* e l'elemento che appartiene contemporaneamente alla riga e alla colonna pivot è detto coefficiente *pivot*, nel nostro caso pari a 1.

Iterazione	Variabili di base	Coefficienti							Termini noti	Rapporto
		Z	x_1	x_2	x_3	x_4	x_5	x_6		
1	Z	1	-3	-5	0	0	0	0	0	
	x_3	0	1	1	1	0	0	0	12	12
	x_4	0	**1**	0	0	1	0	0	10	**10**
	x_5	0	0	1	0	0	1	0	6	-
	x_6	0	1	2	0	0	0	1	16	16

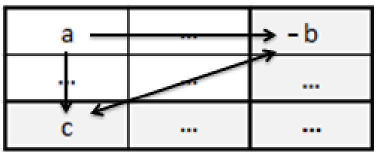
Si procede dunque con il calcolo della nuova SBA per una seconda iterazione, sostituendo la variabile uscente con quella entrante e costruendo il nuovo *tableau* attraverso le seguenti operazioni:

1. si divide la riga pivot per il coefficiente pivot. La riga ottenuta determinerà i valori attribuiti all'equazione della nuova variabile di base;
2. per le righe che presentano un coefficiente positivo nella colonna *pivot*, si sottrae al coefficiente il prodotto tra il coefficiente della sua riga appartenente alla colonna *pivot* e il coefficiente della sua colonna appartenente alla riga *pivot*.



a - b * c

3. per le righe che presentano un coefficiente negativo nella colonna *pivot*, si somma al coefficiente il prodotto tra il coefficiente della sua riga appartenente alla colonna *pivot* e il coefficiente della sua colonna appartenente alla riga *pivot*.



a + b * c

A questo punto il nuovo *tableau* sarà:

Iterazione	Variabili di base	Coefficienti							Termini noti
		Z	x_1	x_2	x_3	x_4	x_5	x_6	
2	Z	1	0	-5	0	3	0	0	30
	x_3	0	0	1	1	-1	0	0	2
	X_1	0	1	0	0	1	0	0	10
	x_5	0	0	1	0	0	1	0	6
	x_6	0	0	2	0	-1	0	1	6

Il risultato della funzione obiettivo è 30 ma la prima riga dei coefficienti ha ancora valori negativi. La SBA (10, 0, 2, 0, 6, 6) non è soluzione ottima.

Bisognerà quindi ripetere le operazioni per far entrare in base x_2. Scegliendo come colonna *pivot* x_2 e effettuando il test del minimo rapporto notiamo che la variabile uscente sarà x_3.

La tabella sarà dunque:

Iterazione	Variabili di base	Coefficienti							Termini noti	Rapporto
		Z	x_1	x_2	x_3	x_4	x_5	x_6		
2	Z	1	0	-5	0	3	0	0	30	
	x_3	0	0	**1**	1	-1	0	0	2	**2**
	X_1	0	1	0	0	1	0	0	10	-
	x_5	0	0	1	0	0	1	0	6	6
	x_6	0	0	2	0	-1	0	1	6	3

Effettuando le operazioni per la costruzione di una nuova SBA avremo:

Iterazione	Variabili di base	Coefficienti							Termini noti
		Z	**x_1**	**x_2**	**x_3**	**x_4**	**x_5**	**x_6**	
3	Z	1	0	0	5	-2	0	0	40
	X_2	0	0	1	1	-1	0	0	2
	X_1	0	1	0	0	1	0	0	10
	x_5	0	0	0	-1	1	1	0	4
	x_6	0	0	0	-2	1	0	1	2

La funzione obiettivo ha valore 40, ma è ancora presente un coefficiente negativo all'interno della riga Z. Dunque la SBA (10, 2, 0, 0, 4, 2) non è ancora una soluzione ottima. Sarà d'uopo un altro ciclo dell'algoritmo al fine di far entrare in base x_4 . Con il metodo del minimo rapporto ci accorgiamo che la variabile uscente è x_6.

Iterazione	Variabili di base	Coefficienti							Termini noti	Rapporto
		Z	**x_1**	**x_2**	**x_3**	**x_4**	**x_5**	**x_6**		
3	Z	1	0	0	5	-2	0	0	40	
	X_2	0	0	1	1	-1	0	0	2	-
	X_1	0	1	0	0	1	0	0	10	10
	x_5	0	0	0	-1	1	1	0	4	4
	x_6	0	0	0	-2	**1**	0	1	2	**2**

La nuova tabella sarà:

Iterazione	**Variabili di base**	**Coefficienti**							**Termini noti**
		Z	**x_1**	**x_2**	**x_3**	**x_4**	**x_5**	**x_6**	
4	Z	1	0	0	1	0	0	2	**44**
	X_2	0	0	1	-1	0	0	1	4
	X_1	0	1	0	2	0	0	-1	8
	x_5	0	0	0	1	0	1	-1	2
	X_4	0	0	0	-2	1	0	1	2

La funzione obiettivo ha valore 44. La riga Z ha tutti coefficienti positivi.

Possiamo finalmente affermare che la SBA (8, 4, 0, 2, 2, 0) è una soluzione ottima del problema.

2.3. Il Metodo del Simplesso in forma matriciale

Il metodo del simplesso in forma matriciale è considerato il metodo più efficiente per la risoluzioni dei problemi di PL attraverso l'algoritmo del simplesso, in quanto utilizza solo le informazioni essenziali del problema, riportate in forma matriciale. Un generico problema di PL in forma matriciale è definito come:

$$\max Z = c^T x$$

$$A^T x \leq b$$

$$x \geq 0$$

dove $c = [c_1 \;\; c_2 \ldots \;\; c_n]$ è il vettore riga dei coefficienti della funzione obiettivo; $x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$, $b = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_m \end{bmatrix}$ e $0 = \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}$ sono i vettori colonna rispettivamente delle variabili, dei termini noti e degli zeri dei vincoli di non negatività e $A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$ è la matrice *m*x*n* dei coefficienti dei vincoli strutturali.

Per passare dalla forma canonica alla forma *standard* si dovrà aggiungere il vettore colonna delle variabili *slack* $x_s = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \cdots \\ x_{n+m} \end{bmatrix}$ e alla matrice A la matrice identità I, che introduce i coefficienti delle variabili *slack*.

Dunque i vincoli strutturali saranno dati da $[A, I] \begin{bmatrix} x \\ x_s \end{bmatrix} = b$, dove I è la matrice identità *m*x*m*, e i vincoli di non negatività saranno $\begin{bmatrix} x \\ x_s \end{bmatrix} \geq 0$.

Riscrivendo il forma matriciale il problema in forma *standard* dell'esempio utilizzato negli scorsi paragrafi avremo:

$$c = [3 \;\; 5 \;\; 0 \;\; 0 \;\; 0 \;\; 0] \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} \qquad b = \begin{bmatrix} 12 \\ 10 \\ 6 \\ 16 \end{bmatrix} \qquad 0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Iterazione 1

Il primo passaggio per la risoluzione del problema è l'individuazione della prima SBA separando le *m* variabili di base dalle *n* variabili fuori base, alle quali attribuiremo il valore 0, in modo da avere il vettore colonna delle variabili in base $x_B = \begin{bmatrix} x_{B1} \\ x_{B2} \\ \dots \\ x_m \end{bmatrix}$.

Facendo in questo modo otterremo il sistema di equazioni $Bx_B = b$ dove B è la matrice di base ottenuta sottraendo a [A, I] le colonne dei coefficienti delle variabili fuori base.

Considerando come variabili di base $x_B = \begin{bmatrix} x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$ avremo $B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Il metodo del simplesso considera solo matrici non singolari **(Definizione)**, sarà dunque possibile ottenere la matrice inversa di $B \rightarrow B^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Bisogna ora moltiplicare B^{-1} per A e per b ottenendo

$$B^{-1}A = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1}b = \begin{bmatrix} 12 \\ 10 \\ 6 \\ 16 \end{bmatrix}.$$

Consideriamo $c_B = [0 \ \ 0 \ \ 0 \ \ 0]$ e lo moltiplichiamo sia per $B^{-1}A$ che per $B^{-1}b$

$$c_B\, B^{-1}A = [0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 0] \qquad c_B\, B^{-1}b = [0],$$

dove $c_B\, B^{-1}b$ è il risultato della funzione obiettivo;
infine sottraiamo c a $c_B\, B^{-1}A$, e avremo:

$$c_B\, B^{-1}A - c = [-3 \ \ -5 \ \ 0 \ \ 0 \ \ 0 \ \ 0].$$

Qualora all'interno di quest'ultimo vettore gli elementi fossero tutti ≥ 0 il problema sarebbe risolto e la SBA (0, 0, 12, 10, 6, 16) sarebbe soluzione ottima. Nel nostro caso sono presenti due elementi negativi. Dovremo, dunque, far entrare una nuova variabile in base.
Per far ciò sarà necessario applicare il test del minimo rapporto dividendo gli elementi di $B^{-1}b$ per i corrispondenti elementi della matrice $B^{-1}A$ appartenenti alla colonna della variabile entrante. Avendo come variabile entrante x_1 avremo : { x_3: $\frac{12}{1} = 12$, x_4: $\frac{10}{1} = 10$, x_5: $\frac{6}{0}$, x_6: $\frac{16}{1} = 16$ }. Sarà dunque x_4 la variabile

uscente.

Iterazione 2

La nuova base sarà formata dalle variabili x_3, x_1, x_5 e x_6

$$B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1} =. \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

È molto importante l'ordine delle colonne.

Moltiplicando B^{-1} per A e b avremo:

$$B^{-1}\mathrm{A} = \begin{bmatrix} 0 & 1 & 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & -1 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1}b = \begin{bmatrix} 2 \\ 10 \\ 6 \\ 6 \end{bmatrix}.$$

c_B sarà uguale a: $c_B = [0\ \ 3\ \ 0\ \ 0]$, che moltiplicato per B^{-1}A e $B^{-1}b$

$$c_B\, B^{-1}\mathrm{A} = [3\ \ 0\ \ 0\ \ 3\ \ 0\ \ 0] \quad \text{e} \quad c_B\, B^{-1}b = [30]$$

Infine:

$$c_B\, B^{-1}\mathrm{A} - c = [0\ \ -5\ \ 0\ \ 3\ \ 0\ \ 0].$$

Il risultato della funzione obiettivo è 30 ma $c_B\, B^{-1}\mathrm{A} - c$ contiene un elemento negativo; ne discende che la SBA (10, 0, 2, 0, 6, 6) non è soluzione ottima.

Sostituendo, quindi, x_2 con la variabile uscente che troviamo con il test del minimo rapporto { x_3: $\frac{2}{1} = 2$, x_1: $\frac{10}{0}$, x_5: $\frac{6}{1} = 6$, x_6: $\frac{6}{2} = 3$ } avremo che la variabile uscente è x_3.

Iterazione 3

La nuova base sarà formata dalle variabili x_2, x_1, x_5 e x_6

$$B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1} =. \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ -2 & 1 & 0 & 1 \end{bmatrix}$$

Moltiplicando B^{-1} per A e b avremo:

$$B^{-1}A = \begin{bmatrix} 0 & 1 & 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 1 & 0 \\ 0 & 0 & -2 & 1 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1}b = \begin{bmatrix} 2 \\ 10 \\ 4 \\ 2 \end{bmatrix}.$$

c_B sarà uguale a: $c_B = [5 \ \ 3 \ \ 0 \ \ 0]$, che moltiplicato per B^{-1}A e $B^{-1}b$

$$c_B\, B^{-1}A = [3 \ \ 5 \ \ 5 \ \ -2 \ \ 0 \ \ 0] \quad \text{e} \quad c_B\, B^{-1}b = [40]$$

Infine:

$$c_B\, B^{-1}A - c = [0 \ \ 0 \ \ 5 \ \ -2 \ \ 0 \ \ 0].$$

Il risultato della funzione obiettivo è 40 ma $c_B\, B^{-1}A - c$ contiene un elemento negativo, quindi la SBA (10, 2, 0, 0, 4, 2) non è soluzione ottima.

Sostituendo x_4 con la variabile uscente che troviamo con il test del minimo rapporto: { x_2: $\frac{2}{-1}$; x_1: $\frac{10}{1} = 10$, x_5: $\frac{4}{1} = 4$, x_6: $\frac{2}{1} = 2$ } la variabile uscente sarà x_6.

Iterazione 4

La nuova base sarà formata dalle variabili x_2, x_1, x_5 e x_4

$$B = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix} \quad \text{e} \quad B^{-1} =. \begin{bmatrix} -1 & 0 & 0 & 1 \\ 2 & 0 & 0 & -1 \\ 1 & 0 & 1 & -1 \\ -2 & 1 & 0 & 1 \end{bmatrix}$$

Moltiplicando B^{-1} per A e b avremo:

$$B^{-1}A = \begin{bmatrix} 0 & 1 & -1 & 0 & 0 & 1 \\ 1 & 0 & 2 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 & -1 \\ 0 & 0 & -2 & 1 & 0 & 1 \end{bmatrix} \quad \text{e} \quad B^{-1}b = \begin{bmatrix} 4 \\ 8 \\ 2 \\ 2 \end{bmatrix}.$$

c_B sarà uguale a: $c_B = [5 \ \ 3 \ \ 0 \ \ 0]$, che moltiplicato per B^{-1}A e $B^{-1}b$

$$c_B\, B^{-1}A = [3 \ \ 5 \ \ 1 \ \ 0 \ \ 0 \ \ 2] \quad \text{e} \quad c_B\, B^{-1}b = [44]$$

Infine avremo:

$$c_B\, B^{-1}A - c = [0\;\; 0\;\; 1\;\; 0\;\; 0\;\; 2].$$

Il risultato della funzione obiettivo è 44 e tutti gli elementi del vettore $c_B\, B^{-1}A - c$ sono positivi, il che vuol dire che la SBA (8, 4, 0, 2. 2, 0) è una soluzione ottima.

Nel prossimo capitolo verranno trattate alcune delle metodologie di risoluzione dei problemi di PL attraverso diversi strumenti informatici.

3. Problemi di PL e Strumenti Informatici

In questo capitolo vengono affrontate le modalità e tecniche utilizzate per risolvere i problemi di PL mediante alcuni strumenti informatici. Nello specifico, si approfondirà il modo in cui lavorano il risolutore di Excel e un programma da me elaborato in linguaggio Python.

3.1. Ottimizzazione attraverso l'utilizzo del risolutore di Excel

Il risolutore di Excel è un potente strumento informatico che riesce a risolvere i problemi di PL presentati in forma matriciale.

Spesso, la barra multifunzione di Excel non lo mostra di default.

In tal caso bisognerà andare dalla barra del menù su "Dati" e, cliccando con il tasto destro sulla barra multifunzione, selezionare "Personalizza barra multifunzione" (Figura 8).

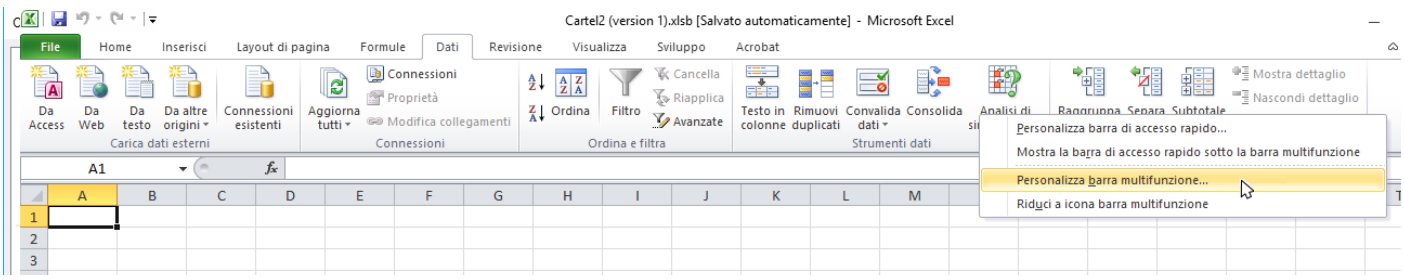


Figura 8: *Inserimento Risolutore (1)*

Successivamente in "Opzioni Excel", si dovrà selezionare dal menù la voce "Componenti aggiuntivi" > "Componente aggiuntivo Risolutore" > "Vai" (Figura 9).

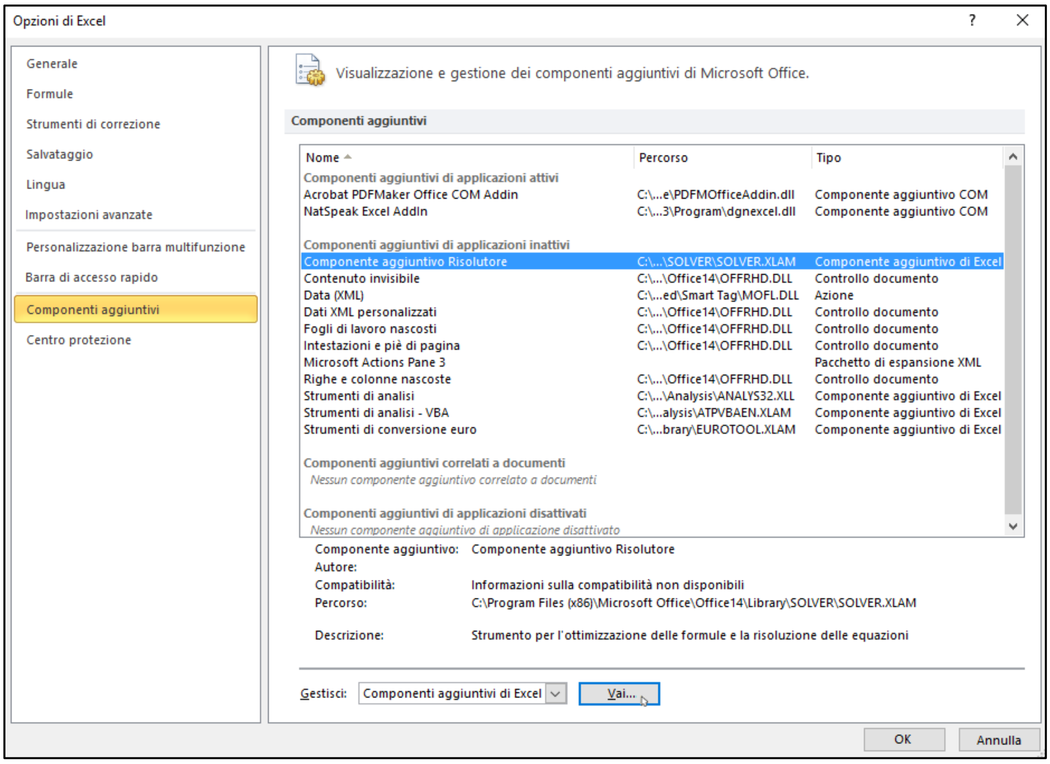


Figura 9: *Inserimento Risolutore (2)*

Dunque, andrà selezionato "Componente aggiuntivo Risolutore" e premuto "Ok" (Figura 10).

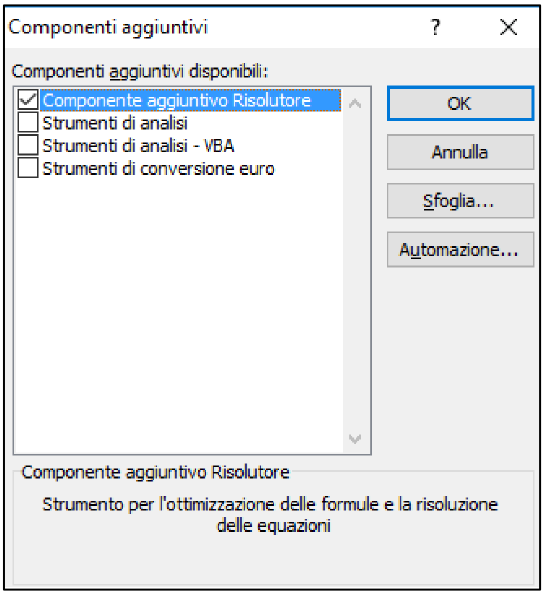


Figura 10: *Inserimento Risolutore (3)*

Ora il Risolutore sarà disponibile nella barra multifunzione alla sezione “Dati”. Riprendendo l’esempio utilizzato nel precedente capitolo si riscriverà il problema in forma matriciale, inserendo i dati come in figura (Figura 11).

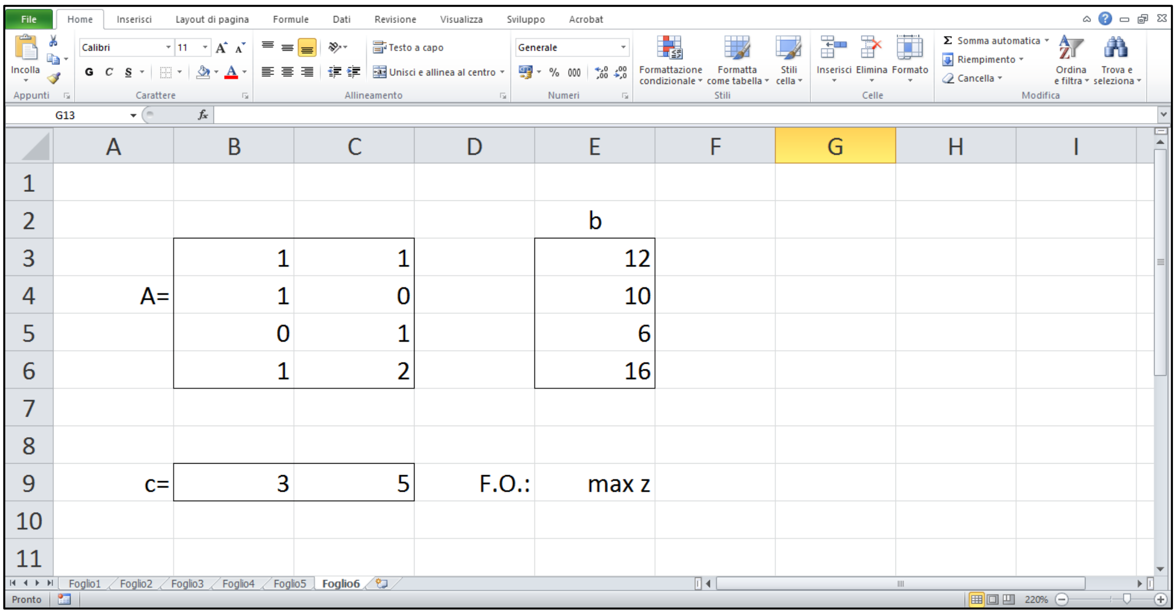


Figura 11: *Inserimento dei dati in Excel (1)*

Si dovrà, quindi, inscrivere il vettore colonna delle variabili (*x*), avente tanti zeri quante le incognite, e la formula della funzione obiettivo da massimizzare, utilizzando la formula =MATR.PRODOTTO(matrice1;matrice2), selezionando le celle del vettore riga *c* e del vettore colonna *x* e premendo i tasti CTRL+Shift+INVIO (Figura 12).

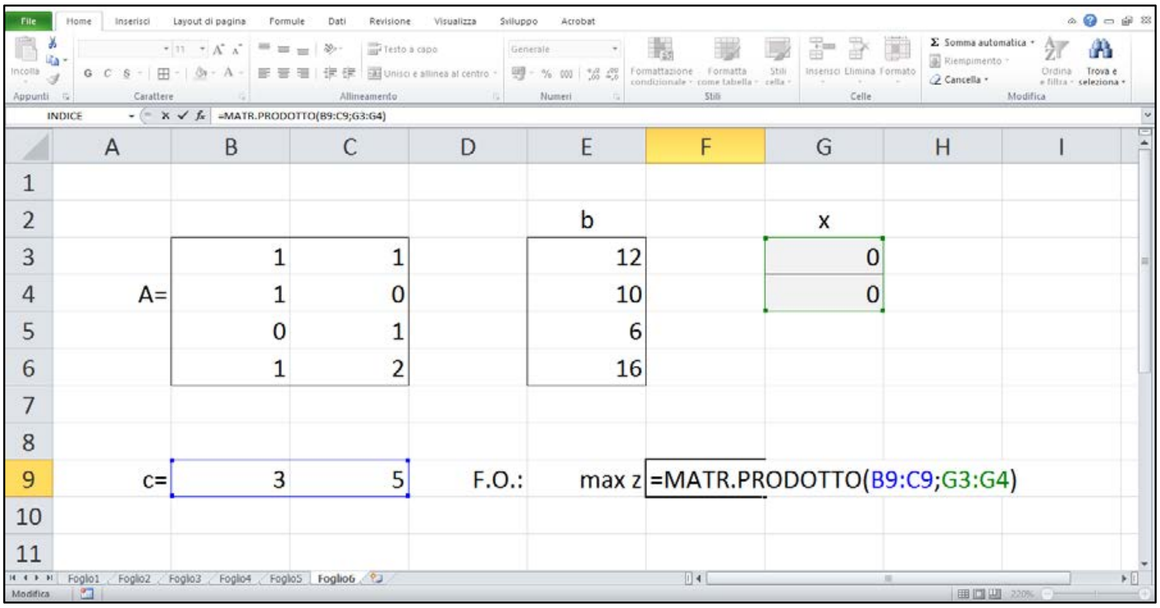


Figura 12: *Inserimento dei dati in Excel (2)*

Si dovrà costruire, poi, il sistema dei vincoli selezionando tante caselle quanti sono i vincoli, inserendo la funzione =MATR.PRODOTTO(matrice1;matrice2), selezionando la matrice dei coefficienti A e il vettore colonna delle incognite *x* e premendo i tasti CTRL+Shift+INVIO (Figura 13).

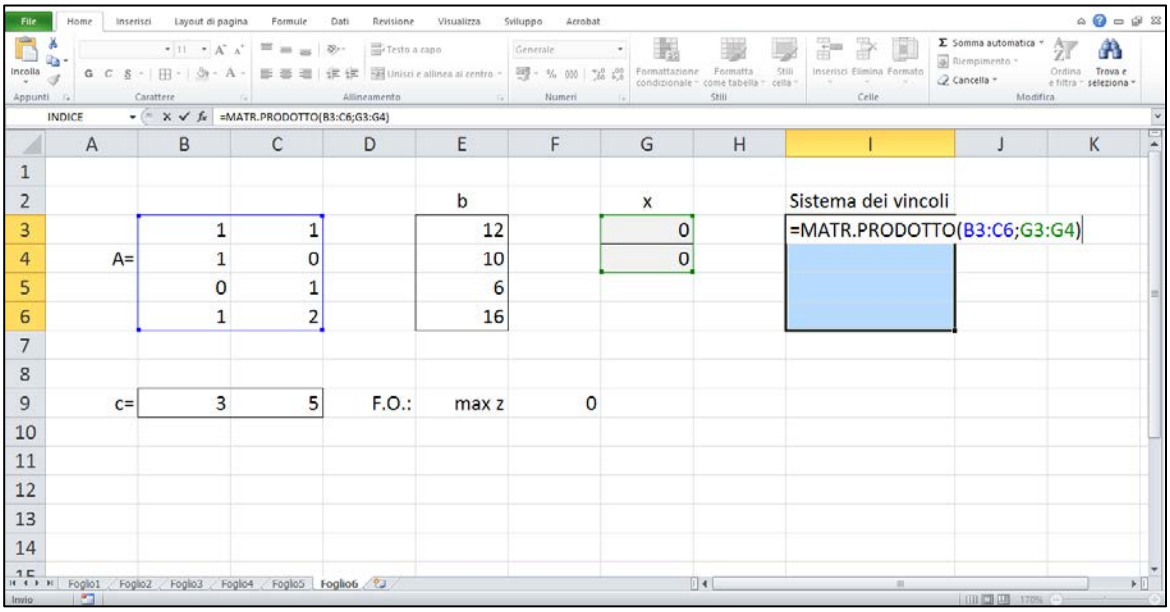


Figura 13: *Inserimento dei dati in Excel (3)*

A questo punto, in “Dati”, si potrà selezionare il Risolutore ed inserire in “Imposta obiettivo” la cella della funzione obiettivo (F9), si seleziona “Max”, volendo massimizzare la funzione (Figura 14),

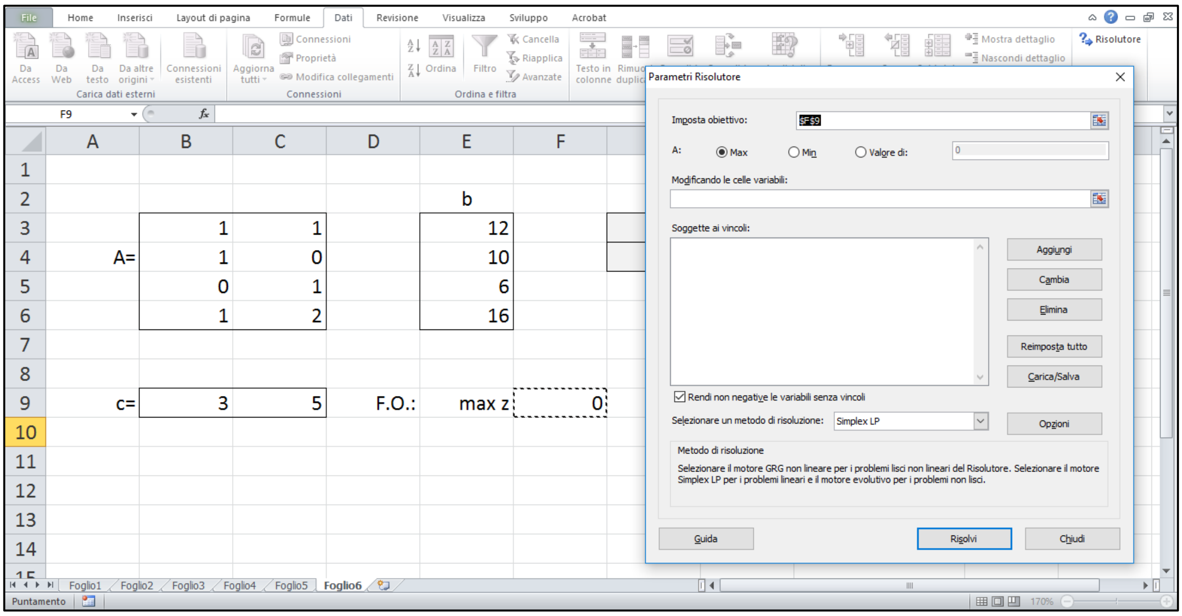


Figura 14: *Il Risolutore (1)*

in “Modificando le celle variabili” si selezionerà il vettore colonna delle incognite *x* (Figura 15),

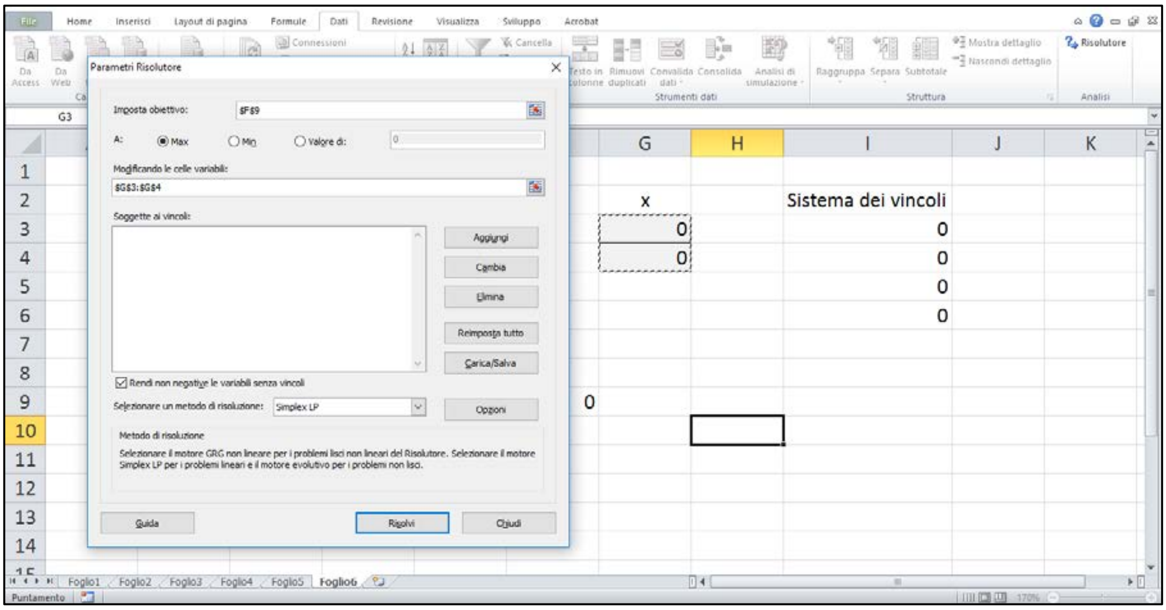


Figura 15: *Il Risolutore (2)*

in “Soggette ai vincoli” si dovrà cliccare su “Aggiungi”, e selezionare il sistema dei vincoli, ponendolo ≤ al vettore dei termini noti b, e premere “Ok” (Figura 16).

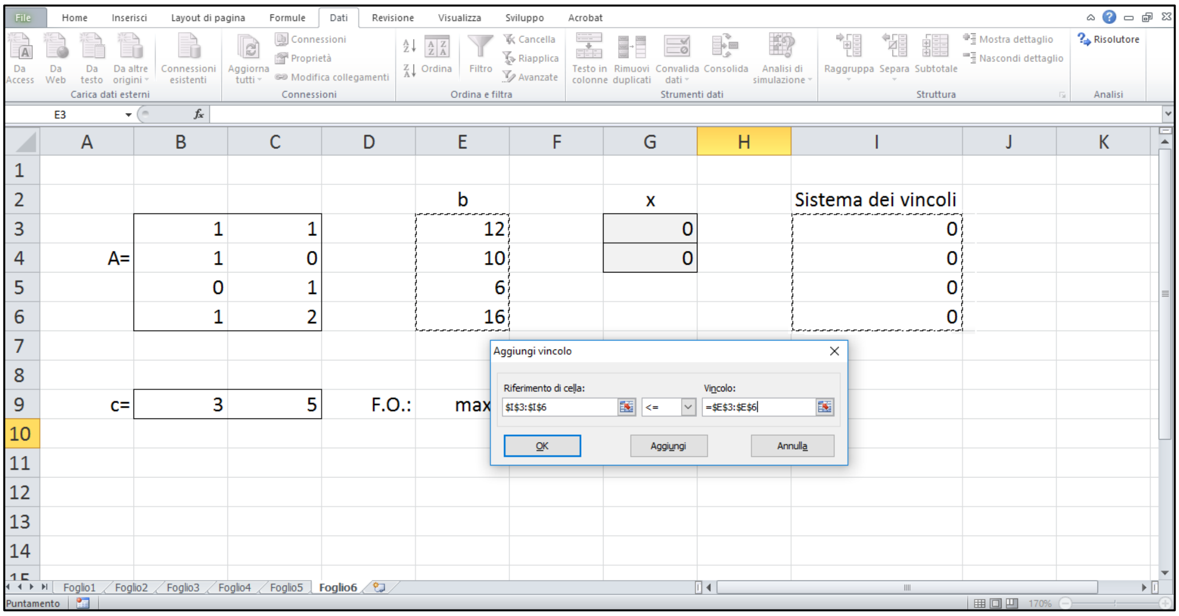


Figura 16: *Il Risolutore (3)*

In “Selezionare un metodo di risoluzione” ci si dovrà assicurare che sia impostato su “Simplex PL”. Premendo sul tasto “Opzioni” sarà possibile deselezionare il comando “Mostra risultati iterazione”, in modo da arrivare direttamente alla soluzione ottima (Figura 17).

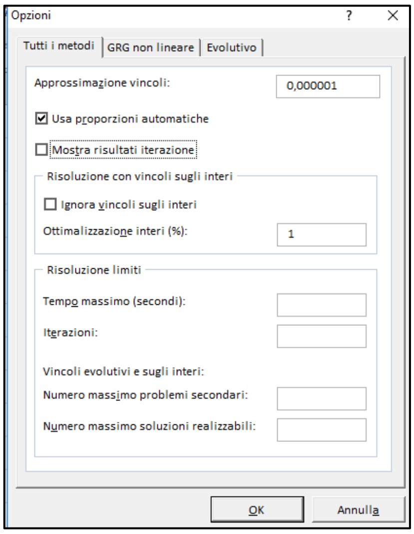


Figura 17: *Il Risolutore (4)*

Infine si selezionerà il tasto “Risolvi” e il risultato dell’ottimizzazione comparirà nella casella della funzione obiettivo (F9) (Figura 18).

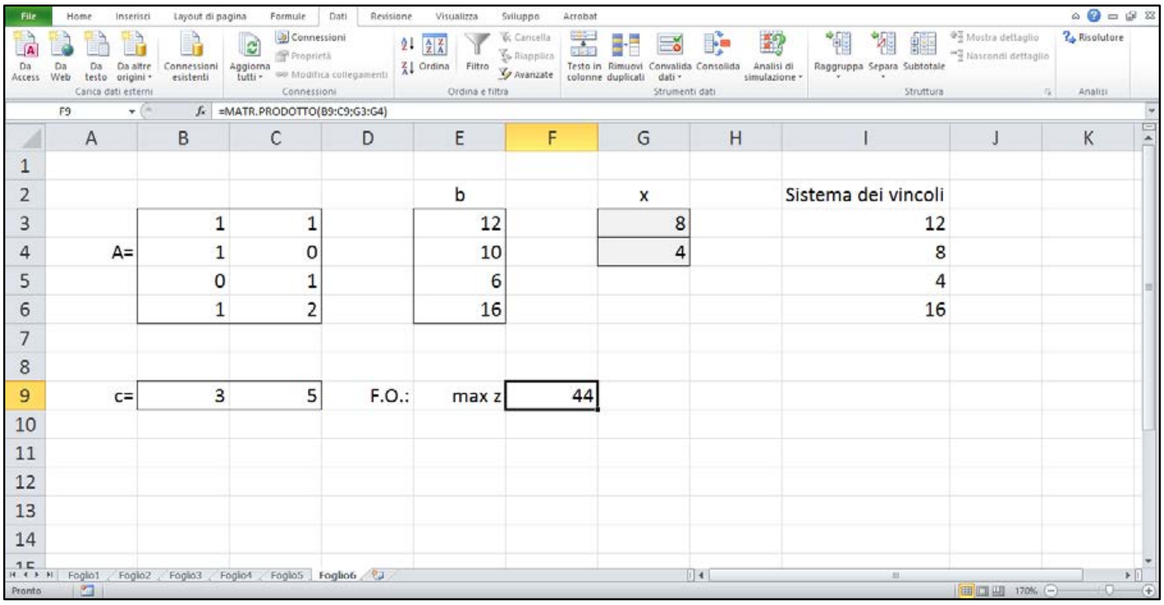


Figura 18: *Il Risolutore (5)*

3.2. Ottimizzazione attraverso la scrittura di un programma in Python

Python è un linguaggio di programmazione ad alto livello, orientato agli oggetti, rilasciato per la prima volta dall'olandese Guido Van Rossum il 20 Febbraio 1991. È un free software, compatibile con numerosi sistemi operativi, integrabile con altri linguaggi e caratterizzato dalla presenza di numerose librerie, il che lo rende uno dei linguaggi più utilizzati ed amati dai programmatori[13].

Proprio grazie all'utilizzo delle librerie, Python risulta il linguaggio ideale per scrivere un programma in grado di risolvere i problemi di PL.

In particolare, per creare il programma ho utilizzato la libreria PuLP, capace di creare file MPS o LP e risolvere problemi richiamando pacchetti e risolutori come GLPK, COIN-OR CLP/CBC, CPLEX, GUROBI, MOSEK, XPRESS, CHOCO, MIPCL e SCIP[14].

Il programma nella sua interezza si presenta come:

```
from pulp import *

print('Benvenuti nel programma di soluzione dei problemi di programmazione lineare.\n1. Si prega di scrivere "massimizzare" se si vuole massimizzare la funzione o "minimizzare" se la si vuole minimizzare seguiti dal pulsante INVIO.\n2. Si prega di inserire la funzione da massimizzare/minimizzare e premere INVIO.\n3. Si prega di inserire i vincoli seguiti dal pulsante INVIO.\n4. Terminati i vincoli si prega di scrivere "fine" e premere INVIO.\n\nOperatori:\nSomma: +\nSottrazione: -\nMoltiplicazione: *\nDivisione: /\nMinore uguale: <=\nMaggiore uguale: >=')

for i in range(0, 999999999):
  str = 'x{0} = LpVariable("x{1}", lowBound=0)'.format(i+1,i+1)
  exec(str)

print('')
maxmin = input('Inserire "massimizzare" o "minimizzare": ')

while True:
  if maxmin != "massimizzare" and maxmin != "minimizzare":
    maxmin = input('Impossimile soddisfare la richiesta.\nInserire "massimizzare" o "minimizzare": ')
  else:
    break

if maxmin == 'massimizzare':
  problem = LpProblem('Massimizzazione',LpMaximize)
  minmax = 'max'
else:
  problem = LpProblem('Minimizzazione',LpMinimize)
  minmax = 'min'

try:
  funzione = eval(input(f'Inserire la funzione da {maxmin}: {minmax} z = '))
  problem += funzione
except:
  funzione = eval(input(f'Tipologia di dato inserito non corretta.\nInserire la funzione da {maxmin}: {minmax} z = '))
  problem += funzione

while True:
  try:
```

```
    vincolo = eval(input('Inserire vincolo: '))
    problem += vincolo
  except NameError:
    break

if maxmin == 'massimizzare':
  illimit = 'superiormente'
else:
  illimit = 'inferiormente'

problem.solve()
stato = LpStatus[problem.status]
if stato == 'Unbounded':
  print(f'Il problema è illimitato {illimit}')
elif stato == 'Infeasible':
  print('Il problema è inammissibile')
else:
  print ("Risultati della ottimizzazione:")
for variabile in problem.variables():
  print (variabile.name, "=", variabile.varValue)
print ("Ottimizzazione totale:")
print (value(problem.objective))
```

Come primo passaggio bisogna importare la libreria PuLP nella sua interezza attraverso la linea di codice:

```
from pulp import *   .
```

Con il seguente testo:

```
print('Benvenuti nel programma di soluzione dei problemi di programmazione lineare.\n1. Si prega di scrivere "massimizzare" se si vuole massimizzare la funzione o "minimizzare" se la si vuole minimizzare seguiti dal pulsante INVIO.\n2. Si prega di inserire la funzione da massimizzare/minimizzare e premere INVIO.\n3. Si prega di inserire i vincoli seguiti dal pulsante INVIO.\n4. Terminati i vincoli si prega di scrivere "fine" e premere INVIO.\n\nOperatori:\nSomma: +\nSottrazione: -\nMoltiplicazione: *\nDivisione: /\nMinore uguale: <=\nMaggiore uguale: >=')
```

all'utilizzatore del programma verranno mostrate sull'interfaccia utente le istruzioni per l'utilizzo dello stesso e verranno presentati i caratteri da utilizzare per inserire le sezioni del problema. Utilizzando il browser Repl.it (ambiente di programmazione interattivo) il messaggio verrà mostrato come in Figura 19:

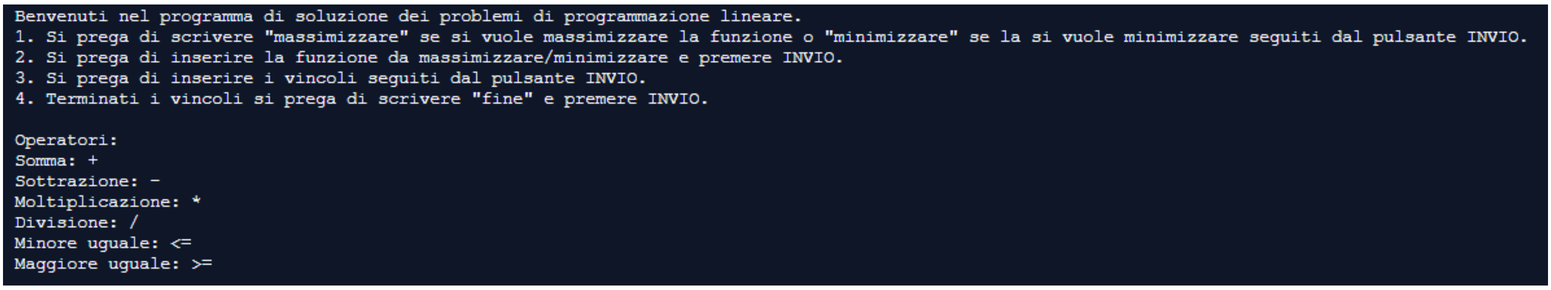


Figura 19: *Programma Python(1)*

Successivamente, attraverso la funzione `LpVariable` di PuLP, saranno create le incognite.

Poiché in Python non è possibile ricreare il concetto di infinito, ma potendo i problemi di PL avere un numero estremamente elevato di incognite, mediante il seguente blocco di codice sarà possibile creare un numero di incognite del tipo x_n molto ingente (in questo caso n=1.000.000.000):

```
for i in range(0, 999999999):
  str = 'x{0} = LpVariable("x{1}", lowBound=0)'.format(i+1,i+1)
  exec(str)
```

Nella funzione `LpVariable` sono presenti due parametri: con il primo (x{0}) viene indicato il *nome* della variabile; con il secondo (`lowBound`), determinato uguale a zero, vengono imposti al problema i vincoli di non negatività.

Attraverso il seguente blocco di codice:

```
print('')
maxmin = input('Inserire "massimizzare" o "minimizzare": ')

while True:
  if maxmin != "massimizzare" and maxmin != "minimizzare":
    maxmin = input('Impossimile soddisfare la richiesta.\nInserire "massimizzare" o "minimizzare": ')
  else:
    break
```

dopo aver creato una spaziatura nel testo sull'interfaccia, viene richiesto all'utente se la funzione ad oggetto debba essere massimizzata o minimizzata. Con l'utilizzo di un ciclo **`while`** si eviterà il rischio che eventuali errori nella digitazione dei comandi "massimizzare" e "minimizzare" possano generare un errore nel programma, con conseguente arresto dell'esecuzione dello stesso; qualora la digitazione fosse errata verrà presentato un messaggio che manifesterà l'errore e inviterà l'interlocutore a reinserire il comando.

Definita la massimizzazione/minimizzazione della funzione, con il comando `LpProblem,` al problema verrà assegnato l'obiettivo di essere massimizzato o minimizzato attraverso l'utilizzo delle istruzioni condizionali **`if`** ed **`else`** e l'inserimento dei parametri `LpMaximiz` e `LpMinimize.`

```
if maxmin == 'massimizzare':
  problem = LpProblem('Massimizzazione',LpMaximize)
  minmax = 'max'
else:
  problem = LpProblem('Minimizzazione',LpMinimize)
  minmax = 'min'
```

Con le variabili “minmax” verrà assegnato il valore “max” se la funzione viene massimizzata e “min” se la funzione viene minimizzata e tali caratteri compariranno all’interno della funzione obiettivo al momento dell’inserimento della stessa nell’interfaccia utente.

Come appena detto, con il seguente blocco **`try`**

```
try:
  funzione = eval(input(f'Inserire la funzione da {maxmin}: {minmax} z = '))
  problem += funzione
except:
  funzione = eval(input(f'Tipologia di dato inserito non corretta.\nInserire la funzione da {maxmin}: {minmax} z = '))
  problem += funzione
```

si richiede all’utente di inserire la funzione obiettivo.

Qualora fossero presenti caratteri errati, il blocco **`except`** mostrerà il messaggio d’errore chiedendo il reinserimento della funzione, che verrà salvata attraverso la linea di codice `problem += funzione.`

Dunque, verrà richiesto l’inserimento dei vincoli attraverso il ciclo **`while`**:

```
while True:
  try:
    vincolo = eval(input('Inserire vincolo: '))
    problem += vincolo
  except NameError:
    break
```

Una volta concluso l’inserimento dei vincoli l’utente dovrà, secondo istruzioni, digitare la parola “fine”, che verrà letta dal programma come un errore del tipo `NameError` e interromperà il ciclo.

Infine, con il seguente blocco di istruzioni:

```
if maxmin == 'massimizzare':
  illimit = 'superiormente'
else:
  illimit = 'inferiormente'

problem.solve()
stato = LpStatus[problem.status]
if stato == 'Unbounded':
  print(f'Il problema è illimitato {illimit}')
elif stato == 'Infeasible':
  print('Il problema è inammissibile')
else:
  print ("Risultati della ottimizzazione:")
for variabile in problem.variables():
  print (variabile.name, "=", variabile.varValue)
print ("Ottimizzazione totale:")
print (value(problem.objective))
```

il problema viene risolto mediante il comando `problem.solve()`.

Attraverso il blocco costituito dalle variabili condizionali **`if`**, **`elif`** ed **`else`** ci si assicura che il problema sia ammissibile e limitato. Qualora lo fosse, viene stampato nell’interfaccia il valore delle variabili ottimizzate e il risultato della funzione obiettivo. Se, al contrario, fosse illimitato o inammissibile, verranno stampati rispettivamente i messaggi “Il problema è superiormente/inferiormente” illimitato, a seconda che sia un problema di massimizzazione o minimizzazione, e “Il problema è inammissibile”.

Dal punto di vista dell’utente, sempre utilizzando Repl.it, terminato l’inserimento dei dati del problema e l’ottimizzazione, il programma si presenterà come in Figura 20:

```
Benvenuti nel programma di soluzione dei problemi di programmazione lineare.
1. Si prega di scrivere "massimizzare" se si vuole massimizzare la funzione o
   "minimizzare" se la si vuole minimizzare seguiti dal pulsante INVIO.
2. Si prega di inserire la funzione da massimizzare/minimizzare e premere INVIO.
3. Si prega di inserire i vincoli seguiti dal pulsante INVIO.
4. Terminati i vincoli si prega di scrivere "fine" e premere INVIO.

Operatori:
Somma: +
Sottrazione: -
Moltiplicazione: *
Divisione: /
Minore uguale: <=
Maggiore uguale: >=

Inserire "massimizzare" o "minimizzare": massimizzare
Inserire la funzione da massimizzare: max z = 3*x1+5*x2
Inserire vincolo: x1+x2<=12
Inserire vincolo: x1<=10
Inserire vincolo: x2<=6
Inserire vincolo: x1+2*x2<=16
Inserire vincolo: fine
Risultati della ottimizzazione:
x1 = 8.0
x2 = 4.0
Ottimizzazione totale:
44.0
>
```

Figura 20: *Programma Python(2)*

4. Un’applicazione pratica del Metodo del Simplesso: Problemi di Flusso a Costo Minimo

In questo capitolo viene trattato il Metodo del Simplesso applicato ad un problema di Flusso a Costo Minimo e dunque alla Teoria dei Grafi, branca della matematica che si occupa di reti di punti collegati da linee. La teoria dei grafi, inizialmente utilizzata per la soluzione di problemi di matematica ricreativa, è cresciuta fino a diventare un'area significativa della ricerca matematica, con applicazioni in chimica, ricerca operativa, scienze sociali e informatica[15].

4.1. Teoria dei Grafi

La Teoria dei Grafi è una branca della matematica che studia i rapporti che intercorrono tra dei punti, chiamati nodi o vertici, collegati da linee, denominate archi, che possono essere orientate o meno.
Lo studio nasce nel 1735, quando il matematico svizzero Eulero risolse il “Problema dei ponti di Königsberg”, ovvero un enigma basato sulla ricerca di un percorso attraverso il quale si possa attraversare una sola volta ognuno dei sette ponti costruiti su un fiume biforcuto avente un’isola al centro (Figura 21).

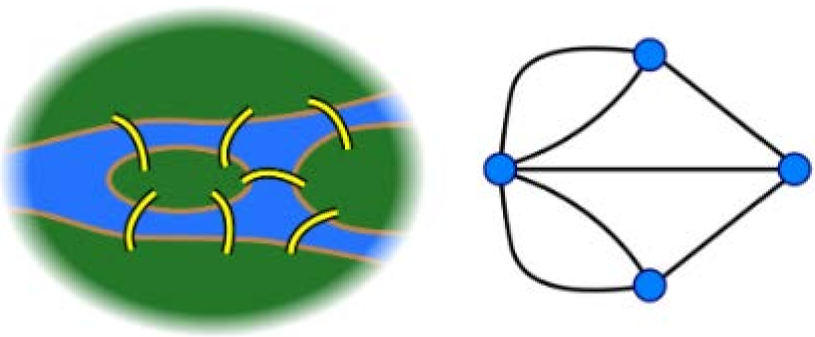


Figura 21: *I ponti di Königsberg*

Nella teoria dei grafi si intende come “multigrafo” la fattispecie nella la quale due vertici qualsiasi sono collegati da più di un arco.
Come visto nel problema dei ponti di Königsberg l’obiettivo è di attraversare tutti i ponti una sola volta: si dirà, quindi, euleriano un grafo nel quale è possibile spostarsi da un nodo all’altro passando per tutti gli archi una sola volta. Con il termine “grado” si fa riferimento al numero degli archi che entrano o escono da esso.

Eulero risolse il problema sostenendo che non esiste un percorso che soddisfi tali requisiti e in questo modo dimostrò il primo teorema della Teoria dei Grafi[15]:

> **Teorema 5** (Primo teorema della teoria dei grafi)**:** Un multigrafo privo di vertici isolati è euleriano se e soltanto se è connesso ed ogni suo vertice ha grado pari.

Tale teorema fu poi modificato ed ampliato al fine di rendere il risultato di Eulero completo:

> **Teorema 6**: Un grafo G privo di vertici isolati ha un cammino euleriano non chiuso se e soltanto se G è connesso ed ha esattamente due vertici dispari. In tal caso, se u e v sono i due vertici dispari, allora tutti i cammini euleriani di G iniziano e terminano in u e v[16].

Nel 1857 il matematico irlandese William Rowan Hamilton (1805 – 1865) inventò un "puzzle" che consisteva nel trovare un percorso lungo i bordi di un dodecaedro che iniziasse e finisse nello stesso vertice e passasse per ogni vertice una e una sola volta.: tale problema prese il nome di "Circuito hamiltoniano".
I problemi connessi a tali circuiti sono più difficili da caratterizzare rispetto ai grafi euleriani, poiché non sono ancora conosciute le condizioni necessarie e sufficienti per la loro esistenza in un grafo connesso. Applicazioni della Teoria dei Grafi furono poi utilizzate nella topologia, nella chimica, nelle scienze sociali, nell'informatica e nella ricerca operativa, nel cui ambito furono applicati a tale teoria algoritmi per l'ottimizzazione di problemi: tra di essi, il Metodo del Simplesso[15].

4.2. Problemi di Flusso a Costo Minimo

Per Problema di Flusso a Costo Minimo si intende un problema di ottimizzazione finalizzato ad individuare il percorso meno costoso per far passare un determinato quantitativo di flusso attraverso una rete.
Immaginando di dover minimizzare il costo di trasporto delle informazioni (rappresentate dai Mbit) che vengono scambiate all'interno di una rete di computer si può formalizzare il problema come segue:
Siano

- G = (V,A) il grafo orientato e connesso che rappresenta la rete di computer;
- c_{ij} il costo di trasporto unitario lungo l'arco (i, j) avente $(i, j) \in$ A;

- x_{ij} le variabili che rappresentano la quantità di informazione (Mbit) inviata lungo l'arco avente (i, j) $\in$ A;
- b_i interi i vincoli di quantità (Mbit) di informazione uscente o entrante nell'unità di tempo (s) aventi *i* $\in$ V e tali che $\sum_{i \in V} b_i = 0$;

si consideri inoltre che:

- i nodi aventi $b_i > 0$ sono detti nodi sorgente e rappresentano i computer che generano e inviano le informazioni;
- i nodi aventi $b_i < 0$ sono detti nodi destinazione e rappresentano i computer che ricevono e processano le informazioni:
- i nodi aventi $b_i = 0$ sono detti nodi transito e rappresentano i computer nei quali le informazioni transitano solo da un computer all'altro

La condizione $\sum_{i \in V} b_i = 0$ garantisce che le informazioni prodotte dai nodi sorgente siano esattamente pari alle informazioni processate dai nodi destinazione.
Avendo il flusso uscente da *i*:

$$\sum_{(i,j) \in A} x_{ij}$$

E il flusso entrante in *i*:

$$\sum_{(j,i) \in A} x_{ji}$$

in ogni nodo *i* $\in$ V si deve avere:

$$\sum_{(i,j) \in A} x_{ij} - \sum_{(j,i) \in A} x_{ji} = b_i \qquad \forall\, i \in V$$

Si avranno inoltre i vincoli di non negatività e di interezza di flusso lungo ciascun arco:

$$x_{ij} \geq 0 \text{ interi} \qquad \forall\, (i, j) \in V$$

Riscrivendo, dunque il modello matematico al completo si avrà:

$$min \sum_{(i,j) \in A} c_{ij} x_{ij}$$

$$\sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = b_i \quad \forall\, i \in V$$

$$x_{ij} \geq 0 \text{ interi} \quad \forall\, (i,j) \in V$$

esprimibile in forma matriciale come:

$$\min CX$$
$$AX = B$$
$$X \geq 0$$

dove:

- X è il vettore le cui | A | componenti sono le variabili x_{ij} del problema;
- C è il vettore le cui | A | componenti sono i costi unitari c_{ij};
- B è il vettore le cui | V | componenti sono i valori b_i ;
- A è la matrice dei vincoli di uguaglianza con | V | righe e | A | colonne.

È importante evidenziare che in un problema di Flusso a Costo Minimo ogni vincolo può essere ottenuto attraverso la somma di tutti gli altri e, quindi, è possibile eliminare uno ed un solo vincolo al fine di non rendere il sistema ridondante.
È ora possibile risolvere il problema mediante l'utilizzo del Metodo del Simplesso.

4.3. Il Metodo del Simplesso e i problemi di Flusso a Costo Minimo

Come primo passo per risolvere il problema di Flusso a Costo Minimo attraverso il Metodo del Simplesso bisogna individuare la prima base e verificarne l'ottimalità.
È doveroso fare un'importante osservazione: In un problema di flusso su rete a costo minimo vi è una corrispondenza uno a uno tra basi ed alberi di supporto, ovvero ad ogni insieme di | V | −1 variabili che formano una base corrisponde un albero di supporto e viceversa.
Per albero di supporto si intende un grafo non orientato, contenente tutti i nodi del grafo principale e solo un sottoinsieme di archi, tali che presi due vertici, essi sono connessi da uno e un solo arco (Figura 22).

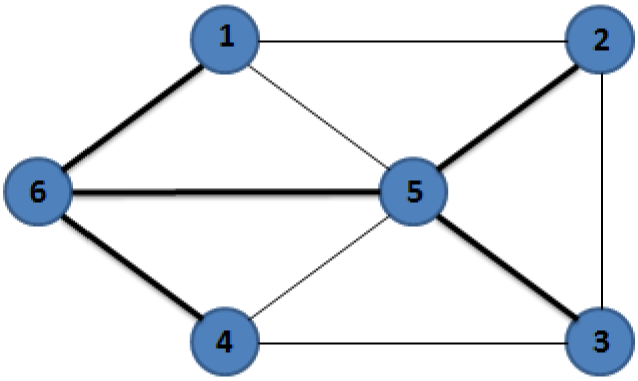


Figura 22: *Albero di Supporto(1)*

Ottenuta la prima base, e individuata dunque la relativa soluzione di base, si procede con la verifica dell'ammissibilità: questa condizione si ottiene se tutte le variabili in base sono non negative.
Qualora non tutte le variabili in base fossero strettamente maggiori di zero si parlerebbe di soluzione di base degenere.
A questo punto si dovrà verificare se la SBA individuata, oltre ad essere ammissibile, sia soluzione ottima: qualora i coefficienti di costo ridotto di tutte le variabili fuori base fossero non negativi, la SBA presa in considerazione potrà dirsi soluzione ottima, in quanto inserendo una nuova variabile in base aumenterebbe il valore della funzione obiettivo che, invece, deve essere minimizzato. Se, viceversa, i coefficienti fossero tutti strettamente positivi, la SBA individuata sarebbe l'unica "soluzione ottima".
Per calcolare i coefficienti di costo ridotto delle variabili fuori base bisognerà prendere l'arco relativo a una di quelle variabili; aggiungere l'arco all'albero di supporto corrispondente alla base attuale; considerare l'unico ciclo che si forma con tale aggiunta; fissare come verso del ciclo quello dell'arco relativo alla variabile fuori base; calcolare il coefficiente di costo ridotto (sommando tra loro tutti i costi relativi agli archi attraversati dal ciclo nel loro stesso verso e sottraendo al risultato i costi degli archi attraversati dal ciclo in senso opposto).
Qualora la condizione di non ottimalità non fosse soddisfatta bisognerà effettuare un cambio di base, attraverso l'inserimento di una delle variabili fuori base avente il coefficiente di costo ridotto negativo e facendo uscire l'arco in base al quale, più rapidamente, si annulla il flusso all'ingresso dell'entrante. Questa procedura si effettua mediante l'utilizzo del test del minimo rapporto, che nei problemi di Flusso a Costo Minimo segue i seguenti passaggi:

- si aggiunge l'arco relativo alla variabile che si farà entrare in base;
- si forma un ciclo che viene orientato secondo il verso di questo arco;

- si porta a Δ il flusso lungo tale arco;
- si incrementa di Δ il flusso lungo gli archi del ciclo attraversati secondo il proprio verso;
- si decrementa di Δ il flusso lungo gli archi attraversati in verso opposto al proprio.

È possibile che il flusso non venga decrementato lungo nessun arco (è questo il caso in cui il ciclo che si forma è orientato): in tal caso il problema è illimitato.

Verificata la limitatezza del problema ed individuata una nuova SBA si ripetono le tre fasi dell'Algoritmo del Simplesso (verifica di ottimalità, verifica di illimitatezza, cambio di base) fino all'individuazione della soluzione ottima[17].

4.4. Un esempio numerico

Riprendendo l'esempio del precedente paragrafo sulle informazioni trasportate attraverso la rete di computer possiamo immaginare un problema come il seguente:

data la rete G = (V, A) avente V = {1,2,3,4,5} e A = {(1, 2); (1, 3); (1, 5); (2, 3); (3, 4); (4, 2); (4, 5); (5, 3)}, i coefficienti della funzione obiettivo pari a:

(i, j)	(1, 2)	(1, 3)	(1, 5)	(2, 3)	(3, 4)	(4, 2)	(4, 5)	(5, 3)
c_{ij}	5	-2	2	-4	0	6	3	4

e i vincoli:

i	1	2	3	4	5
b_i	2	5	1	-4	-4

Il grafo è rappresentato come in figura (Figura 23):

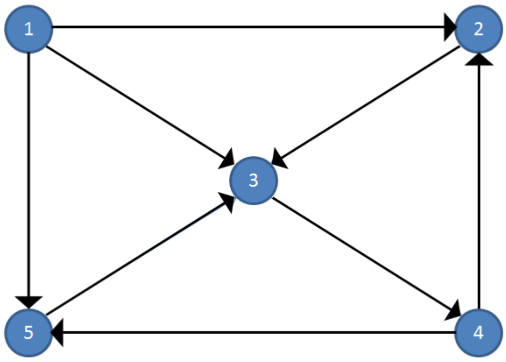


Figura 23: *Grafo*

È possibile formalizzare il modello matematico come:

$$\min 5x_{12} - 4x_{23} + 6x_{42} - 2x_{13} + 0x_{34} + 2x_{15} + 4x_{53} + 3x_{45}$$

$$x_{12} + x_{13} + x_{15} = 2$$

$$x_{23} - x_{12} - x_{42} = 5$$

$$x_{34} - x_{13} - x_{23} - x_{53} = 1$$

$$x_{42} + x_{45} - x_{34} = -4$$

$$x_{53} - x_{15} - x_{45} = -4$$

$$x_{12}, x_{23}, x_{42}, x_{13}, x_{34}, x_{15}, x_{53}, x_{45} \geq 0 \text{ interi.}$$

Esprimendo in forma matriciale la matrice dei vincoli A, avremo:

A =

	(1, 2)	(1, 3)	(1, 5)	(2, 3)	(3, 4)	(4, 2)	(4, 5)	(5, 3)
1	1	1	1	0	0	0	0	0
2	-1	0	0	1	-1	0	0	0
3	0	-1	0	-1	0	1	-1	0
4	0	0	0	0	1	-1	0	1
5	0	0	-1	0	0	0	1	-1

Come visto nello scorso paragrafo, ogni vincolo è calcolabile come la somma degli altri e, dunque, è possibile eliminare uno e un solo vincolo per rendere il sistema non ridondante. Per convenzione si elimina l'ultimo.

Il modello diverrà, dunque:

$$\min 5x_{12} - 4x_{23} + 6x_{42} - 2x_{13} + 0x_{34} + 2x_{15} + 4x_{53} + 3x_{45}$$

$$x_{12} + x_{13} + x_{15} = 2$$

$$x_{23} - x_{12} - x_{42} = 5$$

$$x_{34} - x_{13} - x_{23} - x_{53} = 1$$

$$x_{42} + x_{45} - x_{34} = -4$$

$$x_{12}, x_{23}, x_{42}, x_{13}, x_{34}, x_{15}, x_{53}, x_{45} \geq 0 \text{ interi.}$$

e la matrice A:

A =

	(1, 2)	(1, 3)	(1, 5)	(2, 3)	(3, 4)	(4, 2)	(4, 5)	(5, 3)
1	1	1	1	0	0	0	0	0
2	-1	0	0	1	-1	0	0	0
3	0	-1	0	-1	0	1	-1	0
4	0	0	0	0	1	-1	0	1

Per calcolare la prima SBA si considera il seguente albero di supporto (Figura 24)

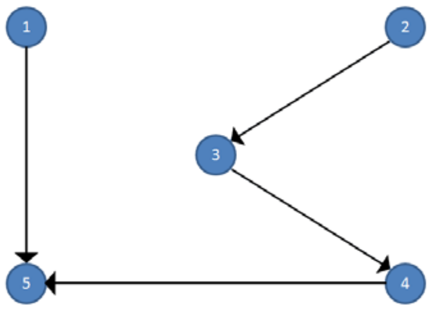


Figura 24: *Albero di supporto(2)*

e si prendendo, quindi, in base x_{15}, x_{23}, x_{34}, x_{45}. Dunque $x_{12} = x_{13} = x_{42} = x_{53} = 0$.

Sostituendo il valore delle variabili nulle si risolve il sistema:

$$x_{15} = 2$$

$$x_{23} = 5$$

$$x_{34} - x_{23} = 1$$

$$x_{45} - x_{34} = -4$$

la cui soluzione è:

$$x_{15} = 2 \quad x_{23} = 5 \quad x_{34} = 6 \quad x_{45} = 2$$

e il valore della funzione obiettivo è pari a -10.

Per verificare l'ottimalità della soluzione bisognerà calcolare i coefficienti di costo ridotti: aggiungendo come primo l'arco (1, 3) all'albero di supporto si crea il ciclo 1 → 3 → 4 → 5 → 1 (Figura 25)

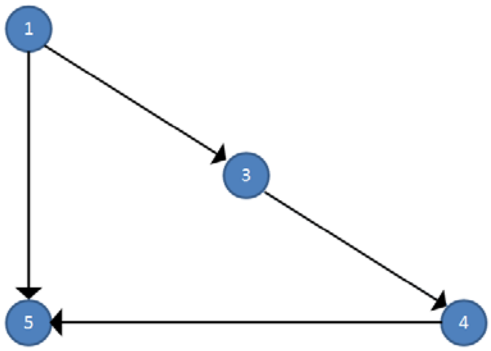


Figura 25: *Calcolo dei coefficienti di costo ridotti*

Il ciclo attraversa nel loro verso gli archi (1, 3), (3, 4) e (4, 5), mentre attraversa nel senso opposto l'arco (5, 1). Dunque, il coefficiente di costo ridotto di (1, 3) sarà pari a:

$$c_{13} + c_{34} + c_{45} - c_{51} = -2 + 0 + 3 - 2 = -1.$$

Essendo il coefficiente negativo ne deriviamo che la SBA non è soluzione ottima al problema.

Gli altri coefficienti sono rispettivamente pari a

$$c_{42} = 2$$

$$c_{12} = 2$$

$$c_{53} = 7.$$

Bisognerà, quindi, sostituire un arco tra quelli fuori base per costruire una nuova SBA e, come detto nel precedente paragrafo, si prenderà quello con il coefficiente dei costi ridotti negativo: nel nostro caso (1, 3).

Inserito il nuovo arco e creato il ciclo avente il suo verso, si porta a Δ il flusso lungo tale arco, si incrementa il flusso lungo gli archi del ciclo attraversati secondo il proprio verso, e si decrementa quello degli archi attraversati in verso opposto, in entrambi i casi di Δ. Si avrà dunque:

$$x_{13} = \Delta$$

$$x_{34} = 6 + \Delta$$

$$x_{45} = 2 + \Delta$$

$$x_{15} = 2 - \Delta.$$

Potendo crescere fino a $\Delta = 2$ sarà x_{15} la variabile uscente.

La nuova base sarà $\{x_{13}, x_{23}, x_{34}, x_{45}\}$, la SBA (Figura 26)

$$x_{13} = 2 \quad x_{23} = 5 \quad x_{34} = 8 \quad x_{45} = 4$$

$$x_{12} = x_{15} = x_{42} = x_{53} = 0$$

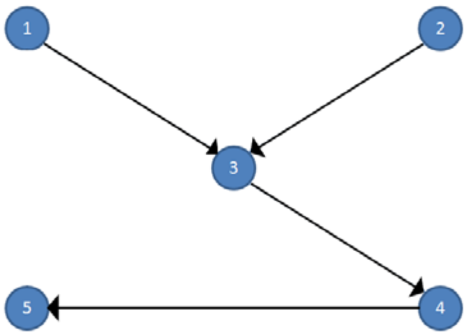


Figura 26: *Albero di supporto(3)*

e il valore della funzione obiettivo è di $-10 + c_{13}\Delta = -12$.

I coefficienti di costo ridotto delle variabili fuori base è pari a

$$c_{42} = 2$$

$$c_{12} = 3$$

$$c_{53} = 7$$

$$c_{15} = 1$$

ed essendo tutti strettamente positivi possiamo affermare che la presente SBA è unica soluzione ottima[17].

Conclusioni

Il presente elaborato si è posto l'obiettivo di analizzare l'origine, il funzionamento e alcune delle possibili applicazioni del metodo del Simplesso.
All'esito dell'attività illustrata nelle pagine che precedono è possibile formulare le seguenti considerazioni.

Attraverso l'analisi delle diverse forme con cui questo potente algoritmo risolve i quesiti che gli vengono sottoposti si è constatato come sia possibile ridurre al minimo il numero di passaggi da effettuare per massimizzare o minimizzare la funzione obiettivo senza incorrere nella necessità di dover analizzare ogni possibile vertice del politopo, così da risolvere problemi anche di assai notevole complessità senza la necessità di ricorrere a computer dotati di potenza di calcolo fuori dall'ordinario.

Di particolare rilievo è stata l'analisi degli strumenti informatici Excel e Python, interscambiabili e reciprocamente integrabili: da un lato, il Risolutore di Excel è un ottimo mezzo per il raggiungimento degli obiettivi preposti in quanto permette, in maniera molto semplice ed intuitiva, di risolvere i problemi, pur mostrando evidenti limiti in presenza di un'elevata mole di dati; al contrario, utilizzando un codice in Python – linguaggio compatibile con quasi tutti i sistemi operativi, tuttavia da "maneggiare con cura" per evitare eventuali bug - è possibile processare più velocemente l'insieme dei dati, anche al crescere degli stessi, e con maggior flessibilità, essendo consentito modificare le linee di codice in funzione delle esigenze contingenti che la situazione pone.

È emerso, poi, come il Metodo del Simplesso possa operare utilmente in un numero assai ampio di campi: i problemi di Flusso a Costo Minimo e la massimizzazione dei profitti o la minimizzazione dei costi per un'azienda, esaminati in questo lavoro, rappresentano una limitata casistica di essi.

Quanto detto ci porta a concludere che il metodo del Simplesso, abbinato alle sempre più evolute conoscenze informatiche, è uno strumento potentissimo per risolvere i problemi, da quelli della vita quotidiana fino ai più complessi del mondo finanziario, e il suo funzionamento ci insegna un'importante lezione di vita: se si usa l'intelligenza è possibile trarre il massimo risultato dal minimo sforzo.

Bibliografia e Sitografia

1. Horst R, Pardalos P.M. (1995) *Handbook of Global Optimization*. Springer-Science+Business Media, B.V.
2. Guarraggio A. (2008) *Enciclopedia della Scienza e Della Tecnica*. Treccani.
3. Eulero L. (1744) *Methodus Inveniendi Lineas Curvas Maximi Minimive Proprietate Gaudentes*. (ed. 2009). Bertrams Print on Demand.
4. Aleksandrov P. S., Bashmakova I.G. Yushkevich A P., (1969). *History of Mathematics in Russia up to 1917* (review). Uspekhi Mat. Nauk, 24:5 256–258.
5. Sakarovitch J. (2005). *Gaspard Monge, Géometrie Descriptive, First Edition* (1795). Capitolo 17 di *Landmark Writings in Western Mathematics 1640-1940*. Elsevier.
6. Koopmans, T.C. (1975). *Nobel Memorial Lecture: Concepts of optimality and their uses*. (December 11).
7. Ekwonwune E.N., Edebatu D.C. (2000). App*lication of Linear Programming Algorithm in the Optimization of Financial Portfolio of Golden Guinea Breweries*. Computing in Science and Engineering. Vol 2 N1.
8. Millan Gasca A. (2005) *Fabbriche, Sistemi, Organizzazioni: Storia Dell'ingegneria Industriale*. Springer.
9. Tichatschke R. (2012). *On the Shoulders of Giants: A brief excursion into the history of mathematical programming.* In Discussiones Mathematicae Differential Inclusions, Control and Optimization 32: 5–44.
10. Bortot P., Magnani U., Olivieri G., Rossi F.A., Torrigiani M. (1998) *Matematica Finanziaria*, II ed, Monduzzi.
11. Roma M. (2019) Appunti Delle Lezioni di Ricerca Operativa. http://www.dis.uniroma1.it/~roma/didattica.
12. Di Summa M. (2019) Il Metodo del Simplesso. Capitolo 3. https://www.math.unipd.it/~disumma/OD-Cap3.pdfhttps://www.python.it/
13. http://www.python.it/
14. https://pypi.org/project/PuLP/
15. Carlson S. C., *Graph Theory*, Encyclopaedia Britannica: https://www.britannica.com/topic/graph-theory
16. Casolo C. (2009) Corso di Teoria dei Grafi. Capitolo 1. http://web.math.unifi.it/users/casolo/dispense/tgrafi2009.pdf
17. Locatelli M. (2008) *Appunti per il corso di Ricerca Operativa.* Capitolo 9 http://www.di.unito.it/~locatell/didattica/ro1/ro1-bf.pdf