Metodi Matematici per le Decisioni d'Impresa
Paolo Pellizzari
This version: Sep 2015

Programma (bozza)

I sett	14/9	Introduzione al corso, sistemi lineari
	15/9	Teoremi di Cramer, Rouché-Capelli...
	16/9	Inversa e pseudo inversa
II sett	21/9	State preference model, replicabilità
	22/9	Copertura (hedging)
	23/9	Esempi di arbitraggio
III sett	28/9	Arbitraggio e prezzi degli stati
	29/9	Introduzioni all'ottimizzazione, condizioni del primo/secondo ordine
	30/10	Ottimizzazione numerica e `optim`
IV sett	5/10	Relazione fra pseudo inversa e ottimizzazione
	6/10	Arbitraggio/copertura approssimata
	7/10	Ottimizzazione vincolata e `constrOptim`
V sett	12/10	Autovettori, autovalori e intro all'AHP
	13/10	Saaty's Analytic Hierarchy Process (AHP)
	14/10	*Mock exam*

Tabella 1: Bozza di programma.

Questo pezzetto di codice è molto utile per il corso, lo includo nella prima pagina in modo che sia facile copiarlo e incollarlo al bisogno.

```
require(MASS)
ker <- function(A){ # utilizzata per determinare il kernel di A
Null(t(A))
}
```

1 Sistemi Lineari

Considerate un sistema di equazioni lineari (in breve, sistema lineare) scritto in forma matriciale

$$\mathbf{A}x = b,$$

in cui $\mathbf{A}$ è una matrice $m \times n$, $x = (x_1, x_2, ..., x_n)'$ è un vettore di n incognite e $b = (b_1, b_2, ..., b_m)'$ è un vettore di m termini noti. Si osservi che le m righe e n colonne corrispondono alle m equazioni e n incognite del sistema.

Un sistema lineare può avere soluzioni (sistema possibile) o non averne (sistema impossibile). Se la soluzione $x \in \mathbf{R}^n$ è unica il sistema di dice *determinato*; altrimenti si dice *indeterminato*.

Un sistema indeterminato ha infinite soluzioni.

Esercizio 1. Dimostrate che un sistema indeterminato non può avere esattamente due soluzioni. Può averne esattamente tre o quattro? [Assumete che x, y siano due soluzioni distinte. Mostrate che anche $(x+y)/2$ è una soluzione. Mostrate poi che $\alpha x + (1-\alpha)y$ è una soluzione. Come potete usare questi conti per concludere?]

I seguenti esempi mostrano i vari casi possibili

Esempio 1. Il sistema

$$\begin{cases} 100x + 110y = 110 \\ 100x + 90y = 100 \end{cases}$$

può essere scritto come $\mathbf{A}x = b$ con

$$A = \begin{pmatrix} 100 & 110 \\ 100 & 90 \end{pmatrix}, b = \begin{pmatrix} 110 \\ 100 \end{pmatrix}.$$

Controllate che l'unica soluzione è $x = (0.55, 0.50)'$.

Esempio 2. Considerate i due sistemi "quadrati" che seguono

$$A = \begin{pmatrix} 100 & 110 & 120 \\ 100 & 100 & 100 \\ 100 & 90 & 80 \end{pmatrix}, b_1 = (110, 105, 100)', b_2 = (110, 110, 100)'.$$

Potete verificare che la soluzione di $\mathbf{A}x = b_1$ è $x_1 = (0.625, 0.350, 0.075)'$. Il secondo sistema invece è impossibile: non esiste x_2 tale che $\mathbf{A}x_2 = b_2$.

È interessante capire quante soluzioni ha il primo sistema: anche $x_3 = (-0.375, 2.350, -0.925)'$ è una soluzione. E fanno due! Ma allora le soluzioni sono infinite e sarebbe bello poterle scrivere tutte.

Esempio 3. Modifichiamo la matrice $\mathbf{A}$ dell'esempio precedente come segue:

$$\begin{pmatrix} 100 & 110 \\ 100 & 100 \\ 100 & 90 \end{pmatrix}$$

e sia $b = (100, 100, 110)'$. Il sistema "rettangolare" risultante non ha soluzioni.

Rivedete i concetti di determinante, rango, dipendenza ed indipendenza lineare. I seguenti teoremi descrivono le soluzioni dei sistemi lineari.

Teorema 1 (Rouchè-Capelli)**.** *Il sistema lineare* $\mathbf{A}x = b$ *ammette soluzioni (almeno una, cioè non è impossibile) se e solo se i ranghi delle sue matrici incompleta e completa sono uguali:* $r(\mathbf{A}) = r(\mathbf{A}|b) = r$*.*

Quando il sistema ammette soluzioni, queste dipendono da $n - r$ *parametri liberi ("numero d'incognite - rango comune" parametri liberi).*

Teorema 2 (Cramer)**.** *Se* $\boldsymbol{A}$ *è quadrata e non singolare (*$\det \mathbf{A} \neq 0$*), il sistema lineare* $\mathbf{A}x = b$ *ha un'unica soluzione (per ogni scelta di* b*). Inoltre, la soluzione può essere scritta come segue. Detta* $\mathbf{A}_j$ *la matrice ottenuta sostituendo la* j*-esima colonna di* $\mathbf{A}$ *con* b*, il* j*-esimo elemento della soluzione è*

$$x_j = \frac{\det \mathbf{A}_j}{\det \mathbf{A}} \text{ per } j = 1, ..., n.$$

Teorema 3. *Tutte le soluzioni del sistema* $\mathbf{A}x = b$ *si possono scrivere come*

$$x = x_0 + y,$$

in cui x_0 *è una soluzione, detta particolare, e* y *risolve il sistema omogeneo associato* $\mathbf{A}y = \mathbf{0}$*.*

Il teorema precedente dice che tutte le soluzioni di un sistema si possono scrivere come somma di una specifica soluzione del sistema e di tutte le soluzioni del sistema omogeneo.

2 Due esempi lavorati

2.1 Il mio sistema preferito

Considerate il sistema $\mathbf{A}x = b$

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix},$$

in cui, con abuso di notazione, denotiamo il vettore di soluzioni con $(x, y, z)'$.

1. Osservate che il vettore b coincide con la prima colonna e, quindi, una soluzione è (1,0,0). Il sistema è possibile.

2. Applicando il teorema di RC otteniamo

```
> A <- matrix(1:9,3,3,byrow=T)
> b <- c(1,4,7)
> qr(A)$rank # rango di A

[1] 2
```

```
> Ab <- cbind(A,b) # matrice estesa
> qr(Ab)$rank
[1] 2
```

I due rangi sono eguali, $r = 2$ e il sistema ammette $\infty^{3-2} = \infty^1$ soluzioni, infinite soluzioni dipendenti da un unico parametro libero.

3. Poiché $\det(\mathbf{A}) = 0$ non è possibile usare il comando `solve`. Sapendo che c'è una soluzione si può ricorrere al comando `ginv`, generalized inverse, della libreria `MASS` per ottenere una soluzione:

```
> require(MASS) # carica la libreria MASS
> source("mmdd14.R") # carica funzioni del corso
> x <- ginv(A) %*% b # pseudo inversa per b
> x # questa è la soluzione
           [,1]
[1,]  0.8333333
[2,]  0.3333333
[3,] -0.1666667
> A %*% x # e Ax produce b
     [,1]
[1,]    1
[2,]    4
[3,]    7
```

4. Il sistema può essere risolto a mano

$$\begin{pmatrix} 1 & 2 & 3 & 1 \\ 4 & 5 & 6 & 4 \\ 7 & 8 & 9 & 7 \end{pmatrix} \begin{matrix} \\ \text{II} \leftarrow \text{II - 4I} \\ \text{III} \leftarrow \text{III-7I} \end{matrix} \begin{pmatrix} 1 & 2 & 3 & 1 \\ 0 & -3 & -6 & 0 \\ 0 & -6 & -12 & 0 \end{pmatrix} \text{III} \leftarrow \text{III - 2II} \begin{pmatrix} \boxed{1} & 2 & 3 & 1 \\ 0 & \boxed{\text{-3}} & -6 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Le variabili x e y, corrispondenti ai pivot evidenziati vengono lasciate a sx; z è priva di pivot, viene spostata a dx e sarà il parametro libero; la terza equazione $(0 = 0)$ può essere cancellata:

$$\begin{cases} x + 2y = 1 - 3z \\ -3y = 0 + 6z \end{cases},$$

da cui $y = -2z$ e, sostituendo nella prima equazione, $x = 1 - 3z + 4z = 1 + z$. Ricordate che, semplicemente, z resta z cioè $z = z$. La soluzione generale del sistema è

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 + z \\ -2z \\ z \end{pmatrix}, \text{ per qualsiasi valore di } z.$$

5. Volendo scrivere tutte le soluzioni come somma di una soluzione particolare e delle soluzioni del sistema omogeneo si può procedere in due modi. Primo, si risolve $\mathbf{A}x = \mathbf{0}$ ottenendo $x = z, y = -2z, z = z$, verificatelo; si prende una soluzione particolare dal punto precedente (una qualsiasi), ad esempio con $z = 1$ si ottiene $x = 2, y = -2, z = 1$.

   (Tutte) le soluzioni sono

$$\begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix} + \begin{pmatrix} z \\ -2z \\ z \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix} + z \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$$

   Il secondo modo consiste nello scomporre le soluzioni date al punto precedente in modo da evidenziare una soluzione (costante) particolare e una parte che dipende da un parametro (o dai parametri), in questo caso z:

$$\begin{pmatrix} 1+z \\ -2z \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} z \\ -2z \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + z \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}.$$

   Osservate che il vettore $(1, -2, 1)$ è soluzione del sistema omogeneo.

6. Nelle righe precedenti le soluzioni sono scritte in modi diversi. La cosa vi preoccupa? Dipende: si, se le cose sono diverse; no, se in realtà l'insieme delle soluzioni è lo stesso. Verifichiamo.

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + z \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} + (z-1) \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix} + z' \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}.$$

   I due insiemi coincidono e cambia di uno il solo parametro libero z. Ad esempio, usando la rappresentazione

$$\begin{pmatrix} 2 \\ -2 \\ 1 \end{pmatrix} + z_1 \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$$

   con $z_1 = 2$ si ottiene la soluzione $x = 4, y = -6, z = 3$. Se si usa invece l'altra rappresentazione

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + z_2 \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix},$$

   si può porre $z_2 = 3$ per ottenere $x = 4, y = -6, z = 3$ che è esattamente la stessa soluzione di prima. Se volete, le soluzioni si possono scrivere in due modi diversi ma sono le stesse. Per essere precisi, i modi per scriverle sono infiniti ma si tratta sempre della "stessa roba".

7. Come si usa `R` per trovare una soluzione? Lo abbiamo già visto al punto 3. Come si trovano tutte le soluzioni? Determiniamo tutte le soluzioni del sistema omogeneo $\mathbf{A}x = \mathbf{0}$:

```
> hom <- ker(A)
> hom

           [,1]
[1,]  0.4082483
[2,] -0.8164966
[3,]  0.4082483

> k <- hom[,1]
> A %*% k

             [,1]
[1,] 0.000000e+00
[2,] 4.440892e-16
[3,] 4.440892e-16
```

Come si vede, `k` contiene è il vettore che risolve il sistema omogeneo e, quindi, $\mathbf{A}k = \mathbf{0}$. Le soluzioni del sistema omogeneo sono i vettori zk e tutte le soluzioni del sistema si possono scrivere come

$$\begin{pmatrix} 0.8333333 \\ 0.3333333 \\ -0.1666667 \end{pmatrix} + z \begin{pmatrix} 0.4082483 \\ -0.8164966 \\ 0.4082483 \end{pmatrix}.$$

Ahi, ahi... un altro modo per scrivere le soluzioni! Non la tiro lunga ma sono ancora le stesse soluzioni di prima. Prendete $z = 7.756718$ per ottenere

$$\begin{pmatrix} 0.8333333 \\ 0.3333333 \\ -0.1666667 \end{pmatrix} + 7.756718 \begin{pmatrix} 0.4082483 \\ -0.8164966 \\ 0.4082483 \end{pmatrix} = \begin{pmatrix} 0.8333333 \\ 0.3333333 \\ -0.1666667 \end{pmatrix} + \begin{pmatrix} 3.166667 \\ -6.333334 \\ 3.166667 \end{pmatrix} = \begin{pmatrix} 4 \\ -6 \\ 3 \end{pmatrix}$$

Le cose veramente importanti sono due:

- `R` può "calcolare" tutte le soluzioni ma serve sapere quel che si fa e, in particolare, trovare tutte le soluzioni del sistema omogeneo. La difficoltà di scrittura non è artificiale: infatti non è banale rappresentare infinite soluzioni dipendenti da un parametro libero che può assumere qualsiasi valore.
- Le soluzioni si possono rappresentare in molti modi, noi ne abbiamo visti tre ma sono infiniti. Ricordo che questa molteplicità è un vantaggio, non una "sfiga": spesso i problemi sembrano insormontabili se visti in un modo e ma risultano attaccabili se visti (cioè rappresentati) in modi diversi...

8. Trovo interessante riesaminare quanto fatto senza il "disturbo" dei numeri decimali. La soluzione trovata al punto 3 è

$$(x, y, z)' = \left(\frac{25}{30}, \frac{1}{3}, -\frac{1}{6}\right)',$$

il vettore `k` è

$$\begin{pmatrix} \frac{1}{\sqrt{6}} \\ -\frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{pmatrix}$$

e 7.756718=$19/\sqrt{6}$. Potete rifare i conti su carta senza digitare tutti i decimali. Osservate che il `k` calcolato da `R` è la versione decimale di un vettore in cui si riconosce chiaramente, a meno del fattore $\sqrt{6}$, il (1,-2,1) che avevamo trovato a mano fra le soluzioni del sistema omogeneo.

9. Siamo alla fine del primo esempio lavorato e concludiamo con un'ultima idea. Visto che abbiamo tante soluzioni (per di più rappresentabili in tanti modi) possiamo chiederci quale fra le soluzioni è migliore rispetto a un criterio arbitrario. In altre parole, avendo la possibilità di scegliere fra tante soluzioni del sistema, potremmo voler selezionare quella che gode di proprietà aggiuntive, che ci fanno comodo in qualche senso.

   A titolo di esempio, quale fra i vettori soluzioni è il più corto? Vedremo in seguito che questa domanda non è priva di senso finanziario. Usiamo la rappresentazione più comoda, $x = (1 + z, -2z, z)$, trovata al punto 4. La lunghezza di x è

$$\sqrt{(1+z)^2 + (-2z)^2 + z^2} = \sqrt{6z^2 + 2z + 1}.$$

   Minimizziamo il radicando, derivandolo rispetto a z e annullandolo: da $12z + 2 = 0$ segue che il minimo è raggiunto quando $z = -1/6$. Ne segue che il vettore soluzione con lunghezza minima è

$$\begin{pmatrix} 1 - \frac{1}{6} \\ \frac{2}{6} \\ -\frac{1}{6} \end{pmatrix} = \begin{pmatrix} \frac{25}{30} \\ \frac{1}{3} \\ -\frac{1}{6} \end{pmatrix}$$

   Toh... la soluzione di minima distanza, quella che si discosta meno dall'origine, è proprio la soluzione prodotta al punto 3 usando `ginv`. Non è una coincidenza e vedremo in seguito perché. Naturalmente, potremmo essere interessati a soluzioni che ottimizzano altri obbiettivi diversi dalla lunghezza, prossimamente su questi schermi!

2.2 Un sistema strano?

Considerate il sistema $\mathbf{A}x = b$

$$\begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix},$$

in cui, con abuso di notazione, denotiamo il vettore di soluzioni con $(x, y, z)'$.

1. Osservate che le tre equazioni sono in realtà una sola: la seconda infatti è $2(x+y+z) = 2 \cdot 2$, due volte per la prima e la terza è $3(x + y + z) = 3 \cdot 2$, tre volte per la prima. In un certo senso, è *come se* il sistema avesse una sola equazione e tre incognite.

2. Teorema di RC: i ranghi si possono vedere anche ad occhio ma usiamo `R`.

```
> A <- matrix(1:3,3,3)
> b <- c(2,4,6)
> qr(A)$rank

[1] 1

> Ab <- cbind(A,b)
> qr(Ab)$rank

[1] 1
```

   Il comune rango r vale 1, che è esattamente il numero di equazioni "vere" come appena visto. Il sistema ammette $\infty^{3-1} = \infty^2$ soluzioni, cioè le soluzioni dipendono da due parametri liberi.

3. `R` può determinare una soluzione con

```
> x <- ginv(A) %*% b
> x

          [,1]
[1,] 0.6666667
[2,] 0.6666667
[3,] 0.6666667
```

   La soluzione trovata è $(2/3, 2/3, 2/3)$. Le soluzioni (questa volta tutte) si possono trovare anche a mano: cancelliamo la seconda e terza riga, che non servono, e risolviamo

$$x + y + z = 2, \text{ cioè } x = 2 - y - z.$$

   x dipende da y e da z che sono libere. Una soluzione si ottiene ponendo, ad esempio $y = z = 0$ che produce $x = 2$: in termini vettoriali $x = (2, 0, 0)'$. Tutte le soluzioni sono date da

$$\begin{pmatrix} 2 - y - z \\ y \\ z \end{pmatrix}, \text{ per ogni } y \text{ e ogni } z.$$

4. Possiamo scrivere tutte le soluzioni come somma di una soluzione particolare e delle soluzioni del sistema omogeneo associato:

$$\begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + y \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} + z \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

5. `R` rappresenta le soluzioni del sistema associato in modo diverso.

```
> hom <- ker(A)
> hom

           [,1]       [,2]
[1,] -0.5773503 -0.5773503
[2,]  0.7886751 -0.2113249
[3,] -0.2113249  0.7886751

> k1 <- hom[,1]
> k2 <- hom[,2]
> A %*% hom # zero!

              [,1]          [,2]
[1,] -1.665335e-16 -2.220446e-16
[2,] -3.330669e-16 -4.440892e-16
[3,] -8.881784e-16 -8.881784e-16
```

I vettori `k1` e `k2` generano tutte le soluzioni del sistema omogeneo (non controlliamo che si tratta di una rappresentazione alternativa ma equivalente a quanto visto prima, fatelo come esercizio). Ne segue che le soluzioni del sistema si possono scrivere come "soluzione particolare" + soluzione sistema omogeneo:

$$\begin{pmatrix} 0.6666667 \\ 0.6666667 \\ 0.6666667 \end{pmatrix} + y \begin{pmatrix} -0.5773503 \\ 0.7886751 \\ -0.2113249 \end{pmatrix} + z \begin{pmatrix} -0.5773503 \\ -0.2113249 \\ 0.7886751 \end{pmatrix} \text{ per ogni } y \text{ e per ogni } z.$$

3 Inversa e pseudo inversa

In questa sezione vedremo che l'inversa di una matrice quadrata non singolare si può generalizzare, definendo una *inversa generalizzata* o *pseudo-inversa di Moore-Penrose*. Rivedete, se necessario, il concetto di inversa.

3.1 C'è un sistema... per risolvere i sistemi?

Considerate questo sistema di due equazioni in due incognite:

$$\begin{cases} x+y &= 3 \\ x-y &= 1 \end{cases}.$$

La soluzione $x = 2, y = 1$ si può determinare in molti modi. Uno in particolare c'interessa ora: l'uso della matrice inversa. Riscritto in forma matriciale il sistema diventa

$$\mathbf{A}x = \begin{pmatrix} 3 \\ 1 \end{pmatrix},$$

e la soluzione è $\mathbf{A}^{-1}(3,1)'$ con inversa

$$\mathbf{A}^{-1} = -\frac{1}{2}\begin{pmatrix} -1 & -1 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix}.$$

Osservate che l'unica soluzione si trova moltiplicando per l'unica inversa $\mathbf{A}^{-1}$ della matrice A. In un mondo perfetto tutto finisce bene, ma cosa succede se aggiungiamo una incognita o un'ulteriore equazione?

3.2 Troppa grazia

Aggiungere una incognita produce ad esempio il sistema

$$\begin{cases} x+y+z &= 3 \\ x-y+z &= 1 \end{cases}.$$

Determinare la soluzione non è più possibile: intanto non esiste l'inversa di

$$\begin{pmatrix} 1 & 1 & \boxed{1} \\ 1 & -1 & \boxed{1} \end{pmatrix}$$

per "colpa" della nuova colonna i cui elementi sono evidenziati. In secondo luogo, un po' di riflessione e di conti mostrano che ci sono infinite soluzioni. Verificate che le soluzioni sono $x = 2 - t, y = 1, z = t$ con $t \in \mathbf{R}$. In un certo senso, ci sono troppe soluzioni per poter usare l'inversa: troppa grazia!

Domanda intelligente: e possibile trovare un'inversa che "funzioni" anche in questo caso?

3.3 Ai confini dell'impossibile

Proviamo ora ad aggiungere un'equazione al sistema iniziale:

$$\begin{cases} x+y &= 3 \\ x-y &= 1 \\ x-2y &= 2 \end{cases}.$$

Adesso le cose vanno male. In generale, i sistemi con più equazioni che incognite non hanno soluzioni (verificate che è proprio così per questo esempio). L'impossibilità di risolvere il sistema è confermata dal fatto che l'inversa della matrice

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \\ \boxed{1} & \boxed{\text{-2}} \end{pmatrix}$$

non esiste (proprio perché abbiamo aggiunto la riga evidenziata). Eppure sarebbe bello avere una specie d'inversa anche in questo caso.

Cosa può voler dire "risolvere" un sistema impossibile? E cosa può voler dire "inversa" per la matrice dei coefficienti del sistema?

3.4 Vecchi problemi, nuove tecniche

Gli esempi che abbiamo visto mostrano che anche un semplice sistema lineare può non essere banalmente risolubile. Una (bella) idea è quella di estendere il concetto di soluzione.

1. Se il sistema ammette un'unica soluzione, non ci sono problemi.

2. Se il sistema ammette infinite soluzioni, potremmo scegliere quella con i coefficienti "più piccoli" (per essere più precisi, quella che dista meno dall'origine).

3. Se il sistema non ha soluzioni, nessun vettore risolve il sistema. Ma potremmo scegliere quello che minimizza a distanza fra $\mathbf{A}x$ e b (in senso minimi quadrati).

Questa impostazione ha il vantaggio che ogni sistema avrebbe un'unica "soluzione".

Potremmo anche ragionare in un altro modo: abbiamo problemi perché non esiste la matrice inversa e quindi non possiamo calcolare in tutti i casi $\mathbf{A}^{-1}b$. Ma allora potremmo cercare di definire una *pseudo-inversa* per A anche quando non esiste $\mathbf{A}^{-1}$. Se avessimo a disposizione una pseudo-inversa, diciamo $\mathbf{A}^{+}$, potremmo sempre calcolare $\mathbf{A}^{+}b$ e "ottenere una soluzione".

In effetti vedremo che le due idee appena esposte (generalizzare la soluzione di un sistema anche quando ci sono infinite o nessuna soluzione e generalizzare l'inversa) sono equivalenti.

3.5 Pseudo-inversa di Moore-Penrose

Dire che non esiste l'inversa significa dire che non c'è una matrice $\mathbf{A}^{-1}$ tale che

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = I,$$

dove I è la matrice identica.

Definizione. Data una matrice $\mathbf{A}_{m\times n}$, si dice pseudo-inversa $\mathbf{A}^{+}$ di $\mathbf{A}$ una matrice $n \times m$ tale che

1. $\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+$;
2. $\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$;
3. $\mathbf{A}\mathbf{A}^+$ è simmetrica;
4. $\mathbf{A}^+\mathbf{A}$ è simmetrica.

Vedremo in classe alcuni dei motivi per cui $\mathbf{A}^+$ somiglia molto all'inversa. I seguenti due teoremi mostrano perché il concetto di pseudo-inversa sia fondamentale.

Teorema. [*Inversa e pseudo-inversa*] Se $\mathbf{A}$ è invertibile (cioè se esiste $\mathbf{A}^{-1}$), allora l'inversa e la pseudo-inversa coincidono:

$$\mathbf{A}^{-1} = \mathbf{A}^+.$$

Teorema. [*Pseudo-inversa e sistemi*] Considerate il sistema lineare $\mathbf{A}\mathbf{x} = \mathbf{b}$. Allora

1. Se la soluzione è unica si tratta di $\mathbf{x} = \mathbf{A}^+\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}$.
2. Se ci sono infinite soluzioni, $\mathbf{x} = \mathbf{A}^+\mathbf{b}$ è la soluzione con norma minima (cioè che dista meno dall'origine). In formule

   $$||\mathbf{A}^+\mathbf{b}||_2 \text{ è minima.}$$

3. Se non ci sono soluzioni, $\mathbf{x} = \mathbf{A}^+\mathbf{b}$ è il vettore che rende minima la distanza fra $\mathbf{A}\mathbf{x}$ e $\mathbf{b}$. In formule

   $$||\mathbf{A}^+\mathbf{x} - \mathbf{b}||_2 \text{ è minima.}$$

4 Calcolo della pseudo-inversa di MP

Caveat: questa sezione va brutalmente al punto senza nemmeno sfiorare molte questioni interessanti. Chi fosse interessato può consultare `http://en.wikipedia.org/wiki/Moore%E2%80%93Penrose_pseudoinverse`

La pseudo-inversa si può calcolare utilizzando la funzione `ginv` della libreria `MASS`, come già visto in precedenza. Può essere utile ripercorrere ora gli esempi già svolti. `ginv` sta per "generalized inverse" e `ginv(A)` calcola $\mathbf{A}^+$ per qualsiasi matrice $\mathbf{A}$.

Esempio 4. Considerate le matrici

$$\mathbf{A}_1 = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}, \mathbf{A}_2 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \mathbf{A}_3 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

Nel primo caso $\mathbf{A}^{-1} = \mathbf{A}^+$:

```
> A1 <- matrix(c(1,1,1,2),2,2)
> solve(A1)
     [,1] [,2]
[1,]    2   -1
[2,]   -1    1
> ginv(A1)
     [,1] [,2]
[1,]    2   -1
[2,]   -1    1
```

$\mathbf{A}_2$ è clamorosamente non invertibile e l'inversa non esiste (provate con `solve(A2)`, errore!) Ma la sua bella pseudo-inversa di Moore-Penrose si calcola in un attimo:

```
> A2 <- matrix(c(1,1,1,1),2,2)
> ginv(A2)
     [,1] [,2]
[1,] 0.25 0.25
[2,] 0.25 0.25
```

La mia matrice preferita $\mathbf{A}_3$ non ammette inversa, dato che $\det \mathbf{A}_3 = 0$ ma $\mathbf{A}_3^+$ è

```
> A3 <- matrix(1:9,3,3,byrow=T)
> Apiu <- ginv(A3)
> Apiu
            [,1]          [,2]
[1,] -0.63888889 -1.666667e-01
[2,] -0.05555556  3.469447e-17
[3,]  0.52777778  1.666667e-01
            [,3]
[1,]  0.30555556
[2,]  0.05555556
[3,] -0.19444444
```

In questo caso verifichiamo le 4 proprietà di $\mathbf{A}^+$:

1. $\mathbf{A}^+\mathbf{A}\mathbf{A}^+ = \mathbf{A}^+$

```
> Apiu %*% A3 %*% Apiu
            [,1]          [,2]
[1,] -0.63888889 -1.666667e-01
[2,] -0.05555556  4.163336e-17
[3,]  0.52777778  1.666667e-01
            [,3]
[1,]  0.30555556
[2,]  0.05555556
[3,] -0.19444444
```

2. $\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$:

```
> A3 %*% Apiu %*% A3
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

3. $\mathbf{A}^+\mathbf{A}$ simmetrica:

```
> Apiu %*% A3
           [,1]      [,2]       [,3]
[1,]  0.8333333 0.3333333 -0.1666667
[2,]  0.3333333 0.3333333  0.3333333
[3,] -0.1666667 0.3333333  0.8333333
```

4. $\mathbf{A}\mathbf{A}^+$ simmetrica:

```
> A3 %*% Apiu
           [,1]      [,2]       [,3]
[1,]  0.8333333 0.3333333 -0.1666667
[2,]  0.3333333 0.3333333  0.3333333
[3,] -0.1666667 0.3333333  0.8333333
```

5 Soluzione di sistemi e pseudo-inversa

Teorema 4 (da Wikipedia)**.** *If the linear system*

$$\mathbf{A}x = b$$

has any solutions, they are all given by

$$x = \mathbf{A}^+ b + [\mathbf{I} - \mathbf{A}^+\mathbf{A}]w$$

for arbitrary vector w. Solution(s) exist if and only if $\boldsymbol{A}\boldsymbol{A}^+ b = b$*. If the latter holds, then the solution is unique if and only if* $\boldsymbol{A}$ *has full column rank, in which case* $[\mathbf{I} - \mathbf{A}^+\mathbf{A}]$ *is a zero matrix.*

Spendo qualche parola per spiegare perché questo è un teorema bellissimo.

1. Vale sempre, per qualsiasi sistema con soluzioni, senza se e senza ma (singolare o meno, quadrato o rettangolare...)
2. C'è bisogno di calcolare solo $\mathbf{A}^+$ (e fare altri quattro conti): niente ranghi, niente soluzioni del sistema omogeneo associato, niente inverse.
3. Le soluzioni si descrivono molto semplicemente tramite il vettore libero $w \in \mathbf{R}^n$.

Come esempio torniamo al sistema $\mathbf{A}x = b$

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix},$$

```
> A <- matrix(1:9,3,3,byrow=T)
> b <- c(1,4,7)
> Apiu <- ginv(A)
> x <- Apiu %*% b
> x
           [,1]
[1,]  0.8333333
[2,]  0.3333333
[3,] -0.1666667
> A %*% x
     [,1]
[1,]    1
[2,]    4
[3,]    7
```

Finora abbiamo calcolata una soluzione x e verificato che $\mathbf{A}x = b$. Procediamo col calcolo di $[\mathbf{I} - \mathbf{A}^+\mathbf{A}]$, chiamata per semplicità $\mathbf{B}$:

```
> B <- diag(c(1,1,1))-Apiu %*% A
> B
           [,1]       [,2]       [,3]
[1,]  0.1666667 -0.3333333  0.1666667
[2,] -0.3333333  0.6666667 -0.3333333
[3,]  0.1666667 -0.3333333  0.1666667
```

Tutte le soluzioni del sistema si scrivono come

$$\begin{pmatrix} 0.8333333 \\ 0.3333333 \\ -0.1666667 \end{pmatrix} + \begin{pmatrix} 0.1666667 & -0.3333333 & 0.1666667 \\ -0.3333333 & 0.6666667 & -0.3333333 \\ 0.1666667 & -0.3333333 & 0.1666667 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix},$$

per qualsiasi x, y, z. Spero che siate contenti: è solo un esempio ma funziona sempre (per il teorema di prima!)

Esercizio 2. Il lettore attento (o, per meglio dire, simpaticamente "fetente") potrebbe e dovrebbe però chiedersi come mai qui ci sono *tre* parametri liberi mentre abbiamo visto che le soluzioni dipendono da uno *solo*. È possibile?

Trovate la soluzione all'inghippo osservando bene `B` che ha una struttura particolare, dato che le sue righe sono uguali a meno di costanti.

Esercizio 3. Considerate il sistema

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & \mathbf{10} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix},$$

facendo attenzione al **10** scritto al posto del 9, e risolvetelo usando il teorema che consente di scrivere tutte le soluzioni con la pseudo-inversa. Il sistema ha soluzioni? Da quante parametri dipendono? Perché?

Esercizio 4. Adesso sapete tutto sui sistemi e sapete molto su `R`. La cautela però è d'obbligo. Leggete queste note che sottolineano la necessità di verificare *sempre* le soluzioni numeriche di sistemi generali (è scritto per MATLAB, ma il concetto è lo stesso): `http://see.stanford.edu/materials/lsoeldsee263/Additional5-sle_matlab.pdf`

6 State preference model

This material is based on notes written for the course "Computational tools for economics and management", 2013.

What you know about systems of linear equations can be used to understand a beautiful model of a financial market, in which assets are traded in two periods of time. The model, known as "State preference model" or "Asset pricing model", is a powerful tool to understand important ideas related to *portfolios, replication of assets, risk-hedging* and *arbitrage*.

If you are interested in the topic you should read `http://economia.unipr.it/DOCENTI/FAVERO/docs/files/CastBreve.pdf`. The text is sparkling, insightful and written in Italian.

I will assume you know how to solve the system $A\mathbf{x} = \mathbf{b}$ using the Rouché-Capelli theorem. A video in Italian to review the material is at `http://www.youtube.com/watch?v=SLqivS2CTTA`.

The model is as follows:

1. There are two periods, today and tomorrow, corresponding to $t = 0$ and $t = 1$. You can buy or (short-)sell assets in $t = 0$ and will get the payoff in $t = 1$.

2. There is uncertainty over the future and m possible states can materialize tomorrow. We often think that *nature will pick the future state*, as opposed to agents that do not have the knowledge to predict what lies ahead. Hence, depending on the state of nature at $t = 1$, the assets will produce different payoffs.

3. There are n available assets. Each of them is a vector of future payoffs y (at $t = 1$), which can be purchased/selled at $t = 0$ for a price π. As there are n assets, it is convenient to describe this market using a payoff matrix collecting all the column vectors $y_1, y_2, ..., y_n$:

$$\mathbf{Y} = \left(\begin{pmatrix} \vdots \\ y_1 \\ \vdots \end{pmatrix} \quad \begin{pmatrix} \vdots \\ y_2 \\ \vdots \end{pmatrix} \quad \vdots \quad \begin{pmatrix} \vdots \\ y_n \\ \vdots \end{pmatrix} \right),$$

with the agreement that π_1 is the cost of y_1, π_2 is the cost of y_2 and so on. We assume all the prices are collected in the vector $\pi = (\pi_1, ..., \pi_n)'$.

The description of the market is over and an example will help. Assume that there are $m = 3$ possible states of nature tomorrow and $n = 3$ assets:

$$Y = \begin{pmatrix} & \text{Bankaccount} & \text{Stock} & \text{Riskystock} \\ \text{Good} & 104 & 108 & 112 \\ \text{Fair} & 104 & 102 & 106 \\ \text{Gloomy} & 104 & 100 & 93 \end{pmatrix}$$

Assume that all assets cost 100, i.e., $\pi_1 = \pi_2 = \pi_3 = 100$. Rows are relative to future states and you can think, say, that the first row is relative to the payoffs that you will get if tomorrow the economy will be good. In this case, the payoffs are high and you will cash 104 if you are the holder of the first asset, 108 if you are the the holder of the second asset and so on. Clearly, if the gloomy state is the prevailing one tomorrow, the owner of the third asset will get only 93: taking into account that it was paid 100, it's a bad loss!

Columns should be thought as assets. The first column y_1 always pay 104 and, hence, it's like a (safe) bank account that pays the same amount regardless of the state. The other two columns have fluctuating payoffs and behave like stocks that depend on the future state of the economy. The third column is dubbed "risky stock" because payoffs are much more variable than the second column.

Portfolio: a vector $x = (x_1, x_2, ..., x_n)' \in \mathrm{R}^n$ of the quantities held of each asset. In the financial jargon, we refer sometimes to x as "the weights of the portfolio". The payoff of a portfolio x is $\mathbf{Y}x$, as this product is exactly a weighted sum of the columns of A, i.e., a weighted sum of the assets.

The cost of a portfolio is the scalar product of π and x, $\pi' \cdot x = \pi_1 x_1 + \pi_2 x_2 + ... + \pi_n x_n$. This is a simple computation of the sum of the prices times the quantities that are held.

Replication: we say that a vector b can be replicated of there is a portfolio that has the same payoffs. In other words, b can be replicated if there is x such that $Ax = b$.

Observe that this concept is tightly related to the solution of a linear system. This is the reason why we devoted a considerable effort in the past to learn the theory and the practice of the solution of linear systems.

Arbitrage: this is a situation in which there is a portfolio $\mathbf{x}$ with null payoff and non-null price.[1]

An arbitrage is a *free-lunch*: you sell for a positive price something that always pays zero and, hence, should cost zero. It's a no-loss gamble that can be sold for a profit but will produce no payoff for the buyer.

Consider the financial market with $\mathbf{Y}$ and π described above: the portfolio $x = (1/3, 1/3, 1/3)'$ has a future payoff[2] of

$$\mathbf{Y}x = \begin{pmatrix} 104 & 108 & 112 \\ 104 & 102 & 106 \\ 104 & 100 & 93 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} 108 \\ 104 \\ 99 \end{pmatrix}.$$

The cost of the portfolio is

$$\pi' \cdot x = \pi_1 x_1 + \pi_2 x_2 + \pi_3 x_3 = 100\frac{1}{3} + 100\frac{1}{3} + 100\frac{1}{3} = 100.$$

So, if you pay 100 for the portfolio that is mixing in equal proportions the original three assets, you can get $(108, 104, 99)'$. Let's notice in passing the power of diversification: mixing the assets has reduced the maximum losses (and the gains) in every state of the world.

Can the asset $b = (108, 104, 99)'$ be replicated? Yes, we have just seen that there is a portfolio, namely $x = (1/3, 1/3, 1/3)'$ such that $\mathbf{Y}x = b$ and the cost of the replicating portfolio is 100.

Can the asset $b = (112, 106, 104)'$ be replicated? We look for the solution x of the system $\mathbf{Y}x = b$:

```
> Y <- matrix(c(104,104,104,108,102,100,112,106,92),3,3)
> b <- c(112,106,104)
> x <- solve(Y,b)
> x
[1] 0.03846154 1.00000000 0.00000000
```

The answer is positive: $\mathbf{b}$ can be replicated with $x_1 \approx 0.038$ of the first asset (putting a small amount in the bank account) and $x_2 = 1$ of the second asset. The cost of replication (i.e., the cost of the replicating portfolio) is

[1] Arbitrage is a serious topic and, here, I'm just scratching the surface. We only cover one special type of arbitrage, called *arbitrage of fist type*, and much more can (and should) be said on this matter, see `http://economia.unipr.it/DOCENTI/FAVERO/docs/files/CastBreve.pdf`.

[2] This is a very important footnote: the product of a matrix times a vector can be thought as a way to produce a new column forming a linear combination of the columns of the matrix. Hence, $\mathbf{Y}(1/3, 1/3, 1/3)'$ gives

$$\frac{1}{3}\begin{pmatrix} 104 \\ 104 \\ 104 \end{pmatrix} + \frac{1}{3}\begin{pmatrix} 108 \\ 102 \\ 100 \end{pmatrix} + \frac{1}{3}\begin{pmatrix} 112 \\ 106 \\ 93 \end{pmatrix}.$$

```
> sum(c(100,100,100)*x)
[1] 103.8462
```

This is an interesting case as b keeps the better payoffs coming from the risky stock but avoids the heavy losses that would be incurred if the risky stock is bought. Don't you think this is nice? Mmhh... where is the trick? Well, on the one hand there is no trick: this can be done just solving linear systems with `R`. On the second hand, however, there are other two equally interesting ways to interpret what happened here.

1. I pay 103.8462 to get b instead of 100 to buy the risky stock. Therefore, the additional amount 3.8462 is the cost to insure my position in the bad state of the world: *I have paid 3.8462 to reduce my risk in one specific future occurrence.*

2. The portfolio x blends a stock with a bond. Often investors reduces the variability of their cashflows using the (certain) proceeds to compensate for poor results of other investments in some states. Here, if the future state is "gloomy" then the stock is making a null profit (it was paid 100, it gives 100) but the 0.03846154 units in the bank account still pays 4% so that $0.03846154 \cdot 104 = 4$ are gained and "compensate" for the outcome of the stock.

Another brave comment on case 1): this is exactly why managers are paid and planning is needed to survive bad occurences. Notice that assuming something on the states (and on the measures to mitigate adverse outcomes) is much better than assuming a loss probability out of the blue.

Esercizio 5. Can you replicate *any* b? Hint: can you solve the system $\mathbf{Y}x = b$ for any b, given this specific $\mathbf{Y}$? Can you imagine why such a market is called *complete*?

Next, we'll discuss an arbitrage arising in a slightly modified market. Let $\mathbf{Y}$ be defined as

$$\mathbf{Y}x = \begin{pmatrix} 104 & 108 & 112 \\ 104 & 102 & 106 \\ 104 & 100 & 104 \end{pmatrix},$$

and let the prices of the asset be $\pi = (100, 100, 104)'$. Observe that the third column of Y is the asset $b = (112, 106, 104)'$ that we have discussed a few lines ago and recall that b can be replicated using the first and second columns (assets) at a cost of about 103.85. Now, something interesting goes on: the third asset can be sold on the market at a price of $\pi_3 = 104$ but the same replicated asset can be obtained (mixing the first two columns) at the cost 103.85. Hence, a clever trader may start a money pump, creating replicates of b for 103.85 each and selling them immediately for 104. For each round, he will make the difference 104-103.85=0.15, regardless of the future state.

We now check the definition stated before: there should be a portfolio x with null payoff vector and strictly positive price. Consider $x = (-0.03846154, -1, 1)$, which should be easily interpreted: 0.038 units of the first asset and 1 unit of the second are bought, whereas 1 unit of the third asset (i.e., b) are sold. The payoff $\mathbf{Y}x$ and the cost of portfolio x are given by:

```
> Y <- matrix(c(104,104,104,108,102,100,112,106,104),3,3)
> x <- c(-0.03846154,-1,1)
> pai <- c(100,100,104) # uso pai perche pi=3.14... in R
> Y %*% x
          [,1]
[1,] -1.6e-07
[2,] -1.6e-07
[3,] -1.6e-07
> sum(pai*x)
[1] 0.153846
```

The trader cashes 0.15 € replicating and selling the asset and his profits can be inflated doing the same operation many times (i.e., trading large volumes). It is likely that once an arbitrage is spotted, traders will soon wipe it, as large volumes are affecting the prices and, in particular, the cost of the third asset will tend to go back 103.846, closing the opportunity for gains at no risk.

This final example illustrates how to deal with singular systems, which is "problematic" in R... and in life! Let

$$\mathbf{Y} = \begin{pmatrix} 105 & 120 & 110 \\ 105 & 110 & 102 \\ 105 & 90 & 98 \\ 105 & 80 & 90 \end{pmatrix},$$

and assume all the prices are 100, $\pi = (100, 100, 100)'$. Observe that we have $m = 4$ states of the world or rows and $n = 3$ assets or columns. Can the vector $b = (119, 119, 95, 95)'$ be replicated? Well, we just have to check whether there is a vector x such that $\mathbf{Y}x = b$. This is what you get trying to solve the problem with `solve(Y,c(119,119,95,95))`:

```
Error in solve.default(Y, c(119, 119, 95, 95), 4, 3) :
  singular matrix 'a' in 'solve'
```

The problem lies in the fact that `solve` can be used to solve square systems with an invertible matrix of coefficients. Here, we have a 4×3 matrix, which is singular as finding an inverse is impossible.

But you should already know the way out, see the previous sections. Preliminary try to figure out whether the system has a solution using the Rouché-Capelli theorem and checking the ranks of $\mathbf{Y}$ and $\mathbf{Y}|b$:

```
> Y <- matrix(c(105, 120, 110,
+               105, 110, 102,
+               105, 90, 98,
+               105, 80, 90),4,3,byrow=T)
> b <- c(119,119,95,95)
> qr(Y)$rank
```

```
[1] 3
> qr(cbind(Y,b))$rank
[1] 3
```

In this case, the ranks of the complete and incomplete matrices are equal and hence the system has $\infty^{n-r} = \infty^{3-3} = 1$ solution. This unique solution can be worked out using the generalized inverse of Y, computed using the function `ginv` of the package `MASS`.[3]

```
> require(MASS)
> giY <- ginv(Y) # compute the generalized inverse
> sol <- giY %*% b # multiply the gen inverse and b
> sol
     [,1]
[1,]  1.4
[2,]  1.6
[3,] -2.0
```

We are ready to answer: yes, indeed $(119, 119, 95, 95)'$ can be replicated by the portfolio $x = (1.4, 1.6, -2.0)'$: buying 1.4 units of the first asset, 1.6 units of the second asset and (short-)selling 2 units of the third asset.

Consider now the vector $b = (119, 118, 117, 115)'$: can it be replicated?

```
> b <- c(119,118,117,115)
> sol <- giY %*% b
> sol
           [,1]
[1,] 0.94206349
[2,] 0.01666667
[3,] 0.16666667
> Y %*% sol
       [,1]
[1,] 119.25
[2,] 117.75
[3,] 116.75
[4,] 115.25
```

We have used the `giY` matrix, which was computed before, and easily obtained `sol`. You may think that `sol` replicates b but it a look at our check shows that this is false, false, false, false. . .

[3] When you solve a system, say $A\mathbf{x} = \mathbf{b}$ and A is invertible, the solution can be obtained by $\mathbf{x} = A^{-1}\mathbf{b}$. When the inverse A^{-1} does not exist, as in this case, you can instead use the generalized inverse. In a way, *if you know that there are solutions*, you can always find one multiplying $A^{+}\mathbf{b}$, where A^{+} is the inverse (if it exists) or the generalized inverse (if the inverse doe not exist).

Indeed, a look at the ranks shows why: we already know that the rank of $\mathbf{Y}$ is 3 but the rank of the complete matrix $\mathbf{Y}|\mathbf{b}$ is 4, as seen below:

```
> qr(cbind(Y,b))
$qr

[1,] -210.0 -200.0000000 -200.0000000
[2,]    0.5  -31.6227766  -13.9140217
[3,]    0.5   -0.5270463   -3.7947332
[4,]    0.5   -0.8432740   -0.9845256
                  b
[1,] -234.5000000
[2,]   -2.8460499
[3,]   -0.6324555
[4,]    0.5000000

$rank
[1] 4

$qraux
[1] 1.500000 1.105409 1.175241 0.500000

$pivot
[1] 1 2 3 4

attr(,"class")
[1] "qr"
```

Hence, the system has no solution and b cannot be replicated.

To sum up:

1. take extra care with singular and rectangular systems.
2. find a candidate solution using `ginv` of `MASS`.
3. *always* check your result.

Esercizio 6. With the previous 4×3 matrix Y and prices, can the vector $\mathbf{b} = (119, 118, 117, 116)'$ be replicated? If so, how much does it cost if no arbitrage is assumed?

7 Un altro approccio all'arbitraggio

Da definizione, siamo in presenza di arbitraggio quando un portafogli con payoff nullo ha prezzo non nullo. I portafogli x con payoff nullo sono quelli per cui $\mathbf{Y}x = \mathbf{0}$ ma allora x

è soluzione del sistema omogeneo associato o, equivalentemente, $x \in \ker \mathbf{Y}$, cioè x sta nel nucleo di $\mathbf{Y}$.

Dato un mercato $\mathbf{Y}$ e il relativo vettore di prezzi, è quindi possibile verificare la presenza di arbitraggi, studiando i prezzi dei portafogli del nucleo. Poiché questi ultimi si possono generare usando una base del nucleo, sia $\{x_1, x_2..., x_q\}$ questa base. Se

$$\begin{aligned} \pi' x_1 &= 0, \\ \pi' x_2 &= 0, \\ &\dots, \\ \pi' x_q &= 0 \end{aligned}$$

significa che tutti i portafogli con payoff nullo hanno prezzo nullo e quindi non c'è possibilità di arbitraggio. In dettaglio questo argomento procede come segue: sia x un portafogli tale che $\mathbf{Y}x = \mathbf{0}$; allora è possibile scriverlo come combinazione lineare degli elementi della base del nucleo, cioè

$$x = \alpha_2 x_1 + \alpha_2 x_2 + ... + \alpha_q x_q,$$

per opportuni scalari $\alpha_1, ...\alpha_q$. Il prezzo di x è

$$\pi' x = \pi'(\alpha_2 x_1 + \alpha_2 x_2 + ... + \alpha_q x_q) = \alpha_1 \pi' x_1 + ... + \alpha_q \pi' x_q = 0 + ... + 0 = 0.$$

Illustriamo quanto detto con l'economia di prima: la matrice di mercato sia

$$\mathbf{Y} = \begin{pmatrix} 105 & 120 & 110 \\ 105 & 110 & 102 \\ 105 & 90 & 98 \\ 105 & 80 & 90 \end{pmatrix}, \tag{1}$$

e i prezzi $\pi = (100, 100, 100)$.

```
> Y <- matrix(c(105, 120, 110,
+               105, 110, 102,
+               105, 90, 98,
+               105, 80, 90),4,3,byrow=T)
> hom <- ker(Y)
> hom
[1,]
[2,]
[3,]
```

Il nucleo non contiene nulla e questo significa che l'unico portafogli con payoff nullo è, banalmente, il vettore nullo (tutte le quantità sono nulle) che naturalmente ha prezzo nullo.

Cambiamo ora l'economia, aggiungendo il titolo (colonna) $(119, 119, 95, 95)'$. Ricordate che questo titolo può essere replicato per mezzo degli altri tre e assumiamo che costi $\pi_4 = 100$. La matrice $\mathbf{Y}$ è ora

$$\mathbf{Y} = \begin{pmatrix} 105 & 120 & 110 & 119 \\ 105 & 110 & 102 & 119 \\ 105 & 90 & 98 & 95 \\ 105 & 80 & 90 & 95 \end{pmatrix}, \tag{2}$$

con nucleo

```
> Y <- matrix(c(105, 120, 110,119,
+               105, 110, 102,119,
+               105, 90, 98,95,
+               105, 80, 90,95),4,4,byrow=T)
> hom <- ker(Y)
> hom
           [,1]
[1,] -0.4537426
[2,] -0.5185630
[3,]  0.6482037
[4,]  0.3241019
```

Il nucleo contiene un solo vettore che ha prezzo

```
> sum(c(100,100,100,100)*hom)
[1] -7.105427e-15
```

I portafogli con payoff nullo costano zero e, quindi, non ci sono arbitraggi.

Esercizio 7. Quanti e quali sono i portafogli con payoff nulli nel caso appena completato?

Esercizio 8. Data un'economia $\mathbf{Y}, \pi$, la presenza o l'assenza di arbitraggi dipendono da come è fatta $\mathbf{Y}$, da π o da ambedue?

Esercizio 9. Considerate la $\mathbf{Y}$ definita in (2) e sia $\pi = (101, 100, 100, 100)'$ (abbiamo cioè cambiato un solo prezzo). Ci sono opportunità d'arbitraggio in quest'economia?

Esercizio 10. Modificate la $\mathbf{Y}$ in (2) scrivendo 122 al posto di 120 e sia $\pi = (100, 100, 100, 100)'$. Ci sono opportunità d'arbitraggio in quest'economia?

Esempio 5 (Arbitraggio del secondo tipo)**.** Con $\mathbf{Y}$ come in (1) e il solito $\pi = (100, 100, 100)'$ considerate il portafogli $x = (0.6, 0.4, -1)'$, ottenuto comprando 0.6 unità del primo, 0.4 unità del secondo asset vendendo (allo scoperto) una unità del terzo titolo. Il payoff è

$$\mathbf{Y} \begin{pmatrix} 0.6 \\ 0.4 \\ -1.0 \end{pmatrix} = \begin{pmatrix} 1 \\ 5 \\ 1 \\ 5 \end{pmatrix},$$

a fronte di un costo di $\pi' x = 100 \cdot 0.6 + 100 \cdot 0.4 - 100 \cdot 1 = 0$. In altre parole, il portafogli x non costa nulla (ho finanziato l'acquisto di 0.4 e 0.6 del primo e del secondo asset vendendo il terzo). Riflettete bene: non pagate nulla, ma incassate un vettore di pagamenti positivi in qualche stato del mondo (in questo caso tutti, dato che $(1, 5, 1, 5)' > (0, 0, 0, 0)'$).

Questa situazione, in cui un portafogli di costo nullo (*self-financing* in inglese) produce pagamenti positivi o nulli in tutti gli stati del mondo e strettamente positivi in almeno uno stato, è detto *arbitraggio del secondo tipo*. In altre parole, è un x che non vi costa nulla ma che non produce *mai* perdite e *può produrre* guadagni in almeno uno stato futuro (nell'esempio, in realtà, siete nel caso assai fortunato in cui guadagnate qualcosa in tutti gli stati).

8 Un terzo approccio all'arbitraggio

Come visto nell'esempio precedente, gli arbitraggi possono essere di due tipi:

1. Primo tipo: un portafogli x con payoff nullo e prezzo diverso da zero. In questo caso, posso guadagnare *con certezza* il prezzo ora a fronte di pagamenti futuri nulli. Si tratta di un guadagno certo immediato senza oneri futuri.

   Abbiamo visto che la presenza/assenza di questo tipo di arbitraggi si verifica studiando il prezzo dei vettori del nucleo di $\mathbf{Y}$: assenza di arbitraggio significa

   $$\mathbf{Y}x = \mathbf{0} \Longrightarrow \pi' x = 0.$$

2. Secondo tipo: un portafoglio x di costo nullo produce pagamenti tutti non negativi (nessuna perdita) e, in almeno uno stato, pagamenti strettamente positivi. A fronte di un costo nullo immediato non subirò perdite e guadagnerò (*forse*) un profitto in qualche stato del mondo. Notate le differenze con il caso precedente: un arbitraggio del secondo tipo si può definire probabilistico mentre uno del primo tipo è deterministico (certo).

   Formalmente, in presenza di un'opportunità di arbitraggio del secondo tipo, esiste un portafogli x tale che

   $$\pi' x = 0 \text{ e } \mathbf{Y}x \geqslant \mathbf{0},$$

   dove $a \geqslant b$ significa che le componenti di a sono $\geq$ di quelle di b e, per almeno una componente j, $a_j > b_j$.

Il seguente risultato teorico, dovuto a B. De Finetti, 1931, fornisce delle condizioni per l'assenza di arbitraggio.

Teorema 5 (De Finetti)**.** *Un'economia $\mathbf{Y}, \pi$ è priva di opportunità d'arbitraggio se esiste un vettore $p > 0$ tale che*

$$p'\mathbf{Y} = \pi'.$$

Il vettore p è detto vettore delle probabilità neutre e il teorema dice che in presenza di almeno un vettore p strettamente positivo tale che $p'\mathbf{Y} = \pi'$ non sono possibili arbitraggi né di primo né di secondo tipo. Osservate che $p'\mathbf{Y} = \pi'$ uguaglia il prodotto di una riga con una matrice a un'altra riga. Credo sia più semplice trasporre ambo i membri ottenendo il sistema $\mathbf{Y}'p = \pi$ che si può risolvere come visto nelle sezioni precedenti, a patto di ricordare di usare come matrice dei coefficients la matrice trasposta $\mathbf{Y}'$ al posto di $\mathbf{Y}$.

Per fare pratica, riprendiamo

$$\mathbf{Y} = \begin{pmatrix} 105 & 120 & 110 \\ 105 & 110 & 102 \\ 105 & 90 & 98 \\ 105 & 80 & 90 \end{pmatrix},$$

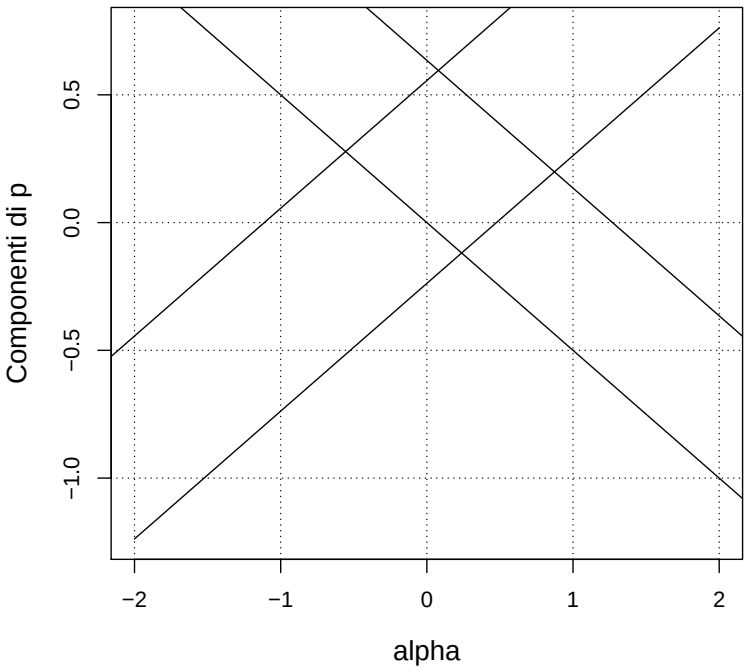
e $\pi = (100, 100, 100)'$. Abbiamo già visto che esiste un arbitraggio del secondo tipo nell'ultimo esempio della sezione precedente. Risolviamo il sistema $\mathbf{Y}'p = \pi$.

```
> Y <- t(matrix(c(105, 120, 110,
+                 105, 110, 102,
+                 105, 90, 98,
+                 105, 80, 90),4,3,byrow=T)) # attenzione: trasposta t()
> pai <- c(100,100,100)
> p <- qr.solve(Y,pai)
> p
[1]  0.6349206 -0.2380952  0.5555556
[4]  0.0000000
> Y %*% p # check: == pai
     [,1]
[1,]  100
[2,]  100
[3,]  100
> hom <- ker(Y) # sistema omogeneo associato
> hom
     [,1]
[1,] -0.5
[2,]  0.5
[3,]  0.5
[4,] -0.5
```

Tutte le soluzioni del sistema si possono scrivere come

$$p = \begin{pmatrix} 0.6349206 \\ -0.2380952 \\ 0.5555556 \\ 0.0000000 \end{pmatrix} + \alpha \begin{pmatrix} -0.5 \\ 0.5 \\ 0.5 \\ -0.5 \end{pmatrix}$$

Cerchiamo una soluzione che abbia tutte le componenti positive, scegliendo un opportuno α. È evidente però che tale p non esiste: la quarta componente è positiva solo se $\alpha < 0$ ma in questo caso la seconda componente, $-0.238 + \alpha 0.5$ sarebbe certamente negativa. Un modo geometrico per vedere lo stesso risultato è disegnare le componenti di p come funzioni di α.



Quindi, non esiste $p > 0$ e ciò conferma che ci sono opportunità d'arbitraggio. In particolare, abbiamo già osservato come il portafogli $(0.6, 0.4, -1)$ abbia costo nullo e payoff strettamente positivo (arbitraggio di secondo tipo).

Modifichiamo ora il primo titolo (colonna) di $\mathbf{Y}$ scrivendo 102 al posto di 105 e procediamo a cercare $p > 0$ tale che $\mathbf{Y}'p = \pi$:

```
> Y <- t(matrix(c(102, 120, 110,
+                 102, 110, 102,
+                 102, 90, 98,
+                 102, 80, 90),4,3,byrow=T)) # attenzione: trasposta t()
> pai <- c(100,100,100)
> p <- qr.solve(Y,pai)
> p
[1] 0.2614379 0.1960784 0.5228758
[4] 0.0000000
```

```
> Y %*% p # check: == pai
     [,1]
[1,]  100
[2,]  100
[3,]  100
> hom <- ker(Y) # sistema omogeneo associato
> hom
     [,1]
[1,] -0.5
[2,]  0.5
[3,]  0.5
[4,] -0.5
```

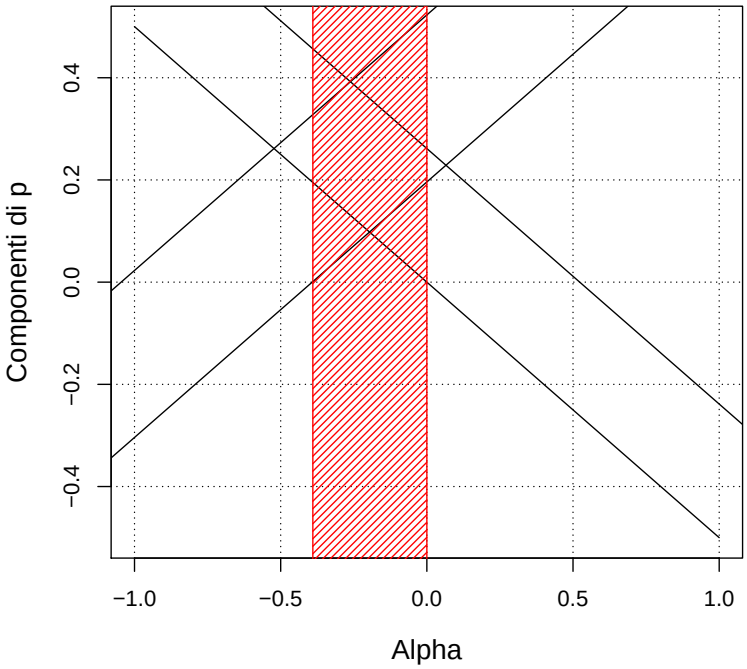
Le soluzioni sono del tipo

$$p = (0.2614379 - 0.5\alpha, 0.1960784 + 0.5\alpha, 0.5228758 + 0.5\alpha, -0.5\alpha)',$$

Basta scegliere valori negativi di α per ottenere $p_4 > 0$ ma sufficientemente piccoli per non rendere negativo p_2. Più formalmente, il sistema di disequazioni:

$$\begin{cases} 0.2614379 - 0.5\alpha > 0 \\ 0.1960784 + 0.5\alpha > 0 \\ 0.5228758 + 0.5\alpha > 0 \\ -0.5\alpha > 0 \end{cases}$$

è risolto per $-0.392 < \alpha < 0$ (io ho risolto il sistemino a mano, lo potete fare anche voi!) Un grafico mostra diversamente lo stesso risultato: per i valori di α nella zona tratteggiata il vettore p è strettamente positivo. Per il teorema di De Finetti si può concludere che non esistono arbitraggi né di primo né di secondo tipo.



Esercizio 11. Considerate l'economia con

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

e $\pi = (3.8, 4.9, 6)'$. Ci possono essere arbitraggi?

Esercizio 12. Stessa matrice dell'esercizio precedente, $\pi = (5.4, 6.3, 7.2)'$. Ci sono opportunità d'arbitraggio?

Esercizio 13. Considerate l'economia con

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{pmatrix}$$

e $\pi = (3, 3.6, 4.5)'$, occhio al 10! Ci possono essere arbitraggi?

Esercizio 14. Stessa matrice dell'esercizio precedente e $\pi = (3, 3.6, 4.8)$. Mostrate che è possibile un arbitraggio e determinatelo.

8.1 On ranks, replication, rounding errors... and much more

I'm indebted to Elena "Cooper Supertramp" who first pointed out the glitch...

This section explores a number of very relevant issues and digs into important ideas related to computational methods, practical finance and common sense.

Consider the following two assets $(104, 104, 104)'$ and $(101, 116, 103)'$ whose prices are 93.60 and 95.85, respectively. In such a market, how much would cost an asset whose payoff is $b = (102.71, 109.14, 103.57)'$?

To see whether the asset (column) b is replicable, we check the ranks:

```
> Y <- matrix(c(104,104,104,101,116,103),3,2)
> pai <- c(93.60,95.85)
> b <- c(102.71,109.14,103.57)
> Yb <- cbind(Y,b)
> qr(Y)$rank
[1] 2
> qr(Yb)$rank
[1] 3
```

Ranks are different, by Rc theorem the system $\mathbf{Y}x = b$ has no solution and b is not replicable. Hence, strictly speaking there is no (unique) price for b as the market is unable to provide a valuation based on the prevailing prices and assets.

However, compare b with 4/7 times the first asset + 3/7 times the second (which I call `bright`):

```
> bright <- 4/7*Y[,1]+3/7*Y[,2]
> bright
[1] 102.7143 109.1429 103.5714
> b
[1] 102.71 109.14 103.57
```

`bright` is obviously replicable (check it!) but b cannot even though, as you see, the only differences between the two vectors are due to rounding.

But there is something more going on. You see, you do not want rounding to stress your life: rounding should be a good thing, allowing to forget lots of irrelevant figures (cifre) and you do not want it to interfere with replication and arbitrage, which are very relevant practical issues.

1. Ranks are delicate objects, who count how many columns or rows are linearly independent. Often ranks are computed checking whether some determinants are null. In both cases, due to the limited precision of our computers things can be hard if columns are *almost* linearly dependent or when determinants are *relatively small* and *close to zero*. In other words, checking whether something is zero is tricky because 0 (and any other number) can only be approximated on a computer. This is particularly true for some matrices, like `Yb`, that are close to singular.

A measure of how much a matrix is close to singular is given by the *condition number* κ, see `http://en.wikipedia.org/wiki/Condition_number`. This number is always bigger or equal to 1 and (loosely said) it captures how problematic is a matrix, the convention being that $\kappa \approx 1$ denotes a perfect matrix, with strongly[4] linearly independent columns; if instead $\kappa >> 1$ (κ is large) then the matrix "has problems" in the sense that there are nearly dependent columns. We can try this on `Yb`

```
> kappa(Yb) # MASS is required
[1] 195713.9
```

The condition number is large (it's up to you to decide what exactly large means...) and there are almost linear columns in $\mathbf{Y}|b$. So, its rank is 3 but it should (or could) be not far from 2.

2. A second explanation looks more carefully at the way the rank is computed:

```
> qr(Yb)
$qr

[1,] -180.1332840 -184.7520861
[2,]    0.5773503  -11.5181017
[3,]    0.5773503   -0.1382626
                 b
[1,] -1.821078e+02
[2,] -4.936577e+00
[3,]  2.005019e-03

$rank
[1] 3

$qraux
[1] 1.577350269 1.990395604 0.002005019

$pivot
[1] 1 2 3

attr(,"class")
[1] "qr"
```

Notice that `$qraux` shows 3 numbers, two of which are clearly different from zero. The third, well, is different from zero but is small. The rank is exactly counting how

[4] In the limiting case, in which $\kappa = 1$, the matrix has orthogonal columns.

many of the elements of `$qraux` are non-zero. If you see 3 non-zero numbers, that's ok; but you should be aware that the 0.002005019 is telling you that the matrix `Yb` is not "far" from being of rank 2. This is precisely the conclusion that was drawn before looking at the condition number κ.

Therefore, b is almost replicable, as can be seen using the Moore-Penrose inverse:

```
> require(MASS)
> sol <- ginv(Y) %*% c(102.71,109.14,103.57)
> sol

          [,1]
[1,] 0.571379
[2,] 0.428593

> Y %*% sol

          [,1]
[1,] 102.7113
[2,] 109.1402
[3,] 103.5685
```

Remember that the Moore-Penrose inverse provides an approximate solution of a linear system in the cases in which there is no solution. Indeed, the portfolio `sol` is almost replicating the asset b and it would cost 94.5617110552765 to almost replicate b, see below.

```
> sum(pai*sol)

[1] 94.56171
```

I complete this section with some additional considerations. The first being that an economic agent may be satisfied with *approximate replication* anyway. Ask yourself whether you really want b or you are happy with `bright`. Framed differently, do you need to hedge perfectly? Or would small errors be tolerated?

This clearly depends on you and on the situation: to exemplify, think to a firm willing to hedge some future state-dependent exposures b due in an year; this can be done buying now about 3/7 and 4/7 units of the assets in the market; even though this does not perfectly replicate b (in strict sense), it may be enough for one's purposes, given the standard operational risks and noise inherently present in firm's activities.

Imagine now a naughty speculator, willing to exploit a tiny mispricing of b. He could trade a large number of shares to multiply his gains and pocket the price difference. However, he would not be exactly sure to have a perfect hedge: as a result, because b and `bright` are different, in some states of the world he could suffer losses (as even small discrepancies

can be relevant when multiplied by zillions of traded units). So, our speculator may well ignore this specific asset and look for better trading opportunities.

Take home message: arbitrages are ways to make money in which you aggressively exploit a mispricing; if the mispricing is 0, then no arbitrage; but if you see a mispricing that is *close* to zero, is that a true mispricing or just due to rounding? Well, you have to check things very carefully as computers have limited accuracy (all computers, not only mine or yours) and exploiting a tiny mispricing requires to trade huge quantities as you cash a tiny amount for each round.

Whether or not you want to engage in massive trading to get 1/1000 euro or less per unit is also related to other practical considerations (transaction costs and taxes, say, could wipe your profits), However, ranks (and `$qraux`!) are here to tell you that linear algebra is important, rounding exists and extra care and wisdom is needed when checking if something is zero on a computer.

9 Ottimizzazione

Un problema di ottimizzazione può essere scritto in forma estremamente compatta come

$$\max_{x \in D} f(x),$$

con $x \in \mathbf{R}^n$ e $f : \mathbf{R}^n \to \mathbf{R}$. Quindi, x è un vettore con n componenti (o variabili), $x = (x_1, x_2, ..., x_n)'$ e f è una funzione di queste n variabili. Sottolineo due cose terminologiche: possiamo pensare a f come a una funzione di n variabili reali ma anche come a una funzione di una sola variabile di $\mathbf{R}^n$, le due cose si equivalgono logicamente ma la scrittura sembra diversa, $f(x)$ e $f(x_1, x_2, ..., x_n)$, e anche il codice per calcolarle in `R`; mirabilmente, la matematica scrive in una riga cose complicate: il dominio D, che descrive il sistema di vincoli o l'insieme di ammissibilità, potrebbe essere complicato e, ad esempio, questo potrebbe rendere faticoso anche il solo verificare se $x \in D$. Nei casi più semplici $D = \mathbf{R}^n$ e non c'è nulla da verificare dato che x è un qualsiasi vettore di $\mathbf{R}^n$. In questo caso si parla di ottimizzazione libera, dato che x è *libero* di vagare in $\mathbf{R}^n$ come gli pare. Negli altri casi, quelli con D non banale, si parla di ottimizzazione vincolata.

Credo che formulare i problemi (tutti, nella vita come nel lavoro) sia utile perché obbliga a decidere quali sono le variabili $x_1, ..., x_n$, ciò su cui è possibile agire, distinguendole da ciò che è dato ed è immodificabile. Allo stesso tempo, obbliga a decidere che cosa si vuole: si tratta di f. Decidere qualsiasi cosa senza nemmeno capire o sapere che cosa si vuole è, semplicemente, cretino ma è molto diffuso (anche in parlamento, negli ultimi decenni). Questo non vuol dire che decidere sia facile, né che determinare f sia una passeggiata. Tutt'altro. La funzione obbiettivo f deve incorporare e sintetizzare in modo sensato diversi aspetti e spesso ai decisori si chiedono cose impossibili. Ma il solo sforzo di chiarire i termini del problema aiuta enormemente gli uomini di buona volontà a meglio comprendere che cosa stanno facendo e le ineliminabili difficoltà pratiche. Fine della predica!

Rivedete la definizione di massimo e minimo locale e globale.

Tipicamente, in teoria i massimi e minimi si caratterizzano con condizioni del primo e secondo ordine: delle prime dirò qualcosa, delle seconde no e da ora in poi assumerò che le funzioni da ottimizzare abbiano tutte le derivate necessarie.

Teorema 6. *Condizione necessaria affinché x^* sia di massimo o minimo locale è che*

$$\mathrm{D}f(x^*) = \mathbf{0},$$

cioè che le derivate si annullino.

Se f è funzione di una sola variabile, allora la derivata è una sola e la condizione dice che "se x^* è di max o min, allora la derivata f' si deve annullare in $x^* \in \mathbf{R}$". Se f è funzione di tante variabili, allora D denota l'operatore che deriva parzialmente rispetto a tutte le variabili e $\mathrm{D}f(x^*)$ è il vettore delle n derivate parziali nel punto x^*. Queste derivate parziali devono essere tutte nulle e il vettore, quindi, deve essere $\mathbf{0} \in \mathbf{R}^n$.

Ulteriori considerazioni sono importanti:

1. Il teorema sulle condizioni necessarie del primo ordine sta in una riga e vale per funzioni di una o n variabili, a patto di capire il senso di D e la differenza fra 0 e $\mathbf{0}$ (numero e vettore, rispettivamente).

2. Il teorema non dice nulla su funzioni non derivabili, si deve poter usare D.

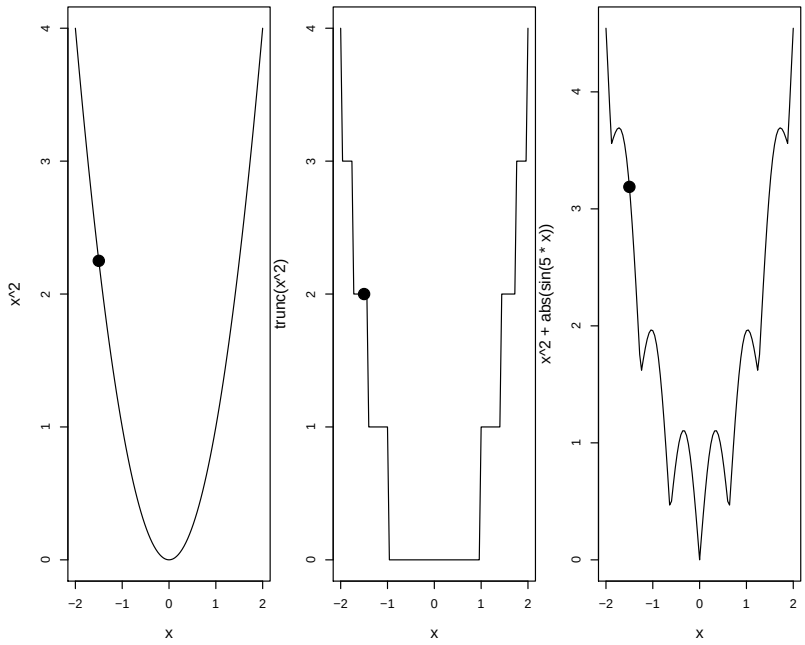
3. Il teorema non dice che è facile risolvere il sistema $\mathrm{D}f(x^*) = \mathbf{0}$, determinando i possibili massimi e minimi x^*. In generale, si tratta pur sempre di un sistema di n equazioni non lineari e talvolta annullare la derivata è difficile o impossibile da risolvere anche per $n = 1$.

4. Il teorema non dice se i punti eventualmente trovati sono di massimo o minimo: bisogna deciderlo da soli (con le condizioni del secondo ordine, ad esempio, o con un grafico o sapendo che le funzioni sono concave o convesse o utilizzando altre informazioni). Non dice nemmeno se i punti trovati sono estremanti globali, che spesso è la cosa che interessa di più.

5. Per questi motivi, pur essendo un teorema assolutamente fondamentale e tutti ricordano la storiella "la derivata è zero", computazionalmente non si usa quasi mai.

`R` e i metodi numerici utilizzano tipicamente idee diverse per determinare i massimi o minimi. Se pensate alla ricerca di un punto di minimo, molti dei metodi sono basati su una metafora gravitazionale: le soluzioni del problema di ottimizzare si determinano partendo da un punto iniziale che poi "rotola" verso il basso cercando il punto migliore (cioè più basso) sulla superficie/grafico definita da f. Numericamente le derivate vengono spesso calcolate o approssimate non tanto per annullarle, come direbbe il teorema su condizioni necessarie del primo ordine, ma per decidere in quale direzione spostare il punto visitando, per quanto possibile, le parti più basse della superficie, dove si trova evidentemente quanto si cerca (ricordate che cerchiamo un minimo).

I disegnini che seguono mostrano qualche situazione interessante (ma sono solo disegnini...) e ci si può immaginare dove finisce la pallina nera seguendo la metafora gravitazionale.



I disegnini mettono umilmente in evidenza che anche i metodi numerici possono incontrare problemi in presenza di funzioni multimodali, prive di derivate in alcuni punti o simili. Il caso rappresentato a sinistra mostra una funzione convessa (parabolica) ed è evidente che la pallina rotolando giù finira nel punto più basso del grafico. Centralmente si vede una funzione con punti di non derivabilità e con lunghi tratti piatti. In simili casi i metodi numerici tendono a "perdersi" perché la derivata non offre indicazioni sulla direzione da esplorare. Il grafico a destra mostra una funzione "patologica", con molti punti di minimo locale in cui la pallina si incastra senza rotolare verso il minimo globale.

9.1 Ottimizzazione per funzioni di una variabile

Il comando di `R` che ottimizza funzioni $f(x)$ di una variabile è `optimize`. Digitate `help(optimize)` sulla console per vedere la pagina di aiuto. `optimize` minimizza di default e richiede di inserire (il nome di) una funzione e un intervallo $[a, b]$ in cui cercare l'estremo.

L'algoritmo utilizzato ricerca in $[a, b]$ un punto di minimo in modo efficace (*golden search*) e approssimando iterativamente la funzione con parabole. Il metodo determina con certezza la soluzione per funzioni unimodali ma potrebbe individuare minimi locali in altri casi. il manuale recita

> If f is not unimodal, then optimize() may approximate a local, but perhaps non-global, minimum to the same accuracy.

```
> f <- function(x) x**2+abs(sin(5*x))
> optimize(f,c(-5,5))
```

```
$minimum
[1] -1.332268e-15

$objective
[1] 6.661338e-15
```

Nell'esempio si vede che `optimize` determina, in questo caso, la soluzione in assenza di regolarità della funzione. Questo non succede in generale, digitate `example(optimize)` per osservare un caso in cui il metodo non converge alla soluzione per certi intervalli di ricerca. Uomo avvisato, mezzo salvato!

Esercizio 15. La pagina di help di `optimize` definisce una funzione patologica `fp`. Disegnate la funzione sull'intervallo $[-10, 10]$. Determinate un altro intervallo in cui `optimize` non converge al punto più basso. Osservate che fornire un buon intervallo di ricerca iniziale è importante per funzioni come questa o multimodali.

9.2 Ottimizzazione di funzioni di più variabili

`optim` è l'ottimizzatore "universale" di `R`, nel senso che è la prima opzione da provare in una generalità di casi (senza che siano note altre informazioni sulla struttura della funzione o della soluzione cercata). Nuovamente, `optim` converge, se riesce, a punti di minimo locale: non cè certezza che abbia successo con funzioni irregolari (ma questa è la norma, *c'est la vie*).

Il suo corretto uso richiede alcune cautele

1. La funzione va definita come $f(x)$ con $x \in \mathbf{R}^n$ e non con la notazione $f(x_1, x_2, ..., x_n)$. Considerate ad esempio la funzione di tre variabili

$$f(x, y, z) = x^2 - e^{-(yz-1)^2} + (z + 1)^2$$

   È molto naturale pensare che si tratti di una funzione di tre variabili, si può definire f in `R` e valutarla in $(1, 2, 3)$ con

```
> f <- function(x,y,z) x^2-exp(-(y*z-1)^2)+(z+1)^2
> f(1,2,3)

[1] 17
```

   esattamente come si scriverebbe $f(1, 2, 3)$ in notazione matematica. Per usare `optim`, invece, bisogna pensare a f come a una funzione di un solo vettore $x \in \mathbf{R}^3$ le cui tre componenti x_1, x_2, x_3 sono presenti nella definizione:

$$f(x) = x_1^2 - e^{-(x_2x_3-1)^2} + (z_3 + 1)^2.$$

   In `R` bisogna scrivere

```
> f <- function(x) x[1]^2-exp(-(x[2]*x[3]-1)^2)+(x[3]+1)^2
> f(c(1,2,3))

[1] 17
```

Osservate che la scrittura è appesantita dalle parentesi quadre che servono per estrarre le componenti dal vettore `x`. Quando si valuta `f`, poi, bisogna ricordare che `x` è un vettore e, come tale, si scrive con l'operatore di concatenazione `c()`.

In pratica si preferisce spesso definire sia la $f(x, y, z)$, che consente di semplificare la notazione e fare grafici che la $f(x)$, con x vettore. In classe parleremo di "versione b" della funzione.

```
> f <- function(x,y,z) x^2-exp(-(y*z-1)^2)+(z+1)^2
> f(1,2,3)

[1] 17

> fb <- function(x) f(x[1],x[2],x[3])
```

Questo trucco è utile perché è ora possibile usare `f` per fare conti e grafici e `fb` per ottimizzare con `optim`.

2. `optim` richiede un vettore iniziale da cui partire con la procedura di minimizzazione (per funzioni di due variabili non confondete questo vettore iniziale con l'intervallo che era richiesto per `optimize` che, peraltro, si usa con funzioni di *una* variabile). Valgono le stesse osservazioni fatte in precedenza: iniziare l'ottimizzazione con un buon punto iniziale è sempre utile e talvolta necessario per trovare la soluzione. Spesso, comunque, non si sa dove si trova il punto di minimo e sono necessarie cautela e diverse prove per evitare di trovare soluzioni sub-ottime o punti di minimo locale.

   Nel caso di prima, iniziando da $(1, 2, 3)$ si ha

```
> optim(c(1,2,3),fb)

$par
[1] -0.0002303276  3.9779994754
[3] -0.9997845433

$value
[1] 9.945493e-08

$counts
function gradient
      86       NA
```

```
$convergence
[1] 0

$message
NULL
```

Sembra che il punto di minimo sia $(0, 4, -1)$ e il (valore) minimo sia 0. *Sembra...*

3. Potete specificare diversi metodi numerici utilizzando l'opzione `method`. Senza entrare nei dettagli, l'elenco che segue li discute brevemente (vedi `help(optim)`).

Nome	Descrizione	Robusto	Accurato
Nelder-Mead	Non utilizza le derivate, potrebbe convergere prematuramente, è il metodo usato di default	Si	No
BFGS	Il nome deriva da Broyden, Fletcher, Goldfarb and Shanno. Utilizza le derivate parziali per approssimare la funzione	No	Si
CG	Conjugate gradients method, funzione molto bene per funzioni regolari e di forma parabolica, più fragile di BFGS, forse più adatto a problemi con molte variabili	No	Si
L-BFGS-B	Si tratta di un BFGS con semplici vincoli "box" del tipo $a_i \leq x_i \leq b_i, i = 1, ..., n$.	No	Si
SANN	Simulated annealing, decantazione simulata. È un metodo stocastico adatto per funzioni difficili (quando funziona)	Si	No

Tabella 2: Metodi utilizzabili con `optim` tramite l'opzione `method`

Esercizio 16. Digitate `optim(c(1,2,3),fb)` e `optim(c(1,2,3),fb,method=''SANN'')`.

1. Quale metodo è stato usato nel primo caso? Quale nel secondo?
2. Quanto vale il valore ottimo trovato nel primo e nel secondo caso? Quale soluzione sembra migliore? Perché?
3. Digitate diverse volte `optim(c(1,2,3),fb,method="SANN")`. Ottenete sempre lo stesso risultato? Perchè?
4. Quanto vale il risultato `$convergence` nei diversi casi? Leggete nel manuale che cosa significa il codice 0?
5. Calcolate le derivate parziali di f nel punto (0,-4,-1). Può trattarsi di un massimo o minimo locale? [Lo potete fare a mano, con `R` definendo da voi le derivate parziali o col pacchetto `numDeriv` e la funzione `grad`]

6. Lanciate nuovamente `optim` con Nelder-Mead partendo dal punto $(0, 4, -1)$, che è il sedicente punto di minimo trovato con Nelder-Mead. Discutete il risultato.

7. Che cosa rappresenta il disegno `curve(f(0,x,-1),-3,3)`?

   Potete digitare `curve(fb(0,x,-1),-3,3)`

8. Digitate `{optim(c(1,2,3),fb,method="CG")`. Qual è il punto di minimo trovato? Si tratta di un punto critico? Il `$par` è il punto di ottimo globale?

Esercizio 17. Considerate la seguente funzione

```
> g <- function(x,y) x**2+y**2+2*abs(sin(5*x+y))
```

1. Rappesentate nel dominio $[-3, 3] \times [-3, 3]$ la funzione con `contour`.

2. Disegnate la superficie corrispondente con `persp`, utilizzando `theta=30,phi=15` e `ticktype="detailed"`.

3. Ottimizzate la funzione con diversi metodi e diversi punti iniziali. Qual è il minimo globale e quanto spesso gli algoritmi si fermano *prematuramente* in altri punti?

Esercizio 18. Determinate il minimo globale di

$$\frac{x^2 + xy + y^2 - y}{10} + e^{-x^2+2x} + e^{-y^2-y}.$$

Esercizio 19. Determinate il minimo globale di

$$\frac{x^2 + xy + y^2 - y}{4} + e^{-x^2-2x} + e^{-y^2}.$$

10 Ottimizzazione e soluzioni di sistemi di equazioni

Risolvere *sistemi di equazioni* e *ottimizzare funzioni* sono operazioni molto diverse. C'è però una brillante idea, chiamata talvolta *trucco di Gauss*, che sfrutta il fatto che è numericamente abbastanza semplice ottimizzare mentre è difficile risolvere sistemi di equazioni non lineari.

Sia dato il sistema di n equazioni in n incognite

$$\begin{aligned} g_1(x_1, x_2, ..., x_n) &= 0 \\ g_2(x_1, x_2, ..., x_n) &= 0 \\ ... &= 0 \\ g_n(x_1, x_2, ..., x_n) &= 0 \end{aligned}$$

In generale, se le equazioni $g_i(...) = 0$ sono non lineari, determinare soluzioni può essere molto difficile. Definite la funzione ausiliaria

$$sq(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} (g_i(x_1, ..., x_n))^2.$$

$sq : \mathbf{R}^n \to \mathbf{R}$ è una funzione a valori reali delle n variabili $x_1, x_2, ..., x_n$ con struttura molto particolare: somma i quadrati delle equazioni del sistema precedente. Osservate che una somma di quadrati è sempre non negativa (perché si sommano cose non negative). Supponete di sapere che $sq(y_1, y_2, ..., y_n) = 0$ per certi valori di $y = (y_1, ..., y_n)$. Allora y è minimo globale per sq: infatti la funzione è sempre positiva o nulla, non può assumere valori negativi e vale 0 in y.

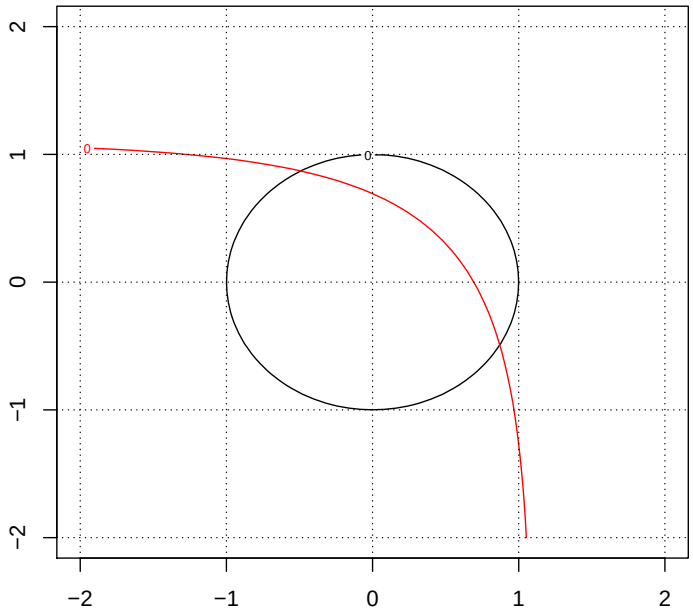
Ma una somma di quadrati vale 0 in y se e solo se tutti gli addendi della somma sono nulli. Ne segue che tutte le equazioni del sistema sono soddisfatte e che per risolvere il sistema si può cercare il minimo di sq e verificare che, nel punto trovato, la funzione valga 0.

Considerate il sistema

$$\begin{aligned} x^2 + y^2 - 1 &= 0 \\ e^x + e^y - 3 &= 0 \end{aligned}$$

Rappresentiamo graficamente il problema con `R`:

```
> x <- seq(-2,2,len=51)
> y <- seq(-2,2,len=51)
> g1 <- function(x,y) x**2+y**2-1
> g2 <- function(x,y) exp(x)+exp(y)-3
> z1 <- outer(x,y,g1)
> z2 <- outer(x,y,g2)
> contour(x,y,z1,levels=0)
> contour(x,y,z2,levels=0,add=T,col=2)
> grid(col=1)
```



Sono evidenti due soluzioni (simmetriche) che si possono determinare definendo e minimizzando la funzione

$$sq(x, y) = (x^2 + y^2 - 1)^2 + (e^x + e^y - 3)^2.$$

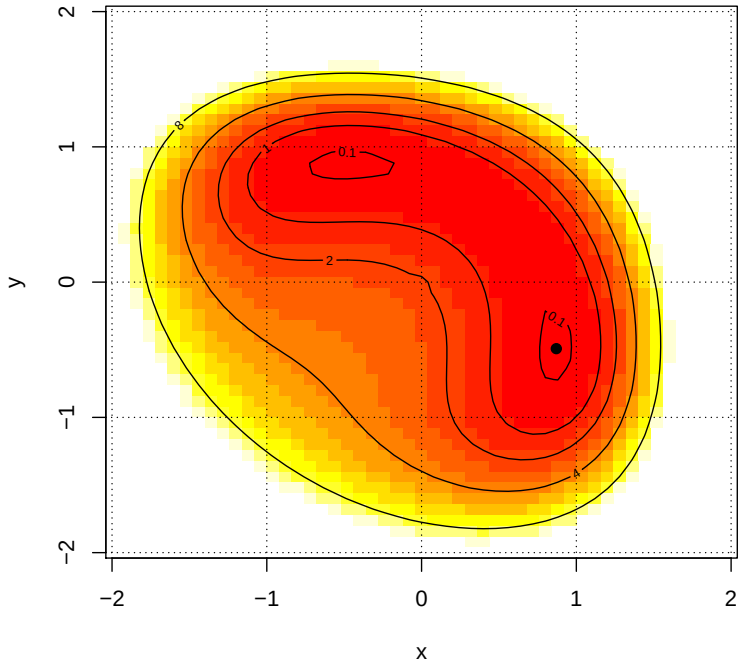
```
> sq <- function(x,y) g1(x,y)**2+g2(x,y)**2
> sqb <- function(x) sq(x[1],x[2])
> res <- optim(c(1,-0.5),sqb,method="BFGS")
> res
$par
[1]  0.8706713 -0.4918641

$value
[1] 2.125731e-12

$counts
function gradient
      40        9

$convergence
[1] 0
```

```
$message
NULL
> sqz <- outer(x,y,sq)
> image(x,y,sqz,zlim=c(0,10));contour(x,y,sqz,add=T,levels=c(0.1,1,2,4,8))
> grid(col=1)
> points(res$par[1],res$par[2],pch=19)
```



Il punto di minimo è circa $(0.87, -0.49)$, il valore minimo è circa nullo (zero a meno di errori numerici) e quindi abbiamo trovato una soluzione del sistema, come si verifica di seguito:

```
> g1(res$par[1],res$par[2])
[1] -1.182943e-06
> g2(res$par[1],res$par[2])
[1] -8.522782e-07
```

L'altro punto si può trovare immediatamente fornendo un adeguato punto iniziale.

Esercizio 20. Risolvete, se possibile, il sistema

$$\begin{aligned} x^2 + y^2 - 1 &= 0 \\ e^x + e^y - 3 &= 5 \end{aligned} \tag{3}$$

Esercizio 21. Ricavate x dalla seconda equazione del sistema e sostituite nella prima per ottenere un'equazione nella sola variabile y, con $-1 \leq y \leq 1$. Disegnate il grafico e risolvete l'equazione risultante con `uniroot`. Ottenete gli stessi risultati visti in precedenza?

10.1 Applicazione: condizioni del primo ordine

Nella precedente sezione abbiamo visto come si risolvono sistemi di equazioni non lineari: si trasforma il problema nella minimizzazione di una somma dei quadrati delle equazioni.

La ricerca dei punti critici per funzioni di due o più variabili è un caso interessante. Le condizioni necessarie del primo ordine, infatti, suggeriscono di cercare gli estremanti locali fra i punti che annullano le derivate parziali. Un esempio, che quasi si può fare a mano, chiarirà il metodo.

Esempio 6. Determinate e classificate i punti stazionari della funzione

$$f(x, y) = x^2y + y^2 + xy + x + 1.$$

Il sistema delle condizioni del primo ordine è

$$\begin{aligned} f'_x &= 2xy + y + 1 = 0 \\ f'_y &= x^2 + 2y + x = 0 \end{aligned}$$

Ricavando la y dalla seconda equazione e sostituendo nella prima si ottiene l'equazione cubica

$$-2x^2 - 3x^2 - x + 2 = 0,$$

che si può risolvere con

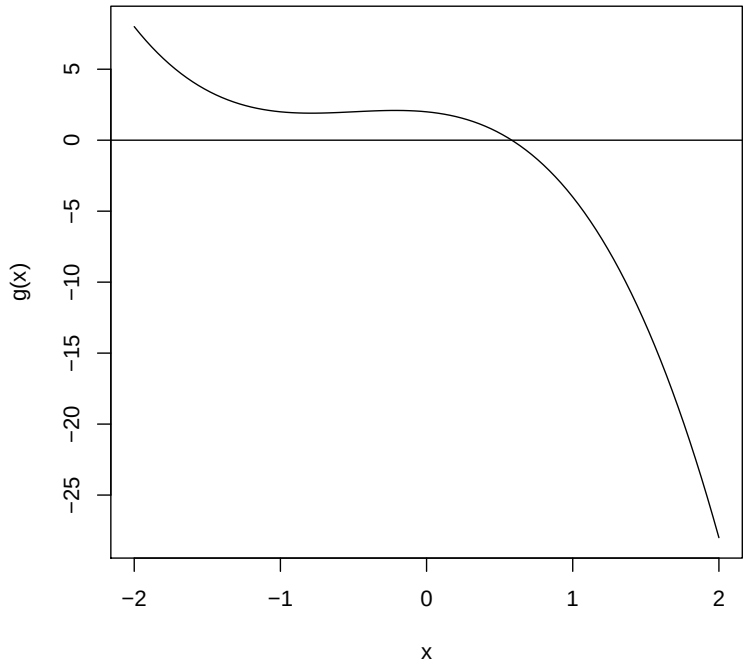
```
> g <- function(x) -2*x**3-3*x**2-x+2
> curve(g,-2,2);abline(h=0)
> uniroot(g,c(0,1))
$root
[1] 0.583162

$f.root
[1] -3.668493e-05

$iter
[1] 6

$init.it
[1] NA

$estim.prec
[1] 6.103516e-05
```



Da $x = 0.583162$ si ottiene che $y = (-x^2 - x)/2 = -0.46162$: l'unico punto critico della funzione è $(0.583162, -0.46162)$. Osservate che abbiamo dovuto utilizzare comunque metodi numerici per risolvere la cubica.

Un approccio più generale è, come visto, quello di minimizzare la somma delle derivate parziali quadrate:

```
> fx <- function(x,y) 2*x*y+y+1
> fy <- function(x,y) x**2+2*y+x
> sq <- function(x,y) fx(x,y)**2+fy(x,y)**2
> sqb <- function(x) sq(x[1],x[2])
> res <- optim(c(0,0),sqb,method="BFGS")
> res$par
[1]  0.5831559 -0.4616137
```

Il metodo è generalizzabile anche a casi con molte variabili e con equazioni (cioè, derivate parziali) più complicate. In realtà tutto si può automatizzare utilizzando il pacchetto `numDeriv` senza nemmeno prendersi la briga di definire o calcolare le derivate parziali. Il caso di prima diventa:

```
> f <- function(x,y) x**2*y+y**2+x*y+x+1
> fb <- function(x) f(x[1],x[2]) # versione b
> require(numDeriv) # carica numDeriv
```

```
> sqb2 <- function(x) sum(grad(fb,x)**2) # scubidoo!
> optim(c(0,0),sqb2,method="BFGS")
$par
[1]  0.5831559 -0.4616137

$value
[1] 9.264177e-13

$counts
function gradient
      45       14

$convergence
[1] 0

$message
NULL
```

Esercizio 22. Siete in gradi di rappresentare graficamente la soluzione del problema? Usate `image` per la funzione e `contour`, con `add=T, levels=0`, per visualizzare i punti dove le derivate parziali si annullano. Poi disegnate il punto critico.

Esercizio 23. Il punto $(0.583162, -0.46162)$ è di massimo, di minimo o di sella per la funzione f? Siete in grado di rispondere alla domanda sia graficamente che numericamente? Digitate `help(hessian)`: siete in grado di usare il test della derivata seconda per concludere? [si]

Esercizio 24. Il punto $(0.583162, -0.46162)$ è di massimo, di minimo o di sella per la funzione *sqb*2? La cosa è in contraddizione con l'esercizio precedente?

10.2 Sistemi lineari e trucco di Gauss

Come ulteriore applicazione del trucco di Gauss, si può risolvere un sistema lineare $\mathbf{A}x = b$, che è pur sempre un sistema di equazioni. L'interesse per la soluzione mediante minimizzazione non sta nell'efficacia del metodo. È infatti sempre preferibile ricorrere a metodi più diretti di algebra matriciale. Spesso però il problema è "risolvere il sistema soddisfacendo certi vincoli" e in questi casi risulta spesso utile ricorrere all'idea di minimizzazione.

Il sistema

$$\begin{pmatrix} -1 & 2 & 0 \\ -1 & 1 & -1 \\ 1 & 1 & 2 \end{pmatrix} x = (1, 2, 3)'$$

è risolto da $x = (11, 6, -7)'$, verificatelo per esercizio. Otteniamo la stessa soluzione mediante ottimizzazione quadrando e sommando le equazioni. In formule:

$$sq(x) = \sum_{i=1}^{n} (\mathbf{A}x - b)_i^2.$$

```
> A <- matrix(c(-1,-1,1,2,1,1,0,-1,2),3,3)
> b <- c(1,2,3)
> sqb <- function(x) sum((A %*% x - b)**2)
> res <- optim(c(0,0,0),sqb,method="BFGS")
> res$par
[1] 11  6 -7
```

Supponente ora di voler capire se esiste una soluzione positiva (cioè avente componenti tutte positive) a un sistema. In questo caso, come vedremo nella sezione che segue, l'idea giusta è di risolvere il sistema mediante minimizzazione tenendo conto dei vincoli (in questo caso di positività).

11 Ottimizzazione vincolata (vincoli lineari)

Abbiamo visto come un generico problema d'ottimizzazione si possa scrivere nella forma

$$\max_{x \in D} \quad f(x).$$

Sono particolarmente interessanti i casi in cui la regione ammissibile si può descrivere in modo lineare:

$$D = \{x \in \mathbf{R}^n : \mathbf{A}x - b \geq \mathbf{0}\}.$$

La gestione di vincoli più generali (non lineari) è più complicata. Come esempio di problema con vincoli lineari considerate il problema

$$\min_{x \geq 0, y \geq 0, x+y \leq 2} x^2 y + y^2 + xy + x + 1.$$

La f da minimizzare è non lineare e la regione ammissibile è definita da tre vincoli lineari. Possiamo descrivere D come l'insieme

$$D = \{x : \mathbf{A}x - b \geq \mathbf{0}\}, A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix}, b = \begin{pmatrix} 0 \\ 0 \\ -2 \end{pmatrix}.$$

Infatti

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ -2 \end{pmatrix}$$

equivale a

$$x \geq 0 \qquad x \geq 0 \tag{4}$$
$$y \geq 0 \qquad y \geq 0 \tag{5}$$
$$-x - y \geq -2 \qquad x + y \leq 2 \tag{6}$$

che è esattamente la regione D.

Problemi di questo tipo si possono risolvere con il comando `constrOptim` di `R`, con sintassi simile a `optim` aggiungendo le informazioni sulla regione ammissibile, cioè quelle relative ad $\mathbf{A}$ e b.

`constrOptim(par,f,grad,A,b,...)`,

in cui `par` è un punto iniziale, `f` è la funzione da ottimizzare, `grad` è il suo gradiente e, come detto, `A` e `b` definiscono il sistema di vincoli. Alcune precisazioni sono importanti:

1. il punto iniziale `par` deve essere *interno* alla regione, cioè deve soddisfare $\mathbf{A}\texttt{par} - b > \mathbf{0}$. È responsabiltà dell'utente inizializzare `constrOptim` con un punto di partenza appropriato, altrimenti si genera un errore;

2. il gradiente `grad` di f può essere omesso e, in questo caso, bisogna scrivere `NULL` consentendo a `R` di approssimare in proprio il gradiente. La specificazione di una funzione aggiuntiva accelera i tempi di calcolo e migliora la precisione ma molto spesso nella pratica si usa `NULL`;

3. la matrice $\mathbf{A}$ e il vettore b servono per specificare i vincoli (o la regione ammissibile). Come visto nell'esempio talvolta è necessario riscrivere opportunamente le disuguaglianze per rispettare la forma obbligatoria $\mathbf{A}x - b \geq 0$.

Il problema di prima si risolve con

```
> f <- function(x,y) x**2*y+y**2+x*y+x+1
> fb <- function(x) f(x[1],x[2]) # versione b
> A <- matrix(c(1,0,0,1,-1,-1),3,2,byrow=T)
> b <- c(0,0,-2)
> res <- constrOptim(c(0.5,0.5),fb,NULL,A,b)
> res
$par
[1] 2.646240e-08 1.561977e-04

$value
[1] 1

$counts
function gradient
     188       NA
```

```
$convergence
[1] 0

$message
NULL

$outer.iterations
[1] 3

$barrier.value
[1] -0.0001381606
```

Esercizio 25. Risolvete lo tesso problema appena visto fornendo come punto iniziale il punto $(0, 0)$. Perché si genera un errore, nonostante il punto appartenga alla regione ammissibile (tutti i vincoli sono soddisfatti)?

Esercizio 26. Risolvete il problema di massimizzazione nel caso precedente, con funzione obbiettivo e vincoli identici.

Consideriamo ora il sistema (fratello di quello già analizzato in precedenza)

$$\mathbf{A}x = b, \mathbf{A} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Verificate che il sistema ha infinite soluzioni una dell quali è data da

```
> require(MASS)
> A <- matrix(1:9,3,3)
> sol <- ginv(A) %*% c(1,2,3)
> sol
           [,1]
[1,]  0.8333333
[2,]  0.3333333
[3,] -0.1666667
```

Esistono delle soluzioni con componenti non negative? Cioè, esiste $x = (x_1, x_2, x_3)'$ tale che $\mathbf{A}x = b$ e $x_i \geq 0, i = 1, 2, 3$? La soluzione fornita dalla pseudo inversa non ha questa proprietà dato che $x_3 = -0.167$. Ma, in presenza di molte altre soluzioni (infinite), nessuno esclude che ci siano terne che soddisfano tutti i requisiti. Per trovarle:

- risolviamo il sistema con minimizzazione;
- e imponiamo i vincoli non non negatività $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$ (osservate che cono lineari).

```
> A <- matrix(1:9,3,3) # matrice del sistema
> b <- c(1,2,3) # b del sistema
> fb <- function(x) sum((A %*% x-b)**2)
> nn <- diag(c(1,1,1)) # nn non neg, matrice identica
> z <- c(0,0,0) # vettore termine noto vincoli
> res <- constrOptim(c(2,2,2),fb,NULL,nn,z)
> res
$par
[1] 9.990980e-01 4.138037e-04
[3] 3.696788e-11

$value
[1] 6.873731e-07

$counts
function gradient
    1232       NA

$convergence
[1] 0

$message
NULL

$outer.iterations
[1] 5

$barrier.value
[1] 0.0001003634
```

La soluzione, contenuta in `res$par`, è $(1, 0, 0)$. Verificate che risolve il sistema e rispetta i vincoli di non negatività.

Utilizzo questo esempio per mostrarvi una cosa forse un po' tecnica ma utile. Ricorderete che `constrOptim` consente di specificare il gradiente della funzione se questo è disponibile (mentre nel caso di prima abbiamo scritto `NULL`). L'informazione sul gradiente consente a `R` di utilizzare metodi di calcolo più sofisticati e la soluzione dello stesso problema diventa:

```
> fbg <- function(x) 2*t(A) %*% (A %*% x-b) # gradiente
> res <- constrOptim(c(2,2,2),fb,fbg,nn,z) # ora c'e' fbg
> res
$par
[1] 9.999858e-01 5.811550e-06
```

```
[3] 2.651525e-07

$value
[1] 1.551321e-10

$counts
function gradient
     252       86

$convergence
[1] 0

$message
NULL

$outer.iterations
[1] 6

$barrier.value
[1] 0.000100012
```

La soluzione trovata è la medesima di prima ma è evidente la maggior precisione sia nel punto di ottimo che nel valore della funzione obbiettivo che passa da 10^{-7} a 10^{-10} (e dovrebbe essere 0). Inoltre, io numero di valutazioni della funzione scende da 1232 a 252 (+ 86 gradienti).

Esercizio 27. Esiste una soluzione del sistema con componenti tutte non minori di 0.1?

11.1 Ottimizzazione vincolata e nucleo

La sezione precedente mostra come ottimizzare una funzione con vincoli di disuguaglianza del tipo $\mathbf{A}x - b \geq 0$. Che cosa possiamo fare però in presenza di *vincoli di uguaglianza*?

`constrOptim` non può essere utilizzato: il comando richiede un punto iniziale *interno* ai vincoli ma questo è impossibile nel caso $\mathbf{A}x = b$: o il punto soddisfa esattamente il vincolo (e non va bene perché non è interno) oppure non lo soddisfa (e non va bene come punto iniziale).

La soluzione consiste nel riscrivere il problema in termini di ottimizzazione libera, dopo che si è risolto il sistema con le conoscenze di algebra lineare della prima parte del corso. Assumete che il problema sia del tipo:

$$\begin{aligned} \min_x \quad & f(x) \\ & \mathbf{A}x = b, x \in \mathbf{R}^n \end{aligned}$$

Se il sistema $\mathbf{A}x = b$ non ha soluzione, non c'è nulla da fare perché la regione ammissibile è vuota. Altrimenti, si può risolvere il sistema ed esplicitare il vincolo: gli x della regione ammissibile sono

$$x = x_0 + y,$$

in cui x_0 è una soluzione particolare del sistema e y appartiene al nucleo di $\mathbf{A}$. Se i vettori $k_1, k_2, ..., k_m, m \leq n$, formano una base del nucleo allora

$$x = x_0 + y = x_0 + \sum_{i=1}^{m} \alpha_i k_i,$$

per $\alpha_1, ..., \alpha_m$. Osservate che gli α_i sono numeri reali, liberi di variare in $\mathbf{R}$. Se riscriviamo la funzione $f(x)$ in termini delle nuove variabili α, i coefficienti dei vettori del nucleo, si ha:

$$g(\alpha) = f(x) = f(x_0 + \alpha_1 k_1 + \alpha_2 k_2 + ... + \alpha_m k_m),$$

e il problema da risolvere è libero (senza vincoli) rispetto alle nuove m variabili decisionali α. Quindi, risolviamo

$$\min_{\alpha=(\alpha_1,...,\alpha_m)} g(\alpha),$$

in cui x_0 e tutti i vettori $k_1, ..., k_m$ sono fissati una volta che il vincolo $\mathbf{A}x - b = \mathbf{0}$ sia noto. Il problema si può anche scrivere definendo la matrice $\mathbf{K}$ che ha per colonne gli m vettori del nucleo di $\mathbf{A}$:

$$\min_{\alpha \in \mathbf{R}^m} g(\alpha) = \min_{\alpha \in \mathbf{R}^m} f(x_0 + \mathbf{K}\alpha).$$

Riassumendo, per risolvere un problema di ottimizzazione con vincoli lineari di uguaglianza basta riscriverlo come problema non vincolato usando come variabili i coefficienti dei vettori del nucleo di $\mathbf{A}$. Questa operazione equivale alla famigliare sostituzione del vincolo dentro alla funzione obbiettivo, anche se la cosa è complicata dalla molteplicità dei punti che appartengono alla regione ammissibile.

Il codice di `R` che implementa questa strategia è semplice (a questo punto del corso).

```
> optimeq <- function(f,A,b,...){
+         source("mmdd14.R")
+         n <- dim(A)[2] # colonne di A, numero di variabili
+         K <- ker(A) # K matrice dei vettori del nucleo
+         m <- dim(K)[2] # numero di vettori del nucleo
+         x0 <- ginv(A) %*% b # soluzione particolare
+         if(sum((A %*% x0-b)**2)>1e-6) stop("Non ci sono soluzioni")
+         g <- function(alpha) f(x0+K %*% alpha,...) # nuova funzione
+         res <- optim(rep(0,m),g,method="BFGS") # call standard
+         # che minimizza la funzione g partendo dall'origine
+         # res  produce la soluzione in termini di alpha
+         x0+K %*% res$par # ricalcola x da alpha=res$par
+ }
```

Esempio 7. Risolvete il problema

$$\min_{x,y,z} \quad x^2 + (y-1)^2 + (z-2)^2$$
$$x + y + z = 1$$

È un problema con tre variabili e un vincolo che si può scrivere come $(1,1,1)(x,y,z)' = 1$ da cui si vede che $\mathbf{A} = (1,1,1)$ (vettore riga) e $b = 1$ (sarebbe un vettore colonna ma, in questo caso, è un numero).

```
> f <- function(x,y,z) x**2+(y-1)**2+(z-2)**2 # definizione di f
> fb <- function(x) f(x[1],x[2],x[3]) # solita versione b
> A<- t(c(1,1,1)) # matrice A
> b <- 1 # termine noto del vincolo b
> optimeq(fb,A,b)
           [,1]
[1,] -0.6666667
[2,]  0.3333333
[3,]  1.3333333
```

La soluzione, come vedete, è $(x = -2/3, y = 1/3, z = 4/3)$ (verificate che soddisfa il vincolo). È interessante vedere da vicino come `R` (e la nostra algebra lineare) ha determinato la soluzione.

- si determina nucleo $\mathbf{K}$ di $\mathbf{A}$:

```
> K <- ker(A)
> K
           [,1]       [,2]
[1,] -0.5773503 -0.5773503
[2,]  0.7886751 -0.2113249
[3,] -0.2113249  0.7886751
```

- si trova una soluzione particolare del sistema/vincolo $\mathbf{A}x = b$:

```
> x0 <- ginv(A) %*% b
> x0
          [,1]
[1,] 0.3333333
[2,] 0.3333333
[3,] 0.3333333
```

- i punti che soddisfano il vincolo si possono scrivere come

$$x = \begin{pmatrix} 0.333 \\ 0.333 \\ 0.333 \end{pmatrix} + \alpha_1 \begin{pmatrix} -0.577 \\ 0.789 \\ -0.211 \end{pmatrix} + \alpha_2 \begin{pmatrix} -0.577 \\ -0.211 \\ 0.789 \end{pmatrix},$$

  e a questo punto la funzione $f(x)$ si può scrivere in funzione di $\alpha = (\alpha_1, \alpha_2)$. Siamo passati da 3 variabili (x, y, z) a 2 (α_1, α_2) perché abbiamo "sostituito" il vincolo.

- adesso definiamo la $g(\alpha)$ come $f(x_0 + \alpha_1 k_1 + \alpha_2 k_2)$ in cui k_1 e k_2 sono i vettori del nucleo:

```
> g <- function(alpha1,alpha2) fb(x0+alpha1*K[,1]+alpha2*K[,2])
> gb <- function(x) g(x[1],x[2])
```

  e siamo pronti ad minimizzare g.

- La soluzione del problema rispetto alle variabili α è

```
> optim(c(0,0),gb,method="BFGS")

$par
[1] 0.3660254 1.3660254

$value
[1] 1.333333

$counts
function gradient
       6        3

$convergence
[1] 0

$message
NULL
```

- abbiamo la soluzione sotto gli occhi ma è in termini di α_1, α_2, i due coefficienti dei due vettori del nucleo: un ultimo passaggio ci consente di tornare alla rappresentazione di partenza con tre variabili

```
> x0+0.3660254*K[,1]+ 1.3660254*K[,2]
```

```
           [,1]
[1,] -0.6666667
[2,]  0.3333333
[3,]  1.3333333
```

che conclude la vicenda.

Esercizio 28. Risolvete il problema precedente a mano, sostituendo il vincolo in f e derivando parzialmente rispetto alle due variabili rimaste.

Esercizio 29. Risolvete il problema

$$\begin{aligned} \min_{x,y,z} \quad & x^2 + (y-1)^2 + (z-2)^2 \\ & x + 2y + 3z = 4 \end{aligned}$$

Esercizio 30. Risolvete il problema

$$\begin{aligned} \min_{x,y,z} \quad & x^2 + (y-1)^2 + (z-2)^2 \\ & x + 2y + 3z = 4 \\ & x = 1 \end{aligned}$$

12 Analytic Hierarchy Process (AHP) - provvisorio

Queste note descrivono per sommi capi la procedura decisionale AHP e non sostituiscono le lezioni in cui sono stati fornite definizioni ed esempi.

12.1 Autovalori e autovettori

Rivedete il concetto di autovettore e autovalore sulla pagina di WIkipedia `http://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors`: leggete la definizione e poi la sezione "Eigenvalues and eigenvectors of matrices". Suggerisco la pagine di Wikipedia in inglese perché quella in italiano è molto meno comprensibile.

Un altro buon file è `http://math.mit.edu/linearalgebra/ila0601.pdf`: per il nostro corso è sufficiente conoscere la nozione di autovalore/autovettore, le modalità di calcolo manuale su semplici matrici 2×2 e 3×3 e il calcolo con `R` a base di `eigen`

Se vi serve material in italiano provate con le prime 3 pagine di `http://www.di.unipi.it/~bevilacq/autoval.pdf` oppure con `http://www.cremona.polimi.it/dispense/vannozzi/esercitazioni/analisi_a/materiale/AUTOVALORI_AUTOVETTORI.pdf` (le prime pagine con definizioni ed esempi, non coprite molteplicità e diagonalizzazione).

12.2 Teorema di Perron-Frobenius

Leggete il testo del teorema su `http://it.wikipedia.org/wiki/Teorema_di_Perron-Frobenius`

Teorema 7. *Se* $\mathbf{A}$ *è una matrice non negativa (cioè, con tutti gli elementi maggiori o uguali a zero) primitiva e indecomponibile allora:*

1. *L'autovalore di modulo massimo* λ *di* $\mathbf{A}$ *è reale positivo;*
2. *L'autovettore corrispondente ha tutte le componenti positive.*

Inoltre se la matrice è strettamente positiva, allora sia l'autovettore di modulo massimo che il relativo autovettore sono strettamente positivi.

Nella teoria dell'AHP, il Teorema di PF si usa su una matrice di confronti fra coppie di possibili decisioni $\mathbf{A}$: poiché questa matrice è a componenti strettamente positive, esiste un autovettore $\mathbf{w}$ che useremo per ordinare le alternative del decisore. Per i dettagli si veda la sezione che segue.

12.3 AHP: l'idea di base

Un problema decisionale può essere ricondotto in molti casi utili alla necessità di fornire una classifica, o un *ranking*, delle alternative possibili: se A è meglio di B che a suo volta è meglio di C, allora conviene scegliere A.

È evidente che, di solito, il decisore deve costruirsi il suo ranking e che questa operazione può essere delicata e coinvolgere diversi criteri. Spesso è più facile confrontare le coppie di

alternative, ad esempio A contro B, A contro C e così via, piuttosto che fornire direttamente una classifica. Ma è comodo assumere che un ranking sia già disponibile per comprendere le proprietà di quello che cerchiamo.

Considerate, ad esempio le alternative A_1, A_2, A_3 e supponete di avere un criterio di merito che genera il vettore $\mathbf{w} = (8, 4, 1)$. Questo significa che secondo il vostro criterio gradite A_1 con intensità 8, A_2 con intensità 4 e A_3 con intensità 1. Se siete coerenti nell'applicazione dei punteggi contenuti in $\mathbf{w}$ potete facilmente confrontare le alternative a due a due: ad esempio A_1 è 2 volte preferibile ad A_2 (8/4=2) e il decisore gradisce A_3 un quarto di quello che gradisce A_2. Continuando in maniera sistematica questi confronti basati sui rapporti delle componenti di $\mathbf{w}$ si giunge alla seguente matrice $\mathbf{A}$:

$$\begin{array}{c|ccc} & A_1 & A_2 & A_3 \\ \hline A_1 & \frac{8}{8} & \frac{8}{4} & \frac{8}{1} \\ A_2 & \frac{4}{8} & \frac{4}{4} & \frac{4}{1} \\ A_3 & \frac{1}{8} & \frac{1}{4} & \frac{1}{1} \end{array} = \begin{pmatrix} 1 & 2 & 8 \\ 1/2 & 1 & 4 \\ 1/8 & 1/4 & 1 \end{pmatrix},$$

che si legge "l'alternativa sulla riga è preferita a quella sulla colonna di x volte" o, più formalmente: la componente a_{ij} della matrice $\mathbf{A}$ denota di quante volte l'alternativa i è preferita all'alternativa j (con l'ovvia convenzione che se $a_{ij} < 1$ allora è j ad essere preferita).

Osservate ora che abbiamo una matrice $\mathbf{A}$ contenente confronti a coppie e un ranking $\mathbf{w}$. Che legame c'è fra la matrice e il vettore? Considerate il prodotto

$$\mathbf{Aw} = \begin{pmatrix} 1 & 2 & 8 \\ \frac{1}{2} & 1 & 4 \\ \frac{1}{8} & \frac{1}{4} & 1 \end{pmatrix} \begin{pmatrix} 8 \\ 4 \\ 1 \end{pmatrix} = \begin{pmatrix} 8+8+8 \\ 4+4+4 \\ 1+1+1 \end{pmatrix} = 3 \begin{pmatrix} 8 \\ 4 \\ 1 \end{pmatrix} = 3\mathbf{w}.$$

Si dice che il vettore $\mathbf{w}$ è un autovettore della matrice $\mathbf{A}$ perché $\mathbf{Aw}$ è una costante per il vettore stesso. In particolare la costante, nel nostro caso 3, è semplicemente il numero di alternative presenti. Riassumendo:

> Il ranking prodotto da un decisore perfettamente coerente (che non "sbaglia mai" i confronti a coppie) è un autovettore della matrice dei confronti. Inoltre, l'autovalore corrispondente è il numero n di alternative.

Questa proprietà è molto utile perchè nella realtà nessuno vi dice $\mathbf{w}$, che è anzi quello che il decisore cerca, mentre è possibile riempire la matrice $\mathbf{A}$ con il confronto, la discussione e l'analisi comparata di due alternative alla volta.

In pratica, supponente che il decisore sia in grado di valutare coppie di opzioni e di riempire la matrice $\mathbf{A}$, con la regola che $a_{ij} = 1/a_{ji}$. Questo significa che se, ad esempio, ritiene che $a_{12} = 3$ (cioè la prima opzione è tre volte preferita alla seconda) allora ritenga anche che $a_{21} = 1/3$ (cioè che la seconda gli piace un terzo della prima[5]). Assumeremo

[5]La gente fa cose strane e capita spesso che se la carne ti piace il doppio del pesce poi tu dica che il pesce è buono quanto la carne. Questo tipo di inconsistenza nelle valutazioni è abbastanza frequente

anche che qualsiasi decisore condivida che $a_{ii} = 1$ per ciascuna alternativa A_i: ogni A_i mi piace quanto A_i stessa (al 100%, per cosi dire) dato che pare assurdo poter pensare, per dirne una, che una cosa mi piace il doppio della cosa stessa (!?).

Nel caso visto prima, una possibile matrice $\mathbf{A}'$ prodotta da un decisore in carne ed ossa potrebbe essere la seguente:

$$\mathbf{A}' = \begin{pmatrix} 1 & 3 & 8 \\ \frac{1}{3} & 1 & 5 \\ \frac{1}{8} & \frac{1}{5} & 1 \end{pmatrix}$$

Osservate che ci sono degli "errori" (prima le avevamo chiamate inconsistenze) ma questo è quello che il decisore ha prodotto. Sfruttiamo la procedura vista prima: il ranking implicito nella matrice è (dovrebbe essere!) un autovettore della matrice.

```
> Ap<- matrix(c(1,3,8,1/3,1,5,1/8,1/5,1),3,3,byrow=T);Ap
          [,1] [,2] [,3]
[1,] 1.0000000  3.0    8
[2,] 0.3333333  1.0    5
[3,] 0.1250000  0.2    1
> eAp <- eigen(Ap);eAp
$values
[1]  3.0440663+0.000000i
[2] -0.0220332+0.365589i
[3] -0.0220332-0.365589i

$vectors
             [,1]
[1,] 0.9208767+0i
[2,] 0.3785129+0i
[3,] 0.0933493+0i
                          [,2]
[1,]  0.92087671+0.00000000i
[2,] -0.18925644+0.32780177i
[3,] -0.04667465-0.08084287i
                          [,3]
[1,]  0.92087671+0.00000000i
[2,] -0.18925644-0.32780177i
[3,] -0.04667465+0.08084287i
```

L'autovettore che cattura il ranking del decisore è $(0.92, 0.38, 0.09)$, in cui abbiamo rimosso l'unità immaginaria i perché moltiplicata per zero. Il vettore è la prima colonna di

e dipende da confusione, da come è formulato linguisticamente il problema, da errori di memoria o di comunicazione, da abuso di sostanze... In ogni caso, è importante individuare le inconsistenze e capire la loro entità. Piccole discrepanze dalla consistenza perfetta si possono accettare ma enormi inconsistenze conducono a decisioni fumose e prive di senso. Inconsistenti, appunto!

`eAp$vectors`. Possiamo normalizzare il vettore, dividendolo per la somma delle componenti, in modo che i numeri si possano interpretare come percentuali di un'ipotetica torta di ampiezza 100%.

```
> wp <- Re(eAp$vectors[,1]/sum(eAp$vectors[,1]))
> wp
[1] 0.6611984 0.2717759 0.0670257
```

Il vettore $\mathbf{w}'$, chiamato `wp` in `R` e relativo alla matrice $\mathbf{A}'$ appare molto sensato (nonostante $\mathbf{A}'$ contenesse dei ragionevoli errori di valutazione (incosistenze). $\mathbf{w}'$ suggerisce che il candidato migliore raccoglie il 66% del consenso mentre le altre alternative raggiungono 27 e 7%, rispettivamente. A ulteriore riprova della bontà del ranking ottenuto a partire dalla matrice dei confronti osservate che il ranking *giusto* è molto simile se viene normalizzato:

```
> c(8,4,1)/sum(c(8,4,1))
[1] 0.61538462 0.30769231 0.07692308
> wp
[1] 0.6611984 0.2717759 0.0670257
```

È evidente che i due ranking sono assai simili e, di conseguenza, le incosistenze dovrebbero essere limitate. Un modo utile per misurare l'inconsistenza del decisore è confrontare l'autovalore di $\mathbf{w}'$ con $n = 3$ che dovrebbe essere il suo valore:

```
> eAp$values[1] # primo autovalore di A'
[1] 3.044066+0i
> 3
[1] 3
```

Il valore 3.04 pare abbastanza vicino a 3 e questo ci conferma nell'idea che, in questo specifico caso, il decisore abbia dato valutazioni consistenti. Una stima più precisa del livello di (in)consistenza richiede un po' più di lavoro ma questo esempio dovrebbe aver chiarito che una buona procedura decisionale si potrebbe sviluppare secondo il seguente schema:

- Obbiettivo: dato un criterio di valutazione, fornire un ranking sulla preferibilità di n opzioni $A_1, A_2, ..., A_n$ (che può essere usato scegliendo il miglior classificato se è richiesta un decisione "secca").

    1. Creazione di una matrice di confronti: il decisore/decisori formulano giudizi su tutti i confronti a coppie delle alternative e riempiono la matrice

    $$\mathbf{A} = \{a_{ij}\},$$

    in cui a_{ij} indica quante volte A_i è preferito a (soddisfa di più di) A_j. Se $a_{ij} > 1$, l'opzione i-esima è meglio della j-esima; se invece $a_{ij} < 1$, vale il contrario;

2. Ovvie incosistenze vanno eliminate subito: $a_{ii} = 1$ per ogni $i = 1, ..., n$ e $a_{ij} = 1/a_{ji}$. Se un decisore non riesce in prima battuta a rispettare queste norme elementari aiutatelo a capire cosa sta facendo, chiedetegli di rianalizzare la cosa o... arrendetevi!

   Nella pratica, la scala usate per le valutazioni è codificata come segue:

The Fundamental Scale for Pairwise Comparisons		
Intensity of Importance	**Definition**	**Explanation**
1	Equal importance	Two elements contribute equally to the objective
3	Moderate importance	Experience and judgment moderately favor one element over another
5	Strong importance	Experience and judgment strongly favor one element over another
7	Very strong importance	One element is favored very strongly over another; its dominance is demonstrated in practice
9	Extreme importance	The evidence favoring one element over another is of the highest possible order of affirmation
Intensities of 2, 4, 6, and 8 can be used to express intermediate values. Intensities of 1.1, 1.2, 1.3, etc. can be used for elements that are very close in importance.		

Figura 1: Scala dei valori per i rapporti (fonte: Wikipedia, `http://en.wikipedia.org/wiki/Talk:Analytic_Hierarchy_Process/Example_Leader`).

3. Calcolo di autovalore e autovettore: si tratta dell'autovettore di Perron-Frobenius, vedi `http://en.wikipedia.org/wiki/Perron_Frobenius_theorem`, che assicura che una matrice positiva come la nostra $\mathbf{A}'$ abbia sempre un autovalore positivo e un autovettore a componenti positive.

   Sia λ l'autovalore e $\mathbf{v}$ l'autovettore risultante. Normalizzate $\mathbf{v}$, dividendo per la somma delle sue componenti, ottenendo $\mathbf{w}$.

   Il vettore $\mathbf{w}$ è il ranking cercato.

4. Valutazione dell'inconsistenza della decisione: si può calcolare un *(in)Consistency Index*:

$$\text{C.I.} = \frac{\lambda - n}{n - 1}.$$

   Si tratta di un valore che è nullo per decisori perfettamente consistenti e cresce all'aumentare delle contraddizioni nelle valutazioni: più basso è, meglio è.

   C.I. è una misura dell'inconsistenza decisionale.

   Questo numero va confrontato con l'inconsistenza di un decisore casuale (una persona cioè che attribuisce ad ogni a_{ij} valori a caso), i cui valori sono riportati nella Tabella 3.

n	Inconsistenza casuale
1	0.00
2	0.00
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41

Tabella 3: Tabella di inconsistenze casuali al variare di n. Un decisore consistente dovrebbe avere un C.I. che non supera il 10% del valore in tabella.

Ad esempio, nell'esempio delle pagine precedenti si avrebbe

$$\mathbf{C.I.} = \frac{3.04 - 3}{3 - 1} = 0.02.$$

L'inconsistenza di un decisore casuale[6] in corrispondenza di $n = 3$ è 0.58. Il 10% di questo numero è circa 0.06: poiché 0.02 è minore di 0.06, nella pratica la decisione è accettata perchè sufficientemente consistente [se volete: è molto meno inconsistente di quella presa a caso.]

Che cosa si fa se il C.I. è troppo grande (cioè supera il 10% del valore in Tabella 3)? La buona pratica impone di discutere col decisore e capire i motivi dell'inconsistenza. Questo dovrebbe condurre, si spera, a una revisione dei giudizi e a una conseguente riduzione del livello d'inconsistenza. E se non si riesce a raggiungere un livello accettabilmente basso, almeno sarà noto a tutti che si tratta di una decisione difficile e probabilmente affetta da stranezze o particolarità.

12.4 AHP: applicazioni

Si veda la pagina `http://en.wikipedia.org/wiki/Analytic_hierarchy_process`

L'AHP si impara anche con esempi: noi abbiamo coperto in classe l'esempio "Choosing a leader" presente nella pagina precedente e scaricabile su `http://en.wikipedia.org/w/index.php?title=Talk:Analytic_Hierarchy_Process/Example_Leader`

Un altro esempio più sofisticato, sempre linkato su `http://en.wikipedia.org/wiki/Analytic_hierarchy_process` è relativo all'acquisto di una vettura: `http://en.wikipedia.org/w/index.php?title=Talk:Analytic_Hierarchy_Process/Example_Car`

[6]Vi ricordo che un decisore casuale è una "bestia", uno che non s'impegna e spara numeri a caso alla richiesta di dire di quanto A_i è meglio di A_j. Non è una cosa bella prendere decisioni a caso e tutti dovrebbero, salvo casi particolarissimi, essere molto più consistenti di questo soggetto.

Un esempio di problema con vincoli lineari di ugiaglianza e disuguaglianza:

$$\max_{x,y,z,w} x^2 + y^2 + z^2 + w^2,$$

con vincoli $x+y = 1, y+z = 2, z+w \geq 3, w+2x \leq 4$. Notate che ci sono due diseguaglianze che posso scrivere come $A\mathbf{x} \geq b$ e due eguaglianze che posso scrivere come $C\mathbf{x} = d$ con

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ -2 & 0 & 0 & -1 \end{pmatrix}, b = \begin{pmatrix} 3 \\ -4 \end{pmatrix}, C = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, d = \begin{pmatrix} 1 \\ 2 \end{pmatrix}.$$

Possiamo riscrivere i vincoli di eguaglianza come

$$x = x_0 + Ker(C) \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix},$$

con la solita notazione in cui x_0 è una soluzione particolare, $Ker(C)$ è la matrice dei vettori del kernel e $\alpha = (\alpha_1, \alpha_2)$ è un vettore di parametri.

A questo punto, se $x = x_0 + Ker(C)\alpha$, posso sostituire dentro all'altro vincolo:

$$Ax \geq b; \quad A(x_0 + Ker(C)\alpha) \geq b \quad Ax_0 + A \cdot Ker(C)\alpha \geq b; \quad A \cdot Ker(C)\alpha \geq b - Ax_0.$$

In altre parole, il problema originario che aveva 4 variabili adess ne ha due che rispettano un vincolo "diverso" che deriva dalla sostituzioni. Inoltre, due variabili sono sparite perché sono state esplicitate usando $Cx = d$ mediante la ben nota relazione per i sistemi $x = x_0 + Ker(C)\alpha$ che a parole si legge: " le x sono date da una soluzione particolare x_0 più i vettore del kernel per dei parametri".

Tutto questo in `R` diventa:

```
> f <- function(x,y,z,w) x^2+y^2+z^2+w^2
> fb <- function(x) f(x[1],x[2],x[3],x[4])
> A <- matrix(c(0,0,1,1,-2,0,0,-1),2,4,byrow=T)
> A
     [,1] [,2] [,3] [,4]
[1,]    0    0    1    1
[2,]   -2    0    0   -1
> b <- c(3,-4)
> C <- matrix(c(1,1,0,0,0,1,1,0),2,4,byrow=T)
> C
     [,1] [,2] [,3] [,4]
[1,]    1    1    0    0
[2,]    0    1    1    0
> d <- c(1,2)
> kC <- ker(C) # kC e' il nome del kernel di C
> kC # qui si vede che kC ha due colonne; per questo ho usato
```

```
           [,1] [,2]
[1,]  0.5773503    0
[2,] -0.5773503    0
[3,]  0.5773503    0
[4,]  0.0000000    1
> # due parametri alpha1 e alpha per il vettore alpha
> x0 <- ginv(C) %*% d
> C %*% x0 # controllo: dovrebbe venire d. ok!
     [,1]
[1,]    1
[2,]    2
> gb <- function(alpha) fb(x0+kC %*% alpha)
> # adesso i vincoli sono A %*% Ker(C) %*% alpha\geq b-A x_0
> R <- A %*% kC
> r <- b-A %*% x0
> sol <- constrOptim(c(1,2),gb,NULL, R,r)
> alpha <- sol$par
> alpha
[1] 0.8659439 1.5000471
```

La soluzione in termini di α è (0.8659439, 1.5000471). Per tonare a (x, y, z, w) uso la solita $x = x_0 + ker(C)\alpha$:

```
> # adesso torno da alpha a (x,y,z,w)
> # con la relazione x=x0+ker(C) %*% alpha
> x0+kC %*% alpha
          [,1]
[1,] 0.4999529
[2,] 0.5000471
[3,] 1.4999529
[4,] 1.5000471
```

che è la soluzione finale: $x = 1/2, y = 1/2, z = 3/2, w = 3/2$, controllate per esercizio che rispetta tutti i vincoli.