Calcolo Simbolico 2

Introduzione

Abbiamo visto, finora, come elaborare in modo banale alcune espressioni. In fondo, però, fino ad adesso non abbiamo veramente eseguito calcoli utili, e non abbiamo risolto un bel niente. Mi dispiace, ma era doveroso scrivere tutto quanto, perchè a mio avviso prima di camminare bisogna cominciare col saper stare in piedi. Non si può apprezzare la potenzialità di un programma come *Mathematica* se prima non se ne conoscono le basi. Andare subito a risolvere un'equazione differenziale del 10° ordine serve a ben poco, se non si sa dove andare a parare, con il risultato che si pensa che non si può usare il programma e che non riesce a fare quello che vogliamo, e quindi lo cestiniamo.

Adesso, però, ne sappiamo abbastanza, e soprattutto si è fatta quel poco di pratica che serve per non spaventarci a scrivere cose nuove, e per essere sicuri che, quando si scriverà una formula complicata, in fondo si saprà scrivere nel modo giusto.

Occhio, però!!!! Non abbiamo mica finito di imparare! Per esempio, non abbiamo ancora visto come si definiscono le funzioni, ma lo faremo fra poco e, soprattutto, le useremo per poterci finalmente lavorare sopra. Cominceremo a fare elaborazioni banali, come le derivazioni, ma pian piano le cose si faranno più complesse, più complete e più utili. Dato che questo è un capitolo che mostra le funzionalità base di *Mathematica*, sarà, credo uno dei più lunghi. Comunque, niente paura. E' niente rispetto a quello che potrete sapere, e i concetti matematici che useremo saranno sicuramente digeriti da molto tempo, per voi.

Ma adesso bando alle ciance, o miei discepoli e diletti sottoposti per la mia futura conquista del mondo!!! Costruiremo la verità che più ci piace!!!

Funzioni

Abbiamo visto alcune delle funzioni tipiche di *Mathematica*. Per lo più, si tratta di funzioni per manipolare dati, ma ha anche un numero incredibile di funzioni matematiche, naturalmente:

```
In[1]:= BesselJ[4, 8] // N
```

```
Out[1]= -0.105357
```

```
In[2]:= Binomial[5, 3]
```

```
Out[2]= 10
```

```
In[3]:= RiemannSiegelTheta[5]

Out[3]= RiemannSiegelTheta[5]

In[4]:= N[%]

Out[4]= -3.45962
```

E così via nei secoli dei secoli... Ma che succede se vogliamo una funzione personalizzata fatta da noi? Semplice, ce la facciamo!!! Il modo normale per poter definire delle funzioni è il seguente;

```
funzione[x_] := espressione
```

Notate due cose importanti: quando defininiamo la funzione, l'argomento (o gli argomenti) devono essere obbligatoriamente accompagnati dal segno di underscore, il che indica un pattern: senza addentrarci troppo, l'underscore indica a *Mathematica* che x può essere una qualsiasi espressione (reale, complesso, incognita etc). Per ora ci basta sapere che è così che si definiscono le funzioni. L'altra cosa da notare è che il segno = è sostituito dal segno := con i due punti che precedono l'uguale: questo significa che *Mathematica* deve valutare l'espressione non subito, ma solo quando viene invocata; per ora ci basti sapere che si definisce così anche questo. Più avanti capiremo meglio tutte queste precisazioni...

Allora, definiamo la nostra funzione d'esempio:

```
In[5]:= f[x_] := x^3 + Sin[Cos[x]]
```

Se vogliamo calcolare la funzione in un punto, basta utilizzarla come qualsiasi altra espressione:

```
In[6]:= f[2]

Out[6]= 8 + Sin[Cos[2]]

In[7]:= f[6.]

Out[7]= 216.819

In[8]:= f[a + b]

Out[8]= (a + b)^3 + Sin[Cos[a + b]]
```

Inoltre, è possibile naturalmente definire funzioni a più variabili:

```
In[9]:= g[x_, y_, z_] := x^2 + y^3 + z^4
```

Il simbolo di underscore serve ad indicare a *Mathematica* che l'argomento della funzione può essere un argomento qualsiasi. A volte, invece, non vogliamo che sia così: per esempio, potremmo voler definire delle funzioni ricorsive, e quindi ci piacerebbe far sapere a *Mathematica* che gli argomenti della funzione possono essere solamente numeri interi. Possiamo naturalmente fare anche questo: vediamo l'esempio più classico possibile ed immaginabile di funzione ricorsiva, considerando (guardacaso) i numeri di Fibonacci; F_1,F_2 sono posti uguali ad 1, e un generico F_n è definito come la somma dei due precedenti. Per poter definire questa funzione, dobbiamo prima definire i primi due numeri:

In[10]:=
```
fibo[1] = fibo[2] = 1;
```

Adesso definiamo la funzione di Fibonacci:

In[11]:=
```
fibo[x_Integer] := fibo[x - 1] + fibo[x - 2]
```

Proviamo a vedere se funziona...

In[12]:=
```
fibo[10]
```

Out[12]=
```
55
```

mentre, se vogliamo calcolarci la funzione per un valore non intero, *Mathematica* non calcola la funzione, perchè l'argomento non corrisponde con quello che si aspetta:

In[13]:=
```
fibo[3.5]
```

Out[13]=
```
fibo[3.5]
```

Effettivamente, funziona. Vediamo di spiegare in modo semplice quello che ho fatto. prima di tutto, per definire la funzione, ho definito la funzione in punti particolari. In questa maniera eseguo una specie di overloading: infatti, quando prima si definisce la funzione in un punto generico x, e poi la si definisce in generale, se poi la calcolo proprio in x *Mathematica*, che è un programma furbo, evita di calcolarsi la funzione, dato che ha già il risultato bello e pronto. Logico no? In questo modo, abbiamo evitato di mettere i valori iniziali nella pancia della funzione, che ne avrebbe complicato la definizione, tramite comandi If e così via. In realtà, si dovrebbe pure considerare il caso dei numeri interi negativi, ma si dovrebbe considerare il costrutto If, che considereremo più avanti.

Inoltre, abbiamo anche visto l'uso di Integer: in questo modo abbiamo detto a *Mathematica* che quell'argomento poteva essere soltanto intero. Naturalmente, abbiamo anche altri tipi di insiemi di numeri: Reals, Rationals, Algebraics, Complex e così via. In questa maniera possiamo imporre delle condizioni specifiche, evitando dei messaggi di errore che possono capitare quando abbiamo delle funzioni che accettano determinati tipi di argomenti.

Derivazione

Le funzioni, ovviamente, possono essere derivate: la funzione di derivazione in *Mathematica* è indicata con la lettera D:

`D[f, x]`	derivata (parziale) $\frac{\partial}{\partial x} f$
`D[f, x1, x2, ... ]`	derivata multipla $\frac{\partial}{\partial x_1} \frac{\partial}{\partial x_2} \ldots f$
`D[f, {x, n}]`	derivata di ordine superiore $\frac{\partial^n f}{\partial x^n}$
`Dt[f]`	differenziale totale $d f$
`Dt[f, x]`	derivata totale $\frac{d}{d x} f$

Data la natura simbolica di *Mathematica*, siamo in grado di effettuare derivazioni anche su funzioni non definite:

```
In[14]:= D[r[q], q]
Out[14]= r'[q]

In[15]:= D[q[t] p[t], t]
Out[15]= q[t] p'[t] + p[t] q'[t]
```

Tuttavia, riprendiamo le funzioni che avevamo scritto poco sopra

```
In[16]:= f[x_] := x^3 + Sin[Cos[x]]

In[17]:= g[x_, y_, z_] := x^2 + y^3 + z^4
```

e proviamo a derivarle:

```
In[18]:= D[f[x], x]
Out[18]= 3 x^2 - Cos[Cos[x]] Sin[x]

In[19]:= D[g[x, 5, z], z]
Out[19]= 4 z^3
```

Potete vedere come sia facile effettuare le derivazioni di funzioni che possono essere anche molto complesse. Ovviamente possiamo effettuare anche derivate di ordine superiore:

In[20]:=
```
D[x^n, {x, 10}]
```

Out[20]=
$$(-9+n)\,(-8+n)\,(-7+n)\,(-6+n)\,(-5+n)\,(-4+n)\,(-3+n)\,(-2+n)\,(-1+n)\,n\,x^{-10+n}$$

In aggiunta a questo, possiamo definire derivate miste di ordine superiore. Vediamo l'esempio, definendo la seguente funzione:

In[21]:=
```
par[x_, y_, z_] := Sin[x Cos[y]] / Log[x^2 y / x]
```

Una volta definita, possiamo calcolarci, per esempio $\partial_{(x,y)}\,\mathrm{par}[x, y, z]$ possiamo elencare semplicemente nel giusto ordine le variabili nel comando D:

In[22]:=
```
D[par[x, y, z], x, y] // FullSimplify
```

Out[22]=
$$\frac{1}{x\,y\,\mathrm{Log}[x\,y]^3}\,(-x\,\mathrm{Cos}[x\,\mathrm{Cos}[y]]\,\mathrm{Log}[x\,y]\,(\mathrm{Cos}[y]+y\,(-1+\mathrm{Log}[x\,y])\,\mathrm{Sin}[y])+(2+x^2\,y\,\mathrm{Cos}[y]\,\mathrm{Log}[x\,y]^2\,\mathrm{Sin}[y])\,\mathrm{Sin}[x\,\mathrm{Cos}[y]])$$

Possiamo anche definire la derivata totale con Dt

In[23]:=
```
Dt[x^n, x]
```

Out[23]=
$$x^n\left(\frac{n}{x}+\mathrm{Dt}[n, x]\,\mathrm{Log}[x]\right)$$

La derivata totale è la derivata della funzione quando ogni parametro dipende dall'argomento, quindi non esistono costanti. Sono importanti, per esempio, quando si effettua il calcolo delle sensibilità di un sistema.

Possiamo anche specificare, se non ci interessa la derivata totale, quali parametri devono essere costanti nel calcolo della derivata, e quali, invece, devono essere considerati in funzione della variabile di derivazione. Questo può essere fatto mediante l'opzione NonConstants, che specifica la lista di parametri della funzione che NON devono essere considerati costanti. Per esempio, definendo la funzione come segue:

In[24]:=
```
nc[x_, y_] := c d f[c x, y] y
```

Proviamo a fare la derivata in *x*:

In[25]:=
```
D[nc[x, y], x]
```

Out[25]=
$$c^2\,d\,y\,f^{(1,0)}[c\,x, y]$$

Tutti i parametri sono considerati costanti, come possiamo vedere dal risultato; adesso, invece imponiamo che il parametro c dipenda da *x*:

```
In[26]:= D[nc[x, y], x, NonConstants → {c}]

Out[26]= d y D[c, x, NonConstants → {c}] f[c x, y] +
          c d y (c + x D[c, x, NonConstants → {c}]) f^(1,0)[c x, y]
```

In questo caso si vede che dove compariva la c elaborata dalla derivata, adesso compare la derivata di questo parametro, permettendo di vedere come cambia il risultato...

Comunque, se le conoscete sarete ben lieti di poterle utilizzare, scommetto. Per quanto riguarda la derivazione, non c'è molto altro da dire, data la semplicità del comando.

Integrazione

Dopo aver definito l'integrale, il passo successivo è quello di vedere come si fa l'integrale, che di solito è più difficile da calcolare a mano, specie se si va oltre il semplice polinomio o prodotto di seni e coseni: possiamo definire l'integrale definito, indefinito, multiplo:

`Integrate[f, x]`	integrale indefinito $\int f\, d\,x$
`Integrate[f, x, y]`	integrale multiplo indefinito $\int d\,x\, d\,y\, f$
`Integrate[f, {x, xmin, xmax}]`	integrale definito $\int_{x_{min}}^{x_{max}} f\, d\,x$
`Integrate[f, {x, xmin, xmax}, {y, ymin, ymax}]`	integrale multiplo definito $\int_{x_{min}}^{x_{max}} d\,x \int_{y_{min}}^{y_{max}} d\,y\, f$

Vediamo subito alcuni esempi di utilizzo dell'integrazione:

```
In[27]:= Integrate[Cos[x]^2, x]

Out[27]= x/2 + 1/4 Sin[2 x]
```

```
In[28]:= Integrate[Exp[-x^2], {x, 0, Infinity}]

Out[28]= Sqrt[π]/2
```

Notate come potete facilmente, grazie al calcolo simbolico, calcolare risultati anche con intervallo di integrazione infinito, e come il risultato non sia approssimato, ma preciso. Naturalmente, se vi piacciono i risultati approssimati potete sempre farlo:

```
In[29]:= N[%]

Out[29]= 0.886227
```

Ma questo era solo per rinfrescarvi la memoria. Questo in quanto ci sono casi che non sono risolvibili, e che quindi *Mathematica* lascia come sono:

```
In[30]:= Integrate[x^x, {x, 1, 6}]
```

$$\text{Out[30]=} \int_1^6 x^x \, dx$$

In questo caso *Mathematica* non è in grado di risolvere analiticamente l'integrale (e se non ci riesce lui dubito che qualche altra cosa nell'universo ci riesca), comunque potete sempre, in questi casi, avere un'approssimazione del risultato, con la precisione che vi serve di più:

```
In[31]:= N[%, 60]
Out[31]= 17128.111274740640079114054836246527589100875605980576316917 4
```

Mathematica riesce anche ad integrare funzioni che richiedono l'uso di altre funzioni speciali, e che quindi è tremendamente difficile tentare di risolvere a mano:

```
In[32]:= Integrate[Log[1 - x^2] / x, x]
```

$$\text{Out[32]=} -\frac{1}{2}\,\text{PolyLog}[2, x^2]$$

```
In[33]:= Integrate[Sin[x^2], x]
```

$$\text{Out[33]=} \sqrt{\frac{\pi}{2}}\,\text{FresnelS}\left[\sqrt{\frac{2}{\pi}}\,x\right]$$

Possiamo anche integrare in regioni di spazio. Se imponiamo limiti solo per le variabili, integreremo in regioni rettangolari; in questo caso la regione è triangolare:

```
In[34]:= Integrate[x^2 + y^2, {x, 0, 1}, {y, 0, x}]
```

$$\text{Out[34]=} \frac{1}{3}$$

Bisogna fare attenzione al fatto che il raggio d'integrazione che scriviamo per ultimo è quello che viene calcolato prima. Infatti, se vogliamo scrivere la formula in forma tradizionale, possiamo usare l'opzione TraditionalForm selezionando l'espressione ed utilizzando con CTRL+SHIFT+T:

$$\int_0^1 \int_0^x (x^2 + y^2)\, dy\, dx$$

Inoltre, con il comando Boole possiamo anche definire altri tipi di regioni di integrazione che con gli indici normali non si possono indicare. Per esempio, questo è un modo per calcolare un integrale in un intervallo di integrazione circolare:

```
In[35]:= Integrate[Sin[x]^2 Boole[x^2 + y^2 ≤ 1], {x, -1, 1}, {y, -1, 1}]
```

$$\text{Out[35]=}\quad \frac{1}{2}\left(\pi - \pi\,\text{BesselJ}[1, 2]\right)$$

Come potete vedere, il comando di integrale permettere di risolvere casi anche abbastanza complicati. Tuttavia, come vedremo più avanti, dove non riesce il calcolo simbolico, riesce quello numerico.

- **Sommatorie e produttorie**

Con *Mathematica* possiamo risolvere facilmente anche produttorie e sommatorie particolarmente ostiche; il programma è in grado, nella maggior parte dei casi in cui qualsiasi sommatoria o produttoria converge, di dare il risultato esatto, anche quando ci sono espressioni particolarmente complicate e, anche in questo caso, dove non ce la fa il calcolo simbolico ce la fa quello numerico. Le funzioni per eseguire sommatorie e produttorie sono se seguenti:

`Sum[`f`, {`i`, `i_{min}`, `i_{max}`}]`	la sommatoria $\sum_{i=i_{min}}^{i_{max}} f$
`Sum[`f`, {`i`, `i_{min}`, `i_{max}`, `di`}]`	sommatoria con i crescente con passo di
`Sum[`f`, {`i`, `i_{min}`, `i_{max}`}, {`j`, `j_{min}`, `j_{max}`}]`	sommatoria nididicata $\sum_{i=i_{min}}^{i_{max}} \sum_{j=j_{min}}^{j_{max}} f$
`Product[`f`, {`i`, `i_{min}`, `i_{max}`}]`	produttoria $\prod_{i=i_{min}}^{i_{max}} f$

Ci sono casi banali, dove il valore iniziale e finale sono finiti, ed in questo caso *Mathematica* calcola semplicemente la somma normale:

```
In[36]:= Sum[x^i / i, {i, 1, 7}]
```

$$\text{Out[36]=}\quad x + \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} + \frac{x^5}{5} + \frac{x^6}{6} + \frac{x^7}{7}$$

Possiamo anche specificare il passo, se vogliamo

```
In[37]:= Sum[Sin[x], {x, 0, Pi / 2, Pi / 6}]
```

$$\text{Out[37]=}\quad \frac{3}{2} + \frac{\sqrt{3}}{2}$$

Una cosuccia interessante è che *Mathematica* è in grado i calcolare anche la sommatoria nel caso che i limiti dell'indice siano incognite:

In[38]:=
```
Sum[x^3 + x, {x, n}]
```

Out[38]=
$$\frac{1}{4} n (1 + n) (2 + n + n^2)$$

E, soprattutto, il programma è in grado di calcolare esattamente anche le sommatorie con indici che vanno ad infinito:

In[39]:=
```
Sum[(1 / 3)^x, {x, 3, Infinity}]
```

Out[39]=
$$\frac{1}{18}$$

Possiamo definire in maniera analoga le produttorie, dato che richiedono gli stessi argomenti

In[40]:=
```
Product[Log[x], {x, 2, 10}]
```

Out[40]=
```
Log[2] Log[3] Log[4] Log[5] Log[6] Log[7] Log[8] Log[9] Log[10]
```

Naturalmente potremmo preferire, in questo caso, un risultato numerico

In[41]:=
```
N[%]
```

Out[41]=
```
62.3216
```

In fondo, non c'è molto altro da dire su questo argomento; probabilmente perchè non ne ho mai avuto un bisogno spropositato per andare avanti con l'università. Ad ogni modo, credo che quanto detto basti per cominciare a lavorarci sopra.

- **Equazioni**

Scrittura di equazioni

Le equazioni sono probabilmente uno degli aspetti più importanti di *Mathematica*, forse perchè lo sono anche nello studio... Per scrivere le equazioni è necessario usare il simbolo del doppio uguale ==

In[42]:=
```
2 + 2 == 4
```

Out[42]=
```
True
```

In questo caso *Mathematica* verifica se l'espressione a sinistra è uguale a quella destra, ed in caso affermativo restituisce True, altrimenti, e qua la fantasia dilaga, restituisce False.

Si usa l'operatore == per non confonderlo con l'operatore = che invece ha il compito di assegnare valori alle variabili. Questo sempre per evitare ambiguità nella scrittura delle varie espressioni

Possiamo anche associare nomi alle equazioni, creando quindi variabili che le rappresentano:

```
In[43]:= eq = x^2 + x / Sin [x] == 0
```

$$\text{Out[43]= } x^2 + x \csc[x] == 0$$

In questo caso non si restituisce nè il valore True, e neanche False, perchè *Mathematica* non è in grado di definire univocamente se l'equazione è verificata oppure no. Dipende dal valore di x, e prima dobbiamo andare a sostituirlo.

Ovviamente ci sono eccezioni definite dalle identità:

```
In[44]:= x + x^2 == x^2 + x
Out[44]= True
```

In questo caso, qualsiasi sia il valore di x, l'equazione è sempre la stessa, e il programma riconosce l'identità.

Disequazioni

Mathematica non riconosce solo le equazioni, ma anche le disequazioni, naturalmente... Gli operatori relazionali sono in questo caso quelli classici:

$x == y$	uguaglianza
$x != y$	diseguaglianza ($x \neq y$)
$x > y$	maggiore di
$x >= y$	maggiore o uguale a ($x \geq y$)
$x < y$	minore di
$x <= y$	minore o uguale a ($x \leq y$)
$x == y == z$	uguaglianza globale
$x != y != z$	elementi distinti
$x > y > z$, etc.	strettamente decrescente, etc.

Le relazioni fra due numeri sono abbastanza ovvie:

```
In[45]:= 1 > 3
Out[45]= False

In[46]:= 1 < 3
Out[46]= True
```

In[47]:=
```
34 ≠ 3 ≠ 34
```

Out[47]=
```
False
```

Notate una piccola cosa: nelle versioni più recenti di *Mathematica* capita che alcuni simboli, come -> oppure != siano poi automaticamente convertiti nella forma tipograficamente corretta, com'è capitato qua sopra scrivendo le diseguaglianze. Naturalmente i casi in cui non è univocamente determinata la diseguaglianza non possono essere verificati da *Mathematica*, e questo è il caso delle incognite:

In[48]:=
```
3 x + y < z
```

Out[48]=
```
3 x + y < z
```

In questo caso il programma restituisce semplicemente la disequazione. Ovviamente ci sono metodi per risolverli, altrimenti che ci stiamo a fare qua?

Operatori logici

Mathematica, com'è facile intuire, dispone anche degli operatori logici:

`!`*p*	not (¬*p*)
p `&&` *q* `&&` …	and (*p* ∧ *q* ∧ …)
p `||` *q* `||` …	or (*p* ∨ *q* ∨ …)
`Xor[`*p*`,` *q*`, … ]`	exclusive or (also input as *p* ⊻ *q* ⊻ …)
`Nand[`*p*`,` *q*`, … ]` e `Nor[`*p*`,` *q*`, … ]`	nand e nor (also input as ⊼ and ⊽)
`If[`*p*`,` *then*`,` *else*`]`	restituisce *then*se *p* è `True`, ed *else* se *p* è `False`
`LogicalExpand[`*expr*`]`	espande espressioni logiche

Ovviamente, i casi ovvi sono risolti immediatamente da *Mathematica*:

In[49]:=
```
5 > 2 && (45 < 3 || 5 ≠ 2)
```

Out[49]=
```
True
```

In questo caso, fra parentesi è verificata soltanto una diseguaglianza, ma il risultato delle parentesi è True, perchè in mezzo c'è l'operatore Or: la prima disequazione è vera, per cui si fa l'And fra due espressioni True, ed il risultato, ovviamente, è True!!! Insomma, devo spiegarvi io come sono le relazioni logiche? Anche in questo caso, ovviamente, ci sono le relazioni che *Mathematica*, a meno di non avere altre informazioni, non riesce a gestire:

```
In[50]:= w || p && c

Out[50]= w || (p && c)
```

Si vede che, in questo caso, *Mathematica* lascia l'espressione così com'è, dato che non è in grado di capire se alle tre incognite corrispondono valori True o False. Possiamo anche vedere, dal risultato, che il programma riconosce le precedenze fra gli operatori logici: infatti, viene evidenziata sotto il segno di parentesi l'And logico, che quindi viene valutato prima dell'Or logico. Esattamente come faremmo noi nella stessa situazione. Possiamo anche effettuare le espansioni logiche delle espressioni logiche, esattamente come quelle, per dire, algebriche:

```
In[51]:= LogicalExpand[(a || b) && (c || d) || ! (v || t)]

Out[51]= (a && c) || (a && d) || (b && c) || (b && d) || (! t && ! v)
```

Tanto semplice quanto efficace. E, adesso che abbiamo capito come si scrivono le equazioni, vediamo di risolverle. In fondo, siamo qui principalmente per questo, no?

- **Risolvere le equazioni**

Equazioni semplici

Adesso, andiamo a vedere un poco più approfonditamente, quello che *Mathematica* ci può offrire: in fondo, non abbiamo fatto poi molto, fino ad adesso. A parte qualche caso, ci siamo limitati a capire come scrivere le nostre formulette delicate ed affascinanti (?!?) dentro il pancino di *Mathematica*. Vediamo ora di fargliele digerire e poi di restituirci il risultato dal suo pancino... MMMmmmmm... Forse il paragone non è stato dei più felici... mi sa che le cose che escono dal pancino non piacciano proprio a tutti....

Comunque, vediamo di cominciare. Il comando principale che si usa per risolvere le equazioni in *Mathematica* è il seguente:

`Solve[`*lhs* `==` *rhs*`, `*x*`]`	risolve l'equazione, dando i valori di *x*
x `/.` *solution*	usa la lista di regole per trovare il valore *x*
expr `/.` *solution*	usa la lista di regole per trovare il valore dell'espressione

Il comando Solve applica tutta una serie di regole per poter cercare di risolvere le equazioni e trovare i valori di x. Questo permette di ottenere soluzioni anche simboliche, invece di dover utilizzare algoritmi numerici per poter trovare un singolo valore numerico. Ovviamente ci sono casi in cui un'espressione analitica della soluzione non è possibile, ed in questo caso il comando Solve non è in grado di restituire l'espressione corretta. In questo caso occorre ricorrere alle soluzioni numeriche, per il semplice fatto che per quella particolare equazione non c'è altra alternativa. Supponiamo, per esempio, di avere la seguente equazione:

In[52]:=
```
eq = x^2 + a x + a == 0;
```

Vi siete ricordati che = significa assegnamento, mentre == serve per l'uguaglianza relazionale, e che si usa quest'ultima nelle equazioni, vero?

Ora che so che siete discepoli fedeli, possiamo andare avanti, andando ad usare il comando Solve per andare a risolvere l'equazione:

In[53]:=
```
Solve[eq, x]
```

Out[53]=
$$\left\{\left\{x \to \frac{1}{2}\left(-\sqrt{-4+a}\,\sqrt{a}-a\right)\right\},\ \left\{x \to \frac{1}{2}\left(\sqrt{-4+a}\,\sqrt{a}-a\right)\right\}\right\}$$

Possiamo anche andarci a trovare il valore a, se ci serve quello:

In[54]:=
```
Solve[eq, a]
```

Out[54]=
$$\left\{\left\{a \to -\frac{x^2}{1+x}\right\}\right\}$$

Mathematica gestisce con altrettanta facilità equazioni in cui sono invocate soluzioni con numeri complessi:

In[55]:=
```
Solve[x^4 + 3 x^2 + 2 == 0, x]
```

Out[55]=
$$\{\{x \to -i\},\ \{x \to i\},\ \{x \to -i\sqrt{2}\},\ \{x \to i\sqrt{2}\}\}$$

In generale, *Mathematica* è in grado di risolvere esattamente equazioni algebriche fino al quarto grado:

In[56]:=
```
Solve[a x^4 + b x^3 + c x^2 + d x + e == 0, x]
```

Out[56]=
$$\left\{\left\{x \to -\frac{b}{4a} - \frac{1}{2}\sqrt{\frac{b^2}{4a^2} - \frac{2c}{3a} + \left(2^{1/3}\left(c^2 - 3bd + 12ae\right)\right) \Big/ \left(3a\left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace + \sqrt{-4\left(c^2 - 3bd + 12ae\right)^3 + \left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace\right)^2}\right)^{1/3}\right) + \frac{1}{3\,2^{1/3}a}\left(\left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace + \sqrt{-4\left(c^2 - 3bd + 12ae\right)^3 + \left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace\right)^2}\right)^{1/3}\right)} - \frac{1}{2}\sqrt{\frac{b^2}{2a^2} - \frac{4c}{3a} - \left(2^{1/3}\left(c^2 - 3bd + 12ae\right)\right) \Big/ \left(3a\left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace + \sqrt{-4\left(c^2 - 3bd + 12ae\right)^3 + \left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace\right)^2}\right)^{1/3}\right) - \frac{1}{3\,2^{1/3}a}\left(\left(2c^3 - 9bcd + 27ad^2 + 27b^2e - 72ace + \right.\right.\right.\right.$$

$$\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big)-\left(-\frac{b^3}{a^3}+\frac{4\,b\,c}{a^2}-\frac{8\,d}{a}\right)\Big/\left(4\sqrt{\left(\frac{b^2}{4\,a^2}-\frac{2\,c}{3\,a}+(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)+\frac{1}{3\,2^{1/3}\,a}\left(\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)\right)\right)\right\},$$

$$\left\{x\to-\frac{b}{4\,a}-\frac{1}{2}\sqrt{\left(\frac{b^2}{4\,a^2}-\frac{2\,c}{3\,a}+(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)+\frac{1}{3\,2^{1/3}\,a}\left(\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)\right)+\frac{1}{2}\sqrt{\left(\frac{b^2}{2\,a^2}-\frac{4\,c}{3\,a}-(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)-\frac{1}{3\,2^{1/3}\,a}\left(\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)-\left(-\frac{b^3}{a^3}+\frac{4\,b\,c}{a^2}-\frac{8\,d}{a}\right)\Big/\left(4\sqrt{\left(\frac{b^2}{4\,a^2}-\frac{2\,c}{3\,a}+(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)+\frac{1}{3\,2^{1/3}\,a}\left(\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)\right)\right)\right\},$$

$$\left\{x\to-\frac{b}{4\,a}+\frac{1}{2}\sqrt{\left(\frac{b^2}{4\,a^2}-\frac{2\,c}{3\,a}+(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)+\frac{1}{3\,2^{1/3}\,a}\left(\left(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e+\sqrt{\left(-4\,(c^2-3\,b\,d+12\,a\,e)^3+(2\,c^3-9\,b\,c\,d+27\,a\,d^2+27\,b^2\,e-72\,a\,c\,e)^2\right)}\right)^{1/3}\right)\right)-\frac{1}{2}\sqrt{\left(\frac{b^2}{2\,a^2}-\frac{4\,c}{3\,a}-(2^{1/3}\,(c^2-3\,b\,d+12\,a\,e))\Big/\left(3\,a\,\left(2\,c^3-9\,b\,c\right.\right.\right.\right.$$

$$d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big) -$$
$$\frac{1}{3\,2^{1/3}\,a}\Big((2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big) + \Big(-\frac{b^3}{a^3} + \frac{4\,b\,c}{a^2} - \frac{8\,d}{a}\Big) \Big/$$
$$\Big(4\sqrt{\Big(\frac{b^2}{4\,a^2} - \frac{2\,c}{3\,a} + (2^{1/3}\,(c^2 - 3\,b\,d + 12\,a\,e)) \Big/ \Big(3\,a\,\big(2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\big)^{1/3}\Big) +$$
$$\frac{1}{3\,2^{1/3}\,a}\Big((2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big)\Big)\Big)\Big)\Big\},$$
$$\Big\{x \to -\frac{b}{4\,a} + \frac{1}{2}\sqrt{\Big(\frac{b^2}{4\,a^2} - \frac{2\,c}{3\,a} + (2^{1/3}\,(c^2 - 3\,b\,d + 12\,a\,e)) \Big/ \Big(3\,a\,\big(2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\big)^{1/3}\Big) + \frac{1}{3\,2^{1/3}\,a}$$
$$\Big((2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big)\Big) +$$
$$\frac{1}{2}\sqrt{\Big(\frac{b^2}{2\,a^2} - \frac{4\,c}{3\,a} - (2^{1/3}\,(c^2 - 3\,b\,d + 12\,a\,e)) \Big/ \Big(3\,a\,\big(2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\big)^{1/3}\Big) -$$
$$\frac{1}{3\,2^{1/3}\,a}\Big((2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big) + \Big(-\frac{b^3}{a^3} + \frac{4\,b\,c}{a^2} - \frac{8\,d}{a}\Big) \Big/$$
$$\Big(4\sqrt{\Big(\frac{b^2}{4\,a^2} - \frac{2\,c}{3\,a} + (2^{1/3}\,(c^2 - 3\,b\,d + 12\,a\,e)) \Big/ \Big(3\,a\,\big(2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\big)^{1/3}\Big) +$$
$$\frac{1}{3\,2^{1/3}\,a}\Big((2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e + \sqrt{\left(-4\,(c^2 - 3\,b\,d + 12\,a\,e)^3 + (2\,c^3 - 9\,b\,c\,d + 27\,a\,d^2 + 27\,b^2\,e - 72\,a\,c\,e)^2\right)}\Big)^{1/3}\Big)\Big)\Big)\Big)\Big\}\Big\}$$

Non vi mostro il risultato perchè è troppo lungo, ma la soluzione in questo caso è pure esatta. Andando a sostituire i valori a quello opportuni numerici, *Mathematica* restituisce il risultato corretto. Per equazioni algebriche con ordine massimo superiore al 4°, può capitare che il programmino non trovi le esatte soluzioni. In alcuni casi ci può anche riuscire:

In[57]:=
```
Solve[x^7 == 5, x]
```

Out[57]=
```
{{x → -(-5)^(1/7)}, {x → 5^(1/7)}, {x → (-1)^(2/7) 5^(1/7)}, {x → -(-1)^(3/7) 5^(1/7)},
 {x → (-1)^(4/7) 5^(1/7)}, {x → -(-1)^(5/7) 5^(1/7)}, {x → (-1)^(6/7) 5^(1/7)}}
```

Ma, in generale, non ci riesce, per il fatto che non esistono funzioni analitiche che permettano di trovare le soluzioni richieste;

In[58]:=
```
Solve[x^6 + x^5 + x^2 == -6 x - 1, x]
```

Out[58]=
```
{{x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 1]},
 {x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 2]},
 {x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 3]},
 {x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 4]},
 {x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 5]},
 {x → Root[1 + 6 #1 + #1^2 + #1^5 + #1^6 &, 6]}}
```

In questo caso, la soluzione restituita non è di facile comprensione, ed è di forma anche abbastanza criptica. Tuttavia, anche se le soluzioni non sono esprimibili in forma normale, *Mathematica* è riuscita a trovarle, per cui possiamo averne un'approssimazione numerica:

In[59]:=
```
N[%]
```

Out[59]=
```
{{x → -1.59062}, {x → -0.171551},
 {x → -0.661785 - 1.24153 i}, {x → -0.661785 + 1.24153 i},
 {x → 1.04287 - 0.874004 i}, {x → 1.04287 + 0.874004 i}}
```

Per il comando Solve valgono pure le considerazioni di precisione numerica: se utilizziamo valori con il punto, allora *Mathematica* passerà al calcolo numerico invece che quello simbolico: riprendendo l'equazione che avevamo scritto prima, possiamo aggiungere un punto

In[60]:=
```
Solve[x^4 + 3 x^2 + 2. == 0, x]
```

Out[60]=
```
{{x → 0. - 1. i}, {x → 0. + 1. i}, {x → 0. - 1.41421 i}, {x → 0. + 1.41421 i}}
```

Otteniamo soluzioni numeriche. Per cui, se nel vostro caso, per esempio, 1.64 non è un numero approssimato, ma è esattamente il vostro valore, se volete soluzioni esatte dovete scrivere questo numero sotto forma di frazione, in modo da comunicare a *Mathematica* che il coefficiente, se pure con la virgola, è esatto:

In[61]:=
```
Solve[1.64 x^2 - 3 x + 5 == 2, x]
```

Out[61]=
```
{{x → 0.914634 - 0.99635 i}, {x → 0.914634 + 0.99635 i}}
```

In[62]:=
```
Solve[(164 / 100) x^2 - 3 x + 5 == 2, x]
```

Out[62]=
$$\left\{\left\{x \to \frac{5}{82}\left(15 - i\sqrt{267}\right)\right\}, \left\{x \to \frac{5}{82}\left(15 + i\sqrt{267}\right)\right\}\right\}$$

Come vedete, un punto può fare la differenza...

Notate un attimo il modo con cui *Mathematica* restituisce le soluzioni. Riuscite a capire in che forma le da? Dai che lo sapete... Uff, va bene, ve lo dico io!!! Le soluzioni non sono date in forma di semplice lista di soluzioni, ma vengono date in forma di regole di sostituzione: Infatti, ponendo per esempio:

In[63]:=
```
eq = s x^2 + 6 x - 2 == 0
```

Out[63]=
$$-2 + 6x + s x^2 == 0$$

In[64]:=
```
Solve[eq, x]
```

Out[64]=
$$\left\{\left\{x \to \frac{-3 - \sqrt{9 + 2s}}{s}\right\}, \left\{x \to \frac{-3 + \sqrt{9 + 2s}}{s}\right\}\right\}$$

Possiamo, ad esempio,andarle a sostituire all'equazione originale:

In[65]:=
```
eq /. %
```

Out[65]=
$$\left\{-2 + \frac{6\left(-3 - \sqrt{9 + 2s}\right)}{s} + \frac{\left(-3 - \sqrt{9 + 2s}\right)^2}{s} == 0,\right.$$
$$\left.-2 + \frac{6\left(-3 + \sqrt{9 + 2s}\right)}{s} + \frac{\left(-3 + \sqrt{9 + 2s}\right)^2}{s} == 0\right\}$$

In[66]:=
```
Simplify[%]
```

Out[66]=
```
{True, True}
```

Avete seguito i calcoli? Dopo aver trovato le soluzioni, sono andato a sostituirle all'equazione. Al posto di risriverle, è bastato utilizzare l'operatore percento, per andare a sostituire al suo posto il risultato precedente, che era proprio la lista di regole. Dopo, andando a semplificare il risultato, abbiamo verificato che le equazioni sono effettivamente verificate. Questo, ovviamente, non è fine a se stesso! Che ci frega, con tutto il rispetto, andare a sostiruire le soluzioni all'equazione per vedere se viene verificata, se intanto sappiamo già che lo saranno? Potrebbe, al massimo, avere un qualche significato per soluzioni numeriche, ma questo lo vedremo in seguito.

In realtà, la scrittura di soluzioni come regole ha un suo significato. A parte il fatto che, se volessimo semplicemente la lista delle soluzioni, potremmo lasciarlo sempre così com'è e ricopiarlo sul

quaderno (...), oppure, se proprio vogliamo, definire velocemente una lista ti soluzioni in questo modo

```
In[67]:= Solve[eq, x]
```

$$\text{Out[67]=}\quad \left\{\left\{x \to \frac{-3-\sqrt{9+2s}}{s}\right\}, \left\{x \to \frac{-3+\sqrt{9+2s}}{s}\right\}\right\}$$

```
In[68]:= lista = x /. %
```

$$\text{Out[68]=}\quad \left\{\frac{-3-\sqrt{9+2s}}{s}, \frac{-3+\sqrt{9+2s}}{s}\right\}$$

In questo modo abbiamo ottenuto la lista di soluzioni che volevamo. In realtà, nei problemi reali, se ci servono le soluzioni per andare a verificare altre equazioni, oppure per andarle a sostituire da qualche altra parte, effettivamente è meglio utilizzare le regole, invece che direttamente le sostituzioni. In questo modo le soluzioni sono sempre pronte per essere sostituite dove vogliamo. *Mathematica* usa questo approccio perchè, data la sua potenza, molto difficilmente verrà usato per risolvere soltanto un'equazione, ma le sue soluzioni faranno parte di un sistema ben più complesso: calcolo con il metodo delle perturbazioni per trovare orbitali, stress di strutture complesse, e chi più ne ha più ne metta. Ripeto: se vi serve soltanto per risolvere calcolucci da strapazzo, compratevi una HP 49G+, che ha lo stesso prezzo della versione studenti di *Mathematica* e che vi potete portare pure agli esami!!!

Sistemi di equazioni

Mathematica è in grado di risolvere in modo rapido ed indolore anche sistemi di equazioni. Il comando principale che permette di farlo è sempre Solve, andando a sostituire all'equazioni ed alle incognite da trovare, le corrispetive liste che contengono tutte le equazioni e incognite che cerchiamo. Per esempio, possiamo definire le due equazioni:

```
In[69]:= eq1 = a x^2 - 2 x y + 5 == 0; eq2 = x - y == 0;
```

E adesso, possiamo risolvere per le incognite che cerchiamo:

```
In[70]:= Solve[{eq1, eq2}, {x, y}]
```

$$\text{Out[70]=}\quad \left\{\left\{y \to -\frac{\sqrt{5}}{\sqrt{2-a}}, x \to -\frac{\sqrt{5}}{\sqrt{2-a}}\right\}, \left\{y \to \frac{\sqrt{5}}{\sqrt{2-a}}, x \to \frac{\sqrt{5}}{\sqrt{2-a}}\right\}\right\}$$

Un altro modo alternativo di risolvere il sistema, è concatenare le equazioni in un'espressione logica:

In[71]:=
```
Solve[eq1 && eq2, {x, y}]
```

Out[71]=
$$\left\{\left\{y \to -\frac{\sqrt{5}}{\sqrt{2-a}}, x \to -\frac{\sqrt{5}}{\sqrt{2-a}}\right\}, \left\{y \to \frac{\sqrt{5}}{\sqrt{2-a}}, x \to \frac{\sqrt{5}}{\sqrt{2-a}}\right\}\right\}$$

Possiamo vedere come le soluzioni siano le stesse, anche se in questo caso possiamo avere soluzioni più generali, combinando opportunamente le equazioni, Tuttavia, al posto di Solve, può essere utile in questo caso quest'altro comando:

`Reduce[ {`*lhs*$_1$ `= =`*rhs*$_1$ `,` *lhs*$_2$`==`*rhs*$_2$`, … }, {`*x*`, `*y*`, … }]`	restituisce un set di equazioni semplificate, includendo tutte le possibili soluzioni

Per esempio, considerando le equazioni di prima:

In[72]:=
```
Reduce[eq1 || eq2, {x, y}]
```

Out[72]=
$$\left(x \neq 0 \&\& y == \frac{5 + a x^2}{2 x}\right) || y == x$$

Possiamo vedere che il comando Reduce tratta soluzioni più generali di Solve, e possiamo vedere anche che restituisce le soluzioni in modo diverso. Tuttavia, se ci servono le regole di sostituzione, possiamo utilizzare il seguente comando ToRules, che prende le soluzioni date da Reduce e le trasforma in regole:

In[73]:=
```
ToRules[%]
```

Out[73]=
$$\text{Sequence}\left[\left\{y \to \frac{5 + a x^2}{2 x}\right\}, \{y \to x\}\right]$$

Reduce può anche essere utilizzato per poter ottenere soluzioni più generali rispetto a Solve. Per esempio, la classica equazione di secondo grado:

In[74]:=
```
eq = a x^2 + b x + c == 0;
```

Risolta con Solve restituisce il seguente risultato;

In[75]:=
```
Solve[eq, x]
```

Out[75]=
$$\left\{\left\{x \to \frac{-b - \sqrt{b^2 - 4 a c}}{2 a}\right\}, \left\{x \to \frac{-b + \sqrt{b^2 - 4 a c}}{2 a}\right\}\right\}$$

Tuttavia, non contempla i casi particolari: per esempio, come facciamo a sapere a priori se $a \neq 0$ a priori? Le soluzioni di Solve sono date solo ammettendo che i calcoli necessari per trovare la soluzione non abbiano eccezioni. Risolviamo la stessa equazione con Reduce, invece:

In[76]:=
```
Reduce[eq, x]
```

Out[76]=
$$\left(a \neq 0 \&\& \left(x == \frac{-b - \sqrt{b^2 - 4\,a\,c}}{2\,a} \;||\; x == \frac{-b + \sqrt{b^2 - 4\,a\,c}}{2\,a}\right)\right) \;||\; \left(a == 0 \&\& b \neq 0 \&\& x == -\frac{c}{b}\right) \;||\; (c == 0 \&\& b == 0 \&\& a == 0)$$

Come posiamo vedere, questo comando permette di trattare anche i casi particolari delle soluzioni, scritti sotto forma di espressioni logiche che risultano vere. Infatti Reduce è nato proprio per risolvere espressioni logiche, e le equazioni, effettivamente, possono anche essere viste come tali, considerando == come un operatore logico. Quindi, il risultato si legge: "L'espressione logica (equazione) è vera quando $a \neq 0 \wedge x$ è dato da quelle soluzioni, oppure dagli altri casi. Possiamo anche in questo caso trasformare l'espressione logica in regole che saranno più facilmente gestibili

In[77]:=
```
ToRules[%]
```

Out[77]=
$$\text{Sequence}\left[\left\{x \to \frac{-b - \sqrt{b^2 - 4\,a\,c}}{2\,a}\right\}, \left\{x \to \frac{-b + \sqrt{b^2 - 4\,a\,c}}{2\,a}\right\}, \{a \to 0, x \to -\frac{c}{b}\}, \{c \to 0, b \to 0, a \to 0\}\right]$$

Un altro comando utile in ambito di sistemi di equazioni è il seguente:

`Eliminate[{`*lhs*$_1$`==`*rhs*$_1$`,` *lhs*$_2$`==`*rhs*$_2$`, … }, {`*x*`, … }]`	elimina *x*, … in un sistema di equazioni

Consideriamo, per esempio, il seguente sistema di equazioni:

In[78]:=
```
sis = {y == m x + m + 4, y == 2 x - 3};
```

Proviamo a risolverlo:

In[79]:=
```
Solve[sis, {x, y}]
```

Out[79]=
$$\left\{\left\{x \to -\frac{7 + m}{-2 + m}, y \to -\frac{8 + 5\,m}{-2 + m}\right\}\right\}$$

Possiamo provare ad eliminare, invece, la variabile y, e vedere l'equazione risultante:

In[80]:=
```
Eliminate[sis, y]
```

Out[80]=
$$(-2 + m)\,x == -7 - m$$

Vediamo che otteniamo l'equazione di una retta, di cui, adesso, possiamo andare a trovare la soluzione:

In[81]:=
```
Solve[%, x]
```

Out[81]=
$$\left\{\left\{x \to \frac{-7-m}{-2+m}\right\}\right\}$$

In alternativa, si poteva utilizzare direttamente la seguente formulazione di Solve:

In[82]:=
```
Solve[sis, x, y]
```

Out[82]=
$$\left\{\left\{x \to -\frac{7+m}{-2+m}\right\}\right\}$$

In questo caso, si è detto a *Mathematica* di risolvere il sistema sis nella variabile x. Inoltre, si è detto, mediante il terzo argomento, di risolverla eliminando la variabile y. Potevamo, per esempio, eliminare m, invece di y:

In[83]:=
```
Solve[sis, x, m]
```

Out[83]=
$$\left\{\left\{x \to \frac{3+y}{2}\right\}\right\}$$

Ed otteniamo il risultato di x in funzione di y. Si utilizza questo metodo perchè abbiamo due equazioni in tre incognite, e dobbiamo decidere di volta in volta rispetto a quali vogliamo risolvere il sistema, mantenendo l'altra incognita come parametro.

In generale, *Mathematica* ha qualche problema a trovare la soluzione di equazioni trigonometriche; questo è dovuto al fatto che di solito il programma inverte le funzioni, ma in questo modo le soluzioni dell'inverso delle funzioni trigonometriche non sonon univocamente determinate. *Mathematica* segnala questo problema con un Warning, anche s restituisce il risultato del valore principale dell'argomento della funzione trigonomentrica:

In[84]:=
```
Solve[Sin[x] == 2 / Sqrt[6], x]
```

```
— Solve::ifun :
   Inverse functions are being used by Solve, so some solutions may not
     be found; use Reduce for complete solution information. More…
```

Out[84]=
$$\left\{\left\{x \to \text{ArcSin}\left[\sqrt{\frac{2}{3}}\right]\right\}\right\}$$

Possiamo vedere che viene data la soluzione, anche se il Warning ci avverte del fatto che stiamo usando una funzione inversa, e che non possiamo trovare tutte le soluzioni.

Disequazioni

Mathematica è anche in grado di risolvere disequazioni, oltre che trovare soluzioni di equazioni e sistemi: i comandi principali sono due, e credo che uno già sapete qual'è, vero?

`Reduce[`*ineqs*`, {`*x*`, `*y*`, ... }]`	riduce un set di disequazioni
`FindInstance[`*ineqs*`, {`*x*`, `*y*`, ... }]`	trova delle istanze che soddisfano *ineqs*

Il metodo più semplice è quello di utilizzare il comando Reduce: abbiamo detto prima che serve per risolvere espressioni logiche, ed è quello che fa anche in questo caso:

In[85]:=
```
Reduce[x^2 - 5 < 0, x]
```

Out[85]=
$$-\sqrt{5} < x < \sqrt{5}$$

In[86]:=
```
Reduce[x + y > z && z^2 > 5, {x, y}]
```

Out[86]=
$$x \in \text{Reals} \;\&\&\; ((z < -\sqrt{5} \;\&\&\; y > -x + z) \;||\; (z > \sqrt{5} \;\&\&\; y > -x + z))$$

Notiamo come, per poter risolvere le disequazioni, *Mathematica* in questo caso particolare dica che x deve appartenere ai numeri reali, anche se di solito evita di scrivere queste avvertenze quando è possibile. Cominquq, non rende di certo meno valida la soluzione...

Con il comando FindInstance, invece, si trova un particolare valore della soluzione che soddisfino le disequazioni:

In[87]:=
```
FindInstance[Sqrt[x] > y && x + y^5 < 3, {x, y}]
```

Out[87]=
$$\left\{\left\{x \to \frac{1}{2},\ y \to -1\right\}\right\}$$

Vediamo se effettivamente sono verificate le diseguaglianze:

In[88]:=
```
Sqrt[x] > y && x + y^5 < 3 /. %
```

Out[88]=
```
{True}
```

Come possiamo vedere, tutto procede secondo i piani...

A volte è necessario, invece, trovare dei casi particolari di FindInstance. Potremmo, per esempio, trovare i valori che rendano massima e minima una funzione, entro determinati vincoli: per questo ci sono due funzioni apposite:

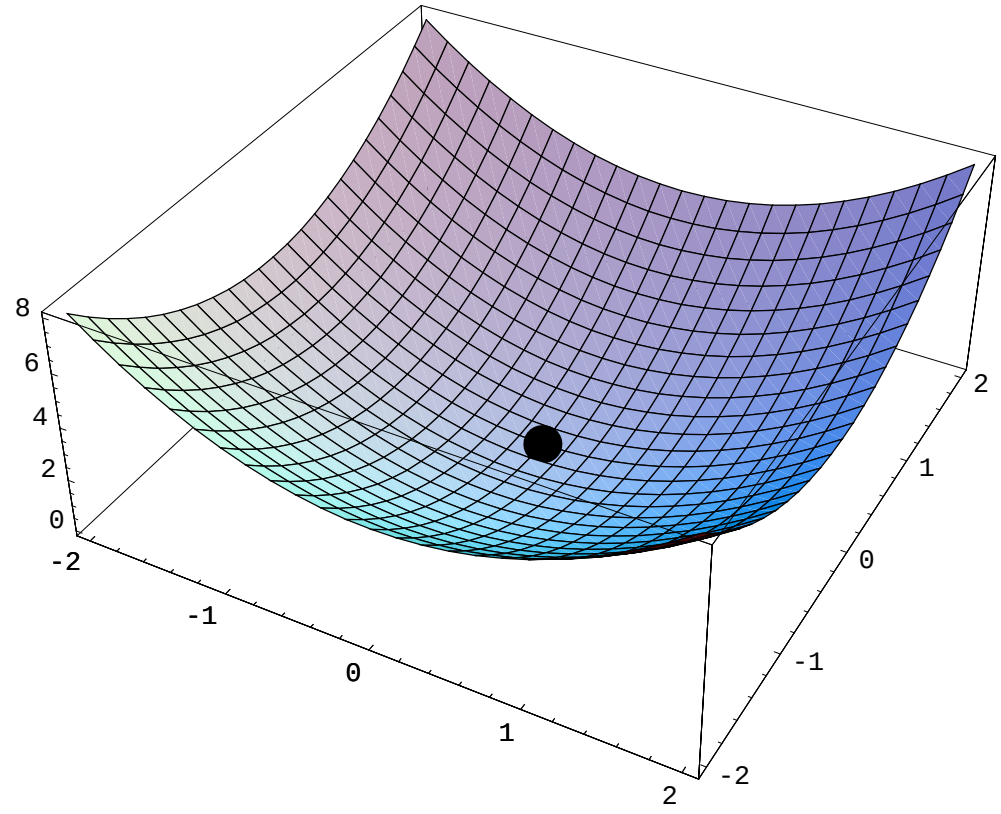
`Minimize[{`*expr*`, ` *ineq*`}, {`*x*`, ` *y*`, … }]`	minimizza *expr* sotto i vincoli imposti da *ineqs*
`Maximize[{`*expr*`, ` *ineq*`}, {`*x*`, ` *y*`, … }]`	se non lo capite da soli, è inutile che andate avanti...

Vediamo un attimino come funzionano:

In[89]:=
```
Minimize[{x^2 + y^2}, {x, y}]
```

Out[89]=
```
{0, {x → 0, y → 0}}
```

In questo caso, la funzione è stata minimizzata, e, dato che si tratta di un paraboloide, si vede che avrà un minimo all'origine. Il comando ha restituito una lista contenente sia il punto (x,y) in cui la funzione è minima, sia il valore della funzione assunta un quel punto.



Supponiamo, adesso, di dover invece calcolarci il valore minimo della funzione, ma solamente nei punti in cui il paraboloide interseca il piano $x + 3\,y - 2$:

In[90]:=
```
Minimize[{x^2 + y^2, x + 3 y - 2 == 0}, {x, y}]
```

Out[90]=
$$\left\{\frac{2}{5}, \left\{x \to \frac{1}{5}, y \to \frac{3}{5}\right\}\right\}$$

Come potete vedere, questa volta il minimo è stato vincolato nella curva di interstezione fra paraboloide e piano...

Vedremo che, quando non è possibile trovare soluzioni simboliche, *Mathematica* dispone di potentissimi comandi per poter trovare soluzioni numeriche anche dei sistemi di equazioni più improbabili che esistano.

Equazioni differenziali

Le equazioni differenziali sono, a mio avviso, uno dei punti forti di *Mathematica*. Qua affronteremo le soluzioni ed il calcolo simbolico, ma il comando per poter risolverle è uno dei più potenti, sia per il calcolo simbolico che per quello, come vedremo più avanti, numerico.

Quello che vediamo qua sotto è un semplice esempio di definizione di equazione differenziale:

```
In[91]:= diffeq = y''[x] + y[x] == a;
```

Come possiamo vedere, è abbastanza semplice poterle scrivere; si definisce la derivata di una funzione ponendo l'apice subito dopo il nome, prima delle parentesi che racchiudono l'argomento (o gli argomenti). Una volta definita, si tratta di risolverla, e possiamo usare il comando dedicato:

`DSolve[`*eqns*`, y[x], x]`	calcola la funzione `y[x]`, considerando `x` come variabile indipendente
`DSolve[`*eqns*`, y, x]`	restituisce la soluzione in forma di funzione pura

Proviamo a risolvele l'equazione differenziale di sopra:

```
In[92]:= DSolve[diffeq, y[x], x]

Out[92]= {{y[x] → a + C[1] Cos[x] + C[2] Sin[x]}}
```

Possiamo vedere come sia risolta velocemente: possiamo anche vedere come, dato che non abbiamo imposto nessuna condizione al contorno, che la soluzione è data in forma generale, con gli appositi coefficienti indeterminati, scritti sotto forma di *C*[*n*]. Volendo, possiamo anche imporre le condizioni al contorno, se le abbiamo:

```
In[93]:= DSolve[{y[x] + y''[x] == a, y'[0] == 4, y[2] == 0}, y[x], x]

Out[93]= {{y[x] → a - a Cos[x] Sec[2] + 4 Sin[x] - 4 Cos[x] Tan[2]}}
```

In questo modo, possiamo trovare la soluzione per il nostro problema specifico. Notate come, anche in questo caso, la soluzione è data sottoforma di regole.

Ovviamente *Mathematica* è anche in grado di risolvere equazioni differenziali semplici, che però danno soluzioni molto complicate. Consideriamo, per esempio, questo esempio:

In[94]:=
```
diffeq = y''[x] + x y[x] == a;
```

E risolviamola:

In[95]:=
```
DSolve[diffeq, y[x], x]
```

Out[95]=
$$\left\{\left\{y[x] \to \text{AiryAi}\left[(-1)^{1/3} x\right] C[1] + \text{AiryBi}\left[(-1)^{1/3} x\right] C[2] + \left(3\ (-1)^{2/3}\ 3^{5/6}\ a\ \pi\ x\ \text{AiryAi}\left[(-1)^{1/3} x\right] \text{Gamma}\left[\frac{1}{3}\right] \text{Gamma}\left[\frac{5}{3}\right] \text{HypergeometricPFQ}\left[\left\{\frac{1}{3}\right\}, \left\{\frac{2}{3}, \frac{4}{3}\right\}, -\frac{x^3}{9}\right] - 3\ (-1)^{2/3}\ 3^{1/3}\ a\ \pi\ x\ \text{AiryBi}\left[(-1)^{1/3} x\right] \text{Gamma}\left[\frac{1}{3}\right] \text{Gamma}\left[\frac{5}{3}\right] \text{HypergeometricPFQ}\left[\left\{\frac{1}{3}\right\}, \left\{\frac{2}{3}, \frac{4}{3}\right\}, -\frac{x^3}{9}\right] - 3\ 3^{1/6}\ a\ \pi\ x^2\ \text{AiryAi}\left[(-1)^{1/3} x\right] \text{Gamma}\left[\frac{2}{3}\right]^2 \text{HypergeometricPFQ}\left[\left\{\frac{2}{3}\right\}, \left\{\frac{4}{3}, \frac{5}{3}\right\}, -\frac{x^3}{9}\right] - 3^{2/3}\ a\ \pi\ x^2\ \text{AiryBi}\left[(-1)^{1/3} x\right] \text{Gamma}\left[\frac{2}{3}\right]^2 \text{HypergeometricPFQ}\left[\left\{\frac{2}{3}\right\}, \left\{\frac{4}{3}, \frac{5}{3}\right\}, -\frac{x^3}{9}\right]\right) \Big/ \left(27\ \text{Gamma}\left[\frac{2}{3}\right] \text{Gamma}\left[\frac{4}{3}\right] \text{Gamma}\left[\frac{5}{3}\right]\right)\right\}\right\}$$

Proviamo a semplificarla:

In[96]:=
```
FullSimplify[%]
```

Out[96]=
$$\left\{\left\{y[x] \to \text{AiryAi}\left[(-1)^{1/3} x\right] C[1] + \text{AiryBi}\left[(-1)^{1/3} x\right] C[2] + a\ x^2\ \text{Hypergeometric0F1}\left[\frac{4}{3}, -\frac{x^3}{9}\right] \text{HypergeometricPFQ}\left[\left\{\frac{1}{3}\right\}, \left\{\frac{2}{3}, \frac{4}{3}\right\}, -\frac{x^3}{9}\right] - \frac{1}{2}\ a\ x^2\ \text{Hypergeometric0F1}\left[\frac{2}{3}, -\frac{x^3}{9}\right] \text{HypergeometricPFQ}\left[\left\{\frac{2}{3}\right\}, \left\{\frac{4}{3}, \frac{5}{3}\right\}, -\frac{x^3}{9}\right]\right\}\right\}$$

Come potete vedere, in questo caso la soluzione non è proprio semplice come al solito. Provate a farlo a mano, se ci riuscite...

Possiamo anche utilizzare le funzioni 'pure':

In[97]:=
```
DSolve[y'[x] == x + y[x], y, x]
```

Out[97]=
$$\{\{y \to \text{Function}[\{x\}, -1 - x + e^x\ C[1]]\}\}$$

In questo caso, si vede che il risultato è leggermente diverso. Tuttavia, le funzioni pure sono un aspetto abbastanza avanzato per noi, per cui vi basti sapere che esistano per adesso, anche se andremo a vederle meglio più avanti. Se volete studiarvele, al solito, andatevi a leggere l'aiuto in linea, oppure proseguite spediti verso le vie tortuose che vi sto indicando.

- **Serie di potenze**

Fino ad adesso abbiamo considerato sempre funzione e soluzioni esatte. Tuttavia, come molti di voi sicuramente sapranno, molte volte è più utile utilizzare delle approssimazioni delle funzioni, scrivendole come serie di potenze. Sicuramente avrete sentito parlare, a magari pure usato, le formule di Taylor e di MacLaurin. Ricordo che in Analisi 1 era una delle cose che avevo odiato di più, ma effettivamente mi rendo conto che sono delle cose estremamente utili per noi... Un esempio semplice semplice? Come studiate i circuiti elettronici a piccolo segnale, senza linearizzazione? E da cosa viene la teoria della linearizzazione? Bravi, non lo avete detto ad alta voce (almeno spero di no), ma lo avete pensato.

Mathematica offre comandi (ma và?) anche per questo tipo di problema:

`Series[`*expr*`, {`*x*`, `x_0`, `*n*`}]`	trova l'espansione in serie di *expr* intorno al punto $x = x_0$ fino all'ordine *n*
`Normal[`*series*`]`	tronca l'espressione in serie in una espressione normale

Magari vi starete chiedendo a cosa serve Normal, vero? No? Pazienza, ve lo spiego comunque...

Consideriamo, per cominciare, l'esempio classico dell'espansione in serie, cioè l'esponenziale:

```
In[98]:= Series[Exp[x], {x, 0, 7}]
```

$$\text{Out[98]= } 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040} + O[x]^8$$

Come potete vedere, *Mathematica* ha effettuato l'espanzione, nel punto 0, troncando al 6° termine, che corrisponde alla derivata di ordine 7, come certamente saprete. Tuttavia, compara anche un altro termine, $O[x]$. Questo è un infinitesimo. In pratica, dice a *Mathematica* che l'espressione che abbiamo trovato non è esatta, ma è troncata e ha un infinitesimo di ordine specificato (nel nostro caso, 6° ordine). Questo ha una conseguenza molto importante: dato che il programma sa, in questo caso, che l'espressione trovata è approssimata, quando andremo ad effettuare operazioni su di essa, darà sempre risultati con la giusta precisione di infinitesimi. E' come se voi avete un numero con due cifre decimali, ne fate un'operazione e vi spunta un numero con 15 cifre decimali. Se il numero iniziale era approssimato, è normale che delle 15 cifre, 13 sicuramente non avranno significato... Lo stesso concetto si può applicare agli infinitesimi. Questo si può notare specialmente quando compaiono nelle operazioni termini che avrebbero come risultato espressioni di grado superiore. Supponiamo, per esempio, di prendere il nostro risultato precedente e di elevarlo al quadrato:

```
In[99]:= %^2
```

$$\text{Out[99]= } 1 + 2x + 2x^2 + \frac{4x^3}{3} + \frac{2x^4}{3} + \frac{4x^5}{15} + \frac{4x^6}{45} + \frac{8x^7}{315} + O[x]^8$$

In questo caso, vediamo quanto detto poc'anzi: se non avessimo considerato l'infinitesimo, avremmo avuto un'espressione di quattordicesimo grado. Ciò non era giustificabile, in quanto il grado massimo della nostra approssimazione era il settimo, e tutti gli altri erano soggetti ad un errore eccessivo. Se, invece, vi serve l'espressione che avete ottenuto con l'espansione in serie completa, cioè se non volete, per qualche motivo, considerare l'infinitesimo, allora con il comando Normal lo potete eliminare (anche un copia ed incolla solo della parte dell'espressione che volete funziona, per dircela tutta, ma non va se volete automatizzare i calcoli):

In[100]:= `Normal[%%]`

Out[100]= $1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040}$

Come vedete l'espressione (notate che ho preso in considerazione il penultimo risultato con %%), è senza infinitesimo, e possiamo quindi elevarla al quadrato per ottenere tutti i termini:

In[101]:= `%^2`

Out[101]= $\left(1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040}\right)^2$

Compare così perchè è la forma più semplice; per averla in forma espansa basta usare il comando che già conoscete:

In[102]:= `Expand[%]`

Out[102]= $1 + 2x + 2x^2 + \frac{4x^3}{3} + \frac{2x^4}{3} + \frac{4x^5}{15} + \frac{4x^6}{45} + \frac{8x^7}{315} + \frac{127x^8}{20160} + \frac{41x^9}{30240} + \frac{19x^{10}}{75600} + \frac{x^{11}}{25200} + \frac{19x^{12}}{3628800} + \frac{x^{13}}{1814400} + \frac{x^{14}}{25401600}$

Et voilà! Pronta per essere cotta a vapore!!!

- **Limiti**

Non c'è bisogno che vi spiega cosa sono i limiti, vero? Risparmiatemi tutta la parte sulle differenze finite e sul rapporto incrementale e su tutto il resto.

Il comando più importante (credo l'unico) che *Mathematica* ha per calcolarsi i limiti è il seguente:

`Limit[`*expr*`, `*x*`->`x_0`]` limite *expr* quando *x* tende a x_0

Cosa c'è di meglio per un classico esempio di limite della Sinc?

In[103]:=
```
sinc = Sin[x] / x
```

Out[103]=
$$\frac{\text{Sin}[x]}{x}$$

Come potete intuire, se cerchiamo di calcolarci la funzione in 0, *Mathematica* ci prende per scemi, ci riderà in faccia e creerà un virus che ci distruggerà il computer e ci farà venire l'orticaria:

In[104]:=
```
sinc /. x → 0
```

— *Power::infy : Infinite expression* $\frac{1}{0}$ *encountered.* More…

— ∞*::indet : Indeterminate expression 0 ComplexInfinity encountered.* More…

Out[104]=
```
Indeterminate
```

Come potete vedere, tutto quello che avevo predetto si è realizzato. Meno male che avevo comprato un computer di riserva prima di eseguire questo comando...

Comunque, abbiamo visto che non è definito, ma sappiamo anche che esiste il limite:

In[105]:=
```
sinc /. x → 0.02
```

Out[105]=
```
0.999933
```

Proviamo, quindi, a calcolarci il limite di questa espressione (non è una funzione, vi ricordo, perchè non l'abbiamo definita come tale. Tuttavia Limit funziona, naturalmente, anche con le funzioni...):

In[106]:=
```
Limit[sinc, x → 0]
```

Out[106]=
```
1
```

Non ve lo sareste mai aspettati, vero?

Comunque, i limiti non finiscono certo qua, come purtroppo sappiamo tutti... Consideriamo, per esempio, questa funzione:

In[107]:=
```
f[x_] := Sqrt[x^2 - 6 x + 9] / (x - 3)
```

Come possiamo vedere, la funzione presenta una discontinuità nel punto 3. Proviamo a calcolarci il limite in questo punto:

In[108]:=
```
Limit[f[x], x → 3]
```

Out[108]=
```
1
```

Il risultato è esatto solamente a metà. In questi casi, possiamo utilizzare l'opzione Direction:

In[109]:=
```
Limit[f[x], x → 3, Direction → -1]
```

Out[109]=
```
1
```

In[110]:=
```
Limit[f[x], x → 3, Direction → 1]
```

Out[110]=
```
-1
```

Se scriviamo Direction $\to -1$, il limite sarà calcolato a partire da valori più grandi, dando quindi il limite destro, mentre se poniamo Direction $\to 1$, allora calcoleremo il limite a partire da valori più piccoli, cioè il limite sinistro.

Possiamo naturalmente, nidificando i limiti, trovare limiti per più variabili. Tuttavia bisogna stare attenti alla direzione da cui si ci arriva. Consideriamo quest'altra funzione classica, che tutti avete fatto in analisi:

In[111]:=
```
g[x_, y_] := x^2 y / (x^2 + y^2)
```

Vediamo che, a seconda della direzione con cui la funzione tende ad infinito, la funzione tenderà a 0 oppure a ∞. Si può vedere questo considerando l'ordine dei limite

In[112]:=
```
Limit[Limit[g[x, y], y → Infinity], x → Infinity]
```

Out[112]=
```
0
```

In[113]:=
```
Limit[Limit[g[x, y], x → Infinity], y → Infinity]
```

Out[113]=
```
∞
```

Come potete vedere, le cose cambiano. Tutavia questi sono problemi propri dell'analisi e della funzione che prendiamo in considerazione, non del calcolo di *Mathematica*.

- **Trasformate integrali**

Vediamo un altro aspetto di *Mathematica*, adesso: quello di saper lavorare velocemente e con buoni risultati con le trasformate integrali, quali quella di Fourier e quella di Laplace:

`LaplaceTransform[`*expr*`, `*t*`, `*s*`]`	trasformata di Laplace di *expr*
`InverseLaplaceTransform[`*expr*`, `*s*`, `*t*`]`	trasformata inversa di Laplace di *expr*
`FourierTransform[`*expr*`, `*t*`, `*w*`]`	trasformata di Fourier di *expr*
`InverseFourierTransform[`*expr*`, `*w*`, `*t*`]`	trasformata inversa di Fourier di *expr*

Il funzionamento di queste funzioni è abbastanza semplice: si scrive l'espressione, poi la variabile dell'espressione che rappresenta quella indipendente, ed infine la nuova variabile:

In[114]:=
```
LaplaceTransform[Sin[x] Cos[x], x, s]
```

Out[114]=
$$\frac{1}{4+s^2}$$

In[115]:=
```
InverseLaplaceTransform[%, s, x]
```

Out[115]=
```
Cos[x] Sin[x]
```

In[116]:=
```
LaplaceTransform[3 x (x - 9) Sin[x], x, s]
```

Out[116]=
$$3\left(-\frac{18\,s}{(1+s^2)^2}+\frac{-2+6\,s^2}{(1+s^2)^3}\right)$$

E andiamo via così all'infinito. Lo stesso possiamo fare, naturalmente, anche per le trasformate di Fourier:

In[117]:=
```
FourierTransform[Sin[x] Cos[x], x, w]
```

Out[117]=
$$\frac{1}{2}\,i\,\sqrt{\frac{\pi}{2}}\,\text{DiracDelta}[-2+w]-\frac{1}{2}\,i\,\sqrt{\frac{\pi}{2}}\,\text{DiracDelta}[2+w]$$

Notate come il programma sia anche in grado di gestirsi espressioni con la Delta di Dirac. Questo perchè *Mathematica* ha un approccio simbolico (quante volte lo avrò ripetuto?), e non è, in fondo, nient'altro che una funzione, e quindi viene trattata come tale. In fondo, non c'è veramente molto altro da dire riguardo questo argomento... Fatevi qualche esempio e vedete da soli la potenza ed i limiti di questi comandi.