Eq differenziali

Prima di cominciare inizializziamo caricando il package per visualizzare i grafici:

```
Needs["Graphics`Graphics`"];
Needs["Graphics`Colors`"];
Needs["Graphics`Animation`"];
Needs["DifferentialEquations`NDSolveProblems`"];
Needs["DifferentialEquations`NDSolveUtilities`"];
```

Introduzione

In quest'appendice vedremo meglio uno degli aspetti migliori (a mio avviso) di *Mathematica*, ossia i suoi comandi e le sue opzioni per la risoluzione di equazioni differenziali, sia in forma simbolica, che in forma numerica. Per esempio, possiamo trattare con estrema semplicità sia le equazioni, che i sistemi di equazioni, anche di tipo misto, dove alcune equazioni del sistema sono differenziali, mentre altre sono algebriche. Inoltre, possiamo trattare anche le equazioni differenziali alle derivate parziali. Insomma, possiamo fare veramente di tutto. Tuttavia, se le equazioni differenziali che trattiamo sono alquanto complicate, dobbiamo cominciare a cercare di capire meglio come funzionano questi comandi, vedendoli in maniera più dettagliata. Il fatto che risolvano da soli la maggior parte dei problemi è sicuramente una grande cosa, ma dobbiamo comunque tener conto del fatto che mettere mano alle opzioni permette di avere un controllo maggiore su quello che si fa, e permette di capire anche il funzionamento di certi algoritmi: inoltre potremmo anche essere in grado di selezionare manualmente le opzioni e gli algoritmi migliori per il problema che ci serve.

Tipi di equazioni

Il comanid DSolve è in grado di risolvere sia singole equazioni differenziali, che sistemi di equazioni.

- ***Equazioni differenziali ordinarie*** **ODE**: sono i tipi di equazioni in cui la variabile indipendente è una sola, mentre quelle dipendenti $y_i(x)$ possono essere una soltanto, nel caso di una sola equazione differenziale, oppure più di una, nel caso in cui abbiamo a che fare con dei sistemi di equazioni differenziali.

- ***Equazioni differenziali alle derivate parziali*** **PDE**: questo tipo di equazioni differenziali si distinguono dalle prime per il fatto che la loro funzione (o variabile dipendente) dipende da più di una variabile indipendente $y(t_1, t_2, ...)$. Questi casi sono più difficili da risolvere simbolicamente rispetto alle ODE: *Mathematica* non è oggettivamente in grado di poter trovare soluzioni a qualsiasi equazione PDE, per il semplice fatto che non esiste una teoria che permetta di calcolarle in maniera esatta. Il problema è talmente difficile che ci sono, per fare un esempio, dei casi in cui ci sono premi per chi riesce a trovare soluzioni esatte per questo tipo di equazioni differenziali; un esempio è dato dal milione di dollari messo in palio (da chi adesso non ricordo, sinceramente), per chi riesca a

trovare soluzioni in forma simbolica delle equazioni di Navier-Stokes, per la particolare importanza che rivestono in moltissimi campi dell'ingegneria.
Mathematica è in grado di risolvere PDE in maniera simbolica nella maggior parte dei casi in cui l'equazione è del primo ordine, ed in un minor numero di casi quando la PDE è di ordine superiore.

- ***Equazioni algebrico-differenziali* DAE**: si tratta in particolar modo di sistemi, in cui compaiono sia equazioni differenziali, che equazioni algebriche, in cui non compaiono le derivate. Per questo tipo di sistemi valgono gli stessi problemi delle PDE, cioè che non esiste una teoria tanto robusta da poter trovare in tutti i casi le soluzioni in maniera simbolica; anche in questo caso *Mathematica* non è in grado di trovare le soluzioni per casi difficili, ma molti problemi sono comunque alla portata di questo programma.

Risoluzione Simbolica

Adesso vedremo meglio, più in dettaglio, e con un maggior numero di esempi, il funzionamento del comando DSolve. La soluzione simbolica, è sempre da preferire, quando è possibile averla, perchè naturalmente non abbiamo in questo caso problemi tipici del calcolo numerico, come errori di approssimazione, necessità di specificare sempre le condizioni iniziali, la scelta dell'intervallo di risoluzione e così via. In casi avanzati le soluzioni possono occupare anche diverse pagine, ma fin quando lasciamo tutto il lavoro di elaborazione a *Mathematica*, non dobbiamo preoccuparcene più di tanto, lasciando al programma il compito di elaborare le funzioni ottenute.

■ DSolve

Come ben sappiamo, è questo il comando che dobbiamo utilizzare quando vogliamo trovare la soluzione di un'equazione differenziale in forma simbolica. Data la natura, appunto, simbolica della soluzione, se si omettono le condizioni iniziali, *Mathematica* darà il risultato in forma generale, con le opportune costanti, che di default sono date nella forma C[n]. Per poter scrivere la derivata, basta utilizzare il carattere ' (quello che vi trovate accanto lo zero nella parte superiore della tastiera, per capirci), come se si trattasse del simbolo che si usa nei libri; per poter creare derivate di ordine superiore basta mettere tanti ' per quanto è necessario. Vediamo questo esempio:

```
DSolve[y''[x] + x y[x] == 0, y[x], x]
```

```
{{y[x] → AiryAi[(-1)^(1/3) x] C[1] + AiryBi[(-1)^(1/3) x] C[2]}}
```

Possiamo vedere due cose, che in fondo già sapevamo (ma ricordare non fa mai male, vero, cari miei folletti che volano sulle favole dell'ingegneria? Ehm....).
Prima di tutto, occorre notare come viene data la soluzione, cioè sotto forma di regola, invece che direttamente come funzione. Questo permette di andare a sostituire la funzione dove serve. Inoltre, come caso più generale, permette anche di andare a sostituire più funzioni in una volta sola, nel caso

avessimo più soluzioni. per andare a sostituirla basta usare ReplaceAll, che come sapete si può scrivere anche come /.

```
y[x] /. %
```

$$\{\text{AiryAi}[(-1)^{1/3}\,x]\,C[1] + \text{AiryBi}[(-1)^{1/3}\,x]\,C[2]\}$$

Viene restituita come lista, anche se è formata da un solo elemento, per manterene la generalità della soluzione: infatti, dato che le soluzioni possono essere più di una, si racchiudono in una lista, e questa è semplicemente il caso particolare in cui abbiamo un solo elemento.

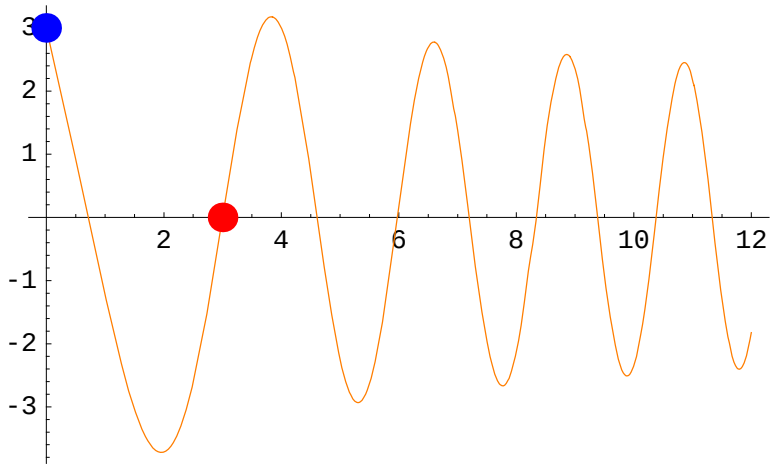
La seconda cosa da notare è come abbiamo introdotto le costanti, che *Mathematica* ha aggiunto automaticamente, in mancanza delle opportune condizioni iniziali. Se le avessimo imposte, avremmo anche ottenuto la soluzione particolare:

```
DSolve[{y''[x] + x y[x] == 0, y[0] == 3, y[3] == 0}, y[x], x] // FullSimplify
```

$$\left\{\left\{y[x] \to \left(3\,3^{2/3}\,(-\text{AiryAi}[-x]\,\text{AiryBi}[-3] + \text{AiryAi}[-3]\,\text{AiryBi}[-x])\,\text{Gamma}\left[\frac{2}{3}\right]\right) \Big/ \left(2\,\text{BesselJ}\left[\frac{1}{3}, 2\sqrt{3}\right]\right)\right\}\right\}$$

In questo caso abbiamo posto le condizioni iniziali, ed abbiamo ottenuto la soluzione particolare:

```
Plot[y[x] /. %, {x, 0, 12}, PlotStyle → {Orange},
 Epilog → {
   {Blue, PointSize[0.04], Point[{0, y[x] //. {%[[1, 1]], x → 0}}]},
   {Red, PointSize[0.04], Point[{3, y[x] //. {%[[1, 1]], x → 3}}]}
  }
]
```




```
- Graphics -
```

Come possiamo vedere, abbiamo aggiunto al grafico dei punti che rappresentano la funzione calcolata nelle condizioni iniziali, e possiamo vedere come coindidano con quelle che avevamo imposto in partenza, cosa che effettivamente era naturale. Guai ad non essere così. Potevo anche semplicemente utilizzare le coordinate {3, 0} e {0, 3} per rappresentare direttamente i punti, ma in questa maniera vi ho fatto vedere come ottenere lo stesso risultato lavorando con le funzioni e con le regole di sostituzione.

Potete anche vedere come abbiamo imposto le condizioni iniziali nel comando DSolve, cioè creando sintatticamente un sistema di equazioni simile al DAE, dove però c'è l'importante differenza che, invece di equaioni delle variabili dipendenti generiche, compaiono equazioni per valori particolari della funzione. Ovviamente, le condizioni iniziali devono essere compatibili con il problema. Sempre nell'esempio fatto, infatti, abbiamo bisogno di due condizioni iniziali per avere la soluzione esatta; un numero maggiore porta a generare un errore:

```
DSolve[{y''[x] + x y[x] == 0, y[0] == 3, y[3] == 0, y'[0] == 3}, y[x], x]
```

```
— DSolve::bvnul : For some branches of the general solution,
    the given boundary conditions lead to an empty solution. More…

{}
```

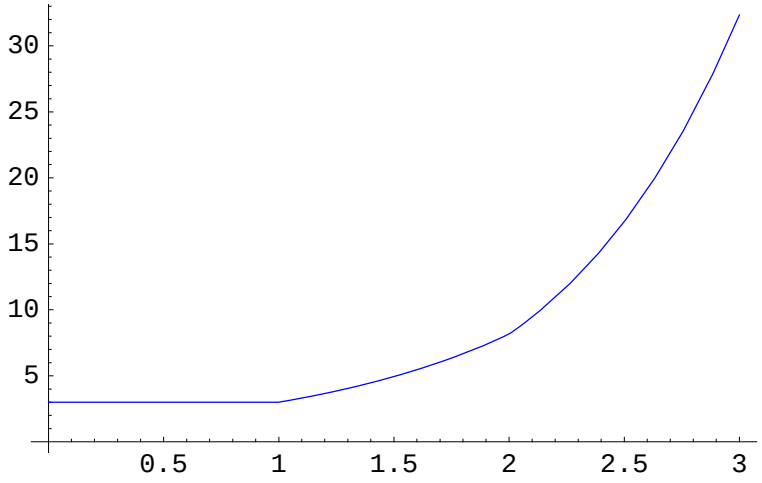
Questo perchè le condizioni iniziali sono inconsistenti con li problema. Ovviamente, in questo caso, l'errore non è del programma, ma è vostro, ed è un modo per dirvi di andare a studiare...

Un'importante aspetto del comando DSolve, specialmente dalla versione 5 di *Mathematica*, è la sua capacità di lavorare con funzioni definite a tratti, che quindi possono presentare delle discontinuità, come la funzione UnitStep:

```
DSolve[
 {y[t] UnitStep[t - 1] + t^2 UnitStep[t - 2] == y'[t], y[0] == 3}, y[t], t]
```

$$\left\{\left\{y[t] \to -\frac{2\,e^2 - 10\,e^t - 3\,e^{1+t} + 2\,e^2\,t + e^2\,t^2}{e^2} + \left(3\,e^{-1+t} + \frac{2\,e^2 - 10\,e^t - 3\,e^{1+t} + 2\,e^2\,t + e^2\,t^2}{e^2}\right) \text{UnitStep}[2 - t] + \text{UnitStep}[1 - t]\,(3 - 3\,e^{-1+t}\,\text{UnitStep}[2 - t])\right\}\right\}$$

```
Plot[y[t] /. %[[1]], {t, 0, 3}, PlotStyle → {Blue}]
```




```
- Graphics -
```

Potete notare i punti di discontinuità che si hanno per $t = 1$, $t = 2$, che sono i punti dove iniziano i gradini nella nostra equazione differenziale.

Le funzioni definite a tratti possono essere anche utilizzate nei sistemi, come in questo caso:

```
sistema = {
   UnitStep[t - 3] y'[t] - x[t] t + UnitStep[-t + 3] x'[t] t^2 == x[t],
   y'[t] == t x'[t],
   x'[1] == 4,
   y[0] == 4
  };
```

Per comodità, oltre al sistema, ho specificato anche le condizioni iniziali nel sistema stesso: adesso posso andare a risolverlo normalmente, utilizzando sempre DSolve:

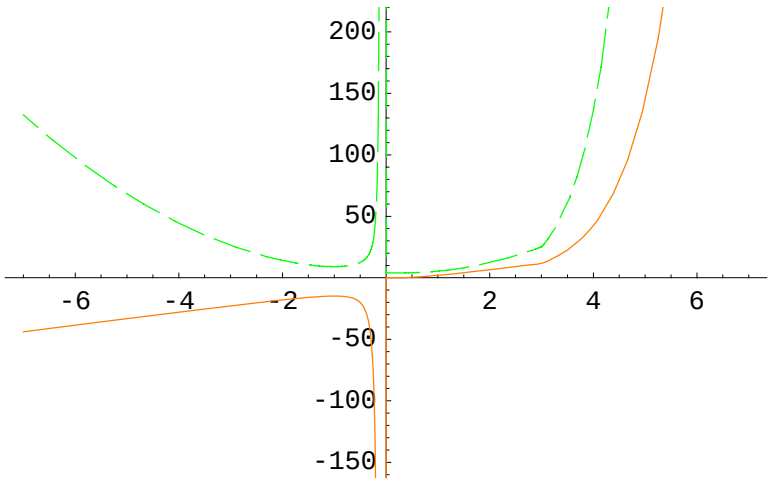
```
DSolve[sistema, {x[t], y[t]}, t] // FullSimplify
```

$$\left\{\left\{x[t] \to 2\, e^{1+\begin{cases} -\frac{1}{t}+\text{Log}[t] & t\le 3 \\ -\frac{10}{3}+t+\text{Log}[t] & \text{True} \end{cases}},\right.$$

$$\left. y[t] \to \begin{cases} 4 - 2\, e^{2/3} + 2\, e^{-\frac{7}{3}+t}\,(1 + (-1 + t)\,t) + e\,\text{ExpIntegralEi}[-\frac{1}{3}] & t > 3 \\ 4 + e^{1-\frac{1}{t}}\,t\,(1 + t) + e\,\text{ExpIntegralEi}[-\frac{1}{t}] & \text{True} \end{cases}\right\}\right\}$$

Notate la potenza del comando, che risulta in grado di definire funzioni a tratti anche nell'argomento dell'esponenziale, e come la soluzione includa funzioni avanzate come ExpIntegralEi

```
Plot[{x[t] /. %, y[t] /. %}, {t, -7, 7},
 PlotStyle → {{Orange}, {Dashing[{0.05, 0.015}], Green}}]
```




```
- Graphics -
```

In maniera analoga, possiamo risolvere anche equazioni differenziali alle derivate parziali, andando a specificare sempre quali sono le variabili indipendenti. Considerando la seguente equazione differenziale

```
eqn = x y^2 D[u[x, y], x] + y^2 x^2 D[u[x, y], y] - x y u[x, y]
```

$$-x\,y\,u[x,\,y] + x^2\,y^2\,u^{(0,1)}[x,\,y] + x\,y^2\,u^{(1,0)}[x,\,y]$$

```
sol = DSolve[eqn == 0, u[x, y], {x, y}]
```

$$\left\{\left\{u[x,\,y] \to e^{\frac{2\,\text{ArcTan}\left[\frac{x}{\sqrt{-x^2+2\,y}}\right]}{\sqrt{-x^2+2\,y}}}\,C[1]\left[\frac{1}{2}\,(-x^2+2\,y)\right]\right\}\right\}$$

Da questa scrittura possiamo notare subito un paio di cose importanti. Prima di tutto, possiamo notare il diverso tipo di notazione, quando dobbiamo scrivere delle PDE. infatti, scrivendo qualcosa come *u*'[*x*, *t*], *Mathematica* (e neanche noi, a dire il vero), non è in grado di distinguere la variabile rispetto alla quale stiamo derivando; per questo motivo, dobbiamo per forza di cose andare ad esplicitare tramite il comando D il tipo di derivata parziale che andiamo ad utilizzare. Comparirà in maniera corretta nell'output, come possiamo andare a vedere.

Inoltre, i coefficienti C[i], in questo caso, non rappresentano più delle costanti, ma sono a tutti gli effetti delle funzioni. Si può notare considerando che C[1], nel nostro caso, è una funzione, avente come argomento $\frac{1}{2}(-x^2+2\,y)$, come possiamo vedere dalla soluzione che abbiamo ottenuto. Dalla soluzione generale, quindi, se vogliamo arrivare ad una particolare, dobbiamo andare a sostituire a C[i] delle funzioni (che naturalmente, a seconda delle condizioni al contorno, possono essere anche

delle costanti...). Possiamo trovare una soluzione particolare del nostro problema, ipotizzando per esempio, di mettere come caso particolare il seno:

```
solpar = u[x, y] /. %[[1]] /. {C[1][t_] → Sin[t]}
```

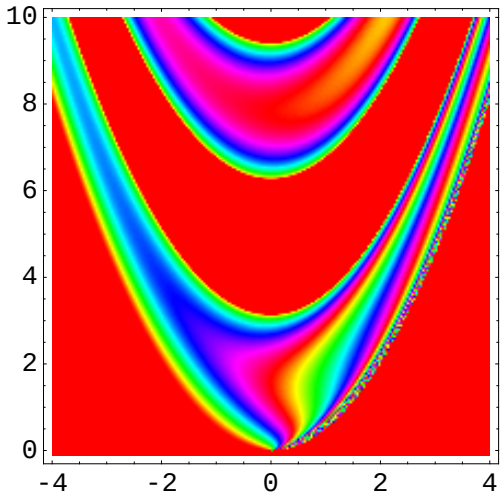
$$e^{\frac{2\,\text{ArcTan}\left[\frac{x}{\sqrt{-x^2+2y}}\right]}{\sqrt{-x^2+2y}}}\,\text{Sin}\left[\frac{1}{2}\,(-x^2+2y)\right]$$

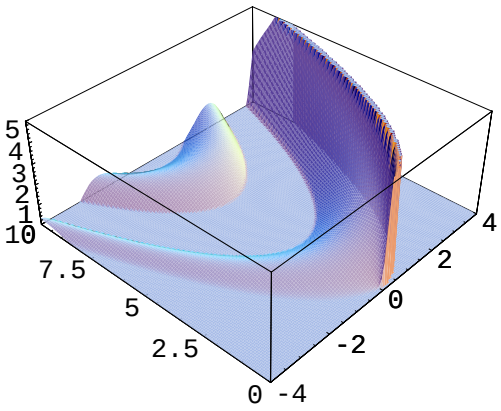
Notiamo come sia stato necessario preservare la struttura dell'argomento della costante C[i], in quanto è quello necessario per la risoluzione del problema. Adesso possiamo andare a visualizzare la funzione, notando che ottengo una soluzione reale soltanto quando $-x^2+2y \geq 0$, cioè quando è reale il radicale, considerato che non ho limiti per quanto riguarda l'argomente del seno e dell'Arc-Tan, sempre che siano ovviamente numeri reali :-). Per cui otterremmo degli errori nel caso in cui specifichiamo durante il plottaggio, dei punti al di fuori questo dominio. Per risolvere il problema allora andiamo a plottare una funzione Piecewise, che facciamo corrispondere alla nostra funzione se rientra nel dominio, e la poniamo uguale a 0 se invece cade fuori i nostri limiti permessi:

```
dominio = Reduce[-x^2 + 2 y > 0, {x, y}]
```

$$x \in \text{Reals}\ \&\&\ y > \frac{x^2}{2}$$

```
DisplayTogetherArray[
 {{
   DensityPlot[Piecewise[{{solpar, dominio}, {0, True}}],
    {x, -4, 4}, {y, -.1, 10},
    PlotPoints → 200,
    Mesh → False,
    ColorFunctionScaling → False,
    ColorFunction → (Hue[Sqrt[#]] &)],
   Plot3D[Piecewise[{{solpar, dominio}, {0, True}}],
    {x, -4, 4}, {y, -.1, 10},
    PlotPoints → 130,
    Mesh → False,
    PlotRange → {Automatic, Automatic, {0, 5}},
    ViewPoint -> {-2.004, -1.853, 2.000}]
  }}
]
```






```
- GraphicsArray -
```

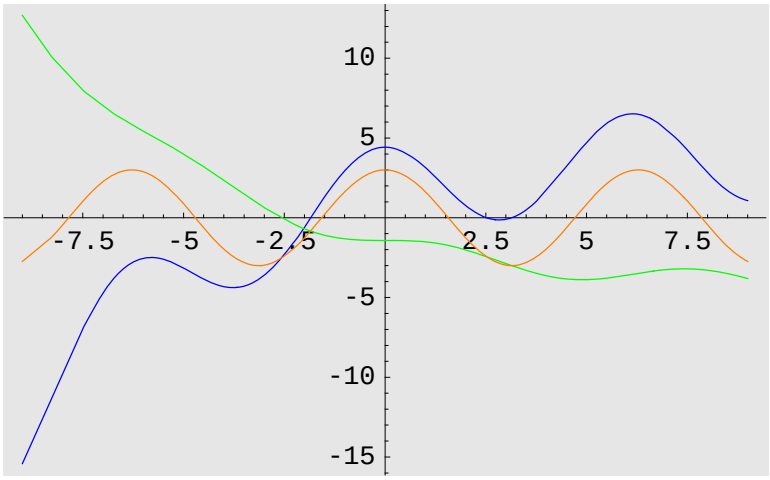
Abbiamo visto come possiamo risolvere sia le ODE che le PDE. Naturalmente, possiamo risolvere anche le DAE, sempre nei casi in cui è permesso. Possiamo considerare il seguente sistema:

```
dae = {
   f'[x] == 5 g''[x],
   f[x] + g[x] == 3 Cos[x],
   f[Pi] == 0, f'[0] == 0
  };
```

```
soluzione = DSolve[dae, {f, g}, x]
```

$$\left\{\left\{f \to \text{Function}\left[\{x\},\ \frac{15}{26}\, e^{-\frac{\pi}{5}-\frac{x}{5}}\left(-5\, e^{\pi/5}+5\, e^{\frac{\pi}{5}+\frac{x}{5}}+5\, e^{x/5}+5\, e^{\frac{\pi}{5}+\frac{x}{5}}\,\text{Cos}[x]-e^{\frac{\pi}{5}+\frac{x}{5}}\,\text{Sin}[x]\right)\right],\right.$$
$$\left.g \to \text{Function}\left[\{x\},\ \frac{3}{26}\, e^{-\frac{\pi}{5}-\frac{x}{5}}\left(25\, e^{\pi/5}-25\, e^{\frac{\pi}{5}+\frac{x}{5}}-25\, e^{x/5}+e^{\frac{\pi}{5}+\frac{x}{5}}\,\text{Cos}[x]+5\, e^{\frac{\pi}{5}+\frac{x}{5}}\,\text{Sin}[x]\right)\right]\right\}\right\}$$

```
Plot[Evaluate[{f[x], g[x], f[x] + g[x]} /. soluzione], {x, -9, 9},
 PlotStyle → {{Blue}, {Green}, {Orange}},
 Background → GrayLevel[0.9]
]
```




```
- Graphics -
```

Possiamo notare come la loro somma dia effettivamente $3\cos(x)$, come avevamo scritto nel sistema misto.

Il modo di lavorare di DSolve è abbastanza complicato ed oscuro per la maggior parte degli esseri umani (diciamo tutti apparte la ventina di persone che l'hanno programmato). Tuttavia imparare ad usarlo bene, permette di inserire l'input in una maniera che può anche agevolare il comando a lavorare.

Quello che faremo adesso è quello di andare ad analizzare un tantino più in dettaglio quello che possiamo ottenere con DSolve, e capire un pochino meglio le equazioni differenziali.

ODE

Sono le equazioni che si incontrano per la prima volta, perchè corrispondono a quelle più semplici da risolvere. Il caso più diretto è un'equazione differenziale di questo tipo:

$$y'(t) = f(t)$$

Questo tipo di equazione differenziale si risolve semplicemente integrando la parte destra dell'equazione stessa:

```
DSolve[y'[t] == Sin[t]^2, y[t], t]
```

$$\left\{\left\{y[t] \to \frac{t}{2} + C[1] - \frac{1}{4}\,\text{Sin}[2\,t]\right\}\right\}$$

La soluzione è semplice e diretta. Dato la natura di questo tipo di equazione, potevamo risolverla anche in questa maniera:

```
Integrate[Sin[t]^2, t]
```

$$\frac{t}{2} - \frac{1}{4}\,\text{Sin}[2\,t]$$

Ottenendo esattamente lo stesso risultato; l'unica differenza risiede nel fatto che Integrate non mette direttamente la costante C (tutti gli integrali indefiniti dovrebbero essere in teoria come $C + F(t)$), mentre l'equazione differenziale l'aggiunge, perchè varia al variare delle condizioni al contorno della soluzione generica

```
DSolve[y'[x] == x Sin[x] y[x]^2 / Sqrt[3 - x], y, x]
```

$$\{\{y \to \text{Function}[\{x\},\ (4\,e^{3\,i+i\,x}) \Big/ (-2\,e^{3\,i}\sqrt{3-x} - 2\,e^{3\,i+2\,i\,x}\sqrt{3-x} - 4\,e^{3\,i+i\,x}\,C[1] - (1+6\,i)\,(-1)^{3/4}\,e^{i\,x}\sqrt{\pi}\,\text{Erfi}[(-1)^{1/4}\sqrt{3-x}] - (1-6\,i)\,(-1)^{1/4}\,e^{6\,i+i\,x}\sqrt{\pi}\,\text{Erfi}[(-1)^{3/4}\sqrt{3-x}])]\}\}$$

Adesso abbiamo invece un esempio di equazione omogenea:

```
omo = y'[x] == -(x^4 - 3 y[x]^3) * (x * y[x]);

sol = DSolve[omo, y, x] // FullSimplify
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, -\frac{(-2)^{1/3}\,(x^6)^{1/9}}{\left(2\,e^{\frac{x^6}{2}}\,(x^6)^{1/3}\,C[1] + 3\,2^{1/3}\,e^{\frac{x^6}{2}}\,x^2\,\text{Gamma}\left[\frac{1}{3}, \frac{x^6}{2}\right]\right)^{1/3}}\right]\right\},$$

$$\left\{y \to \text{Function}\left[\{x\}, \frac{2^{1/3}\,(x^6)^{1/9}}{\left(2\,e^{\frac{x^6}{2}}\,(x^6)^{1/3}\,C[1] + 3\,2^{1/3}\,e^{\frac{x^6}{2}}\,x^2\,\text{Gamma}\left[\frac{1}{3}, \frac{x^6}{2}\right]\right)^{1/3}}\right]\right\},$$

$$\left\{y \to \text{Function}\left[\{x\}, \frac{(-1)^{2/3}\,2^{1/3}\,(x^6)^{1/9}}{\left(2\,e^{\frac{x^6}{2}}\,(x^6)^{1/3}\,C[1] + 3\,2^{1/3}\,e^{\frac{x^6}{2}}\,x^2\,\text{Gamma}\left[\frac{1}{3}, \frac{x^6}{2}\right]\right)^{1/3}}\right]\right\}\}$$

Come vediamo, abbiamo tre funzioni che risolvono questa equazione differenziale. Effettivamente, però, le tre funzioni coincidono:

```
f1 = Table[Abs[y[x]] /. sol[[1]] /. C[1] → p, {p, -5, 5, .5}];
```

```
f2 = Table[Abs[y[x]] /. sol[[2]] /. C[1] → p, {p, -5, 5, .5}];
```

```
f3 = Table[Abs[y[x]] /. sol[[3]] /. C[1] → p, {p, -5, 5, .5}];
```

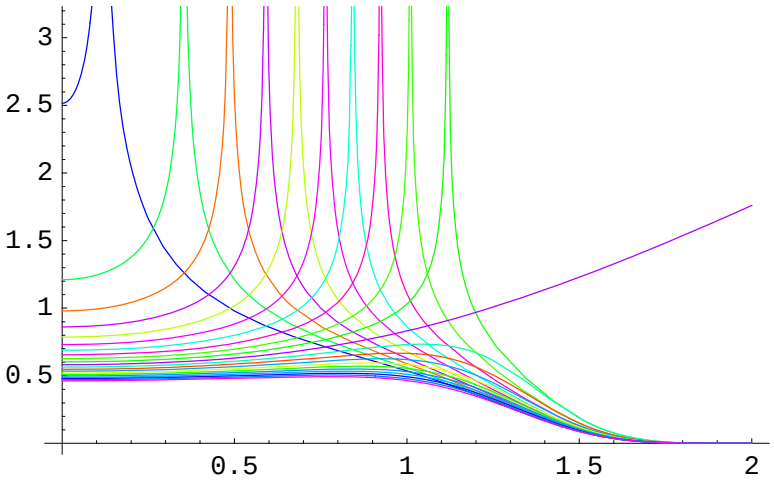
```
f1 == f2 == f3
```

```
True
```

Abbiamo visto che sono uguali, indipendentemente dal parametro, e basta quindi plottare soltanto una famiglia di equazioni, diciamo f1:

```
colori = PlotStyle → Table[{Hue[Random[]]}, {p, 0, 20}];
```

```
Plot[Evaluate[f1, {x, 0, 2}, colori]];
```



Piccola particolarità: non è necessatio che PlotStyle definisca un numero di stili uguale al numeri di funzioni, perchè viene ripetuto ciclicamente.

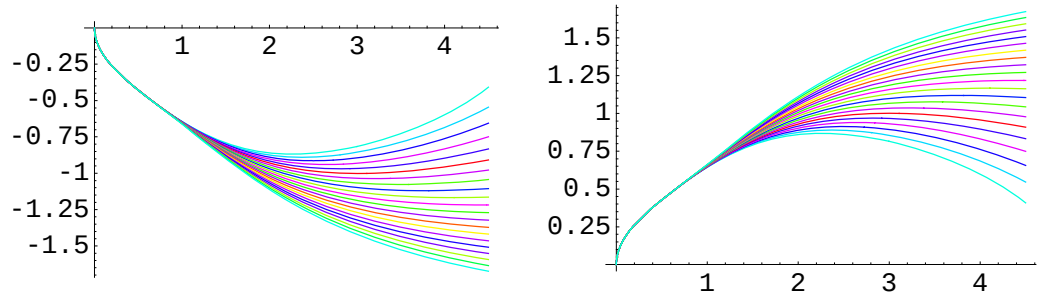
Detto questo, abbiamo visto come ottenere le soluzioni generali. Questo può essere più utile di quanto sembri. Infatti, possiamo avere vari rami della soluzione generale. Per qualcuno di essi, può non valere la condizione iniziale che vogliamo. Ipotizziamo di avere questa equazione differenziale:

```
rami = y'[x] == -(x - 3 y[x]^2) / (x^2 * y[x]);
```

```
sol = DSolve[rami, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, -\sqrt{e^{-6/x}\left(C[1] + 2\,\text{ExpIntegralEi}\left[\frac{6}{x}\right]\right)}\right]\right\},\right.$$
$$\left.\left\{y \to \text{Function}\left[\{x\}, \sqrt{e^{-6/x}\left(C[1] + 2\,\text{ExpIntegralEi}\left[\frac{6}{x}\right]\right)}\right]\right\}\right\}$$

```
r1 = Table[y[x] /. sol[[1]] /. C[1] → p, {p, -5, 5, .5}];

r2 = Table[y[x] /. sol[[2]] /. C[1] → p, {p, -5, 5, .5}];

DisplayTogetherArray[{
  Plot[Evaluate[r1, {x, 0, 4.5}, colori]],
  Plot[Evaluate[r2, {x, 0, 4.5}, colori]]
 }]
```




```
- GraphicsArray -
```

In questo esempio abbiamo evidenziato come, ad un'equazione differenziale, possano corrispondere più rami della soluzione, definite da due (o più, a seconda dei casi) funzioni. Vediamo quello che succede adesso:

```
DSolve[{rami, y[2] == 2}, y, x]
```

```
— DSolve::bvnul : For some branches of the general solution,
    the given boundary conditions lead to an empty solution. More…
```

$$\left\{\left\{y \to \text{Function}\left[\{x\},\right.\right.\right.$$
$$\left.\left.\left.\sqrt{2}\sqrt{e^{-6/x}\left(2\,e^{3} - \text{ExpIntegralEi}[3] + \text{ExpIntegralEi}\left[\frac{6}{x}\right]\right)}\right]\right\}\right\}$$

In questo caso abbiamo avuto un warning... La soluzione generale, infatti. è formata da due rami ma, dal comportamento che abbiamo visto nei grafici, vediamo come solamente uno dei due rami sia in grado di soddisfare la condizione al contorno che abbiamo imposto. Il warning sta appunto a dirci che, nonostante *Mathematica* abbia trovato due soluzioni indipendenti per l'equazione differenziale,

soltanto una viene utilizzata per trovare quella particolare.

In altre occasioni, invece, *Mathematica* riesce a trovare la soluzione, per così dire, in maniera parziale; questo può capitare perchè, ad esempio, durante i calcoli si ritrova un integrale che non riesce a risolvere in maniera analitica, come nell'esempio seguente:

```
integ = x y'[x] Sin[x]^4 == y[x] Cos[x];
```

```
DSolve[integ, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, e^{\int_1^x -\frac{8 \cos[K\$101533]}{K\$101533\,(-3+4\cos[2\,K\$101533]-\cos[4\,K\$101533])}\,dK\$101533}\,C[1]\right]\right\}\right\}$$

Come possiamo vedere, nella soluzione compare un integrale. Questo succede perchè *Mathematica* se lo è ritrovato davanti, senza essere in grado di risolverlo. La variabile utilizzata per l'integrale è una variabile locale, di quelle che il programma si crea da solo. Se vogliamo, possiamo andarlo a sostituire con un'altra variabile, per rendere la soluzione un tantinello più leggibile:

```
% /. K$101533 → t
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, e^{\int_1^x -\frac{8 \cos[t]}{t\,(-3+4\cos[2\,t]-\cos[4\,t])}\,dt}\,C[1]\right]\right\}\right\}$$

Naturalmente, la soluzione non è stata calcolata completamente. Per avere il valore della funzione in un punto, occorre sempre valutare numericamente l'integrale, quindi l'intera espressione

```
y[3] /. % /. C[1] → 1 // N
```

$$\{3.39381 \times 10^{-18}\}$$

Oltre agli integrali, nel comando DSolve occorre anche risolvere delle equazioni, in maniera simbolica, mediante il comando Solve. Quando quest'ultimo non è in grado di lavorare al meglio, *Mathematica* ce lo segnala:

```
errSolve = Sin[y'[x]] x == y[x];
```

```
DSolve[errSolve, y, x]
```

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

— Solve::tdep : The equations appear to involve the variables
     to be solved for in an essentially non-algebraic way. More…

— Solve::tdep : The equations appear to involve the variables
     to be solved for in an essentially non-algebraic way. More…

— Solve::tdep : The equations appear to involve the variables
     to be solved for in an essentially non-algebraic way. More…

— General::stop :
   Further output of Solve::tdep will be suppressed during this calculation. More…
```

$$\text{Solve}\left[\int_1^{\frac{y[x]}{x}} \frac{1}{K\$112646 - \text{ArcSin}[K\$112646]}\, dK\$112646 == C[1] - \text{Log}[x],\ y[x]\right]$$

In questo caso, non viene dato il risultato completo, ma neanche quello che avevamo in partenza. Quello che abbiamo ottenuto è un'espressione che coinvolge il comando Solve, perchè non è stato in grado di dare la soluzione cercata, restituendocelo così com'è. Anche in questo caso, come nel precedente, per avere soluzioni occorrono altre manipolazioni, e risoluzioni per via numerica, dato che non si possono trovare soluzioni in forma chiusa (dubito che possa risultare utile un risultato dove comare Solve ed un integrale non calcolabile simbolicamente). A questo punto conviene usare NDSolve, ma ne parleremo nell'opportuna sezione.

Un altro tipo di equazioni differenziali che *Mathematica* è in grado di risolvere sono quelle di Bernoulli:

```
bern = y'[x] + 3 x y[x] == x^4 y[x]^4;
```

```
solbern = DSolve[bern, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, -\frac{3\,(-2)^{1/3}}{\left(6x + 18x^3 + 54\, e^{\frac{9x^2}{2}} C[1] - e^{\frac{9x^2}{2}} \sqrt{2\pi}\, \text{Erf}\left[\frac{3x}{\sqrt{2}}\right]\right)^{1/3}}\right]\right\},\right.$$
$$\left\{y \to \text{Function}\left[\{x\}, \frac{3\,2^{1/3}}{\left(6x + 18x^3 + 54\, e^{\frac{9x^2}{2}} C[1] - e^{\frac{9x^2}{2}} \sqrt{2\pi}\, \text{Erf}\left[\frac{3x}{\sqrt{2}}\right]\right)^{1/3}}\right]\right\},$$
$$\left.\left\{y \to \text{Function}\left[\{x\}, \frac{3\,(-1)^{2/3}\,2^{1/3}}{\left(6x + 18x^3 + 54\, e^{\frac{9x^2}{2}} C[1] - e^{\frac{9x^2}{2}} \sqrt{2\pi}\, \text{Erf}\left[\frac{3x}{\sqrt{2}}\right]\right)^{1/3}}\right]\right\}\right\}$$

Naturalmente, avendole trovate, sono le soluzioni dell'equazione differenziale, e possiamo verificarlo facilmente:

```
bern /. solbern // Simplify
```

```
{True, True, True}
```

Un altro tipo di equazoni differenziali, particolarmente difficili da risolvere, sono le equazioni di Riccati, che sono nella forma $y'(x) = f(x) + g(x)\,y(x) + h(x)\,y(x)^2$. Qua sotto potete vederne un esempio:

```
riccati = y'[x] + 1 / x^3 - 3 x^3 y[x]^2 == 0;
```

```
DSolve[riccati, y, x]
```

```
{{y → Function[{x},
    -(2 x BesselY[2, -i √3 x] - 1/2 i √3 x^2 (BesselY[1, -i √3 x] -
          BesselY[3, -i √3 x]) + (2 x BesselJ[2, -i √3 x] - 1/2 i √3
          x^2 (BesselJ[1, -i √3 x] - BesselJ[3, -i √3 x])) C[1]) /
      (3 x^3 (x^2 BesselY[2, -i √3 x] + x^2 BesselJ[2, -i √3 x] C[1]))]}}
```

Un altro tipo di equazioni differenziali difficili da risolvere sono quelle di Abel, che vengono poste nella seguente forma: $y'(x) = f(x) + g(x)\,y(x) + h(x)\,y(x)^2 + k(x)\,y(x)^3$

```
abel = y'[x] == y[x]^3 - x y[x]^2 / (x - 1);
```

```
DSolve[abel, y, x]
```

```
- InverseFunction::ifun : Inverse functions are
    being used. Values may be lost for multivalued inverses. More…

- InverseFunction::ifun : Inverse functions are
    being used. Values may be lost for multivalued inverses. More…

- Solve::tdep : The equations appear to involve the variables
    to be solved for in an essentially non-algebraic way. More…
```

```
Solve[E^(1 - x + 1/y[x]) / (-1 + x) + C[1] + ExpIntegralEi[1 - x + 1/y[x]] == 0, y[x]]
```

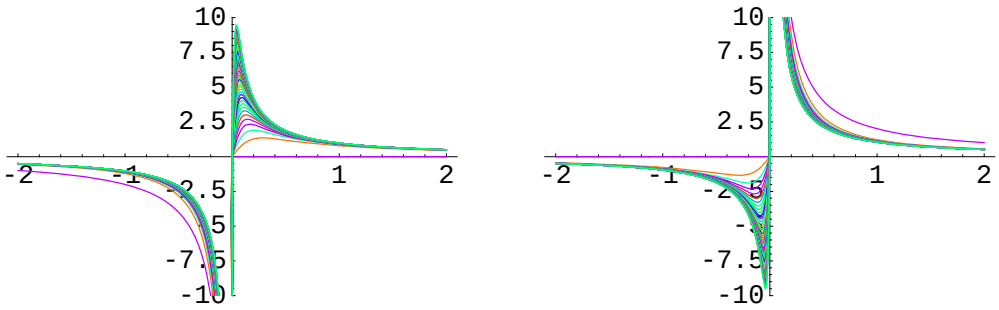
In questo caso abbiamo diversi warning: questi sono dovuti al fatto che, durante la ricerca della soluzione esatta di questo tipo di equazioni, vengono utilizzati inverse di ODE, per cui compaiono delle funzioni inverse, di cui *Mathematica* non conosce direttamente la struttura, e se in questo modo si perdono alcuni valori durante l'inversione. Nel caso seguente, invece, tutto quanto va a buon fine:

```
abel2 = y'[x] == y[x] / x - 3 y[x]^2 + x y[x]^3;
```

```
sol = DSolve[abel2, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{x} - \frac{1}{x^2 \sqrt{\frac{1}{x^2} + C[1]}}\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \frac{1}{x} + \frac{1}{x^2 \sqrt{\frac{1}{x^2} + C[1]}}\right]\right\}\right\}$$

```
a1 = Table[y[x] /. sol[[1]] /. C[1] → p, {p, 0, 1000, 20}];

a2 = Table[y[x] /. sol[[2]] /. C[1] → p, {p, 0, 1000, 20}];

DisplayTogetherArray[{
  Plot[Evaluate[a1, {x, -2, 2}, colori],
   PlotRange → {Automatic, {-10, 10}}],
  Plot[Evaluate[a2, {x, -2, 2}, colori],
   PlotRange → {Automatic, {-10, 10}}]
 }]
```




```
- GraphicsArray -
```

Come possiamo vedere, anche in questo caso abbiamo due rami distinti per la soluzione dell'equazione differenziale.

Finora abbiamo considerato solamente alcuni esempi di equazioni ODE di primo grado. Naturalmente, però, possiamo considerare anche ordini superiori: possiamo prendere un esempio di ODE a coefficienti costanti, che sono il caso sempre più semplice:

```
ode2 = y''[x] - y[x] == 0;

DSolve[ode2, y, x]
```

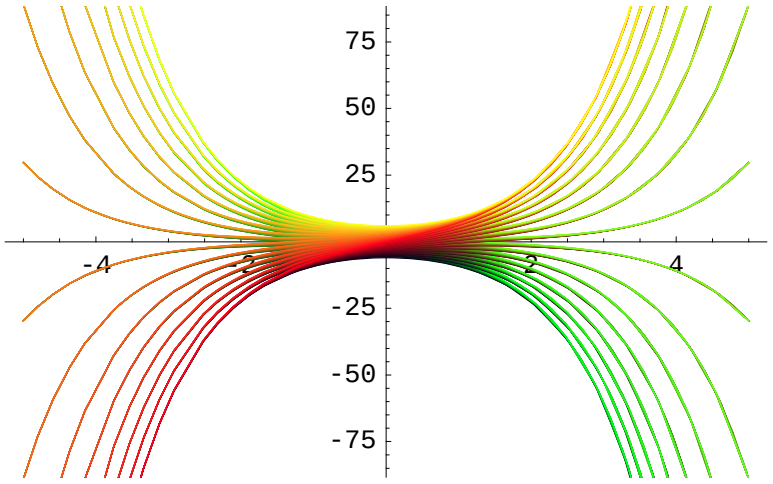
$$\{\{y \to \text{Function}[\{x\}, e^{x} C[1] + e^{-x} C[2]]\}\}$$

Naturalmente, in questo casi abbiamo due costanti di indeterminazione C[1] e C[2]. Vediamo come varia la soluzione al variare di quesi valori:

```
funz = Flatten[Table[{y[x] /. %[[1]] /. {C[1] → a, C[2] → b},
      RGBColor[(a + 3)/6, (b + 3)/6, .1]}, {a, -3, 3, .4}, {b, -3, 3, .4}], 1];
```

Ho creato in questa maniera una lista di elementi, in cui il primo rappresenta la funzione, mentre il secondo il colore della funzione, che varia al variare dei parametri:

```
Plot[Evaluate[funz[[All, 1]]], {x, -5, 5}, PlotStyle → funz[[All, 2]]]
```




```
- Graphics -
```

Per questo tipo di equazioni, la soluzione è data come combinazione lineare di esponenziali. Dato che abbiamo in questo esempio il secondo ordine, il risultato è combinazione lineare di due esponenziali. Per questo tipo di equazioni, come ben saprete, si può ottenere il risultato, a partire dall'equazione caratteristica, dalla quale si ricavano poi gli esponenti degli esponenziali che formano la base per lo spazio delle soluzioni. Prendiamo quest'altra equazione:

```
car = y'''[x] + 4 y''[x] - 11 y'[x] - 30 y[x] == 0;
```

A questo punto, possiamo ricavarci l'equazione caratteristica:

```
car /. {Derivative[n_][y][x] → λ^n, y[x] → 1}
```

$$-30 - 11\,\lambda + 4\,\lambda^2 + \lambda^3 == 0$$

Dato che tutto è rappresentato come espressioni, in *Mathematica*, lo è anche la forma $y''(x)$, che viene descritta da Derivative[*n*][*y*][*x*], dove *n* rappresenta il grado di derivazione, *y* rappresenta la funzione ed *x* la variabile indipendente. Abbiamo utilizzato quindi Derivative nella regola di sostituzione, per andare a sostituire tutte le derivate con la variabile λ, elevata al corrispettivo esponente. Questa regola permette di ricavarci direttamente l'equazione caratteristica, senza bisogno di doverla ricopiare. Una volta ottenuta, possiamo risolverla:

```
Solve[%, λ]
```

$$\{\{\lambda \to -5\}, \{\lambda \to -2\}, \{\lambda \to 3\}\}$$

Ed, arrivati a questo punto, possiamo creare la funzione che è soluzione dell'equazione differenziale, andando a mapparci le soluzioni in una combinazione lineare. Il comando seguente prende le soluzioni che abbiamo trovato, e le trasforma da regole a valori. A questo punto, applichiamo MapIndexed alla lista delle soluzioni, in maniera tale da creare una lista dove ogni elemento è formato dall'esponenziale dove compare la soluzione, e dal coefficiente di ogni elemento, con opportuno indice. A questo punto, applico Plus alla lista trovata, in modo da sommare tutti gli elementi presenti:

```
Plus @@ (MapIndexed[a_#2[[1]] Exp[x #1] &, λ /. %])
```

$$e^{-5x} a_1 + e^{-2x} a_2 + e^{3x} a_3$$

Notate come i comandi che ho utilizzato siano generici, il che mi permette di utilizzarli senza nessuna modifica per ODE a coefficienti costanti di qualsiasi ordine. L'unica accortezza è che, per equazioni di livello elevato, potrebbe essere necessario sostituire Solve con NSolve. Naturalmente, potevamo fare tutto direttamente tramite DSolve:

```
DSolve[car, y, x]
```

$$\{\{y \to \text{Function}[\{x\}, e^{-5x} C[1] + e^{-2x} C[2] + e^{3x} C[3]]\}\}$$

D'altronde, spiegare la risoluzione classica credo che abbia giovato a molti di voi, impratichendovi un pochino con le manipolazioni. D'altronde, qua ormai si parla seriamente...

Possiamo eseguire lo stesso procedimento mediante un'altra equazione:

```
car2 = y''''[x] - 10 * y'''[x] + 54 * y''[x] - 130 * y'[x] + 125 * y[x] == 0;
```

```
car2 /. {Derivative[n_][y][x] → λ^n, y[x] → 1}
```

$$125 - 130\,\lambda + 54\,\lambda^2 - 10\,\lambda^3 + \lambda^4 == 0$$

```
Solve[%, λ]
```

$$\{\{\lambda \to 2 - i\}, \{\lambda \to 2 + i\}, \{\lambda \to 3 - 4\,i\}, \{\lambda \to 3 + 4\,i\}\}$$

```
Plus @@ (MapIndexed[a_#2[[1]] Exp[x #1] &, λ /. %])
```

$$e^{(2-i)x} a_1 + e^{(2+i)x} a_2 + e^{(3-4i)x} a_3 + e^{(3+4i)x} a_4$$

In questo caso abbiamo coefficienti immaginari, per cui possiamo trasformare l'espressione ottenuta in maniera da far comparire seni e tett... ehm, coseni:

Con il comando DSolve avremmo ottenuto direttamente:

```
DSolve[car2, y[x], x]
```

```
{{y[x] →
    ℯ^(2 x) C[2] Cos[x] + ℯ^(3 x) C[4] Cos[4 x] + ℯ^(2 x) C[1] Sin[x] + ℯ^(3 x) C[3] Sin[4 x]}}
```

Lascio a voi il compito di mostrare l'equivalenza di queste due espressioni, tenendo conto del fatto che non è così immediato come possa sembrare a prima vista... Dovrete lavorarci un po', cari miei discepoli.

Vediamo adesso di trattare l'equazione di Eulero, che ha la seguente forma: $x^2\ y''(x) + a\,x\,y'(x) + b\,y(x) = 0$. Possiamo vederne un esempio:

```
eulero = x^2 * y''[x] + 5 * x * y'[x] + 9 * y[x] == 0;
```

```
DSolve[eulero, y[x], x]
```

$$\left\{\left\{y[x] \to \frac{C[2]\ \text{Cos}[\sqrt{5}\ \text{Log}[x]]}{x^2} + \frac{C[1]\ \text{Sin}[\sqrt{5}\ \text{Log}[x]]}{x^2}\right\}\right\}$$

In questo caso non abbiamo nente di particolarmente complicato, come soluzione. Possiamo invece considerare una generalizzazione, data dall'equazione lineare di Legendre, che sono equazioni nella forma $(c\,x + d)^2\,y''(x) + a\,(c\,x + d)\,y'(x) + b\,y(x) = 0$:

```
legendre = (9 x + 1)^2 * y''[x] + 5 * (3 x - 1) * y'[x] + 6 * y[x] == 0;
```

```
DSolve[legendre, y[x], x]
```

$$\left\{\left\{y[x] \to (-5)^{\frac{1}{27}\left(-11-\sqrt{67}\right)}\ 2^{\frac{2}{27}\left(-11-\sqrt{67}\right)}\ 3^{\frac{1}{9}\left(11+\sqrt{67}\right)}\ \left(\frac{1}{1+18\,x+81\,x^2}\right)^{\frac{1}{54}\left(-11-\sqrt{67}\right)} C[1]\right.$$
$$\text{Hypergeometric1F1}\left[-\frac{11}{27}-\frac{\sqrt{67}}{27},\ 1-\frac{2\sqrt{67}}{27},\ \frac{20}{27}\sqrt{\frac{1}{1+18\,x+81\,x^2}}\right] +$$
$$(-5)^{\frac{1}{27}\left(-11+\sqrt{67}\right)}\ 2^{\frac{2}{27}\left(-11+\sqrt{67}\right)}\ 3^{\frac{1}{9}\left(11-\sqrt{67}\right)}\ \left(\frac{1}{1+18\,x+81\,x^2}\right)^{\frac{1}{54}\left(-11+\sqrt{67}\right)} C[2]$$
$$\left.\text{Hypergeometric1F1}\left[-\frac{11}{27}+\frac{\sqrt{67}}{27},\ 1+\frac{2\sqrt{67}}{27},\ \frac{20}{27}\sqrt{\frac{1}{1+18\,x+81\,x^2}}\right]\right\}\right\}$$

Sebbene sia lunga, il ramo della soluzione è unico anche in questo caso.

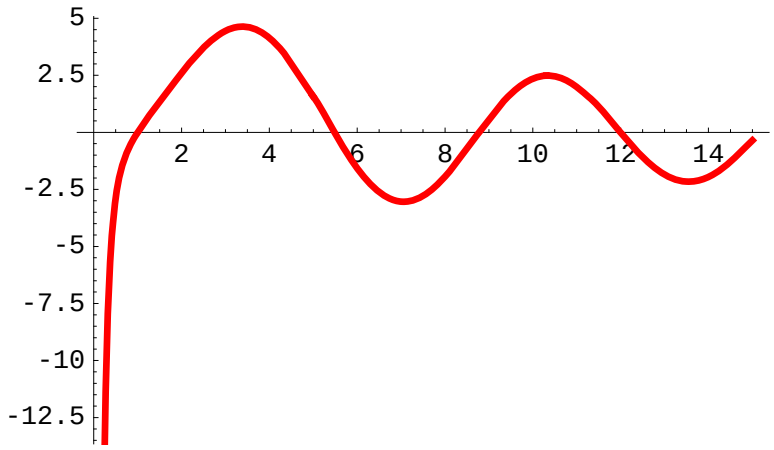
Come avete visto, *Mathematica* è in grado di risolvere equazioni differenziali che coinvolgono funzioni speciali, come quelle di Bessel:

```
bes = x^2 * y''[x] + x * y'[x] + (x^2 - 5) * y[x] == 0;
```

```
DSolve[{bes, y[1] == 0, y'[1] == 3}, y[x], x] // FullSimplify
```

$$\left\{\left\{y[x] \to \frac{3}{2}\,\pi\,(-\text{BesselJ}[\sqrt{5}, x]\,\text{BesselY}[\sqrt{5}, 1] + \text{BesselJ}[\sqrt{5}, 1]\,\text{BesselY}[\sqrt{5}, x])\right\}\right\}$$

```
Plot[y[x] /. %, {x, 0, 15}, PlotStyle → {{Red, Thickness[0.01]}}]
```




```
- Graphics -
```

Una leggera modifica all'equazione di sopra comporta soluzioni con più funzioni speciali:

```
bes2 = x^2 * y''[x] + x y'[x] + (x - 5) ^2 * y[x] == 0;
```

```
DSolve[bes2, y[x], x] // FullSimplify
```

$$\left\{\left\{y[x] \to e^{-i x}\, x^{5 i}\left(C[1]\,\text{HypergeometricU}\left[\frac{1}{2}, 1 + 10\,i, 2\,i\,x\right] + C[2]\,\text{LaguerreL}\left[-\frac{1}{2}, 10\,i, 2\,i\,x\right]\right)\right\}\right\}$$

Possiamo utilizzare anche altri tipi di coefficienti, che non siano semplicemente polunomi:

```
DSolve[ y''[x] - (k^2 + 2 * Sech[x]^2) y[x] == 0, y[x], x]
```

$$\left\{\left\{y[x] \to C[1]\,\text{LegendreP}\left[\frac{1}{2}\,i\,(i + \sqrt{7}), k, \text{Tanh}[x]\right] + C[2]\,\text{LegendreQ}\left[\frac{1}{2}\,i\,(i + \sqrt{7}), k, \text{Tanh}[x]\right]\right\}\right\}$$

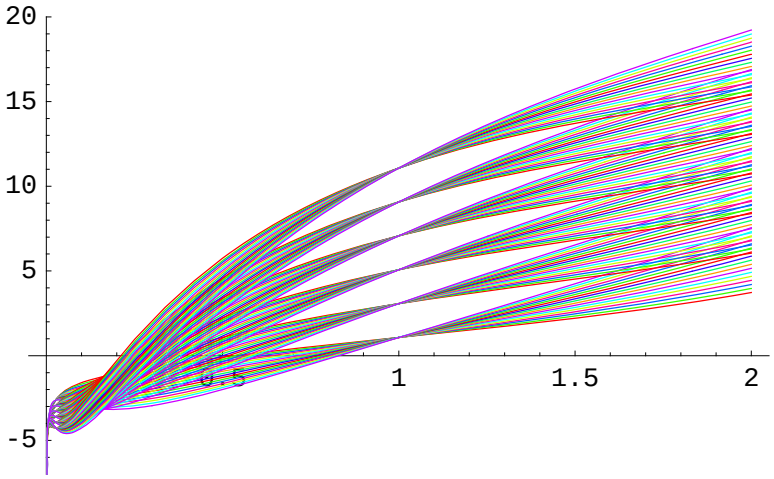
Le equazioni che DSolve è in grado di calcolare possono essere sia omogenee, come abbiamo visto negli esempi precedenti, che inomogenee, come nel caso che segue:

```
trig = DSolve[x^2 y''[x] + y[x] == x^4 + Log[x], y[x], x]
```

$$\left\{\left\{y[x] \to \sqrt{x}\, C[1] \cos\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right] + \sqrt{x}\, C[2] \sin\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right] + \frac{1}{13}\left(13\cos\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2 + x^4 \cos\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2 + 13\cos\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2 \text{Log}[x] + 13\sin\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2 + x^4 \sin\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2 + 13\,\text{Log}[x]\sin\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right)\right\}\right\}$$

Analogamente a quanto abbiamo fatto precedentemente, possiamo particolareggiare le soluzioni, disegnandone alcune con costanti definite mediante il comando Table:

```
sol = Flatten[Table[{y[x] /. trig /. {C[1] → i, C[2] → j}, {Hue[i + j]}},
    {i, 0, 10, 2}, {j, 1, 6, .3}], 1];

Plot[Evaluate[sol[[All, 1]]], {x, 0, 2},
 PlotStyle → sol[[All, 2]] , PlotRange → {Automatic, {-7, 20}}]
```




```
- Graphics -
```

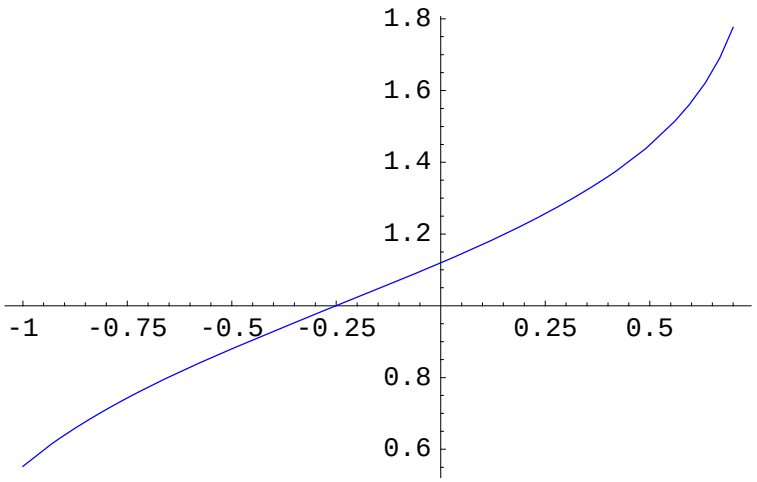
Mathematica è in grado di risolvere equazioni non lineari, anche se ci sono casi in cui non è possibile neanche al programma trovare la soluzione in forma chiusa in nessuna maniera conosciuta fino ad oggi. Un esempio può essere il seguente:

```
eqn = y''[x] == 4 x * y'[x]^2 + y'[x]^2;
```

```
DSolve[eqn, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, -\frac{2\,\text{ArcTan}\left[\frac{1+4x}{\sqrt{-1+8\,C[1]}}\right]}{\sqrt{-1+8\,C[1]}} + C[2]\right]\right\}\right\}$$

```
Plot[Evaluate[y[x] /. % /. {C[1] → -2, C[2] → 1}],
  {x, -1, .7}, PlotStyle → {Blue}];
```



Possiamo avere a che fare, in alcuni casi, con equazioni differenziali che non dipendono esplicitamente da x, $y'(x)$, come in questo caso:

```
DSolve[y''[x] == 2 Exp[3 * y[x]], y[x], x]
```

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

$$\left\{\left\{y[x] \to \text{Log}\left[-\frac{(-3)^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}}{2^{2/3}}\right]\right\},\right.$$

$$\left\{y[x] \to \text{Log}\left[\frac{3^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}}{2^{2/3}}\right]\right\},$$

$$\left.\left\{y[x] \to \text{Log}\left[\frac{(-1)^{2/3}\,3^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}}{2^{2/3}}\right]\right\}\right\}$$

Data la difficoltà intrinseca nel risolvere equazioi differenziali di ordine superiore, e considerando che DSolve utilzza Solve nella sua pancia, può capitare che le soluzioni coinvolgano la funzione Root, dato che è il modo di dare soluzioni per un'equazione di grado elevato, quando non è possibile trovarla in forma chiusa:

```
DSolve[x^5 y'''''[x] + 6 x^4 * y''''[x] -
   2 * x^3 * y'''[x] - x^2 * y''[x] + 5 * x * y'[x] + y[x] == 0, y[x], x]
```

$$\left\{\left\{y[x] \to x^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,1]}\,C[1] + x^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,2]}\,C[2] + x^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,3]}\,C[3] + x^{\frac{3}{2}-\frac{\sqrt{5}}{2}}\,C[4] + x^{\frac{3}{2}+\frac{\sqrt{5}}{2}}\,C[5]\right\}\right\}$$

In questo caso non possiamo trovarci una soluzione simbolica, ma comunque possiamo averla in maniera numerica. Inoltre, consideranto che abbiamo ottenuto una formula sì in maniera non chiusa, ma sempre senza coefficienti numerici, possiamo avere facilmente risultati numerici con precisione arbitraria:

```
f[x_] = y[x] /. %[[1]];
```

Notate come abbia utilizzato = invece di := perchè altrimenti avrei sempre la valutazione di % ad ogni nuovo comando, che non rappresenterebbe più la soluzione di DSolve. Se andiamo a mettere un valore alla soluzione in maniera esatta, otteniamo:

```
f[3]
```

$$3^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,1]}\,C[1] + 3^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,2]}\,C[2] + 3^{\text{Root}[1-7\#1-\#1^2+\#1^3\&,3]}\,C[3] + 3^{\frac{3}{2}-\frac{\sqrt{5}}{2}}\,C[4] + 3^{\frac{3}{2}+\frac{\sqrt{5}}{2}}\,C[5]$$

Il risultato è restituito sempre in forma simbolica, come nel caso generale. Andiamo invece a sostituire un valore numerico:

```
f[3.]
```

$$0.082313\,C[1] + 1.16682\,C[2] + 31.2355\,C[3] + 1.5214\,C[4] + 17.7468\,C[5]$$

Questa volta abbiamo visto come vengano restituiti i numeri, anche se rimangono i coefficienti C che ovviamente non erano calcolati. Possiamo utilizzare una precisione maggiore, se ci serve:

```
N[f[3], 50]
```

$$0.0823130366030684563575307019343195835589120066827 94\,C[1] + 1.1668222437039807891665184590870851804424075884110\,C[2] + 31.2354618267495162301684827452262695246174500199 69\,C[3] + 1.5214024193806499467234230586261890237812609816579\,C[4] + 17.7467839251836286480669457954206387581006598517 92\,C[5]$$

Ah, la potenza di *Mathematica*!!!!

In altri casi possiamo avere soluzioni esatte date in maniera simbolica, come questa sottostante che restituisce la soluzione con le funzioni di Airy che compaiono all'interno di integrali

```
ai = y'''[x] - y''[x] + 5 x y'[x] + 5 y[x] == 0;
```

```
DSolve[ai, y[x], x] // FullSimplify
```

$$\left\{\left\{y[x] \to e^{x/2}\left(\text{AiryAi}\left[\frac{(-1)^{1/3}(-1+20x)}{4\,5^{2/3}}\right]C[2] + \text{AiryBi}\left[\frac{(-1)^{1/3}(-1+20x)}{4\,5^{2/3}}\right]C[3] + \text{AiryBi}\left[\frac{(-1)^{1/3}(-1+20x)}{4\,5^{2/3}}\right]\int_1^x -\frac{1}{5^{1/3}}\left((-1)^{2/3}e^{-K\$4755/2}\pi\,\text{AiryAi}\left[\frac{(-1)^{1/3}(-1+20\,K\$4755)}{4\,5^{2/3}}\right]C[1]\right)dK\$4755 + \text{AiryAi}\left[\frac{(-1)^{1/3}(-1+20x)}{4\,5^{2/3}}\right]\int_1^x \frac{(-1)^{2/3}e^{-K\$4051/2}\pi\,\text{AiryBi}\left[\frac{(-1)^{1/3}(-1+20\,K\$4051)}{4\,5^{2/3}}\right]C[1]}{5^{1/3}}dK\$4051\right)\right\}\right\}$$

Sistemi ODE

Abbiamo visto finora come lavorare con equazioni differenziali, ma possiamo trattare alla stessa maniera i sistemi di equazioni differenziali, che sono sistemi del tipo:

$F(Y^{(n)}(x),\ Y^{(n-1)}(x),\ ...,\ Y'(x),\ Y(x)) == 0$

dove $Y(x)$ rappresenta un vettore di funzioni incognite: il caso più semplice di sistema, naturalmente, è il caso di sistema lineare, che può essere espresso, per il caso di sistema lineare del primo ordine, nella seguente forma:

$$Y'(x) = A(x)\,Y(x) + B(x)$$

Dove è presente il vettore delle derivate prime $Y'(x)$, quello delle funzioni non derivate $Y(x)$, ed inoltre è presente la matrice dei coefficienti $A(x)$, ed il vettore dei termini noti $B(x)$. Naturalmente il caso si semplifica ulteriormente se la matrice A è a termini costanti, invece che dipendenti dalla variabile x.

Il comando DSolve, tuttavia, si aspetta come argomento una lista di equazioni, quindi, se abbiamo la matrice A ed il vettore B, dobbiamo comunque trasformarli nelle rispettive equazioni, prima di dare il sistema in pancia a DSolve. Supponiamo che la nostra matrica A sia la seguente, assieme al vettore B:

```
A = {
    {-3, 3, 0},
    {5, 0, -2},
    {4, -3, -1}
   };
```

Adesso occorre creare il vettore delle incognite, che nel nostro caso sono delle funzioni:

```
Y[x_] := {y1[x], y2[x], y3[x]};
```

Arrivati a questo punto, possiamo crearci il sistema;

```
sisdiff = (#1 == #2) & ~ MapThread ~ {Y'[x], A.Y[x]}
```

$$\{y1'[x] == -3\,y1[x] + 3\,y2[x],\ y2'[x] == 5\,y1[x] - 2\,y3[x],\ y3'[x] == 4\,y1[x] - 3\,y2[x] - y3[x]\}$$

Una volta creato il sistema, possiamo risolverlo:

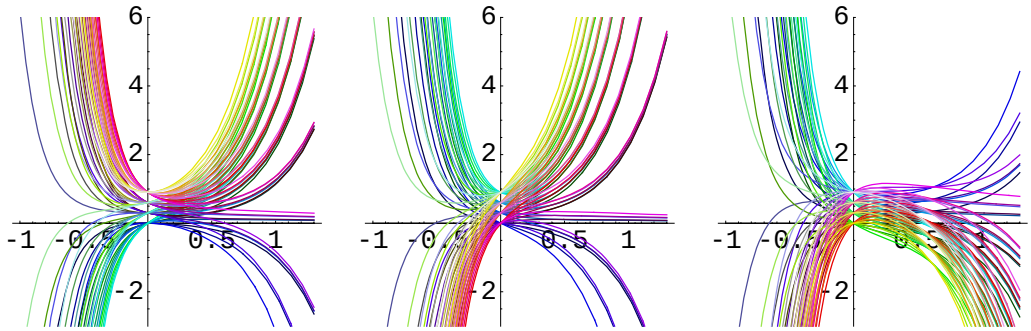
```
solsis = DSolve[sisdiff, Y[x], x] // FullSimplify
```

$$\Big\{\Big\{y1[x] \to \frac{1}{814}\,e^{-\frac{1}{2}(7+\sqrt{37})x}\Big(\Big(333 + 15\sqrt{37} + (333 - 15\sqrt{37})\,e^{\sqrt{37}\,x} + 148\,e^{\frac{1}{2}(13+\sqrt{37})x}\Big)\,C[1] - 2\,(74 + 7\sqrt{37})\,C[2] + 148\,e^{\frac{1}{2}(13+\sqrt{37})x}\,(2\,C[2] - C[3]) + (74 - 26\sqrt{37})\,C[3] + 2\,e^{\sqrt{37}\,x}\,((-74 + 7\sqrt{37})\,C[2] + (37 + 13\sqrt{37})\,C[3])\Big),$$
$$y2[x] \to \frac{1}{814}\,e^{-\frac{1}{2}(7+\sqrt{37})x}\Big(2\Big(-74 - 29\sqrt{37} + (-74 + 29\sqrt{37})\,e^{\sqrt{37}\,x} + 148\,e^{\frac{1}{2}(13+\sqrt{37})x}\Big)\,C[1] + 3\,(37 + 9\sqrt{37})\,C[2] + 296\,e^{\frac{1}{2}(13+\sqrt{37})x}\,(2\,C[2] - C[3]) + 4\,(37 - 2\sqrt{37})\,C[3] + e^{\sqrt{37}\,x}\,(3\,(37 - 9\sqrt{37})\,C[2] + 4\,(37 + 2\sqrt{37})\,C[3])\Big),$$
$$y3[x] \to \frac{1}{814}\,e^{-\frac{1}{2}(7+\sqrt{37})x}\Big(\Big(37 - 101\sqrt{37} + (37 + 101\sqrt{37})\,e^{\sqrt{37}\,x} - 74\,e^{\frac{1}{2}(13+\sqrt{37})x}\Big)\,C[1] + (74 + 40\sqrt{37})\,C[2] - 74\,e^{\frac{1}{2}(13+\sqrt{37})x}\,(2\,C[2] - C[3]) + (370 - 42\sqrt{37})\,C[3] + e^{\sqrt{37}\,x}\,((74 - 40\sqrt{37})\,C[2] + (370 + 42\sqrt{37})\,C[3])\Big)\Big\}\Big\}$$

Vediamo di disegnare queste soluzioni per alcuni valori delle costanti; visualizziamo in tre grafici distinti y1, y2, y3:

```
funz = Flatten[Table[# /. solsis /. {C[1] → a, C[2] → b, C[3] → c},
      {a, 0, 1, .3}, {b, 0, 1, .3}, {c, 0, 1, .3}]] & /@
   {y1[x], y2[x], y3[x]};

colori = PlotStyle → Flatten[Table[RGBColor[a, b, c],
     {a, 0, 1, .3}, {b, 0, 1, .3}, {c, 0, 1, .3}]];
```

```
DisplayTogetherArray[
 Plot[Evaluate[#, {x, -1, 1.3}, colori,
     AspectRatio → 1, PlotRange → {-3, 6}]] & /@ funz
]
```




```
- GraphicsArray -
```

Mamma mia, quante soluzioni...

Proviamo adesso a fare qualcosa di più semplice, invocando soltanto due equazioni, e tracciando la traiettoria che si ottiene utilizzando come coordinate parametriche le due funzioni ottenute:

```
A = {{7, -80}, {5, -5}};
```

```
Y[t_] := {x[t], y[t]}
```

```
sisT = (#1 == #2) & ~ MapThread ~ {Y'[t], A.Y[t]}
```

$$\{x'[t] == 7\,x[t] - 80\,y[t],\ y'[t] == 5\,x[t] - 5\,y[t]\}$$

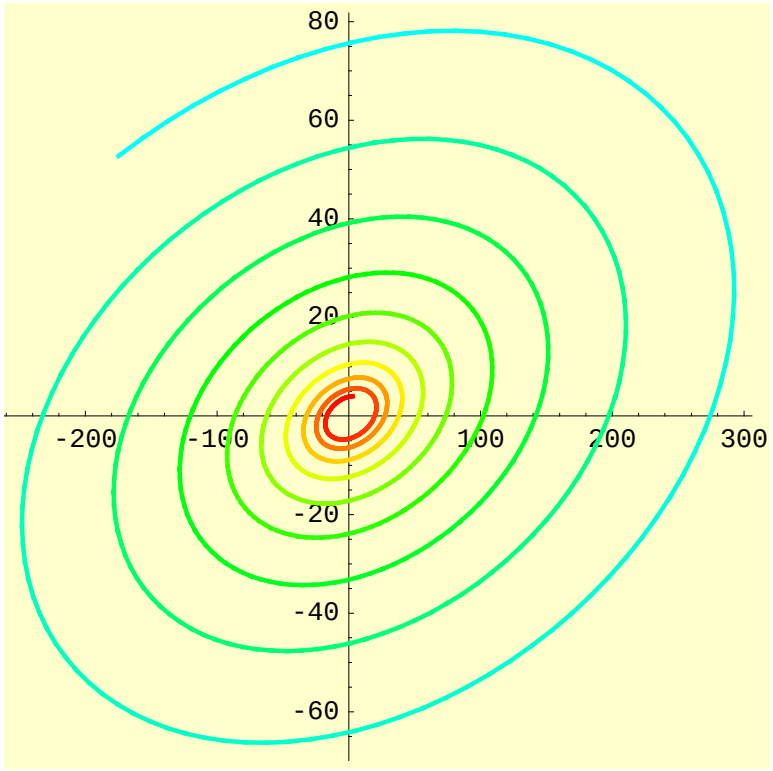
```
solT = DSolve[sisT, Y[t], t] // FullSimplify
```

$$\left\{\left\{x[t] \to e^{t}\,C[1]\,\text{Cos}[2\sqrt{91}\,t] + \frac{e^{t}\,(3\,C[1] - 40\,C[2])\,\text{Sin}[2\sqrt{91}\,t]}{\sqrt{91}},\right.\right.$$
$$\left.\left. y[t] \to e^{t}\,C[2]\,\text{Cos}[2\sqrt{91}\,t] + \frac{e^{t}\,(5\,C[1] - 6\,C[2])\,\text{Sin}[2\sqrt{91}\,t]}{2\sqrt{91}}\right\}\right\}$$

Arrivati a questo punto, possiamo utilizzare ParametricPlot per disegnare la traiettoria; tuttavia creerò qualcosa di più attraente, colorando la curva a con un gradiente che dipende dal parametro:

```
traiettoria[t_] =
  Evaluate[{x[t], y[t]} /. solT /. {C[1] → 2, C[2] → 4}][[1]];
```

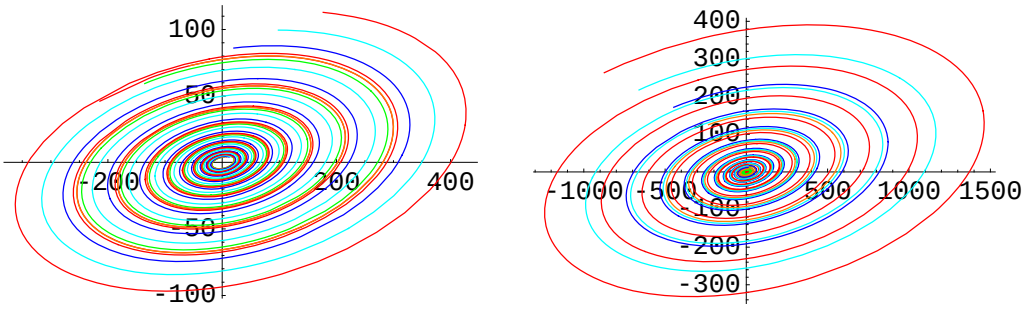
```
Module[

 (* Parametri del disegno *)

 {inizio = 0, fine = 3, passo = 1 / 300},

 (* Creazione della curva *)

 Show[Graphics[

   (* Creazione delle linee colorate che compongono il grafico *)

   {Table[{Thickness[0.006], Hue[t / 6],
      Line[{traiettoria[t - passo], traiettoria[t]}]},
     {t, inizio, fine, passo}]},

   (* Opzioni del disegno *)

   AspectRatio → 1, Axes → True, Background → RGBColor[1, 1, .8]]
 ]
]
```




```
- Graphics -
```

Possiamo anche vedere come varia la spirale al variare di un singolo parametro: vediamo i due grafici distinti, dove prima variamo C[1], e dopo C[2]:

```
traiettoriamultiplaC1 =
  Table[{x[t], y[t]} /. solT[[1]] /. {C[1] → a, C[2] → 4}, {a, 0, 20, 4}];

traiettoriamultiplaC2 = Table[
   {x[t], y[t]} /. solT[[1]] /. {C[1] → 10, C[2] → a}, {a, 0, 20, 4}];

DisplayTogetherArray[{
  ParametricPlot[Evaluate[#], {t, 0, 3},
     PlotRange → All, PlotStyle → {Red, Green, Orange, Blue, Cyan}] &
   /@ {traiettoriamultiplaC1, traiettoriamultiplaC2}
 }]
```




```
- GraphicsArray -
```

I coefficienti del sistema lineare, comunque, possono anche essere non costanti, cioè dipendere dalla variabile indipendente anch'essi, come in questo caso, in cui le due funzioni i vettori della matrice sono ortogonali fra di loro:

```
A = {{t, Cos[t]}, {- Cos[t], t}};

Y[t_] := {x[t], y[t]}

noncostante = MapThread[#1 == #2 &, {Y'[t], A.Y[t]}]
```

$$\{x'[t] == t\, x[t] + \text{Cos}[t]\, y[t],\ y'[t] == -\text{Cos}[t]\, x[t] + t\, y[t]\}$$

```
DSolve[noncostante, Y[t], t]
```

$$\left\{\left\{x[t] \to e^{\frac{t^2}{2}} C[1]\, \text{Cos}[\text{Sin}[t]] + e^{\frac{t^2}{2}} C[2]\, \text{Sin}[\text{Sin}[t]],\right.\right.$$
$$\left.\left. y[t] \to e^{\frac{t^2}{2}} C[2]\, \text{Cos}[\text{Sin}[t]] - e^{\frac{t^2}{2}} C[1]\, \text{Sin}[\text{Sin}[t]]\right\}\right\}$$

Anche nel caso dei sistemi di equazioni differenziali possiamo avere casi che non sono omogenei, come abbiamo trattato finora, cioè casi in cui il vettore B non è nullo:

```
A = {{7, -8}, {1, -1}};
B = {t, Gamma[t / 5]};
X[t_] = {x[t], y[t]};
inomogeneo = MapThread[#1 == #2 &, {X'[t], A.X[t] + B}]
```

$$\left\{x'[t] == t + 7\,x[t] - 8\,y[t],\ y'[t] == \text{Gamma}\left[\frac{t}{5}\right] + x[t] - y[t]\right\}$$

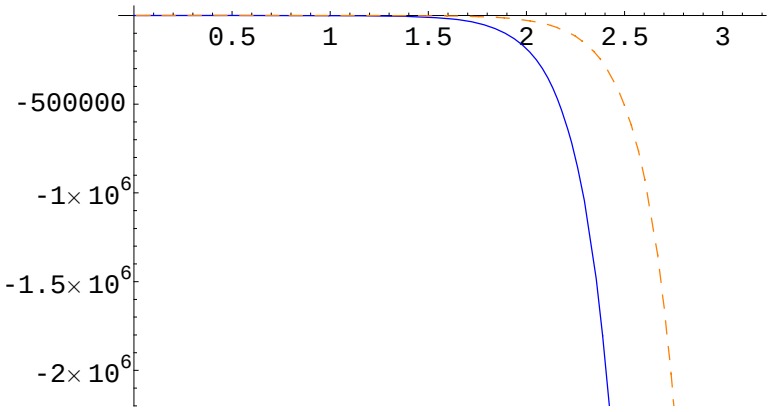
```
DSolve[inomogeneo, {x, y}, t] // FullSimplify
```

$$\left\{\left\{x \to \text{Function}\left[\{t\},\right.\right.\right.$$
$$\frac{1}{2}\left(e^{\left(3-2\sqrt{2}\right)t} - \sqrt{2}\,e^{\left(3-2\sqrt{2}\right)t} + e^{\left(3+2\sqrt{2}\right)t} + \sqrt{2}\,e^{\left(3+2\sqrt{2}\right)t}\right)C[1] +$$
$$\sqrt{2}\left(e^{\left(3-2\sqrt{2}\right)t} - e^{\left(3+2\sqrt{2}\right)t}\right)C[2] +$$
$$\frac{1}{2}\left(e^{\left(3-2\sqrt{2}\right)t} - \sqrt{2}\,e^{\left(3-2\sqrt{2}\right)t} + e^{\left(3+2\sqrt{2}\right)t} + \sqrt{2}\,e^{\left(3+2\sqrt{2}\right)t}\right)$$
$$\int_1^t\left(\frac{1}{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} - \sqrt{2}\,e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} + e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}} + \sqrt{2}\,e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}}\right)\text{K\$1025371} + \sqrt{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} - e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}}\right)\text{Gamma}\left[\frac{\text{K\$1025371}}{5}\right]\right)$$
$$\text{d}\text{K\$1025371} + \sqrt{2}\left(e^{\left(3-2\sqrt{2}\right)t} - e^{\left(3+2\sqrt{2}\right)t}\right)$$
$$\int_1^t\left(-\frac{\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} - e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}}\right)\text{K\$1029249}}{4\sqrt{2}} + \frac{1}{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} + \sqrt{2}\,e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} + e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}} - \sqrt{2}\,e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}}\right)\text{Gamma}\left[\frac{\text{K\$1029249}}{5}\right]\right)\text{d}\text{K\$1029249}\Big],$$

$$y \to \text{Function}\left[\{t\},\ -\frac{\left(e^{\left(3-2\sqrt{2}\right)t} - e^{\left(3+2\sqrt{2}\right)t}\right)C[1]}{4\sqrt{2}} +$$
$$\frac{1}{2}\left(e^{\left(3-2\sqrt{2}\right)t} + \sqrt{2}\,e^{\left(3-2\sqrt{2}\right)t} + e^{\left(3+2\sqrt{2}\right)t} - \sqrt{2}\,e^{\left(3+2\sqrt{2}\right)t}\right)C[2] -$$
$$\frac{1}{4\sqrt{2}}\left(\left(e^{\left(3-2\sqrt{2}\right)t} - e^{\left(3+2\sqrt{2}\right)t}\right)\right.$$
$$\int_1^t\left(\frac{1}{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} - \sqrt{2}\,e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} + e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}} + \sqrt{2}\,e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}}\right)\text{K\$1025371} + \sqrt{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1025371}} - e^{-\left(3+2\sqrt{2}\right)\text{K\$1025371}}\right)\text{Gamma}\left[\frac{\text{K\$1025371}}{5}\right]\right)\text{d}\text{K\$1025371}\Big) +$$
$$\frac{1}{2}\left(e^{\left(3-2\sqrt{2}\right)t} + \sqrt{2}\,e^{\left(3-2\sqrt{2}\right)t} + e^{\left(3+2\sqrt{2}\right)t} - \sqrt{2}\,e^{\left(3+2\sqrt{2}\right)t}\right)$$
$$\int_1^t\left(-\frac{\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} - e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}}\right)\text{K\$1029249}}{4\sqrt{2}} + \frac{1}{2}\left(e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} + \sqrt{2}\,e^{-\left(3-2\sqrt{2}\right)\text{K\$1029249}} + e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}} - \sqrt{2}\,e^{-\left(3+2\sqrt{2}\right)\text{K\$1029249}}\right)\text{Gamma}\left[\frac{\text{K\$1029249}}{5}\right]\right)\text{d}\text{K\$1029249}\Big]\Big\}\Big\}$$

Anche in questo caso possiamo visualizzare soluzioni partcolari nel solito modo. Dato che però la soluzione non è data in forma chiusa, il calcolo del grafico è computazionalmente intensivo, e potrebbe impiegarci parecchio prima di completare il disegno. Oltre al grafico è mostrato il tempo impiegato sul mio AthlonXP64 3200, con sistema operativo WinXP Pro a 32 bit:

```
particularsol = {x[t], y[t]} /. %[[1]] /. {C[1] → 1, C[2] → 2};

Plot[Evaluate[particularsol], {t, 0, Pi},
  PlotStyle → {{Blue}, {Dashing[{0.02}], Orange}}] // Timing
```




```
{599.25 Second, - Graphics -}
```

Cavolo!!! Dopo aver aspettato tutto questo tempo, poteva almeno uscire qualcosa di più carino... Ma non ho intenzione di rifare i calcoli un'altra volta. Accontentatevi di questo. Adesso, comunque, dopo questo 'grande' finale, possiamo passare alla sezione successiva.

PDE

La caratteristica che differenzia le PDE dalle ODE è il fatto che in questo caso abbiamo più variabili indipendenti, invece di una soltanto. Questo complica parecchio la risoluzione, specialmente nei casi non lineari dove, a parte pochi casi, possiamo tranquillamente metterci le mani in testa e passare alla sezione successiva dove si parla della risoluzione numerica. Ma per ora tralasciamola...

Vediamo subito un esempio, come il seguente:

```
pde = D[u[x, y], x] + y D[u[x, y], y] == y Sin[x]
```

```
y u^(0,1)[x, y] + u^(1,0)[x, y] == y Sin[x]
```

```
pdesol = DSolve[pde, u, {x, y}]
```

$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, \frac{1}{2}\left(-y \cos[x] + y \sin[x] + 2 C[1]\left[e^{-x} y\right]\right)\right]\right\}\right\}$$

Vediamo come sia stato necessario definire mediante l'operatore D la derivata perchè, come avevamo detto all'inizio, in questo caso dobbiamo specificare rispetto a quale variabile è calcolata la derivata, specialmente nelle derivate miste. Anche in questo caso, come nel caso unidimensionale, *Mathematica* lo interpreta tramite il comando Derivative:

```
InputForm[pde]
```

```
y*Derivative[0, 1][u][x, y] + Derivative[1, 0][u][x, y] ==
 y*Sin[x]
```

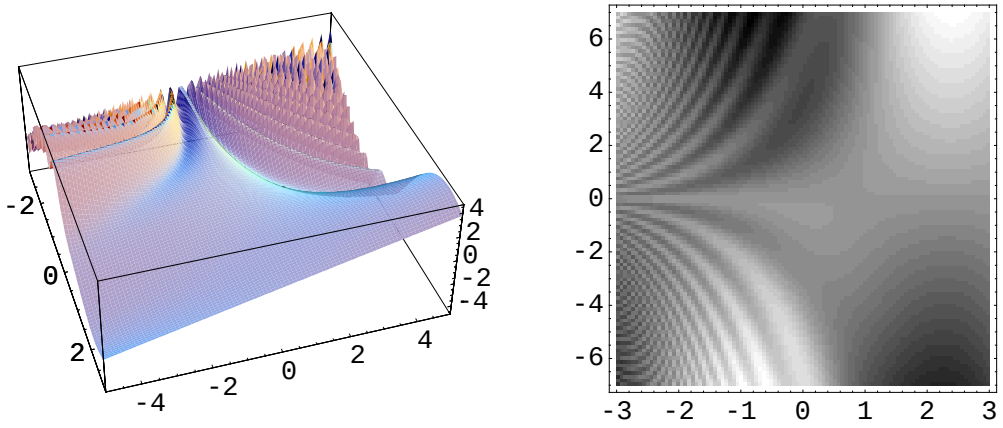
Ricordate? L'abbiamo visto a proposito della risoluzione manuale delle ODE a coefficienti costanti, quando siamo andati a trovarci l'equazione caratteristica...

Vi ricordo anche che, in questo caso, abbiamo le C[n] che adesso diventano delle funzioni arbitrarie, anche se l'argomento è specificato. Nel nostro caso abbiamo $2\,C[1][e^{-x}\,y]$, dove l'argomento è unico, ma è dato come funzione delle due variabili indimententi x, y. Se vogliamo trovarci una soluzione particolare, dobbiamo sostituire a $C[n]$ al posto di un valore, l'head di una funzione:

```
pdeparticolare[x_, y_] := u[x, y] /. pdesol[[1]] /. C[1] → Cos
```

In questo modo abbiamo creato la funzione. Notate come il coseno abbia lo stesso argomento di $C[1]$, cosa che fra l'altro era scontata. Adesso, tanto per farvi contenti, plottiamo questqa funzione:

```
DisplayTogetherArray[{
  Plot3D[pdeparticolare[x, y], {x, -3, 3}, {y, -5, 5},
   PlotPoints → 100, Mesh → False,
   ViewPoint -> {3.252, -1.028, 2.732}],
  DensityPlot[pdeparticolare[x, y], {x, -3, 3}, {y, -7, 7},
   PlotPoints → 100, Mesh → False]
 }]
```




```
- GraphicsArray -
```

Le equazioni PDE sono di solito classificate in tre tipi: lineati, quasilineari e non lineari. Per cominciare, vediamo un esempio di equazione omogenea a coefficienti costanti;

```
omo = 2 * D[u[x, y], x] + 9 * D[u[x, y], y] + u[x, y] == 0
```

$$u[x, y] + 9\, u^{(0,1)}[x, y] + 2\, u^{(1,0)}[x, y] == 0$$

```
sol = DSolve[omo, u, {x, y}]
```

$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, e^{-x/2}\, C[1]\left[\frac{1}{2}\,(-9\,x + 2\,y)\right]\right]\right\}\right\}$$

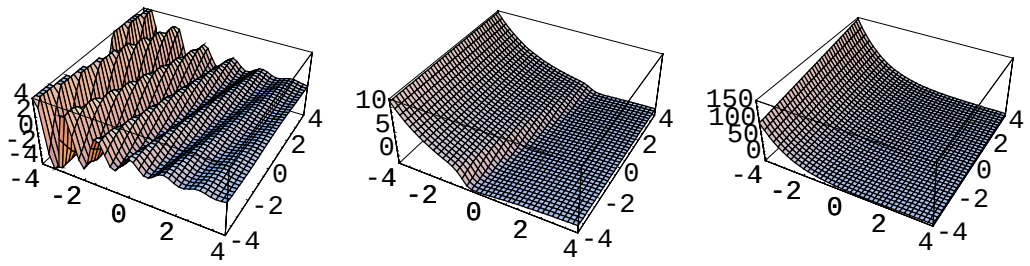
Non è veramente una delle equazioni più complicate che abbiamo visto finora... Vediamo delle soluzioni che si ottengono andando a sostituire alla funzione arbitraria *C*[1] delle funzioni specifiche:

```
omop = u[x, y] /. sol[[1]] /.
  {{C[1] → Sin}, {C[1] → ArcTan}, {C[1] → (# &)}}
```

$$\left\{e^{-x/2}\,\text{Sin}\left[\frac{1}{2}\,(-9\,x + 2\,y)\right],\ e^{-x/2}\,\text{ArcTan}\left[\frac{1}{2}\,(-9\,x + 2\,y)\right],\ \frac{1}{2}\,e^{-x/2}\,(-9\,x + 2\,y)\right\}$$

Andiamo a plottare adesso queste funzioni:

```
DisplayTogetherArray[
 Plot3D[#, {x, -4, 4}, {y, -4, 4}, PlotPoints → 40] & /@ omop
]
```



```
- GraphicsArray -
```

Come potete vedere la funzione arbitrarià da una maggiore libertà alla funzione, rispetto a dei coefficienti indeterminati...

Un esempio interessante di PDE è l'equazione di trasporto:

$\frac{\partial u}{\partial x} + \kappa \frac{\partial u}{\partial y} = 0$

dove κ rappresenta una quantità costante. Come potete vedere è un'equazione lineare, ed è una delle più semplici PDE che si possano incontrare (forse la più semplice?).

```
trasporto = DSolve[D[u[x, y], x] + κ D[u[x, y], y] == 0, u[x, y], {x, y}]
```

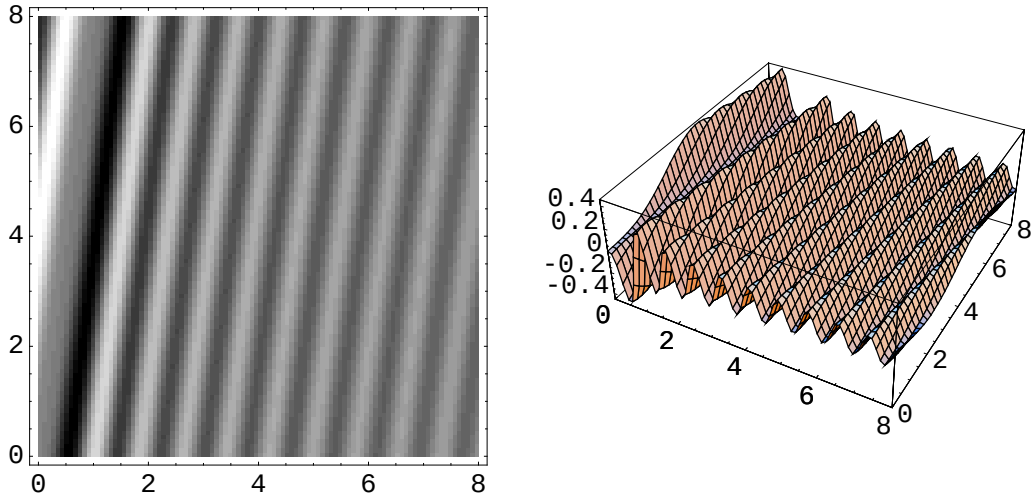
```
{{u[x, y] → C[1][y - x κ]}}
```

Come potete vedere è una soluzione abbastanza semplice, ed è praticamente data da qualsiasi funzione avente l'argomento specificato:

```
trabel = u[x, y] /. trasporto[[1]] /. C[1] → (BesselJ[3, #] &)
```

```
BesselJ[3, y - x κ]
```

```
DisplayTogetherArray[{
  DensityPlot[trabel /. κ → 8, {x, 0, 8},
   {y, 0, 8}, PlotPoints → 100, Mesh → False],
  Plot3D[trabel /. κ → 8, {x, 0, 8}, {y, 0, 8}, PlotPoints → {70, 40}]
 }]
```




```
- GraphicsArray -
```

Adesso andiamo a vedere un semplice esempio di equazione differenziale PDE inomogenea, con l'aggiunta di coefficienti non costanti, data dalla seguente equazione:

```
inom = D[u[x, y], x] + x D[u[x, y], y] == Sin[x] + x
```

$$x\, u^{(0,1)}[x, y] + u^{(1,0)}[x, y] == x + \text{Sin}[x]$$

```
solino = DSolve[inom, u, {x, y}]
```

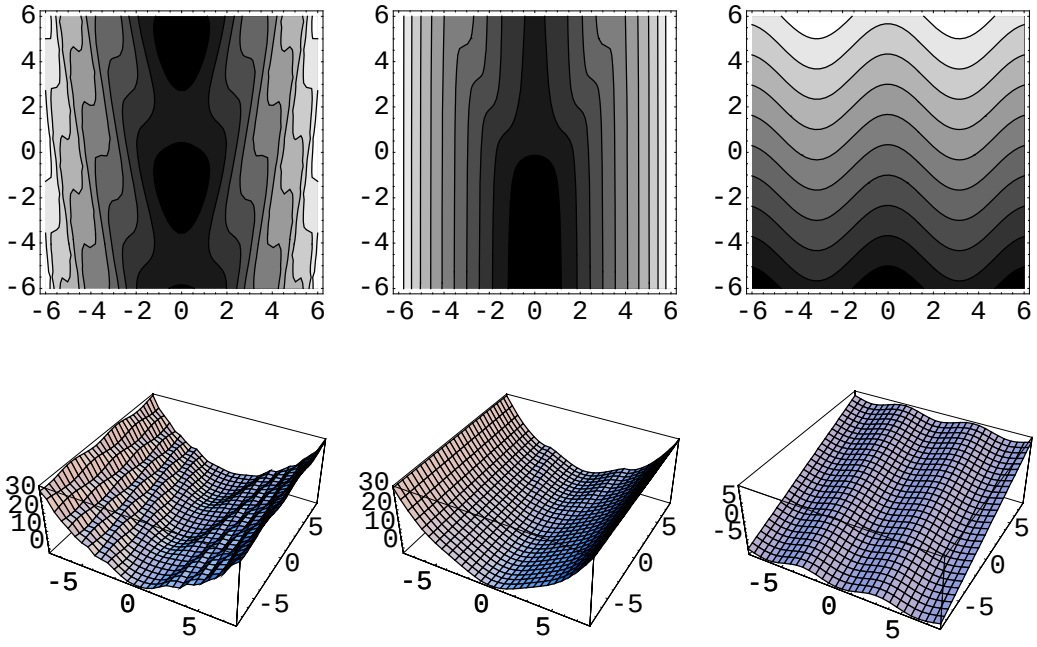
$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, \frac{1}{2}\left(x^2 - 2\,\text{Cos}[x] + 2\,C[1]\left[\frac{1}{2}\left(-x^2 + 2\,y\right)\right]\right)\right]\right\}\right\}$$

Anche in questo caso la soluzione è abbastanza semplice, seppure possa essere resa complicata a piacere scegliendo la funzione *C*[1]:

```
inopar = u[x, y] /. solino[[1]] /.
  {{C[1] → Sin}, {C[1] → ArcTan}, {C[1] → (# &)}}
```

$$\left\{\frac{1}{2}\left(x^2 - 2\,\text{Cos}[x] + 2\,\text{Sin}\left[\frac{1}{2}\left(-x^2 + 2\,y\right)\right]\right),\right.$$
$$\left.\frac{1}{2}\left(x^2 + 2\,\text{ArcTan}\left[\frac{1}{2}\left(-x^2 + 2\,y\right)\right] - 2\,\text{Cos}[x]\right),\ \frac{1}{2}\left(2\,y - 2\,\text{Cos}[x]\right)\right\}$$

```
DisplayTogetherArray[{
  ContourPlot[#, {x, -6, 6}, {y, -6, 6}, PlotPoints → 50] & /@ inopar,
  Plot3D[#, {x, -8, 8}, {y, -8, 8}, PlotPoints → 30] & /@ inopar
 }]
```



```
- GraphicsArray -
```

I gradi di liberta, nelle PDE, è generalmene maggiore, come possiamo vedere da questi esempi.

La caterogira successiva è data dalle equazioni quasi lineari, che ha termini lineari nelle derivate prime rispetto ad *x* ed ad *y*, ma che possono contenere termini non lineari dati da queste derivate e della funzione non derivata, come in questo esempio:

```
ql = D[u[x, y], x] + x^3 D[u[x, y], y] == u[x, y]^2 + x
```

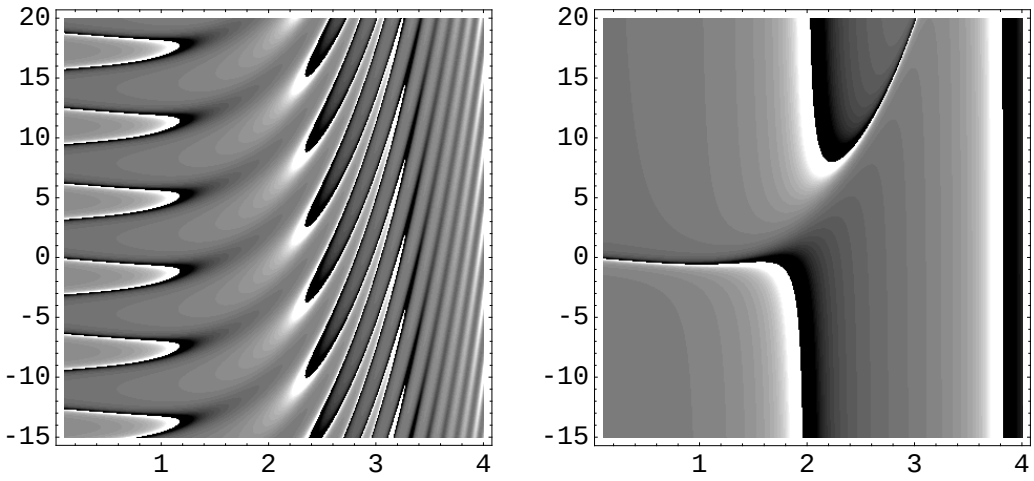
$$x^3\, u^{(0,1)}[x, y] + u^{(1,0)}[x, y] == x + u[x, y]^2$$

```
solql = DSolve[ql, u, {x, y}] // FullSimplify
```

$$\left\{\left\{u \to \text{Function}\left[\{x, y\},\right.\right.\right.$$
$$\left(-2\, x^{3/2}\, \text{BesselJ}\left[-\frac{2}{3}, \frac{2\, x^{3/2}}{3}\right] + 2\, x^{3/2}\, \text{BesselJ}\left[\frac{2}{3}, \frac{2\, x^{3/2}}{3}\right]\right.$$
$$\left.\text{C}[1]\left[\frac{1}{4}\,(-x^4 + 4\, y)\right]\right) \Big/ \left(2\left(x\, \text{BesselJ}\left[\frac{1}{3}, \frac{2\, x^{3/2}}{3}\right] +\right.\right.$$
$$\left.\left.\left.\left.x\, \text{BesselJ}\left[-\frac{1}{3}, \frac{2\, x^{3/2}}{3}\right]\text{C}[1]\left[\frac{1}{4}\,(-x^4 + 4\, y)\right]\right)\right)\right]\right\}\right\}$$

Possiamo vedere come anche in questo caso cambia radicalmente la funzione se andiamo a sostituire nel nostro caso due funzioni distinte, ad esempio il seno e l'identità:

```
DisplayTogetherArray[
  DensityPlot[#, {x, 0.1, 4},
     {y, -15, 20}, PlotPoints → 400, Mesh → False] & /@
   Evaluate[u[x, y] /. solql[[1]] /. {{C[1] → Sin}, {C[1] → (# &)}}]
 ] //
 Timing
```




```
{33.797 Second, - GraphicsArray -}
```

Quello che possiamo notare, come principale differenza dalle lineari alle quasilineari, è la presenza di discontinuità nella soluzione, che si possono vedere come passaggi bruschi dal bianco al nero, che sono caratteristiche di quest'ultimo tipo di equazioni, sebbene il metodo risolutivo sia per certi versi simile al caso lineare.

Le cose diventano decisamente più toste quando abbiamo a che fare con equazioni PDE non lineari, nel senso che la funzione

F(u,p,q)=0

Dove *p*, *q* rappresentano le derivate prime rispetto alle due variabili (sempre considerando equazioni differenziali del primo ordine ed a due variabili), ed *F* in questo caso non è lineare.

Possiamo definire per semplicità, le nostre derivate, e la funzione, in questa maniera:

```
z := u[x, y]; p := D[u[x, y], x]; q := D[u[x, y], y];
```

In questa maniera non dobbiamo per forza riscrivere tutto quanto quando abbiamo bisogno di usare le derivate nelle nostre equazioni. Effettivamente potevamo farlo dall'inizio, ma ci ho pensato solamente adesso... :-(Spero che voi l'abbiate fatto e che siate stati più intelligenti di me!!!

Comunque, vediamo subito un veloce esempio:

```
nonlin = p q == z^2
```

$$u^{(0,1)}[x, y]\, u^{(1,0)}[x, y] == u[x, y]^2$$

```
DSolve[nonlin, u, {x, y}]
```

```
— DSolve::nlpde : Solution requested to nonlinear partial
     differential equation. Trying to build a complete integral.
```

$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, e^{-\frac{y}{\sqrt{C[1]}} - x\sqrt{C[1]} - \frac{C[2]}{\sqrt{C[1]}}}\right]\right\}\right\}$$

In questo caso *Mathematica* ci avverte con gentilezza e garbo che abbiamo a che fare con PDE non lineari, prima di creare l'integrale completo, che serve per calcolare simbolicamente la PDE, quando obbiamente ci riesce. In questo caso, inoltre, abbiamo ottenuto dei valori di *C*[1], *C*[2] che non sono delle funzioni, ma delle costanti. Possiamo anche parametrizzare il tutto, in questo esempio, ponendo una costante in funzione dell'altra. In questo modo si ottiengono delle superfici dipendenti da un parametro, ovvero una famiglia di superfici, il cui inviluppo è pure soluzione della PDE, che è dato dall'integrale completo. Quest'ultimo include qualsiasi elemento della famiglia di curve ottenuta variando arbitrariamente i paramentri, l'inviluppo che abbiamo appenda detto, l'inviluppo dell'intera famiglia (integrale singolare), e altri integrali completi della PDE si possono dal processo di formazione degli inviluppi.

Come potete vedere, quindi, la teoria delle PDE è alquanto corposa e cummattusa (i siciliani mi capiranno). Tuttavia ci possono essere casi in cui possono essere comunque risolte. Un caso utile si ha quando possiamo applicare la separazione delle variabili, in modo da integrare in maniera indipendente le due parti dell'equazione ottenuta:

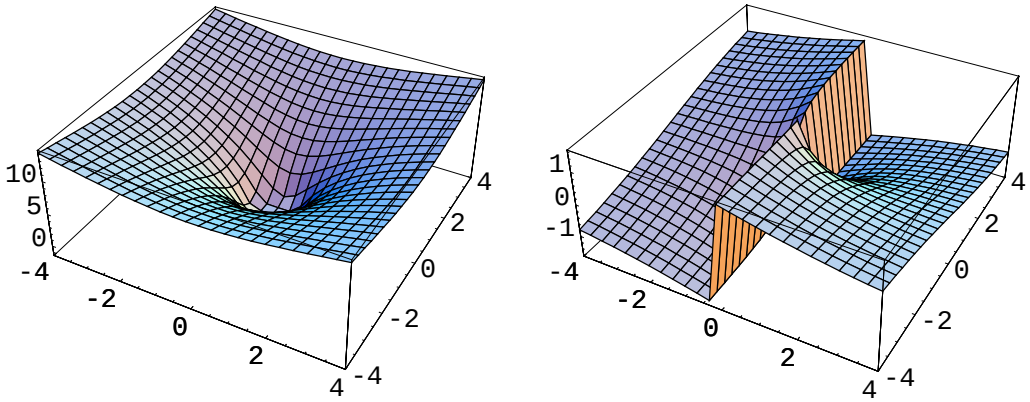
```
sepa = DSolve[(y p - x q)^2 + (x p + y q) == 6, z, {x, y}]
```

```
— DSolve::nlpde : Solution requested to nonlinear partial
     differential equation. Trying to build a complete integral.
```

$$\left\{\left\{u[x, y] \to -i\,\text{ArcTan}\left[\frac{y}{x}\right]\sqrt{C[1]} + C[2] + 6\,\text{Log}\left[\sqrt{x^2 + y^2}\right] + C[1]\,\text{Log}\left[\sqrt{x^2 + y^2}\right]\right\}\right\}$$

Dato che la funzione che abbiamo ottenuto è complessa, andiamo a visualizzare la parte reale e quella immaginaria, rispettivamente, fissando dei parametri per permetterci di avere una risoluzione numerica

```
DisplayTogetherArray[
 Plot3D[#, {x, -4, 4}, {y, -4, 4}] & /@ (({Re[#], Im[#]} &) /@
    Evaluate[u[x, y] /. sepa[[1]] /. {C[1] → 1, C[2] → 2}])
]
```




```
- GraphicsArray -
```

Notate la discontinuità tipica dell' ArcTan nel grafico della fase...

Il passo successivo, adesso, è quello delle PDE del secondo ordine. Come prima, partiamo dal caso più semplice, che è dato da quello lineare. In questo caso, essendo la funzione a più variabili (nel nostro caso due), dovranno comparire anche le derivate miste:

$$a \frac{\partial^2 u}{\partial x^2} + b \frac{\partial^2 u}{\partial x \partial y} + c \frac{\partial^2 u}{\partial y^2} + d \frac{\partial u}{\partial x} + e \frac{\partial u}{\partial y} + f u = g$$

Possiamo distunguere, qua come in tutti li altri casi, le equazioni omogenee da quelle inomogenee, a seconda se g è uguale o diverso da zero. Richiamando velocemente le proprietà delle PDE lineari del secondo ordine, possiamo dividerle in tre categorie principali, che dipendono dalle derivate di ordine due, in particolar modo dai loro coefficienti.

Se abbiamo $b^2 - 4\,a\,c < 0$, allora l'integrale è detto ellittico. Come caso particolare abbiamo l'equazione di Laplace, che si ottiene per $a = 1$, $b = 0$, $c = 1$. Poi abbiamo le equazioni parabolice, nel caso in cui $b^2 - 4\,a\,c = 0$, l'equazione è detta parabolica, come l'equazione per la diffuzione del calore. Infine, per $b^2 - 4\,a\,c > 0$ l'equazione viene chiamata iperbolica, come l'equazione d'onda.

Come potete vedere, nelle PDE del secondo ordine, anche solo fermandoci alle lineari, sorgono molti casi importanti. La soluzione simbolica può essere trovata in non tutti i casi, per esempio si possono trovare nei casi in cui l'equazione è omogenea, e i coefficienti delle derivate prime è nullo,

come appunto l'equazione di Laplace; in questo caso la soluzione è quella generica, dato che manca ogni condizione al contorno:

```
laplace = D[u[x, y], {x, 2}] + D[u[x, y], {y, 2}] == 0;

DSolve[laplace, u, {x, y}]
```

$$\{\{u \to \text{Function}[\{x, y\}, C[1][i\,x + y] + C[2][-i\,x + y]]\}\}$$

Possiamo vedere un altro esempio di PDE dove compaiono solamente le derivate di ordine 2:

```
pde2 = 8 D[u[x, y], {x, 2}] + 9 D[u[x, y], x, y] + 9 D[u[x, y], {y, 2}] == 0
```

$$9\,u^{(0,2)}[x, y] + 9\,u^{(1,1)}[x, y] + 8\,u^{(2,0)}[x, y] == 0$$

```
solpde2 = DSolve[pde2, u, {x, y}]
```

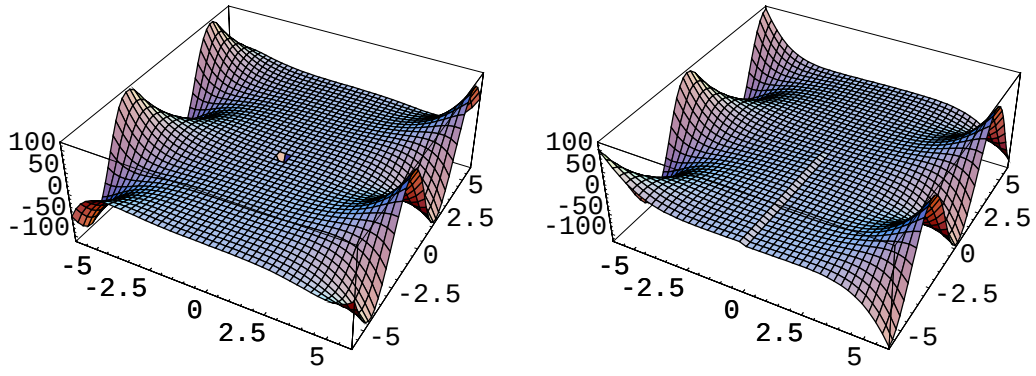
$$\left\{\left\{u \to \text{Function}\left[\{x, y\},\ C[1]\left[\frac{1}{16}\left(-9 + 3\,i\,\sqrt{23}\right)x + y\right] + C[2]\left[\frac{1}{16}\left(-9 - 3\,i\,\sqrt{23}\right)x + y\right]\right]\right\}\right\}$$

```
pde2 /. solpde2 // Simplify
```

$$\{\text{True}\}$$

Come possiamo vedere, le soluzioni sono rispettate. Anche in questo caso la funzione che abbiamo ottenuto ha valori sia reali che immaginari:

```
DisplayTogetherArray[
 Plot3D[#, {x, -6, 6}, {y, -6, 6}, PlotPoints → 45, PlotRange → All] & /@
  (({Re[#], Im[#]} &) /@
    Evaluate[u[x, y] /. solpde2[[1]] /. {C[1] → Sin, C[2] → Log}])
]
```




```
- GraphicsArray -
```

DAE

Molti sistemi naturali possono essere modellati tramite un insieme di equazioni che sono sia differenziali che algebriche: come non citare i circuiti elettronici, per chi fa ingegneria elettronica come me? Dalle leggi di Kirkoff, che sono algebriche, alle cariche e scariche dei condensatori ed induttori, che sono equazioni differenziali, ottenendo una bellissima ed affascinante DAE...

.....

Meglio se me ne vado a letto, per stasera.....

.....zzzzzzzzzzzzzzzzzzzzzzzzz.....

.....

Arieccoci, qua, per cominciare di nuovo. Di cosa parlavamo? delle letterine?!?!? Ah, ora ricordo... Bene, rientiramo nel nostro mondo fatto di matematica e di tante figurine colorate, che non sono quelle dei calciatori...

Abbiamo appena definito le DAE, facendo un piccolo esempio su un sistema che le sfruttasse. Come problema analitico sono abbastanza complicate, specialmente se andiamo a considerare i casi non lineari, che possono essere veramente delle bruttissime bestie.

Un caso in cui le DAE possono essere risolte è il caso dei sistemi lineari, scritti nella seguente maniera:

$A.x'(t) + B.x(t) = F$

Le matrici sono funzioni del parametro t, come pure il vettore F. Si distguono anche qua, tanto per cambiare, le equazioni omogenee da quelle inomogenee. Se A e B sono costanti, DSolve può essere in grado di trovare la soluzione. Inoltre, in questo tipo di problema bisogna porre particolare attenzione alle condizioni iniziali del problema, per assicurarci la soluzione.

Possiamo vedere qua un semplicissimo esempio di DAE:

```
dae = {
   x'[t] - 3 y[t] == 0,
   x[t] - y[t] == 0
  };
```

```
daesol = DSolve[dae, {x, y}, t]
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{4} e^{3t} C[1]\right], y \to \text{Function}\left[\{t\}, \frac{1}{4} e^{3t} C[1]\right]\right\}\right\}$$

Possiamo scrivere la corrispondente soluzione equazione in foma non omogenea, per compilcarci un tantinello la vita:

```
inodae = {
   x'[t] + y[t] == t,
   x[t] - y[t] == Cos[t]
  };

inodaesol = DSolve[inodae, {x, y}, t]
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{4}\left(e^{-t} C[1] + 2\left(-2 + 2t + \text{Cos}[t] + \text{Sin}[t]\right)\right)\right], y \to \right.\right.$$
$$\left.\left.\text{Function}\left[\{t\}, -\text{Cos}[t] + \frac{1}{4}\left(e^{-t} C[1] + 2\left(-2 + 2t + \text{Cos}[t] + \text{Sin}[t]\right)\right)\right]\right\}\right\}$$

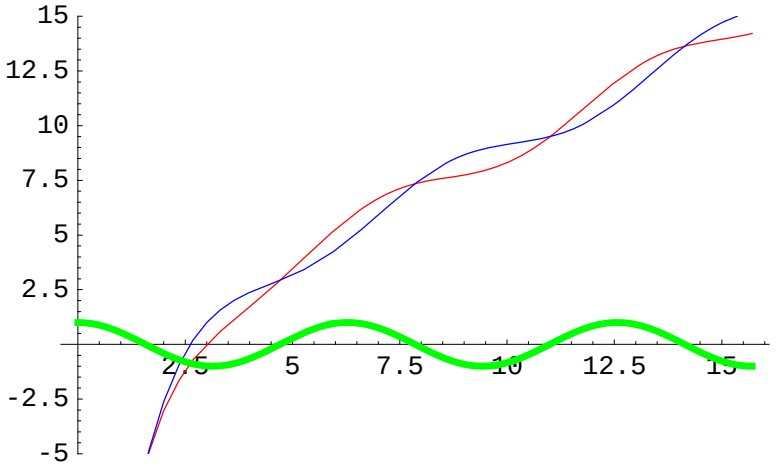
Come possiamo vedere il risultato è dato dalla soluzione omogenea, più una soluzione particolare di quella inomogenea, come dice la teoria delle equazioni differenziali riguardo, appunto, alle eq. inomogenee. Possiamo anche imporre delle opportune condizioni iniziali per le nostre funzioni:

```
inodaesolpar =
 DSolve[Flatten[{inodae, x[3] == 0}], {x, y}, t] // FullSimplify
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{2} e^{-t}\right.\right.\right.$$
$$\left.\left.\left.\left(-4e^3 - 2e^t + 2e^t t - e^3 \text{Cos}[3] + e^t \text{Cos}[t] - e^3 \text{Sin}[3] + e^t \text{Sin}[t]\right)\right],\right.\right.$$
$$y \to \text{Function}\left[\{t\}, -\frac{1}{2} e^{-t}\left(4e^3 + 2e^t - 2e^t t + e^3 \text{Cos}[3] + \right.\right.$$
$$\left.\left.\left.\left.e^t \text{Cos}[t] + e^3 \text{Sin}[3] - e^t \text{Sin}[t]\right)\right]\right\}\right\}$$

In questo modo possiamo plottare le funzioni ottenute:

```
Plot[Evaluate[{x[t], y[t], x[t] - y[t]} /. inodaesolpar],
 {t, 0, 5 Pi}, PlotRange → {-5, 15},
 PlotStyle -> {{Red}, {Blue}, {Thickness[0.01], Green}}]
```




```
- Graphics -
```

Possiamo vedere come la differenza delle due funzioni sia effettivamente il coseno che avevamo imposto nell'equazione algebrica della DAE.

Possiamo trattare le DAE anche in generale, volendo. Per esempio, potremmo volere che la funzione della parte inomogenea sia indefinita, come nel caso che segue:

```
indef = {
   x''[t] + y[t] == f[t],
    2 x[t] == h[t]
  };
```

```
DSolve[indef, {x, y}, t]
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{h[t]}{2}\right], y \to \text{Function}\left[\{t\}, f[t] - \frac{h''[t]}{2}\right]\right\}\right\}$$

In questo caso possiamo vedere che nella soluzione compaiono le funzioni arbitrarie. Possiamo notare anche un'altra cosa, però, cioè la mancanza di costanti arbitrarie, il che ci fa capire che nella soluzione del nostro problema non abbiamo gradi di libertà. Possiamo tentare di risolvere il caso in cui le funzioni diventano particolari:

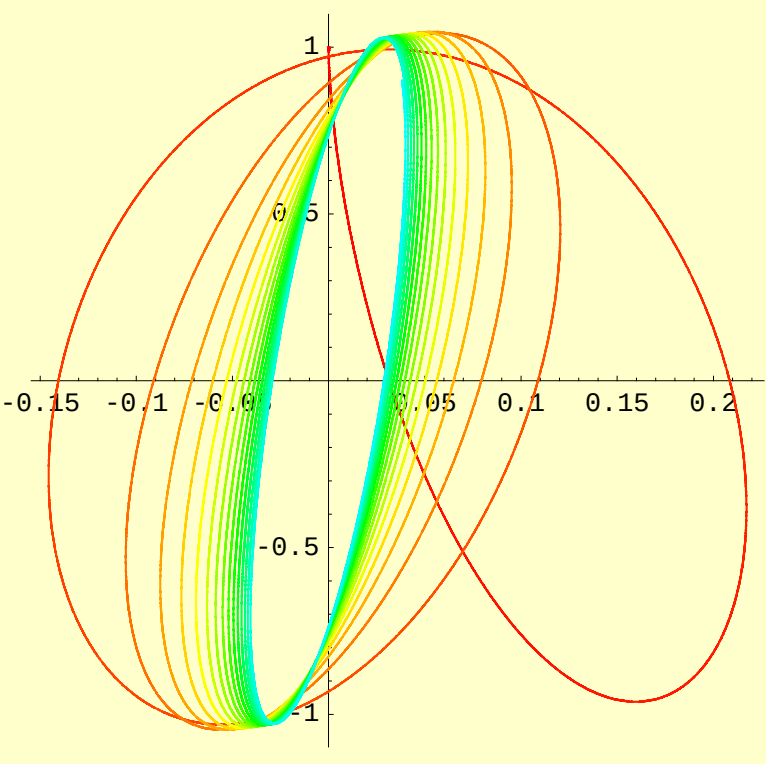
```
def = {
   x''[t] + y[t] == Cos[t],
    2 x[t] == BesselJ[3, t]
  };
```

```
sol = DSolve[def, {x, y}, t]
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{2}\,\text{BesselJ}[3, t]\right],\right.\right.$$
$$\left.\left. y \to \text{Function}\left[\{t\}, \frac{1}{4}\left(\frac{1}{2}\,(-\text{BesselJ}[1, t] + \text{BesselJ}[3, t]) + \frac{1}{2}\,(\text{BesselJ}[3, t] - \text{BesselJ}[5, t])\right) + \text{Cos}[t]\right]\right\}\right\}$$

Possiamo notare la corrispondenza fra le fue soluzioni che abbiamo ottenuto, nel caso generale ed in quello particolare. Possiamo plottare il grafico parametrico nello stesso modo di come abbiamo fatto prima

```
tr[t_] = Evaluate[{x[t], y[t]} /. sol][[1]];

Module[
 {inizio = 0, fine = 100, passo = 1 / 300, scalahue = 200},
 Show[Graphics[
   {Table[{Thickness[0.003],
      Hue[t / scalahue], Line[{tr[t - passo], tr[t]}]},
     {t, inizio + passo, fine, passo}]},
   AspectRatio → 1, Axes → True, Background → RGBColor[1, 1, .8]]
 ]
]
```




```
- Graphics -
```

Possiamo fare adesso un passetto avanti, ed andare a considerare, per esempio, un problema DAE del terzo ordine, imponendo pure le condizioni iniziali, date sulla *x*:

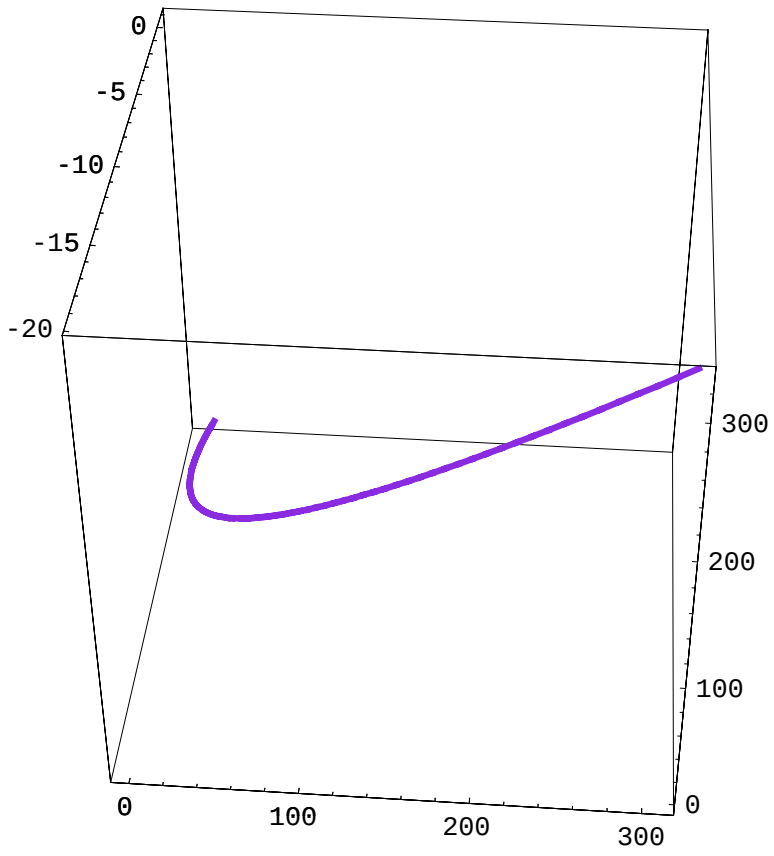
```
Clear[x, y, z]
```

```
terzo = {
   x'''[t] - y[t] + z[t] == 0,
   x'[t] + z[t] - t == 0,
   z[t] - y[t] == 0,
   x[0] == 1,
   x'[0] == 4,
   x''[0] == 2
  };
```

```
solterzo = DSolve[terzo, {x, y, z}, t]
```

```
{{x → Function[{t}, 1 + 4 t + t^2],
  y → Function[{t}, -4 - t], z → Function[{t}, -4 - t]}}
```

```
ParametricPlot3D[Evaluate[
  {x[t], y[t], x[t] - y[t], {BlueViolet, Thickness[0.01]}} /. solterzo],
 {t, -5, 5 Pi}, BoxRatios → {1, 1, 1},
 ViewPoint -> {0.279, -3.548, 2.851}]
```




```
- Graphics3D -
```

Il problema diventa invece di più difficile soluzione, diciamo impossibilile nella maggior parte dei casi, quando i coefficienti non sono costanti. In questo caso si rendono necessari metodi di risoluzione numerica.

Problema del valore al contorno

Fino ad adesso abbiamo cercato, in generale, soluzioni generiche per le nostre equazioni differenziali. Queste danno idea della struttura della soluzione, ma in pratica si è interessati ad una particolare soluzione, che soddisfi la condizione al contorno specifica per il nostro problema. Possiamo imporre due tipi di condizioni: le condizioni iniziali e le condizioni al contorno. Per il primo tipo imponiamo che ad un determinato valore iniziale del parametro t (consideriamo ad esempio l'istante iniziale dell'evoluzione di un sistema), siano date delle specifiche condizioni sul valore della funzione, e sulle due derivate. Invece, per le condizioni al contorno, specifichiamo il valore che deve assumere la funzione in particolari punti del dominio della funzione stessa, per esempio il valore che deve assumere all'inizio ed alla fine dell'intervallo di spazio che prendiamo in considerazione. Analiticamente possiamo trattare allo stesso modo entrambi i tipi di restrizioni della soluzione, che chiamiamo quindi valori al contorno.

La soluzione nei casi lineari è abbastanza semplice, mentre non lo è altrettanto nei casi non lineari, per i quali possono comparire più rami, per i quali soltanto alcuni sono in grado di soddisfare precise condizioni al contorno che andiamo a specificare nel nostro problema.

Prendendo come sempio un caso lineare, possiamo avere un'equazione del seguente tipo:

```
lineare = y'[t] + t^2 y[t] == t
```

$$t^2\, y[t] + y'[t] == t$$

La soluzione generale sarà data dal risultato dato da DSolve, senza specificare nessuna condizione al contorno:

```
generale = DSolve[lineare, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, e^{-\frac{t^3}{3}} C[1] - \frac{e^{-\frac{t^3}{3}}\, t^2\, \text{Gamma}\left[\frac{2}{3}, -\frac{t^3}{3}\right]}{3^{1/3}\, (-t^3)^{2/3}}\right]\right\}\right\}$$

Possiamo notare come, essendo del primo ordine, compaia soltanto un coefficiente di indeterminazione. Supponiamo, adesso, di volere una precisa condizione al contorno, per esempio specifichiamo il valore che deve assumere la funzione quando si ha $t = 0$, per esempio 3:

```
particolare = DSolve[{lineare, y[1] == 3}, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\},\ \frac{1}{6\,t}\left(e^{-\frac{t^3}{3}}\left(18\,e^{1/3}\,t - 3\,i\,3^{1/6}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] - 3^{2/3}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] + 2\,3^{2/3}\,(-t^3)^{1/3}\,\text{Gamma}\left[\frac{2}{3}, -\frac{t^3}{3}\right]\right)\right)\right]\right\}\right\}$$

```
y[1] /. particolare[[1]] // Simplify
```

```
3
```

Abbiamo trovato il caso particolare. Tuttavia, possiamo anche creare delle condizioni al contorno simboliche; possiamo ad esempio supporre che per $t = 0$ la la funzione assuma un generico valore α:

```
particolareα = DSolve[{lineare, y[1] == α}, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\},\ \frac{1}{6\,t}\left(e^{-\frac{t^3}{3}}\left(6\,e^{1/3}\,t\,\alpha - 3\,i\,3^{1/6}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] - 3^{2/3}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] + 2\,3^{2/3}\,(-t^3)^{1/3}\,\text{Gamma}\left[\frac{2}{3}, -\frac{t^3}{3}\right]\right)\right)\right]\right\}\right\}$$

Possiamo notare che, non essendo il coefficiente semplicemente additivo, non basta andare a sostituire la condizione iniziale al coefficiente per ottenere il risultato, ma abbiamo bisogno di ulteriori elaborazioni:

```
f[t_] = y[t] /. particolareα[[1]]
```

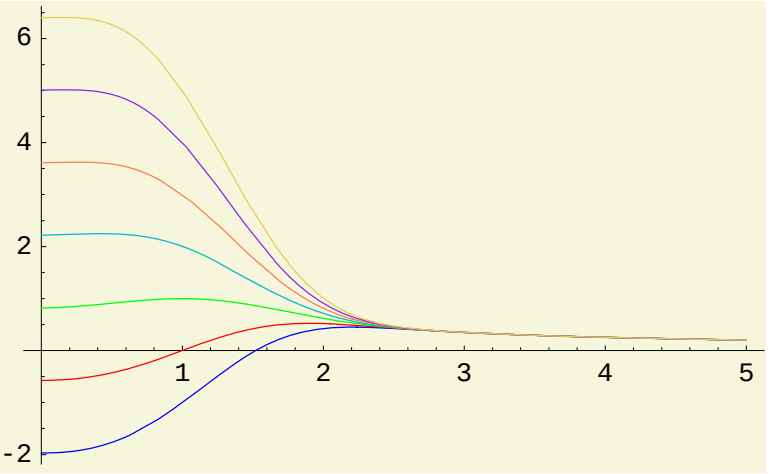
$$\frac{1}{6\,t}\left(e^{-\frac{t^3}{3}}\left(6\,e^{1/3}\,t\,\alpha - 3\,i\,3^{1/6}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] - 3^{2/3}\,t\,\text{Gamma}\left[\frac{2}{3}, -\frac{1}{3}\right] + 2\,3^{2/3}\,(-t^3)^{1/3}\,\text{Gamma}\left[\frac{2}{3}, -\frac{t^3}{3}\right]\right)\right)$$

```
f[1] // Simplify
```

$$\alpha$$

Come vedete, tutto persegue il grande cerchio della vita...

```
Plot[
 Evaluate[Table[f[t] /. α → αvalue, {αvalue, -1, 5}]]
 , {t, 0, 5},
 PlotStyle → {Blue, Red, Green, Cerulean, Coral, BlueViolet, Banana},
 Background → LightBeige]
```




```
- Graphics -
```

Possiamo vedere un altro esempio, questa volta con un'equazione del secondo oridine; in questa maniera compariranno due costanti di indeterminazione:

```
lineare2 = y''[t] + 3 y'[t] - y[t] == t
```

$$-y[t] + 3\,y'[t] + y''[t] == t$$

```
generale = DSolve[lineare2, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, -3 - t + e^{\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right) t}\, C[1] + e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right) t}\, C[2]\right]\right\}\right\}$$

Come possiamo vedere sono spuntati, in questo caso, sia *C*[1] che *C*[2]; di conseguenza abbiamo bisogno di due condizioni al contorno per poter risolvere questa equazione differenziale per un caso particolare. Possiamo dare le due condizioni nei due modi citati: possiamo specificare il valore che deve assumere la funzione in due punti distinti, oppure specificare il valore che deve assumere in un punto, ed il valore della derivata in un punto, che possono pure coincidere. Un esempio del primo caso può essere:

```
DSolve[{lineare2, y[0] == 0, y[2] == 3}, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{1}{-1+e^{2\sqrt{13}}}\left(3-3e^{2\sqrt{13}}-3e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}-8e^{3+\sqrt{13}+\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}+3e^{2\sqrt{13}+\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}+8e^{3+\sqrt{13}+\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}+t-e^{2\sqrt{13}}t\right)\right]\right\}\right\}$$

Mentre per il secondo caso possiamo avere, ad esempio:

```
DSolve[{lineare2, y[0] == 0, y'[0] == 2}, y, t]
```

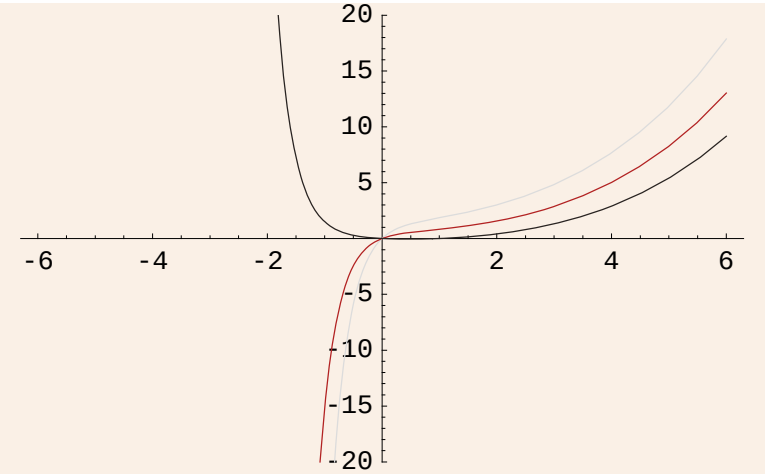
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{1}{26}\left(-78+39e^{\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}-15\sqrt{13}\,e^{\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}+39e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}+15\sqrt{13}\,e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}-26t\right)\right]\right\}\right\}$$

Il punto dove specifichiamo la derivata può essere anche diverso dal primo:

```
DSolve[{lineare2, y[0] == 0, y'[4] == 2}, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, \left(-9-3\sqrt{13}+9e^{4\sqrt{13}}-3\sqrt{13}\,e^{4\sqrt{13}}+9e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}+3\sqrt{13}\,e^{\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}-6e^{6+2\sqrt{13}+\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}-9e^{4\sqrt{13}+\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}+3\sqrt{13}\,e^{4\sqrt{13}+\left(-\frac{3}{2}-\frac{\sqrt{13}}{2}\right)t}+6e^{6+2\sqrt{13}+\left(-\frac{3}{2}+\frac{\sqrt{13}}{2}\right)t}-3t-\sqrt{13}\,t+3e^{4\sqrt{13}}t-\sqrt{13}\,e^{4\sqrt{13}}t\right)\Big/\left(3+\sqrt{13}-3e^{4\sqrt{13}}+\sqrt{13}\,e^{4\sqrt{13}}\right)\right]\right\}\right\}$$

```
Plot[Evaluate[y[t] /. {{%%%[[1]]}, {%%[[1]]}, {%[[1]]}}],
 {t, -6, 6},
 PlotStyle → {Gainsboro, Firebrick, IvoryBlack},
 Background → Linen, PlotRange → {-20, 20}
]
```




```
- Graphics -
```

Come possiamo vedere tutte e tre le soluzioni passano per l'origine, come specificato dalle nostre condizioni al contorno.

Possiamo risolvere un altro tipo di equazione differenziale:

```
problemacontorno = y''[x] + y[x] == E^x
```

$$y[x] + y''[x] == e^x$$

```
prgen = DSolve[problemacontorno, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, C[1]\,\text{Cos}[x] + C[2]\,\text{Sin}[x] + \frac{1}{2}\,e^x\,(\text{Cos}[x]^2 + \text{Sin}[x]^2)\right]\right\}\right\}$$

Proprio non riuscite a capire perchè l'ho chiamata così questa equazione, vero? Vedete un poco adesso quello che succede. In generale, posso porre delle condizioni al contorno per avere il risultato voluto, come in questo caso:

```
DSolve[{problemacontorno, y[0] == 1, y[1] == 1 / 2}, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{2}\,(\text{Cos}[x] + e^x\,\text{Cos}[x]^2 - \text{Cot}[1]\,\text{Sin}[x] + \text{Csc}[1]\,\text{Sin}[x] - e\,\text{Csc}[1]\,\text{Sin}[x] + e^x\,\text{Sin}[x]^2)\right]\right\}\right\}$$

Guardate con più attenzione il risultato, e noterete che in effetti non possiamo trovare tutte le soluzioni che vogliamo.

```
y[0] /. prgen
```

$$\left\{\frac{1}{2} + C[1]\right\}$$

Se imponiamo $y[0] == 1$, allora per forza di cose $C[1]$ deve essere uguale ad $1/2$. In questo caso nel resto possiamo giostrarci solamente con $C[2]$. Se però ci mettiamo a calcolare la funzione in π:

```
y[π] /. prgen
```

$$\left\{\frac{e^{\pi}}{2} - C[1]\right\}$$

Notiamo come in questo caso il valore dipenda anch'esso esclusivamente da $C[1]$. In questo caso, fissato $y[0]$, non possiamo fissare con altrettanta arbitrarietà il valore della funzione in π, perchè avrei delle condizioni inconsistenti. Abbiamo la soluzione se imponiamo i valori che abbiamo trovato:

```
DSolve[{problemacontorno, y[0] == 1, y[π] == E^π / 2 - 1 / 2}, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{2}\left(\text{Cos}[x] + e^{x}\,\text{Cos}[x]^{2} + 2\,C[2]\,\text{Sin}[x] + e^{x}\,\text{Sin}[x]^{2}\right)\right]\right\}\right\}$$

Se invece andiamo a inserire un valore diverso nel punto π otteniamo un errore, perchè non è obiettivamente possibile andare a trovare un valore di $C[1]$ che soddisfi entrambe le condizioni:

```
DSolve[{problemacontorno, y[0] == 1, y[π] == 1}, y, x]
```

```
— DSolve::bvnul : For some branches of the general solution,
    the given boundary conditions lead to an empty solution. More…

{}
```

In questo caso otteniamo un errore, e non viene restituita nessuna soluzione, per il semplice fatto che non esistono: nell'ultimo caso $C[2]$ dovrebbe essere pari ad $1/2$ nel punto $x = 0$, mentre dovrebbe essere pari a $1 - e^{\pi}/2$ nel punto $x = \pi$, cosa che ovviamente non è possibile in quanto $C[1]$ rappresenta una quantità costante.

Naturalmente, con le opportune condizioni al contorno, possiamo trovare soluzioni al contorno di equazioni non lineari, di grado più elevato e così via, a patto comunque, come abbiamo appena visto, che le condizioni siano consistenti:

```
quarto = y''''[x] + 2 y''[x] - 2 y[x] == Cos[x]
```

$$-2\,y[x] + 2\,y''[x] + y^{(4)}[x] == \text{Cos}[x]$$

```
DSolve[quarto, y, x]
```

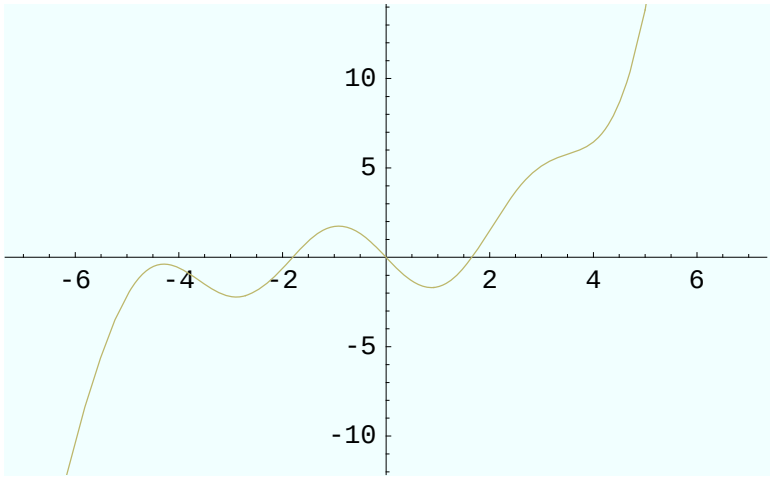
$$\left\{\left\{y \to \text{Function}\left[\{x\},\ e^{\sqrt{-1+\sqrt{3}}\,x}\,C[3] + e^{-\sqrt{-1+\sqrt{3}}\,x}\,C[4] + C[1]\,\text{Cos}\left[\sqrt{1+\sqrt{3}}\,x\right] + C[2]\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right] + \frac{1}{12}\left(-\sqrt{2\,(-1+\sqrt{3})\,(1+\sqrt{3})}\,\text{Cos}[x] - \sqrt{2\,(-1+\sqrt{3})\,(1+\sqrt{3})}\,\text{Cos}[x]\,\text{Cos}\left[\sqrt{1+\sqrt{3}}\,x\right]^2 - \sqrt{2\,(-1+\sqrt{3})\,(1+\sqrt{3})}\,\text{Cos}[x]\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right]^2\right)\right]\right\}\right\}$$

Un caso particolare si ha per:

```
DSolve[{quarto, y[0] == 0, y'[0] == -3, y''[0] == 0, y'''[0] == 9}, y, x]
```

$$\left\{\left\{y \to \text{Function}\left[\{x\},\ \frac{1}{24\sqrt{1+\sqrt{3}}}\left(e^{-\sqrt{-1+\sqrt{3}}\,x}\left(-9\sqrt{2} + 3\sqrt{6} + 2\sqrt{1+\sqrt{3}} + 9\sqrt{2}\,e^{2\sqrt{-1+\sqrt{3}}\,x} - 3\sqrt{6}\,e^{2\sqrt{-1+\sqrt{3}}\,x} + 2\sqrt{1+\sqrt{3}}\,e^{2\sqrt{-1+\sqrt{3}}\,x} - 4\sqrt{1+\sqrt{3}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Cos}[x] + 4\sqrt{1+\sqrt{3}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Cos}\left[\sqrt{1+\sqrt{3}}\,x\right] - 4\sqrt{1+\sqrt{3}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Cos}[x]\,\text{Cos}\left[\sqrt{1+\sqrt{3}}\,x\right]^2 - 72\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right] - 18\sqrt{\frac{2\,(-1+\sqrt{3})}{1+\sqrt{3}}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right] + 6\sqrt{\frac{6\,(-1+\sqrt{3})}{1+\sqrt{3}}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right] - 4\sqrt{1+\sqrt{3}}\,e^{\sqrt{-1+\sqrt{3}}\,x}\,\text{Cos}[x]\,\text{Sin}\left[\sqrt{1+\sqrt{3}}\,x\right]^2\right)\right)\right]\right\}\right\}$$

```
Plot[Evaluate[y[x] /. %], {x, -7, 7},
 PlotStyle → DarkKhaki, Background → Azure]
```




```
- Graphics -
```

Analogamente alle equazioni possiamo trattare anche i sistemi. A volte è conveniente scrivere separatamente equazioni e condizioni iniziali, e congiungerli nel comando DSolve:

```
A = {
    {0, -2, 0},
    {2, 1, 0},
    {1, 0, 1}
   };

Y[t_] := {x[t], y[t], z[t]};

sislin = (#1 == #2) &~ MapThread ~ {Y'[t], A.Y[t]}
```

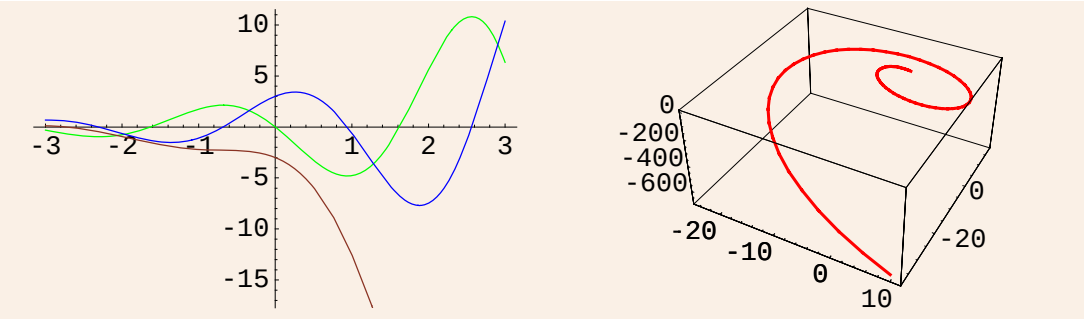
$$\{x'[t] == -2\,y[t],\ y'[t] == 2\,x[t] + y[t],\ z'[t] == x[t] + z[t]\}$$

```
valiniziali = {x[0] == 0, y[0] == 3, z[0] == -3};

DSolve[Join[sislin, valiniziali], {x, y, z}, t]
```

$$\left\{\left\{x \to \text{Function}\left[\{t\},\ -4\sqrt{\frac{3}{5}}\ e^{t/2}\ \text{Sin}\left[\frac{\sqrt{15}\ t}{2}\right]\right],\right.\right.$$
$$y \to \text{Function}\left[\{t\},\ \frac{1}{5}\ e^{t/2}\left(15\,\text{Cos}\left[\frac{\sqrt{15}\ t}{2}\right] + \sqrt{15}\ \text{Sin}\left[\frac{\sqrt{15}\ t}{2}\right]\right)\right],\ z \to$$
$$\left.\left.\text{Function}\left[\{t\},\ -\frac{1}{10}\ e^{t/2}\left(45\,e^{t/2} - 15\,\text{Cos}\left[\frac{\sqrt{15}\ t}{2}\right] - \sqrt{15}\ \text{Sin}\left[\frac{\sqrt{15}\ t}{2}\right]\right)\right]\right\}\right\}$$

Visualizziamo separatemente, prima, e con un grafico parametrico dopo:

```
DisplayTogetherArray[{
    Plot[#, {t, -3, 3}, PlotStyle → {Green, Blue, BurntUmber}],
    ParametricPlot3D[Join[#, {{Red, Thickness[0.01]}}],
     {t, 0, 5}, BoxRatios -> {1, 1, .5}]} & @@
  {{x[t], y[t], z[t]} /. %[[1]]},
 Background -> Linen
]
```




```
- GraphicsArray -
```

Molte volte abbiamo a che fare con equazioni non lineari, invece che lineari, che sono utili in parecchi casi reali. Un esempio è l'equazione logistica $y'(t) = r\left(1 - \frac{y(t)}{K}\right) y(t)$, detta anche di Verhlust, che altro non è se un'equazione di Riccati:

```
eqlog = y'[t] == r (1 - (y[t] / K)) * y[t]
```

$$y'[t] == r\, y[t] \left(1 - \frac{y[t]}{K}\right)$$

```
DSolve[eqlog, y, t]
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{e^{r\,t+K\,C[1]}\,K}{-1 + e^{r\,t+K\,C[1]}}\right]\right\}\right\}$$

Quest'equazione serve per descrivere l'andamento di una popolazione. Il tasso di crescita è dato da *r*, mentre *K* rappresenta la saturazione della popolazione: avremo una crestica più o meno veloce, a seconda della popolazione iniziale, che viene definita nelle condizioni iniziali:

```
andamento = DSolve[{eqlog, y[0] == α}, y, t]
```

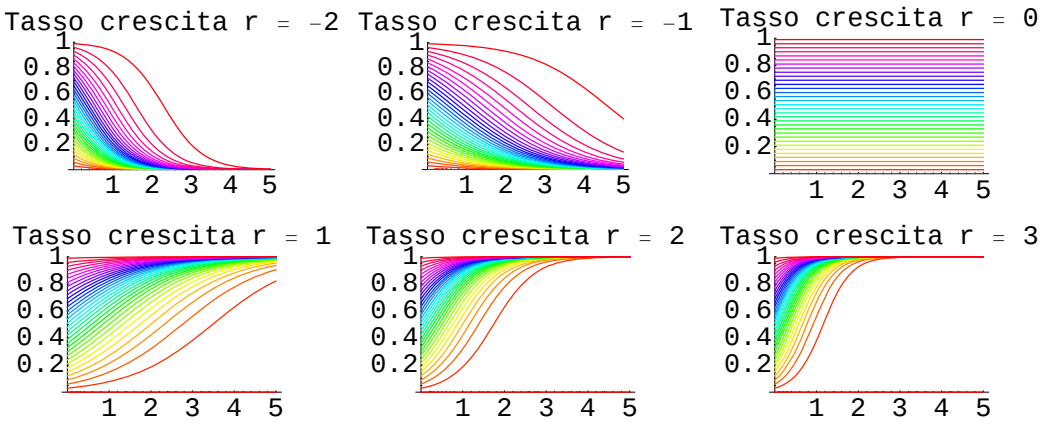
```
- Solve::ifun :
   Inverse functions are being used by Solve, so some solutions may not
     be found; use Reduce for complete solution information. More…
```

$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{e^{r\,t}\,K\,\alpha}{K - \alpha + e^{r\,t}\,\alpha}\right]\right\}\right\}$$

Il warning ottenuto è dato dall'inversione dell'esponenziale, ma siccoma sappiamo essere nel nostro caso una funzione monotona, possiamo ignorarlo. In questo modo possiamo plottare il grafico per

diversi valori iniziali della popolazione. Se normalizziamo il tutto, quindi ponendo $K = 1$, allora possiamo disegnare diversi grafici, che rappresentano l'andamento della popolazione che varia a seconda del tasso di crescita e della popolazione iniziale. Ho dovuto lavorare un poco per ottenere un array bidimensionale, ma è sempre meglio di dover scrivere i vari plot singolarmente:

```
DisplayTogetherArray[
 {Take[#, 3], Take[#, -3]} &@
  (DisplayTogether /@
    Table[
     Plot[
      Evaluate[
       y[t] /. andamento[[1]] /. {α → a, r → b, K → 1}
      ],
      {t, 0, 5}, PlotRange → {0, 1}, PlotStyle → Hue[a],
      PlotLabel -> "Tasso crescita r = " <> ToString[b]
     ],
     {b, -2, 3}, {a, 0, 1, .03}
    ])
]
```




```
- GraphicsArray -
```

Possiamo vedere che all'aumentare oppure al diminuire del tasso di crescita, il raggiungimento della saturazione (o l'estinzione, nel caso di tasso di crescita negativo), sia più o meno rapido.

Nelle nostre condizioni al contorno, possiamo anche specificare l'infinito, come nel caso seguente di un'equazione non lineare del secondo ordine:

```
nonlinord2 = y''[x] / 3 == y[x]^3 - y[x]
```

$$\frac{y''[x]}{3} == -y[x] + y[x]^3$$

```
DSolve[nonlinord2, y, x]
```

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

$$\left\{\left\{y \to \text{Function}\left[\{x\}, i\sqrt{3}\sqrt{\frac{1}{-3+\sqrt{3}\sqrt{3-2C[1]}}}\,\text{JacobiSN}\left[-\frac{1}{\sqrt{2}}\left(\sqrt{\left(3x^2+\sqrt{3}x^2\sqrt{3-2C[1]}+6xC[2]+2\sqrt{3}x\sqrt{3-2C[1]}C[2]+3C[2]^2+\sqrt{3}\sqrt{3-2C[1]}C[2]^2\right)}\right), \frac{3-\sqrt{9-6C[1]}}{3+\sqrt{9-6C[1]}}\right] - i\sqrt{\frac{1}{-3+\sqrt{3}\sqrt{3-2C[1]}}}\sqrt{3-2C[1]}\,\text{JacobiSN}\left[-\frac{1}{\sqrt{2}}\left(\sqrt{\left(3x^2+\sqrt{3}x^2\sqrt{3-2C[1]}+6xC[2]+2\sqrt{3}x\sqrt{3-2C[1]}C[2]+3C[2]^2+\sqrt{3}\sqrt{3-2C[1]}C[2]^2\right)}\right), \frac{3-\sqrt{9-6C[1]}}{3+\sqrt{9-6C[1]}}\right]\right]\right\},$$

$$\left\{y \to \text{Function}\left[\{x\}, i\sqrt{3}\sqrt{\frac{1}{-3+\sqrt{3}\sqrt{3-2C[1]}}}\,\text{JacobiSN}\left[\frac{1}{\sqrt{2}}\left(\sqrt{\left(3x^2+\sqrt{3}x^2\sqrt{3-2C[1]}+6xC[2]+2\sqrt{3}x\sqrt{3-2C[1]}C[2]+3C[2]^2+\sqrt{3}\sqrt{3-2C[1]}C[2]^2\right)}\right), \frac{3-\sqrt{9-6C[1]}}{3+\sqrt{9-6C[1]}}\right] - i\sqrt{\frac{1}{-3+\sqrt{3}\sqrt{3-2C[1]}}}\sqrt{3-2C[1]}\,\text{JacobiSN}\left[\frac{1}{\sqrt{2}}\left(\sqrt{\left(3x^2+\sqrt{3}x^2\sqrt{3-2C[1]}+6xC[2]+2\sqrt{3}x\sqrt{3-2C[1]}C[2]+3C[2]^2+\sqrt{3}\sqrt{3-2C[1]}C[2]^2\right)}\right), \frac{3-\sqrt{9-6C[1]}}{3+\sqrt{9-6C[1]}}\right]\right]\right\}\right\}$$

Possiamo andare a trovarci una soluzione particolare, andando ad invocare la derivata calcolata all'infinito, il che permette di dire che la funzione deve avere un asintoto orizzontale:

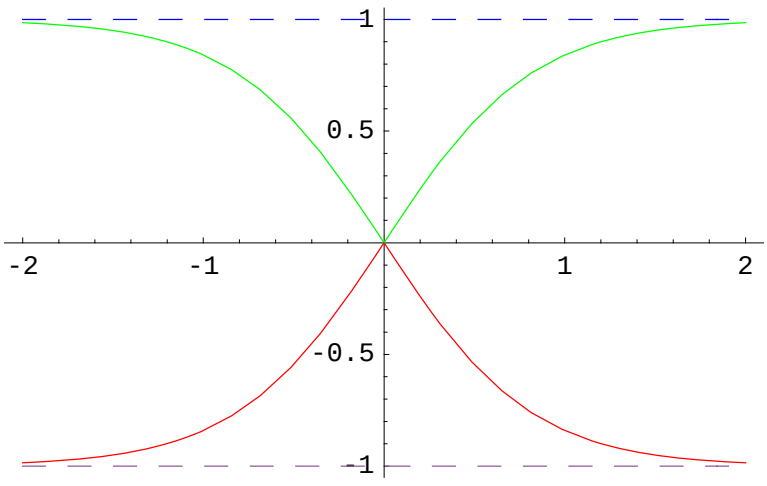
```
DSolve[{nonlinord2, y[0] == 0, y'[∞] == 0}, y, x]
```

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

– DSolve::bvlim :
   For some branches of the general solution, unable to compute the limit
     at the given points. Some of the solutions may be lost. More…

– DSolve::bvlim :
   For some branches of the general solution, unable to compute the limit
     at the given points. Some of the solutions may be lost. More…
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, -\text{Tanh}\left[\sqrt{\frac{3}{2}}\ \sqrt{x^2}\right]\right]\right\},\ \left\{y \to \text{Function}\left[\{x\}, \text{Tanh}\left[\sqrt{\frac{3}{2}}\ \sqrt{x^2}\right]\right]\right\}\right\}$$

Nonostante i warning, che avvertono che non possiamo trovare soluzioni per tutti i rami, abbiamo ottenuto due soluzioni che soddisfano la nostra condizione al contorno

```
Plot[{y[x] /. %[[1]], y[x] /. %[[2]], 1, -1},
 {x, -2, 2}, PlotRange → All, PlotStyle → {{Red}, {Green},
   {Dashing[{0.03}], Blue}, {Dashing[{0.03}], Violet}}]
```




```
- Graphics -
```

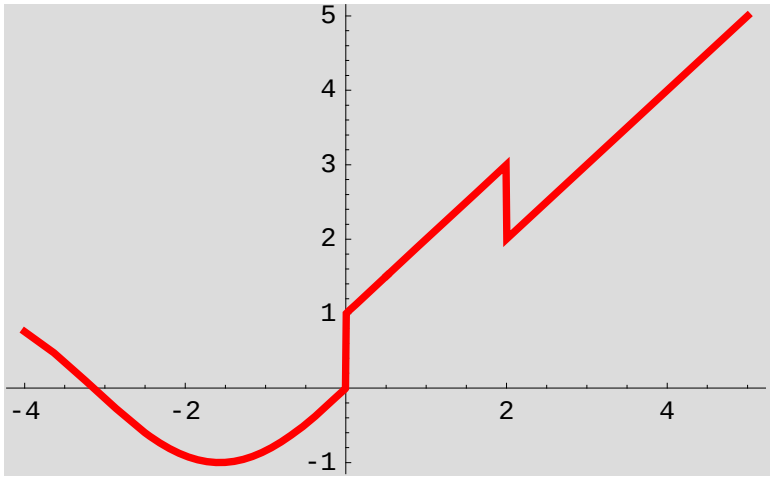
Una caratteristica molto importante, spuntata dalla versione 5.0 di *Mathematica*, è la caratteristica di poter utilizzare come coefficienti non costanti funzioni definite a tratti, tramite Piecewise. Possiamo quindi definire delle funzioni definite a tratti, come la seguente:

```
tratto = PiecewiseExpand[UnitStep[x] - UnitStep[x - 2] + Max[x, Sin[x]]]
```

$$\begin{cases} x & (x \geq 2 \&\& x - \sin[x] \geq 0) \,||\, (x < 0 \&\& x - \sin[x] \geq 0) \\ 1 + x & 0 \leq x < 2 \&\& x - \sin[x] \geq 0 \\ \sin[x] & (x \geq 2 \&\& x - \sin[x] < 0) \,||\, (x < 0 \&\& x - \sin[x] < 0) \\ 1 + \sin[x] & \text{True} \end{cases}$$

Come possiamovedere otteniamo una funzione definita a tratti.

```
Plot[tratto, {x, -4, 5},
 PlotStyle → {Red, Thickness[0.01]}, Background → Gainsboro]
```




```
- Graphics -
```

Possiamo utilizzare quindi anche questi coefficienti. Visualizziamo il risultato generico (stavolta in forma tradizionale. Troppa *Mathematica* d'estate mi fa male...):

```
eqtratto = y'[x] Max[x, x^2] == y[x] + x;
```

```
DSolve[eqtratto, y, x] // TraditionalForm
```

$$\left\{\left\{y \to \text{Function}\left[\{x\}, e^{\begin{cases} -\frac{1}{x} & x \leq 0 \\ \log(x) & 0 < x \leq 1 \\ 1 - \frac{1}{x} & \text{True} \end{cases}} c_1 + e^{\begin{cases} -\frac{1}{x} & x \leq 0 \\ \log(x) & 0 < x \leq 1 \\ 1 - \frac{1}{x} & \text{True} \end{cases}} \left(\begin{cases} -\text{Ei}\left(\frac{1}{x}\right) & x \leq 0 \\ \log(x) & 0 < x \leq 1 \\ \frac{\text{Ei}(1)}{e} - \frac{\text{Ei}\left(\frac{1}{x}\right)}{e} & \text{True} \end{cases} \right)\right]\right\}\right\}$$

Anche in questo caso possiamo imporre delle condizioni iniziali opportune per il nostro problema:

```
DSolve[{eqtratto, y[1] == 0}, y, x]
```

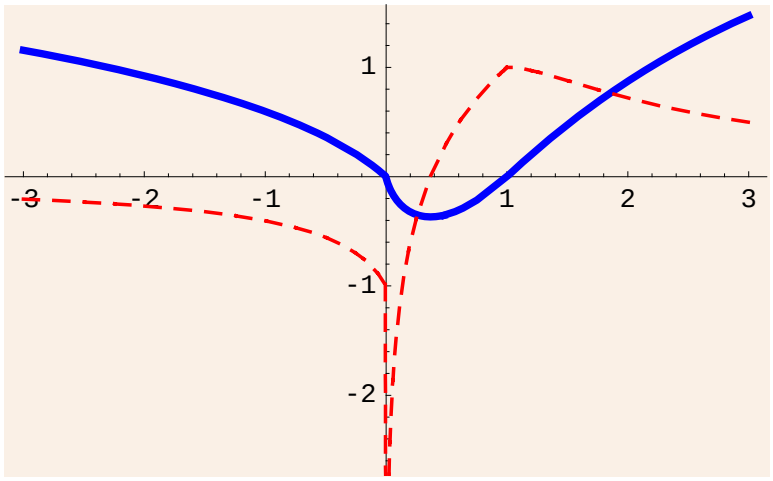
$$\left\{\left\{y \to \text{Function}\left[\{x\},\ e^{\begin{cases} -\frac{1}{x} & x\leq 0 \\ \text{Log}[x] & 0<x\leq 1 \\ 1-\frac{1}{x} & \text{True} \end{cases}} \left(\begin{cases} -\text{ExpIntegralEi}\left[\frac{1}{x}\right] & x \leq 0 \\ \text{Log}[x] & 0 < x \leq 1 \\ \frac{\text{ExpIntegralEi}[1]}{e} - \frac{\text{ExpIntegralEi}\left[\frac{1}{x}\right]}{e} & \text{True} \end{cases}\right)\right]\right\}\right\}$$

Possiamo plottare adesso la funzione, assieme alla sua derivata, per far vedere meglio dove sono presenti le discontinuità nella derivata, che diventano cuspidi nella funzione

```
Plot[
 Evaluate[{y[x], D[y[x], x]} /. %], {x, -3, 3},
 PlotStyle → {{Blue, Thickness[0.01]},
   {Red, Thickness[0.005], Dashing[{0.02}]}},
 Background → Linen
]
```

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

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



- Graphics -

Come abbiamo visto, abbiamo il plottaggio della soluzione, a parte i warning dovuti alla singolarità nel punto 0.

Con questo, credo di aver concluso la panoramica sulla risoluzione simbolica delle equazioni differenziali. Credo che adesso possiamo passare alla risoluzione numerica, che dal punto di vista teorico è più difficile, dato che fino ad adesso ha fatto tutto DSolve...

Risoluzione numerica

Quando non possiamo trovare la soluzione esatta di un'equazione differenziale, magari perchè non si riesce a trovarla, l'equazione è troppo difficile da risolvere etc etc etc, allora si rende necessaria la risoluzione numerica.

- **NDSolve**

Il comando di *Mathematica* dedito a questo fine è NDSolve. Il suo utilizzo per equazioni abbastanza elementari è abbastanza semplice. Basta infatti specificare l'equazione differenziale (oppure il sistema), come abbiamo fatto per DSolve, con un paio di modifiche: prima di tutto, siccome la soluzione è numerica e non simbolica, NDSolve non utilizza i parametri indeterminati $C[n]$, per cui abbiamo sempre bisogno di specificare le condizioni al contorno del nostro problema, per trovare una soluzione esclusivamente numerica. Inoltre, è necessario anche specificare il range entro cui vogliamo calcolare la funzione. Infatti *Mathematica* crea una funzione interpolata all'interno dell'intervallo definito: al di fuori dell'intervallo dobbiamo utilizzare delle estrapolazioni, che rendono impreciso il risultato all'aumentare della distanza dal punto.

In teoria basterebbe estendere il range ad un intervallo più grande di quello che ci interessa, per evitarci problemi. Tuttavia questo può essere eccessivamente dispendioso in termini di tempo di calcolo della nostra funzione interpolata, oltre che eccessivo in termini di memoria e di pesantezza di gestione della funzione. Occorre sempre studiarsi prima il problema per cercare l'intervallo che ci interessa, senza sprecare tempo prezioso. Se questo appare abbastanza superfluo per problemi semplici, può diventare determinante quando i calcoli si fanno pesanti, come per esempio si ha nelle equazioi di Navier-Stokes.

Inoltre, a seconda della natura delle equazioni, sono adatti certi algoritmi di risoluzione rispetto ad altri. Trovare il giusto algoritmo è sempre stato un problema; NDSolve effettua ogni volta un pre-processing della nostra equazione, cercando di scoprire quale algoritmo meglio si presta alla risoluzione del nostro problema. Comunque in casi particolarmente ostici, oppure quando semplicemente vogliamo maggior controllo su quello che facciamo, possiamo selezionare manualmente le varie opzioni, per rendere più specifico il risultato e forzare NDSolve a lavorare come vogliamo noi.

Data l'estrema generalizzazione della risoluzione numerica, possiamo trovare soluzioni per tutti i tipi di equazioni differenziali, tempo di calcolo permettendo; quindi lavora tranquillamente anche con tutti i tipi di equazioni differenziali finora elencate nella risoluzione simbolica: ODE, PDE, DAE.

Vediamo un esempio per capire il funzionamento base di NDSolve:

```
esnum = NDSolve[{x'[t] == Sin[x[t] + Cos[t^2]], x[0] == 3}, x, {t, 0, 3 Pi}]
```

```
{{x → InterpolatingFunction[{{0., 9.42478}}, <>]}}
```

Possiamo notare come abbiamo utilizzato per forza anche le condizioni iniziali, e come assieme alla variabile indipendente sia stato specificato l'intervallo di integrazione dell'equazione differenziale. *Mathematica* naturalmante restituisce un errore se mettiamo un numero insufficiente di condizioni iniziali, oppure se non specifichiamo l'intervallo:

```
NDSolve[x'[t] == Sin[x[t] + Cos[t^2]], x, {t, 0, 3 Pi}]
```

```
— NDSolve::ndnco :
   The number of constraints (0) (initial conditions) is not equal to
     the total differential order of the system (1). More…
```

```
NDSolve[x′[t] == Sin[Cos[t^2] + x[t]], x, {t, 0, 3 π}]
```

```
NDSolve[{x'[t] == Sin[x[t] + Cos[t^2]], x[0] == 3}, x, t]
```

```
— NDSolve::ndlim :
   Range specification t is not of the form {x, xend} or {x, xmin, xmax}. More…
```

```
NDSolve[{x′[t] == Sin[Cos[t^2] + x[t]], x[0] == 3}, x, t]
```

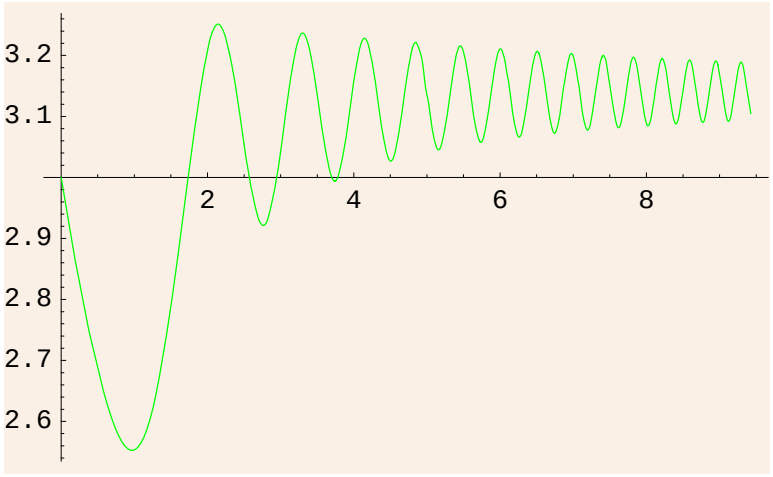
Una volta ottenuta la soluzione, possiamo utilizzarla come abbiamo fatto per il risultato di DSolve. Inatti anche in questo caso il risultato è formito come regola di sostituzione da utilizzare dove serve:

```
x[4] /. esnum
```

```
{3.13899}
```

Possiamo utilizzarla ovunque possiamo mettere una regola, ergo anche nel plottaggio, naturalmente:

```
Plot[x[t] /. esnum, {t, 0, 3 Pi},
 PlotRange → All, PlotStyle → Green, Background → Linen]
```

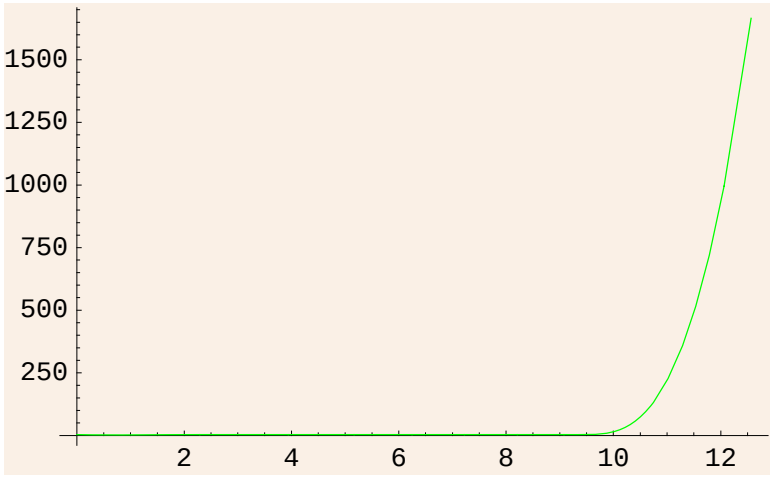



```
- Graphics -
```

Come potete notare la funzione viene smorzata, fino a stabilizzarsi attorno ad un valore limite. Tuttavia, se proviamo a calcolare la funzione in punti che cadono al di fuori dell'intervallo di definizione, viene utilizzata l'estrapolazione, che porta ad un grande errore, che cresce, in generale, man mano che ci si allontana dagli estremi dell'intervallo. *Mathematica* segna questo problema con un warning, da prendere abbastanza seriamente, anche se esegue comunque l'estrapolazione:

```
Plot[x[t] /. esnum, {t, 0, 4 Pi},
 PlotRange → All, PlotStyle → Green, Background → Linen]
```

```
– InterpolatingFunction::dmval :
   Input value {9.47541} lies outside the range of data in
     the interpolating function. Extrapolation will be used. More…

– InterpolatingFunction::dmval :
   Input value {9.99568} lies outside the range of data in
     the interpolating function. Extrapolation will be used. More…

– InterpolatingFunction::dmval :
   Input value {9.72325} lies outside the range of data in
     the interpolating function. Extrapolation will be used. More…

– General::stop : Further output of InterpolatingFunction::dmval will
     be suppressed during this calculation. More…
```




```
- Graphics -
```

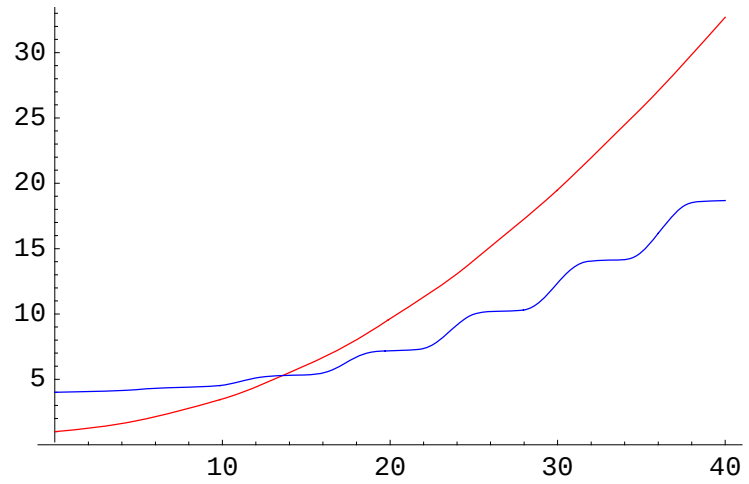
Come potete vedere, al di fuori dell'intervallo praticamente la funzione interpolata non serve a niente; questo è vero in particolar modo per le funzioni oscillanti come questa.

Analogamente a questo, possiamo anche risolvere sistemi di equazioni differenziali, come in questo esempio:

```
sol = NDSolve[{
   x''[t] == y'[t] / x[t],
   y'[t] == x[t] / (y[t] + 30 x[t]^Sin[t]),
   x[0] == 1,
   x'[0] == .1,
   y[0] == 4
  }, {x, y}, {t, 0, 40}]
```

```
{{x → InterpolatingFunction[{{0., 40.}}, <>],
  y → InterpolatingFunction[{{0., 40.}}, <>]}}
```

```
Plot[Evaluate[{x[t], y[t]} /. %], {t, 0, 40},
  PlotRange → All, PlotPoints → 200, PlotStyle → {Red, Blue}];
```



Il sistema può contenere un numero arbitrario di incognite, ovviamente. L'esempio serve solo per farvi capire come funziona il tutto.

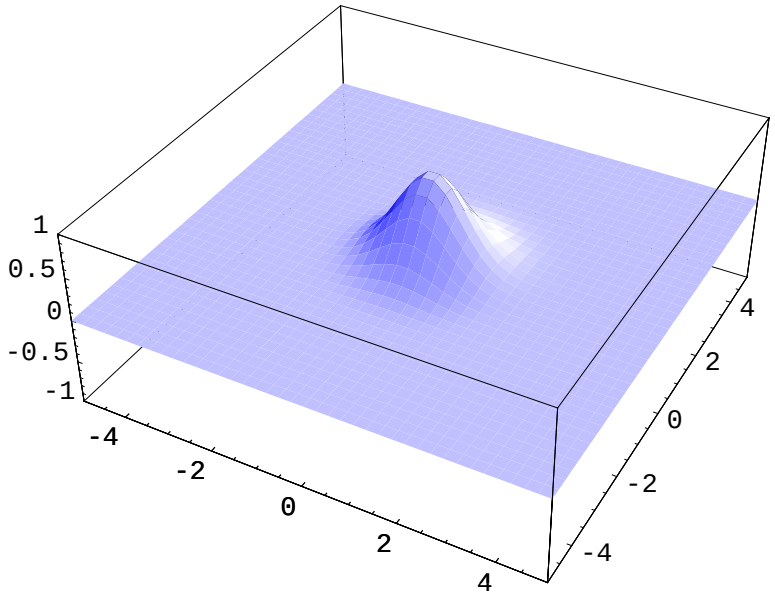
Possiamo risolvere numericamente anche delle PDE, come nel caso seguente, dove consideriamo l'evoluzione di un'equazione differenziale che dipende da due coordinate spaziali e da una temporale; questo esempio può prendere un po' di tempo in più rispetto alle equazioni banali. Potremmo dire che è un esempio quasi serio...

```
sol = NDSolve[{
   D[u[t, x, y], t, t] ==
    D[u[t, x, y], x, x] + D[u[t, x, y], y, y] - Sin[u[t, x, y]],
   u[0, x, y] == Exp[-(x^2 + y^2)], Derivative[1, 0, 0][u][0, x, y] == 0,
   u[t, -5, y] == u[t, 5, y] == Sin[t y / 2] / 2,
   u[t, x, -5] == u[t, x, 5] == 0
  },
  u, {t, 0, 25}, {x, -5, 5}, {y, -5, 5}]
```

```
— NDSolve::ibcinc :
   Warning: Boundary and initial conditions are inconsistent. More…

— NDSolve::eerr :
   Warning: Scaled local spatial error estimate of 508.4402959177667`
     at t = 25.` in the direction of independent variable x is much
     greater than prescribed error tolerance. Grid spacing with 91 points
     may be too large to achieve the desired accuracy or precision.  A
     singularity may have formed or you may want to specify a smaller
     grid spacing using the MaxStepSize or MinPoints options. More…

{{u → InterpolatingFunction[{{0., 25.}, {-5., 5.}, {-5., 5.}}, <>]}}
```

Adesso possiamo vedere il risultato tramite un'animazione che mostra l'evoluzione temporale della soluzione:

```
animazione = MoviePlot3D[First[u[t, x, y] /. sol],
  {x, -5, 5}, {y, -5, 5}, {t, 0, 25},
  PlotRange → {-1, 1}, PlotPoints → 40, Frames → 110, Mesh → False,
  LightSources → {{{2, 2, 2}, White}, {{-2, 2, 2}, Blue}}]
```



Se procedete con l'animazione, vedrete nel vostro schermo una simpatica cosuccia...

Quello che fa NDSolve è considerare ina sequenza di passi nella variabile indipendente *t*, e cercare il metodo migliore per la risoluzione dell'equazione differenziale. Inoltre, adatta automaticamente lo step size della variabile, in maniera da rispettare la precisione voluta da PrecisionGoal, che rappreseta l'errore relativo, ed AccuracyGoal, che rappresenta quello assoluto. Nel nostro caso abbiamo una discontinuità ai vertici della regione d'integrazione, perchè in due lati la condizione al contorno è 0, mentre negli altri 2 è Sin[*t*]: allora ai vertici ho contemporaneamente 0 e Sin[*t*], e *Mathematica* ci avvisa di questo, non riuscendo fra l'altro ad imporre un giusto step size per l'algoritmo, in quanto le discontinuità non sono ben digerite. In ogni caso otteniamo un risultato che giudico personalmente più che soddisfacente, almeno per poter fare un esempio pratico. In altri casi si arriva alla produzione di artefatti, che possono falsare il risultato. Possiamo anche aumentare la precisione della soluzione numerica, quando ci serve:

```
NDSolve[{
  x''[t] == x[t],
  x[0] == 1,
  x'[0] == 0
 },
 x, {t, 3},
 AccuracyGoal → 20, PrecisionGoal → 20, WorkingPrecision → 25
]
```

```
{{x → InterpolatingFunction[{{0, 3.0000000000000000000000000}}, <>]}}
```

Una volta ottenuta la soluzione con la precisione voluta, possiamo calcolare i risultati:

```
x[2] /. %
```

```
{3.76219569108363660919707}
```

Come possiamo vedere, abbiamo ottenuto una precisione superiore a quella di macchina, che avremmo ottenuto sfruttando le impostazioni di default.

Moltissimo c'è da dire ancora sulla risoluzione numerica delle equazioni differenziali, ma purtroppo il tempo mi manca.... Se avete dubbi leggete il manuale, sperimentate e non esistate a cercare nuove vie e nuovi comandi!!!!!