Assegnazione immediata, ritardata e condizionata. Definizione di funzioni composte

Assegnazione immediata

Esistono diversi modi per definire una funzione. L'approccio più immediato (*assegnazione immediata*), ha il seguente costrutto:

```
f[x_] = expr
```

dove <expr> è l'espressione analitica della funzione. Ad esempio:

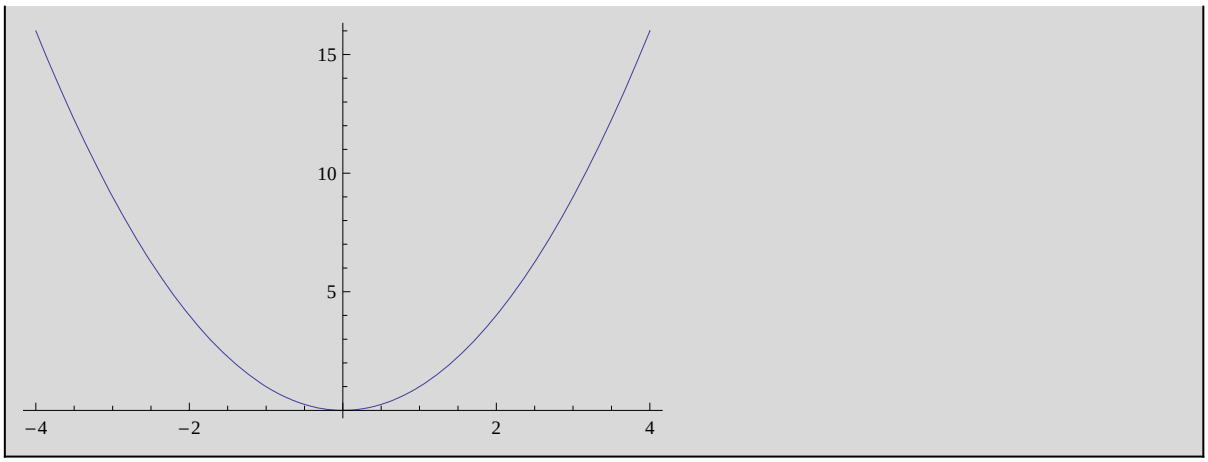
In[1]:=
```
f[x_] = x^2
```

Out[1]=
$$x^2$$

Una volta definita la funzione, possiamo determinare i valori assunti per opportuni valori della variabile indipendente, oppure possiamo graficarla in un dato sottoinsieme del campo di esistenza, calcolarne la derivata prima, seconda, etc. Ad esempio:

In[2]:=
```
Plot[
 f[x],
 {x, -4, 4}
]
```

Out[2]=



Assegnazione ritardata

Nell' *assegnazione ritardata* la funzione viene valutata solo quando viene richiamata dall'utente. La sintassi è:

In[3]:=
```
f[x_] := expr
```

In output non viene restituito nulla. Per visualizzare l'espressione della funzione, dobbiamo dare in input `f[x]`

In[3]:=
```
Clear[f]
```

In[4]:=
```
f[x_] := x^2
```

In[5]:=
```
f[x]
```

Out[5]=
$$x^2$$

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

Assegnazione condizionata

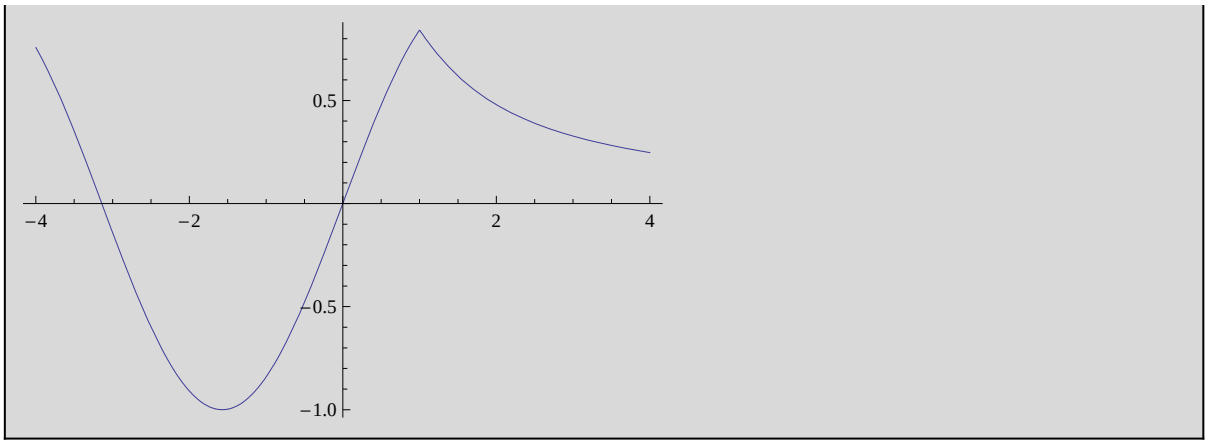
L'*assegnazione condizionata* si utilizza quando dobbiamo definire una funzione che ha espressioni diverse per differenti sottoinsiemi dell'insieme di definizione. Ad esempio, consideriamo la funzione reale di una variabile reale: $f(x) = \sin(x)$, se $x \in (-\infty,1]$; $f(x) = \sin\left(\frac{1}{x}\right)$, se $x \in (1,+\infty)$. Utilizziamo l'istruzione `/;` la cui sintassi è:

In[7]:=
```
f[x_] := Sin[x] /; x ≤ 1
f[x_] := Sin[1/x] /; x > 1
```

Grafichiamo f

In[9]:=
```
Plot[
 f[x],
 {x, -4, 4}
]
```

Out[9]=



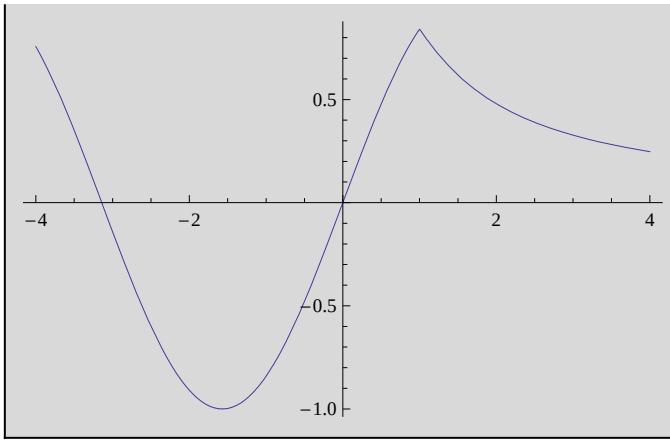
L'istruzione `Piecewise` ha lo stesso effetto:

In[10]:=
```
Clear[f]
```

In[11]:=
```
f[x_] := Piecewise[
  {
   {Sin[x], x ≤ 1},
   {Sin[1/x], x > 1}
  }
 ]
```

In[12]:=
```
Plot[
 f[x],
 {x, -4, 4}
]
```

Out[12]=



Spesso è richiesta l'azione degli operatori AND (`&&`) oppure OR (`||`). Ad esempio:

In[13]:=
```
Clear[f]
```

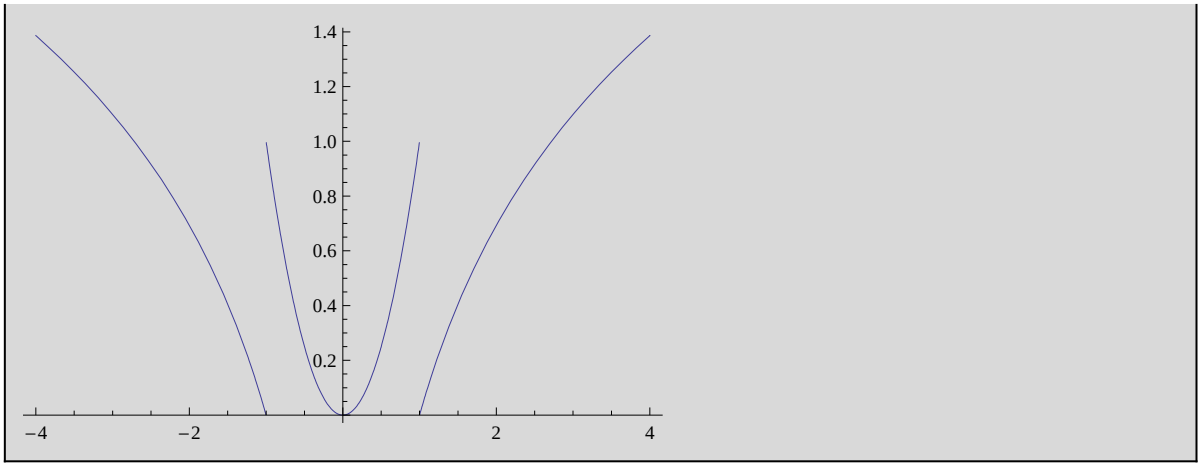
In[14]:=
```
f[x_] := x^2 /; (x ≥ -1 && x < 1)
f[x_] := Log[Abs[x]] /; (x < -1 || x > 1)
```

In[16]:=
```
f[1.1]
```

Out[16]=
```
0.0953102
```

In[17]:=
```
Plot[
 f[x],
 {x, -4, 4},
 Exclusions →
  {
   x == 1, x == -1
  }
]
```

Out[17]=



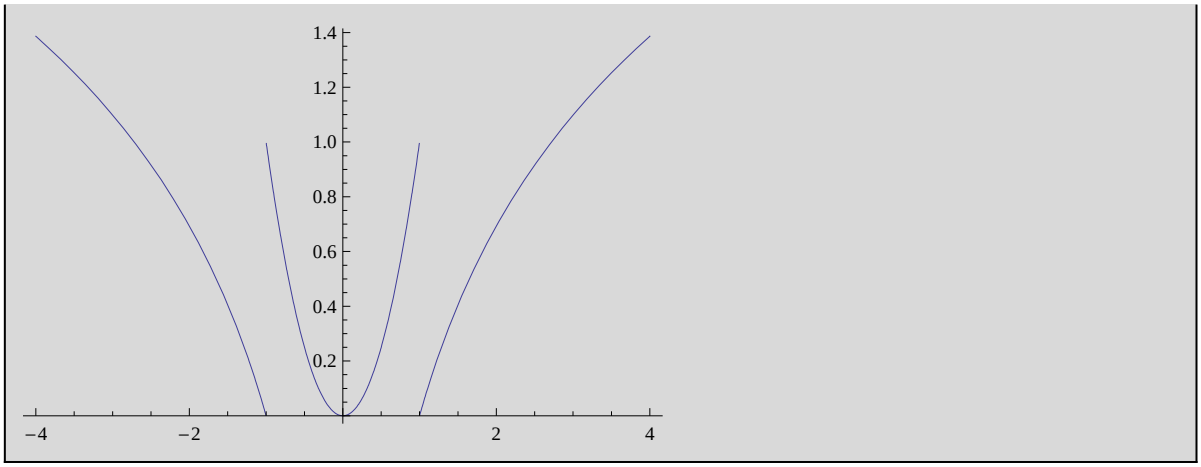
In[18]:=
```
Clear[f]
```

La stessa funzione si definisce con l'istruzione **`Piecewise:`**

In[19]:=
```
f[x_] := Piecewise[
  {
   {x^2, x ≥ -1 && x < 1},
   {Log[Abs[x]], x < -1 || x > 1}
  }
 ]
```

In[20]:=
```
Plot[
 f[x],
 {x, -4, 4},
 (*escludiamo i punti di discontinuità
  di prima specie*)
 Exclusions →
  {
   x == 1, x == -1
  }
]
```

Out[20]=



In[21]:=
```
Clear[f]
```

Definizione di funzioni composte

Si consideri la funzione reale di variabile reale $f(x) = x^5 + 6\,x^4 - 2\,x + \ln\left(x^2 + 1\right)$. Per ipotesi x è una funzione della variabile reale t, cioè $x = x(t)$, per cui abbiamo la funzione composta (sotto opportune condizioni sui rispettivi campi di esistenza) : $f\,(x(t)) = x(t)^5 + 6\,x(t)^4 - 2\,x(t) + \ln\left(x(t)^2 + 1\right)$. Se ad esempio è $x(t) = \sin(t)$, per definire la funzione composta utilizziamo *l'* istruzione **ReplaceAll**

In[22]:=
```
f[x_] := x^5 + 6 x^4 - 2 x + Log[x^2 + 1]
```

In[23]:=
```
ReplaceAll[f[x], x → Sin[t]]
```

Out[23]=
```
Log[1 + Sin[t]^2] - 2 Sin[t] + 6 Sin[t]^4 + Sin[t]^5
```

Tuttavia esiste una modalità postfissa per definire una funzione composta. Si tratta della potente istruzione `/.`

In[24]:=
```
f[x] /. x → Sin[t]
```

Out[24]=
```
Log[1 + Sin[t]^2] - 2 Sin[t] + 6 Sin[t]^4 + Sin[t]^5
```

In[25]:=
```
Clear[f]
```

Nel caso di più variabili, in `/.` vanno utilizzate le parentesi graffe. Ad esempio, supponiamo di avere la funzione:

In[26]:=
```
f[x_, y_] := Sqrt[x^2 - y^2]
```

con $x = \cos(t)$, $y = \sin(t)$, onde la funzione composta:

In[27]:=
```
f[x, y] /. {x → Cos[t], y → Sin[t]}
```

Out[27]=
$$\sqrt{\text{Cos}[t]^2 - \text{Sin}[t]^2}$$

L'istruzione `/.` è estremamente efficace nel caso di plotting di integrali di equazioni differenziali. Vedremo più avanti che la soluzioni di equazioni differenziali (ODE) si effettua con l'istruzione **`DSolve`**. Supponiamo di avere l'equazione differenziale del secondo ordine: $y'' - 2y' + y = \sin(x)$. Determiniamo un integrale particolare che soddisfa le condizioni iniziali $y(0) = 1,\ y'(0) = -1$

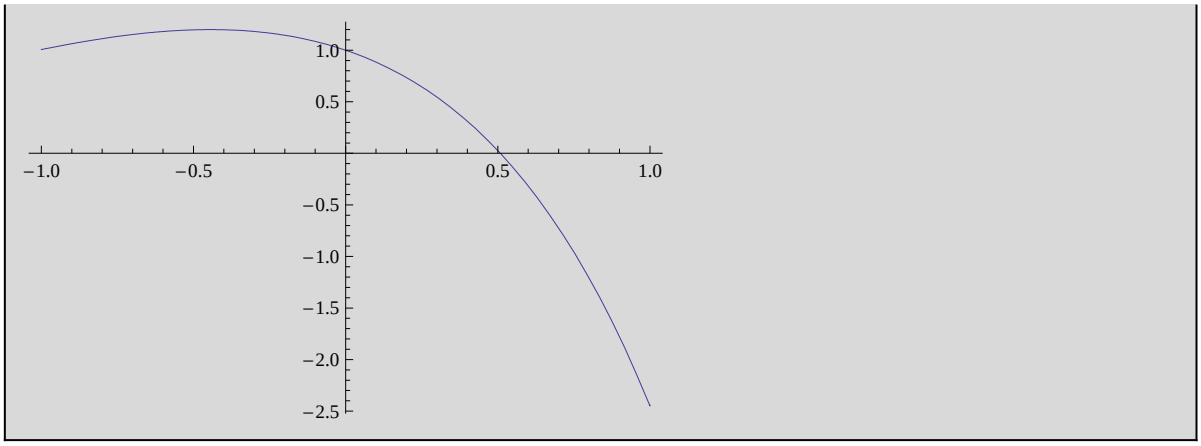
In[28]:=
```
sol = DSolve[
  {
   (*ODE*)
   y''[x] - 2 y'[x] + y[x] == Sin[x],
   (*condizioni iniziali*)
   y[0] == 1,
   y'[0] == -1
  },
  (*funzione incognita*)
  y[x],
  (*variabile indipendente*)
  x
 ]
```

Out[28]=
$$\left\{\left\{y[x] \to \frac{1}{2}\left(e^x - 3\, e^x\, x + \text{Cos}[x]\right)\right\}\right\}$$

Grafichiamo tale integrale particolare:

In[29]:=
```
Plot[
 y[x] /. sol,
 {x, -1, 1}
]
```

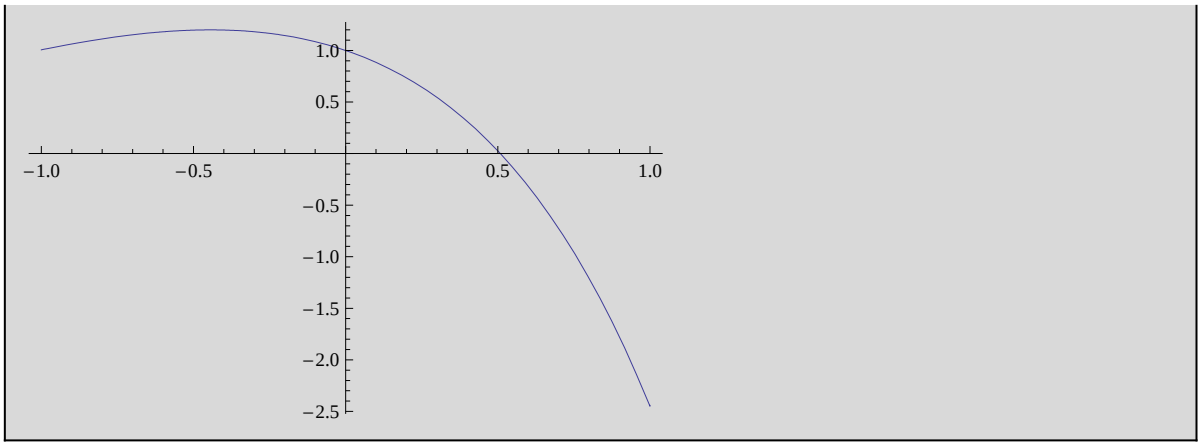
Out[29]=



Utilizzando **`ReplaceAll`**:

In[30]:=
```
Plot[
 ReplaceAll[y[x], sol],
 {x, -1, 1}
]
```

Out[30]=



L'istruzione Evaluate

Nel calcolo delle derivate è necessario definire la derivata con l'assegnazione immediata. Nel caso contrario, *Mathematica* resituisce un messaggio di errore, poichè cerca di calcolare prima la funzione in un punto *x* per poi calcolare la derivata. Ad esempio:

In[31]:=
```
Clear[f]
```

In[32]:=
```
f[x_] := x^10
```

Definiamo la derivata con l'istruzione **`D[f[x],x]`**

In[33]:=
```
Df[x_] := D[f[x], x]
```

In[34]:=
```
Df[2]
```

General::ivar : 2 is not a valid variable. ≫

Out[34]=
$$\partial_2 1024$$

Idem se plottiamo:

In[35]:=
```
Plot[
 Df[x],
 {x, 0, 1}
]
```

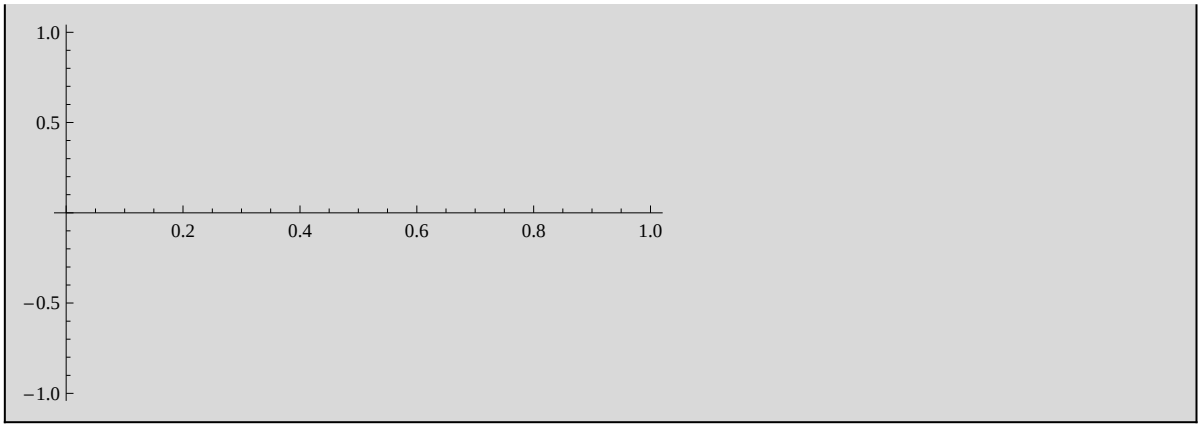
General::ivar : 0.000020428571428571428` is not a valid variable. ≫

General::ivar : 0.02042859183673469` is not a valid variable. ≫

General::ivar : 0.04083675510204081` is not a valid variable. ≫

General::stop : Further output of General::ivar will be suppressed during this calculation. ≫

Out[35]=



Allora proviamo a forzare il kernel con l'istruzione **`Evaluate`**:

In[36]:=
```
Clear[Df]
```

In[37]:=
```
Df[x_] := Evaluate[D[f[x], x]]
```

In[38]:=
```
Df[x]
```

Out[38]=
$$10\, x^9$$

In alternativa, avremmo potuto utilizzare l'assegnazione immediata:

In[39]:=
```
Clear[Df]
```

In[40]:=
```
Df[x_] = D[f[x], x]
```

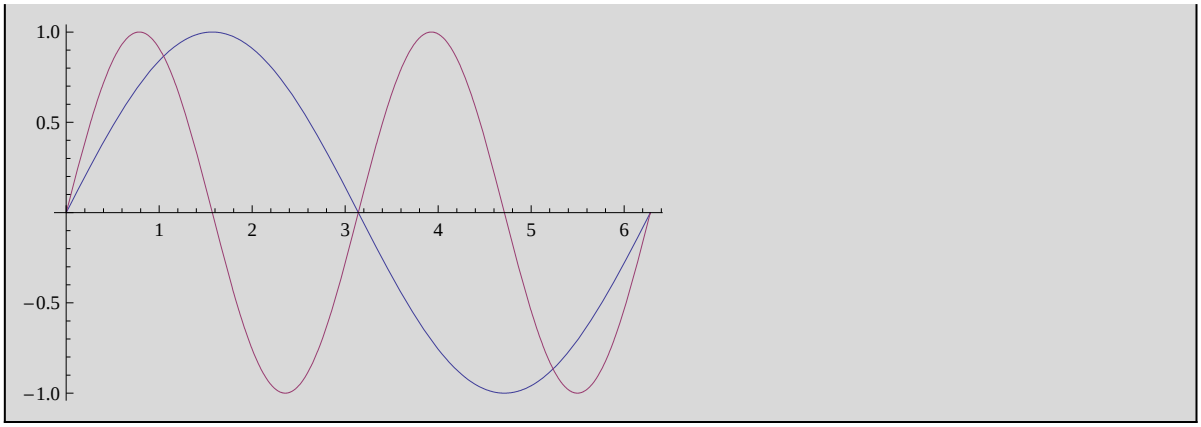
Out[40]=
$$10\,x^9$$

Grafici multipli

Un esempio banale:

In[41]:=
```
Plot[
 {Sin[x], Sin[2 x]},
 {x, 0, 2 π}
]
```

Out[41]=



In questo caso abbiamo introdotto le singole funzioni nell'istruzione **`Plot`**. Diversamente, immaginiamo di avere le funzioni $\sin(k\,x)$, dove $k = 1, 2, \ldots 10$. Scriviamo la lista :

In[42]:=
```
funzioni = Table[
  (*termine generale*)
  Sin[k * x],
  (*iteratore. Omettiamo k=1 poichè definito per default*)
  {k, 10}
 ]
```

Out[42]=
```
{Sin[x], Sin[2 x], Sin[3 x], Sin[4 x],
 Sin[5 x], Sin[6 x], Sin[7 x], Sin[8 x], Sin[9 x], Sin[10 x]}
```

Se grafichiamo:

In[43]:=
```
Plot[
 funzioni,
 {x, 0, 2 π}
]
```

Out[43]=