Capitolo 6

■ 6.1 Grafici 2D (...creare grafici piu' complicati...)

Abbiamo fino ad ora incontrato i comandi **Plot, ParametricPlot e ListPlot** per disegnare grafici bidimensionali. Vediamo ora come e' possibile creare immagini anche in modo diverso.

Mathematica rappresenta tutti i grafici in termini di insieme di *primitive grafiche*. Le *primitive grafiche* sono oggetti astratti nel senso che contengono le istruzioni per costruire l'immagine ma non l'immagine stessa. Per trasformare una *primitiva grafica* in un *oggetto grafico* si usa il comando **Graphics[{lista di primitive}]**. Per visualizzare l'oggetto grafico si usa il comando **Show[Graphics[{lista di primitive}]].**

Le *primitive grafiche* maggiormente usate sono

Point[{x,y}] rappresenta un punto nella posizione {x,y}

Line[{{x1,y1}...{xn,yn}}] rappresenta una linea che unisce i punti {x1,y1},...,{xn,yn}

Rectangle[{xmin,ymin},{xmax,ymax}] rappresenta un rettangolo pieno di estremi assegnati

Polygon[{{x1,y1},...,{xn,yn}}] rappresenta un poligono pieno di n lati con vertici assegnati

Circle[{x,y},r] rappresenta un cerchio centrato in {x,y} e raggio r

Circle[{x,y},{a,b}] rappresenta una ellisse centrata in {x,y} e semiassi a e b

Disk[{x,y},r] rappresenta un disco di centro {x,y} e raggio r

Disk[{x,y},{a,b}] rappresenta una ellisse piena di centro {x,y} e semiassi a e b

Text["expr",{x,y}] che scrive *expr* centrato nella posizione {x,y}

Le *direttive grafiche* (**PointSize**, **Thickness**, **Dashing**, **RGBColor**, **Graylevel** etc.) si specificano all'interno di **Graphics**, cioe' prima che Ma*thematica* crei l'*oggetto grafico*. La sintassi e' **Graphics[{direttiva grafica, primitiva grafica}]**. Se si considerano due o piu' *primitive grafiche* l'argomento di **Graphics** e' una lista: **Graphics[{{dir1,prim1},...,{dirn,primn}}]** oppure se le *direttive grafiche* sono le stesse per tutte le *primitive* basta indicarle una volta per tutte **Graphics[{direttiva grafica,prim1,...,primn}]** **.** La regola e' che una particolare *direttiva grafica* agisca su tutti gli elementi successivi della lista in cui e' contenuta.

Le *opzioni grafiche* (**AspectRatio**, **AxesLabel**, **PlotRange** etc.) che riguardano modifiche globali all'intero grafico, si possono indicare in **Show** cioe' **Show[Graphics[{lista di primitive}], opzione->valore].**

Esempio 1. Disegnamo tre punti nel piano nelle posizioni (0,0), (1,0) e (0,1).

```
p1=Point[{0,0}];p2=Point[{1,0}];p3=Point[{0,1}];
Show[Graphics[{p1,p2,p3}]]
```

Se li vogliamo ingrandire usiamo la *direttiva grafica* **PointSize[d],** dove d e' la frazione della larghezza complessiva del grafico. Dato che per *default* d=0.008, per raddoppiare la dimensione scriviamo 0.016 e cosi' via.

```
Show[Graphics[{{PointSize[.025],p1},{PointSize[.05],p2},p3}]]
```

Esempio 2. Disegnamo una linea punteggiata che congiunge (-1,-1) e (1,1), usando la *direttiva grafica* **Dashing.**

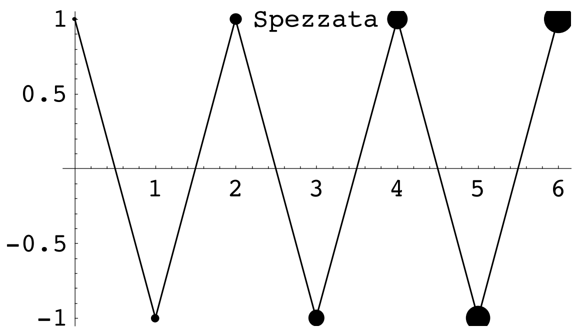
```
lineapunteggiata={Dashing[{.01,.05,.03}],Line[{{-1,-1},{1,1}}]};
scritta=Text["linea tratteggiata",{0,0}];
Show[Graphics[{lineapunteggiata,scritta}]]
```

Esempio 3. Disegnamo le linee che congiungono i punti $(n,(-1)^n)$ al variare di n da 0 a 6. Facciamo comparire gli assi coordinati con l'opzione grafica **Axes->True.**

```
linee=Line[Table[{n,(-1)^n},{n,0,6}]];
scritta=Text["Spezzata",{3,1}];
Show[Graphics[{linee,scritta}],Axes->True]
```

Proviamo adesso a disegnare anche i punti $(n,(-1)^n)$ sempre piu' visibili

```
punti = Table[{PointSize[0.008 * (n + 1)], Point[{n, (-1) ^n}]}, {n, 0, 6}];
Show[Graphics[{linee, scritta, punti}], Axes → True]
```



- Graphics -

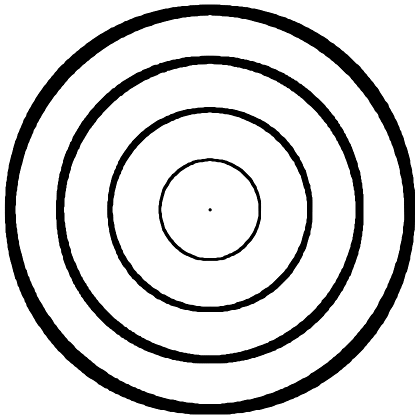
Esempio 4. Disegnamo quattro ellissi "piene" di centro (0,n) e semiassi n e 1/(2n), per n=1,2,3,4. Successivamente, con il comando **Circle**, le disegnamo "vuote".

```
ellissi=Table[Disk[{0,n},{n,1/(2 n)}],{n,4}]
Show[Graphics[ellissi],AspectRatio->Automatic]

ellissivuote=Table[Circle[{0,n},{n,1/(2 n)}],{n,4}]
Show[Graphics[ellissivuote],AspectRatio->Automatic]
```

Esempio 5. Disegnamo quattro cerchi di centro l'origine e raggio n, con n=1,2,3,4, disegnando il tratto sempre piu' spesso con la *direttiva grafica* **AbsoluteThickness[w]** (per *default* w=1)

```
c[n_]:={AbsoluteThickness[n],Circle[{0,0},n]};
Table[c[n],{n,4}];
Show[Graphics[{%,Point[{0,0}]}],AspectRatio->Automatic]
```



```
- Graphics -
```

Esempio 6. Disegnamo un triangolo "vuoto" di vertici (0,0), (1,0), (1/2,2), con la scritta "triangolo vuoto"

```
t1 = Graphics[Text["triangolo vuoto", {1 / 2, 1}]];
g1 = Graphics[Line[{{0, 0}, {1, 0}, {1 / 2, 2}, {0, 0}}]];
Show[{g1, t1}, Axes → True]
```

Disegnamo lo stesso triangolo, pero' "pieno", con la scritta "triangolo pieno"

```
t2 = Graphics[Text["triangolo pieno", {0.5, 2.1}]];
g2 = Graphics[Polygon[{{0, 0}, {1, 0}, {1 / 2, 2}}]];
Show[{g2, t2}, Axes → True]
```

Disegnamo un'ellisse vuota di centro l'origine e semiassi rispettivamente 2 e 1, con la scritta "ellisse vuota"

```
g3 = Graphics[Circle[{0, 0}, {2, 1}]];
t3 = Graphics[Text["ellisse vuota", {0, 0}]];
Show[{g3, t3}, AspectRatio → Automatic]
```

e ora la stessa ellisse piena

```
g4 = Graphics[Disk[{0, 0}, {2, 1}]];
t4 = Graphics[Text["ellisse piena", {0, 1.2}]];
Show[{g4, t4}, AspectRatio → Automatic]
```

Adesso visualizziamo questi disegni in un array

```
Show[GraphicsArray[{g1, t1}]]
Show[GraphicsArray[{{g1, t1}, {g2, t2}}]]
Show[GraphicsArray[{{g1, t1}, {g2, t2}, {g3, t3}}]]
Show[GraphicsArray[{{g1, t1}, {g2, t2}, {g3, t3}, {g4, t4}}],
 AspectRatio → Automatic]
```

Esempio 7. Disegnamo due cerchi di centro, rispettivamente, in (0,0) e (1,1) e raggio 2.

```
c1={0,0};c2={1,1};
cerchi={Circle[c1,2],Circle[c2,2]};
Show[Graphics[cerchi],AspectRatio->Automatic]
```

Visualizziamo anche i centri dei circonferenze

```
p1=Point[c1];p2=Point[c2];
Show[Graphics[{cerchi,p1,p2},AspectRatio->Automatic]]
```

Determiniamo le coordinate dei punti di intersezione delle due circonferenze e visualizziamoli

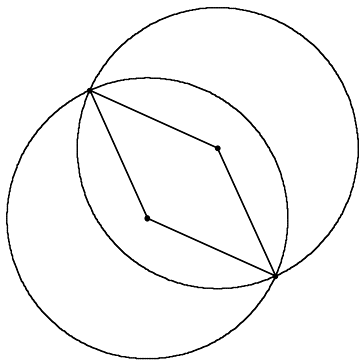
```
eq1=x^2+y^2-4==0;
eq2=(x-1)^2+(y-1)^2-4==0;
sol=Solve[{eq1,eq2},{x,y}]
Point[{x,y}]/.sol

Show[Graphics[{PointSize[0.016],cerchi,p1,p2,Point[{x,y}]/.sol},
AspectRatio->Automatic]]
```

Se vogliamo disegnare le linee che congiungono questi quattro punti in modo da formare un parallelogramma, usiamo il comando **Line**

```
q1={x,y}/.sol[[1]];
q2={x,y}/.sol[[2]];

Show[Graphics[{PointSize[0.016],cerchi,p1,p2,Point[q1],Point[q2],
Line[{c1,q2,c2,q1,c1}]}],AspectRatio->Automatic]
```



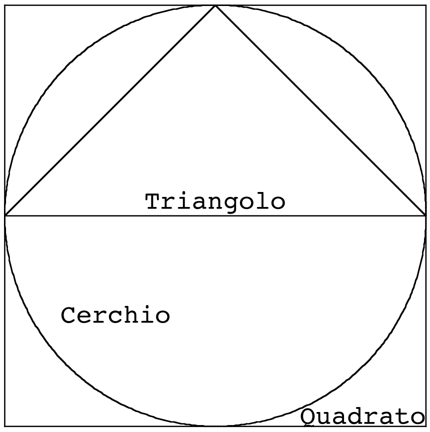
```
- Graphics -
```

Esempio 8. Creare un grafico che contiene, uno dentro l'altro, un quadrato, un cerchio, un triangolo. Ogni figura deve avere scritto vicino il suo nome.

```
c=Circle[{0,0},1];
t=Line[{{-1,0},{0,1},{1,0},{-1,0}}];
q=Line[{{-1,-1},{-1,1},{1,1},{1,-1},{-1,-1}}];
Show[Graphics[{c,t,q}],AspectRatio->Automatic]
```

e adesso le scritte

```
ctesto=Text["Cerchio",{2/3 Cos[5 Pi/4],2/3 Sin[5 Pi/4]}];
ttesto=Text["Triangolo",{0,.07}];
qtesto=Text["Quadrato",{0.7,-1+.05}];
Show[Graphics[{c,t,q,ctesto,ttesto,qtesto}],AspectRatio->Automatic]
```



- Graphics -

Esempio 9. Consideriamo il seguente problema algebrico (Porta, Davis, Uhl 1994): determinare i numeri positivi r tali che il sistema

$$(x-1)^2+(y-1)^2=2\;;\;\;(x+3)^2+(y-4)^2=r^2$$

ha una e una sola soluzione in x e y.

```
ClearAll["Global`*"]
Solve[{(x - 1) ^2 + (y - 1) ^2 == 2, (x + 3) ^2 + (y - 4) ^2 == r^2}, {x, y}]
{x, y} /. %
{x1, y1} = %[[1]]
{x2, y2} = %%[[2]]
Solve[{x1 == x2, y1 == y2}, r] // Flatten
```

Si trova quindi la soluzione algebrica

```
r1=5+√2 ;r2=5-√2 ;
```

Disegnamo il cerchio centrato in (1,1) e raggio 2 e i cerchi di raggi r1 e r2 e centro (-3,4)

```
cerchio={Thickness[0.01],RGBColor[0,1,0],Circle[{1,1},Sqrt[2]]};
cerchio1={Thickness[0.01],RGBColor[1,0,0],Circle[{-3,4},r1]};
cerchio2={Thickness[0.01],RGBColor[0,0,1],Circle[{-3,4},r2]};
Show[Graphics[{cerchio,cerchio1,cerchio2}],Axes→
True,AspectRatio->Automatic]
```

Esempio 10. Costruiamo una funzione f(n) che costruisca la lista delle n radici ennesime dell'unita'

```
f[n_Integer /; n > 0] :=
 Module[{l}, l = Table[2 k Pi / n, {k, 0, n - 1}]; Cos[l] + I Sin[l]]
```

Ora costruiamo una funzione che le visualizzi

```
radici[n_Integer /; n > 0] :=
 Module[{l}, l = Table[{Cos[2 k Pi / n], Sin[2 k Pi / n]}, {k, 0, n - 1}];
  Show[Graphics[{Red, PointSize[0.05], Map[Point, l]},
    AspectRatio → Automatic, Axes → True]]]
```

• I grafici possono essere salvati su file per poi essere esportati in un altro testo (per esempio di Word, Excel, Tex). Per salvare un grafico in formato PDF e' sufficiente selezionare il grafico e scegliere nel menu **File**, **Print Selection** e poi **Save as PDF**. Per salvare il grafico *graphics* in un formato specifico si usa il comando **Display["file",graphics,"format"]**. Per esempio **Display["sinplot.GIF",g,"GIF"]** salva il grafico g nel file sinplot.-GIF, in formato grafico GIF. Si possono salvare grafici in formati, per esempio, EPS, PICT, GIF. Per una lista completa dei formati e delle varie opzioni rimandiamo al manuale.

Un'altra possibilita' e' usare il comando **Export["file",expr,"format"]** che esporta dati numerici, grafici, suoni in un file convertendoli nel formato scelto.

```
g = Plot[Sin[x], {x, 0, 4}]

Display["sinplot.gif", g, "GIF"](*salva in formato grafico GIF*)

Display["sinplot.EPS", g, "EPS"](*salva in formato EPS*)

Display["sinplot.PDF", g, "PDF"](*salva in formato PDF*)

Display["sinplot.PICT", g, "PICT"](*salva in formato PICT*)

Export["sinplot2.gif", g, "GIF"](*salva in formato grafico GIF*)

Export["sinplot2.EPS", g, "EPS"](*salva in formato EPS*)

Export["sinplot2.PDF", g, "PDF"](*salva in formato PDF*)

Export["sinplot2.PICT", g, "PICT"](*salva in formato PICT*)
```

■ 6.2 Grafici 3D

Le funzioni di una variabile si rapprentano come curve nel piano. Le funzioni di due variabili f(x,y) si visualizzano come superfici nello spazio. Per fare questo *Mathematica* ha diversi strumenti a disposizione. L'analogo del comando **Plot** e' **Plot3D[f,{x,xmin,xmax},{y,ymin,ym}]**

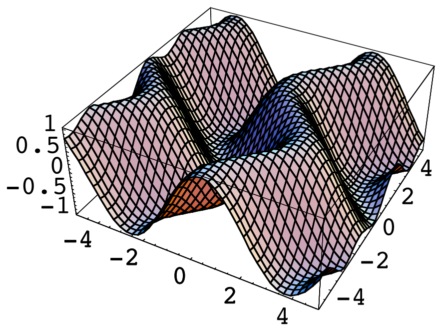
```
Plot3D[Sin[x+Sin[y]],{x,-5,5},{y,-5,5}]
```

Mathematica crea un *oggetto grafico* che denota con *SurfaceGraphics*.

Le opzioni per disegnare i grafici tri-dimensionali sono molte, alcune delle quali analoghe a quelle viste nel caso bi-dimensionale. Il comando e' **Plot3D[f,{x,xmin,xmax},{y,ymin,ymax}, opzione1→v1, opzione2→v2,...].** Vediamone alcune tra le piu' comuni. Per una lista completa rimandiamo all'**Help Browser** (nelle **Built-in Functions**, **Graphics and Sounds**, **3D Options**).

Possiamo chiedere a *Mathematica* di disegnare un grafico "piu' accurato" cioe' specificando, prima di disegnare il grafico, il numero di punti in ogni direzione in cui calcolare la funzione. Per far questo usiamo la *direttiva grafica* **PlotPoints->{nx,ny}** (per *default* n=15). **PlotPoints→n** vuol dire **PlotPoints→{n,n}**.

```
g1 = Plot3D[Sin[x + Sin[y]], {x, -5, 5}, {y, -5, 5}, PlotPoints → 40]
```

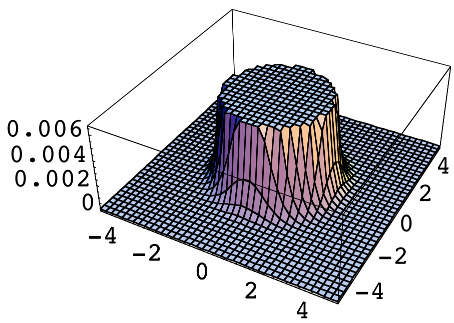



```
- SurfaceGraphics -
```

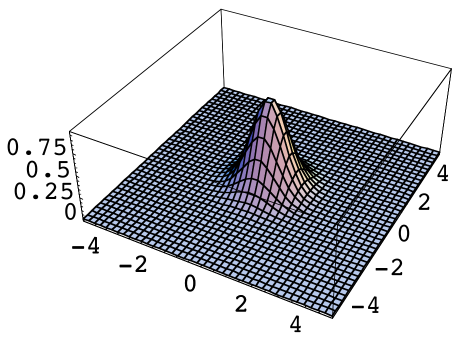
L'opzione**PlotRange→{a,b}** mostra la parte di grafico con a≤z≤b. **PlotRange** e' un'*opzione grafica* che possiamo cambiare all'interno di **Show**. Se vogliamo essere sicuri di visualizzare tutto il grafico scegliamo **PlotRange→All**

```
Show[g1,PlotRange->{-1,1}]
```

```
g2=Plot3D[Exp[-(x^2+y^2)],{x,-5,5},{y,-5,5},PlotPoints→40]
Show[g2,PlotRange→All]
```




```
- SurfaceGraphics -
```



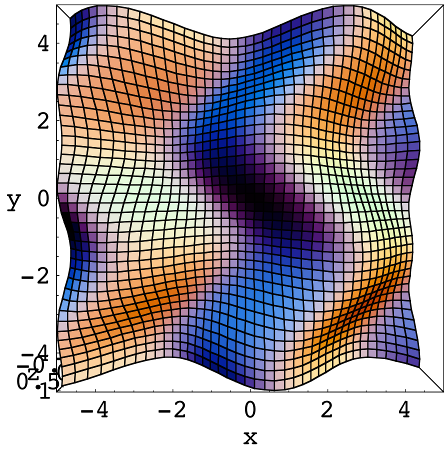

```
- SurfaceGraphics -
```

Possiamo dare un nome agli assi con **AxesLabel→{"nome1","nome2","nome3"}**

```
Show[g1,AxesLabel->{"x","y","z"}]
```

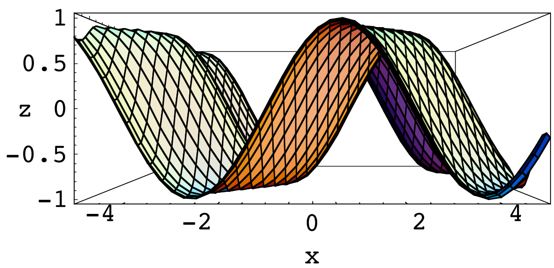
oppure possiamo ridisegnare il grafico da un altro punto di vista con l'opzione **ViewPoint->{x0,y0,z0}** (per *default* **ViewPoint->{1.3,-2.4,2}** rispetto al centro del parallelepipedo che contiene la superficie). Scegliere le coordinate (x0,y0,z0) non e' facile. Per aiutarvi andate nel menu' principale **Input** e selezionate il comando **3DView Selector**. Si apre una finestra con un sistema di riferimento che si puo' far ruotare con il mouse. Una volta individuato il punto di vista premete **Paste** e sul *Notebook* vi apparira' il comando **ViewPoint** con il punto richiesto.

```
Show[g1,ViewPoint->{0,0,2},AxesLabel->{"x","y","z"}]
(*da sopra*)
```




```
- SurfaceGraphics -
```

```
Show[g1,ViewPoint->{0,-2,0},AxesLabel->{"x","y","z"}]
(*di fronte*)
```




```
- SurfaceGraphics -
```

Altre *opzioni grafiche* sono, ad esempio

AspectRatio→r analoga all'opzione vista in due dimensioni;

Axes→False se non vogliamo visualizzare gli assi (per *default* **Axes→True**);

Boxed→False se non vogliamo disegnare la scatola che contiene la superficie;

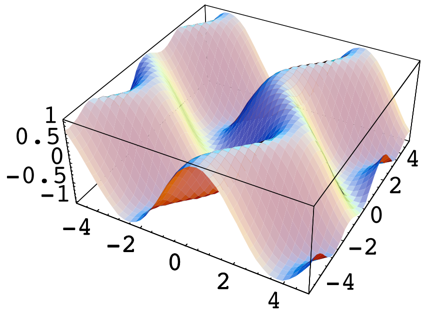
Mesh→False per impedire che una griglia xy compaia sulla superficie disegnata ;

Shading→False per disegnare la superficie in bianco;

Lighting→False per non colorare le ombre del grafico ma riprodurle in bianco e nero;

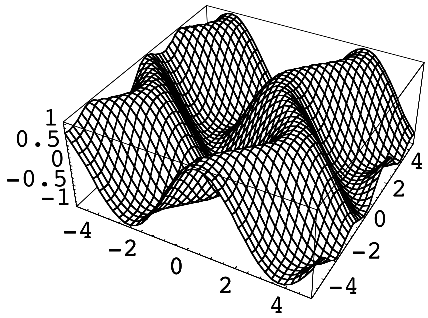
HiddenSurface→False per disegnare la superficie trasparente;

```
Show[g1, Mesh -> False]
```



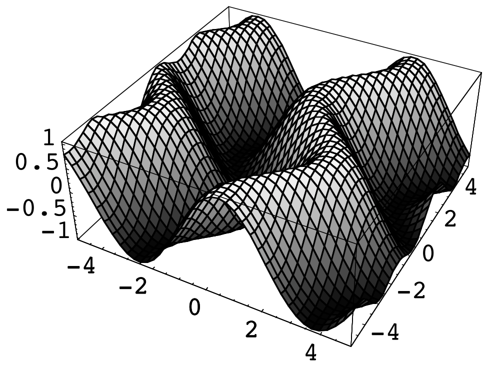
- SurfaceGraphics -

```
Show[g1,  Shading -> False]
```



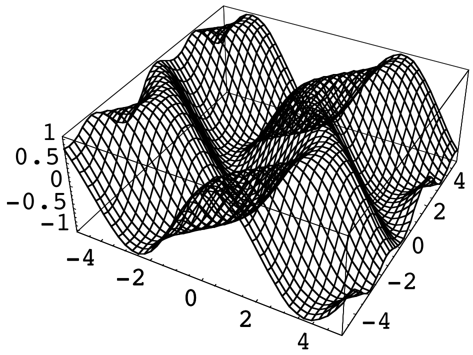
- SurfaceGraphics -

```
Show[g1,  Lighting -> False ]
```



- SurfaceGraphics -

```
Show[g1,  HiddenSurface -> False ]
```



- SurfaceGraphics -

Esempio 1. Disegnamo un paraboloide ellittico

```
g3 = Plot3D[x^2 + y^2, {x, -25, 25}, {y, -25, 25}, PlotPoints → 50]
```

Disegnamo un paraboloide iperbolico o a sella

```
g4 = Plot3D[x^2 - y^2, {x, -25, 25}, {y, -25, 25}, PlotPoints → 50]
```

Provate adesso a modificare le *opzioni grafiche* con **Show[g3,opzioni->...]** e **Show[g4,opzioni->...]**

La stessa superficie puo' essere vista in modi diversi. Per esempio il comando **ContourPlot[f,{x,xmin,xmax},{ymin,ymax}]** rappresenta la superficie con le linee di livello f(x,y)=costante. Disegnamo le linee di livello dell'ellissoide (...che sono ellissi...)

```
ContourPlot[2 x^2+3 y^2,{x,-5,5},{y,-2,2}]
```

oppure senza colorare le ombre

```
Show[%,ContourShading→False]
```

Si puo' specificare quante linee di livello disegnare con l'opzione **Contours→n**

```
ContourPlot[2 x^2+3 y^2,{x,-5,5},{y,-2,2},Contours→20]
```

• Per rappresentare curve nello spazio, rappresentate in forma parametrica, si usa il comando

ParametricPlot3D [{f1,f2,f3},{t,tmin,tmax}].

Esempio 2. Disegnamo l'elica cilindrica

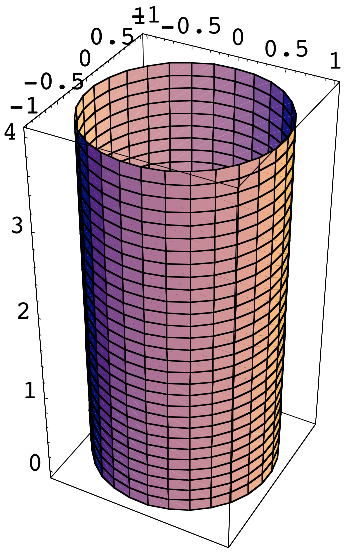
```
ParametricPlot3D[{Sin[t], Cos[t], t / 2}, {t, 0, 4 Pi}]
```

• Per disegnare una superficie rappresentata in forma parametrica si usa il comando

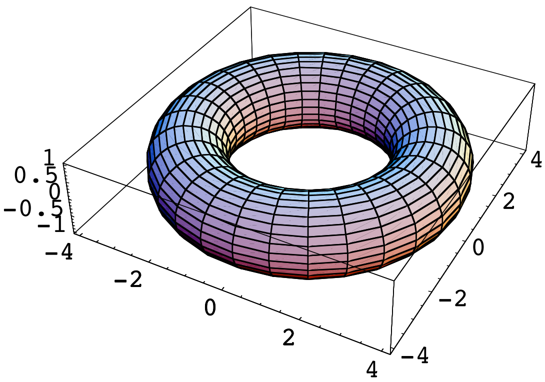
ParametricPlot3D[{f1,f2,f3},{u,umin,umax},{v,vmin,vmax}].

Dalla rappresentazione parametrica di alcune superfici "famose" otteniamo i seguenti grafici:

```
ParametricPlot3D[{Sin[t],Cos[t],u},{t,0,2 Pi},{u,0,4}];(*cilindro*)
```

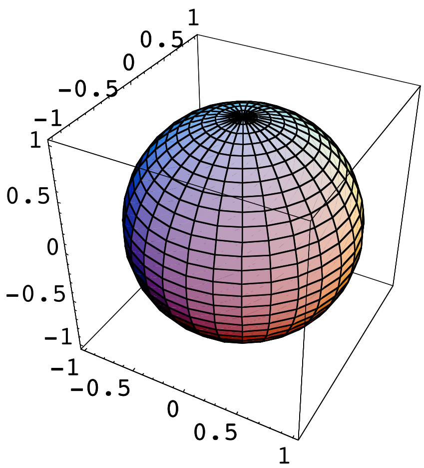


```
ParametricPlot3D[{Cos[u] (3+Cos[v]),Sin[u] (3+Cos[v]),
Sin[v]},{u,0,2 Pi},{v,0,2 Pi}](*toro*)
```



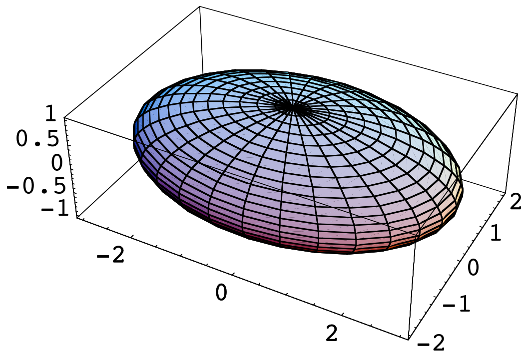
- Graphics3D -

```
ParametricPlot3D[{Cos[u] Cos[v],Sin[u] Cos[v],Sin[v]},
{u,0,2 Pi},{v,-Pi/2,Pi/2}](*sfera*)
```



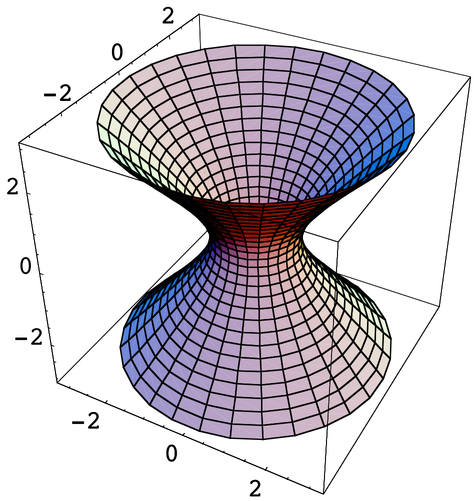
- Graphics3D -

```
ParametricPlot3D[{3 Cos[u] Cos[v],2 Sin[u] Cos[v],Sin[v]},
{u,0,2 Pi},{v,-Pi/2,Pi/2}](*ellissoide*)
```



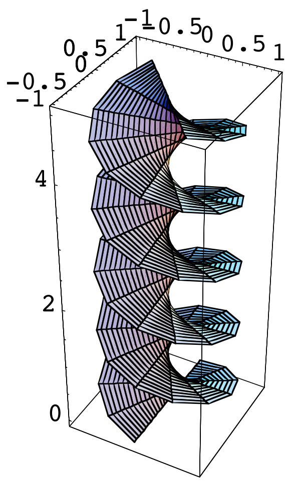

```
- Graphics3D -
```

```
ParametricPlot3D[{ Sin[u] Sqrt[1 + v^2], Cos[u] Sqrt[1 + v^2], v},
 {u, 0, 2 Pi}, {v, -3, 3}](*iperboloide a una falda*)
```




```
- Graphics3D -
```

```
ParametricPlot3D[{v Sin[u], v Cos[u], u / 3},
 {u, 0, 15}, {v, -1, 1}](*superficie elicoidale*)
```



```
- Graphics3D -
```

Sono disponibile molte *primitive grafiche* anche in tre dimensioni. Combinando le *primitive grafiche* si ottengono grafici tri-dimensionali. Le principali sono

Point[{x,y,z}] che rappresenta il punto di coordinate {x,y,z}

Line[{{x_1 ,y_1 ,z_1 },{x_2 ,y_2 ,z_2 },...}] che rappresenta le linee che congiungono i punti {x_1 ,y_1 ,z_1 },{x_2 ,y_2 ,z_2 },...

Polygon[{{x_1 ,y_1 ,z_1 },{x_2 ,y_2 ,z_2 },...}] che rappresenta il poligono pieno di vertici assegnati

Cuboid[{x_1 ,y_1 ,z_1 },{x_2 ,y_2 ,z_2 }] che rappresenta il parallelepipedo di vertici opposti assegnati

Text["expr",{x,y,z}] che scrive il testo *expr* centrato in {x,y,z}

Con il comando **Graphics3D** *Mathematica* genera l'oggetto grafico; con **Show** lo visualizza.

Esempio 3. Disegnamo il triangolo pieno di vertici random in (0,1)

```
punto := Table[Random[], {3}]
vertici := Table[punto, {3}]
Show[Graphics3D[Polygon[vertici]]]
```

Provate a eseguire piu' volte questi comandi

Esempio 4. Disegnamo il parallelepipedo di vertici opposti {0,1,12} e {3,14,5}

```
Show[Graphics3D[Cuboid[{0, 1, 12}, {3, 14, 5}], AspectRatio → Automatic]]
```

Alcune *direttive grafiche*, che devono essere specificate prima di disegnare il grafico, sono analoghe a quelle viste nel caso bidimensionale. Per esempio **PointSize**, **Thickness**, **Dashing** per disegnare punti e linee o anche **AbsolutePoint-Size**,**AbsoluteThickness**,**AbsoluteDashing.**

Esempio 5. Generiamo una lista di 20 punti con coordinate **Random** e disegnamoli ben visibili.

```
punto := Table[Random[], {3}]
ventipunti := Table[punto, {20}]
Show[Graphics3D[{PointSize[0.032], Map[Point, ventipunti]}]]
```

Esempio 6. Consideriamo la spezzata che congiunge i punti $((-1)^n, n, \frac{n}{2}\}$,per n=0,1,2,3,4,5,6 e disegnamola di spessore *d* volte l'ampiezza dell'intero grafico, con *d=0.02*

```
spezzata = Line[Table[{(-1)^n, n, n / 2}, {n, 0, 6}]];
Show[Graphics3D[{Thickness[.02], spezzata}]]
```

Adesso disegnamo la stessa figura con linea tratteggiata, di uguale spessore

```
Show[Graphics3D[{Thickness[.02], Dashing[{.05, .05}], spezzata}]]
```

• Numerosi sono i *packages* di grafica bi e tri-dimensionale. Rimando all' **Help Browser** per la lista dei *packages* standard. Per esempio il *package* **Graphics`Shapes`** contiene una lista delle *primitive grafiche* delle piu' comuni superfici. Carichiamolo

```
Needs["Graphics`Shapes`"]
```

Sphere[r,n,m] e' la primitiva grafica per visualizzare la sfera di raggio r, sfaccettata disegnando n(m-2)+2 poligoni

```
Show[Graphics3D[Sphere[1, 60, 80]]]
```

Visualizziamo un toro con la *primitiva grafica* **Torus[r1,r2,n,m]** che contiene le istruzioni per disegnare il toro di raggi r1 e r2 e mesh n e m (se non specifico nulla *Mathematica* usa valori di *default*)

```
Show[Graphics3D[Torus[2, 1, 14, 15]]]
```

Adesso visualizziamo il cono di raggio r e altezza h disegnando n poligoni, con la primitiva grafica **Cone[r,h,n]**

```
Show[Graphics3D[Cone[2, 4, 122]]]
```

MoebiusStrip[r1,r2,n] e' la *primitiva grafica* per visualizzare il nastro di Möbius di raggi r1 e r2, disegnato usando n poligoni.

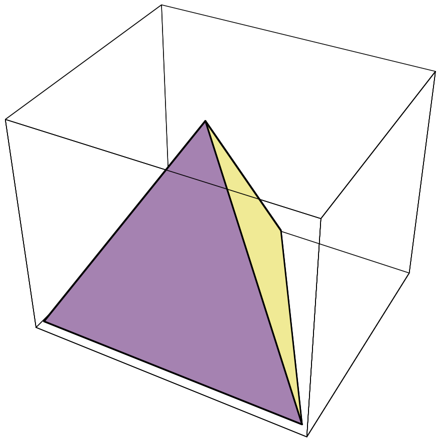
```
Show[Graphics3D[MoebiusStrip[2, 1, 80]]]
```

Il *package* **Graphics`Polyhedra`** contiene le p*rimitive grafiche* per disegnare alcuni poliedri regolari. Carichiamolo

```
Needs["Graphics`Polyhedra`"]
```

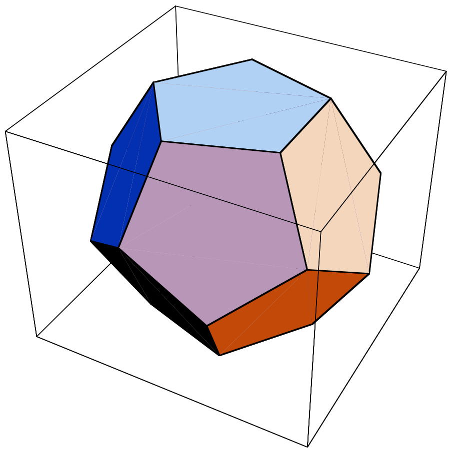
e proviamo a disegnare qualche poliedro

```
Show[Graphics3D[Tetrahedron[]]](*poliedro a 4 facce*)
```



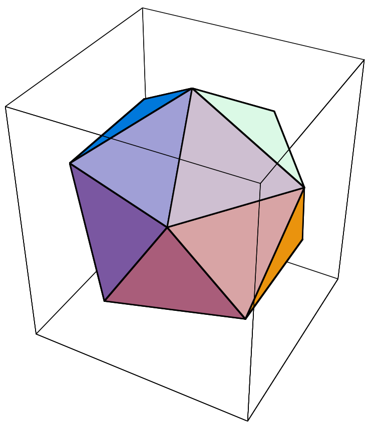
```
- Graphics3D -
```

```
Show[Graphics3D[Dodecahedron[]]](*poliedro a 12 facce*)
```



```
- Graphics3D -
```

```
Show[Graphics3D[Icosahedron[]]](*poliedro a 20 facce*)
```



```
- Graphics3D -
```

■ 6.3 Grafici Animati

Mathematica puo' produrre, oltre alle immagini, anche grafici animati. L'idea per i grafici animati e' quella di generare una lista di grafici e poi visualizzarli in successione rapida. Possiamo costruire la lista di grafici con il comando **Table** e poi animarli cliccando due volte su uno dei grafici prodotti

```
grafico[i_] := Plot[i x, {x, 0, 1},
  PlotRange → {0, 5}, PlotStyle → RGBColor[1 / i, 1 / i^2, 1 - 1 / i]]
Table[grafico[i], {i, 1, 20}];
```

Analogamente per i grafici in tre dimensioni.

Nel primo esempio si vede un piano passante per l'origine che ruota

```
grafico2[i_] := Plot3D[i x + i y, {x, -1, 1}, {y, -1, 1}, PlotRange → {-50, 50}]
Table[grafico2[i], {i, 0, 15}]
```

In questo esempio si vede un paraboloide ellittico che si deforma

```
grafico3[i_, j_] :=
 Plot3D[i x^2 + j y^2, {x, -2, 2}, {y, -2, 2}, PlotRange → {0, 30}]
Table[grafico3[i, j], {i, 0, 6}, {j, 0, 6}]
```

Oppure possiamo caricare il *package* **Graphics`Animation`**

```
Needs["Graphics`Animation`"]
foto := Plot[Sin[k x], {x, 0, 8 Pi}, Axes → False, DisplayFunction → Identity]
film = {};
For[k = 1, k < 2, k = k + 0.025, AppendTo[film, foto]]
```

L'intera sequenza di immagini si visualizza con il comando **ShowAnimation**. Per generare un'animazione si selezionano i fotogrammi e si attiva, nel menu' **Cell**, l'opzione **Animate Selected Graph** oppure si clicca due volte su un grafico.

```
ShowAnimation[film]
```

Rimandiamo al manuale per conoscere i dettagli del *package* **Graphics`Animation`.**