

L'ARTE DI DISEGNARE GRAFICI CON L^AT_EX

LORENZO PANTIERI *g* TOMMASO GORDINI

25 giugno 2012

INDICE

| | | |
|-----|---------------------------------------|----|
| 1 | Introduzione | 1 |
| 1.1 | Grafici e tipografia | 1 |
| 1.2 | Il pacchetto pgfplots | 2 |
| 2 | Funzioni espresse analiticamente | 8 |
| 2.1 | Funzioni reali d'una variabile reale | 8 |
| 2.2 | Curve in forma parametrica | 13 |
| 2.3 | Funzioni reali di due variabili reali | 15 |
| 2.4 | Superfici in forma parametrica | 18 |
| 3 | Curve e superfici date per coordinate | 19 |
| 4 | Curve e superfici campionate da file | 20 |
| 5 | Altri sistemi di riferimento | 22 |
| 6 | Diagrammi a barre | 24 |
| 6.1 | Ortogrammi | 24 |
| 6.2 | Istogrammi | 25 |
| | Riferimenti bibliografici | 25 |

Quest'articolo, basato su [Feuersänger, 2011] e [De Marco e Giacomelli, 2011], cui si rimanda per ogni approfondimento, presenta pgfplots, un pacchetto per la rappresentazione grafica di dati derivato da PGF/TikZ. Si mostrerà come impostare un sistema di riferimento, come dare le istruzioni per creare grafici di funzione e d'altro tipo, come realizzare diagrammi a barre e come personalizzare l'aspetto dei vari elementi del disegno.

1 INTRODUZIONE

1.1 Grafici e tipografia

La rappresentazione di dati numerici tramite grafici di vario tipo è una parte consistente e indispensabile della comunicazione tecnico-scientifica, perché permette di esporre concetti matematici e fenomeni fisici in modo molto più semplice e intuitivo di quanto farebbero formule da sole o elenchi di numeri o tabelle.

Come fare per inserire un grafico in un documento L^AT_EX? Una via potrebbe essere quella di includerlo come file esterno prodotto con un programma

specializzato: Mathematica, MATLAB e Octave, per citarne alcuni, generano grafici molto sofisticati e li possono esportare nel formato PDF accettato da \LaTeX .

Gli inconvenienti, però, non tardano a presentarsi, perché un disegno importato, in generale:

- usa font diversi da quelli del proprio documento;
- contiene simboli matematici ed elementi grafici che male si adattano allo stile scelto;
- peggio: le formule matematiche, se presenti, appaiono completamente diverse;
- presenta linee o troppo grosse o troppo sottili.

1.2 Il pacchetto pgfplots

Il pacchetto `pgfplots`, compreso in tutte le distribuzioni complete di \LaTeX , permette di comporre grafici coerenti con le impostazioni tipografiche del documento in lavorazione scrivendone le istruzioni *direttamente* nel testo sorgente e assicura i più alti vertici qualitativi tipici di \LaTeX .

Con `pgfplots` si possono tracciare curve e superfici di qualunque tipo, in due e tre dimensioni, creare diagrammi a barre e altri grafici particolari, aggiungervi etichette, legende, titoli e personalizzare completamente ogni elemento del disegno. Inoltre, `pgfplots` può eseguire i calcoli necessari sfruttando le stesse capacità di \LaTeX senza appoggiarsi a strumenti esterni.

Il pacchetto carica automaticamente `PGF/TikZ` (permettendo eventualmente di usarne i comandi) e `xcolor`, *dopo* il quale (se già presente nel preambolo) va sempre caricato. Eventuali opzioni valide per tutti i grafici del documento si possono mettere nell'argomento di `\pgfplotsset` *nel preambolo*:

```
\usepackage{pgfplots}
\pgfplotsset{/pgf/number_format/use_comma,compat=newest, <altre opzioni>}
```

Si raccomanda di scrivere *sempre* le opzioni *rispettando gli spazi* (qui evidenziati con `_`), se presenti, e in particolare le due nel codice precedente:

- la prima imposta la virgola come separatore decimale (le istruzioni nel testo sorgente, invece, richiedono il punto);
- la seconda assicura che si usino le caratteristiche della versione più recente del pacchetto.

Si noti, infine, che i grafici prodotti da `pgfplots` sono oggetti *in testo*, cioè nel documento composto appariranno *proprio lì* dove li si è definiti nel testo sorgente, con tutti i possibili inconvenienti del caso (per risolverli si veda [Pantieri e Gordini, 2012]). Nulla vieta però di renderli *mobili*, semplicemente inserendone il codice in un ambiente `figure` come si mostra di seguito: sarà \LaTeX a metterli nel punto più opportuno.

```
\begin{figure}
\centering
\begin{tikzpicture}
<...>
\end{tikzpicture}
\caption{<...>}
\label{fig:<...>}
\end{figure}
```

Tabella 1: Sistemi di riferimento disponibili in pgfplots e librerie richieste.

| Sistema di riferimento | Ambiente | Libreria richiesta |
|------------------------|--------------|--------------------|
| Cartesiano | axis | |
| Cartesiano logaritmico | loglogaxis | |
| Ascissa logaritmica | semilogxaxis | |
| Ordinata logaritmica | semilogyaxis | |
| Coordinate polari | polaraxis | polar |
| Diagramma ternario | ternaryaxis | ternary |
| Carta di Smith | smithchart | smithchart |

1.2.1 Impostare il sistema di riferimento: l'ambiente axis

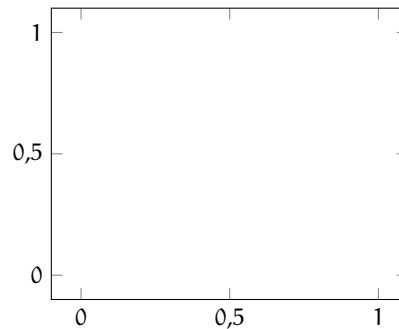
L'ambiente fondamentale di pgfplots è `axis`, da inserire a propria volta nell'ambiente `tikzpicture` (definito da PGF/TikZ) con la sintassi:

```
\begin{tikzpicture}
\begin{axis}[opzioni]
istruzioni di pgfplots o PGF/TikZ
\end{axis}
\end{tikzpicture}
```

Si noti che le *opzioni*, se presenti, agiranno su *tutti* i grafici inseriti in quell'ambiente `axis` (e solo su quelli).

Eccone un esempio davvero minimo:

```
\begin{tikzpicture}
\begin{axis}
\end{axis}
\end{tikzpicture}
```

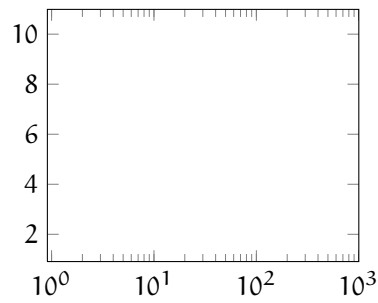


Si noti che:

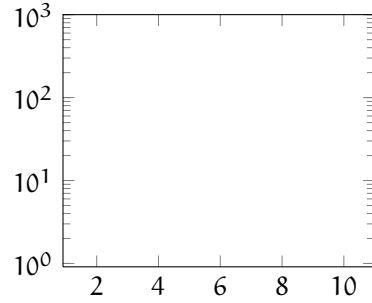
- `axis` definisce un sistema di riferimento cartesiano ortogonale visualizzandone parte del primo quadrante (e una piccola porzione degli altri tre) come un riquadro;
- sui lati del riquadro compaiono tacche di marcatura distanti tra loro un certo passo, che il pacchetto calcola automaticamente sulla base di parametri interni;
- le esigenze tipografiche di questo documento hanno imposto di assegnare una larghezza fissa ai disegni: riproducendoli, si potrebbero avere risultati diversi (ma non scorretti).

Gli altri ambienti elencati nella tabella 1 producono i sistemi di riferimento mostrati nella figura 1 nella pagina successiva e nel paragrafo 5 a pagina 22, nel quale si spiega anche come caricare le librerie richieste.

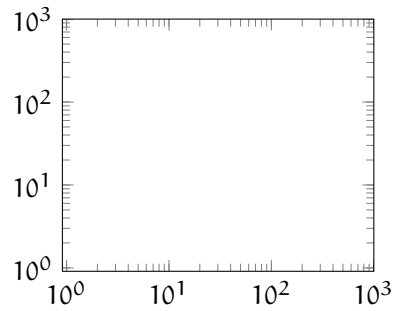
Si può personalizzare il risultato predefinito passando ad `axis` opportune opzioni separate dalla virgola, se più d'una, nella notazione *<chiave>=<valore>* o anche solo *<chiave>* (i valori `=true` infatti si possono omettere).



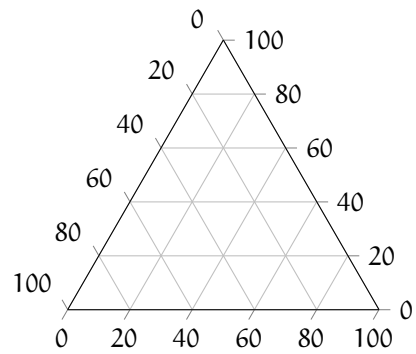
(a) Piano con ascissa logaritmica.



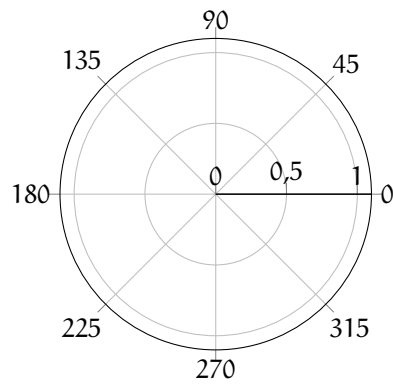
(b) Piano con ordinata logaritmica.



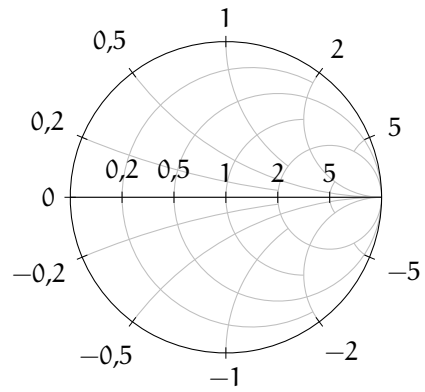
(c) Piano logaritmico.



(d) Diagramma ternario.



(e) Sistema di coordinate polari.

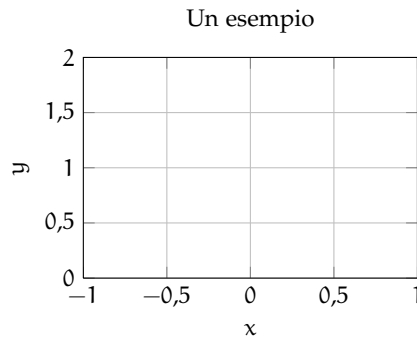


(f) Carta di Smith.

Figura 1: Sistemi di riferimento predefiniti di pgfplots (tranne quello prodotto da axis).

Il prossimo è un esempio con qualche opzione:

```
\begin{tikzpicture}
\begin{axis} [xmin=-1,xmax=1,
ymin=0,ymax=2,grid=major,
xlabel=$x$,ylabel=$y$,
title={Un esempio},
width=6cm,height=4.5cm]
\end{axis}
\end{tikzpicture}
```



Si noti quanto segue.

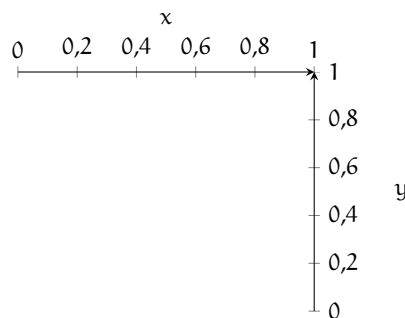
- `xmin` e `xmax` fissano rispettivamente il valore minimo e massimo delle ascisse; analoghe chiavi si useranno per le ordinate.
- `grid=major` visualizza una griglia agganciata alle tacche di marcatura degli assi per leggere più facilmente il grafico.
- `xlabel` e `ylabel` producono le etichette degli assi (non obbligatorie), qui x e y . Quest'ultima per impostazione predefinita è ruotata di 90° in senso antiorario e centrata verticalmente: per averla diritta, basta scrivere tra le opzioni di `axis`, come mostra l'esempio seguente,

```
{nome dell'etichetta}label style={rotate=-90}
```

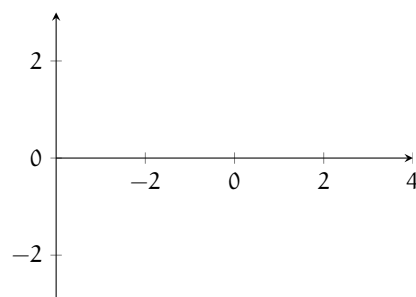
- `title` produce un titolo sopra il disegno (ma se il grafico è mobile, lo si metta nell'argomento di `\caption`).
- `width` e `height`, esprimibili in una qualsiasi delle unità di misura accettate da \LaTeX , impostano rispettivamente larghezza e altezza dell'intero disegno (etichette e titoli compresi).

Di seguito si mostrano alcuni esempi in cui si è modificata la posizione degli assi. Si noti che dichiararli esplicitamente attiva lo stile tradizionale ed elimina il riquadro.

```
\begin{tikzpicture}
\begin{axis}
[axis x line=top,
axis y line=right,
xlabel=$x$,ylabel=$y$,
ylabel style={rotate=-90}]
\end{axis}
\end{tikzpicture}
```



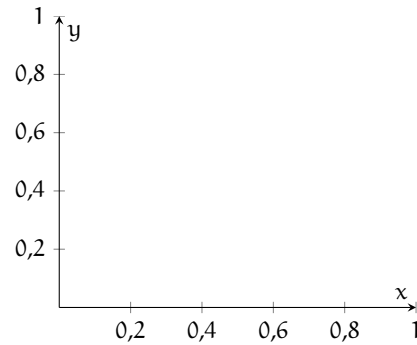
```
\begin{tikzpicture}
\begin{axis}
[xmin=-4,xmax=4,ymin=-3,ymax=3,
axis x line=middle,
axis y line=left]
\end{axis}
\end{tikzpicture}
```



```

\begin{tikzpicture}
\begin{axis}
[axis lines=middle,
xlabel=$x$,ylabel=$y$]
\end{axis}
\end{tikzpicture}

```



Si noti quanto segue.

- `axis x line` regola la posizione delle ascisse: accetta i valori `bottom`, `middle` e `top`, che ne impongono il passaggio per $y = y_{\min}$, $y = 0$ e $y = y_{\max}$, rispettivamente. (Quando però il grafico è tutto al di sotto o al di sopra dell'asse x , il valore `middle` corrisponde a `top` o `bottom` rispettivamente.)
- `axis y line` regola la posizione delle ordinate: accetta i valori `left`, `middle` e `right`, che ne impongono il passaggio per $x = x_{\min}$, $x = 0$ e $x = x_{\max}$, rispettivamente. (Quando però il grafico è tutto a sinistra o a destra dell'asse y , il valore `middle` corrisponde a `right` o `left` rispettivamente.)
- `axis lines` imposta contemporaneamente entrambi gli assi con il valore scelto, di solito `middle` (se non li si desidera visualizzare, il valore da dare è `none`) e raddrizza automaticamente l'etichetta delle ordinate.
- La documentazione del pacchetto, consultabile con `texdoc pgfplots`, spiega come personalizzare la posizione delle etichette.

1.2.2 Disegnare il grafico: i comandi `\addplot` e `\addplot3`

Per disegnare un grafico nel piano, dentro l'ambiente corrispondente al tipo di grafico da realizzare va dato il comando `\addplot`, ripetibile per ogni grafico da aggiungere nel sistema di riferimento. La sintassi generale del comando è la seguente:

```

\addplot [opzioni]
<superistruzione>
{<istruzioni>};

```

Si noti che:

- `<opzioni>`, nella notazione `<chiave>=<valore>` o solo `<chiave>`, sono le opzioni, eventualmente separate con la virgola se più d'una, che definiscono l'aspetto del grafico;
- la `<superistruzione>` (`coordinates`, `file` o `table`) va indicata nei casi spiegati tra poco;
- `<istruzioni>` sono le istruzioni per ottenere il grafico vero e proprio;
- il comando *esige* il punto e virgola immediatamente dopo la graffa di chiusura del proprio argomento;
- tra i vari elementi si può lasciare o meno uno spazio bianco per rendere il codice più leggibile.

Tabella 2: Alcune funzioni e costanti matematiche predefinite da pgfplots.

| Funzione | Significato | Funzione | Significato |
|--------------------|--------------------|-------------------|---------------------|
| <code>sqrt</code> | radice quadrata | <code>cot</code> | cotangente |
| <code>exp</code> | esponenziale | <code>asin</code> | arcoseno |
| <code>ln</code> | logaritmo naturale | <code>acos</code> | arcocoseno |
| <code>log10</code> | logaritmo decimale | <code>atan</code> | arcotangente |
| <code>sin</code> | seno | <code>sinh</code> | seno iperbolico |
| <code>cos</code> | coseno | <code>cosh</code> | coseno iperbolico |
| <code>tan</code> | tangente | <code>tanh</code> | tangente iperbolica |
| <code>sec</code> | secante | <code>e</code> | numero di Nepero |
| <code>cosec</code> | cosecante | <code>pi</code> | π |

Le *istruzioni* da dare a `\addplot` possono essere di tre tipi, descritti di seguito.

- Un'espressione matematica, che verrà valutata per un opportuno numero di punti del dominio di definizione, con il codice

```
\begin{axis}[...]
\addplot [...]
{ <espressione matematica> };
\end{axis}
```

- Una sequenza di coppie di valori (corrispondenti a coordinate di punti del piano), con il codice

```
\begin{axis}[...]
\addplot [...] coordinates
{(x1, y1) (x2, y2)
... (xn, yn)};
\end{axis}
```

dove `coordinates` ordina a pgfplots di disegnare il grafico usando le coordinate scritte immediatamente dopo.








- Una sequenza di coppie di valori separati da almeno uno spazio e disposte su più righe, contenuta in un file esterno prodotto con uno dei programmi nominati nel paragrafo 1.1 a pagina 1 e sistemato nella cartella di lavoro. Il codice generale è il seguente:

```
\begin{axis}[...]
\addplot [...]
file { <nome del file di dati con l'estensione> };
\end{axis}
```

dove `file` ordina a pgfplots di usare il file indicato (registrato con una delle estensioni accettate dal pacchetto). In alternativa si può usare `table`, una variante di `file` personalizzabile (si veda la documentazione del pacchetto).

Con la stessa sintassi, il comando `\addplot3` permette di disegnare un grafico nello spazio. Nei casi in cui l'istruzione non sia un'espressione matematica, le coordinate vanno espresse come terne anziché coppie di valori.

Tabella 3: Spessori di linea disponibili in pgfplots.

| Chiave | Risultato | Chiave | Risultato |
|------------|---|-------------|---|
| ultra thin |  | thick |  |
| very thin |  | very thick |  |
| thin |  | ultra thick |  |
| semithick |  | | |

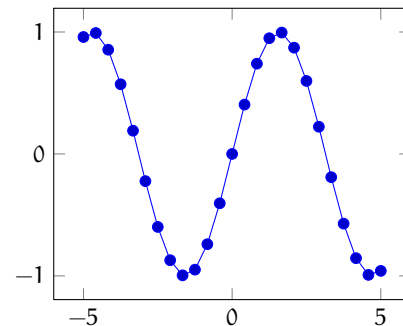
2 FUNZIONI ESPRESSE ANALITICAMENTE

La tabella 2 nella pagina precedente mostra le più importanti funzioni e costanti matematiche predefinite da pgfplots. Di seguito se ne presentano alcune realizzazioni.

2.1 Funzioni reali d'una variabile reale

Un semplice esempio senza opzioni per cominciare.

```
\begin{tikzpicture}
\begin{axis}
\addplot {sin(deg(x))};
\end{axis}
\end{tikzpicture}
```

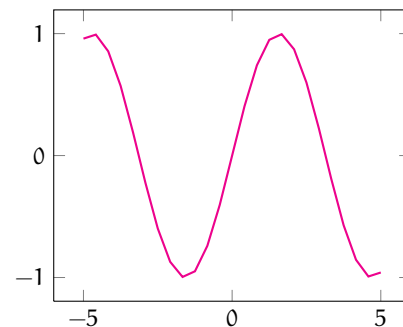


Si noti che:

- il risultato predefinito è una curva blu contrassegnata da marcatori circolari (per avere solo i marcatori, questa volta in nero, si passi a `\addplot` la chiave `only marks`);
- `deg` trasforma in gradi il proprio argomento, da mettere fra parentesi tonde (pgfplots, infatti, assume che l'argomento delle funzioni trigonometriche sia espresso in quest'unità di misura e non in radianti).

Ora un esempio con qualche personalizzazione:

```
\begin{tikzpicture}
\begin{axis}
\addplot [thick,color=magenta]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}
```



Si noti che:

- la curva non presenta marcatori perché le opzioni di `\addplot` sostituiscono le impostazioni predefinite (per mantenerle aggiungendovi nuove opzioni o per ridefinire localmente un'opzione, il comando è `\addplot+`; in questo caso, per eliminare del tutto i marcatori la chiave è `no marks`);

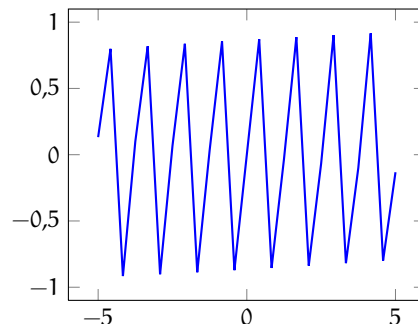
Tabella 4: Colori disponibili in pgfplots.

| Chiave | Risultato | Chiave | Risultato | Chiave | Risultato |
|---------|---|----------|---|-----------|---|
| red |  | brown |  | orange |  |
| green |  | lime |  | olive |  |
| blue |  | cyan |  | teal |  |
| magenta |  | purple |  | violet |  |
| gray |  | darkgray |  | lightgray |  |
| yellow |  | black |  | pink |  |
| white |  | | | | |

- `thick` imposta lo spessore della curva, da scegliere tra quelli mostrati nella tabella 3 nella pagina precedente;
- `color=magenta` (ma basta scrivere `magenta`) imposta il colore desiderato per la curva, da scegliere tra quelli elencati nella tabella 4 (se non bastassero, si possono usare quelli di `xcolor`, più numerosi, dopo averlo caricato con almeno un'opzione, oppure se ne definisca uno a scelta regolando a mano le componenti).

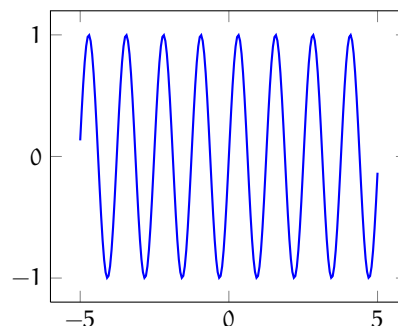
La curva dell'esempio precedente è leggermente "spigolosa". Il pacchetto `pgfplots`, infatti, disegna le curve approssimandole alla spezzata che unisce un opportuno campione di punti. Se questi sono sufficientemente vicini, si percepisce la spezzata come una curva "regolare", tracciata con precisione e qualità; ma se sono insufficienti, il risultato è inaccettabile, come nell'esempio seguente:

```
\begin{tikzpicture}
\begin{axis}
\addplot [thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}
```



Aumentando il valore della chiave `samples`, che gestisce il numero di punti campionati (il suo valore predefinito è 25), si risolve il problema:

```
\begin{tikzpicture}
\begin{axis}
\addplot
[samples=200,thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}
```

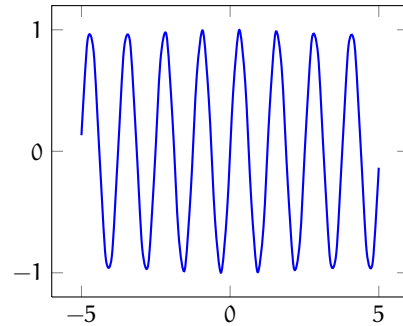


Si può anche usare la chiave `smooth`, da sola o insieme alla precedente: essa impone al programma di tracciare il grafico con curve di Bézier (cubiche raccordate con continuità di tangente e concavità) anziché con segmenti.

Tabella 5: Trattati disponibili in pgfplots.

| Chiave | Risultato | Chiave | Risultato |
|----------------|-----------|-----------------------|-----------------|
| solid | ————— | dashdotted | -. - . - . - . |
| dotted | | densely dashdotted | - . - . - . - . |
| densely dotted | | loosely dashdotted | - . - . - . - . |
| loosely dotted | | dashdotdotted | - . - . - . - . |
| dashed | - - - - - | densely dashdotdotted | - . - . - . - . |
| densely dashed | - - - - - | loosely dashdotdotted | - . - . - . - . |
| loosely dashed | - - - - - | | |

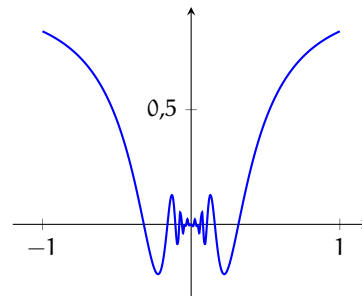
```
\begin{tikzpicture}
\begin{axis}
\addplot
[samples=50,smooth,thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}
```



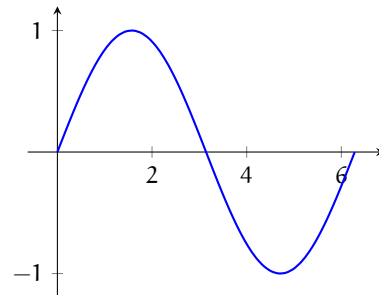
Permettendo (spesso, ma non sempre) di diminuire il valore di `samples`, `smooth` può ridurre o evitare del tutto le probabilità di eccedere la memoria di calcolo del programma (che reagisce arrestando la composizione), ciò che troppi grafici ad altissima risoluzione potrebbero causare: la documentazione del pacchetto spiega come risolvere questi problemi.

Altri due esempi:

```
\begin{tikzpicture}
\begin{axis}
[axis lines=middle,
enlargelimits]
\addplot
[domain=-1:1,samples=200,smooth,
thick,blue]
{x*sin(deg(1/x))};
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}[axis lines=middle,
enlargelimits]
\addplot
[domain=0:2*pi,samples=40,smooth,
thick,blue]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}
```



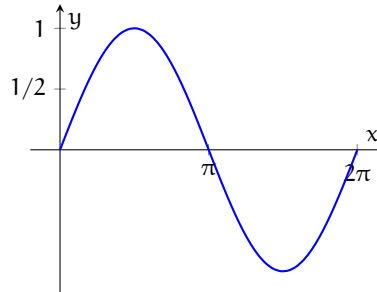
Si noti quanto segue.

- La chiave `enlargelimits`, da mettere *dopo* `axis lines`, aumenta di poco la lunghezza di *entrambi* gli assi oltre i limiti stabiliti da `xmax` e `ymax` (`enlarge x limits` e `enlarge y limits` permettono di farlo separatamente), evitando che la curva oltrepassi i limiti del sistema di riferimento o finisca proprio sulla freccia di uno degli assi, oppure ancora solo per far “respirare” il grafico.

- La chiave `domain`, per impostazione predefinita pari a $[-5, 5]$, imposta il dominio, nel primo esempio pari a $[-1, 1]$ e nel secondo a $[0, 2\pi]$.

Ora si ripropone l'esempio precedente con qualche variante:

```
\begin{tikzpicture}
\begin{axis} [axis lines=middle,
enlargelimits,
xtick={3.14,6.28},ytick={0.5,1},
xticklabels={\pi$, $2\pi$},
yticklabels={$1/2$, $1$},
xlabel=$x$,ylabel=$y$]
\addplot [domain=0:2*pi,
samples=40,smooth,thick,blue]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}
```

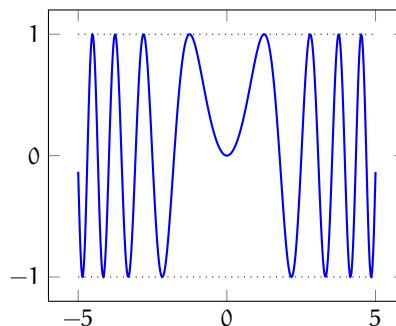


Si noti che:

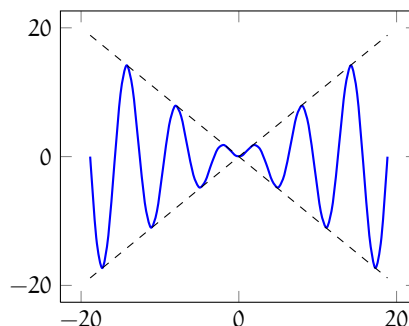
- con `xtick` e `ytick` si sceglie il valore delle tacche sugli assi x e y ;
- `xticklabels` e `yticklabels` producono le etichette corrispondenti quando quelle automatiche non risultano soddisfacenti.

Ecco un paio d'esempi con più `\addplot` consecutivi:

```
\begin{tikzpicture}
\begin{axis}
\addplot [samples=200,thick,
smooth,blue] {sin(deg(x^2))};
\addplot [dotted] {1};
\addplot [dotted] {-1};
\end{axis}
\end{tikzpicture}
```



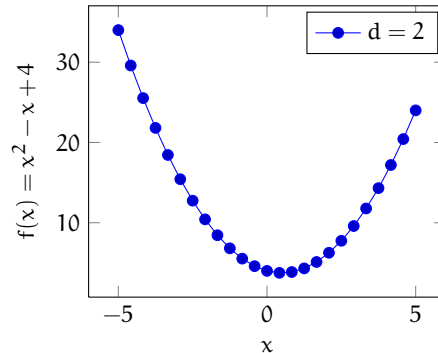
```
\begin{tikzpicture}
\begin{axis} [domain=-6*pi:6*pi]
\addplot [samples=50,smooth,
thick,blue] {x*sin(deg(x))};
\addplot [dashed] {x};
\addplot [dashed] {-x};
\end{axis}
\end{tikzpicture}
```



dove `dotted` disegna una linea punteggiata e `dashed` una linea tratteggiata (i tratti disponibili sono mostrati nella tabella 5 nella pagina precedente).

Il comando `\legend` produce una legenda:

```
\begin{tikzpicture}
\begin{axis}
[xlabel=$x$,
ylabel={$f(x)=x^2-x+4$}]
\addplot {x^2-x+4};
\legend{$d=2$}
\end{axis}
\end{tikzpicture}
```

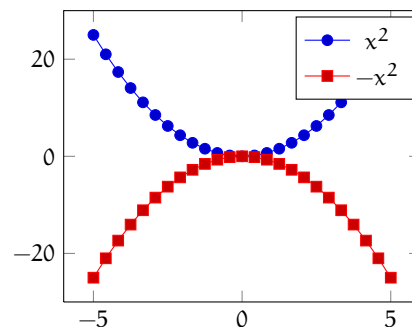


Si noti che:

- aspetto (riquadrata) e posizione (in alto a destra) della legenda nel disegno sono predefiniti ma personalizzabili;
- se contiene caratteri particolari come '=' o la virgola, l'etichetta di un asse va racchiusa tra parentesi graffe;
- se è una formula matematica lunga, l'etichetta dell'asse y può rimanere nella posizione predefinita (in alternativa, si metta la formula nell'argomento di title).

Una legenda può contenere una voce per ciascun grafico tracciato:

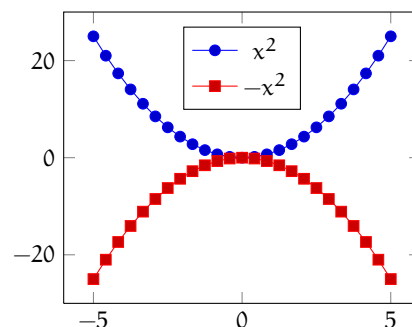
```
\begin{tikzpicture}
\begin{axis}
\addplot {x^2};
\addplot {-x^2};
\legend{$x^2$, $-x^2$}
\end{axis}
\end{tikzpicture}
```



Si noti che in caso di più grafici in uno stesso sistema di riferimento, pgfplots assegna automaticamente a ogni curva colore e marcatore distinti, scegliendoli secondo un ordine interno.

Come si vede, però, per impostazione predefinita la legenda compare sempre *dentro* il sistema di riferimento (se cartesiano) e può sovrapporsi al grafico nascondendone una parte. Si risolve il problema spostandola (la si può mettere in qualunque posizione dentro e fuori il sistema di riferimento), come si mostra nell'esempio seguente:

```
\begin{tikzpicture}
\begin{axis}
[legend style={anchor=north,
at={(0.5,0.95)}}]
\addplot {x^2};
\addplot {-x^2};
\legend{$x^2$, $-x^2$}
\end{axis}
\end{tikzpicture}
```



Qualche prova e la lettura della documentazione del pacchetto permetteranno d'ottenere il risultato desiderato. Si noti che:

- `legend style` modifica lo stile predefinito della legenda;
- `anchor` specifica uno dei punti d'ancoraggio predefiniti per il riquadro della legenda (coincidenti essenzialmente con le direzioni d'una rosa dei venti a otto punte più un punto per il centro): qui `north` indica il punto medio del lato superiore;
- `at` definisce le coordinate del punto d'ancoraggio.

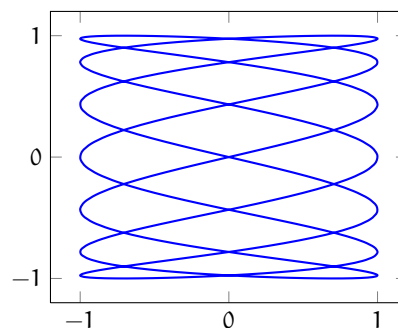
2.2 Curve in forma parametrica

2.2.1 Curve nel piano

Di seguito si mostrano alcuni esempi di curve nel piano.

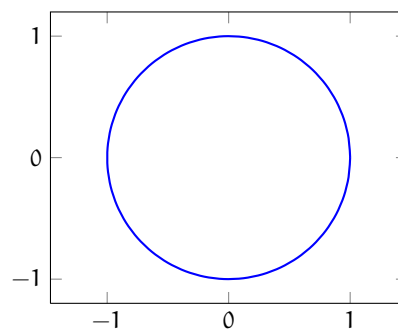
```
\begin{tikzpicture}
\begin{axis}
[title={Figura di Lissajous}]
\addplot
[domain=0:360,variable=\t,
samples=200,smooth,thick,blue]
({sin(7*t)},{sin(2*t)});
\end{axis}
\end{tikzpicture}
```

Figura di Lissajous



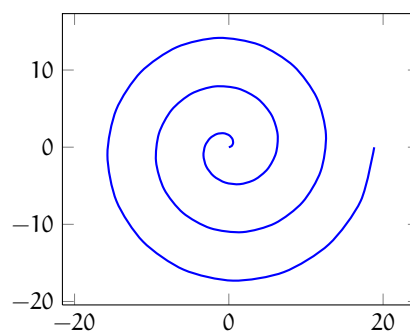
```
\begin{tikzpicture}
\begin{axis} [axis equal,
title={Circonferenza}]
\addplot
[domain=0:360,variable=\t,
samples=40,smooth,thick,blue]
({cos(t)},{sin(t)});
\end{axis}
\end{tikzpicture}
```

Circonferenza



```
\begin{tikzpicture}
\begin{axis} [axis equal,
title={Spirale di Archimede}]
\addplot
[domain=0:6*pi,variable=\t,
samples=50,smooth,thick,blue]
({t*cos(deg(t))},
{t*sin(deg(t))});
\end{axis}
\end{tikzpicture}
```

Spirale di Archimede



Si noti che:

- `axis equal` imposta la stessa unità di misura su entrambi gli assi;
- le istruzioni per le curve parametriche nel piano vanno date nella forma

$$(\{x\}, \{y\});$$

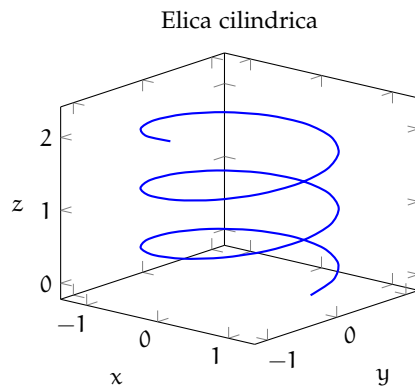
dove x e y sono funzioni del parametro;

- `variable` imposta il parametro (t , nei casi considerati), che nella propria definizione va preceduto da una barra rovescia.

2.2.2 Curve nello spazio

Ecco un esempio di curva nello spazio tridimensionale:

```
\begin{tikzpicture}
\begin{axis}
[view={40}{20},axis equal,
xlabel=$x$,ylabel=$y$,
zlabel=$z$,
zlabel style={rotate=-90},
title={Elica cilindrica}]
\addplot3
[domain=0:5.5*pi,variable=\t,
samples=40,samples y=0,
smooth,thick,blue]
({cos(deg(t))},{sin(deg(t))},
{2*t/(5*pi)});
\end{axis}
\end{tikzpicture}
```



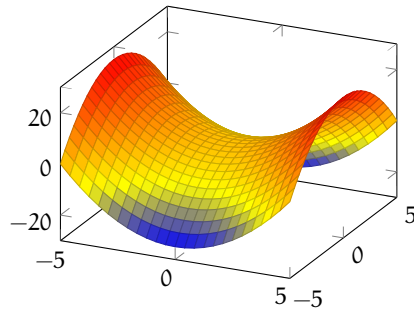
Si noti quanto segue.

- Il comando per tracciare grafici nello spazio è `\addplot3`, che richiede la sintassi spiegata nel paragrafo 1.2.2 a pagina 6.
- La chiave `view` imposta il punto di vista dell'osservatore (per una sua descrizione più completa si veda il paragrafo successivo).
- La chiave `zlabel` produce l'etichetta dell'asse z che, per una migliore leggibilità, è stata ruotata con l'opzione descritta nel paragrafo 1.2.1 a pagina 3 (in un grafico a tre dimensioni, in genere l'etichetta dell'asse y è dritta).
- L'opzione `samples y=0` dichiara che si tratta di una curva e non di una superficie.
- Le istruzioni per le curve parametriche nello spazio seguono la stessa sintassi già vista per quelle nel piano.

2.3 Funzioni reali di due variabili reali

Un paraboloide iperbolico per cominciare:

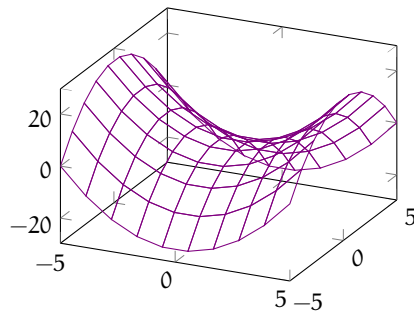
```
\begin{tikzpicture}
\begin{axis}
\addplot3 [surf]
{x^2-y^2};
\end{axis}
\end{tikzpicture}
```



Si noti che surf disegna una superficie, visualizzata con colori predefiniti.

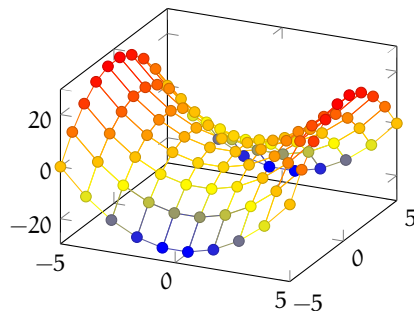
La chiave mesh produce grafici “a rete”:

```
\begin{tikzpicture}
\begin{axis}
\addplot3
[mesh,samples=10,violet]
{x^2-y^2};
\end{axis}
\end{tikzpicture}
```



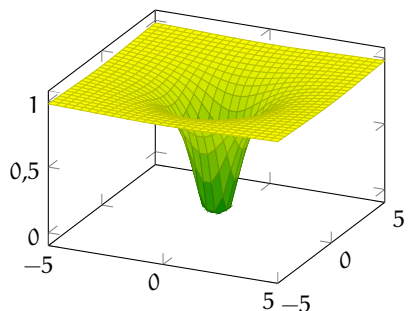
Aggiungendole scatter i nodi sono evidenziati con marcatori:

```
\begin{tikzpicture}
\begin{axis}
\addplot3
[mesh,scatter,samples=10]
{x^2-y^2};
\end{axis}
\end{tikzpicture}
```



Ecco un altro esempio:

```
\begin{tikzpicture}
\begin{axis}
\addplot3 [samples=30,surf,
colormap/greenyellow]
{exp(-1/(x^2+y^2))};
\end{axis}
\end{tikzpicture}
```



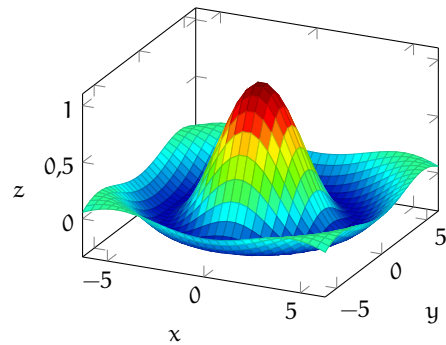
Si noti che colormap/⟨colorazione⟩ personalizza i colori predefiniti (si veda la documentazione del pacchetto per le possibilità disponibili, alcune delle quali verranno mostrate nei prossimi esempi).

Ora due varianti d’uno stesso grafico ottenute modificando i valori di domain, che imposta il dominio della funzione. Nel primo:

```

\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [domain=-2*pi:2*pi,
samples=30,surf,
colormap/bluered]
{sin(deg(sqrt(x^2+y^2)))/%
sqrt(x^2+y^2)};
\end{axis}
\end{tikzpicture}

```



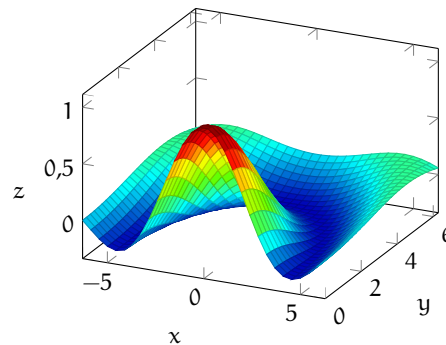
il dominio della funzione è $[-2\pi, 2\pi] \times [-2\pi, 2\pi]$: specificando solo `domain`, infatti, con `domain=A` il dominio della funzione è il quadrato $A \times A$.

Nel secondo:

```

\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [domain=-2*pi:2*pi,
y domain=0:2*pi,samples=30,
surf,colormap/bluered]
{sin(deg(sqrt(x^2+y^2)))/%
sqrt(x^2+y^2)};
\end{axis}
\end{tikzpicture}

```



il dominio della funzione è $[-2\pi, 2\pi] \times [0, 2\pi]$: specificando anche `y domain`, infatti, con `domain=A` e `y domain=B` il dominio della funzione è il rettangolo $A \times B$.

Il punto di vista: la chiave view

La chiave `view` imposta il punto di vista dell'osservatore e richiede la seguente sintassi generale:

```
view={\azimut}{\elevazione}
```

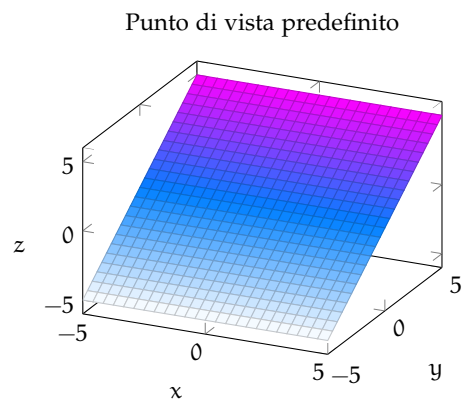
dove i due valori (pari a 25 e 30 per impostazione predefinita) indicano rispettivamente azimuth ed elevazione nel sistema di coordinate sferiche. La figura 2 a fronte mostra come pgfplots gestisce il punto di vista.

Seguono ora alcune varianti notevoli d'uno stesso piano ottenute modificando i valori di `view`.

```

\begin{tikzpicture}
\begin{axis} [view={25}{30},
title={Punto di vista
predefinito},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}

```



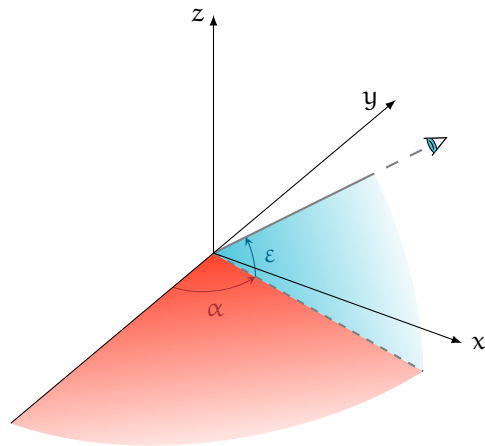
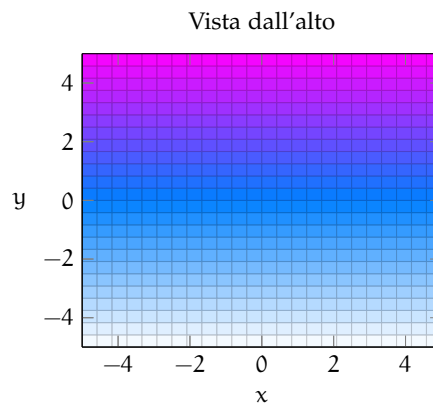
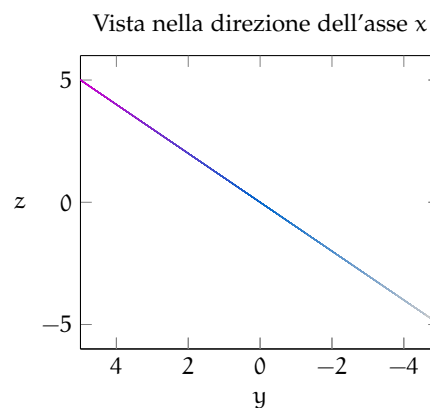


Figura 2: Definizione di azimuth (α) ed elevazione (ε) secondo la convenzione adottata da pgfplots. Nel caso mostrato: $\alpha = 70^\circ$ e $\varepsilon = 35^\circ$.

```
\begin{tikzpicture}
\begin{axis} [view={0}{90},title=
{Vista dall'alto},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
ylabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}
```



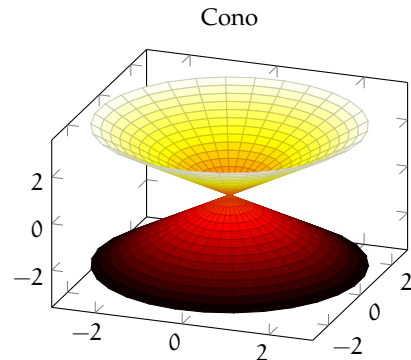
```
\begin{tikzpicture}
\begin{axis} [view={270}{0},
title={Vista nella direzione
dell'asse $x$},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}
```



2.4 Superfici in forma parametrica

Un esempio:

```
\begin{tikzpicture}
\begin{axis}
[view={20}{30},title={Cono}]
\addplot3
[domain=-3:3,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/hot2]
({u*cos(v)}, {u*sin(v)}, u);
\end{axis}
\end{tikzpicture}
```

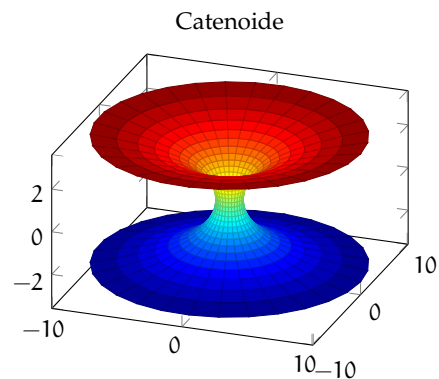


Si noti che:

- `variable y` imposta il secondo parametro coordinato (v , in questo caso);
- la chiave `z buffer=sort` suggerisce a `pgfplots` i criteri da seguire per proiettare i punti dello spazio tridimensionale sul quadro di proiezione (in questo caso, `sort` traccia per primi i segmenti più distanti dal punto di osservazione).

Di seguito si mostra una galleria d'esempi.

```
\begin{tikzpicture}
\begin{axis}
[view={20}{35},title={Catenoide}]
\addplot3
[domain=0:360,y domain=-3:3,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/jet]
({cos(u)*cosh(v)},
{sin(u)*cosh(v)},v);
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}
[view={20}{35},title={Elicoide}]
\addplot3
[domain=-3:3,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/redyellow]
({u*cos(v)}, {u*sin(v)}, v);
\end{axis}
\end{tikzpicture}
```

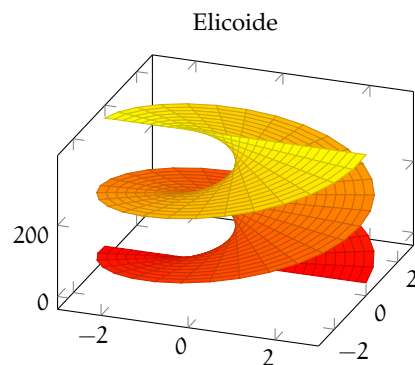
















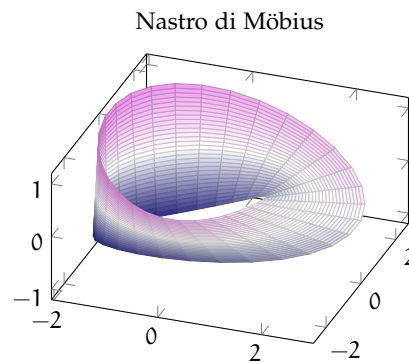


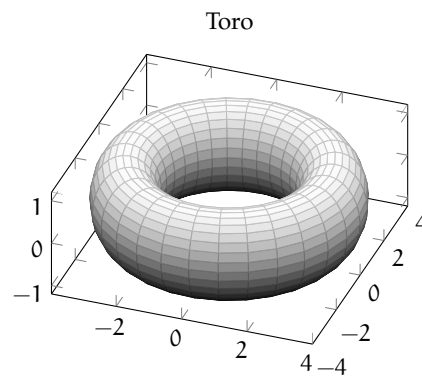
Tabella 6: Alcuni marcatori disponibili in pgfplots.

| Chiave | Risultato | Chiave | Risultato |
|----------|---|--------------|---|
| * |  | square |  |
| o |  | square* |  |
| x |  | halfsquare* |  |
| + |  | triangle |  |
| asterisk |  | triangle* |  |
| star |  | diamond |  |
| oplus |  | diamond* |  |
| otimes |  | halfdiamond* |  |

```
\begin{tikzpicture}
\begin{axis}
[view={20}{45},
title={Nastro di M\obius}]
\addplot3
[domain=0:360,y domain=-1:1,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/violet]
({2*cos(u)+v*cos(u)*cos(u/2)},
{2*sin(u)+v*sin(u)*sin(u/2)},
{v*sin(u/2)});
\end{axis}
\end{tikzpicture}
```



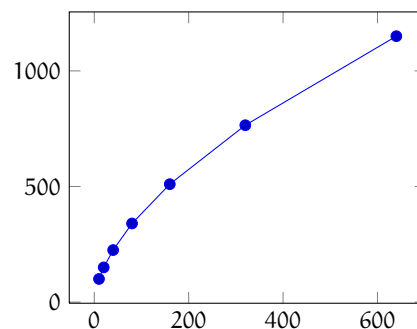
```
\begin{tikzpicture}
\begin{axis}
[view={20}{55},title={Toro}]
\addplot3
[domain=0:360,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/blackwhite]
({(3+cos(u))*cos(v)},
{(3+cos(u))*sin(v)},
{sin(u)});
\end{axis}
\end{tikzpicture}
```



3 CURVE E SUPERFICI DATE PER COORDINATE

I prossimi sono due esempi di curve date per coordinate:

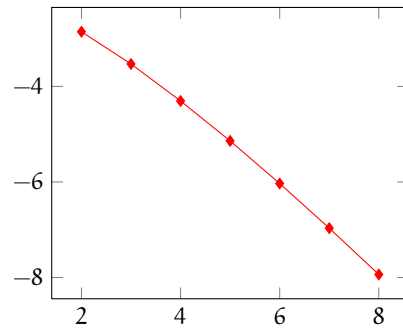
```
\begin{tikzpicture}
\begin{axis}
\addplot coordinates
{(10, 100) (20, 150)
(40, 225) (80, 340)
(160, 510) (320, 765)
(640, 1150)};
\end{axis}
\end{tikzpicture}
```



```

\begin{tikzpicture}
\begin{axis}
\addplot [red,mark=diamond*]
coordinates
{(2, -2.855) (3, -3.530)
(4, -4.305) (5, -5.141)
(6, -6.032) (7, -6.967)
(8, -7.937)};
\end{axis}
\end{tikzpicture}

```



Si noti che mark imposta il tipo di marcatore, nel secondo esempio un rombo. In alternativa, se ne può scegliere un altro tra quelli mostrati nella tabella 6 nella pagina precedente o definirne uno a propria scelta.

Infine un esempio di superficie data per coordinate:

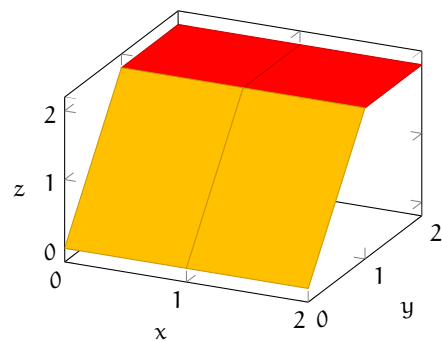
```

\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf] coordinates
{(0,0,0) (1,0,0) (2,0,0)

(0,1,2) (1,1,2) (2,1,2)

(0,2,2) (1,2,2) (2,2,2)};
\end{axis}
\end{tikzpicture}

```



Si noti che in questo caso la sequenza delle coordinate assume la struttura di una matrice, nella quale ogni riga di valori va separata da quella successiva con una riga vuota.

4 CURVE E SUPERFICI CAMPIONATE DA FILE

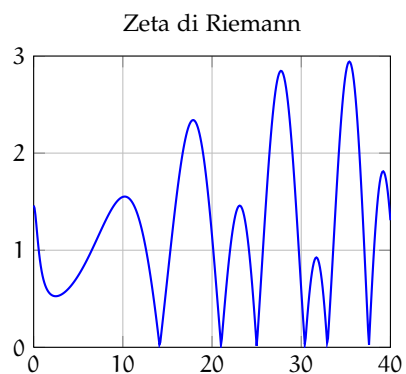
A titolo esemplificativo, si mostrano qui tre grafici (i primi due nel piano, il terzo nello spazio) costruiti con l'aiuto di file esterni ottenuti nei modi spiegati nel paragrafo 1.2.2 a pagina 6.

Il primo esempio mostra il grafico del valore assoluto della funzione Zeta di Riemann sulla retta $\text{Re } z = 1/2$.

```

\begin{tikzpicture}
\begin{axis} [xmin=0,xmax=40,
ymin=0,ymax=3,grid=major,
title={Zeta di Riemann}]
\addplot [thick,blue]
file {zeta.txt};
\end{axis}
\end{tikzpicture}

```

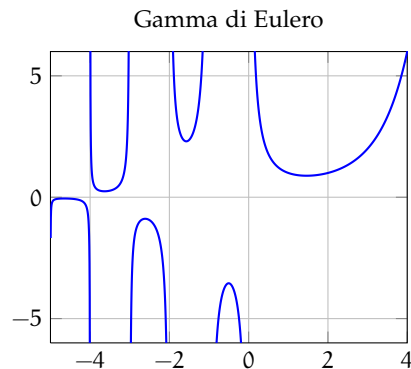


Di seguito si mostrano le prime righe del file `zeta.txt`:

```
0.0 1.460354508809587
0.1 1.433807867750897
0.2 1.362770945580488
0.3 1.266515016158303
```

Il prossimo esempio mostra il grafico della funzione Gamma di Eulero:

```
\begin{tikzpicture}
\begin{axis} [xmin=-5,xmax=4,
  ymin=-6,ymax=6,grid=major,
  title={Gamma di Eulero}]
\addplot
[unbounded coords=jump,
  thick,blue]
file {gamma.txt};
\end{axis}
\end{tikzpicture}
```



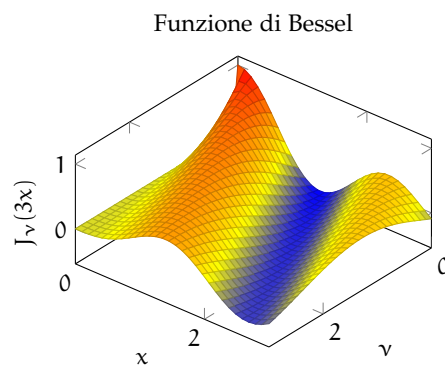
dove l'opzione `unbounded coords=jump` gestisce i punti di discontinuità. Ed ecco le prime righe del file `gamma.txt`:

```
-5.000 NaN
-4.995 -1.6810104460206580
-4.990 -0.8478047198471037
-4.985 -0.5701560523263895
```

Si noti che le righe contenenti i valori `NaN` (*Not a Number*) e `inf` (*infinity*), indicanti rispettivamente un valore non numerico e infinito, vengono sempre ignorate da `pgfplots`.

Il prossimo grafico, infine, mostra la funzione di Bessel (di prima specie):

```
\begin{tikzpicture}
\begin{axis} [view={130}{50},
  xlabel=$\nu$,ylabel=$x$,
  zlabel={$J_\nu(3x)$},
  title={Funzione di Bessel}]
\addplot3
[surf,z buffer=sort]
file {bessel.txt};
\end{axis}
\end{tikzpicture}
```



Di seguito si mostrano le prime righe del file `bessel.txt`:

```
0.000000 0.000000 1.000000
0.000000 0.103448 0.976066
0.000000 0.206897 0.905981
0.000000 0.310345 0.794755
```

Si noti che in questo caso le terne di coordinate che rappresentano i punti devono rispettare un ordine ben preciso ed essere in un formato opportuno (la documentazione del pacchetto spiega come farlo).

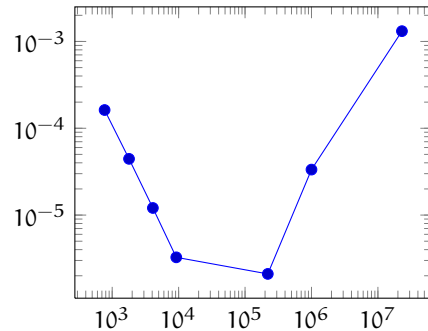
5 ALTRI SISTEMI DI RIFERIMENTO

In questa sezione si mostrano alcuni esempi degli altri sistemi di riferimento definiti da pgfplots. Alcuni di essi richiedono di caricare la libreria indicata *nel preambolo* con:

```
\usepgfplotslibrary{<libreria>}
```

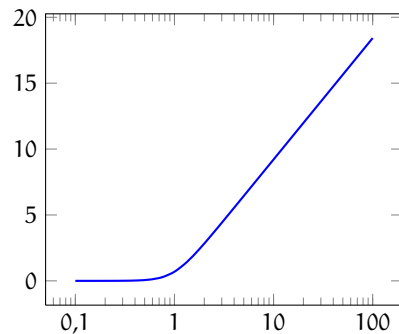
Per cominciare, un esempio di piano cartesiano logaritmico:

```
\begin{tikzpicture}
\begin{loglogaxis}
\addplot coordinates
{(769, 1.6227e-04)
(1793, 4.4425e-05)
(4097, 1.2071e-05)
(9217, 3.2610e-06)
(2.2e5, 2.1E-6)
(1e6, 0.00003341)
(2.3e7, 0.00131415)};
\end{loglogaxis}
\end{tikzpicture}
```



Ora un grafico con ascissa logaritmica:

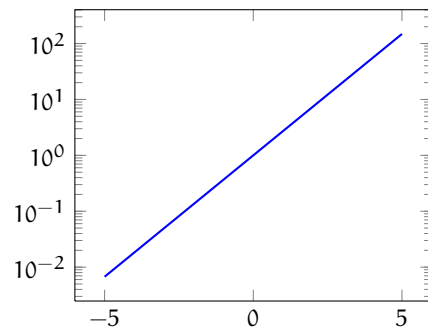
```
\begin{tikzpicture}
\begin{semilogxaxis}
[log ticks with fixed point]
\addplot
[domain=0.1:100,
thick,blue,smooth]
{ln(1+x^4)};
\end{semilogxaxis}
\end{tikzpicture}
```



L'opzione `log ticks with fixed point` evita marcatori di tacca con esponenti.

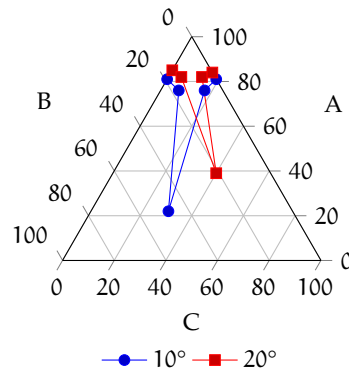
Il prossimo è un grafico con ordinata logaritmica:

```
\begin{tikzpicture}
\begin{semilogyaxis}
\addplot [thick,blue]
{exp(x)};
\end{semilogyaxis}
\end{tikzpicture}
```



Segue un esempio di diagramma ternario (richiede la libreria ternary):

```
\begin{tikzpicture}
\begin{ternaryaxis}
[xlabel=A,ylabel=B,zlabel=C,
legend style={anchor=north,
at={(0.5,-0.35)},draw=none},
legend columns=-1]
\addplot3 coordinates
{(0.81, 0.19, 0.00)
(0.76, 0.17, 0.07)
(0.22, 0.40, 0.30)
(0.76, 0.07, 0.17)
(0.81, 0.00, 0.19)};
\addplot3 coordinates
{(0.85, 0.15, 0.00)
(0.82, 0.13, 0.05)
(0.39, 0.30, 0.40)
(0.82, 0.06, 0.13)
(0.84, 0.00, 0.16)};
\legend{$10^\circ$ \textdegree,
$20^\circ$ \textdegree}
\end{ternaryaxis}
\end{tikzpicture}
```

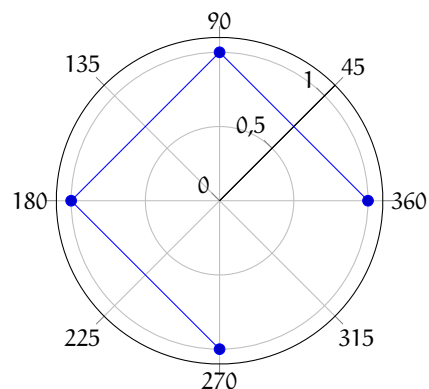


Si noti che;

- draw=none elimina il riquadro della legenda;
- legend columns=-1 ne dispone gli elementi orizzontalmente.

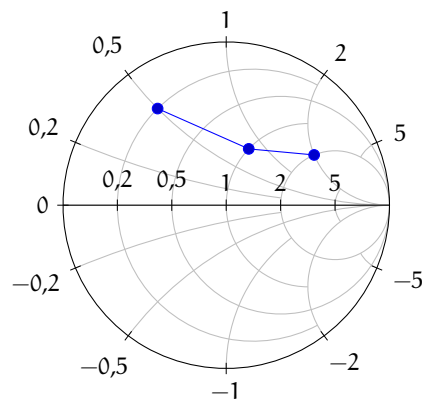
Il prossimo è un sistema di coordinate polari (richiede la libreria polar):

```
\begin{tikzpicture}
\begin{polaraxis}
[xmin=45,xmax=405]
\addplot coordinates
{(0, 1) (90, 1)
(180, 1) (270, 1)};
\end{polaraxis}
\end{tikzpicture}
```



Infine una carta di Smith (richiede la libreria smithchart):

```
\begin{tikzpicture}
\begin{smithchart}
\addplot coordinates
{(0.2, 0.5) (1, 0.8) (2, 2)};
\end{smithchart}
\end{tikzpicture}
```



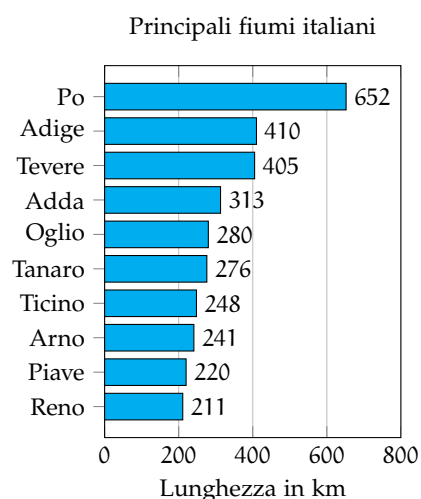
6 DIAGRAMMI A BARRE

Con `pgplots` si possono realizzare anche diagrammi a barre. L'*ortogramma* si distingue dall'*istogramma* non solo per il diverso orientamento delle barre, ma anche perché di solito queste sono staccate tra di loro.

6.1 Ortogrammi

Un ortogramma:

```
\begin{tikzpicture}
\begin{axis}
[xbar,xmin=0,xmax=800,
height=6.5cm,
xmajorgrids=true,
ytick pos=left,
title={Principali fiumi
italiani},
xlabel={Lunghezza in km},
symbolic y coords={Reno,Piave,
Arno,Ticino,Tanaro,Oglio,Adda,
Tevere,Adige,Po},
ytick=data,nodes near coords,
nodes near coords align=%
{horizontal}]
\addplot
[fill=cyan,draw=black]
coordinates
{(211,Reno) (220,Piave)
(241,Arno) (248,Ticino)
(276,Tanaro) (280,Oglio)
(313,Adda) (405,Tevere)
(410,Adige) (652,Po)};
\end{axis}
\end{tikzpicture}
```



Si noti che:

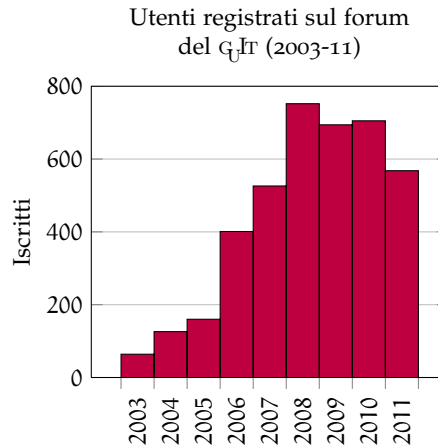
- `xbar` produce le barre orizzontali;
- `xmajorgrids` produce una “griglia” di sole linee verticali;
- `ytick pos` imposta la posizione delle tacche sull’asse y;
- `symbolic y coords` produce il proprio argomento come etichetta della barra;
- `data` attribuisce alle etichette i valori dichiarati come ordinate;
- `node near coords` mette il valore indicato nelle coordinate alla fine di ogni barra;
- `node near coords align` ne aggiusta l’allineamento in modo che non si sovrapponga alla barra;
- `fill` imposta il colore che riempirà le barre;
- `draw` ne imposta il colore di contorno.

Si possono creare ortogrammi anche con il pacchetto `bchart` (se ne veda la documentazione), che richiede una sintassi più semplice.

6.2 Istogrammi

Un istogramma:

```
\begin{tikzpicture}
\begin{axis} [ylabel={Iscritti},
ybar,ymin=0,ymax=800,xtick=data,
ymajorgrids=true,xtick pos=left,
x tick label style={
{rotate=90,anchor=east}},
xticklabel interval boundaries,
symbolic x coords={$2003$,
$2004$, $2005$, $2006$, $2007$,
$2008$, $2009$, $2010$, $2011$,
$2012$},
title style={align=center},
title={Utenti registrati sul
forum\\ del \GuIT{} (2003-11)}]
\addplot
[ybar interval,fill=purple,
draw=black] coordinates
{($2003$, 64) ($2004$, 126)
($2005$, 160) ($2006$, 401)
($2007$, 526) ($2008$, 752)
($2009$, 694) ($2010$, 705)
($2011$, 568) ($2012$, 0)};
\end{axis}
\end{tikzpicture}
```



Alle osservazioni sul grafico precedente, qui riferibili all'asse delle y, si aggiungono le seguenti:

- `xtick pos` imposta la posizione delle tacche sull'asse x;
- `x tick label style` imposta l'aspetto delle etichette delle colonne (qui ruotate di 90° in senso antiorario);
- `xticklabel interval boundaries` centra l'etichetta di ogni colonna;
- `title style` imposta lo stile del titolo, necessario per averlo su due righe;
- `ybar interval` rende adiacenti le colonne, altrimenti staccate;
- un istogramma richiede di definire una colonna *in più* rispetto a quelle necessarie (qui è quella relativa all'anno 2012).

RIFERIMENTI BIBLIOGRAFICI

De Marco, Agostino e Roberto Giacomelli

2011 «Creare grafici con pgfplots», *ArTeXnica*, 12.

Feuersänger, Christian

2011 *Manual for Package pgfplots*, <http://www.ctan.org/tex-archive/graphics/pgf/contrib/pgfplots/doc/latex/pgfplots/pgfplots.pdf>.

Pantieri, Lorenzo e Tommaso Gordini

2012 *L'arte di scrivere con L^AT_EX*, http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf.