Apuntes de LaTeX

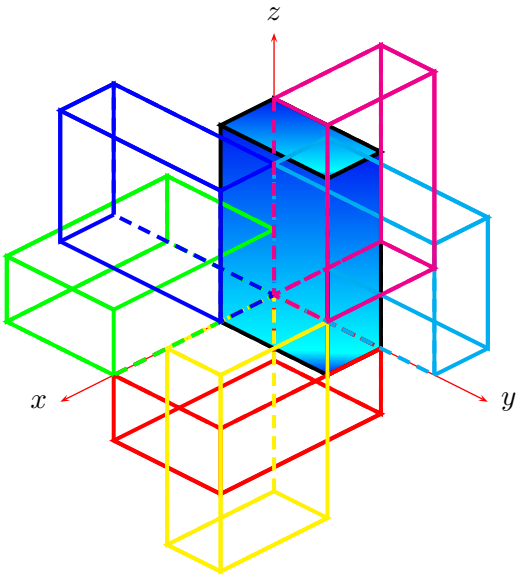
Capítulo 9-3: Dibujando con LaTeX

(Más cosas sobre **PSTricks** [2])

1 El paquete pst-3dplot (continuación)

Las rotaciones (Parámetros `RotX`, `RotY` y `RotZ`) pueden ser también aplicadas, además de a los ejes tridimensionales, a cualquier objeto. Podemos asimismo combinar varias de ellas; en tal caso, se debe especificar, mediante el parámetro `RotSequence` (que toma un valor formado por una combinación de las letras x, y, z) cúal es el orden en el que se aplican las rotaciones (que, en un entorno 3D, no son conmutativas!!!). Veamos un ejemplo donde dibujamos un paralelepípedo con una cierta orientación (a través del comando `\pstThreeDBoX`, que se verá más adelante), y lo sometemos a múltiples rotaciones:

```
{\psset{unit=2,linewidth=1.5pt}
\begin{pspicture}(-2,-1.5)(2,2.5)%
  \pstThreeDCoor[xMin=0,xMax=2,yMin=0,yMax=2,zMin=0,zMax=2]%
  \pstThreeDBox[fillstyle=gradient,RotX=0](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotX=90,RotY=90,RotZ=90,%
    linecolor=red](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotSequence=xzy,RotX=90,RotY=90,RotZ=90,%
    linecolor=yellow](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotSequence=zyx,RotX=90,RotY=90,RotZ=90,%
    linecolor=green](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotSequence=zxy,RotX=90,RotY=90,RotZ=90,%
    linecolor=blue](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotSequence=yxz,RotX=90,RotY=90,RotZ=90,%
    linecolor=cyan](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
  \pstThreeDBox[RotSequence=yzx,RotX=90,RotY=90,RotZ=90,%
    linecolor=magenta](0,0,0)(.5,0,0)(0,1,0)(0,0,1.5)
\end{pspicture}}%
```



1.1 Cuadrículas

Con el comando

`\pstThreeDPlaneGrid[Parámetros](xMin,yMin)(xMax,yMax)`

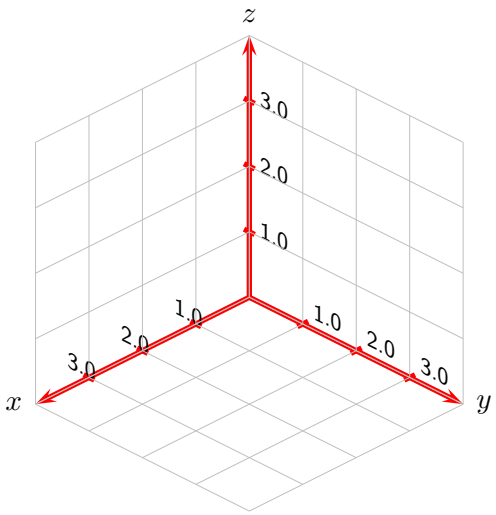
podemos poner cuadrículas en cualquiera de los tres planos xy, xz e yz definidos por los ejes de coordenadas tridimensionales. Para ello debe especificarse en el parámetro `planeGrid` a qué plano de esos tres nos referimos. Las coordenadas `(xMin,yMin)(xMax,yMax)` corresponden al rectángulo sobre el que se extenderá la cuadrícula. Otros parámetros ajustables son:

`subticks` ⟶ Número de divisiones de la cuadrícula (10 por defecto)

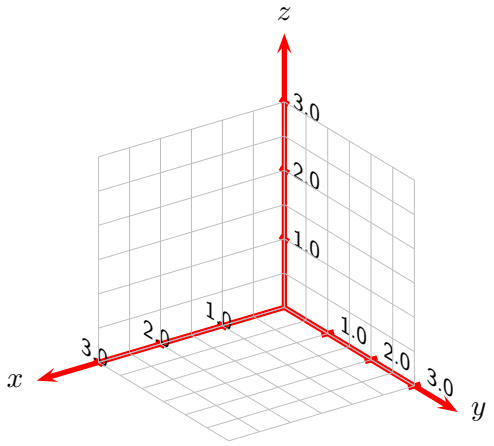
`planeGridOffset` ⟶ Desplazamiento del plano de la cuadrícula en la dirección perpendicular (por defecto 0)

Veamos unos ejemplos:

```
\begin{pspicture}(-3,-3)(3,4)
\pstThreeDCoor[xMin=0,yMin=0,zMin=0,xMax=4,%
yMax=4,zMax=4,linewidth=2pt,IIIDticks=true]
\psset{linewidth=0.2pt,linecolor=lightgray}
\pstThreeDPlaneGrid[subticks=4](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=xz](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=yz](0,0)(4,4)
\end{pspicture}
```

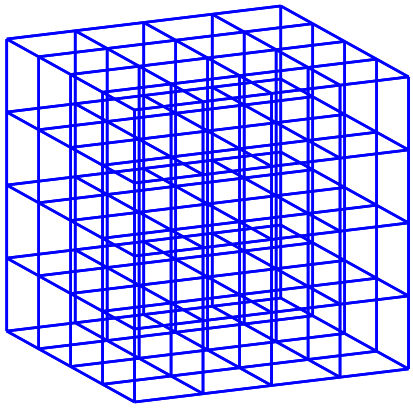


```
\begin{pspicture}(-3,-3)(3,4)
\psset{Alpha=35,Beta=25,planecorr=off}
\pstThreeDCoor[xMin=0,yMin=0,zMin=0,xMax=4,
yMax=4,zMax=4,linewidth=2pt,IIIDticks]
\psset{linewidth=0.2pt,linecolor=lightgray}
\pstThreeDPlaneGrid[subticks=6](0,0)(3,3)
\pstThreeDPlaneGrid[subticks=6,%
planeGrid=xz](0,0)(3,3)
\pstThreeDPlaneGrid[subticks=6,%
planeGrid=yz](0,0)(3,3)
\end{pspicture}
```



En el siguiente ejemplo, vemos como construir una red tridimensional, aplicando desplazamientos crecientes al parámetro `planeGridOffset` con la ayuda de un bucle `\multido`. Nótese también cómo eliminar la representación de los ejes mediante la opción `drawing=false`.

```
\begin{pspicture}(-3,-2)(3,4)
\psset{Alpha=25,Beta=15}
\pstThreeDCoor[xMin=0,yMin=0,zMin=0,xMax=4,%
yMax=4,zMax=4,drawing=false]
\psset{linewidth=1pt,linecolor=blue}
\multido{\n=0+1}{5}{
\pstThreeDPlaneGrid[subticks=4,%
planeGridOffset=\n](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGridOffset=\n,planeGrid=xz](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,
planeGridOffset=\n,planeGrid=yz](0,0)(4,4)}
\end{pspicture}
```



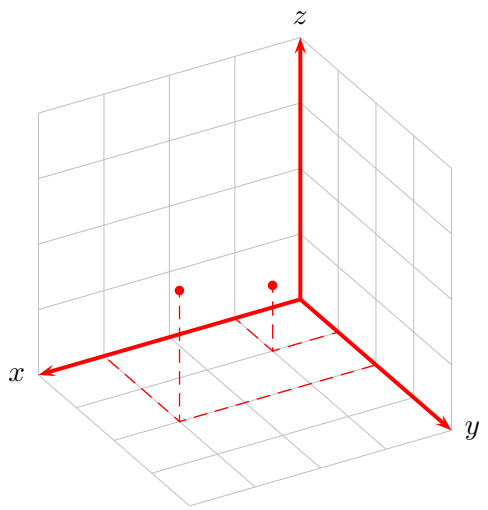
1.2 Objetos gráficos

Para numerosos objetos 2-D, existe su correspondiente versión en tres dimensiones:

- **Puntos** Su sintaxis es `\pstThreeDDot[Parámetros](x,y,z)`

  (x,y,z) son las coordenadas del punto referidas a los ejes coordenados 3-D. Con el parámetro `drawCoor=true` se dibujan líneas guía que ayudan a visualizar la posición 3-D del punto.

```
\begin{pspicture}(-3,-3)(4,4)
\psset{Alpha=30,Beta=30}
\psset{linewidth=0.2pt,linecolor=lightgray}
\pstThreeDPlaneGrid[subticks=4](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=xz](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=yz](0,0)(4,4)
\psset{dotstyle=*,
dotscale=1.2,linecolor=red}
\pstThreeDDot[drawCoor=true](1,1,1)
\pstThreeDDot[drawCoor=true](3,2,2)
\pstThreeDCoor[xMin=0,xMax=4,
yMin=0,yMax=4,zMin=0,zMax=4,linewidth=1.5pt]
% ponemos los ejes al final para que no
% los tapen las líneas de la cuadrícula
\end{pspicture}
```

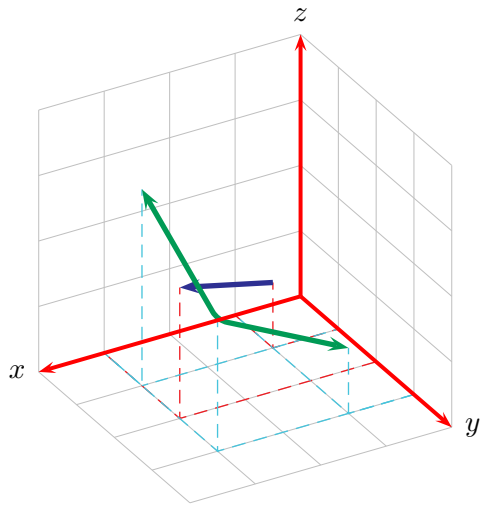


- **Líneas** La sintaxis es la misma que para líneas 2D con `\psline`:

  `\pstThreeDLine[Parámetros]{TipoDeFlecha}(x0,y0,z0)(x1,y1,z1)...(xn,yn,zn)`

  lo cual dibuja líneas tomando como referencia puntos en el espacio tridimensional.

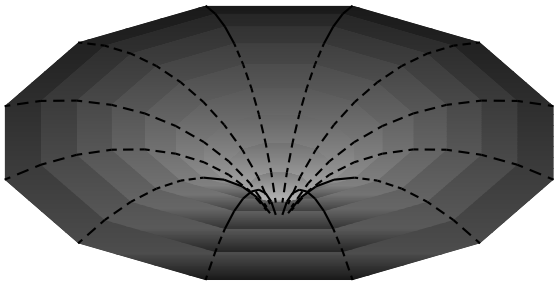
```
\begin{pspicture}(-3,-3)(4,4)
\psset{Alpha=30,Beta=30}
\psset{linewidth=0.2pt,linecolor=lightgray}
\pstThreeDPlaneGrid[subticks=4](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=xz](0,0)(4,4)
\pstThreeDPlaneGrid[subticks=4,%
planeGrid=yz](0,0)(4,4)
\psset{dotstyle=none,linecolor=Red}
\pstThreeDDot[drawCoor=true](1,1,1)
\pstThreeDDot[drawCoor=true](3,2,2)
\pstThreeDLine[linewidth=2pt,
linecolor=Blue]{->}(1,1,1)(3,2,2)
\psset{dotstyle=none,linecolor=SkyBlue}
\pstThreeDDot[drawCoor=true](1,3,1)
\pstThreeDDot[drawCoor=true](3,3,2)
\pstThreeDDot[drawCoor=true](3,1,3)
\pstThreeDLine[linewidth=2pt,
linecolor=ForestGreen,linearc=0.3]
{<->}(1,3,1)(3,3,2)(3,1,3)
\pstThreeDCoor[xMin=0,xMax=4,
yMin=0,yMax=4,zMin=0,zMax=4,linewidth=1.5pt]
\end{pspicture}
```



Se puede ver en el ejemplo como, para dibujar las líneas guía, utilizamos el comando `\pstThreeDDot` con las opciónes `dotstyle=none` y `drawCoor=true`.

En el ejemplo siguiente, se combinan las coordenadas polares con el comando `\multido` para producir líneas poligonales de radio decreciente; nótese cómo se utiliza el comando `\pscustom` para rellenarlas

```
\begin{pspicture}(-3,-2)(4,5)
\pstThreeDCoor[xMin=-3,xMax=3,yMin=-1,
yMax=4,zMin=-1,zMax=3,drawing=false]
\multido{\rA=5+-0.5,\iB=20+5,
 \rC=0.1+0.05,\rD=0.3+0.05}{9}{
 \definecolor{grisA}{gray}{\rC}
 \definecolor{grisB}{gray}{\rD}
 \pscustom[fillstyle=gradient,linestyle=none,
 gradbegin=grisA,gradend=grisB]{
 \multido{\iC=0+30,\iD=30+30}{12}{
  \pstThreeDLine[SphericalCoor=true]
  (\rA,\iC,\iB)(\rA,\iD,\iB)}}}
\multido{\iA=0+30}{12}{
 \multido{\iC=20+5,\iD=25+5,
  \rE=5+-0.5,\rF=4.5+-0.5}{9}{
    \pstThreeDLine[SphericalCoor=true,
linestyle=dashed](\rE,\iA,\iC)(\rF,\iA,\iD)}}
\end{pspicture}
```

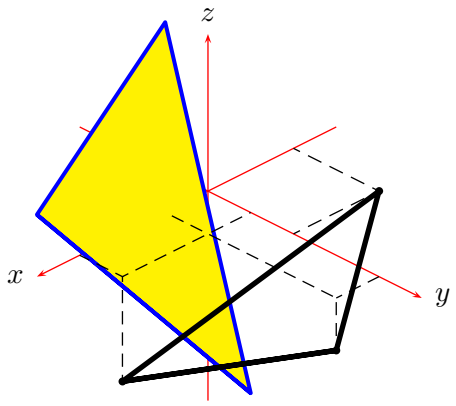


- **Triángulos** Se construyen dando los puntos de sus tres vértices:

  `\pstThreeDTriangle[Parámetros](x0,y0,z0)(x1,y1,z1)(x2,y2,z2)`

  y pueden ser rellenados ajustando la opciones `fillstyle` y `fillcolor`. A diferencia de las líneas, para este objeto sí funciona la opción `drawCoor=true`

```
\begin{pspicture}(-3,-4.25)(3,3.25)
\pstThreeDCoor[xMin=-3,xMax=4,
yMin=-3,yMax=5,zMin=-4,zMax=3]
\pstThreeDTriangle[fillcolor=yellow,
fillstyle=solid,linecolor=blue,
linewidth=1.5pt](5,1,2)(3,4,-1)(-1,-2,2)
\pstThreeDTriangle[drawCoor=true,
linecolor=black,linewidth=2pt]
(3,1,-2)(1,4,-1)(-2,2,0)
\end{pspicture}
```



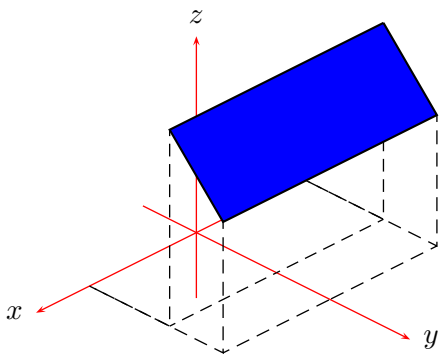
- **Cuadrados y rectángulos** Los cuadrados y rectángulos se definen a través del punto de origen del cuadrilátero y de dos vectores:

  `\pstThreeDSquare[Parámetros](Ox,Oy,Oz)(V1x,V1y,V1z)(V2x,V2y,V2z)`

  Pueden dibujarse líneas guía con las opciones `drawCoor=true` y `dotstyle=none` (para ocultar

los puntos, que en caso contrario son mostrados), pero éstas han de declararse externamente al comando `\pstThreeDSquare` (como se ve en el ejemplo, en un comando `\psset`):

```
\begin{pspicture}(-3,-2)(4,3)
\pstThreeDCoor[xMin=-3,xMax=3,
yMin=-1,yMax=4,zMin=-1,zMax=3]
{\psset{drawCoor=true,dotstyle=none}
\pstThreeDSquare[fillcolor=blue,
fillstyle=solid](-2,1.5,3)(4,0,0)(0,1,-1)}
\end{pspicture}
```

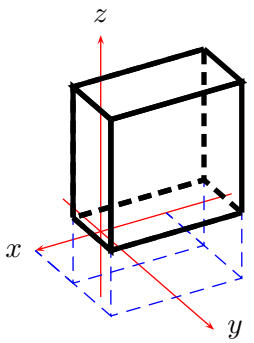


- **Cajas** Para las cajas (paralelepípedos), análogamente al caso de los cuadriláteros, se especifica el punto de origen y tres vectores:

  `\pstThreeDBox[Parámetros](Ox,Oy,Oz)(V1x,V1y,V1z)(V2x,V2y,V2z)(V3x,V3y,V3z)`

  Nota: el parámetro drawCoor=true funciona, al igual que en el caso de los cuadriláteros, pero produce resultados curiosos debido probablemente a un pequeño error en la macro (el lector interesado puede comprobarlo), por lo que es mejor no emplearlo.

```
\begin{pspicture}(-2,-1.25)(3,2.5)
\psset{Alpha=30,Beta=30}
\pstThreeDCoor[xMin=-2,xMax=1,
yMin=-1,yMax=3,zMin=-1,zMax=3]
{\psset{drawCoor=true,linecolor=blue,
dotstyle=none}
\pstThreeDDot(-1,1,1) \pstThreeDDot(1,1,1)
\pstThreeDDot(-1,2,1) \pstThreeDDot(1,2,1)}
\pstThreeDBox[linewidth=2pt]
(-1,1,1)(0,0,2)(2,0,0)(0,1,0)
\end{pspicture}
```

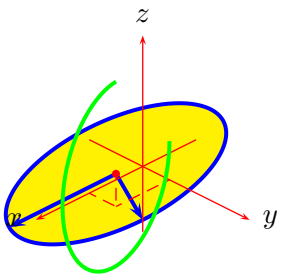


- **Círculos y elipses** Para dibujar círculos ó elipses se deben especificar el centro de la elipse y los dos vectores directores de sus semiejes:

  `\pstThreeDEllipse[Parámetros](Ox,Oy,Oz)(V1x,V1y,V1z)(V1x,V1y,V1z)`

  Podemos dibujar solamente un arco de circunferencia modificando los parámetros `beginAngle` y `endAngle`. Es importante tener en cuenta que la macro que dibuja la elipse se basa en dibujar su ecuación en paramétricas con el comando `\psplot`, por lo que podemos mejorar la resolución aumentando el número de puntos definido en `plotpoints` (50 por defecto)

```
\begin{pspicture}(-2,-2.25)(2,2.25)
\psset{linecolor=blue,linewidth=1.5pt}
\pstThreeDEllipse[fillstyle=solid,
fillcolor=yellow,plotpoints=100]
(1,0.5,0.5)(2,0,0)(0,0.5,-0.5)
\pstThreeDLine{->}(1,0.5,0.5)(3,0.5,0.5)
\pstThreeDLine{->}(1,0.5,0.5)(1,1,0)
\pstThreeDCoor[xMax=2,yMax=2,zMax=2]
\pstThreeDDot[linecolor=red,
drawCoor=true](1,0.5,0.5)
\psset{beginAngle=0,endAngle=270,
linecolor=green}
\pstThreeDEllipse(1,0.5,0.5)
(-0.5,0.5,0.5)(0.5,0.5,-1)
\end{pspicture}
```

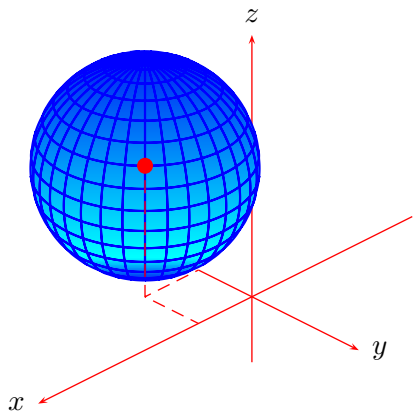


- **Esferas** Se pueden dibujar esferas con el comando:

  `\pstThreeDSphere[Parámetros](O1,O2,O3){Radio}`

  donde `(O1,O2,O3)` son las coordenadas del centro de la esfera.

```
\begin{pspicture}(-4,-2)(2,3.5)
  \pstThreeDCoor[xMin=-3,yMax=2]
  \pstThreeDSphere[linecolor=blue,
  fillstyle=gradient](1,-1,2){1.5}
  \pstThreeDDot[dotstyle=*,dotscale=2,
  linecolor=red,drawCoor=true](1,-1,2)
\end{pspicture}
```



1.3 Colocando objetos

1.3.1 pstThreeDPut

`\pstThreeDPut` funciona de forma similar a `\rput`:

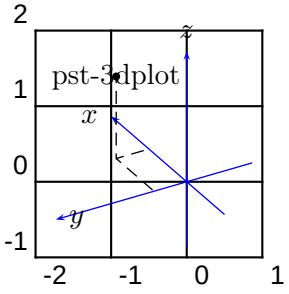
`\pstThreeDPut[Parámetros](x,y,z){Objeto}`

colocando `Objeto` en las coordenadas (x,y,z). Se puede ajustar el parámetro `origin`, con valores:

`origin=lt|lB|lb|t|c|B|b|rt|rB|rb`

Estas opciones tienen el mismo significado que para el comando `\rput`, esto es, el punto de referencia para la colocación del objeto, como se ilustra en el siguiente esquema:

```
\begin{pspicture}(-2,-1.25)(1,2.25)
  \psgrid[subgriddiv=0]
  \psset{Alpha=-60,Beta=30}
  \pstThreeDCoor[linecolor=blue,%
xMin=-1,xMax=2,yMin=-1,yMax=2,zMin=-1,zMax=2]
  \pstThreeDPut(1,0.5,1.25){pst-3dplot}
  \pstThreeDDot[drawCoor=true](1,0.5,1.25)
\end{pspicture}
```



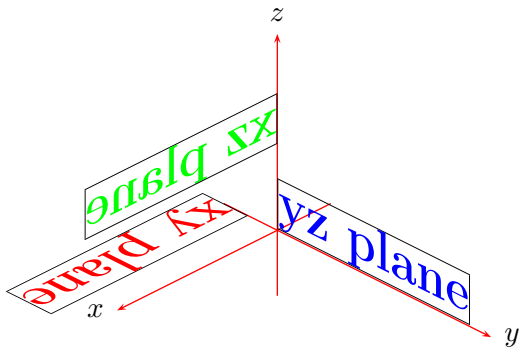
1.3.2 pstPlanePut

La sintaxis de este comando es:

`\pstPlanePut[plane=<plano 2D>,planecorr=<Corrección de plano>](x,y,z){Objeto}`

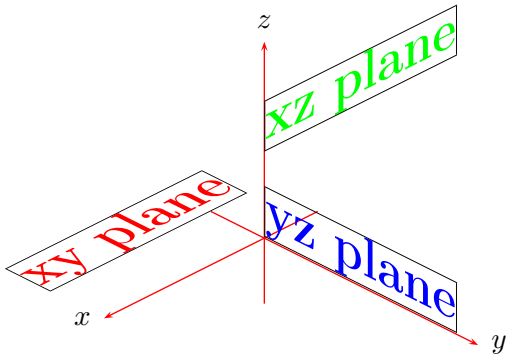
y su utilidad es colocar `Objeto` en las coordenadas (x, y, z), orientándolo en la dirección del plano indicado por el parámetro `plane` (a elegir entre `xy`, `xz` e `yz`). Veamos un ejemplo:

```
\begin{pspicture}(-3,-2)(3,3)
  \pstThreeDCoor[xMin=-1,xMax=3,
  yMin=-1,yMax=4,zMin=-1,zMax=3]
  \pstPlanePut[plane=xy](0,-1.2,0)
  {\fbox{\huge\red\textbf{xy plane}}}
  \pstPlanePut[plane=xz](0,0,1.5)
  {\fbox{\huge\green\textbf{xz plane}}}
  \pstPlanePut[plane=yz](0,0,0.2)
  {\fbox{\huge\blue\textbf{yz plane}}}
\end{pspicture}
```



Puede verse en el ejemplo anterior que los objetos pueden aparecer en ocasiones con orientaciones no deseadas. Para solucionar esto se debe ajustar el parámetro `planecorr`, con las opciones `off` (sin corrección), `normal` (los planos se rotan para hacer los texto legibles) y `xyrot` (corrección adicional útil para algunos casos especiales). En el siguiente ejemplo puede apreciarse el efecto de la corrección:

```
\begin{pspicture}(-3,-2)(3,3)
\pstThreeDCoor[xMin=-1,xMax=3,
yMin=-1,yMax=4,zMin=-1,zMax=3]
\pstPlanePut[plane=xy,planecorr=normal]
(3,-1.2,0)
{\fbox{\huge\red\textbf{xy plane}}}
\pstPlanePut[plane=xz,planecorr=normal]
(0,0,1.5)
{\fbox{\huge\green\textbf{xz plane}}}
\pstPlanePut[plane=yz,planecorr=normal]
(0,0,0.2)
{\fbox{\huge\blue\textbf{yz plane}}}
\end{pspicture}
```



1.4 Representación de funciones matemáticas y datos externos

Existen dos comandos para la representación de funciones de dos variables, que funcionan de forma similar a `\psplot` y `\parametricplot`:

- `\psplotThreeD[Parámetros](xMin,xMax)(yMin,yMax){función}` Se utiliza para representar funciones de dos variables $z = f(x, y)$; debemos especificar los valores `(xMin,xMax)` e (yMin,yMax) del intervalo de variación de x e y, respectivamente, así como la expresión de la función en notación postscript (RPN). Podemos dar valores a los siguientes parámetros, similares a los propios de `\psplot`:

Nombre de opción	**Valor**
`plotstyle`	`dots`
	`line`
	`polygon`
	`curve`
	`ecurve`
	`ccurve`
	`none` (defecto)
`showpoints`	falso por defecto
`xPlotpoints`	25 por defecto
`yPlotpoints`	25 por defecto
`drawStyle`	`xLines` (defecto)
	`yLines`
	`xyLines`
	`yxLines`
`hiddenLine`	falso por defecto

Al igual que en el caso de `\psplot`, existe la posibilidad de utilizar el paquete pst-infixplot para traducir el nombre de la función, utilizando el comando `\infixtoRPN{f(x,y)}` y pasando a continuación el comando `\RPN` como argumento `función` de la instrucción `\psplotThreeD`. En los ejemplos

siguientes, utilizaremos éste procedimiento para simplificar la escritura de la función.[1]

Por tanto, tomemos la función:

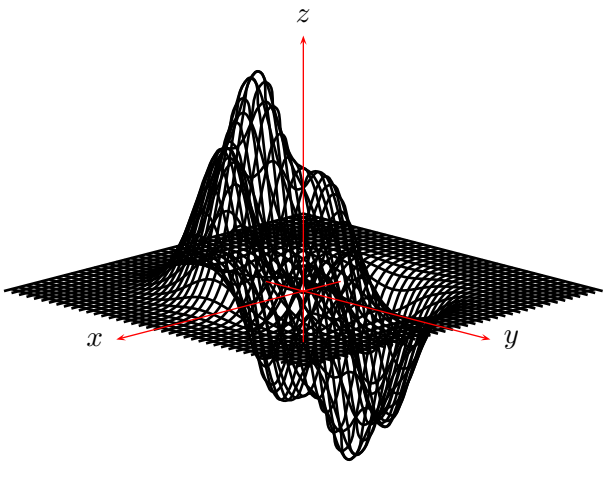
$$z = 10\left(x^3 + xy^4 - \frac{x}{5}\right) e^{-\left(x^2+y^2\right)} + e^{-\left((x-1.225)^2+y^2\right)} \tag{1}$$

que representamos como:

```
\infixtoRPN{(10*((x^3)+(x*(y^4))-(x/5))*(2.71828^(-(x^2+y^2))))+^(2.71828-((x-1.225)^2+(y^2)))}
```

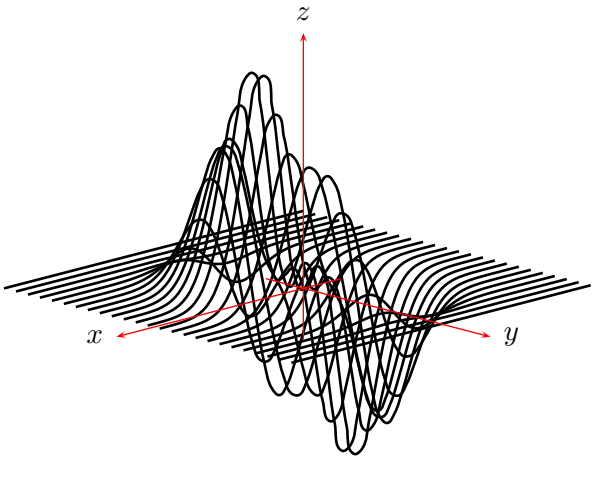
y a continuación la dibujamos en 3D:

```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=xyLines,xPlotpoints=40,
yPlotpoints=40,
linewidth=1pt](-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```



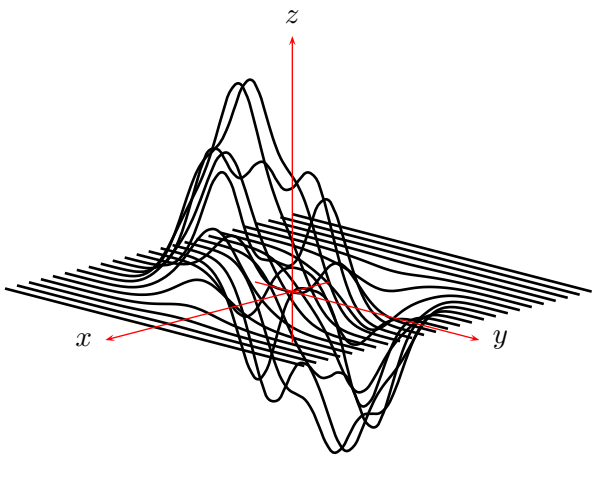
podemos elegir, mediante `drawStyle`, si queremos pintar líneas sólo en la dirección x ó y:

```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=xLines,xPlotpoints=40,
linewidth=1pt](-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```



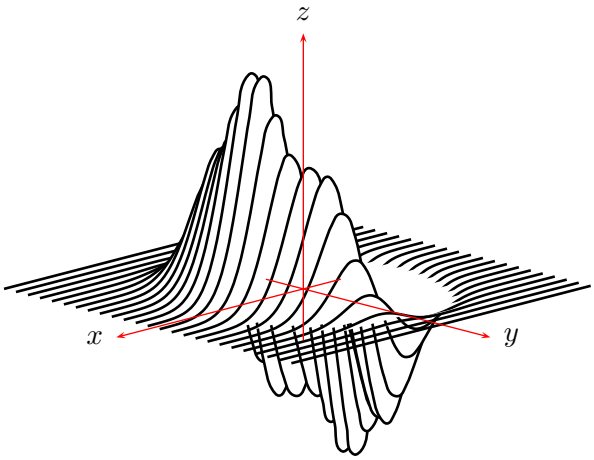
[1] Nótese que, en este caso, no está disponible la opción algebraic=true, como sucedía en `psplot`, por lo que el uso del paquete pst-infixplot es la única alternativa

```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=yLines,yPlotpoints=40,
linewidth=1pt](-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```

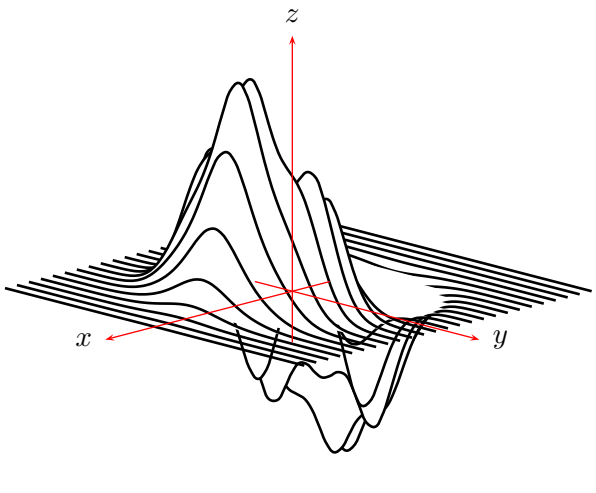


el resultado puede mejorarse con la opción `hiddenLine=true`:

```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=xLines,xPlotpoints=40,
linewidth=1pt,hiddenLine=true]
(-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```

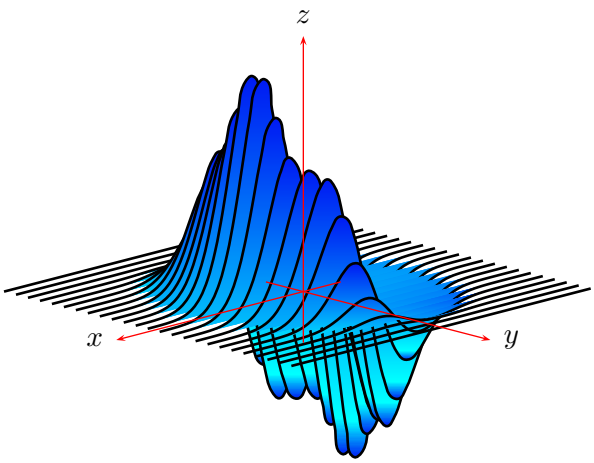



```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=yLines,yPlotpoints=40,
linewidth=1pt,hiddenLine=true]
(-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```



por último, las líneas pueedn rellenarse de la forma que queramos; empleando la opción fillstyle=gradient, se obtiene un resultado muy elegante:

```
\begin{pspicture}(-3,-2.5)(3,2.5)
\psset{Beta=15,unit=0.7cm}
\psplotThreeD[plotstyle=curve,
drawStyle=xLines,xPlotpoints=40,
linewidth=1pt,hiddenLine=false,
fillstyle=gradient](-4,4)(-4,4){\RPN}
\pstThreeDCoor[xMin=-1,xMax=5,
yMin=-1,yMax=5,zMin=-1,zMax=5]
\end{pspicture}
```



- La sintaxis para producir gráficas de funciones en forma paramétrica es:

  `\parametricplotThreeD[Parámetros](tMin,tMax)(uMin,uMax){función}`

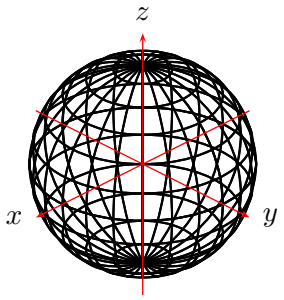
  donde ahora las variables han de llamarse `t`, `u`, y `función` se debe expresar en la forma (x(t,u), y(t,u), z(t,u)). Es posible también representar funciones dependientes sólo de un parámetro (una espiral, por ejemplo), para lo cual se cambiaría el comando a:

  `\parametricplotThreeD[Parámetros](tMin,tMax){función}`

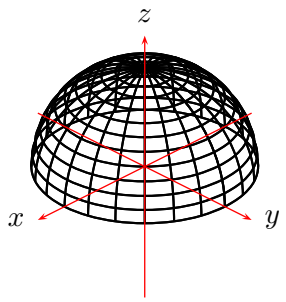
  con `función` en la forma (x(t), y(t), z(t)).

Como ejemplo, dibujemos una esfera y una semiesfera:

```
\begin{pspicture}(-2,-2)(2,2)
\setDefaults
\infixtoRPN{1.5*cos(t)*sin(u),
1.5*cos(t)*cos(u),1.5*sin(t)}
\parametricplotThreeD[plotstyle=curve]
(0,360)(0,360){\RPN}
\infixtoRPN{1.5*cos(u)*sin(t),
1.5*cos(u)*cos(t),1.5*sin(u)}
\parametricplotThreeD[plotstyle=curve]
(0,360)(0,360){\RPN}
\pstThreeDCoor[xMin=-2,xMax=2,
yMin=-2,yMax=2,zMin=-2,zMax=2]
\end{pspicture}
```



```
\begin{pspicture}(-2,-2)(2,2)
\setDefaults
\infixtoRPN{1.5*cos(t)*sin(u),
1.5*cos(t)*cos(u),1.5*sin(t)}
\parametricplotThreeD[plotstyle=curve]
(0,180)(0,360){\RPN}
\infixtoRPN{1.5*cos(u)*sin(t),
1.5*cos(u)*cos(t),1.5*sin(u)}
\parametricplotThreeD[plotstyle=curve]
(0,360)(0,180){\RPN}
\pstThreeDCoor[xMin=-2,xMax=2,
yMin=-2,yMax=2,zMin=-2,zMax=2]
\end{pspicture}
```



Es importante tener en cuenta que para `\parametricplotThreeD` los parámetros `drawStyle` y `hiddenLine` no están definidos, así que para dibujar líneas a lo largo de x e y, se debe utilizar el "truco" mostrado en el ejemplo: intercambiar los parámetros `t` y `u` en las ecuaciones, y superponer las dos representaciones de la función.