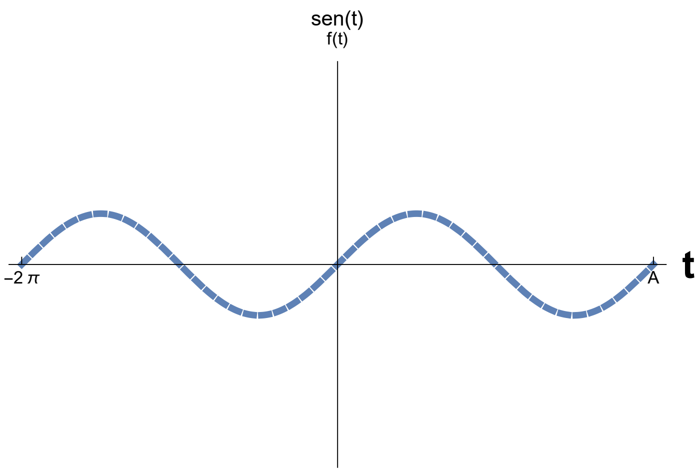
```
(* Comandos usan [], listas o estructuras de almacenamiento usan {}, metodos usan → *)
(* Si pones xy wolfram piensa que es el nombre de una variable,
si pones x y, como tiene espacio, wolfram piensa que es el producto x * y *)
(* Ctrl + 6: exponentes
  Con tab mueves entre espacios de escritura, el actual espacio se pinta en azul.
   Alt + arriba o abajo con mouse: zoom a un plot (debe hacerse sobre el plot) *)
(* Tienes un metodo, por ejemplo PlotTheme,
y quiero que el tema sea tanto Detailed como clasico,
entonces cambias el input del metodo por una lista de parámetros dentro de {}
 A cualquier constante de texto ("") se le puede hacer un StyleForm *)
Plot[Sin[x], {x, -2 Pi, 2 Pi}, PlotRange → {-8/2, 8/2},
 PlotStyle -> {Dashed, Thickness[0.01]}, AxesLabel → {StyleForm["t", Bold, 20], "f(t)" },
 Ticks → {{-2 Pi, {2 Pi, "A"}}, None }, PlotLabel → "sen(t)"]
```

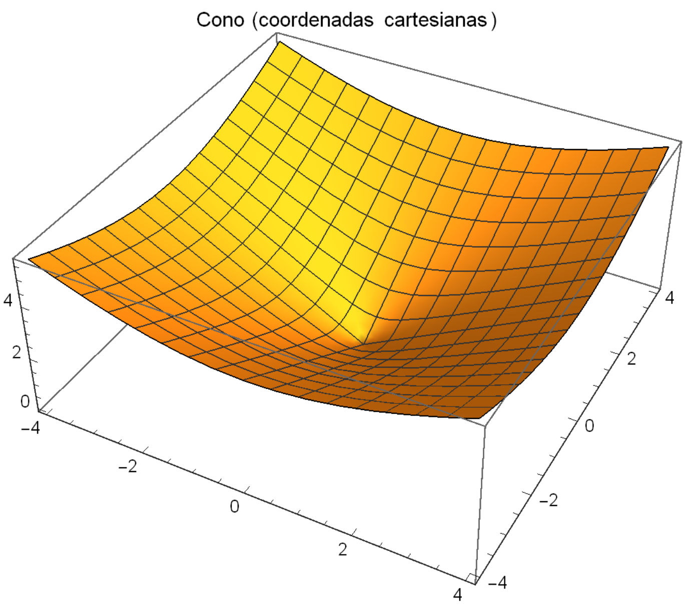
Out[ ]=



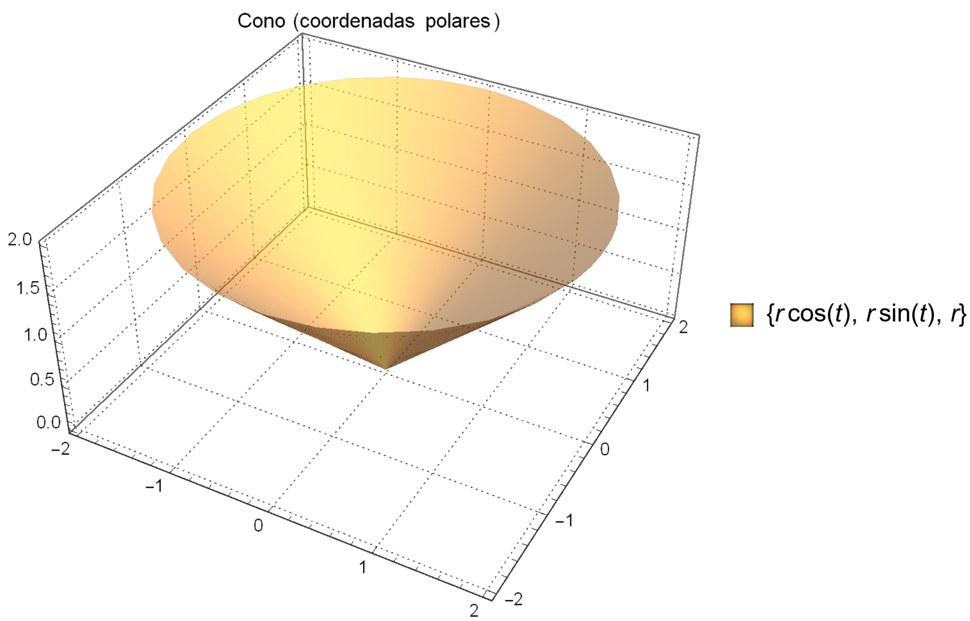
In[ ]:=

```
Plot3D[√(x^2 + y^2) , {x, -4, 4}, {y, -4, 4},
 PlotLabel → "Cono (coordenadas cartesianas)", PlotTheme → "Automatic"]
 (* Cono con Plot3D, y este comando necesita funciones*)
ParametricPlot3D[{{r Cos[t], r Sin[t], r}}, {r, 0, 2 },
 {t, 0, 2 Pi}, Mesh → None, PlotStyle → {Opacity[0.5]},
 PlotLabel → "Cono (coordenadas polares)", PlotTheme → "Detailed"]
(* Si no tienes o quieres comandos, mejor parametrizas con coordenadas polares *)
(* Si definimos parametrización en coordenadas cartesianas, tendríamos x = u,
y = v, donde z (dependiente) resultaría igual. Como no sirve,
usamos parametrizaciones llamadas coordenadas polares, esféricas, etc.  *)
ParametricPlot3D[{{r Cos[t], r Sin[t], r^2}}, {r, 0, 2 },
 {t, 0, 2 Pi}, Mesh → None, PlotStyle → {Opacity[0.8]}, PlotLabel →
  StyleForm["Paraboloide (coordenadas polares)", Bold, 15, FontFamily → "Helvetica"],
 PlotTheme → {"Detailed", "Classic"}]
```

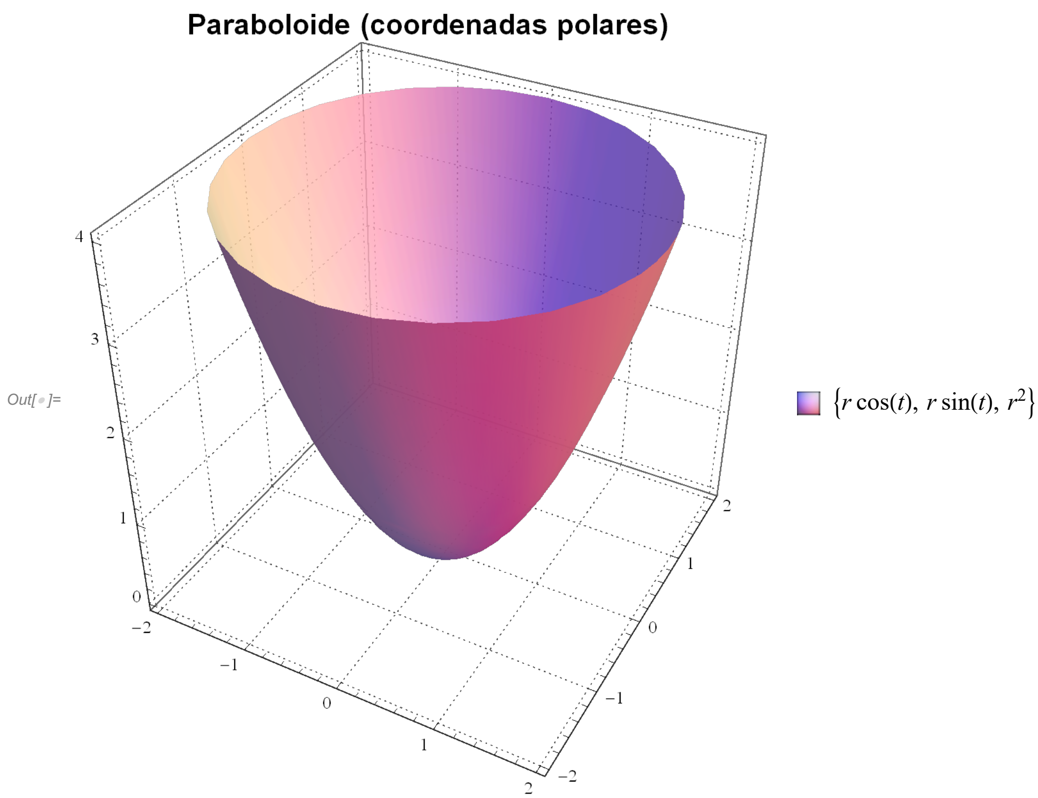
Out[∘]=


Cono (coordenadas cartesianas)
4
2
0
−4
−2
0
2
4
4
2
0
−2
−4

Out[∘]=


Cono (coordenadas polares)
2.0
1.5
1.0
0.5
0.0
−2
−1
0
1
2
2
1
0
−1
−2
{r cos(t), r sin(t), r}

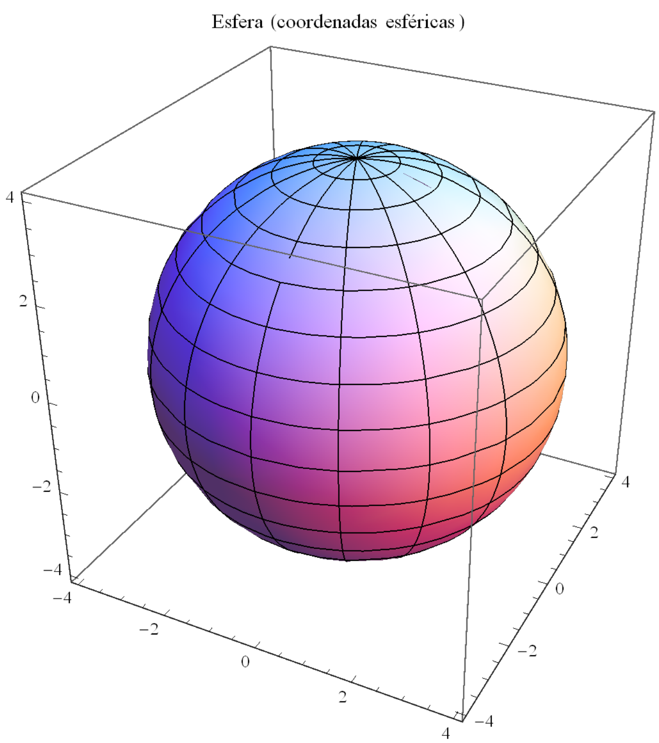
Out[ ]=




```
(* Hacer esfera con parametrización de coordenadas esféricas *)
```

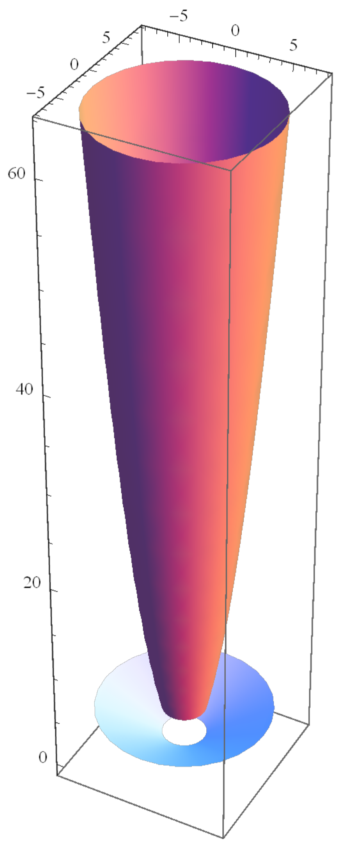
In[ ]:=
```
ParametricPlot3D[{{4 Sin[x] Cos[t], 4 Sin[x] Sin[t], 4 Cos[x]}}, {x, 0, Pi},
  {t, 0, 2 Pi}, PlotLabel → "Esfera (coordenadas esféricas)", PlotTheme → "Classic"]
```

Out[ ]=



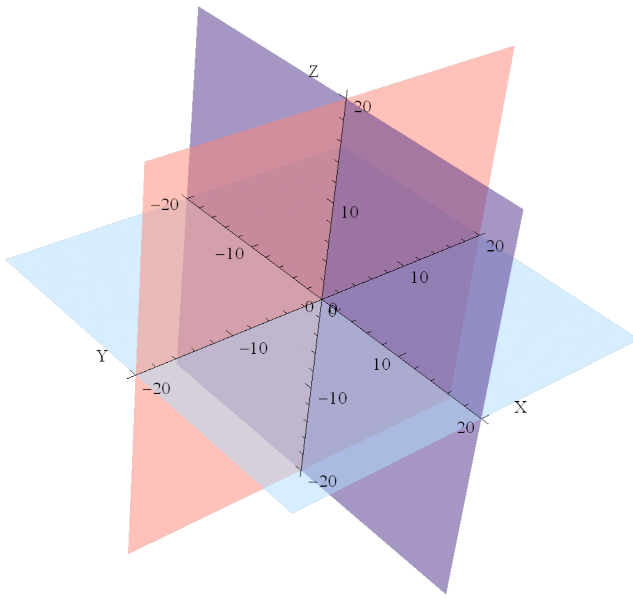
In[ ]:=
```
fig1 = ParametricPlot3D[{{r Cos[t], r Sin[t], 1/2 r}, {r Cos[t], r Sin[t], r^2}},
  {r, 2, 8}, {t, 0, 2 Pi}, Mesh → None, PlotTheme → "Classic"]
```

Out[ ]=

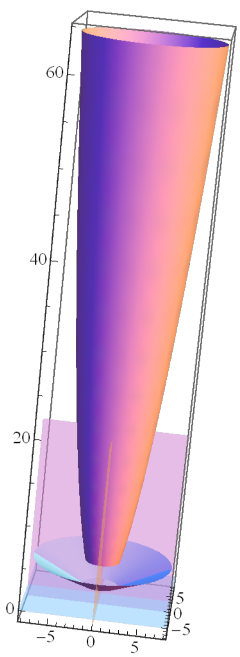


In[ ]:=
```
planos = ParametricPlot3D[{{a, b, 0}, {a, 0, b}, {0, a, b}}, {a, -20, 20},
  {b, -20, 20}, Mesh → None, PlotStyle → {Opacity[0.5]}, PlotTheme → "Classic",
  Boxed → False, AxesOrigin → {0, 0, 0}, AxesLabel → {"X", "Y", "Z"}]
```

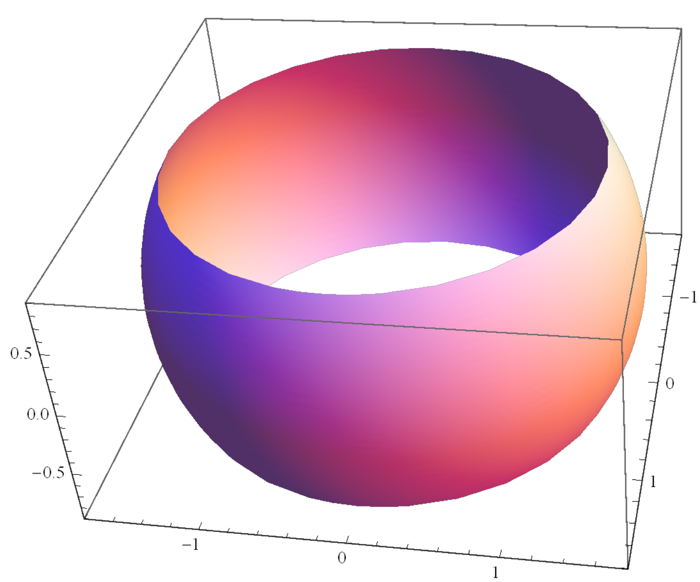
Out[ ]=



In[•]:= Show[fig1, planos]

Out[•]= 


```
In[•]:= ParametricPlot3D[{{√3 Sin[x] Cos[t], √3 Sin[x] Sin[t], √3 Cos[x]}},
  {x, Pi / 3, (2 * Pi) / 3}, {t, 0, 2 Pi}, Mesh → None, PlotTheme → "Classic"]
```

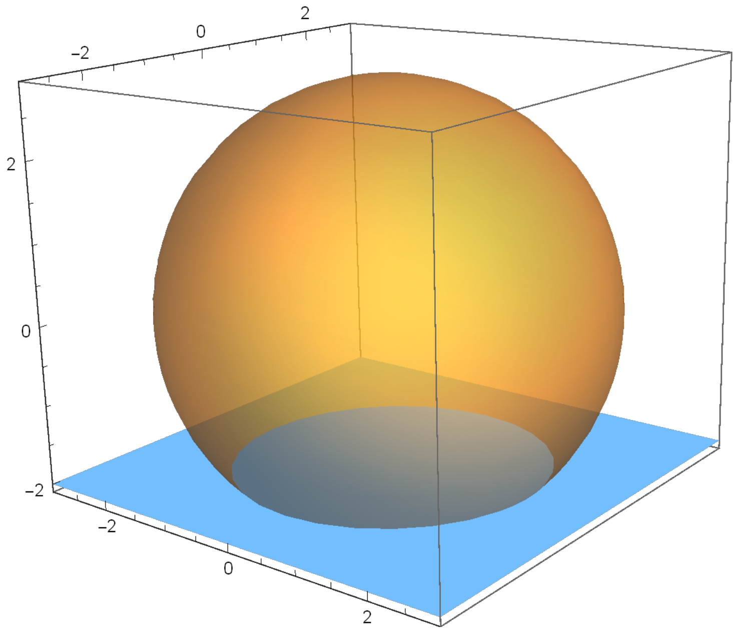
Out[•]= 


```
(* porción superior de esfera x^2 +  y^2 + z^2 = 8 cortada por plano z = - 2 *)
```

In[∘]:=
```
esfera1 =
 ParametricPlot3D[{2 √2 Sin[x] Cos[t], 2 √2 Sin[x] Sin[t], 2 √2 Cos[x]}, {x, 0, 3π/4},
  {t, 0, 2π}, Mesh → None, PlotTheme → "Automatic", PlotStyle → {Opacity[0.5]}]
planoZ = ParametricPlot3D[{x, y, -2}, {x, -4, 4},
  {y, -4, 4}, Mesh → None, PlotTheme → "Classic"]
```

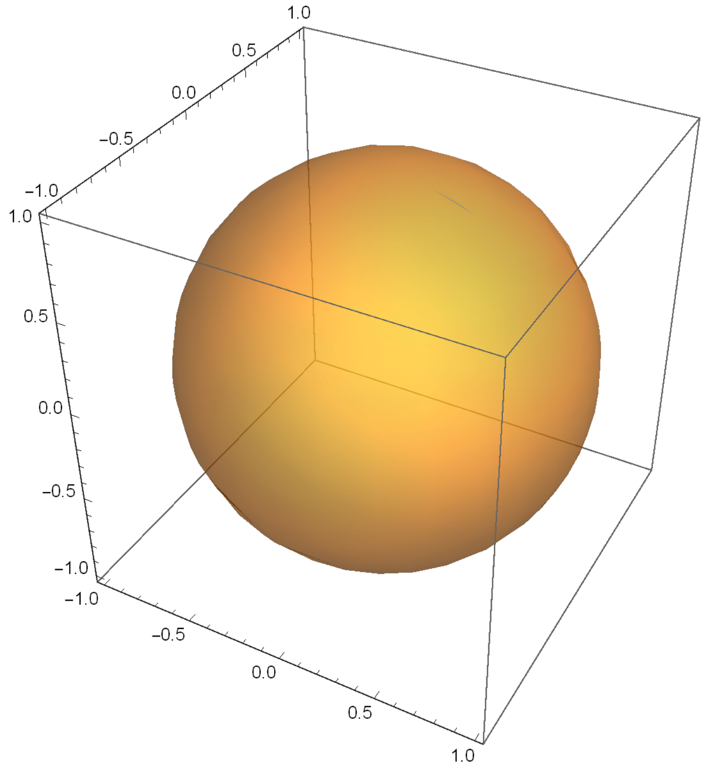
In[∘]:=
```
Show[esfera1, planoZ]
```

Out[∘]=



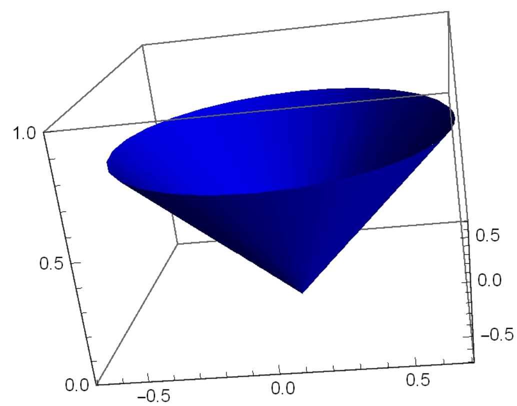
In[•]:=
```
esfera2 = ParametricPlot3D[{Sin[x] Cos[t], Sin[x] Sin[t], Cos[x]}, {x, 0, Pi},
  {t, 0, 2 π}, Mesh → None, PlotTheme → "Automatic", PlotStyle → {Opacity[0.5]}]
```

Out[•]=



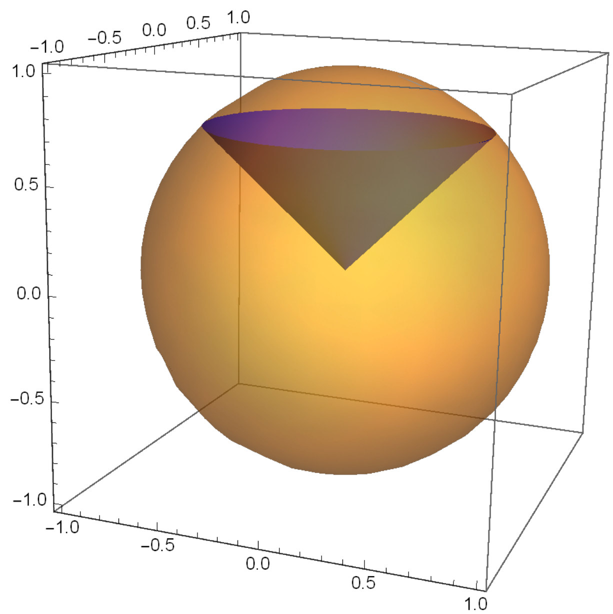
In[•]:=
```
cono2 = ParametricPlot3D[{r Cos[t], r Sin[t], r}, {r, 0, 1/Sqrt[2]}, {t, 0, 2 Pi},
  Mesh → None, PlotStyle → {Blue}, PlotRange → {{-1/Sqrt[2], 1/Sqrt[2]}, {-1/Sqrt[2], 1/Sqrt[2]}, {0, 1}}]
```

Out[•]=



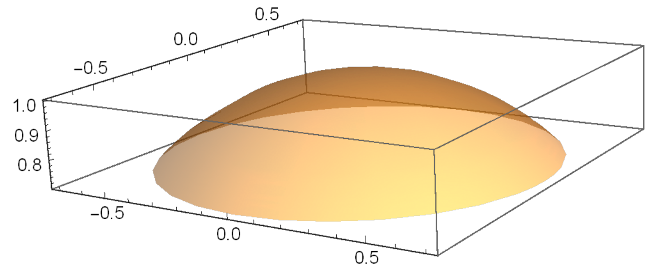
In[•]:=
```
Show[esfera2, cono2]
```

Out[•]=



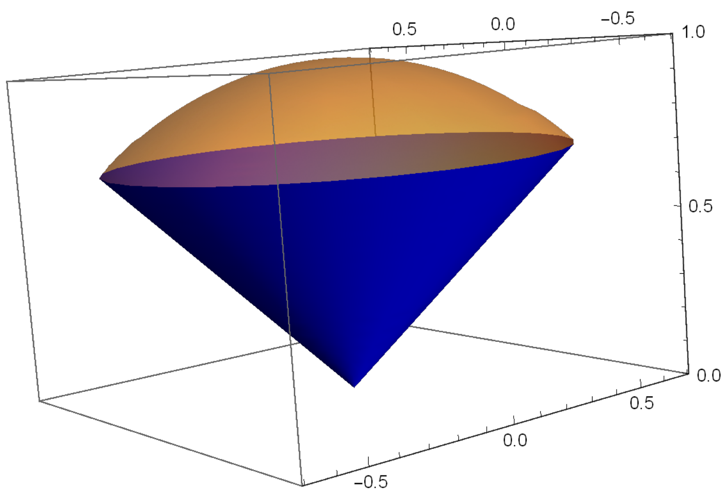
```
esfera3 = ParametricPlot3D[{Sin[x] Cos[t], Sin[x] Sin[t], Cos[x]}, {x, 0, Pi / 4},
  {t, 0, 2 π}, Mesh → None, PlotTheme → "Automatic", PlotStyle → {Opacity[0.5]}]
```

Out[∘]=



In[∘]:=
```
Show[cono2, esfera3]
```

Out[∘]=

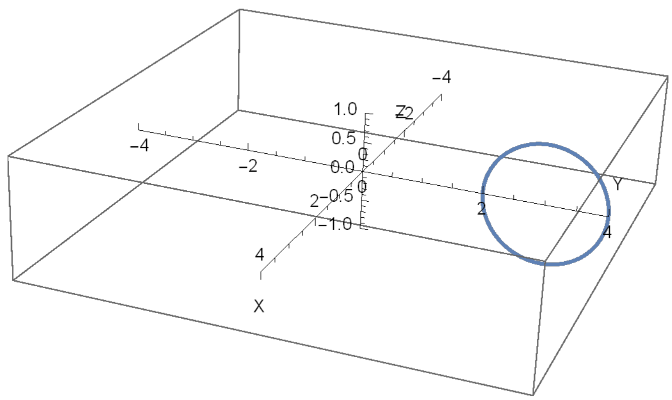



```
(* 1*)

(*ParametricPlot3D[{0, r Cos[x], r Sin[x]}, {r, 0, 2 Cos[x]},
 {x, -Pi  / 2,  Pi / 2}, AxesOrigin→{0, 0, 0}, AxesLabel→ {"X", "Y", "Z"}]*)
(* cambias centro en x, y la dejas en sinX porque ya está centrada *)
```

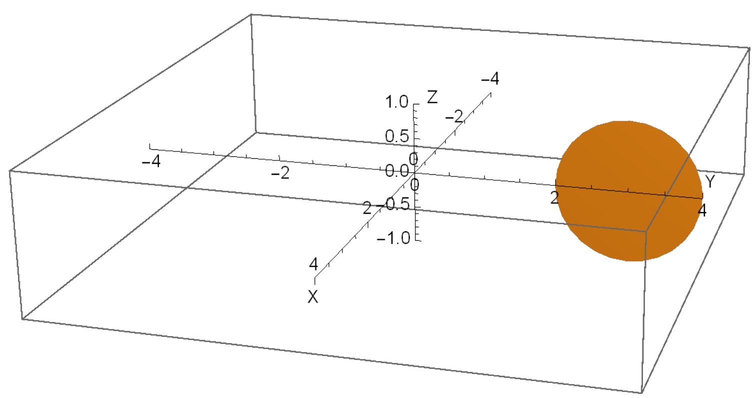
In[∘]:=
```
ParametricPlot3D[{0, (3 + Cos[x]), Sin[x]}, {x, 0, 2 Pi}, AxesOrigin → {0, 0, 0},
 AxesLabel → {"X", "Y", "Z"}, PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}]
```

In[∘]:=




```
ParametricPlot3D[{0, (3 + r Cos[x]), r Sin[x]}, {x, 0, 2 Pi},
 {r, 0, 1}, AxesOrigin → {0, 0, 0}, AxesLabel → {"X", "Y", "Z"},
 PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}, Mesh → None]
```

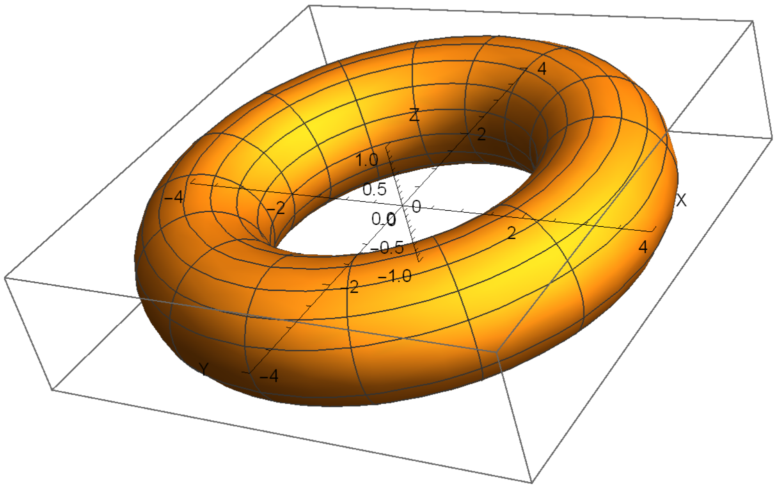
Out[∘]=




```
(* Ahora la distancia al origen es 3 + cosX *)
distance = (3 + Cos[x])
```

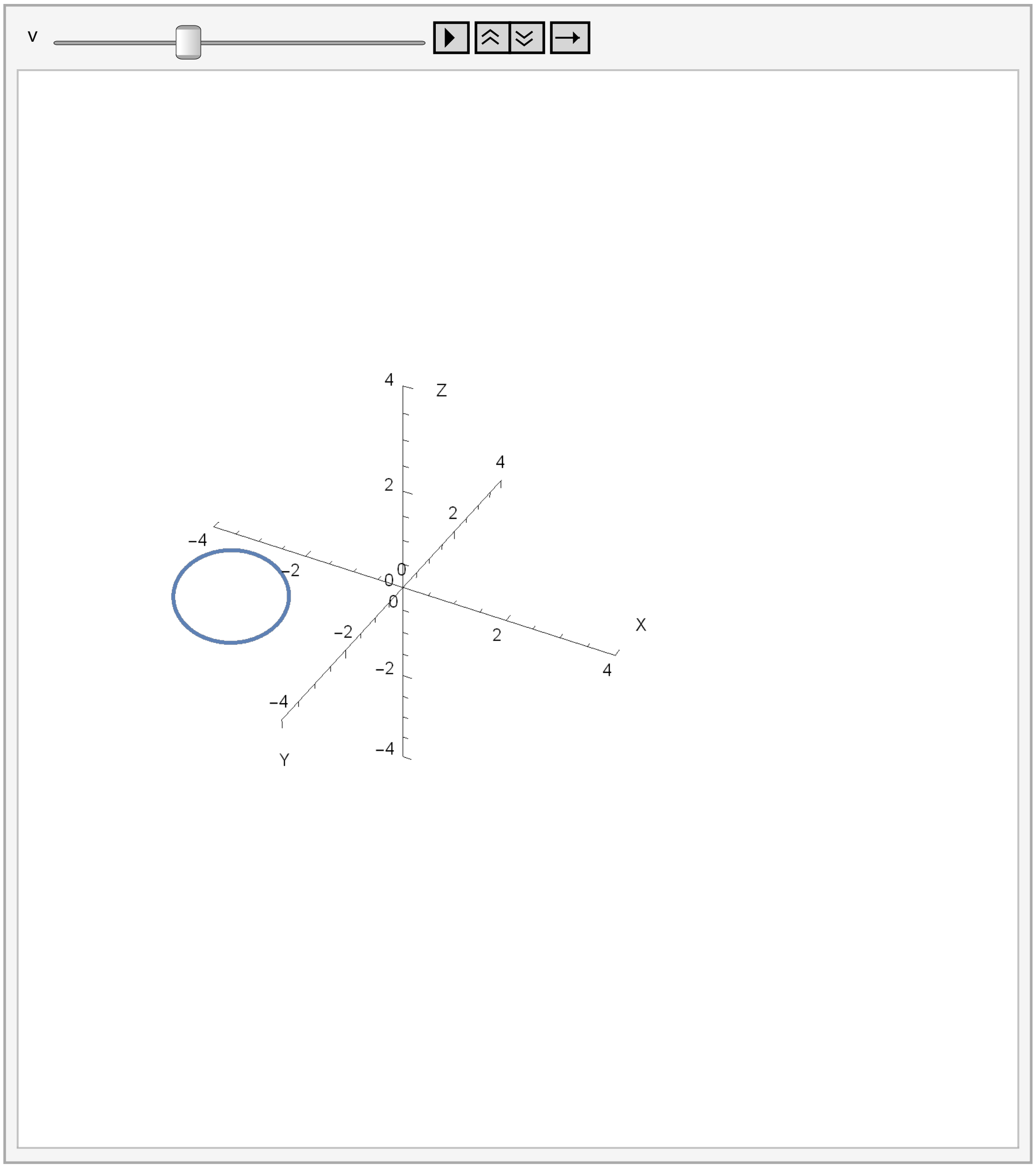
In[ ]:=
```
ParametricPlot3D[{distance * Cos[t], distance * Sin[t], Sin[x]},
 {x, 0, 2 Pi}, {t, 0, 2 Pi}, AxesOrigin → {0, 0, 0},
 AxesLabel → {"X", "Y", "Z"}, PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}]
```

Out[ ]=



```
Animate[ParametricPlot3D[RotationMatrix[v, {0, 0, 1}].{0, (3 + Cos[x]), Sin[x]},
  {x, 0, 2 Pi}, AxesOrigin → {0, 0, 0}, AxesLabel → {"X", "Y", "Z"}, Boxed → False,
  PlotRange → {{-4, 4}, {-4, 4}, {-4, 4}}], {v, 0, 2 Pi}, AnimationRunning → False]
RotationMatrix[v, {0, 0, 2}] // MatrixForm
(*RotationMatrix[v, {0, 0, 1}].{0, (3 + Cos[x]), Sin[x]} esto funciona
 generando cada punto y rotándolo, porque una parametrización ya es una matriz *)
```

Out[ ]=



Out[ ]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[v] & -\text{Sin}[v] & 0 \\ \text{Sin}[v] & \text{Cos}[v] & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```
(* Esto es el eje de rotacion, si dejan de ser ejes coordenados {1, 0, 0},
{0, 1, 0}, {0, 0, 1}, la matriz se complica mas. *)
RotationMatrix[v, {0, 1, 0}] // MatrixForm
```

Out[∘]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[v] & 0 & \text{Sin}[v] \\ 0 & 1 & 0 \\ -\text{Sin}[v] & 0 & \text{Cos}[v] \end{pmatrix}$$

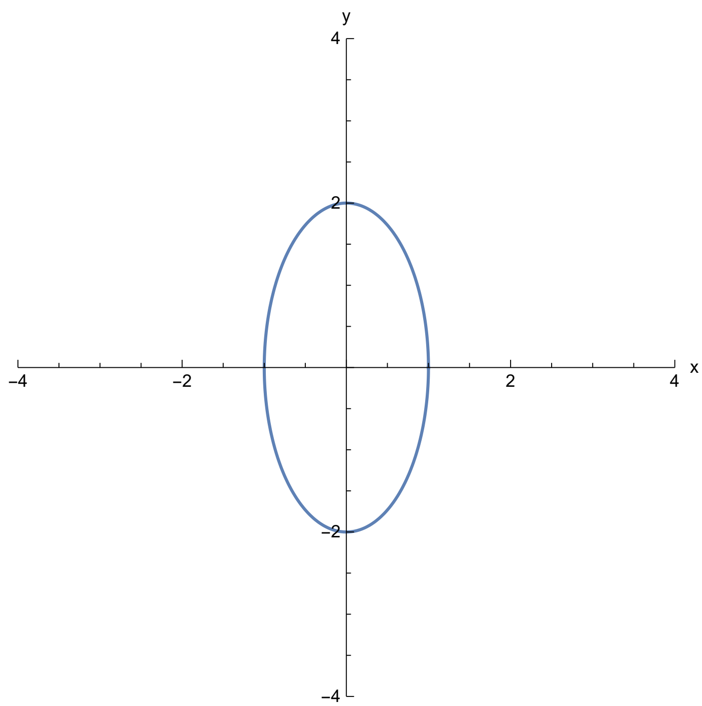
In[∘]:=
```
RotationMatrix[v, {1, 0 , 0}] // MatrixForm
```

Out[∘]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \text{Cos}[v] & -\text{Sin}[v] \\ 0 & \text{Sin}[v] & \text{Cos}[v] \end{pmatrix}$$

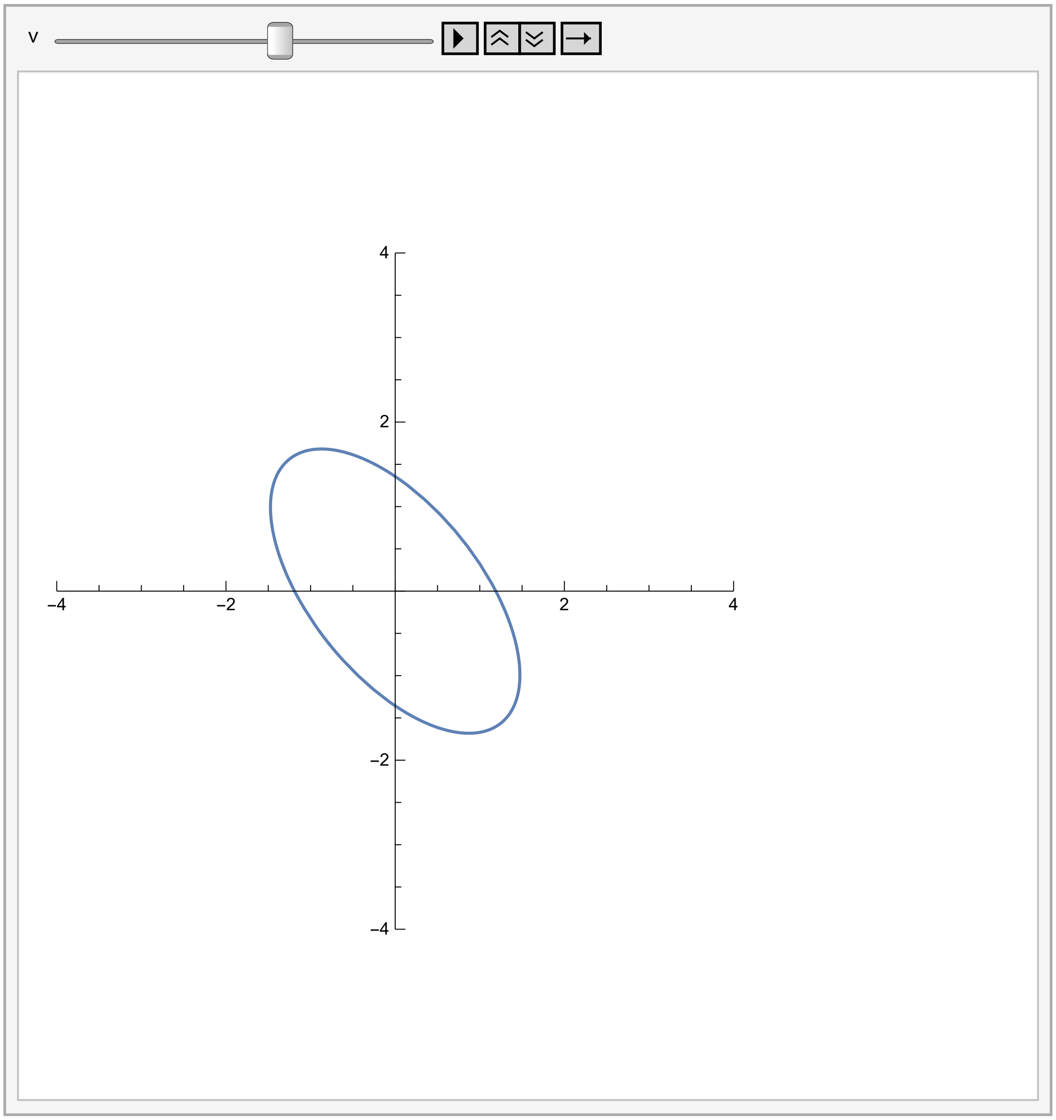
In[∘]:=
```
ParametricPlot[{Cos[t], 2 Sin[t]}, {t, 0, 2 π},
 PlotRange → {{-4, 4}, {-4, 4}}, AxesLabel → {"x", "y"}]
```

Out[∘]=



In[ ]:=
```
Animate[ParametricPlot[RotationMatrix[v].{Cos[t], 2 Sin[t]}, {t, 0, 2 π},
  PlotRange → {{-4, 4}, {-4, 4}}], {v, 0, 2 Pi}, AnimationRunning → False]
```

Out[ ]=



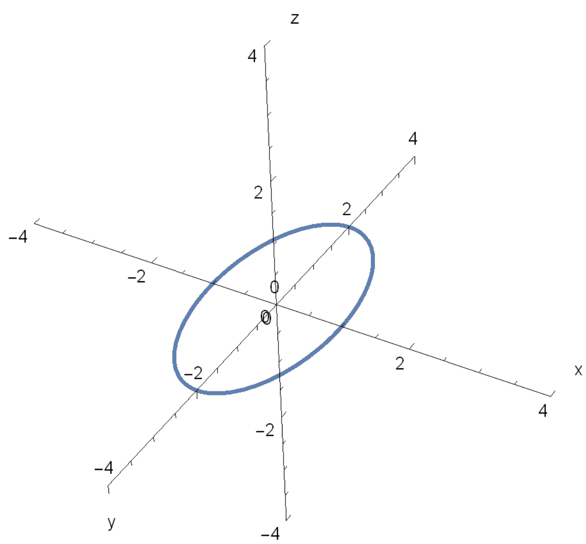

```
RotationMatrix[v] // MatrixForm (*Matriz de rotación 2D, con eje y {0, 1}*)
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} \text{Cos}[v] & -\text{Sin}[v] \\ \text{Sin}[v] & \text{Cos}[v] \end{pmatrix}$$

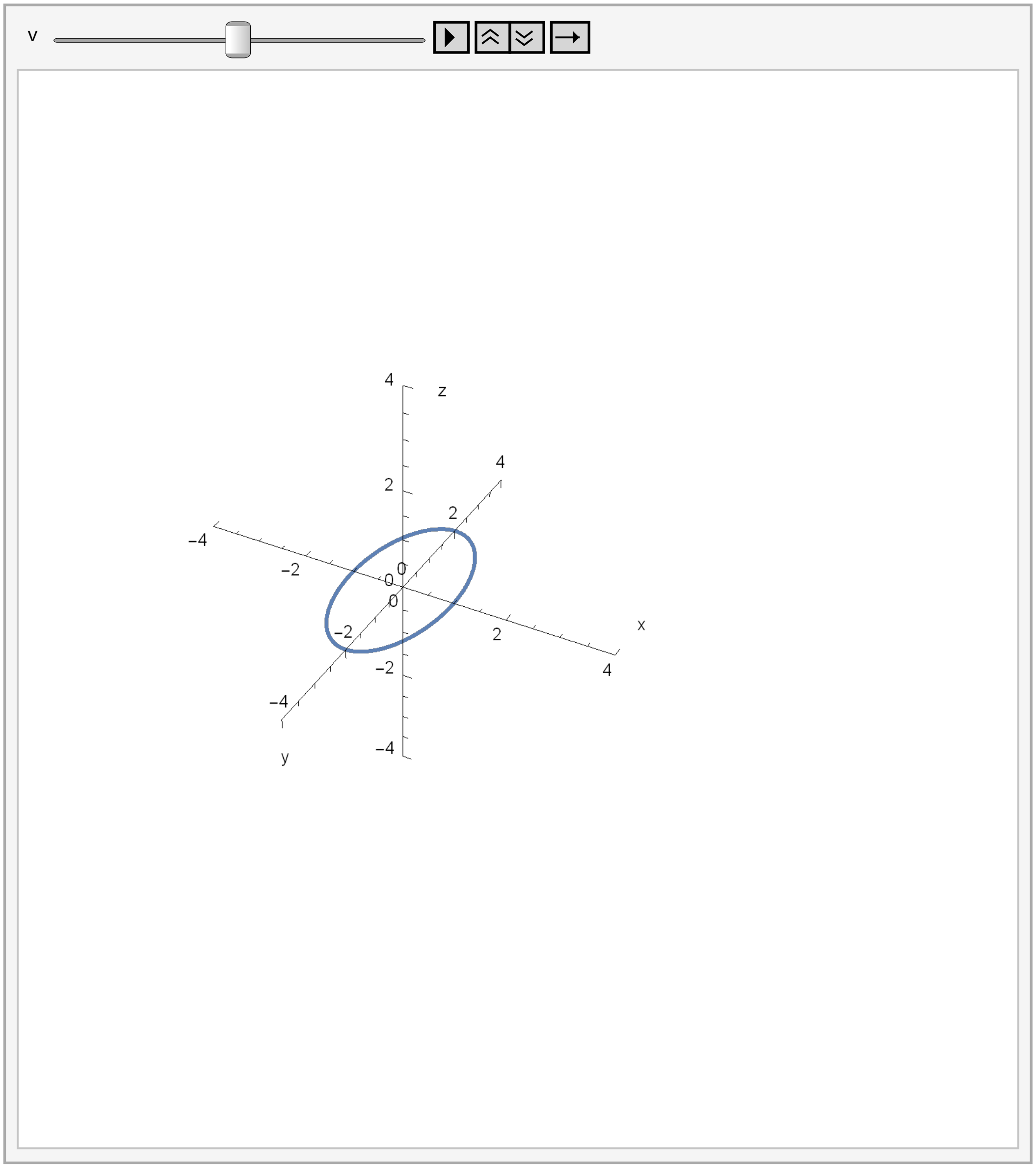
In[ ]:= ParametricPlot3D[{Cos[t], 2 Sin[t], 0}, {t, 0, 2 π}, AxesOrigin → {0, 0, 0},
AxesLabel → {"x", "y", "z"}, PlotRange → {{-4, 4}, {-4, 4}, {-4, 4}}, Boxed → False]

Out[ ]=



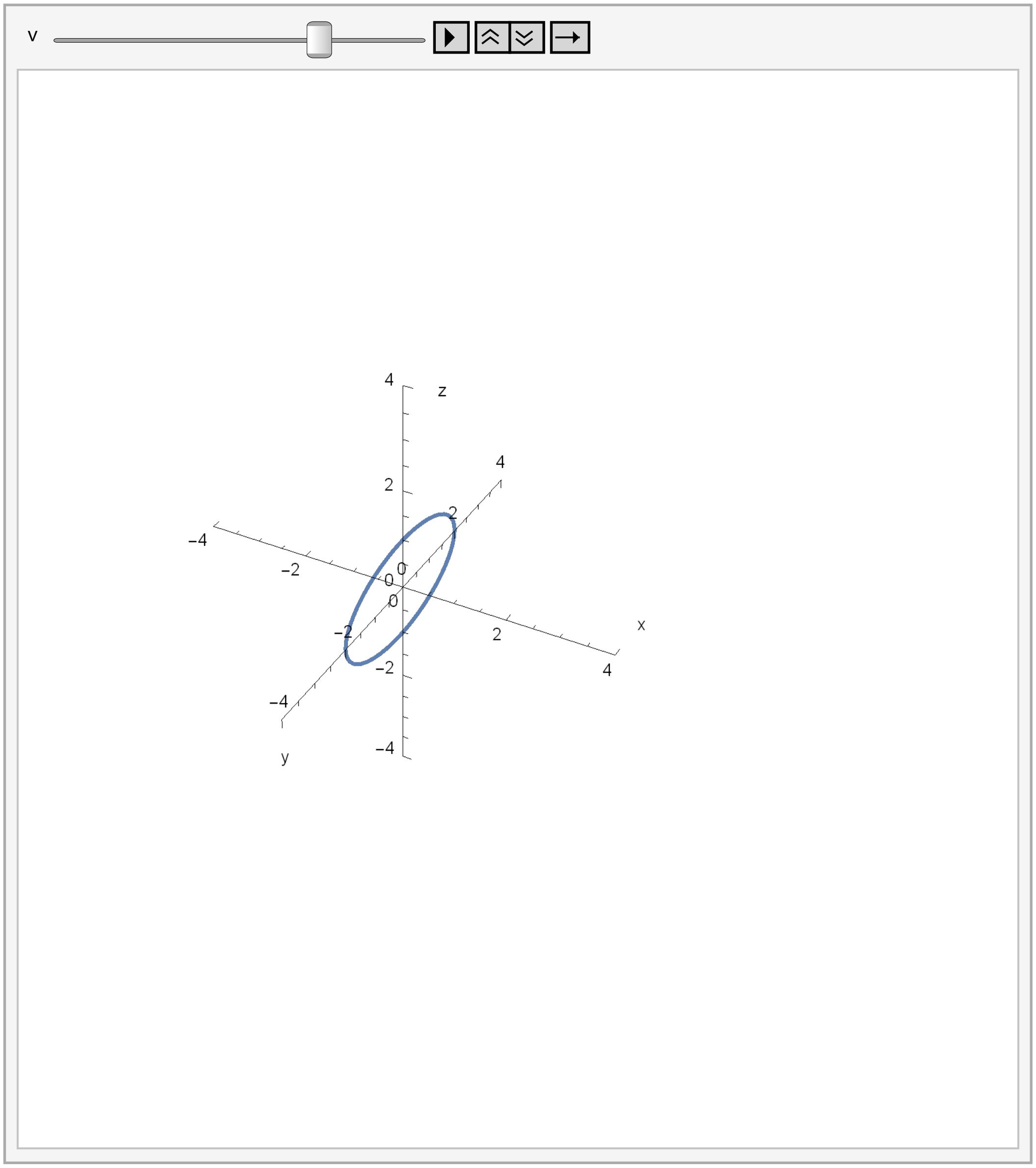
In[ ]:=
```
Animate[ParametricPlot3D[RotationMatrix[v, {0, 0, 1}].{Cos[t], 2 Sin[t], 0},
   {t, 0, 2 π}, AxesOrigin → {0, 0, 0}, AxesLabel → {"x", "y", "z"},
   PlotRange → {{-4, 4}, {-4, 4}, {-4, 4}}, Boxed → False],
  {v, 0, 2 π}, AnimationRunning → False]
```

Out[ ]=



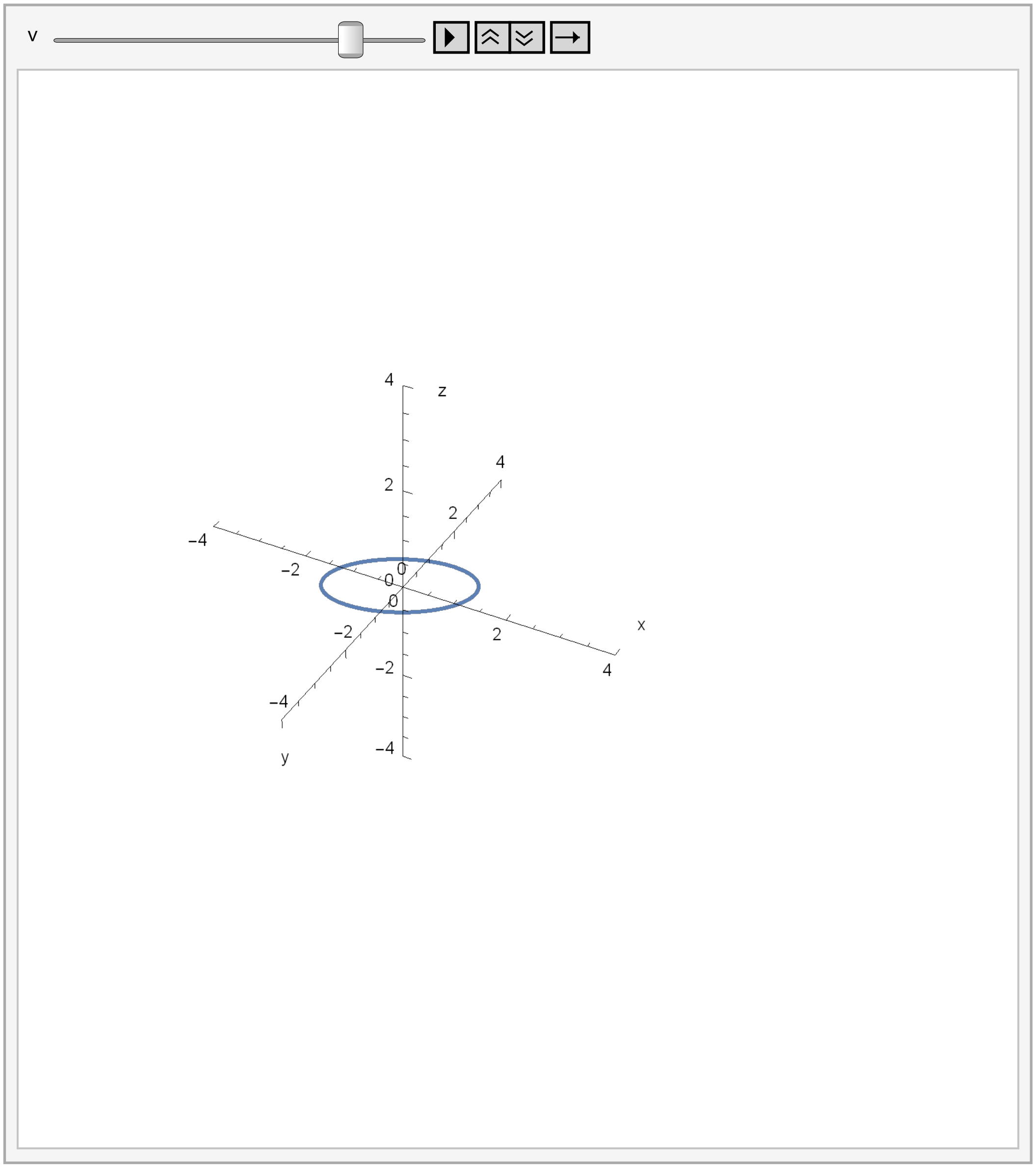
In[ ]:=
```
Animate[ParametricPlot3D[RotationMatrix[v, {0, 1, 0}].{Cos[t], 2 Sin[t], 0},
   {t, 0, 2 π}, AxesOrigin → {0, 0, 0}, AxesLabel → {"x", "y", "z"},
   PlotRange → {{-4, 4}, {-4, 4}, {-4, 4}}, Boxed → False],
  {v, 0, 2 π}, AnimationRunning → False]
```

Out[ ]=



In[∘]:=
```
Animate[ParametricPlot3D[RotationMatrix[v, {1, 1, 0}].{Cos[t], 2 Sin[t], 0},
   {t, 0, 2 π}, AxesOrigin → {0, 0, 0}, AxesLabel → {"x", "y", "z"},
   PlotRange → {{-4, 4}, {-4, 4}, {-4, 4}}, Boxed → False],
  {v, 0, 2 π}, AnimationRunning → False]
```

Out[∘]=




```
(* Si el eje de rotación deja de ser un eje coordenado {0, 0, 1},
{0, 1, 0}, {0, 0, 1}, la matriz de rotación se complica *)
RotationMatrix[v, {1, 1, 0}] // MatrixForm
 (*Matriz de rotación para el eje de rotación i + j *)
```

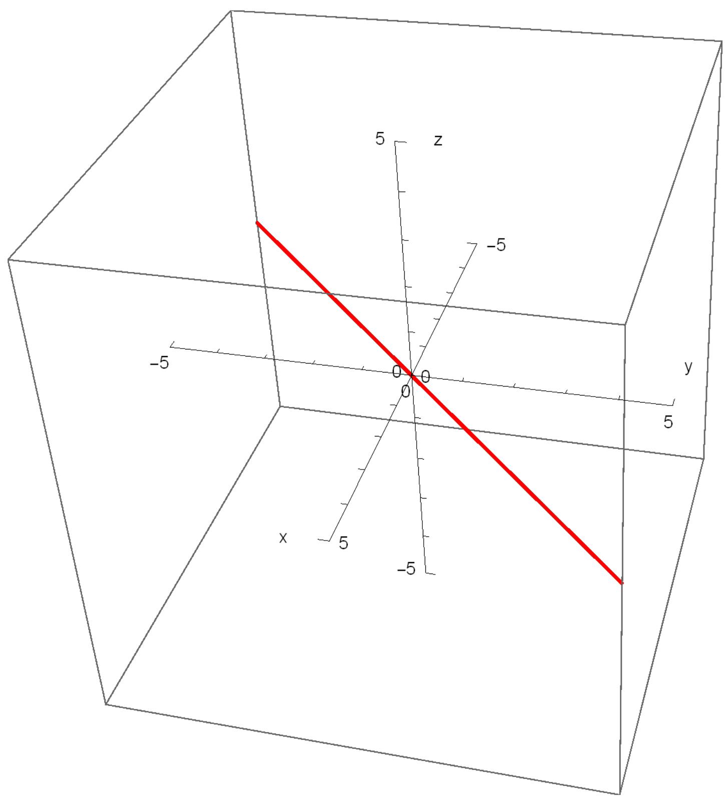
Out[∘]//MatrixForm=

$$\begin{pmatrix} \frac{1}{2}\left(1+\mathrm{Cos}[v]\right) & \frac{1}{2}\left(1-\mathrm{Cos}[v]\right) & \frac{\mathrm{Sin}[v]}{\sqrt{2}} \\ \frac{1}{2}\left(1-\mathrm{Cos}[v]\right) & \frac{1}{2}\left(1+\mathrm{Cos}[v]\right) & -\frac{\mathrm{Sin}[v]}{\sqrt{2}} \\ -\frac{\mathrm{Sin}[v]}{\sqrt{2}} & \frac{\mathrm{Sin}[v]}{\sqrt{2}} & \mathrm{Cos}[v] \end{pmatrix}$$

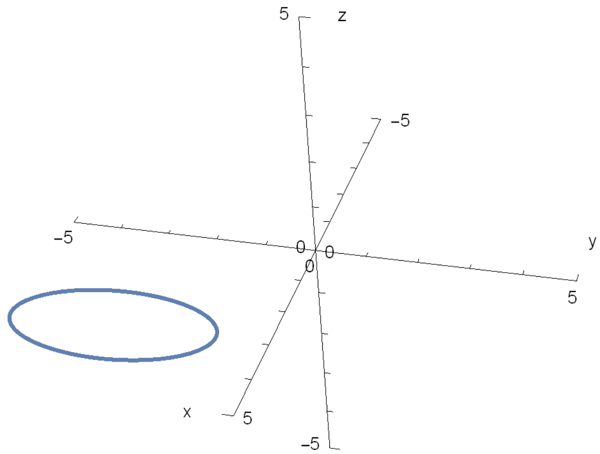
```
(*Parametrización de la recta z = y en coordenadas polares,
extensión del vector i + j *)
```

In[∘]:=
```
recta = ParametricPlot3D[{r Cos[π/4], r Cos[π/4], 0}, {r, -√50, √50},
  AxesOrigin → {0, 0, 0}, PlotRange → {{-5, 5}, {-5, 5}, {-5, 5}},
  AxesLabel → {"x", "y", "z"}, Boxed → True, PlotStyle → {Red}]
elipse = ParametricPlot3D[{ 3 + Cos[t], -3 + 2 Sin[t], 0},
  {t, 0, 2 π}, AxesOrigin → {0, 0, 0}, AxesLabel → {"x", "y", "z"},
  PlotRange → {{-5, 5}, {-5, 5}, {-5, 5}}, Boxed → False]
Show[recta, elipse]
```

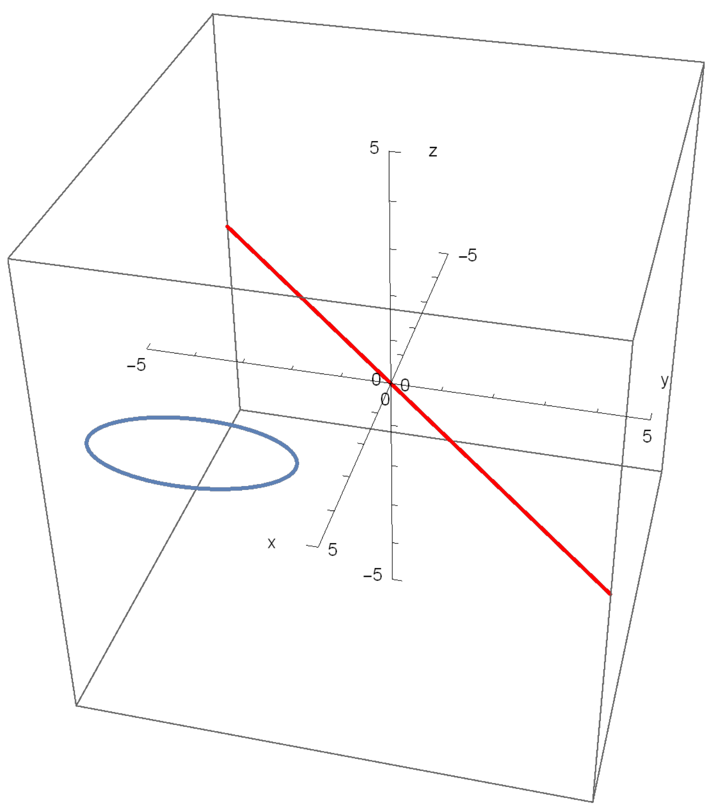
Out[∘]=



Out[ ]=



Out[ ]=



In[ ]:=
```
(* Si giramos esa elipse y mostramos la recta i + j:*)
Animate[
 Show[ParametricPlot3D[RotationMatrix[v, {1, 1, 0}].{3 + Cos[t], -3 + 2 Sin[t], 0},
   {t, 0, 2 π}, AxesOrigin → {0, 0, 0}, AxesLabel → {"x", "y", "z"},
   PlotRange → {{-5, 5}, {-5, 5}, {-5, 5}}, Boxed → False],
  ParametricPlot3D[{r Cos[π/4], r Cos[π/4], 0}, {r, -√50, √50}, AxesOrigin → {0, 0, 0},
   PlotRange → {{-5, 5}, {-5, 5}, {-5, 5}}, AxesLabel → {"x", "y", "z"},
   Boxed → True, PlotStyle → {Red}]], {v, 0, 2 π}, AnimationRunning → False]
```

Out[ ]=