




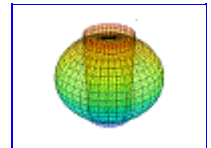
## - Exemples d'utilisation des commandes de DPGraph



**DPGraph** représente graphiquement des fonctions, des équations, les sections coniques, plans, sphères, tores, égalités et inégalités implicites, intersections de volumes, volumes d'intégration, surfaces de révolution, surfaces équipotentielles, zones de vecteur, et beaucoup plus, dans des coordonnées cartésiennes, polaires, sphériques ou de cylindrique.

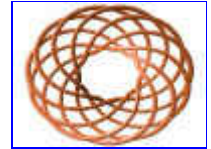
Voici quelques exemples illustrant les possibilités de DPGraph.


En vis à vis de chaque  paragraphe, figure une illustration. De manière générale, les menus déroulants donnent aisément accès à toutes les fonctionnalités du logiciel avec une aide systématique.

  En plus de ses possibilités de création de graphes implicites et vectoriels, **DPGraph** permet de créer des graphes paramétrés plans et dans l'espace et permet de les animer et de modifier leur couleur dans le temps.

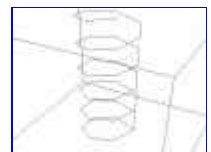



  On peut utiliser les fonctions **RECTANGULAR(X,Y,Z)**, **CYLINDRICAL(R,THETA,Z)**, ou **SPHERICAL(RHO,THETA,PHI)** pour créer des graphes paramétrés en coordonnées cartésiennes, cylindriques ou sphériques. Les variables **U** et **V** sont les noms de variables utilisés par **DPGraph** pour créer les courbes et surfaces.



 Par exemple, pour créer une hélice en coordonnées cartésiennes, on peut utiliser la commande suivante après avoir cliqué sur **EDIT** et supprimé toutes les autres commandes :

```
GRAPH3D( RECTANGULAR( COS(6*U), SIN(6*U), U ) )
```



 Pour créer, la même hélice en coordonnées cylindriques :  
GRAPH3D( CYLINDRICAL( 1, 6\*U, U ) )  
DPGraph ne tient pas compte des majuscules ou minuscules.



L'hélice est "hachée" car par défaut **DPGraph** effectue 40 calculs pour **U** et **V**. On peut créer un graphe plus régulier et plus s'affichant plus rapidement en augmentant à 200 le nombre de calculs dans la direction de **U** et en le diminuant à 0 dans celle de **V** puisque **V** n'intervient pas.

On peut de plus enlever la boîte.

```
GRAPH3D.BOX := FALSE
```

```
GRAPH3D.STEPSU := 200
```

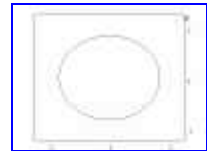
```
GRAPH3D.STEPSV := 0
```

```
GRAPH3D( CYLINDRICAL( 1, 6*U, U ) )
```

 **Astuce :**

Les graphes paramétrés plans ne sont rien d'autre que des graphes paramétrés dans l'espace observés par le dessus sans perspective.


Voici comment créer un cercle de rayon 2 en coordonnées cartésiennes. ( Remarquer que la plage de valeurs de **U** a été modifiée de 0 à  $2\pi$  car elle est de -3 à 3 par défaut :



```
GRAPH3D.VIEW := TOP
GRAPH3D.PERSPECTIVE := FALSE
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 0
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D( RECTANGULAR( 2*COS(U), 2*SIN(U) ) )
```

le même cercle en coordonnées cylindriques :

```
GRAPH3D.VIEW := TOP
GRAPH3D.PERSPECTIVE := FALSE
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 0
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D( CYLINDRICAL( 2, U ) )
```

 On peut créer une hélice plus épaisse en dessinant un tube à la place. Pour ne pas que le haut et le bas du tube ne soient coupés par le cadre, il suffit de modifier les valeurs extrêmes de Z à -3.5 et 3.5 au lieu de -3 et 3 par défaut :

```
GRAPH3D.BOX := FALSE
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 10
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D.MINIMUMZ := -3.5
GRAPH3D.MAXIMUMZ := 3.5
GRAPH3D( CYLINDRICAL( 1+0.2*COS(V), 6*U, U+0.2*SIN(V) ) )
```



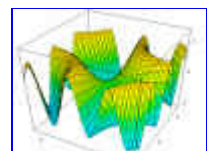
Pour agrandir ou rétrécir la figure, utiliser les touches **PageHaut** et **PageBas** du clavier. Le **pavé de flèches** permet de faire pivoter la figure et la touche se trouvant à gauche de **PageHaut** de revenir à la position initiale.


### **Astuce :**

Comme les valeurs par défaut de U et V sont aussi celles de X, Y et Z ( à savoir de -3 à 3 ) il est facile de créer  $Z=F(X,Y)$  avec U et V.

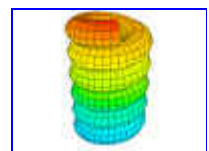
Par exemple, pour dessiner  $2\sin(X*Y)$ :

```
GRAPH3D( RECTANGULAR( U, V, 2*SIN(U*V) ) )
```




 On peut également faire varier les paramètres du graphe en temps réel en utilisant les variables A, B, C et D et en les modifiant à l'aide du menu **Scrollbar** puis en utilisant l'ascenseur. On peut ainsi régler différents aspects du graphe.

```
Par exemple :
GRAPH3D.BOX := FALSE
A := 0.2
A.MINIMUM := 0.0
```




```
A.MAXIMUM := 0.5
B := 6
B.MINIMUM := 1
B.MAXIMUM := 20
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 10
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D.MINIMUMZ := -3.5
GRAPH3D.MAXIMUMZ := 3.5
GRAPH3D( CYLINDRICAL( 1+A*COS(V), B*U, U+A*SIN(V) ) )
```

 Pour animer le tube hélicoïdal, on peut utiliser la variable TIME. Ainsi pour compresser et décompresser le tube :


```
GRAPH3D.BOX := FALSE
A := 0.2
A.MINIMUM := 0.0
A.MAXIMUM := 0.5
B := 6
B.MINIMUM := 1
B.MAXIMUM := 20
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 10
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D.MINIMUMZ := -3.5
GRAPH3D.MAXIMUMZ := 3.5
GRAPH3D( CYLINDRICAL( 1+A*COS(V), B*U, U*(0.75+SIN(TIME)/4)+A*SIN(V) ) )
```



 Pour modifier la palettes des couleurs : Par exemple, un fond NOIR, un tube ROUGE, pas de quadrillage, des éclairages et ombres (ces deux derniers sont réglables de 0 à 1).

```
GRAPH3D.BOX := FALSE
GRAPH3D.BACKGROUND := BLACK
GRAPH3D.COLOR := RED
GRAPH3D.MESH := FALSE
GRAPH3D.HIGHLIGHT := 1.0
GRAPH3D.SHADING := 1.0
A := 0.2
A.MINIMUM := 0.0
A.MAXIMUM := 0.5
B := 6
B.MINIMUM := 1
B.MAXIMUM := 20
GRAPH3D.STEPSU := 200
GRAPH3D.STEPSV := 10
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D.MINIMUMZ := -3.5
GRAPH3D.MAXIMUMZ := 3.5
GRAPH3D( CYLINDRICAL( 1+A*COS(V), B*U, U*(0.75+SIN(TIME)/4)+A*SIN(V) ) )
```

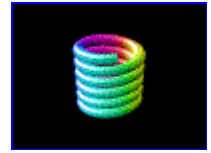



 Il est également possible de paramétrer la couleur par une expression fonction de U, V, X, Y, Z, R, THETA, RHO, PHI, A, B, C, D, ou TIME. La palette de couleurs est alors échantillonnée pour aller de Rouge à 0.0, passe magenta, blue, cyan, green, yellow, and revient à Rouge en 1.

Voici comment 0.

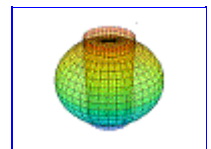
Par exemple, si l'on veut paramétrer la couleur par l'expression  $(\text{THETA}+\text{TIME})/(2*\text{PI})$

```
:  
GRAPH3D.BOX := FALSE  
GRAPH3D.BACKGROUND := BLACK  
GRAPH3D.COLOR := (THETA+TIME)/(2*PI)  
GRAPH3D.MESH := FALSE  
GRAPH3D.HIGHLIGHT := 1.0  
GRAPH3D.SHADING := 1.0  
A := 0.2  
A.MINIMUM := 0.0  
A.MAXIMUM := 0.5  
B := 6  
B.MINIMUM := 1  
B.MAXIMUM := 20  
GRAPH3D.STEPSU := 200  
GRAPH3D.STEPSV := 10  
GRAPH3D.MINIMUMV := 0  
GRAPH3D.MAXIMUMV := 2*PI  
GRAPH3D.MINIMUMZ := -3.5  
GRAPH3D.MAXIMUMZ := 3.5  
GRAPH3D( CYLINDRICAL( 1+A*COS(V), B*U, U*(0.75+SIN(TIME)/4)+A*SIN(V) ) )
```



 On peut représenter plusieurs graphes paramétrés simultanément. Il suffit de créer une liste dans la commande GRAPH3D. La liste des surfaces à construire est placée entre parenthèses et chaque équation est séparée de la suivante par une virgule. Ainsi, pour une sphère et un cylindre transparents qui pulsent : ( la transparence est réglable de 0.0 à 1.0 ) :

```
GRAPH3D.BOX := FALSE  
GRAPH3D.TRANSPARENCY := 0.5  
GRAPH3D.STEPSU := 40  
GRAPH3D.STEPSV := 20  
GRAPH3D.MINIMUMU := 0  
GRAPH3D.MAXIMUMU := 2*PI  
GRAPH3D.MINIMUMV := 0  
GRAPH3D.MAXIMUMV := PI  
GRAPH3D( ( SPHERICAL(3*COS(TIME/4),U,V),  
CYLINDRICAL(3*SIN(TIME/4),U,6*V/PI-3) ) )
```

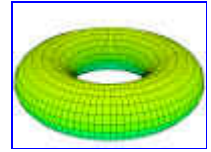



## **Création de noeuds**

Il est possible de créer des noeuds avec des graphes paramétrés. Par exemple avec des noeuds à partir de tores de type (A,B).

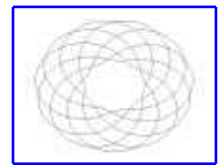
 L'idée de base est de commencer avec un tore, par exemple de rayons 2 et 0.9 :


```
GRAPH3D.BOX := FALSE
GRAPH3D.STEPSU := 100
GRAPH3D.STEPSV := 20
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D( CYLINDRICAL( 2+0.9*COS(V), U, 0.9*SIN(V) ) )
```



 Ensuite, choisir deux entiers A et B. A représente le nombre de fois que le noeud tourne autour du trou et B le nombre de fois ou il passe dans le trou. Si A et B ne sont pas premiers entre eux, le noeud sera fermé. Remplacer U par A\*U, et V par B\*U, pour créer un noeud-tore (A,B). En utilisant l'ascenseur, on peut modifier A et B pour voir différents noeuds. La fonction FLOOR permet de s'assurer que A et B sont entiers.

```
GRAPH3D.VIEW := TOP:
GRAPH3D.BOX := FALSE
GRAPH3D.VIEW := TOP
A := 7
A.MINIMUM := 0
A.MAXIMUM := 16
B := 11
B.MINIMUM := 0
B.MAXIMUM := 16
GRAPH3D.STEPSU := 400
GRAPH3D.STEPSV := 0
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D( CYLINDRICAL( 2+0.9*COS(FLOOR(B)*U), FLOOR(A)*U,
0.9*SIN(FLOOR(B)*U) ) )
```



 On peut placer un tube autour du noeud en utilisant la variable V :

```
GRAPH3D.MESH := FALSE
GRAPH3D.BOX := FALSE
GRAPH3D.VIEW := TOP
GRAPH3D.HIGHLIGHT := 0.5
GRAPH3D.SHADING := 1.0
GRAPH3D.COLOR := BROWN
A := 7
A.MINIMUM := 0
A.MAXIMUM := 16
B := 11
B.MINIMUM := 0
B.MAXIMUM := 16
GRAPH3D.STEPSU := 400
GRAPH3D.STEPSV := 20
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := 2*PI
GRAPH3D( CYLINDRICAL( 2+0.9*COS(FLOOR(B)*U)+0.1*COS(V), FLOOR(A)*U,
0.9*SIN(FLOOR(B)*U)+0.1*SIN(V) ) )
```



 D'autres possibilités sont accessibles avec les noeuds-tore.

Par exemple, désactiver la vue TOP et utiliser une coloration fonction du temps TIME pour créer un effet dynamique de déplacement.

Remarquer que la bande mobile a un aspect plat contrairement au reste du tube ; V varie dans une plage de 0 à PI de 1 en 1.

```
GRAPH3D.MESH := FALSE
GRAPH3D.BOX := FALSE
GRAPH3D.BACKGROUND := BLACK
GRAPH3D.HIGHLIGHT := 0.5
GRAPH3D.SHADING := 1.0
GRAPH3D.COLOR := 0.1 + 0.6*FLOOR( 2*PI/5.8*( U/(2*PI)+TIME/16 -
FLOOR(U/(2*PI)+TIME/16) ) )
A := 3
A.MINIMUM := 0
A.MAXIMUM := 16
B := 5
B.MINIMUM := 0
B.MAXIMUM := 16
GRAPH3D.STEPSU := 400
GRAPH3D.STEPSV := 1
GRAPH3D.MINIMUMU := 0
GRAPH3D.MAXIMUMU := 2*PI
GRAPH3D.MINIMUMV := 0
GRAPH3D.MAXIMUMV := PI
GRAPH3D( CYLINDRICAL( 2+0.9*COS(FLOOR(B)*U)+0.1*COS(V), FLOOR(A)*U,
0.9*SIN(FLOOR(B)*U)+0.1*SIN(V) ) )
```

