

Documentation du package geomana3d.mac pour Maxima

Version 2.5 du 18 janvier 2026

Auteur : Michel Gosse, michel.gosse@free.fr

Présentation

Ce package pour Maxima a pour objectif de proposer des fonctions et macros permettant de traiter les exercices de géométrie analytique en 3D des programmes de lycée. On définit les différents objets nécessaires : points, vecteurs, plans, équation cartésienne de plan, équations paramétriques de droites, puis on choisit les commandes à appliquer pour résoudre le problème posé.

On se place bien sur dans un repère orthonormé de l'espace afin de pouvoir travailler directement sur les coordonnées pour tous les problèmes de géométrie analytique en lycée.

Les intérêts pédagogiques

- L'étudiant est placé dans une démarche active de recherche. Il peut tester différentes solutions en appliquant aux objets de l'espace les commandes pré-définies.
- L'étudiant est déchargé des calculs fastidieux et peut se concentrer sur les raisonnements géométriques à mettre en oeuvre. Bien sur, il peut utiliser l'outil comme vérificateur de ses propres calculs.
- Les notations utilisées pour les objets points, vecteurs, plans, droites sont identiques à celles en vigueur dans les programmes de lycée.
- Lors de l'utilisation d'une commande, des commentaires sur la méthode utilisée, ou des explications sur les calculs effectués, sont indiqués dans la sortie de Maxima. Cela permet de comprendre les raisonnements géométriques ayant abouti aux résultats.

De plus ces commentaires et indications sont personnalisables par l'enseignant en fonction du vécu de ses élèves et des indications qu'il souhaite leur communiquer (voir annexe technique en fin du document).

D'autre part, il est possible de désactiver l'impression de ces messages en changeant simplement la valeur de la variable `bavard`.

- Maxima est en capacité de représenter des objets de l'espace, ce qui peut aider à la compréhension du problème et aider à mieux voir dans l'espace. En particulier il est possible de choisir de représenter les objets de l'espace dans une fenêtre graphique 3D de gnuplot, ce qui permet de manipuler à la souris ces objets de l'espace.
- Pour l'enseignant, ce package lui permet de générer et de tester rapidement des configurations de l'espace pour construire des exercices destinés à ses étudiants, sans avoir à effectuer tous les calculs nécessaires qui sont délégués à Maxima.

Installation

Une fois le package téléchargé, deux possibilités :

1. soit on copie le fichier dans un dossier connu du path de Maxima, et il suffit de le charger avec la commande `load(geomana3d)` ;
2. ou dans wxMaxima, on fait Fichier -> Charger un paquetage et on choisit le fichier `geomana3d.mac`.

Commandes du package

La commande `info_package_geom3d()` donne les informations sur le package et les commandes disponibles.



Astuce

Par défaut, Maxima affiche en sortie le dernier résultat calculé lors de l'exécution d'une commande (lorsque cette commande est suivie de ;). Pour supprimer cette sortie, quelquefois inutile avec les commandes de ce package, il suffit de terminer la commande par \$ à la place de ;.

Les objets

Objet	Syntaxe	Commentaire
Point	A :[1,-2,3]	Un point est défini par ses coordonnées. On peut indifféremment utiliser dans les commandes soit le nom, soit les coordonnées.
Vecteur	n :[0,1,3]	De même, n ou [0,1,3] sont équivalents dans les calculs.
Equation de plan	eqp :2*x+3*y-z+1=0	On saisit l'équation cartésienne selon la syntaxe habituelle. On peut utiliser soit le nom soit l'équation de manière identique dans les commandes.
Equations de droite	eqd :[x=1+2*t,y=-t,z=3+t,t]	On doit donner le système des 3 équations paramétriques suivi du paramètre choisi pour décrire la droite. Le nom ou la liste des 4 éléments s'utilise indifféremment.

Précisions concernant les vecteurs

Pour un logiciel de calcul formel tel que Maxima, un point ou un vecteur est défini de la même manière par ses coordonnées. En conséquence, si O est l'origine du repère, on aura par exemple :

- le point A :[1,-2,3]
- le vecteur \overrightarrow{OA} :[1,-2,3]

Selon le contexte, on peut donc interpréter les coordonnées précédentes soit comme celles d'un point, soit comme celles d'un vecteur. Le calcul vectoriel devient alors très simple si l'on comprend cette dualité :

► Comme $\overrightarrow{AB} = \overrightarrow{OB} - \overrightarrow{OA}$, on peut donc calculer ce vecteur par B-A avec Maxima, puisque \overrightarrow{OA} a les mêmes coordonnées que A sachant O a des coordonnées nulles dès lors que les points A et B ont été définis.

► Autre exemple, pour calculer le vecteur \overrightarrow{OM} défini par :

$$\overrightarrow{OM} = \overrightarrow{OA} + k \cdot \overrightarrow{BC}$$

il suffit de faire avec Maxima A+k*(C-B).

Volet pédagogique

Afin de ne pas favoriser la confusion point/vecteur pour un étudiant, il peut être intéressant d'imposer de définir systématiquement un vecteur en rappelant sa nature. Par exemple, on peut imaginer mettre v devant chaque objet vecteur :

- vAB :B-A est la notation pour définir le vecteur AB
- en posant vBC :C-B, on se rapproche des notations habituelles pour le calcul vectoriel.

Le calcul de $\overrightarrow{AB} + 2 \cdot \overrightarrow{BC}$ s'écrit alors $vAB + 2 \cdot vBC$, expression qui prête moins à confusion que B-A+2*(C-B).

Les commandes de calcul

<i>Commande</i>	<i>Syntaxe</i>	<i>Commentaire</i>
genere_point()	M :genere_point()	Le point M est généré aléatoirement. Chacune de ses coordonnées est comprise entre -10 et 10.
point_droite(eq,t0)	M :point_droite(eq,t0)	Renvoie le point de paramètre t0 de la droite dont les équations sont eq (voir définition d'une droite au paragraphe précédent). On peut l'affecter à une variable.
produit_vectoriel(u,v)	pv :produit_vectoriel(u,v)	Cette commande renvoie le produit vectoriel des deux vecteurs u et v et l'affecte à la variable pv.
equation_plan(A,B,C)	eq :equation_plan(A,B,C)	Cette commande renvoie une équation cartésienne du plan ABC. On peut l'affecter à la variable eq selon l'exemple proposé.
equation_plan2(n,M)	eq :equation_plan2(n,M)	Cette commande renvoie une équation cartésienne du plan de vecteur normal n passant par M. On peut l'affecter à la variable eq selon l'exemple proposé.
equation_droite(M,u,t)	eqd :equation_droite(M,u,t)	Cette commande renvoie un système d'équations pour la droite passant par M dirigée par le vecteur u, en prenant t comme paramètre. On peut l'affecter à la variable eqd selon l'exemple proposé.
vecteur_directeur_droite(eq)	u :vecteur_directeur_droite(eq)	Cette commande renvoie un vecteur directeur de la droite d'équations paramétriques eqd. On peut l'affecter à la variable vecteur u selon l'exemple proposé.
normal(eq)	n :normal(eq)	Cette commande renvoie un vecteur normal du plan dont une équation est eq. On peut l'affecter à la variable n selon l'exemple proposé.
norme(A,B)	long :norme(A,B)	Renvoie la longueur du segment [A,B] et l'affecte à la variable long si on le souhaite.
norme2(A,B)	long :norme2(A,B)	Renvoie la longueur du segment [A,B] mais sans aucun commentaire (utile pour les calculs).
produit_scalaire(u,v)	ps :produit_scalaire(u,v)	Renvoie le produit scalaire des deux vecteurs u et v. On peut l'affecter à la variable ps par exemple si on le souhaite.
normev(u)	norm :normev(u)	Renvoie la norme du vecteur u et l'affecte à la variable norm si on le souhaite.
aire_triangle(X,Y,Z)	a1 :aire_triangle(X,Y,Z)	Renvoie l'aire du triangle XYZ et l'affecte à la variable a1 si on le souhaite.
proj_ortho(M,eq)	H :proj_ortho(M,eq)	Renvoie les coordonnées du point projection orthogonale de M sur le plan d'équation eq. Le paramètre eq doit être une équation cartésienne de plan. On peut affecter le résultat au point H comme montré.
intersection_plan_droite(d,eq)	P :intersection_plan_droite(d,eq)	Renvoie les coordonnées du point d'intersection entre la droite d et le plan eq (s'il existe). La droite est définie par ses équations paramétriques (voir objets) et le plan par son équation cartésienne. On peut affecter le résultat à une variable.
intersection_droite_droite(d1,d2)	P :intersection_droite_droite(d1,d2)	Teste si les deux droites d1 et d2 définies par leurs équations paramétriques sont sécantes. Renvoie les coordonnées du point d'intersection s'il existe, que l'on peut affecter à un point dans ce cas.

Les commandes de test

Commande	Commentaire
is_plan(A,B,C)	teste si les trois points forment un plan. Si c'est le cas, la commande renvoie un vecteur normal et une équation cartésienne ainsi qu'une représentation graphique des points et du plan. Si les points sont alignés, le programme l'indique et les dessine.
is_plan2(A,B,C)	il s'agit de la même commande que précédemment, mais qui cette fois dessine les objets graphiques dans une fenêtre gnuplot 3D. Dans cette fenêtre, les objets de l'espace se manipulent à la souris.
is_normal(n,A,B,C)	teste si le vecteur n est normal au plan ABC.
is_point_plan_eq(M,eq)	teste si le point M appartient au plan dont on donne une équation eq.
is_point_plan_pts(M,A,B,C)	teste si le point M appartient au plan ABC. Il faut évidemment savoir ou vérifier dans ce cas que les 3 points ABC forment un plan (par exemple avec la commande <code>is_plan(A,B,C)</code>).
compare_equation_plan(eq1, eq2)	teste si les deux équations de plan eq1 et eq2 correspondent à un même plan. On normalise les équations pour les comparer.
is_rectangle(A, B, C)	teste si le triangle ABC est rectangle.
is_colineaire(u,v)	teste si les vecteurs u et v sont colinéaires.
is_point_droite(M,d)	teste si le point M appartient à la droite d définie par ses équations paramétriques.

Annexe technique

- Par défaut, lors du chargement du package, on a `bavard:1`, ce qui initialise cette variable `bavard` à 1. Dans ce cas, tous les commentaires sont imprimés. Pour changer ce comportement, il suffit d'entrer comme commande Maxima `bavard:0` dans la feuille de calcul (et réciproquement).
- Pour modifier et personnaliser les messages et commentaires, il suffit d'ouvrir avec un éditeur de textes le package `geomana3d.mac`. Dans les commandes concernées, on trouve une variable `message` :
`message:"message ou commentaire à afficher"` qu'il suffit de modifier.
Les variables `message1` et `message2` jouent le même rôle pour des messages qui doivent s'afficher sur plusieurs lignes.
- Les graphiques utilisent le package `draw` de Maxima, qui est chargé automatiquement dans les dernières versions de Maxima. Si les graphiques n'apparaissent pas, alors il faut charger manuellement ce package via la commande `load(draw)` au début du notebook.
- Maxima ne sait pas récupérer le nom d'une variable passée en paramètre d'une fonction, il la remplace par sa valeur. Ce comportement explique pourquoi il nous faut introduire un nom ou une notation différente dans les commentaires affichés par une fonction ou lors de l'affichage des résultats.
- Pour éviter les bugs ou résultats erronés, il faut impérativement respecter l'écriture attendue pour les objets (points, vecteurs, plans et droites). En ce qui concerne les commandes, il faut aussi bien vérifier l'ordre des paramètres, l'objet attendu et leur nombre.