Utilisation de logiciels

Pour la session 2013, le jury attendra un usage beaucoup plus systématique des moyens informatiques mis à disposition pour les sujets qui s'y prêtent...

Geogebra

Remarques générales

Distinguons deux types de problèmes de géométrie qui appellent naturellement l'utilisation de Geogebra :

1. Les problèmes de construction. La construction de la figure est alors un point essentiel de l'exercice. On doit alors montrer les étapes, et la figure finale doit pouvoir être modifiée (en particulier pour étudier divers cas particuliers.
2. Les problèmes de lieux géométriques, faisant ou non appel à des transformations. On pourra utiliser la "trace" ou l'outil "lieu".

Il faut faire attention à ce que la figure ne soit pas trop pauvre, que le logiciel apporte quelque chose (à justifier).
Ne pas oublier d'utiliser les entrées algébriques.

Exemples

Voici quelques exemples :

1. Tangentes communes à deux cercles. Construire les tangentes issues d'un point à une parabole, à une ellipse (dans le second cas, utilise la notion de cercle directeur). Prolongement (éventuellement sans logiciel) : quand ces tangentes sont-elles orthogonales ? (341" se prêtent admirablement à l'usage d'un logiciel de géométrie dynamique ; il convient cependant que le logiciel serve à autre chose qu'à voir et faire voir une solution... RJ2010", 332, 334 "un bon logiciel de géométrie dynamique permet de visualiser la situation proposée, RJ2010)
2. Étude de la cissoïde, application à la trisection de l'angle, lieu des centres de courbure d'une ellipse et enveloppe des normales. (335)
3. Étude d'une application affine non standard (transvection : x'=x+y, y'=y), image d'un cercle ou d'une conique.
4. Utilisation en analyse : Par exemple, pour programmer une suite récurrente $u_n + 1 = f(u_n)$: la fonction f est entrée, $U0$ est un point sur l'axe des x, n est entré (ou est un curseur).

```
L=ItérationListe[f, x(U0), n]
L2=Séquence[Segment[(Elément[L,i],Elément[L,i]),(Elément[L,i],Elément[L,i+1])],i,2,n]
L3=Séquence[Segment[(Elément[L,i],Elément[L,i+1]),(Elément[L,i+1],Elément[L,i+1])],i,1,n]
```

Exercice 1
On se place dans le plan affine euclidien. Soit D_1, D_2 deux droites sécantes en un point O et A un point. Construire un cercle passant par A et tangent à chacune des droites. On pourra utiliser la méthode d'affaiblissement des contraintes : construire d'abord un cercle tangent aux deux droites, puis utiliser une transformation.

Question supplémentaire : combien y a t-il de solution ? Cas particuliers ?

Reprendre le problème quand les droites sont parallèles.

(leçons : 328,354)

Exercice 2, droites de Simson et de Steiner
Soit ABC un (vrai) triangle du plan affine euclidien. Si M est un point du plan, on note P, Q et R ses projections respectives sur les côtés (AB), (BC) et (CA).

1. Montrer que P, Q et R sont alignés (sur la "droite de Simson" de M) si et seulement si M est sur le cercle circonscrit au triangle ABC.
2. On note P', Q' et R' les symétriques de M par rapport aux côtés. Montrer qu'ils sont alignés si et seulement si M est sur le cercle circonscrit à ABC, et qu'alors la droite $P'Q'R'$ passe par H orthocentre du triangle (droite de Seiner).
3. (*) Déterminer l'enveloppe de la droite de Simson.

(ref. Ladegaillerie, p. 344, 345)

Exercice 3 D est la droite d'équation $y = -1$, F le point de coordonnées $(0, 1)$ (repère orthonormé). Si M est un point du plan, on note H sa projection sur D, et on note $\mathscr{P}$ l'ensemble des points M tels que $MF = MH$.

1. Prenant un point H quelconque sur D, montrer qu'on peut construire un point M dont la projection et H et qui appartient à $\mathscr{P}$.
2. Déterminer une équation de $\mathscr{P}$.

3. Montrer que la tangente à $\mathscr{P}$ en M est la médiatrice de $[FH]$.
4. Proposer une construction géométrique des tangentes à $\mathscr{P}$ qui passent par un point P du plan. Quand ces tangentes sont-elles orthogonales ?

Instructions :

```
d:y=-1  F=(0,1)  H=point(d) ...
```

(leçons 334, 342, 330, ref. Ladegaillerie, exos, p.279 pour la dernière question)

Exercice 4

Soit a un nombre réel et $\mathscr{C}_a$ la courbe définie par :

$$\begin{cases} x(t) = t - a\sin t \\ y(t) = 1 - a\cos t \end{cases}$$

1. Étudier la courbe $\mathscr{C}_1$. Donner en particulier l'allure en les points singuliers.
2. Calculer la longueur de la courbe entre les points de paramètre 0 et 2π.
3. Déterminer l'allure de la courbe $\mathscr{C}_a$ au voisinage de 0 suivant les valeurs de a. On pourra utiliser un logiciel pour tracer les différentes courbes possibles.

(Ramis L1 p.537.) La courbe $\mathscr{C}_1$ est très classique, c'est la cycloïde. Les autres sont les courbes cycloïdales. On pourra faire le lien avec un cercle de rayon 1 qui roule sur l'axe des x.

```
d:y=0  A=point(d) O=A+(0,1) c=Cercle[O,1] M=Rotation[A,-x(A),O]
```

Exercice 5

Un problème de construction et de lieu. On essaiera de bien employer la méthode d'analyse-synthèse.

1. Soit $\mathscr{C}$ un cercle, $\mathscr{D}$ une droite extérieure au cercle. Si on choisit un point M sur le cercle, construire un cercle Γ tangent à la fois à la droite $\mathscr{D}$ et au cercle $\mathscr{C}$. Y a t-il une seule solution ? On pourra utiliser une homothétie transformant Γ en $\mathscr{C}$ et chercher quelle peut être l'image de $\mathscr{D}$.
2. Déterminer l'ensemble des centres de ces cercles, lorsque M varie. En déduire que le problème se ramène à l'intersection d'une parabole avec une droite passant par le foyer.
3. Examiner le cas où le cercle et la droite sont tangents, sont sécants... (cf. Ladegaillerie, exercices sur le cercle).

Utilisation de Xcas

Généralités : difficile, en peu de temps, de maîtriser la programmation. On pourra se contenter d'illustrer des exercices, avec une illustration qui utilise la puissance de l'ordinateur (sinon, ce n'est pas la peine...)

Algèbre arithmétique

1. On connaît le théorème de Bezout, l'algorithme d'Euclide. Le théorème de Bezout a une démonstration "abstraite" (laquelle ?), mais peut être aussi démontré de façon "algorithmique". Voici un énoncé possible (cf ; par exemple Pearson MPSI p. 584)
   (a) Soient $a > b > 0$ deux entiers. Montrer qu'il existe deux suites finies l'une strictement décroissante $(r_i)_{i=0..n}$ l'autre (q_i) telles que
   $$\begin{cases} r_0 = a, r_1 = b \\ r_{i+1} = r_{i-1} - q_i r_i \qquad \text{pour tout } i = 1..n \\ r_{n+1} = 0 \end{cases}$$
   Montrer alors que r_n (le dernier reste non nul) est le p.g.c.d. de a et b.
   (b) Avec les mêmes notations, on définit les suites (λ_i) et (μ_i) par
   $$\begin{cases} \lambda_0 = 1,\ \mu_0 = 0 \\ \lambda_1 = 0,\ \mu_1 = 1 \\ \lambda_{i+1} = \lambda_{i-1} - q_i\lambda_i \text{ pour tout } i = 1..n \\ \mu_{i+1} = \mu_{i-1} - q_i\mu_i \text{ pour tout } i = 1..n \end{cases}$$
   Montrer que $r_i = \lambda_i a + \mu_i b$. Qu'obtient-on pour $i = n$ et pour $i = n+1$?

(c) Mettre en œuvre ces algorithmes avec un tableur : il y a seulement deux formules à mettre, et à recopier. On utilisera quatre lignes : la première contiendra les quotients, les trois suivantes les suites (r_n), (λ_n) et (μ_n). Avec un logiciel : écrire l'algorithme de façon "informelle", le mettre dans un langage (Xcas, Maxima...). Tester avec, par exemple $a = 1445$, $b = 1311$.

Une référence possible, Ramis-Warusfel L1, p. 434. Voir aussi la documentation de Xcas.

```
fonction Bezout(A,B)
          local U,V,W,X,S,T,Q,R
    1->U 0->V 0->W 1->X
tantque B $\neq$ 0 faire
          A mod B->R
          E(A/B)->Q
          U-W*Q->S
          V-X*Q->T
          B->A W->U X->V
          R->B S->W T->X
ftantque
        retourne {U, V, A}
ffonction
```

Voici une version Python

```
 #algorithme d'Euclide etendu
def pgcdex(a,b):
    u,w=1,0;v,x=0,1
    while b<>0:
        q,r= a/b ,a%b
        u,w=w,u-w*q
        v,x=x,v-x*q
        a,b=b,r
    return(u,v,a)

print(pgcdex(114,121))
```

2. Polynômes : **Les polynômes de Tchebychef** Voir Pearson MP, p641, qui donne la procedure Maple.

```
tchebychev:=proc(n)
  assume(t,real);
  subs(cos(t)=X,re(evalc(simplify(expand((cos(t)+I*sin(t))^n)))));
  end;
```

On peut également (en faisant l'exercice), calculer par une formule de récurrence, faire tracer les graphes, déterminer les racines...

3. Nombres premiers : nous avons vu le test de Lucas-Lehmer. Une suggestion : - un exercice qui démontre le petit théorème de Fermat (éventuellement par plusieurs méthodes) et une illustration de la "réciproque" : on appelle pseudo-premier de base 2 tout nombre premier avec 2 qui n'est pas premier mais qui vérifie néanmoins :

$$2^{p-1} \equiv -1 \pmod{p}$$

Donner la liste des pseudo premier de base 2 et inférieur à 1000 (à 10000). Éventuellement, chercher les pseudo-premiers de base 3. Remarque, il existe des non premiers qui résistent à tous les tests de Fermat (nombres de Carmichaël). Référence : Gourdon, p. 35

```
Liste:=[];
for n from 1 to 2000 by 2 do
            if powmod(2,n-1,n)=1 then Liste:=append(Liste,n)
            fi
                        od;
```

(on peut tout taper sur une ligne). Il est possible alors de "filtrer" les non premiers (éventuellement on les mettra dans une autre liste) :

```
for n from 1 to length(Liste)-1 do
            if not(isprime(Liste[n]))
               then print(Liste[n])
            fi
                         od;
```

On trouve

$$341, 561, 645, 1105, 1387, 1729, 1905$$

à comparer avec la taille de la liste : 310.

4. On peut implémenter les quaternions, à l'aide des matrices (voir Pearson MP, p. 743) On pourra écrire une procédure qui au quaternion $a + bi + cj + dk$ associe la matrice correspondante, calculer le déterminant, l'inverse... et montrer que l'ensemble des entiers qui s'écrivent comme somme de 4 carrés est stable pour le produit : si $N = 1^2 + 2^2 + 3^2 + 4^2$ et $M = 5^2 + 6^2 + 7^2 + 8^2$, montrer que MN est aussi somme de quatre carrés.
5. L'algorithme de Gauss est très important. Regardons comment on peut « à la main » inverser une matrice. Ce n'est pas un programme mais une suite de commande, il est extrait de « Guide de calcul », de Guillaume Connan et Stéphane Grognet (dunod). Il suffit d'utiliser les fonctions mRow et mRowAdd.
6. Voici un exemple d'algorithme simple qui calcule le polynôme minimal de ...presque toutes les matrices. (ref. doc Xcas)

```
polmin:=proc(A) local k,n,vk,B,noyau;
 n:=size(A);
 vk:=ranm(n);
 B:=[vk];
 for k from 2 to n+1 do
     vk:=A*vk;B[k]:=vk;
 od;
 B:=tran(revlist(B));
 noyau:=ker(B);
 return(noyau[1]);
 end;
```

A utiliser avec une matrice "aléatoire" :

```
randmatrix(3,3)
```

et à comparer avec

```
pmin(A)
```

L'idée est qu'on choisi un vecteur "au hasard" et qu'on regarde ses images successives par la matrice A. Si on trouve une relation de liaison, ce sera (souvent) celle donnée par le polynôme minimal. Pour plus de précision, voir la notion de polynôme minimal ponctuel. Par exemple, dans Gourdon, exo 3 p. 177, ou Goblot (Algèbre linéaire, p. 165, ancienne édition).

7. Suite arithmético-géométrique : Soit A une matrice telle que $\|A\| < 1$ pour une norme matricielle. On se donne une suite de vecteurs (u_k) telle que

$$u_{k+1} = Au_k + v$$

où v est un vecteur constant. Montrer que la suite (u_k) converge vers une limite indépendante de u_0. Il y a une application à certaines résolutions de systèmes : Petits problèmes de mathématiques appliquées et de modélisation (Bidégaray chez Springer)

Analyse

1. Suite récurrente. Tout est préprogrammé...

```
plotseq(3.71*x*(1-x),x=[0.21,0,1],100);
```

Dans cet exemple, il est intéressant de jouer sur le coefficient 3.71. Voir un exemple où on joue aussi sur la fonction : XENS, 2.19, p.85. ou le suivant.

2. Approximation de fonctions (417) On peut y mettre un calcul approché d'une intégrale (RJ) (voir aussi 420). L'algorithme de la méthode de Simpson est très simple à mettre en oeuvre. Voir par exemple de Biasi (p. 138). Elle est basée sur

$$\int_a^b f(x)dx \approx \int_a^b P(x)dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]$$

```
simpson:=proc(n,a,b) local s,k,d;s:=0;d:=(b-a)/n;
                for k from 0 to n-1 do
                s:=s+f(a+k*d)+4*f(a+k*d+d/2)+f(a+(k+1)*d);
                end_for;
            return evalf(s*d/6)
        end;
```

Toujours dans ce registre, on peut utiliser Lagrange et observer l'écart entre une fonction et son polynôme de Lagrange. On utilisera par exemple :

```
f:=x->1/(1+x^2);
L(x):=Lagrange([0,1/4,1/2,3/4,1],[f(0),f(1/4),f(1/2),f(3/4),f(1)]);
plotfunc([f(x),L(x)],x=-2..2);
```

(remarquer les variantes pour définir une fonction) ;

3. On peut faire calculer les coefficients de Fourier d'une fonction, par

   ```
   fourier_an(f(x),x,2*Pi,n,-Pi)
   ```

   où 2*Pi est la période, et -Pi le point de départ de l'intégration.

4. Equadif : on peut illustrer une équadif par :

   ```
   interactive_plotode(-y+x+1,[x=-4..4,y]) ;
   ```

   qui dessine le champ des tangentes. On peut obtenir des courbes intégrales par simple "clic".
   Il est possible également de résoudre certaines équations différentielles.

   ```
   desolve(diff(y(t),t$2)+2*diff(y(t),t)+y(t),y(t));
   ```

   donne comme résultat

   ```
   (t*c_0+t*c_1+c_0)*exp(-t)
   ```

   Pour les systèmes différentiels, on peut faire dessiner le champ des tangentes :

   ```
   plotfield([-y+x,x+2*y],[x=-1..1,y=-1..1]);
   ```

   Il est possible également de programmer la méthode d'Euler.