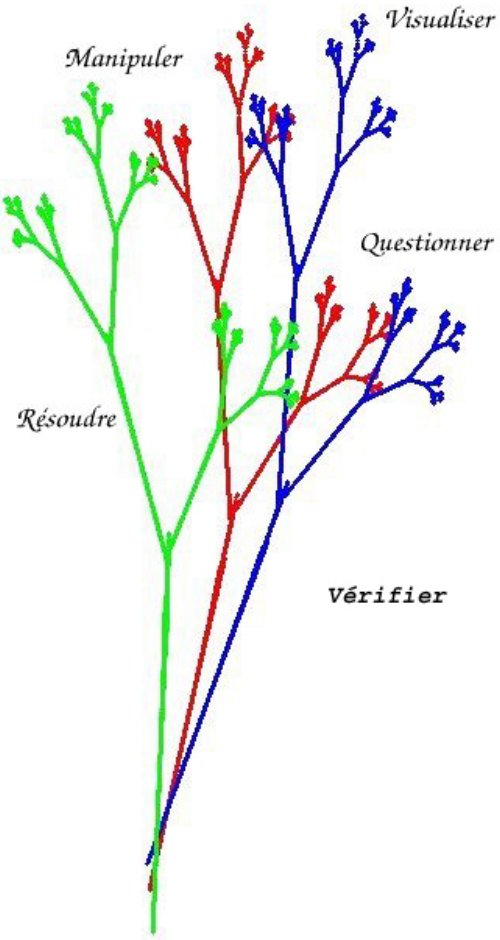
Le calcul formel dans l'enseignement des mathématiques

Michel Mizony

Lille, Avril 2005



mizony@univ-lyon1.fr

Résumé

Il existe deux sortes de logiciels de calcul **symbolique** qui bousculent nos pratiques : les logiciels de géométrie **dynamique** et ceux de **calcul formel**.

L'utilisation du calcul formel renouvelle profondément notre rapport à l'enseignement des mathématiques. Bien sûr, les exemples que nous donnerons sont des produits manufacturés qui masquent les nombreux allers-retours entre réflexion théorique et mise en pratique. Comme le calcul formel permet de par son langage une grande interaction entre la programmation et les mathématiques, il remet à l'honneur l'aspect **expérimental** de celles-ci, mais aussi fait apparaître un **"triangle didactique"**.

Mais le calcul formel change aussi nos pratiques de recherche (ce qui est une autre histoire) et donc, à terme, une évolution dans la production de mathématiques et de leur enseignement.

Introduction

Les exemples seront donnés avec le logiciel "Maple", et sont issus de cours donnés en licence de mathématiques et à la préparation à l'agrégation" ; Les illustrations auraient pu être faites avec d'autres logiciel comme Mupad, Mathematica, Maxima, etc. ,les différences sont minimes. On en donnera des versions simplifiées telles que de fait la plupart seront transposables et donc utilisables avec le logiciel "Derive", très répandu dans les établissements du secondaire (et nettement moins cher, c'est important).

Le but essentiel de mon exposé n'est pas tant de vous donner des exemples directement transposables dans les classes, il existe une nombreuse littérature à ce sujet, mais comme je le dis toujours aux étudiants : comment utiliser un logiciel de calcul formel pour **faire des maths** ?

Plan

1- **Trois préjugés : le logiciel peut tout faire ; il ne sait rien faire ; il se trompe souvent.** Exemple : y"=0.

2- **Sur la nécessité de nommer les objets, vérifier les résultats.**

3- **De la boite noire à la boite grise.** Exemple : la procédure "limit"

4- **Les "théorèmes d'un logiciel formel".** Exemple de l'interversion "limite" et "intégrale".

5- **Sur l'intégration.** Quelle théorie est-elle implémentée ? celle de Riemann ?

6- **Il n'y a pas de corrigé type.** Le triangle didactique.

7- **Sur les procédures.** Le nombre 19 m'intrigue.

1 Trois préjugés :

Le logiciel peut tout faire ; il ne sait rien faire ; il se trompe souvent.

A- "Le logiciel peut tout faire" :
c'est manifestement faux, il y a des problèmes pour lesquels le calcul formel n'est pas d'un grand secours.
Si je vous demande d'intégrer l'équation différentielle y"=0, qu'allez-vous me répondre ? y(t)=at+b probablement ; le logiciel va répondre de la même manière. J'aime bien cette équation y"(t)=0 car elle exprime un principe fondamental de la mécanique stipulant qu'un corps en chute libre possède une accélération nulle. Et pourtant c'est un problème mal posé ; en effet on résoud une équation différentielle dans un espace de fonctions précis.
Par exemple : Intégrer $y''(t) = 0$ dans $\mathcal{C}^2(\mathcal{R})$, puis dans $\mathcal{C}^2(\mathcal{R} - \mathcal{Z})$ puis dans $L^1(\mathcal{R})$ et enfin dans l'ensemble des fonctions continues sur $\mathcal{R}$ et différentiables en aucun point (si, si, cette question a du sens, le mouvement brownien serait solution, c'est une recherche actuelle).
Il est évident que les espaces de solutions sont très différents. Pour ce type de problèmes, un logiciel de calcul formel n'est d'aucune utilité.

B- "Le logiciel ne sait rien faire" :
alors pourquoi Maple résoud quasiment instantanément les équations différentielles proposées dans les livres utilisés pour la préparation à l'Agregation, par exemple celui de J.-P. Demailly (Analyse numérique et équations différentielles) ?

C- "Le logiciel se trompe souvent" :
ceci est vrai et faux, le préjugé est alors de croire que l'on n'y peut rien. De fait quand la réponse est erronée, la plupart du temps c'est la conséquence de l'utilisateur qui "force le logiciel à répondre faux". Les "bugs" sont de plus en plus rares.

2 Nommer, Vérifier

Etude de la limite d'une fonction.

Entrons et nommons f la fonction suivante :

```
> f := (sinh(sin(x)) - sin(sinh(x)))/(tanh(tan(x)) - tan(tanh(x)));
```

$$f := \frac{\sinh(\sin(x)) - \sin(\sinh(x))}{\tanh(\tan(x)) - \tan(\tanh(x))}$$

Demandons la limite de cette fonction en zéro.

```
> limit(f, x = 0);
```

$$\frac{-1}{2}$$

Un effort de présentation (pour faciliter une relecture)

$> Limit(f, x = 0) = limit(f, x = 0);$

$$\lim_{x \to 0} \frac{\sinh(\sin(x)) - \sin(\sinh(x))}{\tanh(\tan(x)) - \tan(\tanh(x))} = -1/2$$

Est-ce juste ? Quelles vérifications possibles ? Autrement dit la réponse $\frac{-1}{2}$ du logiciel est-elle valide, comment le vérifier ?

Voici deux vérifications possibles (il y en a d'autres) ; une première par l'étude des développements limités du numérateur et du dénominateur, une deuxième graphique.

Nous avons nommé la fonction, cela permet de prendre son numérateur et son dénominateur facilement ; de plus nommer un objet mathématique c'est aussi se l'approprier, mieux le saisir ; et l'on sait combien il est psychologiquement important de *nommer* dans la vie courante, c'est la même chose en mathématique, en plus de l'aspect pratique dans l'utilisation d'un calcul formel.

$> numerateur := numer(f);$

$$numerateur := -\sinh(\sin(x)) + \sin(\sinh(x))$$

$> denominateur := denom(f);$

$$denominateur := -\tanh(\tan(x)) + \tan(\tanh(x))$$

Faisons maintenant un dl du numérateur en utilisant l'instruction *taylor*

$> taylor(numerateur, x = 0, 5);$

$$O\left(x^5\right)$$

Faisons ce dl à l'ordre 12 :

$> tn := taylor(numerateur, x = 0, 12);$

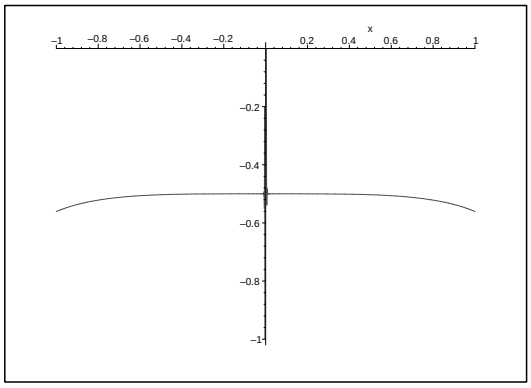
$$tn := \frac{1}{45}x^7 - \frac{1}{1575}x^{11} + O\left(x^{12}\right)$$

$> td := taylor(denominateur, x, 12);$

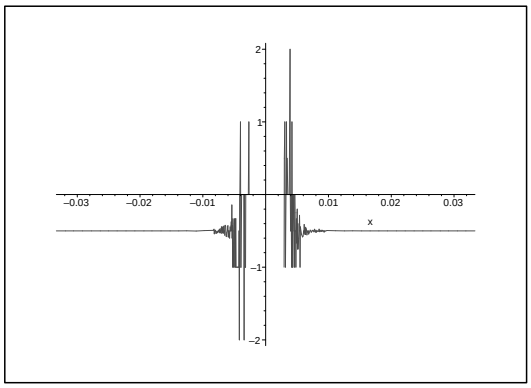
$$td := -\frac{2}{45}x^7 + \frac{26}{4725}x^{11} + O\left(x^{12}\right)$$

Cette première vérification est finie, puisqu'à l'évidence la limite du quotient de ces deux développements est $\frac{-1}{2}$; on remarque que si l'on voulait vérifier à la main (papier et crayon) le résultat, il faudrait prendre du temps et être méticuleux, car il faudrait faire des dl à l'ordre 8 ! Passons à une vérification de nature numérique :

$> plot(f, x = -1..1);$



Il apparaît visuellement que la limite est bien $\frac{-1}{2}$, mais avec un affolement numérique vers zéro. Faisons un zoom :
$> plot(f, x = -1/30..1/30);$



Nous obtenons confirmation visuelle (numérique) de ce que l'on pouvait déduire du premier tracé. Alors la limite est-elle exacte ? N'y a-t-il pas une conspiration du logiciel donnant le même résultat. On pourrait faire encore d'autres vérifications, par exemple en itérant sept fois la règle de l'Hospital ... De fait un logiciel de calcul formel ne donnera jamais une démonstration en toute rigueur ! Cependant si plusieurs vérifications donnent le même résultat, on peut être persuadé de la validité de celui-ci. On remarquera au passage que chaque vérification nécessite la mobilisation de savoirs mathématiques, et donc on fait des maths en cherchant des protocoles de vérifications différentes.

Le logiciel "derive" échoue face à cette fonction ; mais en utilisant la fonction $\frac{sin(x)-sinh(x)}{tan(x)-tanh(x)}$, on peut faire le même travail en terminale.

Suggestion : On peut tracer le graphe de cette fonction ($> plot(f, x = 0..100);$) et se demander si le graphe est bien correct ... et comment vérifier.

3 De la boite noire à la boite grise.

Chaque instruction d'un calcul formel est de fait une boite noire car on ne sait pas, a priori, ce qu'elle fait, on ne sait pas comment elle a été programmée. Un exemple (presque au hasard) : que fait donc l'instruction (la procédure) *limit* que nous venons d'utiliser ?

Examinons la réponse que le logiciel va donner à la recherche de la limite de $sin(x)$ quand x tend vers l'infini.
$> Limit(sin(x), x = infinity) = limit(sin(x), x = infinity);$

$$\lim_{x\to\infty} \sin(x) = -1 \ldots 1$$

? ? ? ? ?
$> Limit(a * sin(x), x = infinity) = limit(a * sin(x), x = infinity);$

$$\lim_{x\to\infty} a \sin(x) = \min(a, -a) \ldots \max(a, -a)$$

Est-ce un peu moins opaque ? On reconnaît la notation d'un segment par le logiciel Maple. Est-ce l'ensemble des valeurs d'adhérence ? Si oui la réponse est exacte, mais alors cette instruction *limit* ne cherche pas la limite ! Que fait-elle donc ? Un autre exemple :
$> limit((-1)^n, n = infinity);$

$$-1 \ldots 1$$

La réponse ne laisse pas de doute dans ce cas, *limit* nous donne le segment ayant pour bornes la limite inférieure et la limite supérieure de cette suite.

La boite noire *limit* devient grise, mais que fait-elle vraiment ? Donne-t-elle le segment contenant toute les valeurs d'adhérence possibles en calculant les limite inférieure et supérieure ? On est obligé de faire des maths (définitions de valeur d'adhérence, de limite inférieure, ...). Heureusement que lorsque ces limites inférieure et supérieure coïncident alors on obtient la limite. Pour que cette boite noire devienne blanche, il faudrait étudier ce que l'on appelle "le code source" que l'on ne connaît évidemment pas. Un aspect que j'ai oublié de signaler dans ma conférence est le fait, important, de recourir à l'aide (Help) du logiciel ; il y a toujours un paragraphe "Description" qui permet de "dégriser" une boite noire.

4 Sur les théorèmes de calcul formel.

L'exemple de la permutation de la limite et de l'intégrale.

Soit $x- > g(x)$ une fonction (définie sur R) ; alors l'instruction
$> Limit(int(g(x), x = a * y..b * y), y = 0) = limit(int(g(x), x = a * y..b * y), y = 0);$ donne

$$\lim_{y\to 0} \int_{ay}^{by} g(x)\, dx = 0.$$

Notre fonction g étant formelle, nous avons donc le :
$> THEOREME\_FORMEL := Limit(int(g(x), x = a*y..b*y), y = 0) = limit(int(g(x), x = a * y..b * y), y = 0);$

$$THEOREME\_FORMEL := \lim_{y \to 0} \int_{ay}^{by} g(x)\, dx = 0$$

Est-ce toujours vrai ? prenons l'exemple de $sin(x)/x^2$:
$> subs(g(x) = sin(x)/x^2, THEOREME\_FORMEL);$

$$\lim_{y \to 0} \int_{ay}^{by} \frac{\sin(x)}{x^2} dx = 0$$

mais si l'on calcule directement :
$> Limit(Int(sin(x)/x^2, x = 3*y..2*y), y = 0) = limit(int(sin(x)/x^2, x = 3*y..2*y), y = 0);$

$$\lim_{y \to 0} \int_{3\,y}^{2\,y} \frac{\sin(x)}{x^2} dx = \ln(2) - \ln(3)$$

Qu'est-ce qui est juste ? (On notera la puissance du logiciel et la nécessité de contrôles, avec retour à la théorie.) ; Le théorème formel est un théorème mathématiquement juste pour les fonctions continues et bornées au voisinage de la limite, ce qui n'est pas le cas de notre fonction $sin(x)/x^2$; le résultat $\ln(2) - \ln(3)$ est bien le bon. Il est important de signaler qu'un aspect de la puissance d'un calcul formel provient du fait qu'il admet comme théorèmes tout ce qui concerne des permutations d'opérations (limite, intégrale, série, dérivée, ...). Les procédures prédéfinies sont programmées souvent sous la forme d'un arbre, et donc quand un calcul pas à pas échoue, le logiciel utilise d'autres branches qui utilisent ces théorèmes génériques. C'est à l'utilisateur de se méfier. Pour $sin(x)/x^2$, Derive répond faux (il est de fait moins puissant, mais suffisamment pour le lycée)

5 Quelle théorie de l'intégration ?

Pour les parties 5 et 6 du plan je n'utiliserai pas de logiciel en session projetée.

Vous savez qu'il existe plusieurs théories de l'intégration, celle de Riemann, celle de Riemann généralisée, celle de Lebesgue. Elles n'intègrent pas les mêmes fonctions, car il existe des fonctions intégrables au sens de Lebesgue, non intégrables au sens de Riemann ; de même il existe des fonctions intégrables au sens de Riemann généralisée mais non intégrables au sens de Lebesgue. La question se pose donc de savoir quelle théorie est implémentée dans un logiciel de calcul formel.

Dans la mesure où c'est assez technique, nous regarderons ce problème en atelier cette après-midi ; cependant je vous livre le résultat essentiel : le logiciel intègre (correctement) des fonctions Lebesgue-intégrables et non Riemann-intégrables ; de même ce logiciel intègre des fonctions Riemann-généralisée-intégrables qui ne sont pas Lebesgue-intégrables ! Alors quelle théorie est-elle implémentée ? ... *C'est la "théorie Maple de l'intégration"*. Nous devons donc faire très attention lorsque l'on cherche à intégrer une fonction et au vu d'un résultat fourni par le logiciel, revenir aux théories de l'intégration pour saisir le sens du résultat et faire des vérifications. Et si l'on s'amuse à dire que la machine s'est trompée,

c'est une erreur, en fait c'est l'utilisateur qui a forcé la machine à fournir un résultat faux en ne prenant pas des précautions théoriques qui s'imposent. Là encore on fait des maths.

Il semble surprenant qu'un logiciel de calcul formel puisse intégrer des fonctions Lebesgue-intégrables et non Riemann-intégrables. Il y a deux raisons que je vois qui permettent de saisir que cela est pourtant possible. La première est liée à la structure en arbre des procédures, mais la deuxième plus fondamentale provient du typage des nombres dans un calcul formel ; en effet il faut se rappeler qu'un entier n'est pas un décimal, qu'un décimal n'est pas un rationnel et qu'un rationnel n'est pas un réel pour un logiciel. J'oserai dire que le concept de typage (concept d'informatique) a un lien avec celui de borélien (concept de la théorie de l'intégrale de Lebesgue). Je pense que les concepteurs des "noyaux durs" des procédures d'intégration en calcul formel n'ont pas pensé à cela lors de leur implémentation (sinon ils l'auraient dit).

6 Pas de corrigé type ;

le "triangle didactique".

J'ai insisté sur le fait qu'il fallait vérifier, réfléchir sur les stratégies de vérifications possibles ; et il est naturel de penser que derrière presque toute vérification il y a un théorème mathématique. Une expérience de 15 ans d'enseignement de calcul formel m'a montré que régulièrement des étudiants ont trouvé des procédures de vérifications auxquelles je n'avais jamais pensé, certaines très élégantes, d'autres impossibles à faire avec papier et crayon (tellement les calculs seraient longs), mais la puissance de calcul d'un ordinateur efface souvent ce problème de technicité et de longueur de calcul. Ainsi j'affirme (je témoigne) qu'il n'existe pas de corrigé type, pour un exercice donné à résoudre, avec un logiciel de calcul formel. De l'IREM de Lyon, je suis évidemment très sensibilisé par la pratique des "problèmes ouverts". Poser un problème sous forme ouverte, c'est une floraison de stratégies de résolution qui s'exprime avec de tels logiciels.

Mais derrière cette réalité "pas de corrigé type" se cache un problème, disons de nature didactique, plus profond : je le nomme "le triangle didactique". Il existe beaucoup de livres pour s'initier (ou approfondir) à la pratique d'un calcul formel. j'ai toujours été déçu par ces livres, et pourtant certains ont été rédigés par des collègues que j'estime beaucoup. Avec des collègues de Lyon (Olivier, Michel, Nik) on a réfléchi beaucoup sur la rédaction de documents transmissibles pour un "auto-apprentissage". Bilan : à chaque personne, une manière de mettre en oeuvre un logiciel de calcul formel ; pas de meilleure manière, chaque tempérament d'enseignant, de chercheur, s'exprime dans la manière d'utiliser un calcul formel. De même chaque étudiant appréhende de manière très personnalisée de tels logiciels. Bref je ne vois pas comment transmettre mon savoir (faire ou théorique) par l'intermédiaire d'un livre. Il y a "l'enseignant", "l'apprenant" et la "machine". Un triplet "enseignant-élève-machine" avec tant d'interaction dans ce triangle dynamique ... que pour moi le livre (statique) devient caduque (et je le répète, malgré la qualité admirable des auteurs).

Savoirs faire et savoirs savants sont intriqués (j'ai oublié de le dire dans la conférence), et pourtant c'est le noeud de ce que j'appelle le "triangle didactique".

7 Sur les procédures

Sur le thème : le nombre 19 m'intrigue

Les procédures, l'OUTIL par excellence de tout calcul formel !

Mais auparavant, je voudrai dire que cette partie est délicate, et pour ceux qui n'ont jamais utilisé un logiciel de calcul formel et qui se sentent un peu submergés par ce que je viens d'exposer, je les comprend ; pour eux j'ai choisi un thème d'arithmétique qui en lui-même peut intéresser l'auditoire.

Pourquoi le nombre 19 ? Il y a quelques années une équipe internationale s'est mis en chasse pour trouver des suites de nombres premiers consécutifs et en progression arithmétique. La plus longue connue comportait 7 nombres. Nous avons trouvé une suite de 8 nombres, puis avec l'aide de 200 personnes (et de leur ordinateur) nous avons trouvé une suite de 9 nombres (premiers consécutifs en progression arithmétique de raison 210=2*3*5*7). L'équipe voulait s'en arrêter là, mais Nik et moi avons proposé de continuer en changeant l'initialisation du programme ; l'initialisation que j'ai proposée était basée sur le nombre 19 pour des raisons à la fois intuitives et expérimentales. L'équipe s'est reformée, et trois mois après le record de 10 nombres tombait, plus de 100 fois plus rapidement que ne le prévoyaient des estimations probabilistes. Est-ce un heureux hasard, ou est-ce lié à des propriétés de ce nombre 19. En tout cas notre algorithme n'est pas assez performant pour espérer obtenir une suite de 11 nombres, car les estimations probabilistes nous donnent un temps de calcul de l'ordre de l'âge de l'univers ! Depuis le nombre 19 m'intrigue et je cherche des propriétés arithmétiques de ce nombre.

$> 19^2 \bmod 30$;

1

C'est le calcul de 19^2 modulo 30, ou reste de la division euclidienne de 19^2 par 30.
$> 17^2 \bmod 30$;

19

problème 1 : quels sont les couples d'entiers (k,n) tels que k soit premier, $k^2 = 1$ modulo n et $(k-2)^2 = k$ modulo n.
Une procédure :
> couple :=proc(n)
local i,L ;
L :=NULL :
for i from 3 by 2 to n do
if isprime(i) and $(i)^2$ mod n=1 and $(i-2)^2$ mod n=i then
$L := [n, i]$ end if ;
end do ; L end ;

la structure algorithmique de la procédure est donnée par le logiciel sous la forme suivante :
couple :=**proc**(n)
local i, L ;
L := NULL ;
for i **from** 3 **by** 2 **to** n **do**
if isprime(i) **and** $(i)^2$ mod n=1 **and** $(i-2)^2$ mod n=i **then** L := [n, i] **end if**
end do ;
L
end proc
On applique la procédure *couple* que l'on vient de définir pour n=30
> couple(30) ;

$$[30, 19]$$

puis on cherche les premiers couples [n,k], en calculant la suite (par l'instruction *seq*)
> seq(couple(n),n=2..400) ;

$$[30, 19], [35, 29], [45, 19], [70, 29], [90, 19], [110, 89], [130, 79], [135, 109], [140, 29],$$

$$[145, 59], [180, 109], [185, 149], [195, 79], [220, 89], [230, 139], [270, 109], [285, 229],$$

$$[290, 59], [330, 199], [335, 269], [345, 139], [370, 149], [380, 229], [390, 79]$$

Comme les plus simples des procédures sont les fonctions (qui sont de plus pratiques avec le logiciel Derive), écrivons la procédure "couple" ci-dessus en utilisant deux fonctions :
> test :=$(n, i)-$ >if isprime(i) and $(i)^2$ mod n=1 and $(i-2)^2$ mod n=i then [n,i] else NULL end if ;
> couplef :=$n->$ $seq(test(n, 2*i+1), i = 1..n/2)$;
puis on vérifie pour n=30
> couplef(30) ;

$$[30, 19]$$

puis on cherche les premiers couples [n,k] en utilisant ces fonctions
> seq(couplef(n),n=2..400) ;

$$[30, 19], [35, 29], [45, 19], [70, 29], [90, 19], [110, 89], [130, 79], [135, 109], [140, 29],$$

$$[145, 59], [180, 109], [185, 149], [195, 79], [220, 89], [230, 139], [270, 109], [285, 229],$$

$$[290, 59], [330, 199], [335, 269], [345, 139], [370, 149], [380, 229], [390, 79]$$

Si l'on cherche les couples [n,k] qui vérifient de plus k-2 premier, on modifie la procédure couple :
> couplej :=proc(n)
local i,L ;
L :=NULL :
for i from 3 by 2 to n do
if isprime(i) and isprime(i-2) and $(i)^2$ mod n=1 and $(i-2)^2$ mod n=i then

$L := [n, i]$ end if ;
end do ; L end ;
puis on cherche ces couples
> L :=seq(couplej(n),n=2..1000) ;

$$L := [30, 19], [45, 19], [90, 19], [135, 109], [180, 109], [230, 139], [270, 109], [285, 229],$$

$$[330, 199], [345, 139], [380, 229], [435, 349], [495, 199], [540, 109], [570, 229], [580, 349],$$

$$[690, 139], [870, 349], [990, 199]$$

Il y en a très peu ; 19 a donc une drole de propriété.

Mais suite à une remarque d'un ami (animateur à l'IREM de Lyon) sur 18=19-1, on remarque que $30 = 5/3 * (19 - 1)$, et plus généralement on a
Lemme :$(19+30*k)^2$ modulo $5/3*(18+30*k) = 1$ et $(17+30*k)^2$ modulo $5/3*(18+30*k) = 19 + 30 * k$. J'ai testé ce lemme avant de prendre papier et crayon pour le démontrer.
> verif1 :=$k- > (19 + 30 * k)^2$ mod $5/3 * (18 + 30 * k)$:
> seq(verif1(k),k=0..30) ;

$$1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1$$

> verif2 :=$k- > (17 + 30 * k)^2$ mod $5/3 * (18 + 30 * k)$-(19+30*k) :
> seq(verif2(k),k=0..30) ;

$$0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0$$

Conséquence : comme on obtient des nombres premiers jumeaux de la forme 17+30*n,19+30*n,
problème 2 : a-t-on tous les jumeaux se terminant par 7 et 9 ?
> jumeaux :=$n- >$ if isprime(9+10*n) and isprime(7+10*n) then 9+10*n end if ;
> Lj :=seq(jumeaux(i),i=1..100) ;

$$Lj := 19, 109, 139, 199, 229, 349, 619, 829, 859$$

> [Lj] mod 30 ;

$$[19, 19, 19, 19, 19, 19, 19, 19, 19]$$

Cela semble être vrai, il reste à prendre du papier et un crayon. Au passage signalons que l'on n'a toujours pas démontré qu'il existe un infinité de premiers jumeaux. Mais passons à un autre aspect.

11,13,17,19 est un quadruplet formé d'un couple de premiers jumeaux. De même 5,7,11,13. Cherchons les autres.
> quadruplet1 :=proc(n)
local i ;
i :=3+2*n ;
if isprime(i) and isprime(i-2) and isprime(i-6) and isprime(i-8) then return([i-8,i-6,i-2,i])
end if ;

end :
> [seq(quadruplet1(i),i=0..500)] ;

$$[[5, 7, 11, 13], [11, 13, 17, 19], [101, 103, 107, 109], [191, 193, 197, 199], [821, 823, 827, 829]]$$

Modifions légèrement la procédure en ne faisant apparaître que le dernier nombre du quadruplet :
> quadruplet2 :=proc(n)
local i ;
i :=3+2*n ;
if isprime(i) and isprime(i-2) and isprime(i-6) and isprime(i-8) then return(i) end if ;
end :
> L :=[seq(quadruplet2(i),i=0..3000)] ;

$$L := [13, 19, 109, 199, 829, 1489, 1879, 2089, 3259, 3469, 5659]$$

> L mod 30 ;

$$[13, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19]$$

Une propriété émerge : à part le premier quadruplet, tous vérifient i modulo 30 = 19. On prend une feuille de papier et on démontre cette propriété. D'où une procédure plus rapide :
> quadruplet :=proc(n)
local i ;
i :=19+30*n ;
if isprime(i) and isprime(i-2) and isprime(i-6) and isprime(i-8) then return(i) end if ;
end :
> L4 :=[seq(quadruplet(i),i=0..5000)] ;

$$L4 := [19, 109, 199, 829, 1489, 1879, 2089, 3259, 3469, 5659, 9439, 13009, 15649, 15739, 16069,$$

$$18049, 18919, 19429, 21019, 22279, 25309, 31729, 34849, 43789, 51349, 55339, 62989, 67219,$$

$$69499, 72229, 77269, 79699, 81049, 82729, 88819, 97849, 99139, 101119, 109849, 116539, 119299,$$

$$122209, 135469, 144169]$$

Etc. Trouver d'autres propriétés liées au nombre 19.

FIN

Evidemment il y aurait des tas de choses à dire et sur le nombre 19 qui m'intrigue toujours et sur le calcul formel.

J'espère avoir pu faire passer l'essentiel de mon message qui se résume à "Faire (faire) des mathématiques avec un logiciel de calcul formel, c'est non seulement souhaitable mais inévitable", ou plus prosaïquement comme aime à le dire un collègue "l'ordinateur remplace la main mais pas la pensée" ; au-delà du fait que les logiciels de calculs symboliques bousculent nos pratiques d'enseignement.

Bibliographie :

* *Enseignement des mathématiques et Logiciels de Calcul Formel.* DERIVE un outil à intégrer. Ministère de l'éducation nationale, 1994.
* *Des outils informatiques dans la classe aux calculatrices symboliques et géométriques : quelles perspectives pour l'enseignement des mathématiques ?* Commission Inter-IREM Mathématiques et Informatique. Actes de l'université d'été Rennes, Août 1996.
* *Faire des mathématiques avec un système de calcul formel.* Ministère de l'éducation nationale, deux tomes, 1998.
* *Calculatrices symboliques et géométriques dans l'enseignement des mathématiques.* IREM de Montpellier. Actes du colloque francophone européen, Mai 1998.
* *Environnements informatiques de calcul symbolique et apprentissage des mathématiques.* INRP, Commission Inter-IREM Mathématiques et Informatique, université de Rennes. Actes des journées de Rennes, juin 2000.

Pour d'autres sources, aller sur **Publimath** un site de l'APMEP et des IREM :
http ://publimath.irem.univ-mrs.fr/
et taper "calcul formel", il y a plus de 100 fiches.
Si vous tapez "derive", il y a 90 fiches.

Pour des activités en ligne aller sur le site national des IREM
http ://www.univ-irem.fr/
cliquez sur **publirem** (dans le bandeau en haut) puis entrez "calcul formel".