Chapitre 8

Suites et séries

Nous allons ici étudier la résolution symbolique de quelques exercices à l'aide de la TI-Nspire CAS. Vous trouverez également des informations précieuses pour le cas où vous auriez à faire des calculs utilisant des séries géométriques de raison négative. La dernière partie de ce chapitre traite les suites et séries de fonctions.

Sommaire

Philippe Fortin (Lycée Louis Barthou – Pau) / Roland Pomès (Lycée René Cassin – Bayonne)

1. Suites

Pour illustrer les possibilités de la TI-Nspire CAS dans ce domaine, nous allons étudier la suite définie par $u_{n+1} = \frac{4-{u_n}^2}{\sqrt{5}}$, avec $u_0 \in [0,2]$.

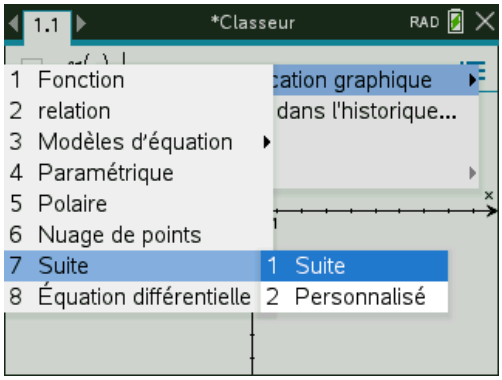
1.1 *Étude directe (graphique et numérique)*

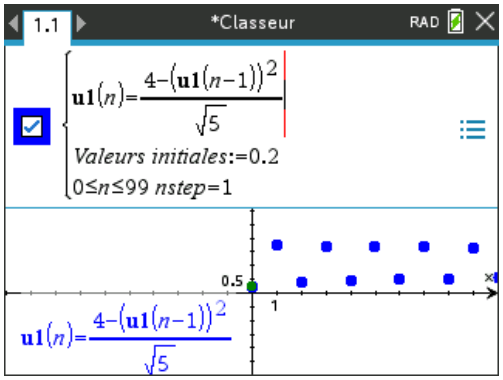
Cette possibilité était absente sur les premières versions de TI-Nspire CAS et il était nécessaire d'utiliser un classeur spécifique. Ce n'est plus le cas sur les versions récentes.

Sur la version 3.01 et les versions suivantes, il est possible d'étudier directement une ou plusieurs suites (représentation graphique et table de valeurs).
Voir manuel d'utilisation sur le site : **https://education.ti.com/fr/guidebook/**.

Il est possible de définir la suite, puis de choisir entre deux types de représentations (**Temps** ou **Toile**). On peut faire varier directement le point définissant la condition initiale, et obtenir un tableau donnant les valeurs de la suite.

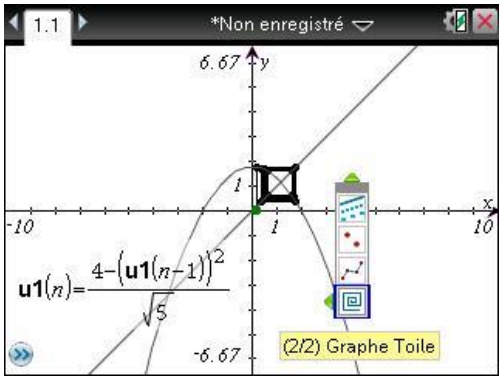
On ouvre un classeur avec l'application Graphiques, ctrl G, puis ctrl menu dans la ligne de saisie afin de choisir le type de représentation graphique, on choisit **Suite**. On saisit la définition de la suite ainsi que la condition initiale. Un premier tracé apparaît, n est porté en abscisse, u_n est ordonnées, c'est le mode **Temps**.

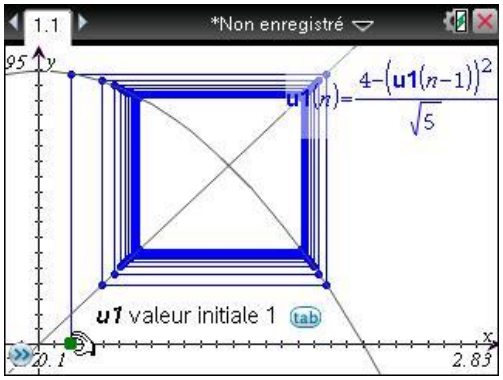




Pour passer en mode **Toile**, on sélectionne un des points de la suite, ctrl menu sélectionner **Attributs**.

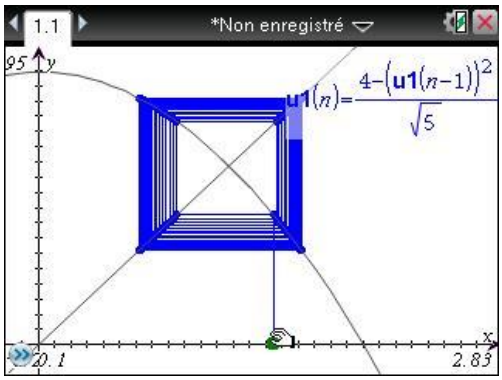
On peut changer la couleur du trait ainsi que les paramètres de la fenêtre d'affichage.

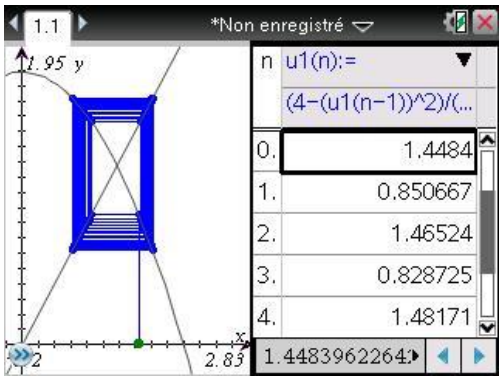




Il est possible de changer de condition initiale, directement sur le graphique, en saisissant le point vert qui représente u_0, et en le déplaçant.

ctrl T permet de faire afficher dans un partage d'écran un tableau donnant les valeurs des termes de la suite.



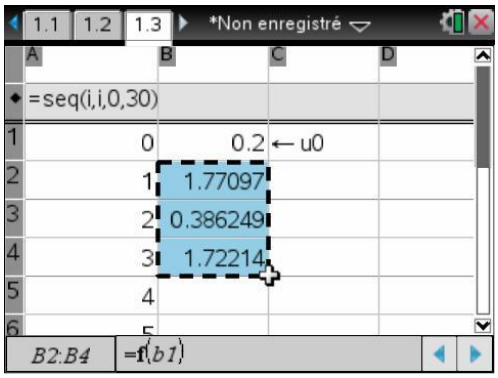


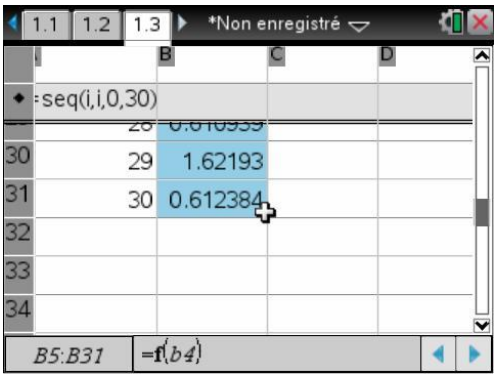
1.2 *Utilisation du tableur ou d'un programme*

On peut aussi construire directement une table de valeurs à l'aide de l'application Tableur & listes.
On commence par définir la fonction utilisée pour la définition de la suite récurrente dans l'application Calculs. On peut utiliser une instruction **define** (à gauche), ou la syntaxe abrégée (à droite) :

Il suffirait de modifier cette définition pour travailler avec une autre suite récurrente.

On ouvre ensuite l'application Tableur & listes et, dans la première colonne du tableau, on définit la suite des entiers naturels de 0 à 30 : **=seq(i,i,0,30)**. Dans la première case de la seconde colonne, on entre u_0, dans la deuxième **=f(b1)** et on copie vers le bas grâce à la fonction de saisie rapide (menu 3 3).

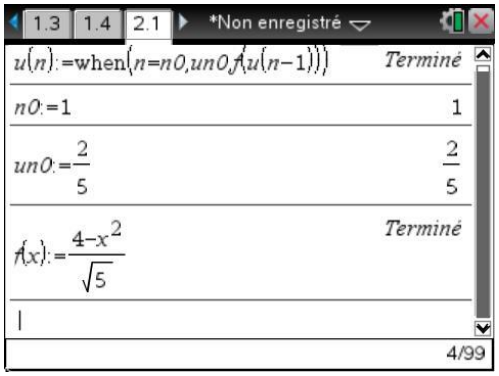


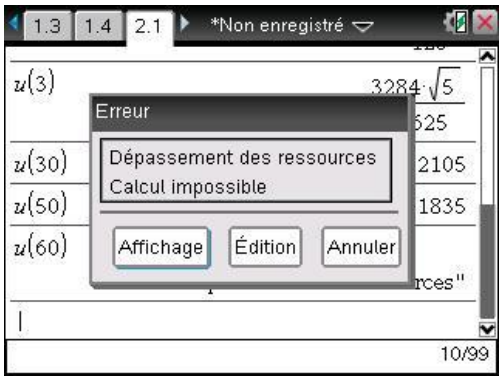


On peut aussi obtenir les valeurs de termes de la suite dans l'application Calculs :

Version récursive :

On peut utiliser la fonction **when** pour définir la suite de façon récursive, l'écriture est très simple, mais l'efficacité limité par le nombre d'appels récursifs. On verra en plus un autre problème au paragraphe 2, lors du calcul exact des termes d'une suite définie par récurrence.

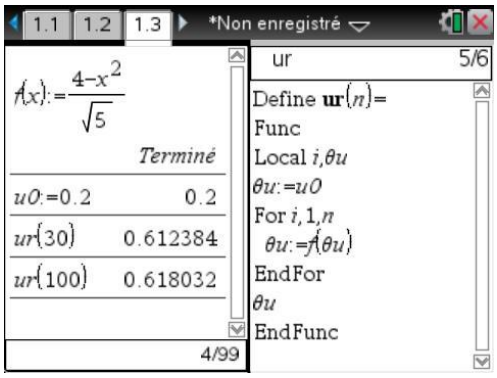




Version itérative :

La version itérative nécessite l'écriture d'une fonction, mais est plus rapide et permet de calculer des termes d'indice plus grands.

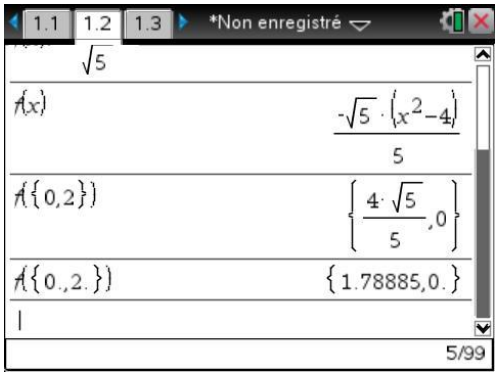
```
Define ur(n)=Func
  Local θu,i
  θu:=u0
  For i,1,n
  θu:=f(θu)
  EndFor
  θu
EndFunc
```



1.3 *Étude symbolique des limites éventuelles des suites extraites*

Cette suite est définie par une relation du type $u_{n+1} = f(u_n)$, avec f continue, décroissante sur $\mathbb{R}_+$. En particulier, f est décroissante sur $[0,2]$ et $f([0,2]) = [f(2), f(0)] = \left[0, \frac{4\sqrt{5}}{5}\right] \subset [0,2]$.

On peut naturellement confier à la TI-Nspire CAS le soin de faire ces derniers calculs (on remarquera qu'il est possible d'obtenir directement l'image d'une liste de valeurs et des valeurs approchées en faisant suivre d'un point les entiers).



On peut en déduire que tous les termes de la suite sont dans $[0,2]$. La continuité de f entraîne que si la suite (u_n) converge, cela ne peut être que vers un point fixe de f contenu dans cet intervalle.

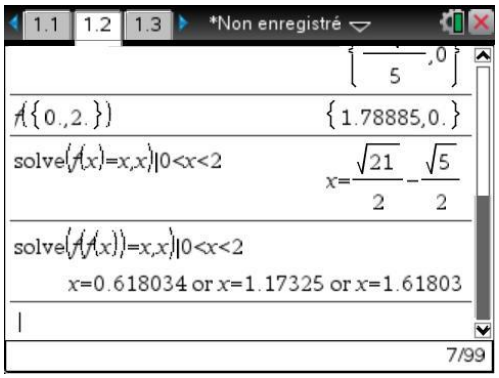
Les suites (u_{2n}) et (u_{2n+1}) sont toutes les deux monotones, puisque définies à partir de $g = f \circ f$, fonction croissante et continue sur $[0,2]$:

$v_n = u_{2n}$, $v_{n+1} = u_{2n+2} = f(u_{2n+1}) = f(f(u_{2n})) = f(f(v_n))$

$w_n = u_{2n+1}$, $w_{n+1} = u_{2n+3} = f(u_{2n+2}) = f(f(u_{2n+1})) = f(f(w_n))$

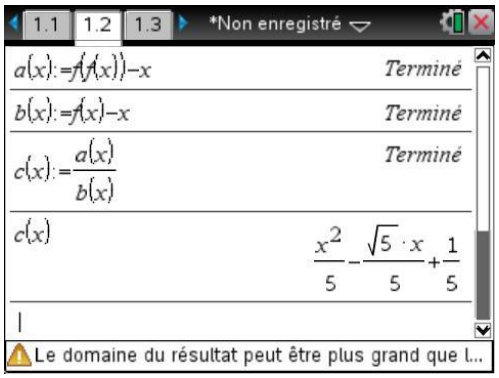
Ces suites sont bornées (on reste dans $[0,2]$), et monotones. Elles vont donc être convergentes. La continuité de $f \circ f$ montre que la limite ne peut être que l'un des points fixes de cette fonction.

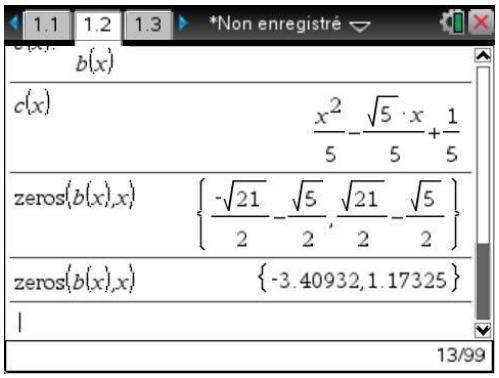
Il reste à étudier les solutions des équations $f(x) = x$ et $f(f(x)) = x$ dans l'intervalle $[0,2]$ pour déterminer les limites éventuelles de (u_n), (u_{2n}) et (u_{2n+1}).

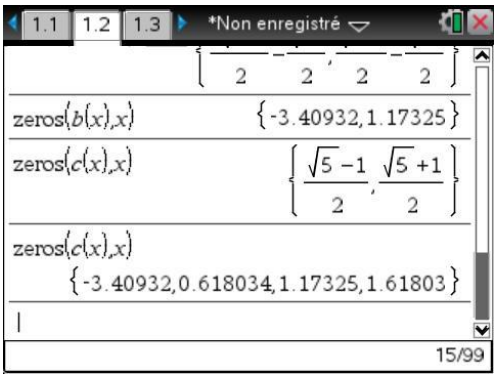


La TI-Nspire CAS a échoué dans la résolution symbolique de l'équation $f(f(x)) = x$...
Nous allons devoir lui donner un petit coup de main. Il s'agit en fait d'une équation polynomiale de degré 4, ce qui explique l'échec de la résolution, mais nous sommes ici dans un cas très favorable, puisque nous savons que les solutions de l'équation $f(x) = x$ (équation de degré 2) sont également solutions de l'équation $f(f(x)) = x$.

Cela montre que le polynôme $A(x) = f(f(x)) - x$ est divisible par $B(x) = f(x) - x$. Il suffit donc de calculer $C(x) = \dfrac{A(x)}{B(x)}$, puis de résoudre l'équation $C(x) = 0$ pour obtenir les solutions de $A(x) = 0$ (réunion des solutions de $B(x) = 0$ et de $C(x) = 0$).







☞ *On valide les calculs avec* ctrl enter *pour obtenir les valeurs approchées des solutions.*

En conclusion,

- les points fixes de $f \circ f$ dans l'intervalle $[0,2]$ sont : $r_1 = \dfrac{\sqrt{5}-1}{2}$, $r_2 = \dfrac{\sqrt{21}-\sqrt{5}}{2}$ et $r_3 = \dfrac{\sqrt{5}+1}{2}$.
  (La valeur $r_0 = -\dfrac{\sqrt{21}+\sqrt{5}}{2}$ n'est pas dans l'intervalle.)
- Les limites éventuelles des suites (u_{2n}) et (u_{2n+1}) sont égales à r_1, r_2 ou r_3.
- La valeur r_2, seul point fixe de f dans l'intervalle $[0,2]$ est la seule limite possible de la suite (u_n).

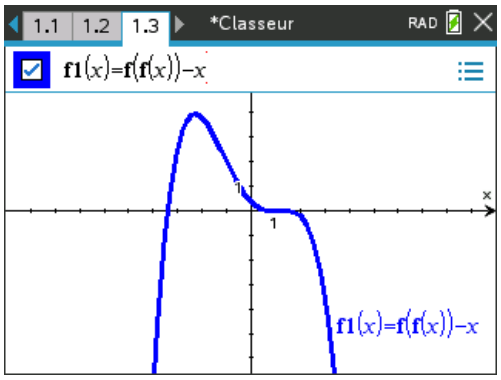
On peut également vérifier que $f(r_1) = r_3$ et $f(r_3) = r_1$.

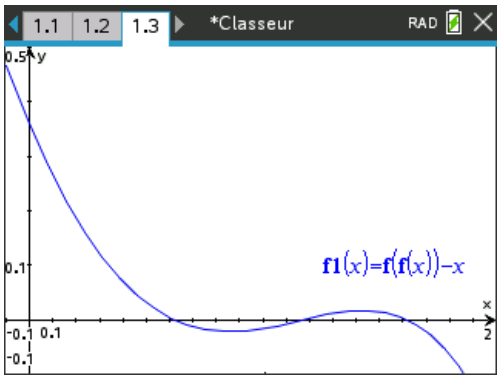
1.4 *Étude symbolique des suites extraites*

L'étude complète des suites extraites repose sur le fait que la suite est définie par une relation du type $u_{n+1} = f(u_n)$, avec f décroissante vérifiant $f(x) = x$ en r_1, r_2 et r_3.

- Les suites (u_{2n}) et (u_{2n+1}) sont toutes les deux monotones, puisque définies à partir de $g = f \circ f$, fonction croissante comme composée de deux fonctions décroissantes.
- La décroissance de f entraîne que si $u_{2n-1} \le u_{2n+1}$ alors $f(u_{2n-1}) \ge f(u_{2n+1})$, c'est à dire $u_{2n} \ge u_{2n+2}$. On montre facilement ainsi que les suites (u_{2n}) et (u_{2n+1}) ont donc des sens de variation opposés.
- Si $u_{2n+1} \to \ell$, alors, en utilisant la continuité de f, $f(u_{2n+1}) \to f(\ell)$. Donc (u_{2n}) converge vers $f(\ell)$. On peut montrer de la même manière que si (u_{2n}) converge vers ℓ', alors (u_{2n+1}) converge vers $f(\ell')$.
- Si le premier terme u_1 est dans un intervalle du type $[a,b]$ avec $f(f(a)) = a$ et $f(f(b)) = b$, la croissance de $f \circ f$, et donc de $(f \circ f)^n$ permet d'obtenir : $(f \circ f)^n(a) \le (f \circ f)^n(u_1) \le (f \circ f)^n(b)$, c'est-à-dire $a \le u_{2n+1} \le b$. On reste dans $[a,b]$.
- Si de plus $f(f(x)) - x$ est positif sur cet intervalle $[a,b]$, la suite (u_{2n+1}) est croissante sur cet intevalle car $u_{2n+3} - u_{2n+1} = f \circ f(u_{2n+1}) - u_{2n+1} \ge 0$. Elle est décroissante si $f(f(x)) - x$ est négatif sur $[a,b]$,

Il est facile d'étudier graphiquement le signe de $f(f(x)) = x$ sur $[0,2]$:





On peut justifier ce résultat en remarquant que le polynôme du quatrième degré $f(f(x))-x$ se factorise sous la forme $f(f(x))-x=-\frac{\sqrt{5}}{25}(x-r_0)(x-r_1)(x-r_2)(x-r_3)$.

Il est donc négatif sur $]-\infty,r_0[$, $]r_1,r_2[$ et $]r_3,+\infty[$, positif sur $]r_0,r_1[$ et $]r_2,r_3[$.

Si l'on se limite à l'intervalle $[0,2]$, on a un signe positif sur $I_1=]0,r_1[$ et $I_3=]r_2,r_3[$, négatif sur $I_2=]r_1,r_2[$ et $I_4=]r_3,2[$.

En utilisant les idées précédentes, il est alors facile de préciser le sens de variation des suites extraites, et de montrer que la suite ne converge pas, sauf dans le cas particulier où $u_1=r_1$.

- Si $u_1\in[0,r_1[$, tous les termes de la suite (u_{2n+1}) sont dans $[0,r_1]$. Cette suite est donc croissante ($f\circ f(x)-x\geq 0$), majorée par r_1. La suite converge, et sa limite comprise entre 0 et r_1, et ne peut être égale qu'à r_1, r_2 ou r_3. Cette limite est donc égale à r_1. Dans ce cas, la suite (u_{2n}) est décroissante, et converge vers $r_3=f(r_1)$.
- Si $u_1=r_1$, on a $u_2=f(r_1)=r_3$, puis $u_3=f(r_3)=r_1$, etc.
- Si $u_1\in]r_1,r_2[$, tous les termes de la suite (u_{2n+1}) sont dans $[r_1,r_2]$. Cette suite est donc décroissante ($f\circ f(x)-x\leq 0$), minorée par r_1, et la limite est comprise entre r_1 et $u_1<r_2$, et ne peut être égale qu'à r_1, r_2 ou r_3. Cette limite est donc égale à r_1. Dans ce cas, la suite (u_{2n}) est croissante, et converge vers $r_3=f(r_1)$.
- Si $u_1=r_2$, on a $u_2=f(r_2)=r_2$, etc. La suite (u_n) est stationnaire.
- Si $u_1\in]r_2,r_3[$, tous les termes de la suite (u_{2n+1}) sont dans r_2,r_3. Cette suite est donc croissante ($f\circ f(x)-x\geq 0$), majorée par r_3, et la limite est comprise entre $u_1>r_2$ et r_3, et ne peut être égale qu'à r_1, r_2 ou r_3. Cette limite est donc égale à r_3. Dans ce cas, la suite (u_{2n}) est décroissante, et converge vers $r_1=f(r_3)$.
- Si $u_1=r_3$, on a $u_2=f(r_3)=r_1$, puis $u_3=f(r_1)=r_3$, etc.
- Si $u_1\in]r_3,2]$, tous les termes de la suite (u_{2n+1}) sont dans $[r_3,2]$. Cette suite est donc décroissante ($f\circ f(x)-x\leq 0$), minorée par r_3, et la limite ne peut être égale qu'à r_1, r_2 ou r_3. Cette limite est donc égale à r_3. Dans ce cas, la suite u_{2n} est croissante, et converge vers $r_1=f(r_3)$.

2. Calcul symbolique des termes d'une suite récurrente

Dans l'exemple précédent, nous n'avons pas calculé les valeurs exactes des termes de la suite. Cela peut cependant être utile dans certains cas.

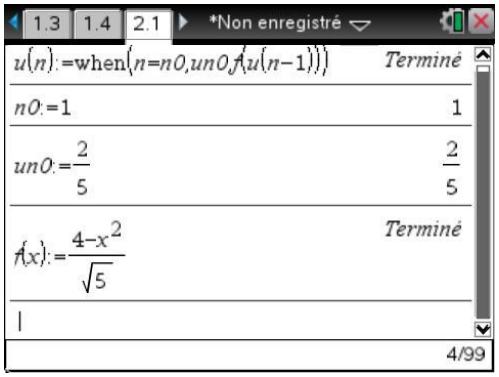
2.1 *Calcul par récurrence*

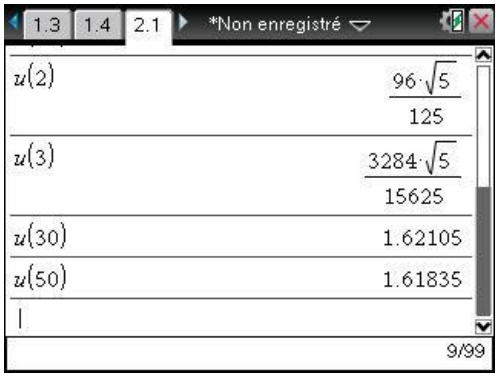
L'utilisation de la fonction **when** permet un calcul par récurrence. Pour définir une suite de premier terme u_{n_0}, et vérifiant $u_{n+1} = f(u_n)$, on peut utiliser l'instruction :

```
u(n):=when(n=n0,un0,f(u(n−1))
```

Il suffit d'indiquer la définition de **f** et la valeur de **n0** et **un0** pour que la machine puisse faire les calculs nécessaires.

Voici le calcul de termes de la suite étudiée dans le paragraphe précédent.





Le calcul d'un terme par cette méthode peut donc parfois être relativement compliqué, et même tout simplement impossible comme on peut le voir dans l'écran ci-dessus lors de la tentative de calcul de u_{60}. Le précédent calcul de u_{30} passe directement en valeur approchée.

Il est important de comprendre que si un calcul de type récursif est très simple à programmer, il peut cependant entraîner de gros calculs, d'autant que les résultats intermédiaires seront recalculés plusieurs fois si l'expression est mal formulée...

Prenons la suite définie par $u_0 = 1$ et $u_{n+1} = \dfrac{u_n + 1}{u_n + 2}$.

Si l'on utilise l'instruction

```
u(n):=when(n=0,1,(u(n−1)+1)/(u(n−1)+2))
```

le calcul du terme u_n va entraîner deux fois le calcul du terme u_{n-1}. Ainsi le simple calcul de u_{11} entraînera 2 calculs de u_{10}, 4 calculs de u_9, 8 calculs de u_8, $2^{10} = 1\,024$ calculs de u_1 ... si la machine n'abandonne pas avant !

Il suffit d'entrer la définition de la suite en deux temps, sous la forme :

```
f(x):=(x+1)/(x+2)

u(n):=when(n=0,1,f(u(n-1))
```

pour faire disparaître ce problème : lors du calcul de u_n, on ne calcule plus qu'une seule fois u_{n-1}, puis la valeur obtenue est utilisée par la fonction f pour déterminer u_n.

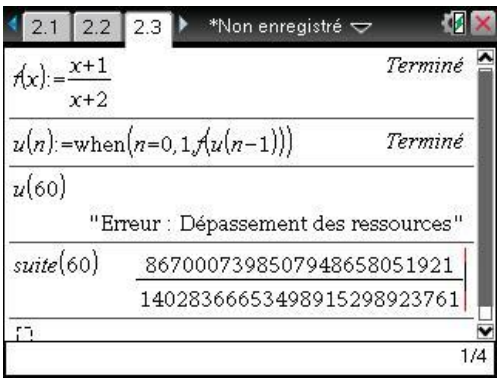
Une autre solution aurait été de remarquer que $u_{n+1} = 1 - \dfrac{1}{u_n + 2}$, et donc d'écrire

```
u(n):=when(n=0,1,1-1/(u(n-1)+2))
```

2.2 *Calcul itératif des termes*

Un calcul utilisant un calcul de proche en proche à l'aide d'une boucle est en fait nettement plus efficace en terme de temps de calcul. Voici une fonction permettant de faire cela.

```
Define suite(n)= Func
Local i,u
If n=0 Then
 1
Else
 u:=1
For i,1,n
   u:=(u+1)/(u+2)
EndFor
EndIf
EndFunc
```



Ci-dessus, à droite, on trouve la définition d'une fonction itérative permettant de calculer les termes de la suite à l'aide d'une boucle. Dans l'écran de droite, on a commencé par utiliser la méthode décrite dans le paragraphe précédent, mais il n'a pas été possible d'obtenir le calcul d'un terme d'indice un peu élevé. En revanche, ce calcul a été obtenu sans problème en utilisant la fonction itérative.

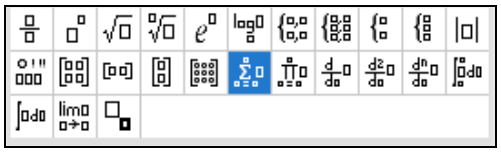
Voici quelques éléments pour comprendre les opérations effectuées par la fonction **suite**.

On commence par tester si l'indice est égal à celui du premier terme, et dans ce cas, on retourne la valeur de ce premier terme. Dans le cas contraire, on stocke la valeur du premier terme dans une variable *u* puis on répète l'opération $u := (u+1)/(u+2)$ autant de fois que nécessaire pour arriver au terme d'indice *n*. Le test sur la valeur de l'indice, et le choix des actions à entreprendre est fait par la structure **if** ... **then** ... **EndIf**. La répétition de l'opération de calcul des termes, et le décompte du nombre de répétitions nécessaires sont automatiquement gérés par l'instruction **For** ... **EndFor**. Si vous êtes débutant dans ce domaine, vous trouverez plus d'informations dans le **chapitre 14**.

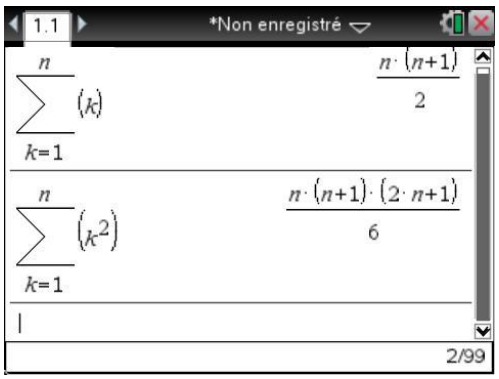
3. Séries

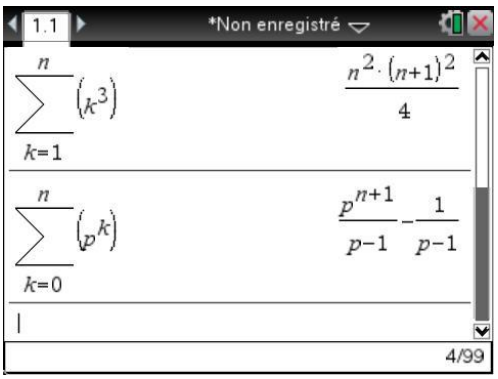
3.1 *Calculs directs de sommes finies ou infinies*

On entre le symbole somme soit à l'aide des modèles, soit à partir du menu **Analyse.** Dans de nombreux cas il est possible d'obtenir l'expression symbolique d'une somme de termes.

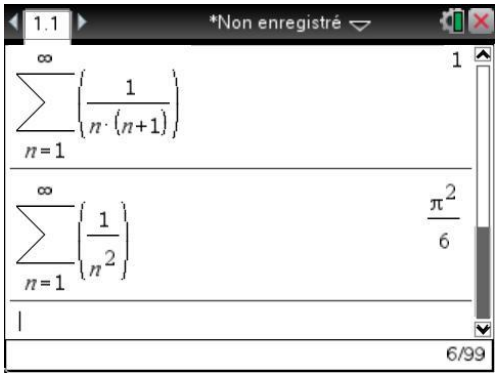


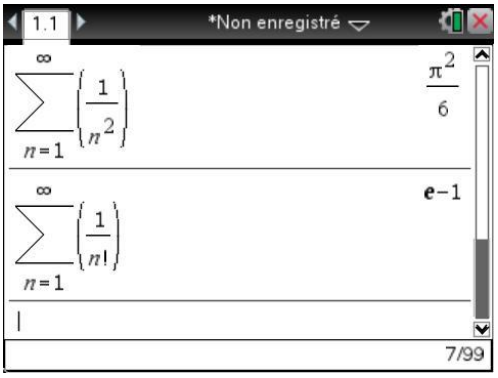
C'est en particulier le cas pour toutes les sommes finies classiques :



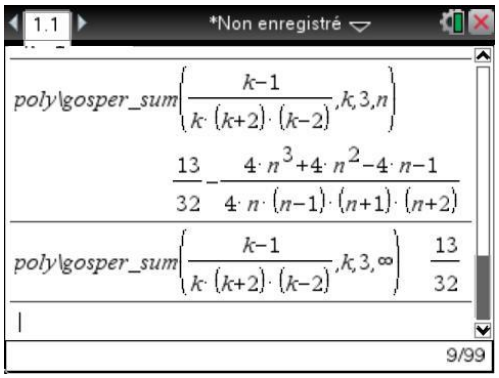


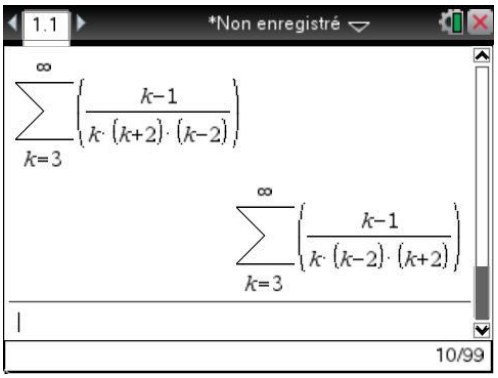
Il est donc inutile de perdre du temps à recopier ce type de formule dans un fichier texte de votre calculatrice… Celle-ci permet aussi le calcul de nombreuses sommes infinies.





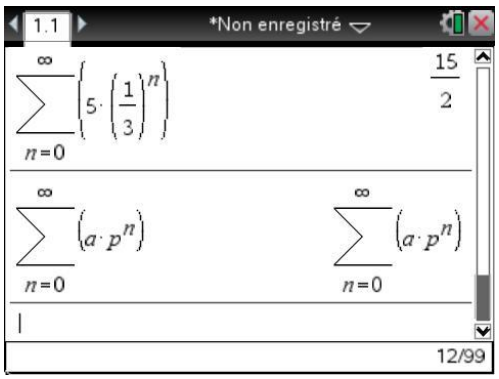
On peut aussi utiliser la fonction **gosper_sum** que vous trouverez dans la bibliothèque de programmes **poly** disponible avec ce document, voir également le **chapitre 15**. Elle permet d'obtenir des résultats dans certains cas où la fonction **sum** échoue.

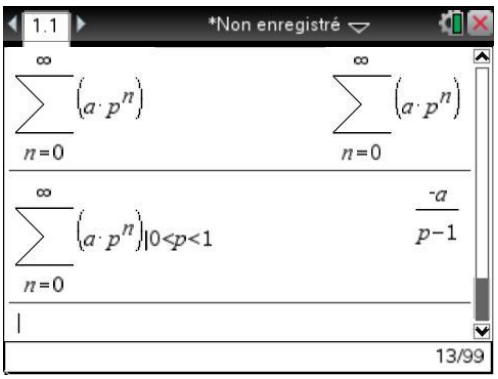




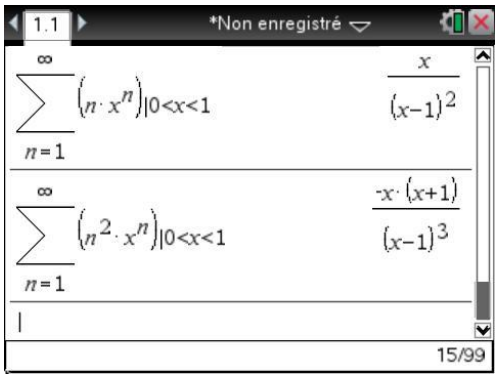
En ce qui concerne les séries géométriques, il vaut mieux connaître les possibilités exactes de la calculatrice. Le calcul ne pose aucun problème lorsque la raison est comprise entre 0 et 1.

Cela reste vrai dans le cas d'une raison dont on ne précise pas la valeur, mais il faut dans ce cas indiquer la condition sur cette raison :

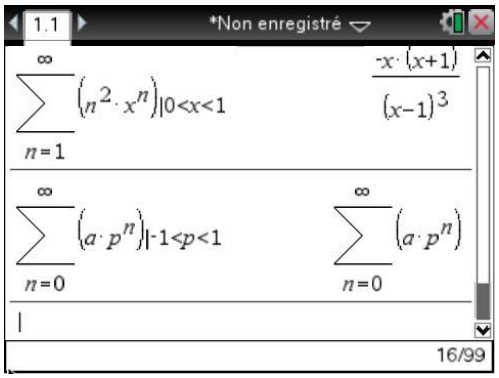


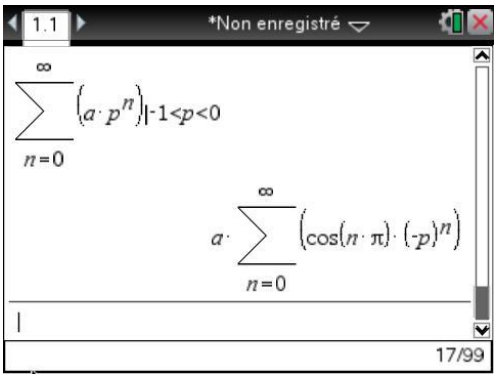


C'est la méthode à utiliser pour retrouver les formules au programme de classes préparatoires sur les sommes des $n\,x^n$ et n^2x^n :



En revanche, on n'obtient pas de résultat directement si l'on indique seulement que p est compris entre -1 et 1.





C'est également vrai, même pour une somme finie avec p fixé numériquement. Nous allons voir comment procéder dans ce cas dans la section suivante.