2^{de} Résolution d'une équation de type $f(x)=0$ à l'aide d'une méthode approchée

Lorsqu'on ne sait pas résoudre algébriquement une équation du type $f(x)=0$, on peut utiliser des méthodes approchées.

1. Méthode graphique.

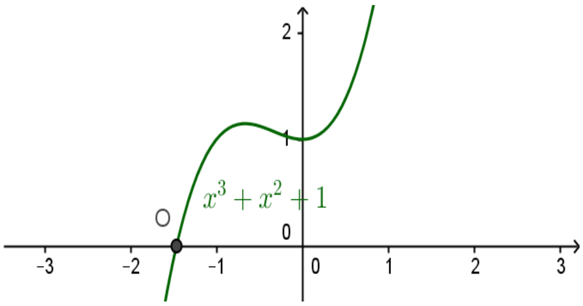
Le tracé de la courbe de la fonction f à l'aide d'une calculatrice graphique donne de précieuses informations quant-à l'existence et au nombre des solutions de l'équation ainsi que leurs valeurs approchées que l'on peut préciser en zoomant (en modifiant les paramètres de la fenêtre d'affichage). Certaines calculatrice vont plus loin et donnent également des solutions approchées...

On veut résoudre l'équation E_1 : $x^3+x^2+1=0$.

Vérifier qu'aucune des valeurs de l'ensemble $\{0, 1, 2, -1, -2\}$ n'est une solution évidente pour E_1.

On peut utiliser le tableur de la calculatrice, après avoir entré l'expression Y1=X^3+X^2+1 dans le module de saisie. En réglant (*set*) le tableur pour un « départ » en -2 , et un « pas » de 1, cela donne :

-2	-1	0	1	2
-3	1	1	3	13



On voit qu'aucune des images obtenues n'est 0 ; on n'a donc pas de solution évidente pour cette équation.

Taper l'expression $y=x^3+x^2+1$ dans le mode de saisie des fonctions de la calculatrice, puis tracer le graphique de cette fonction.

On règle ici les paramètres de la fenêtre d'affichage :

Xmin= -3 ;Xmax= 3 ;Xstep= 1 ;Ymin= -3 ;Ymax= 3 ;Ystep= 1

On lance alors le tracé du graphique. Le résultat est la courbe ci-dessus.

Observer la courbe (au besoin, adapter les paramètres de la fenêtre d'affichage) et conclure : combien pensez-vous que cette équation a de solutions ?

Il n'y en a qu'une (la courbe de la fonction tracée ne coupe l'axe des abscisses qu'en un seul point).

Donner un intervalle pour chacune des solutions observées.

Il n'y a qu'un intervalle car il n'y a qu'une solution, notée x_0 .

Graphiquement, on voit que $x_0 \in [-2 ; -1]$.

2. Méthode algorithmique : la dichotomie

Lorsque, sur un intervalle, le sens de variation d'une fonction f est constant et qu'il existe sur cet intervalle, une solution à l'équation $f(x)=0$, on peut s'en approcher autant que l'on veut en utilisant une méthode *algorithmique* comme la *dichotomie*.

Utilisons la dichotomie pour préciser la solution de E_1 : $x^3+x^2+1=0$.

Idée de l'algorithme : Dans l'intervalle [A;B] où la fonction f garde le même sens de variation et où $f(A)$ et $f(B)$ n'ont pas le même signe (on a $f(A)\times f(B)<0$), la solution de $f(x)=0$ appartient à l'intervalle. On détermine le centre C de l'intervalle : $C=\frac{A+B}{2}$ et son image $f(C)$:

- Si $f(A)\times f(C)<0$, $f(A)$ et $f(C)$ n'ayant pas le même signe, la solution appartient à [A;C] et on met la valeur de C dans B,
- Sinon la solution appartient à [C;B] et on met la valeur de C dans A.

On continue ce procédé jusqu'à ce que la différence entre A et B soit inférieure à une valeur donnée d'avance (appelée *précision*).

L'algorithme suivant traduit l'idée ci-dessus.

1. Entrer A et B (Sur Numworks A=... ; B=....) *A<B*
2. P=6 *P est un entier positif qui indique la précision, sous la forme 10^{-P}, de la solution*
3. Calculer X=f(A) et Y=f(B) *f est une fonction qui doit être entrée préalablement à l'exécution dans la calculatrice*
4. Tant que B−A>10^{-P} :
   Calculer C=(A+B)/2 et Z=f(C)
   Si X×Z<0 alors : B=C ; Y=Z *$f(A)$ et $f(C)$ sont de signes contraires quand $f(A)\times f(C)<0$*
   Sinon : A=C ; X=Z
5. Afficher C *la solution approchée avec une erreur inférieure à la précision demandée*

a) Faire fonctionner l'algorithme pas-à-pas, en remplissant le tableau ci-dessous pour les premières étapes :

N	A	B	$X=f(A)$	$Y=f(B)$	$C=\frac{A+B}{2}$	$Z=f(C)$	*Signe de* X×Z	$B-A$
1	−2	−1	−3	1	−1,5	−0,125	>0	1
2	−1,5	−1	−0,125	1	−1,25	0,6093	<0	0,5
3	−1,5	−1,25	−0,125	0,61	−1,375	≈0,291	<0	0,250
4	−1,5	−1,375	−0,125	≈0,291	−1,4375	≈0,0959	<0	0,1250

Pour passer de la ligne 1 à la ligne 2, on utilise le signe de X×Z (c'est-à-dire le signe de $f(A)\times f(C)$) : comme c'est positif (noté >0 ou +) on fait ce qui est dit dans l'algorithme « Sinon : A=C ; X=Z », on affecte

donc C à A et Z à X, soit A=−1,5 (c'est l'ancien C qui devient le nouveau A) et B=−1(inchangé). On calcule alors C=−1,25 et son image affectée dans Z=0,6093.
Pour passer de la ligne 2 à la ligne 3, on utilise le signe de X×Z qui est, cette fois, négatif (noté <0 ou −) on fait ce qui est dit dans l'algorithme « alors : B=C ; Y=Z », on affecte donc C à B et Z à Y, soit A=−1,5 (inchangé) et B=−1,25 (c'est l'ancien C qui devient le nouveau B). Et ainsi de suite...

Remarque : pour obtenir les valeurs de ce tableau facilement, on peut utiliser le mode tableau de la calculatrice en entrant pour un « départ » en -2, et un « pas » de 0,5 (première ligne). Ensuite, on change juste le « pas » : 0,25 pour la 2ème ligne, 0,125 pour la suivante, etc.

b) Programmer cet algorithme sur votre calculatrice
Pour une calculatrice Casio, on peut entrer le programme suivant (ne pas entrer les commentaires!!) :

?→A
?→B (il faudra entrer B>A pour que cela fonctionne correctement)
6→P (il faudra entrer P positif, par exemple 6 pour une précision de 10^{-6})
A→*X* (on utilise la touche X,θ,t pour cette mémoire qui est le nom obligé de la variable)
Y_1→X (Y_1 est calculé à partir de la fonction qui a été entrée dans le menu de saisie des fonctions ; en mode graphique, on s'en sert pour tracer la courbe d'équation $Y_1=f(X)$; nombre noté f(A) dans l'algorithme ; pour entrer Y_1 en mode Prgm, il faut taper *Vars* F_4 F_1 1 (les trois premières touches sont pour trouver le Y qui correspond au X,θ,t car ce n'est pas *Alpha* Y, le dernier 1 est le chiffre 1), avec une TI, il faut plutôt entrer *Var-Y* 1 :*fonction* 1:Y1)
B→*X* (encore X,θ,t pour cette mémoire)
Y_1→Y (nombre noté f(B) dans l'algorithme)
While B−A>10^-P (attention, c'est la touche notée (-) et non la soustraction notée −)
(A+B)÷2→C
C→*X* (encore X,θ,t)
Y_1→Z
If *X*×Z<0
Then
C→B
Z→Y
Else
C→A
Z→X
IfEnd
WhileEnd
C ◢

Ce programme est sans doute à adapter pour différent modèles.

Les utilisateurs de TI traduirons :
l'entrée des données se fait avec *Input*, l'affichage avec *Disp* et les fins de *While* ou de *If* avec *End*).
En testant ce programme sur une TI-82 Stats.fr je me suis aperçu que le résultat n'était pas correct. J'ai cherché l'explication qui s'est avérée que, pour cette calculatrice au moins (pour d'autres modèles aussi sans doute), les mémoires X,θ,t et X sont les mêmes !! Du coup, on ne peut pas noter le programme comme ça.
Il faut par exemple noter (j'ai surligné ce qui change, au lieu des mémoires X, Y et Z j'ai pris D, E et F) :

Input A
Input B
6→P
A→X
Y_1→D
B→X
Y_1→E
While B−A>10^-P
(A+B)÷2→C
C→X
Y_1→F
If D×F<0
Then
C→B
F→E
Else
C→A
F→D
End
End
Disp C

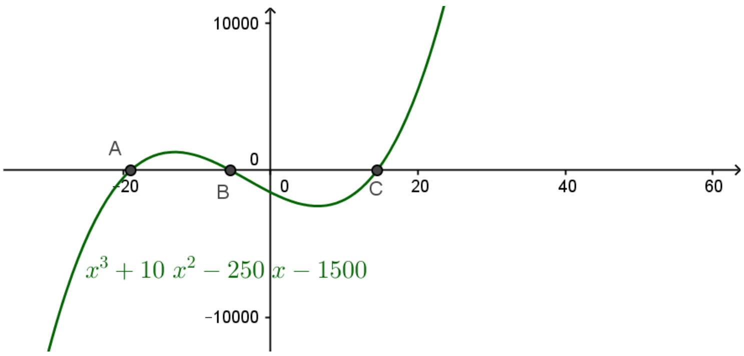
Les utilisateurs de *Numworks* traduirons également cet algorithme pour leur machine qui n'est pas encore très complète (pour ce qui est de la programmation Python, toujours en version bêta) :
D'après ce que j'ai compris, on ne peut pas utiliser les fonctions entrées dans le module (pour le tracé graphique et le tableau). Il faut donc entrer 3 fois l'expression de f dans le programme (pour les images de A, de B et puis de C). Je n'ai pas essayé, mais je crois qu'on peut laisser X, Y et Z (en Python on a plus de marge pour les noms de variables : on peut aussi utiliser les minuscules et les noms composés de plusieurs

lettres). Attention ici aux « : » qui suivent *while* et *if* (après le test) et aussi aux indentations (1 espace qui peut se cumuler quand il y en a plusieurs).

A = −2 B = −1 P = 6 X=A^3+A^2+1 (ou plutôt en Python A**3+A**2+1) Y=B^3+B^2+1 (ou plutôt en Python B**3+B**2+1)	*While* B−A>10^-P : (A+B)÷2→C Z= C^3+C^2+1 (ou plutôt en Python C**3+C**2+1) *If X*×*Z*<0 : B=C Y=Z	*Else* : A=C X=Z *Print*(C)

c) Tester ce programme (on entre A=−2, B=−1).
La solution de l'équation est, approximativement :
$x_0 \approx -1{,}465571404$ soit $x_0 = -1{,}465571 \pm 10^{-6}$ (arrondir le résultat à 10^{-6} près).
Donner, en changeant la valeur de P, un résultat à 10^{-8} près : on note 8→P dans le programme directement
$x_0 \approx -1{,}465571232$ soit $x_0 = -1{,}46557123 \pm 10^{-8}$ (arrondir à 10^{-8} près).
Ce n'était pas demandé, mais on peut en profiter pour vérifier à la fois le programme et le tableau ci-dessus : on va tester le résultat de la ligne 4 du tableau. Comme B-A est alors égal à 0,125 qui est la dernière valeur de B-A qui sera supérieure à 0,1 (la suivante sera 0,0625 car on divise à chaque fois par 2 cette amplitude), on devrait retrouver notre résultat en mettant P=1. Entrons 1→P dans le programme et exécutons :
$x_0 \approx -1{,}4375$ soit $x_0 = -1{,}4 \pm 10^{-1}$ (on a arrondi le résultat à 10^{-1} près).

3. De A à Z :
Résoudre l'équation $x^3 + 10x^2 - 250x - 1500 = 0$ par dichotomie :
Taper l'expression dans la fonction.
Visualiser le graphique (prendre xmin=−20 ; xmax=+20)
Vous remarquerez que pour voir la courbe, il faut aussi changer
Ymin= −1000 ;Ymax= 1000
Combien de solutions a cette équation E_2 ? trois
(au plus, une équation du 3ème degré a 3 solutions)
Donner un encadrement pour chaque solution
$x_A \in [-20 ; -10]$,
$x_B \in [-10 ; 0]$,
$x_C \in [10 ; 20]$.



J'ai rebaptisé les valeurs pour tenir compte des noms des points d'intersection de la courbe avec l'axe des abscisses.
Exécuter le programme pour ces différents intervalles et donner les solutions approchées attendues :
$x_A \approx -19{,}00220692$ soit $x_A = -19{,}002207 \pm 10^{-6}$,
$x_B \approx -5{,}458722711$ soit $x_B = -5{,}458723 \pm 10^{-6}$,
$x_C \approx 14{,}46092904$ soit $x_C = 14{,}460929 \pm 10^{-6}$,
Finalement, à 10^{-6} près, les solutions sont :
$x \approx -19{,}002207$ ou $x \approx -5{,}458723$ ou $x \approx 14{,}460929$