Informatique

**Newton – Dichotomie**

***Résumé***

|  |  |
| --- | --- |
| Recherche par dichotomie dans un tableau trié | |
| Le principe de la recherche par dichotomie dans une liste triée est le suivant :   * Vérifier que est non vide, sinon retourner False * Définir la plage d’indices de recherche notée (pour Indice gauche, Indice droite) qui au départ vaut * Déterminer l’indice milieu de la plage de recherche : * Déterminer la valeur de l’élément de la liste à l’indice * Tant que et , répéter la procédure suivante :   + et   + Si , définir la nouvelle plage de recherche à   + Si , définir la nouvelle plage de recherche à * A la sortie de la boucle deux possibilités :   + , renvoyer True   + , renvoyer True si , False sinon | |
| **def** recherche\_dicho**(**L**,**E**):**  **if** len**(**L**)==**0**:**  **return** **False**  ig **=** 0  id **=** len**(**L**)** **-** 1  im **=** **(**ig **+** id**)** **//** 2  Tm **=** L**[**im**]**  **while** ig **!=** id **and** Tm **!=** E**:**  im **=** **(**ig **+** id**)** **//** 2  Tm **=** L**[**im**]**  **if** E **>=** Tm**:** # Droite  ig **=** im **+** 1 # im exclus  **else:** # Gauche  id **=** im **-** 1 # im exclus  **if** Tm**==**E**:**  **return** **True**  **else:**  Tm **=** L**[**ig**]**  **return** Tm **==** E | |
| Meilleur des cas |  |
| Pire des cas |  |

|  |  |  |
| --- | --- | --- |
| Résolution de l’équation | | |
| Soit une fonction définie et continue sur un intervalle telle qu’il existe tel que : | | |
| Dichotomie | Newton | |
| Déterminer l’abscisse au centre de l’intervalle  et calculer son image par  Identifier dans quel intervalle ou se trouve la solution de à l’aide de du produit ou  Définir le nouvel intervalle de recherche comme celui dans lequel la solution existe  Continuer jusqu’à obtention du critère d’arrêt . Renvoyer  ATTENTION : test ou , si , doit être dans le nouvel intervalle | En une abscisse , déterminer l’équation de la tangente à la fonction :  Déterminer l’abscisse où cette tangente croise l’axe des abscisses :  Soit :  Continuer jusqu’à obtention du critère d’arrêt . Renvoyer , précision | |
| avec  Condition de convergence :  Variant : itération strictement croissant dans l’intervalle | Convergence quadratique (la précision double à chaque itération)  Condition de convergence : sur  de classe  ;  ;  ; | |
|  |  | |
| Division par 0 | Dérivée approchée  Méthode de la sécante |
| **Remarque pour les deux méthodes**  Ne pas être trop gourmand avec en gardant en tête la limite de la représentation des nombres en virgule flottante  Il arrive que l’on souhaite un critère d’arrêt sur , soit  Dans ce cas, renvoyer le associé pour obtenir la précision sur | Mauvaise solution | Non convergence |

|  |  |
| --- | --- |
| Exemples de programmations | |
| Dichotomie | Newton |
| **def** f**(**x**):**  **return** x**\*\***2**-**1 | |
| **def** dicho**(**f**,**a**,**b**,**eps**):**  **assert** f**(**a**)\***f**(**b**)** **<=** 0**,**"Pas de solution dans [a,b]"  **while** **abs(**b**-**a**)** **>** 2**\***eps**:**  m **=** **(**a**+**b**)/**2  **if** f**(**a**)\***f**(**m**)** **<=** 0**:**  b **=** m  **else:**  a **=** m  **return** **(**a**+**b**)/**2  **def** dicho**(**f**,**a**,**b**,**eps**):**  **assert** f**(**a**)\***f**(**b**)** **<=** 0**,**"Pas de solution dans [a,b]"  **while** **abs(**b**-**a**)** **>** eps**:**  m **=** **(**a**+**b**)/**2  **if** f**(**a**)\***f**(**m**)** **<=** 0**:**  b **=** m  **else:**  a **=** m  **return** m  sol **=** dicho**(**f**,**0**,**1**,**0.00001**)** | **def** newton**(**f**,**fp**,**x0**,**eps**):**  xi **=** x0  xip1 **=** x0 **+** 2**\***eps  **while** **abs(**xip1**-**xi**)** **>=** eps**:**  xi **=** xip1  xip1 **=** xi **-** f**(**xi**)/**fp**(**f**,**xi**)**  **return** xip1  **def** fp1**(**f**,**x**):**  # arg f nécessaire pour newton  Val **=** 2 **\*** x  **return** Val  **def** fp2**(**f**,**x**):** # Approximation  dx **=** 1e-5  Val **=** **(**f**(**x**+**dx**)-**f**(**x**))/**dx  **return** Val  sol **=** newton**(**f**,**fp1**,**0**,**0.00001**)** |
| **def** dicho\_rec**(**f**,**a**,**b**,**eps**):**  **assert** f**(**a**)\***f**(**b**)** **<=** 0**,**"Pas de solution dans [a,b]"  m **=** **(**a**+**b**)/**2  **if** **abs(**b**-**a**)** **<** 2**\***eps**:**  **return** m  **else:**  **if** f**(**a**)\***f**(**m**)<=** 0**:**  b **=** m  **else:**  a **=** m  **return** dicho\_rec**(**f**,**a**,**b**,**eps**)**  Sol **=** dicho\_rec**(**f**,**0**,**1**,**0.00001**)** |
| **print(**Sol**)** | |
| Sans assertion, une solution est renvoyée même si elle n’existe pas ! | Si la solution n’existe pas, le code risque de ne jamais s’arrêter |

Concernant la dichotomie, voici les tests possibles et l’intervalle associé :

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

|  |
| --- |
| Fonction native python pour la dichotomie : bisect de scipy.optimize |
| **from** scipy**.**optimize **import** bisect  **def** f**(**x**):**  **return** x**\*\***2**-**1  a **=** 0  b **=** 1  Res **=** bisect**(**f**,**a**,**b**)**  **print(**Res**)**  Ne fonctionne que si |

|  |
| --- |
| Fonction native python pour Newton : newton de scipy.optimize |
| **from** scipy**.**optimize **import** newton  **def** f**(**x**):**  **return** x**\*\***2**-**1  **def** fp**(**x**):**  **return** 2**\***x  x0 **=** 2  Res **=** newton**(**f**,**x0**,**fp**)**  **print(**Res**)**  On peut préciser la fonction dérivée fp si connue |
| **from** scipy**.**optimize **import** newton  **def** f**(**x**):**  **return** x**\*\***2**-**1  x0 **=** 2  Res **=** newton**(**f**,**x0**)**  **print(**Res**)**  Sans fprime en 3° argument ou avec fprime**=None**: Méthode de la sécante (dérivée inutile). |
| **from** scipy**.**optimize **import** newton  **def** f**(**x**):**  **return** x**\*\***2**-**1  **def** fp**(**x**):**  **return** 2**\***x  x0 **=** 2  Res **=** newton**(**f**,**x0**,**tol**=**0.1**)**  **print(**Res**)**  On peut préciser l’écart entre deux valeurs successives arrêtant les iterations en précisant tol**=**0.1  Lorsque rien n’est précisé, la fonction adapter automatiquement cet écart intelligement à chaque itération |