Chapitre 1

Prologue

Ce Prologue est consacré à quelques éléments d'optimisation unidimensionnelle et d'analyse numérique matricielle.

1.1 Algorithmes

Un peu de vocabulaire

Dans la suite, on va présenter un certain nombre de procédés de calcul itératifs qui engendrent des suites de nombres réels, de vecteurs ou de matrices. Ces procédés sont appelés *algorithme*s. Un algorithme peut être vu comme une application F d'un espace U dans lui-même. Le déroulement de cet algorithme à partir d'une valeur initiale u^0 sera de la forme

$$u^0 \in U,\ u^k \Rightarrow u = F(u^k),\ k = 1, 2, \ldots$$

À F, on peut associer l'ensemble de ses *points fixes* $F^\infty(U)$ défini par

$$F^\infty(U) = \{u \in U \mid F(u) = u\}\,.$$

Un point fixe $\overline{u}$ est dit *attractif* s'il admet un voisinage $V(\overline{u}) \subset U$ tel que

$$\forall u \in V(\overline{u}), \lim_{k\to\infty} F^k(u) = \overline{u}\,.$$

De la même façon, on dira qu'un sous-ensemble $\overline{U}$ est attractif si

$$\forall \overline{u} \in \overline{U},\ \exists V(\overline{u}),\ \forall u \in V(\overline{u}), \lim_{k\to\infty} F^k(u) \in \overline{U}\,.$$

Tout l'art de l'algorithmique consiste, pour résoudre dans un espace U un problème dont l'ensemble des solutions est U^*, à trouver un algorithme F tel

que $F^\infty(U)$ soit "proche" de U^* et attractif. ([1]) On définit aussi le *bassin d'attraction* associé à un point fixe $\overline{u}$. C'est l'ensemble de toutes les conditions initiales u^0 qui conduisent à une suite $F^k(u^0)$ convergeant vers $\overline{u}$. Un tel sous-ensemble est généralement très difficile à décrire, sauf lorsque que tout U est bassin d'attraction d'un unique point fixe.

On dira qu'un *algorithme converge* si les suites de valeurs qu'il engendre tendent vers une limite désirée dans l'espace considéré. En d'autres termes, il converge si ses points fixes attractifs sont candidats pour être solution du problème posé et leurs bassins d'attractions recouvrent la totalité de la région de U où l'on se pose le problème. ([2])

La *vitesse de convergence* d'un algorithme mesure la décroissance vers zéro de la distance entre les valeurs engendrées et leur limite. Par exemple, dans $\mathbb{R}^n$, notons $u^\star$ le vecteur vers lequel converge la suite $\left(u^k\right)_{k\in\mathbb{N}}$ (engendrée par un algorithme F donné) et $\|\ .\ \|$ la norme euclidienne :

Définition. (Vitesse de Convergence).

Si $\limsup_{k\to+\infty} \dfrac{\|u^{k+1}-u^\star\|}{\|u^k-u^\star\|} = \alpha < 1$, on dit que la convergence est *linéaire* et α est le *taux de convergence* associé. [3]

Si $\lim_{k\to+\infty} \dfrac{\|u^{k+1}-u^\star\|}{\|u^k-u^\star\|} = 0$, on dit que la convergence est *superlinéaire.*

Si $\exists\gamma > 1,\quad \limsup_{k\to+\infty} \dfrac{\|u^{k+1}-u^\star\|}{\|u^k-u^\star\|^\gamma} = M < +\infty$, la convergence est dite superlinéaire d'ordre γ, et en particulier, si $\gamma = 2$, on parle de *vitesse de convergence quadratique.*

Ces définitions ont intrinsèquement un caractère asymptotique, et on ne peut – en général – démontrer que la convergence est superlinéaire ou quadratique, que dans un voisinage de la limite recherchée $u^\star$ (loin de $u^\star$, il se peut très bien que l'algorithme concerné converge très lentement – ou pas du tout, bien que la vitesse convergence asymptotique soit en théorie quadratique). D'autre part, deux algorithmes ayant la même vitesse de convergence asymptotique peuvent très bien nécessiter des temps de calculs très différents si le nombre d'opérations exécutées à chaque itération est très différent.

1. La notion de proximité peut être comprise comme une inclusion dans un sens ou dans l'autre – ce qui n'est bien sûr pas la même chose sur le plan du résultat – ou comme une mesure d'approximation si l'espace U est par exemple normé. On pourrait demander à avoir l'égalité entre $F^\infty(U)$ et U^*, mais ce serait très exigeant... et peu efficace. On verra qu'il est souvent possible de tester directement si un candidat solution – obtenu par un procédé itératif – satisfait ou non à des *conditions* dites *d'optimalité.*

2. Les points fixes non attractifs, n'ont aucun intérêt puisqu'on ne les rencontre que par hasard...

3. On remarquera que c'est la convergence des suites géométriques.

Cette définition est néanmoins utile pour mesurer la qualité de la limite obtenue par un algorithme. Par exemple, supposons pour fixer les idées que la vitesse de convergence asymptotique d'un algorithme soit quadratique dans un voisinage de $u^\star$, avec $M = 100$. C'est dire qu'il existe un rang k tel que

$$\frac{\|u^{k+p+1} - u^\star\|}{\|u^{k+p} - u^\star\|^2} < 100, \quad \forall\ p = 0,\ 1,\ \ldots$$

Si, pour le rang k, on a $\|u^k - u^\star\| < 10^{-3}$ (on est dans un voisinage de la solution) alors, on vérifie aisément que

$$\|u^{k+1} - u^\star\| < 10^{-4}, \quad \|u^{k+2} - u^\star\| < 10^{-6}, \quad \|u^{k+3} - u^\star\| < 10^{-10} \text{ etc.}$$

L'erreur d'approximation passe donc, en seulement trois itérations, de 10^{-3} à 10^{-10} !

1.2 Optimisation unidimensionnelle

L'optimisation unidimensionnelle est l'étude des minima ou des maxima des fonctions de la variable réelle. Ce n'est pas un exercice gratuit de s'y intéresser uniquement en dimension un. En effet, d'une part on y retrouve souvent les mêmes difficultés que dans les dimensions supérieures (existence de minima locaux, non-différentiabilité, coût élevé d'évaluation de la fonction, etc.), d'autre part des méthodes d'optimisation unidimensionnelle sont utilisées sous forme de "procédures de recherche linéaire" dans les algorithmes d'optimisation vectorielle. Enfin, il est intéressant de se poser au moins une fois le problème de trouver le minimum d'une fonction que l'on ne peut pas se représenter : en effet, un minimum, un maximum, un zéro d'une fonction, cela saute aux yeux lorsque l'on trace la fonction. Mais si l'on peut tracer la fonction, il est clair que l'on triche : on dispose de toute l'information en même temps, et il suffit de faire glisser verticalement la règle vers le haut ou vers le bas et de s'arrêter lorsqu'un seul point de la courbe touche la règle. Or, dans la pratique, il est important de comprendre que l'on ne dispose que d'une information *locale* sur la fonction à minimiser. Cette information peut être un ensemble de valeurs de la fonction en des points (forcément en nombre fini, et donc de mesure nulle) choisis par l'utilisateur. On peut aussi disposer de valeurs de la dérivée de cette fonction ou d'approximations de cette dérivée (différences finies ou valeurs exactes entachées d'un bruit de mesure ou de calcul), ou d'indications sur le comportement *à l'infini* de la fonction. Pour se représenter la situation dans laquelle se trouve réellement l'optimiseur, il suffit de tracer le graphe d'une fonction (éventuellement avec des discontinuités) au crayon, d'en marquer à

l'encre quelques points et tangentes, puis d'effacer le graphe. Il s'agit maintenant d'élaborer une stratégie intelligente pour choisir en quels points on désire plus d'information, sachant que celle-ci coûte cher. On réalisera alors d'une part que les algorithmes présentés ici ne marchent bien que parce que l'on a supposé une très grande régularité des fonctions considérées (dérivabilité, convexité, unimodalité), d'autre part que ces "recettes de cuisine numérique" font preuve – dans le cadre de ces hypothèses simplificatrices – d'une extrême ingéniosité, tirant parfois le maximum de l'information disponible.

Nous allons présenter ici deux méthodes d'optimisation unidimensionnelle : la méthode de dichotomie et la méthode de la section dorée (ou du *nombre d'or*). Ces méthodes ne supposent pas la dérivabilité ni même la continuité de la fonction étudiée, mais seulement son *unimodalité*. Elles s'appliquent à une fonction réelle f à valeurs dans un intervalle $[a, b]$, dit *intervalle d'incertitude*, à l'intérieur duquel nous savons (ou supposons) qu'elle possède un minimum unique. Si la fonction avait - par hasard - plusieurs minima, on aurait toutes les chances de n'en trouver qu'un seul, de toute façon.

Définition. Soit f une fonction numérique définie sur $[a, b]$ et soit x^* l'argument de son minimum sur cet intervalle([4]). On dit que f est *unimodale* sur $[a, b]$, si elle est strictement monotone décroissante sur $[a, x^*[$ et strictement monotone croissante sur $]x^*, b]$.

En d'autres termes, si y et z sont du même côté de x^*, alors leurs images par f sont aussi du même côté de $f(x^*)$.

La méthode de dichotomie qui suit est simple mais peu efficace. A chaque itération, elle nécessite deux évaluations de la fonction f et permet de réduire de moitié l'intervalle. Sa vitesse de convergence est donc linéaire, de rapport $\frac{1}{2}$. La méthode de la section dorée est plus économique, puisqu'elle conduit à une vitesse de convergence linéaire de rapport $\frac{\sqrt{5}-1}{2} \approx 0.6$ pour une seule évaluation de la fonction f à chaque itération.

1.2.1 Méthode de dichotomie

Supposons que l'on ait calculé la valeur de f en cinq points régulièrement espacés de l'intervalle $[a, b]$: $a = x_1 < x_2 < x_3 < x_4 < x_5 = b$, avec x_3 milieu de $[a, b]$ et x_2 et x_4 milieux respectifs de $[x_1,\ x_3]$ et $[x_3,\ x_5]$. Comme la fonction est unimodale, seuls les cinq cas suivants peuvent se présenter (exercice : faire un dessin, ou voir plus loin...) :

1. $f(x_1) < f(x_2) < f(x_3) < f(x_4) < f(x_5) \Rightarrow$ on peut éliminer x_3, x_4 et x_5 car le minimum est forcément entre x_1 et x_2.

4. $f(x^*) = \min_{y \in [a,b]} f(y)$.

2. $f(x_1) > f(x_2) < f(x_3) < f(x_4) < f(x_5) \Rightarrow$ élimination de x_4 et x_5
3. $f(x_1) > f(x_2) > f(x_3) < f(x_4) < f(x_5) \Rightarrow$ élimination de x_1 et x_5
4. $f(x_1) > f(x_2) > f(x_3) > f(x_4) < f(x_5) \Rightarrow$ élimination de x_1 et x_2
5. $f(x_1) > f(x_2) > f(x_3) > f(x_4) > f(x_5) \Rightarrow$ élimination de x_1, x_2 et x_3, car le minimum est forcément entre x_4 et x_5.

Sauf dans les cas où l'une des bornes est la solution, on se retrouve avec un triplet que l'on notera (y_1, y_3, y_5) tel que $f(y_1) > f(y_3) < f(y_5)$, et la solution optimale appartient donc à l'intervalle $[y_1, y_5]$ strictement inclus dans $[a, b] = [x_1, x_5]$. Dans la plupart des cas, la longueur du segment $[y_1, y_5]$ sera la moitié de la longueur de $[a, b]$ ([5]) puisqu'on aura enlevé deux quarts. En déterminant les nouveaux points y_2, y_3, et y_4, de manière consistante, on assure la régularité de l'algorithme :

$$y_3 = (y_1 + y_5)/2, \;\; y_2 = (y_1 + y_3)/2, \;\; y_4 = (y_3 + y_5)/2.$$

Au total, après n évaluations de la fonction f , on aura réduit l'intervalle d'origine d'un facteur $2^{(n-3)/2}$

```
dicho[f_, a_, b_, n_] := (* fonction, bornes, # évaluations *)
Module[{x1, x2, x3, x4, x5, p, f1, f2, f3, f4, f5},
p = (n - 3) / 2; (* nombre d'itérations à faire *)
x1 = a; x5 = b; x3 = (x1 + x5)/2; xmin = {a}; xmax = {b};
  Do[ x2 = (x1 + x3) / 2 ; x4 = (x3 + x5) / 2;
   f1 = f[x1]; f2 = f[x2]; f3 = f[x3]; f4 = f[x4]; f5 = f[x5];
Which[f1 < f2 < f3 < f4 < f5, y1 = x1; y5 = x2; y3 = (y1 + y5) / 2,
    f1 > f2 < f3 < f4 < f5, y1 = x1; y3 = x2;
    y5 = x3, f1 > f2 > f3 < f4 < f5, y1 = x2; y3 = x3; y5 = x4,
    f1 > f2 > f3 > f4 < f5, y1 = x3; y3 = x4; y5 = x5,
    f1 > f2 > f3 > f4 > f5, y1 = x4; y5 = x5;
    y3 = (y1 + y5) / 2]; x1 = y1 ; x3 = y3 ; x5 = y5;
AppendTo[xmin, x1]; AppendTo[xmax, x5], {p}];
  ListPlot[{xmin, xmax}, Joined → True, PlotMarkers → Automatic,
   PlotLabel → "Convergence →" <> ToString[(x1 + x5) / 2.] <>
     " en " <> ToString[n] <> " évaluations"]]
```

FIGURE 1.1 – Algorithme de dichotomie

5. D'un point de vue étymologique, dichotomie signifie couper en deux.

```
f[x_] := (x - 1)^2; a = 0;
b = 3; dicho[f, a, b, 20]
```

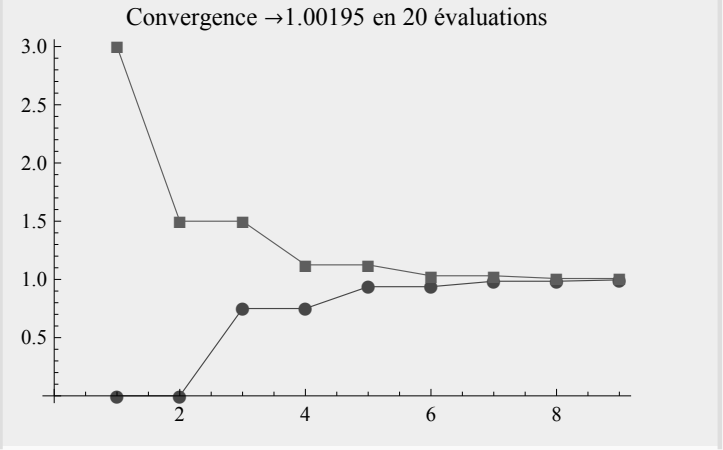

FIGURE 1.2 – Application de la dichotomie à f *unimodale*

```
f[x_] := 1 / 3 (x - 1)^3; a = -3;
b = 3; dicho[f, a, b, 20]
(* convergence vers une
 borne : f non unimodale *)
```

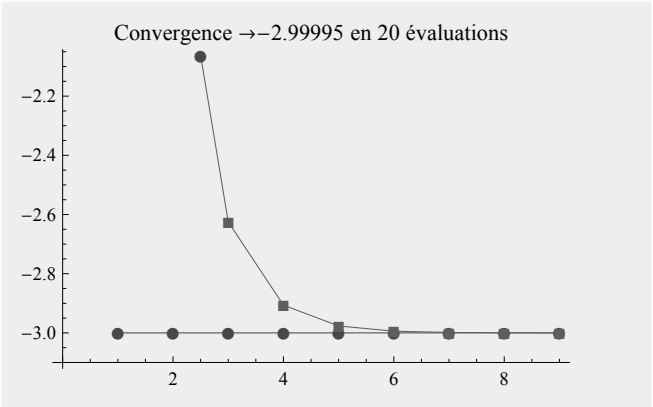

FIGURE 1.3 – Cas pathologique : fonction *non unimodale*

```
g[x_] := N[If[x < 0 || x > 5, 2 (x - 2.5)^2,
    If[x < 3, 10 - x^2, 1 + (x - 3)^2]]],
Plot[g[x], {x, -1, 6}, PlotStyle → Thick]
```

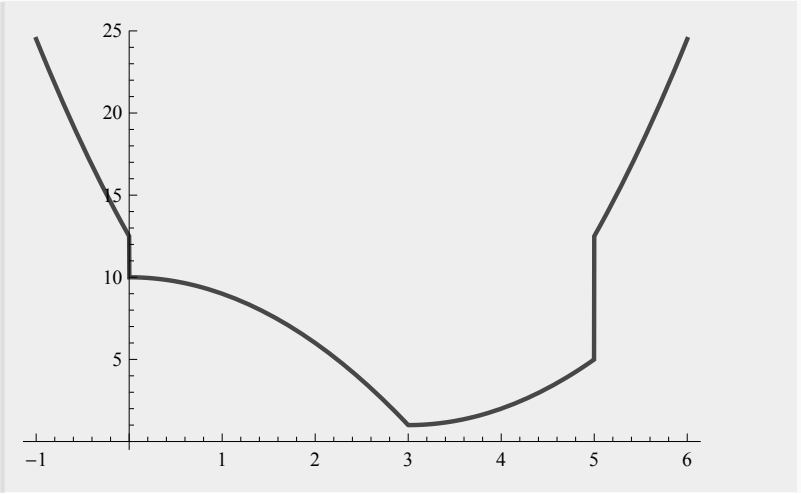

FIGURE 1.4 – g, une fonction unimodale non différentiable. Son minimum est $x = 3$.

```
dicho[g, -2, 4, 21]
```

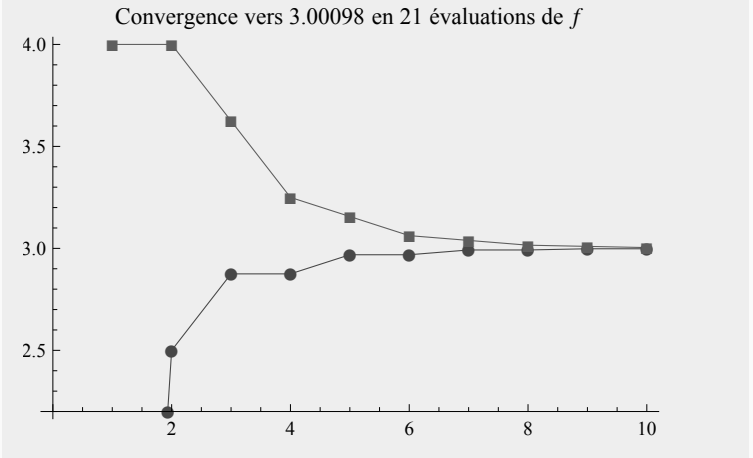

FIGURE 1.5 – Succès de la méthode de dichotomie pour g

Exercice 1. Le programme de dichotomie contient une faute. Laquelle ?

1.2.2 Méthode de la section dorée

Cette méthode est presque la même que la précédente, à la différence près qu'au lieu de découper l'intervalle d'incertitude en 4, on le découpe en 3, ce qui ne coûte, à chaque itération, qu'une seule évaluation supplémentaire de la fonction. Notons qu'il est impossible de découper l'intervalle courant en trois morceaux égaux à chaque itération. En effet, découpons I en trois intervalles de longueurs égales : $I = [x_1, x_2] \cup [x_2, x_3] \cup [x_3, x_4]$. Supposons qu'après avoir comparé comme ci-dessus les valeurs $f(x_1),\ f(x_2),\ f(x_3),\ f(x_4)$, on ait éliminé le premier sous-segment $[x_1, x_2]$. Le nouveau point y où évaluer la fonction serait forcément dans l'intervalle complémentaire $[x_2, x_4]$, d'un côté ou de l'autre du milieu x_3, rompant ainsi la symétrie : les trois nouveaux intervalles ne peuvent donc être égaux. Bien que cela semble étonnant à première vue, il est possible d'obtenir une réduction (relative) constante de l'intervalle d'incertitude à chaque itération, tout en découpant l'intervalle en trois morceaux inégaux ! Et donc, on aura quand même une vitesse de convergence linéaire.

En effet, notons L^k la longueur de l'intervalle d'incertitude à l'itération k, c'est-à-dire $L^k = L([x_1^k, x_4^k])$. On désire avoir $\frac{L^{k+1}}{L^k} = \gamma < 1$. Pour que cette relation soit vraie, il faut que les intervalles $[x_1^k, x_2^k]$ et $[x_3^k, x_4^k]$ soient de même longueur, c'est-à-dire que x_2^k et x_3^k soient symétriques par rapport au milieu de $[x_1^k, x_4^k]$, puisqu'on ne sait pas lequel d'entre eux va être éliminé. Cette hypothèse de symétrie implique que (voir la Fig. 1.2.2) :

$$L^k = L^{k+1} + L^{k+2}\ . \tag{1.1}$$

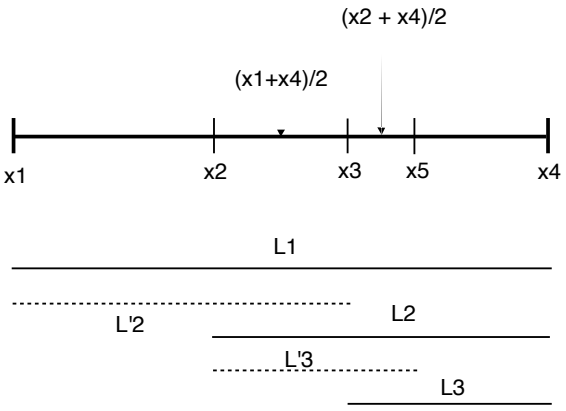

FIGURE 1.6 – Premières étapes de la méthode de la section dorée