L'énoncé démontré par Steven Piantadosi est le suivant.

- On se fixe une précision e, par exemple 1/1000, ce qui signifie que l'on recherchera à avoir des égalités avec une erreur inférieure à e.

- Il existe une fonction $f_p(x)$ dépendant d'un paramètre réel p, telle que, quelles que soient les données $y_0, y_1, \ldots, y_n$ en nombre fini, il existe une valeur p du paramètre pour laquelle $f_p(0)=y_0$, $f_p(1)=y_1$, ..., $f_p(n)=y_n$, avec une erreur inférieure à e pour chaque égalité.

En clair, en faisant varier le paramètre p, la fonction $f_p(x)$ s'ajuste avec une précision e à tout jeu de données fixé à l'avance, aussi grand soit-il.

La fonction miraculeuse de Steven Piantadosi est : $f_p(x)=\sin^2(2^{kx}\arcsin(\sqrt{p}))$. Rappelons que $\arcsin(x)$ est la fonction inverse de $\sin(x)$, ce qui signifie que pour tout x compris entre $-\pi/2$ et $\pi/2$, $y=\sin(x) \Leftrightarrow x=\arcsin(y)$.

La valeur k qui apparaît dépend de la précision e souhaitée, mais une fois e fixée, k l'est aussi. Bien évidemment, le chercheur a calculé le paramètre p qui donne un éléphant. L'encadré 1 montre le résultat.

D'OÙ VIENT L'EXPLOIT ?

Comprendre en détail comment la fonction $f_p(x)$ réalise cet exploit, et comment s'y prendre pour trouver les bonnes valeurs du paramètre p, est un jeu mathématique qui exige un peu de calcul (*voir les encadrés 3 et 4*). Cependant, l'idée générale est simple : on va cacher dans le paramètre p qui, parce qu'il est un nombre réel, peut contenir une infinité d'informations, les nombres $y_0, y_1, \ldots, y_n$. Ensuite, la >

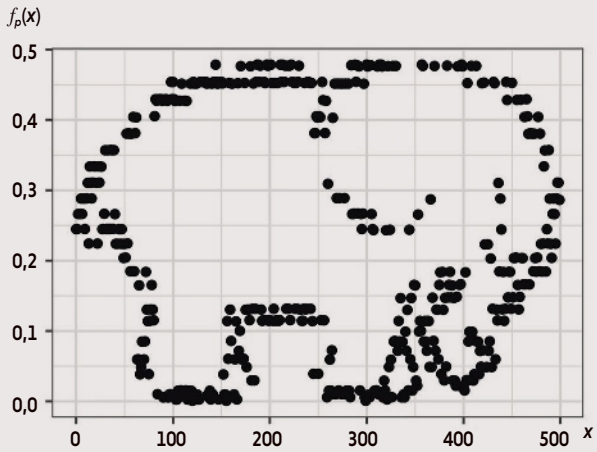
UN SEUL PARAMÈTRE POUR L'ÉLÉPHANT

1

Steven Piantadosi a proposé la fonction : $f_p(x) = \sin^2(2^{kx} \arcsin(\sqrt{p}))$, où le nombre réel p est un paramètre, et a montré qu'elle présente une extraordinaire propriété d'ajustement. La valeur de k est fixée une fois pour toutes, et indique la précision avec laquelle on approchera les données. On considère par exemple des valeurs $y_0, y_1, \ldots, y_n$ telles que les points de coordonnées $(0, y_0), (1, y_1), \ldots, (n, y_n)$ dessinent un éléphant, en serrant assez les abscisses et en dessinant des points assez gros.

La méthode de Steven Piantadosi (expliquée dans les encadrés 3 et 4) permet alors de déterminer la valeur du paramètre p pour que : $f_p(0) = y_0$, $f_p(1) = y_1$, ..., $f_p(n) = y_n$, autrement dit pour que le graphe de $f_p(x)$ dessine un éléphant. Un seul paramètre suffit pour avoir un éléphant ! Bien sûr, on peut obtenir tous les dessins qu'on veut en changeant la valeur de p.

Laurent Boué a écrit des programmes disponibles sur internet qui permettent le calcul de p pour les dessins souhaités et a fait les calculs pour un oiseau et une tortue *(voir la bibliographie)*.



$p = 0{,}244684726673474545 82\ldots$

$p = 0{,}74933466\ldots$

$p = 0{,}70704013\ldots$

> fonction $f_p(x)$ pour $x = 0, 1, ..., n$ les fera remonter une à une : quand x vaudra 0, y_0 remontera, quand x vaudra 1, y_1 remontera, etc.

L'encadré 3 explique que cette remontée des valeurs approchées de $y_0, y_1, ..., y_n$ s'opère indirectement grâce à la fonction $2x$. Quand on applique la fonction $2x$ à un nombre réel écrit en base 2, par exemple $x = 0{,}110011101111_2$, elle fait remonter tous les chiffres vers la gauche : $2x = 1{,}10011101111_2$ (le 2 en indice indique qu'il s'agit d'une écriture en base 2).

Cette propriété est équivalente, pour la base 2, à ce que l'on connaît bien en base 10 : multiplier par 10 décale toutes les décimales d'un chiffre vers la gauche. Par exemple, $10 \times 3{,}1415926... = 31{,}415926...$

En utilisant la fonction qui enlève ce qui est devant la virgule, la fonction $(x \bmod 1)$, parfois appelée « partie fractionnaire de x », on a alors un moyen de faire remonter les chiffres binaires d'un nombre x donné en effaçant ceux qui sont placés en tête. Posons en effet $D(x) = 2x \bmod 1$, et prenons par exemple $x = 0{,}1100111011110101_2$. On a alors :

$D(x) = 0{,}1001110111101012$,
$D(D(x)) = 0{,}0011101111010_12$,
$D(D(D(x))) = 0{,}0111011110101_2$, etc.

En exploitant ce moyen de cacher des informations dans un paramètre réel p, ainsi que quelques autres astuces (*voir l'encadré 4*), on arrive à la fonction qui, avec le bon choix de p, donne y_0 pour $x = 0$, y_1 pour $x = 1$, ..., y_n pour $x = n$ avec une précision fixée à l'avance. Cela montre qu'un seul paramètre p et une seule fonction $f_p(x)$ permettent d'approcher toutes les données que l'on veut, quel que soit leur nombre, pourvu qu'il soit fini !

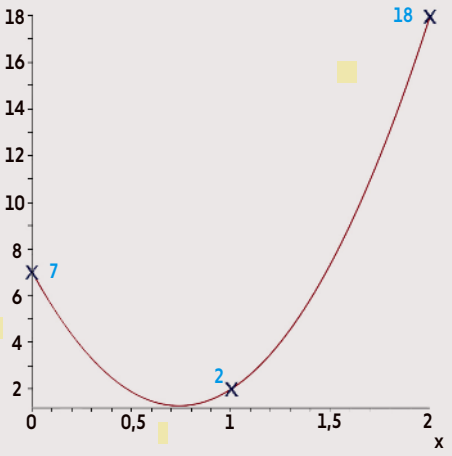
Ainsi, pas besoin de quatre paramètres pour dessiner un éléphant : avec un paramètre unique, on peut s'arranger, et la même fonction servira toujours, en changeant p, pour dessiner un poisson, une tortue, un oiseau ou tout ce que l'on veut.

NOMBRE DE PARAMÈTRES ET COMPLEXITÉ DES MODÈLES

On mesure parfois la complexité d'un modèle en comptant le nombre de paramètres utilisés par le modèle. C'est une façon de procéder qui ne peut pas se justifier sans précisions supplémentaires sur la nature du modèle. En effet, la fonction $f_p(x)$ de Steven Piantadosi montre que l'on peut toujours se ramener à un seul paramètre. Si l'on veut donner un sens au décompte des paramètres d'un modèle, il faut détailler les contraintes qu'on impose au modèle, par exemple « ne faire intervenir que des polynômes » ce qui interdira d'accepter la fonction $f_p(x)$ ou l'une de ses variantes.

Il faut insister sur l'idée que l'article de Steven Piantadosi ne prétend nullement

2 L'INTERPOLATION



Imaginons que l'on veuille trouver un polynôme P(x) qui prenne la valeur 7 pour $x = 0$, la valeur 2 pour $x = 1$, et la valeur 18 pour $x = 2$. Voici une méthode bien connue qui permet de le construire et qui se généralise à un nombre quelconque de données.

On écrit : $P(x) = 7(x-1)(x-2)/[(0-1)(0-2)] + 2(x-0)(x-2)/[(1-0)(1-2)] + 18(x-0)(x-1)/[(2-0)(2-1)]$.

Le premier terme est nul pour $x = 1$ et $x = 2$, le second pour $x = 0$ et $x = 2$, le troisième pour $x = 0$ et $x = 1$. Donc, seul le premier terme n'est pas nul pour $x = 0$, seul le second n'est pas nul pour $x = 1$, seul le troisième n'est pas nul pour $x = 2$.

De plus, les facteurs 7, 2 ou 18 au début de chaque terme assurent que le terme non nul pour $x = 0$ prenne la valeur 7, pour $x = 1$ prenne la valeur 2, pour $x = 2$ prenne la valeur 18. C'est ce que l'on recherchait. En développant et en simplifiant, on trouve :

$P(x) = 21x^2/2 - 31x/2 + 7$.

On vérifie aisément que $P(0) = 7$, $P(1) = 2$, $P(2) = 18$. La méthode se généralise et permet, si $y_0, y_1, ..., y_n$ sont des valeurs données, de construire sans mal un polynôme qui prendra la valeur y_0 pour $x = 0$, y_1 pour $x = 1$, ..., y_n pour $x = n$. Telle est la « méthode d'interpolation de Lagrange ».

Elle montre qu'autoriser beaucoup de paramètres, ici les coefficients du polynôme, permet sans mal de faire passer un polynôme par les valeurs souhaitées, quelles qu'elles soient.

Cependant, en général, la fonction trouvée oscillera beaucoup, alors qu'on préférerait une fonction aussi régulière et stable que possible : son ajustement aux valeurs voulues est exact, mais insatisfaisant. Il existe de meilleures méthodes permettant d'approcher les valeurs voulues en limitant les oscillations de la fonction résultat.

Retenons qu'interpoler exactement est facile, mais que cela ne satisfait pas en général le désir de trouver des lois lisses et régulières.

Cette logique de l'interpolation de Lagrange est poussée jusqu'à l'absurde dans les résultats présentés dans le texte principal : la méthode découverte par Steven Piantadosi, bien plus puissante que l'interpolation de Lagrange, permet, en changeant un seul paramètre p d'une fonction $f_p(x)$, d'obtenir que pour des valeurs quelconques $y_0, y_1, ..., y_n$, on ait :

$f_p(0) = y_0$, $f_p(1) = y_1$, ..., $f_p(n) = y_n$.