MATHÉMATIQUES ET SCIENCES HUMAINES

Problèmes d'enseignement

Mathématiques et sciences humaines, tome 33 (1971), p. 57-71

<http://www.numdam.org/item?id=MSH_1971__33__57_0>

Math. Sci. hum. (9e année, n° 33, 1971, p. 57-61)

L'ALGORITHMIQUE ET L'ENSEIGNEMENT DU SECOND DEGRÉ [1]

par

B. JAULIN[2]

En 1902, on instaura en France deux ordres d'enseignement dans le second degré. Le but de l'un, selon G. Gendarme de Bevotte « était de former l'homme en cultivant ce qu'il y a de plus humain et de plus général en lui, à savoir le jugement et la raison; le but de l'autre était de développer les aptitudes et les facultés particulières de chacun ».

Bien que cette opposition entre humanisme et sciences et techniques ne soit plus de mise ou ne se pose plus dans ces termes, on peut avoir envie d'aborder le problème de l'enseignement de l'informatique par référence à elle, indépendamment de l'importance prise par les machines à calculer dans les divers secteurs de l'économie et de la recherche, ou du souci d'avoir un enseignement qui donne la possibilité de vivre avec son temps.

Mais une telle entreprise dépasse mes intentions, comme d'ailleurs mes moyens. Signalons simplement qu'en 1815 déjà, H. Wronski, mathématicien philosophe, mettait l'accent, dans son traité sur la « philosophie de la technie algorithmique », sur l'opposition entre l'algorithmie (ou algèbre) et « l'algorithmique » mot ancien qui préfigure celui que l'usage des ordinateurs modernes a créé, l'informatique.

Je me contenterai donc de montrer certaines vertus de l'algorithmique pour la définition d'exercices dans des disciplines variées sans me préoccuper de savoir si leur difficulté n'excède pas celle autorisée dans l'enseignement du second degré.

A cette fin, je présenterai trois types d'exercices de construction d'algorithmes que la mathématique enseignée dans le second degré laisse (ou laissait) entrevoir, puis j'évoquerai quelques algorithmes relatifs à des données non numériques et enfin, afin de mettre en évidence la relation qui existe entre un programme et la fonction que celui-ci permet de calculer, je définirai le principe du fonctionnement des ordinateurs modernes en indiquant la parenté de ce mécanisme avec celui de la démonstration dans une théorie formelle de l'arithmétique.

Ces exercices peuvent être traités dans le cadre d'un modèle théorique indépendamment de la pratique des calculatrices actuelles. Bien que la question de savoir si l'on peut ou non enseigner un langage de programmation réel comme Fortran dans le second degré est peut-être primordiale, je définirai la notion de mécanisme en utilisant un langage de programmation rudimentaire permettant

1. Texte d'un exposé présenté au congrès organisé en mars 1970 par l'O.C.D.E., sur le thème : « Enseignement de l'informatique dans le second degré ».

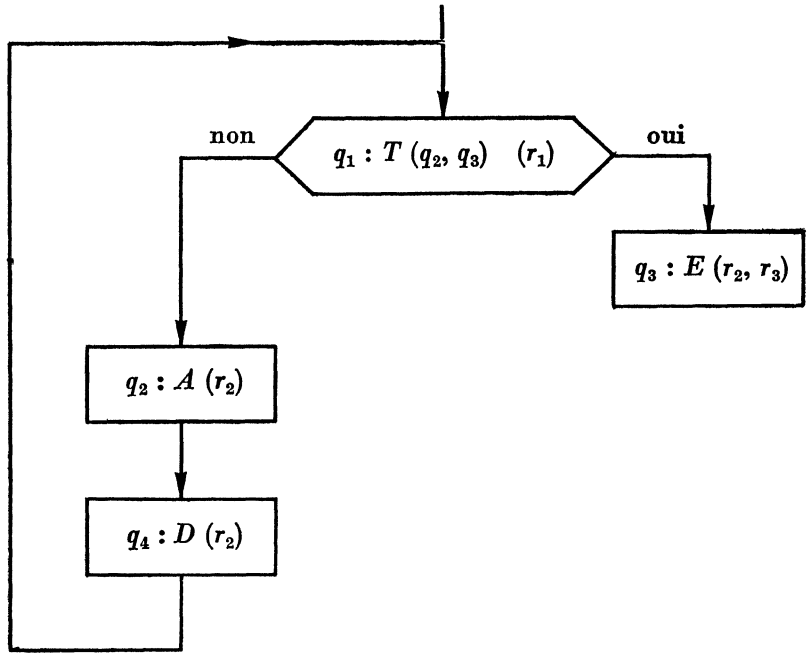
2. Ecole Pratique des Hautes Etudes. VIe Section.

de définir le fonctionnement de n'importe quelle machine. Les exercices que l'on considèrera alors seront l'occasion d'enrichir ce langage en utilisant la notion de sous-programme et également, lorsque l'on considèrera des algorithmes « non numériques », d'introduire des instructions relatives à des données différentes des nombres. Il est possible que l'on puisse arriver ainsi, dans les classes terminales, à une famille d'instructions aussi riche que celle d'un langage de programmation réel sans toutefois, évidemment, avoir utilisé des instructions de format, d'entrée et de sortie, etc., nécessaires pour l'usage des ordinateurs.

Le modèle le plus simple de mécanisme que l'on peut utiliser est inspiré par la structure des calculatrices et est constitué de boîtes ou registres de mémoire dans lesquels on place des nombres[1] que l'on modifie à l'aide « d'instructions » servant à écrire des programmes. Quelles instructions choisir ou, en d'autres termes, quel langage de programmation introduire pour faire apparaître les notions de sous-programme, d'organigramme, à propos de problèmes relativement simples ? On peut se contenter des quatre instructions élémentaires suivantes, qui sont suffisantes pour décrire n'importe quel algorithme de calcul[2].

Ajouter 1 ou enlever 1 au nombre contenu dans la boîte de numéro r (A (r) ou D (r)), porter dans la boîte de numéro r_1 le nombre contenu dans la boîte de numéro r_2 (E (r_1, r_2)),« tester » si le nombre contenu dans une boîte r vaut 0 ou non [T (q_1, q_2) (r)].

A l'aide de ces instructions on peut décrire, par exemple, le fonctionnement d'une machine telle que si x et y sont les nombres placés initialement dans les registres r_1 et r_2, la machine s'arrêtera avec le nombre $x + y$ dans le registre r_3. Un tel exercice met immédiatement en évidence la notion d'organigramme, c'est-à-dire la description sous forme de schéma de la succession d'opérations explicitée dans la phrase suivante : « Regarder si le nombre contenu dans la boîte de numéro r_1 est 0, si oui porter le nombre contenu dans la boîte de numéro r_2 dans le registre r_3 et s'arrêter ensuite, sinon diminuer de 1 le nombre contenu dans la boîte r_1, ajouter 1 au nombre contenu dans la boîte r_2 et recommencer... ».



1. On considère en premier lieu des fonctions calculables de $\mathbf{N}^p$ dans N, N désignant l'ensemble des entiers naturels.

2. De façon plus précise : Pour toute semi-fonction récursive f à p arguments, il existe une infinité de programmes écrits à l'aide de ces instructions de telle sorte que si $x_1 \ldots x_p$ sont les nombres entiers placés dans le registre 1 à p avant calcul, la machine s'arrêtera avec f ($x_1 \ldots x_p$) dans le registre du n° 1 si f ($x_1 \ldots x_p$) est défini et sinon la machine ne s'arrêtera pas.

Traduire cet organigramme sous la forme d'une suite finie d'instructions est un exercice qui ne présente pas de difficultés essentielles dans la mesure où toutes les opérations que l'on fait figurer dans les cases de ce dessin correspondent à des instructions du langage de programmation que l'on utilise. On est donc conduit à introduire la notion de sous-programme c'est-à-dire la possibilité d'écrire dans un programme sous la forme d'une seule instruction, par exemple $[r_1, r_2] \xrightarrow{f} [r_3]$, une suite finie d'instructions dont l'effet serait, dans notre cas, de placer dans le registre r_3 la valeur de la fonction f calculée sur les nombres contenus dans les registres r_1 et r_2. Ainsi, si l'on ajoute l'instruction $[r_1, r_2] \xrightarrow{+} [r_3]$ au langage L_0 constitué des 4 instructions élémentaires précédentes, il n'est pas difficile d'expliciter l'organigramme du programme d'une machine permettant de calculer la multiplication, le p.p.c.m. ou, le p.g.c.d. de deux nombres, puis en utilisant éventuellement d'autres instructions, écrire le programme de calcul de la $n^{ième}$ décimale de la racine carrée ou cubique d'un nombre entier et même, en prenant certaines précautions relatives à la façon de placer les données, expliciter sous forme d'organigramme le procédé de division de deux polynomes à coefficients entiers, faire des calculs modulo un nombre quelconque, etc. Le calcul des fonctions élémentaires de l'arithmétique fournit donc des exemples simples pour l'introduction des notions de programme, de sous-programme, d'organigramme et l'algorithmique permet la description, dans un langage précis, des algorithmes introduits dans l'enseignement du second degré comme l'algorithme d'Euclide, etc. C'est l'occasion d'utiliser un langage de programmation du type de ceux des machines à calculer de bureau et ce serait un avantage certain si les élèves pouvaient avoir accès à de tels appareils, ne serait-ce que pour vérifier la correction de leur programme. D'un autre côté, ces diverses notions de l'informatique peuvent être utilisées pour expliciter la définition de certaines opérations que l'on effectue couramment sur les fonctions usuelles de l'arithmétique, comme la récurrence, la composition, etc. En effet, si l'on demande par exemple d'écrire le programme d'une machine permettant de calculer la fonction $f(x, y) = y!(x + 1)$ à l'aide d'un langage de programmation comportant les quatre instructions élémentaires et les instructions $[r_1, r_2] \xrightarrow{x} [r_3]$ et $[r'] \xrightarrow{suc} [r_3]$ (suc désignant la fonction successeur : $suc(x) = x + 1$), et si ensuite on remplace dans l'organigramme obtenu l'instruction $[r_1, r_2] \xrightarrow{x} [r_3]$ et $[r_1'] \xrightarrow{suc} [r_3']$ par deux instructions $[r_1, r_3] \xrightarrow{g} [r_3]$ et $[r_1'] \xrightarrow{k} [r_3']$ signifiant placer dans le registre r_3 (resp. r_3') le nombre $g(< r_1 >, < r_2 >)$ (resp. $k(< r_1' >)$) $< r_t >$ étant le nombre contenu dans le registre r_t et k et g des symboles désignant des fonctions quelconques, on obtiendra ainsi, si le programme initial a été bien construit, un programme explicitant la façon de calculer la valeur d'une fonction obtenue par récurrence à partir de deux fonctions k et g. Il en va de même pour la composition de deux fonctions et, plus généralement pour la plupart (sinon toutes) des opérations définies sur les fonctions que l'on rencontre dans la pratique de l'arithmétique élémentaire et dont la définition purement mathématique est parfois rebutante. Enfin, on peut faire apparaître certaines techniques de l'analyse dans la mesure où les fonctions les plus usuelles, celles que l'on introduit dans les classes terminales, sont pour la plupart calculables au sens suivant : soit L_1 le langage de programmation obtenu en ajoutant aux 4 instructions élémentaires, l'instruction $[r_1] \xrightarrow{f} [r_3]$ signifiant comme précédemment : placer dans le registre r_3 la valeur de la fonction f pour l'argument contenu dans le registre r_1. Si l'on se rappelle qu'un nombre irrationnel x peut être vu comme une application f_x de N dans N en posant $f_x(n)$ = le nombre défini par les n premiers chiffres dans l'expression décimale de x, on se rend compte aisément que l'on peut écrire un programme dans ce langage L_1 permettant de calculer par exemple la valeur de e^x pour un nombre irrationnel x quelconque en ce sens que si n est l'entier naturel placé dans le registre 1 la machine s'arrêtera avec dans le registre 1 le nombre défini par les n premiers chiffres de l'expression décimale de e^x lorsque dans l'instruction de sous programme $[r_1] \xrightarrow{f} [r_3]$ on donne à f la valeur f_x. Écrire de tels algorithmes dans le langage L_1 fait en général appel aux formules permettant d'évaluer une majoration du « reste » que l'on obtient lorsqu'on se limite à la somme des n premiers termes dans le développement en série entière de la fonction que l'on évalue, mais on peut également utiliser d'autres procédures de calcul.

Ces trois types d'algorithmes ne sont pas les seuls que l'on rencontre dans la mathématique du secondaire. Les procédures de résolutions approchée des équations constituent une autre famille, etc.,

mais nous allons maintenant rechercher, en dehors de la mathématique enseignée dans le second degré, d'autres exemples de procédure de calcul.

Les algorithmes non numériques, c'est-à-dire ceux se rapportant à des entités qui ne sont pas perçues comme des nombres, par exemple des mots sur un alphabet fini, etc., sont nombreux. Ils sont généralement plus complexes que les précédents; indiquons en deux exemples : l'algorithme de coupure des mots de la langue française bien connu des informaticiens qui se préoccupent de composition automatique des textes (bien que les règles données dans les grammaires traditionnelles soient insuffisants pour le construire), l'algorithme permettant de reconnaître si une expression de l'arithmétique ou de la mathématique formalisée est bien formée et dont la construction a l'avantage de mettre en évidence le poids des divers symboles que l'on emploie et également, la structure syntaxique d'une expression. Ce dernier exercice appartient à la classe de ceux que l'on peut imaginer à propos de problèmes de caractère combinatoire tels qu'ils se posent en linguistique « formelle ». Les instructions qu'il faut utiliser pour écrire de tels programmes n'ont évidemment pas le caractère arithmétique des instructions précédentes. Ces exercices peuvent être l'occasion d'introduire des « macro-instructions » du type de celles que l'on rencontre dans un langage comme Fortran, par exemple. D'autre part, il faut admettre que l'on puisse placer dans les boîtes ou registres de mémoire non pas des nombres mais des symboles ou des mots construits sur un alphabet fini, c'est-à-dire en fait, introduire un système de représentation des nombres. Ceci apparaît clairement dans l'exemple suivant d'un « additionneur » : on suppose que les symboles utilisés sont *0* et 1, et l'on demande d'écrire dans un langage de programmation dont les instructions correspondent à la « conjonction » « négation », « coordination », etc., le programme d'une machine telle que si x et y sont les mots placés dans les boîtes 1 et 2 la machine s'arrêtera avec dans le registre 3 le mot qui est la représentation en binaire du nombre somme de ceux dont x et y sont les représentations binaires. On voit dans cet exemple, si on se limite à des mots de longueur n, que les boîtes 1 et 2 peuvent être matérialisées par les $2n$ fils d'entrées d'un « circuit additionneur » et la boîte n° 3 par les $n + 1$ fils de sortie d'un tel mécanisme.

J'emprunterai enfin le dernier exemple d'algorithme à l'informatique elle-même afin de mettre en évidence le principe du fonctionnement des ordinateurs modernes, indépendamment de la nature des circuits électroniques à l'aide desquels on les réalise. Reprenons les quatre instructions élémentaires du début. Un programme écrit à l'aide de ces quatre instructions peut s'écrire sous la forme d'un mot construit sur un alphabet à 10 symboles X = { A, D, E, T, (,), q, |,; , *, } si l'on convient d'utiliser l'astérisque pour séparer deux instructions et n barres verticales pour représenter le nombre n. Si au lieu d'utiliser ces symboles on utilise les chiffres 1, 2, ..., 9, *0*, un programme correspondra alors à un nombre, celui dont la représentation décimale est donnée par l'écriture du programme à l'aide de ces chiffres.

Soit alors P un programme. On désigne par $\overline{P}$ le nombre que définit son écriture avec les conventions précédentes et par f la semi-fonction de 1 argument que la machine de programme P permet de définir lorsque l'on place l'argument dans la boîte n° 2 et que l'on prend le résultat dans la boîte de n° 1. Le problème consiste à écrire le programme d'une machine telle que si $\overline{P}$ est placé dans le registre 1, l'entier x dans le registre 2, la machine s'arrêtera avec $f(x)$ dans le registre 1 si $f(x)$ est défini et sinon la machine ne s'arrêtera pas[1].

Un tel programme existe et peut être écrit à l'aide des quatre instructions élémentaires. Toutefois, la simplicité du langage L_0 rend inutilement compliqué le travail de programmation. Aussi est-il préférable de travailler avec un langage plus riche, comprenant notamment une instruction d'adressage indirect, I (r), consistant à ajouter 1 au nombre contenu dans le registre dont le numéro est le nombre contenu dans le registre r. Regardons brièvement en quoi consiste son organigramme; il comprend essentiellement deux parties : une analyse « syntaxique » (de la représentation décimale) du nombre P

1. Lorsque l'on spécifie les registres dans lesquels on place les arguments avant calcul et le registre où l'on prend le résultat, une machine, en général définit une semi-fonction de $\mathbb{N}^q$ dans N, c'est-à-dire une fonction définie sur une partie X de $\mathbb{N}^q$. Si l'argument appartient au complémentaire de X, le calcul se poursuit indéfiniment, on dit alors que la machine ne s'arrête pas.

afin de reconnaître si ce nombre correspond ou ne correspond pas à l'écriture d'un programme puis, la description du fonctionnement des calculatrices qui consiste à appliquer sur ces arguments un programme enregistré en mémoire. Ainsi, en première approximation, ce programme correspond à la description, dans un certain langage de programmation, du « hardware » des ordinateurs actuels. En choisissant bien le langage que l'on utilise pour écrire ce programme (dont les instructions devraient correspondre aux opérations cablées) et en admettant que le langage « utilisateur », c'est-à-dire celui qui sert à écrire les programmes que l'on enregistre dans la machine, soit celui à quatre instructions introduit précédemment, l'organigramme auquel on est conduit ne présente pas de difficultés excessives.

Cet exercice met clairement en évidence la relation existant entre un programme quelconque et la semi-fonction qu'il permet de calculer dans certaines conditions. En effet, le programme que l'on vient d'évoquer, c'est-à-dire celui décrivant le fonctionnement d'un ordinateur permet de calculer la semi-fonction à deux arguments $\varphi(x, y)$ ou $\varphi(x, y)$ est la valeur pour l'argument y de la semi-fonction définie par le programme de numéro x.

On constate alors que ce fonctionnement n'est pas sans analogie avec le mécanisme de démonstration dans l'arithmétique de Peano en un sens que nous allons préciser maintenant, ceci à titre de curiosité, mais évidemment pas en vue de la définition d'un exercice : l'Arithmétique de Peano est une théorie formelle du 1^0 ordre et chaque formule de cette théorie peut être considérée comme un nombre pour la même raison qui nous a fait considérer un programme comme un nombre, car une telle formule est un mot sur un alphabet fini. Si $F(z)$ est une formule à une variable libre de cette théorie, désignons par $\overline{F(z)}$ le nombre qui lui correspond dans cette numérotation et soit $B = \{(x, y)/x \in \mathbf{N}, y \in \mathbf{N}$ et $F(\overline{y})$ est un théorème de la théorie lorsque l'on substitue dans la formule $F(z)$ à une variable libre de numéro x, la variable z par le symbole $\overline{y}$ qui représente le nombre naturel y dans le langage de cette théorie $\}$. B est donc le sous-ensemble de $\mathbf{N}^2$ tel que pour tout $x \in \mathbf{N}$, $B(x) = \{y/(x, y) \in B\}$ est le sous-ensemble de N défini par la formule $F(z)$ de n° x par l'intermédiaire de la « démonstration » dans l'arithmétique de Peano c'est-à-dire tel que $y \in B(x)$ si et seulement si, $F(\overline{y})$ est *démontrable* dans cette théorie. De la même façon, si $A = \{(x, y)/\varphi(x, y)$ est défini $\}$ alors $A(x) = \{y/(x, y) \in A\}$ est le sous-ensemble de N défini par le programme P de numéro x en ce sens que $y \in A(x)$ si et seulement si, la machine dont le programme est P s'arrête lorsqu'on *l'applique* sur l'argument y. Cette analogie n'est pas de pure forme. Dans les deux cas, on étudie une relation qui existe entre le nom d'un objet (un programme ou une formule) et l'objet (un sous-ensemble de N) défini par l'intermédiaire d'un certain mécanisme (celui des ordinateurs, celui de la démonstration formelle) et l'on montre que A et B sont des sous-ensembles récursivement énumérables de $\mathbf{N}^2$ universelles et sont donc récursivement isomorphes. Ainsi, c'est à partir de l'analyse du mécanisme de la démonstration dans une théorie formalisée que plusieurs logiciens, Markov, Post, Turing ont introduit différents types d'algorithmes dont certains, notamment les machines de Turing, ont une structure très voisine de ceux que l'on a utilisé, à ceci près que l'on considère là un seul registre et que les instructions des machines de Turing sont définies dans les termes des symboles à l'aide desquels on représente les nombres.

Après cette digression et en conclusion à ces différents types d'algorithmes que j'ai essayé de présenter dans un cadre unique en recherchant les liens qu'ils pouvaient avoir avec l'enseignement traditionnel du second degré, indiquons qu'un exercice d'algorithmique, bien que ne présentant pas de difficulté conceptuelle très grande, notamment si l'on est guidé dans le choix des instructions à utiliser pour construire un organigramme, exige une très grande attention dans la mesure où la traduction dans un langage de programmation de la description d'un procédé de calcul requiert une précision « totale ». Aussi il me semble nécessaire de prendre certaines précautions si l'on veut définir des exercices d'algorithmique accessibles à des élèves du second degré. Le vocabulaire technique que j'ai parfois utilisé m'a fait oublier cette contrainte, m'a éloigné également de la pratique des ordinateurs actuels et d'une discussion sur le but d'un enseignement de l'informatique dans le second degré, ce que chacun est libre de considérer comme étant un avantage ou un inconvénient.

Math. Sci. hum. (9e année, n° 33, 1971, p. 63-71)

APPLICATIONS PRATIQUES DES LOIS DE PROBABILITÉ (10)

par

B. LECLERC

LOI NORMALE
LOI LOG-NORMALE
LOIS DE PEARSON DE TYPES 3 ET 5 (LOI GAMMA) *HYDROLOGIE*
LOIS DE HALPHEN

MORLAT, G., "Les lois de probabilité de Halphen", *Revue de Statistique Appliquée* IV (3), 1956, pp. 21-43.

Dans cette conférence, l'auteur présente les travaux de E. Halphen sur la recherche de lois de probabilité auxquelles peuvent s'ajuster les séries d'observations hydrométriques (débits de rivières, par exemple), et les applications de ces travaux. A cette occasion, l'auteur fait un tableau général des lois utilisées dans ce domaine.

Modèles

Loi normale (de densité non précisée).

Loi log-normale, ou loi de Galton-Gibrat à trois paramètres positifs u_0, m, σ, de densité.

$$f(u) = \frac{1}{\sqrt{2\pi}\,\sigma\,u} e^{-\frac{1}{2}\left(\frac{\log(u-u_0)-m}{\sigma}\right)^2} \qquad u \geqslant u_0$$

la variable $x = \log(u - u_0)$ suit une loi normale N (m, σ).

Loi de Pearson de type 3 à deux paramètres $a > 0, \gamma > 0$ (loi gamma), de densité :

$$f(x) = \frac{1}{a^\gamma\,\Gamma(\gamma)} e^{-ax}\,x^{\gamma-1} \qquad x \geqslant 0$$

où Γ est la fonction eulérienne gamma.

Loi de Pearson de type 5 à deux paramètres $b > 0, \gamma < 0$, de densité :

$$f(x) = \frac{b^\gamma}{\Gamma(-\gamma)} e^{-\frac{b}{x}}\,x^{\gamma-1} \qquad x \geqslant 0$$

Loi de Halphen de type A à trois paramètres μ et a positifs, γ quelconque, de densité :

$$f(x) = \frac{1}{2\,\mu^{\gamma}\,K_{\gamma}(a)}\, e^{-\frac{a}{2}\left(\frac{x}{\mu}+\frac{\mu}{x}\right)}\, x^{\gamma-1} \qquad x \geqslant 0$$

où K_{γ} est la fonction de Bessel d'ordre γ.

Loi de Halphen de type B à trois paramètres, ν et α positifs, b quelconque, de densité :

$$f(x) = \frac{2}{\nu^{2\alpha}\,ef_{\alpha}(b)}\, e^{-\left(\frac{x}{\nu}\right)^2 + b\frac{x}{\nu}}\, x^{2\alpha-1} \qquad x \geqslant 0$$

où ef_{α} est une fonction de Hermite, ou fonction factorielle.

Loi de Halphen de type B^{-1}, introduite par M. Larcher, pour compléter le système des lois de Halphen, à trois paramètres ν et α positifs, b quelconque, de densité :

$$f(x) = \frac{2}{\nu^{-2\alpha}\,ef_{\alpha}(b)}\, e^{-\left(\frac{x}{\nu}\right)^2 + b\frac{\nu}{x}}\, x^{-2\alpha-1} \qquad x \geqslant 0$$

On note que si la variable aléatoire u suit une loi B, alors $\frac{1}{u}$ suit une loi B^{-1} et si u suit une loi A, il en est de même de $\frac{1}{u}$. Par ailleurs, les lois A, B, B^{-1} forment une classe complète de lois à 3 paramètres. Si l'on se donne les trois premiers moments, on peut trouver une loi de Halphen de type A, ou B ou B^{-1} admettant ces premiers moments.

Estimation

Seules les estimations des paramètres des lois de Halphen sont étudiées ici.

Soient x_i, $i = 1, \ldots, n$ les observations, et les statistiques :

moyenne arithmétique m	$m = \frac{1}{n}\sum x_i$
moyenne quadratique q	$q^2 = \frac{1}{n}\sum x_i^2$
moyenne géométrique g	$\log g = \frac{1}{n}\sum \log x_i$
moyenne harmonique h	$h = \frac{1}{n}\sum \frac{1}{x_i}$
moyenne quadratique inverse k	$\frac{1}{k^2} = \frac{1}{n}\sum \frac{1}{x_i^2}$

Pour l'estimation des paramètres, un résumé exhaustif des observations est fourni par :

m, g et h pour les lois de type A

m, g et q pour les lois de type B

g, h et k pour les lois de type B^{-1}.

On identifie les moyennes constituant les résumés exhaustifs avec les moyennes théoriques correspondantes. Les fonctions nécessaires sont tabulées, ou en cours de tabulation au moment de la rédaction de l'article.

Pour les lois de type A, on dispose également d'un abaque.

On peut représenter une loi de Halphen par un point du plan (cv, λ_0), où :

$$\text{cv} = \frac{\sigma}{m} = \frac{\sqrt{q^2 - m^2}}{m} \qquad \text{et} \qquad \lambda. = \frac{m - g}{\sigma}.$$

On a :

$$\text{cv} > 0, \lambda_0 > 0 \qquad \text{et} \qquad \lambda_0, \text{cv} < 1.$$

Les quantités cv et λ_0 se prêtent bien à un ajustement de données observées à une loi de Halphen de type non déterminé à l'avance. L'ajustement est le meilleur pour le type B et probablement d'une bonne efficacité pour le type A. Pour le type B^{-1}, on peut ajuster l'inverse de la variable.

Applications

(Séries des débits observés, année par année, sur une même période de l'année.)

L'utilisation de la loi normale est d'autant meilleure que la série porte sur les débits moyens pris sur une durée assez longue, que l'aire du bassin versant est importante, que l'altitude de ce bassin est élevée. On a, dans ces cas, des phénomènes de régularisation. Ainsi, la loi normale a, selon l'auteur, été utilisée presque exclusivement, et avec succès, pour les travaux de prévision des apports d'été aux réservoirs alpins et pyrénéens.

La loi log-normale a également donné de bons résultats dans de nombreux cas; cependant, sa décroissance pour les grandes valeurs de la variable est trop rapide. De plus, u_0, borne inférieure des débits, est difficile à estimer et d'existence discutable.

Les lois de Pearson de type 3 et 5 ont également été employées, spécialement les lois de type 3, par E. E. Foster.

Pour étudier sa classe de lois, E. Halphen note que deux paramètres (échelle et dispersion) sont insuffisants pour caractériser une distribution de débits. C'est pourquoi, il a établi sa famille de lois à trois paramètres, à partir de considérations sur les données observées.

Les lois A ont rendu des services pour l'évaluation des probabilités des valeurs mensuelles des débits dans les régions hydroélectriques Alpes, Pyrénées, Massif Central. L'introduction des lois B^{-1} a été nécessaire par suite de l'irrégularité des débits d'été des rivières du Massif Central. Une étude systématique des débits mensuels pour chacun des 12 mois de l'année a été effectuée dans les stations de l'Annuaire Hydrologique de la France.

L'auteur termine en évoquant le difficile problème de l'évaluation des probabilités de fortes crues. Beaucoup de lois de probabilité sont au dire des hydrauliciens « trop optimistes », sous-estimant cette probabilité. Certaines lois de type B^{-1} ne semblent pas avoir ce défaut. En particulier, pour évaluer les probabilités de crues de la Durance à Serre-Ponçon, l'auteur et ses collaborateurs ont construit une loi empirique qui s'est révélée après coup être très proche d'une loi B^{-1}.

BIBLIOGRAPHIE

GIBRAT, R., "Aménagement hydro-électrique des cours d'eau : statistique mathématique et probabilités", *Rev. gén. Hydraul.*, septembre-octobre 1936, pp. 586-595.

COUTAGNE, A., "Étude statistique et analytique des crues du Rhône à Lyon", *Congrès pour l'utilisation des eaux*, Lyon 1938.

MASSE, P., "Situation, perspectives et applications de l'hydrologie statistique", *Annu. hydrol. Fr.*, 1940.

HALPHEN, E., "Sur un nouveau type de courbes de fréquence", *C. R. Acad. Sci., Paris*, 10 novembre 1941.

LE CAM et MORLAT, G., "Les lois des débits des rivières françaises", *Houille Blanche*, numéro spécial B, 1949.

HALPHEN, E., "Les fonctions factorielles", *Publ. Inst. Statist. Univ. Paris*, vol. IV, fasc. 1, 1955.

FOSTER, E. E., *Rainfall and runoff* (sans autre précision).

LOI EXPONENTIELLE
LOI DE HALPHEN
LOI NORMALE — *HYDROLOGIE*
« LOI DES FUITES »

BERNIER, J. et FANDEUX, F., "Théorie du renouvellement : Application à l'étude statistique des précipitations mensuelles", *Rev. Statist. appl.*, XVIII, 2, 1970.

Modèle

Un « processus de renouvellement » peut être défini ainsi : une suite d'événements aléatoires est telle que les intervalles de temps entre deux événements consécutifs sont des variables aléatoires indépendantes et de même distribution. Si $X_1, X_2, \ldots, X_t, \ldots$ est la suite de ces variables, on suppose qu'à chaque X_t correspond une variable aléatoire Y_t, et on considère les variables suivantes :

N_t nombre d'événements survenus dans l'intervalle de temps $[0, t]$.

$$Z_t = \sum_{t=1}^{N_t} Y_t, \qquad \text{avec} \qquad Z_t = 0 \text{ si } N_t = 0.$$

Ce modèle se traite par les transformations de Laplace : la transformée de la densité de Z_t s'obtient à partir des densités des X_t et des couples (X_t, Y_t). Si $f(x)$ (resp. $f(x, y)$) est la densité de X (resp. du couple X, Y) la transformée de Laplace de $f(x)$ (resp. $f(x, y)$) est la fonction :

$$g(s) = \int_0^\infty e^{-sx} f(x)\, dx$$

$$(\text{resp. } g(s, u) = \int_0^\infty \int_0^\infty e^{-sx-uy} f(x, y)\, dx\, dy).$$

Le cas étudié plus particulièrement ici est celui-ci : la loi des X_i est exponentielle de moyenne μ définie par sa fonction de répartition :

$$F(x) = \text{Prob}\,[X_i \geqslant x] = e^{-\frac{x}{\mu}}$$

Les événements correspondants surviennent donc suivant un processus poissonnien. La loi des Y_i est également la loi exponentielle, de moyenne $\frac{1}{\rho}$.

Les Y_i sont indépendantes des X_i et entre elles :

$$\text{Prob}\,[Y_i \leqslant y] = 1 - e^{-\rho y}.$$

On en tire la « loi des fuites » pour la variable Z_t, ainsi nommée parce qu'elle a été utilisée pour représenter la distribution des débits de fuite des joints dans une conduite de gaz. On a :

$$\text{Prob}\,[\,Z_t = 0] = e^{-\frac{t}{\mu}}$$

$$\text{Prob}\,[z \geqslant Z_t \geqslant z + dz] = e^{-\frac{t}{\mu}-\rho z}\sqrt{\frac{\rho t}{\mu z}}\,I_1\left(2\sqrt{\frac{\rho t z}{\mu}}\right)dz$$

où I_1 est la fonction de Bessel modifiée de première espèce.

Cette loi a été tabulée pour diverses valeurs de ses deux paramètres ρ et μ.

D'autre part, la variable $u = \sqrt{2\rho}\left(\sqrt{z} - \sqrt{\frac{\rho}{\mu}}\right)$ a pour loi limite la loi normale réduite centrée lorsque μ tend vers zéro. $\sqrt{z}$ est donc approximativement normale pour μ assez petit.

Estimation

Si $\bar{x}$ et s^2 sont la moyenne et la variance observées, on estime ρ et μ par les formules :

$$\bar{x} = \frac{1}{\rho\,\mu} \qquad \text{et} \qquad s^2 = \frac{2}{\mu\,\rho^2}.$$

On peut aussi, dans le cas où u est approximativement normale, estimer ρ et μ avec la moyenne $m(\sqrt{z})$ et la variance $s^2(\sqrt{z})$ observées de $\sqrt{z}$, par :

$$m(\sqrt{z}) = \sqrt{\frac{\rho}{\mu}} \qquad s^2(\sqrt{z}) = \frac{1}{\sqrt{2\,\rho}}.$$

Application

La « loi des fuites », ou son approximation normale pour la racine des observations, est ajustée à des séries de précipitations mensuelles observées à la station météorologique de Châteauneuf de Randon. X_i est l'intervalle de temps entre deux précipitations successives, Y_i la quantité d'eau tombée correspondante, Z_t le total des précipitations durant la période $[0, t]$.

Quatre graphiques permettent de voir les ajustements des distributions des précipitations pour les mois de février (84 observations), mai, août, septembre (86 observations pour chaque mois) à la « loi des fuites » et à une loi de Halphen non précisée dans l'article (voir fiche correspondante ci-avant). Les deux lois sont très proches, la loi des fuites présentant l'avantage d'une estimation plus aisée des paramètres.

Deux graphiques présentent les ajustements des distributions des précipitations pour février et pour janvier (84 observations) à la loi des fuites et à la loi normale pour la racine des observations. Les deux méthodes d'estimation donnent pratiquement les mêmes valeurs de ρ et μ, et les ajustements sont tous satisfaisants.

La « loi des fuites » sera donc utile aux hydrologues pour décrire les séries chronologiques de précipitations mensuelles. L'hypothèse d'instantanéité d'une observation devient moins justifiable pour des périodes plus courtes (précipitations hebdomadaires), et le modèle de la loi des fuites risque de ne plus être alors aussi adéquat.

LOI NORMALE *INDUSTRIE*

GELAIN, J., "Influence de la vitesse de mise en charge de la machine d'essais sur la résistance à la traction de la fonte", *Rev. Statist. appl.*, I, 1, 1953, pp. 47-53.

Des éprouvettes de fonte sont soumises à des essais de traction : elles sont soumises à une traction progressivement augmentée jusqu'à la rupture. La charge subie est alors observée. Trois diamètres d'éprouvette sont utilisés : 20 mm, 12,5 mm et 5,64 mm et pour chaque diamètre, trois vitesses de mise en charge non précisées.

Modèle

Les distributions des charges de ruptures, pour un diamètre et une vitesse donnés, sont ajustées à la loi normale, de forme analytique non précisée ici.

Estimation et test

Tracés de droites de Henri. Les trois tracés correspondant au diamètre de 12,5 mm sont inclus dans l'article. L'auteur juge les ajustements très satisfaisants. Il en est d'ailleurs de même pour les autres diamètres, pour lesquels les tracés ne sont pas inclus dans l'article.

L'auteur précise en outre que les méthodes statistiques, jugées par le test du χ^2 ont donné également de très bons ajustements à la loi normale.

L'article comporte ensuite une analyse de variance pour la détermination de l'influence de la vitesse de mise en charge sur la charge de rupture. Les tests effectués ayant été loin d'être significatifs, on conclut à l'absence d'influence.

BIBLIOGRAPHIE

BASTIEN, P. et PRACHE, J., "Recherches sur la dispersion des essais mécaniques sur fontes", *Journées de la fonderie*, 19 et 20 octobre 1945, pp. 57-76, Éditions du Centre technique des industries de la fonderie.

LOI DE POISSON
LOI BINOMIALE NÉGATIVE *SOCIOLOGIE*

COLEMAN, James S., "The Poisson process and its contagious relatives", *Introduction to mathematical sociology*, chap. 10 et 11, Londres, Collier-Macmillan, 1964.

1. L'auteur décrit d'abord le processus de Poisson : un événement d'un certain type est susceptible de se produire avec la probabilité $\alpha\ \mathrm{d}\ t$ dans l'intervalle de temps $[t,\ t + \mathrm{d}\ t]$, constante dans le temps et indépendante des événements s'étant déjà produits. Si p_{it} est la probabilité qu'il se produise i événements dans l'intervalle $[0,\ t]$, on en déduit les équations différentielles :

$$\frac{d\ p_{0t}}{\mathrm{d}\ t} = -\ \alpha\ p_{0t}$$

$$\frac{\mathrm{d}\ p_{it}}{\mathrm{d}\ t} = \alpha\ p_{i-1,t} - \alpha\ p_{it}$$

avec pour condition initiale $p_{00} = 1$.

On trouve finalement pour p_{it} la loi de Poisson de moyenne $\alpha\ t$:

$$p_{it} = \frac{(\alpha\ t)^i\ e^{-\alpha t}}{i\ !}\ .$$

L'auteur donne deux exemples. Le premier, dû à Bortkewistch (1898), est célèbre : c'est la distribution du nombre de tués par coups de pied de cheval par corps d'armée dans l'armée prussienne sur une période de vingt ans (voir *Math. Sci. hum.*, n° 25). Le second, dû à Thorndike (1926), est la distribution du nombre d'appels téléphoniques par intervalles de cinq minutes au départ d'un groupe de six cabines publiques. Le comptage a été effectué pendant sept jours entre midi et quatorze heures. L'auteur présente ces données en remarquant que l'indisponibilité des cabines pendant leur utilisation ne concorde pas avec le modèle poissonnien. Ainsi, le nombre maximum d'appels durant une période de cinq minutes est limité. Cependant, l'ajustement (par une méthode non précisée) est jugé assez bon par l'auteur.

2. Une première modification du modèle précédent est la suivante, d'où le « modèle de Poisson avec exhaustion ». Il existe un entier N tel que, si i événements se sont produits, la probabilité de l'occurrence du $(i + 1)$ - ème dans l'intervalle de temps $[t,\ t + \mathrm{d}\ t]$ est $\alpha\ [N - i]$. (Par exemple, l'événement considéré est la disparition d'un élément d'un groupe de taille N.) On aboutit à :

$$p_{i,t} = \binom{N}{i} (1 - e^{-\alpha t})^i\ e^{-\alpha t(N-i)}, \qquad 0 \geqslant i \geqslant N.$$

C'est la loi binomiale : $\mathscr{B}\ (N,\ 1 - e^{-\alpha t})$.

L'auteur donne un exemple auquel le modèle a été appliqué, sans succès toutefois, la valeur de α estimée à partir de la moyenne $N\ (1 - e^{\alpha t})$ du nombre d'événements survenus à l'instant t ne cessant de croître avec t. Il s'agit d'un test psychologique : on observe le nombre de mots (syllabes dénuées de sens) appris dans une liste par un sujet dans le temps t.

3. Le modèle suivant est dénommé « poissonnien avec contagion » par l'auteur. Si i événements se sont produits, la probabilité d'occurrence du $(i + 1)$ -ème dans l'intervalle de temps $[t, t + \mathrm{d}\ t]$ est égale à :

$$(\alpha + i\ \beta)\ \mathrm{d}\ t.$$

d'où :

$$p_{it} = \frac{\alpha\ (\alpha + \beta)\ \ldots\ (\alpha + (i - 1)\ \beta)\ e^{-\alpha t}\ (1 - e^{-\beta t})^i}{i\ !\ \beta^i}\ .$$

La moyenne μ et la variance σ^2 sont :

$$\mu = \frac{\alpha}{\beta}(e^{\beta t} - 1)$$

$$\sigma^2 = \mu\, e^{\beta t}$$

On en déduit l'estimation de α et β par la méthode des moments. On a :

$$\beta = \log \frac{\sigma^2}{\mu}$$

$$\alpha = \frac{\mu^2 \beta}{\sigma^2 - \mu}.$$

Cette loi a été donnée sous une autre forme par Polya, travaillant également sur un modèle de contagion.

En posant :

$$\frac{\alpha}{\beta} = \lambda\, \rho,\ \rho = \frac{1}{e^{-\beta t - 1}} \quad ,$$

on trouve :

$$p_{it} = \binom{\lambda\, \rho + i}{i} \left(\frac{\rho}{1+\rho}\right)^{\lambda\, \rho} \left(\frac{1}{1+\rho}\right)^{i}$$

où les factorielles sont généralisées à $\lambda\ \rho$ non entier par la fonction Γ. On reconnaît la loi binomiale négative, obtenue par divers auteurs (Yule et Greenwood, 1920) à partir de modèles fondés non sur la contagion mais sur l'hétérogénéité de la population étudiée. Ainsi que Feller (1943) l'a démontré, il n'est pas possible, étant donnée une distribution observée s'ajustant bien à cette loi, de fournir une interprétation dans le sens soit de la contagion, soit de l'hétérogénéité.

L'auteur donne deux exemples. Une étude effectuée par lui-même en 1961 : il a demandé à des adolescentes (treize à dix-neuf ans) combien de disques elles avaient acheté au cours du dernier mois. L'auteur présente les données et les ajuste à la loi de Poisson et à la loi binomiale négative.

La distribution observée ne s'ajuste pas à la loi de Poisson. L'ajustement à la loi binomiale négative n'est pas excellent, mais rend bien compte de la forme générale de la distribution (figure incluse dans le texte).

Le deuxième exemple porte sur des données de Hill et Trist (1953) sur des accidents dans l'industrie. La distribution du nombre d'accidents par travailleur dans un établissement industriel pendant une période donnée, s'ajuste très bien à la loi binomiale négative, tandis que le modèle poissonnien doit être rejeté.

L'auteur présente ensuite, sans ajustements, un modèle rassemblant les hypothèses des deux précédents.

4. Au chapitre suivant, la variable étudiée se présente comme la taille d'un groupe pouvant à chaque instant s'accroître ou diminuer d'une unité. On s'intéresse à la distribution de la taille des groupes lorsque l'équilibre est réalisé. Divers modèles sont proposés. Pour deux d'entre eux, l'auteur présente une distribution observée s'ajustant bien au modèle.

Pour le premier, on suppose qu'un groupe de taille i a des probabilités $(\alpha + i\,\gamma)\,\mathrm{d}\,t$ de gagner un élément et $i\,\beta$ d'en prendre un dans l'intervalle $[t, t + \mathrm{d}\,t]$. On en déduit la probabilité pour un groupe d'être de taille i :

$$p_i = \binom{k+i-1}{i} (1-c)^k c^i \qquad \text{avec} \qquad c = \frac{\gamma}{\beta}, \; k = \frac{\alpha}{\gamma}.$$

On retrouve la loi binomiale négative. Les paramètres sont estimés par la méthode des moments, avec :

$$c = 1 - \frac{\mu}{\sigma^2} \qquad k = \frac{\mu^2}{\sigma^2 - \mu}.$$

Ce modèle est appliqué à des données sociométriques. Dans sept maisons de vingt-six habitants chacune d'une école de fille, chaque fille choisit trois membres de sa propre maison. L'auteur présente les données et considère comme relativement bon l'ajustement à la loi binomiale négative du nombre de choix reçus par fille.

Le second modèle a été étudié par l'auteur avec J. James (1961, voir *Math. Sci. hum.*, n° 24). L'auteur reprend ici ce travail déjà présenté.

BIBLIOGRAPHIE

De nombreux titres, dont :

COLEMAN, J. S., *The adolescent society*, New York, Free Press of Glencoe, 1961.

COLEMAN, J. S. et JAMES, J., "The equilibrium size distribution of freely-forming groups", *Sociometry*, 24, 1, mars 1961.

FELLER, W., "On a general class of contagious distributions", *Ann. math. Statist.*, 14, 1943, pp. 389-400.

FELLER, W., *An introduction to probability theory and its applications*, New York, Wiley, 1957.

FRY, T., *Probability and its engineering*, New York, Van Nostrand, 1928.

LOTKA, A. J., *Elements of physical biology*, Baltimore, Md., Williams et Wilkins, 1925.

LUNDBERG, O., *On random processes and their application to sickness and accident statistics*, Uppsala, Almquist et Wicksells, 1940.

MORENO, J. L. et JENNINGS, H. H., "Sociometric measurements of social configurations", *Sociometry Monographs*, 3, 1945.

RAPOPORT, A. et HORVATH, W. J., "A study of large sociogram", *Behavioural Science*, 6, 1961.

THORNDYKE, F., "Applications of Poisson probability summation", *Bell System Techn. J.*, 5, 1926, pp. 604-624.

GREENWOOD, M. et YULE, G. U., "An inquiry into the nature of frequency distributions of multiple happenings", *J. r. statist. Soc.*, 1920, 83, 255.

(à suivre).