Les nombres réels calculables selon Alan Turing

(un exemple de définition constructive)

H. Lombardi

Résumé

L'article fondateur d'Alan Turing, dans lequel il définit la notion de « fonction mécaniquement calculable », est intéressant à plus d'un titre. D'une part, sa définition a vite remporté l'adhésion unanime des théoriciens ayant réfléchi sur le sujet. Et l'on peut considérer qu'elle est à l'origine de la conception des ordinateurs. D'autre part, dans la note rectificative qu'il publie six mois plus tard, Alan Turing souligne un hiatus qu'il y a entre la notion purement intuitive de « nombré réel calculable » et la première définition qu'il en a donné. Il rectifie donc sa définition initiale avec une argumentation précise qui remet en cause « les standards mathématiques en vigueur » et le raisonnement par tiers exclu. C'est cet aspect de l'affaire que nous présentons et discutons ici, en le situant dans le contexte de l'époque, marqué par la crise des fondements des mathématiques.

La crise des mathématiques

À la fin du 19^{e} siècle, les mathématiques, envahies par la théorie des ensembles, semblent mises en péril par les paradoxes que recèle cette dernière. Devant les critiques implacables de Brouwer, Poincaré [14] et Hermann Weyl [19], une solution de repli assez convaincante est mise au point par Hilbert, qui s'inspire de la tentative par Frege de fonder les mathématiques sur la base de la seule logique.

La stratégie de Hilbert est la suivante. À défaut de pouvoir convaincre que les ensembles cantoriens sont des êtres mathématiques aussi crédibles que les entiers naturels, on va essayer de montrer que ces abstractions si utiles sont inoffensives.

Pour cela il faut réduire, non pas les mathématiques cantoriennes, mais la pratique des mathématiques cantoriennes par les mathématiciens, à un pur calcul élémentaire. On met en place une théorie axiomatique purement formelle qui décrit les principales propriétés désirables des ensembles cantoriens, et l'on démontre par les moyens de l'arithmétique élémentaire, que cette théorie ne conduit à aucune contradiction.

C'est une sorte d'arithmétisation complète des mathématiques.

On sait que Gödel avec son théorème d'incomplétude a ruiné en 1931 les espoirs de Hilbert. À vrai dire, il y a un peu de flou dans le programme de Hilbert, précisément à l'endroit où l'on dit « on démontre par les moyens de l'arithmétique élémentaire, que cette théorie ne conduit à aucune contradiction ». Il faut savoir ce que sont les arguments acceptables en arithmétique élémentaire. Dans la pensée de Hilbert, les arguments acceptables étaient de nature très simples, et certainement suceptibles d'être formalisés dans un système axiomatique simple pour les entiers naturels. C'est sous cette forme précise que Gödel tue le programme de Hilbert : non seulement une théorie axiomatique formelle des ensembles, mais même les systèmes axiomatiques les plus simples pour les entiers naturels, ne peuvent pas être démontrés indemnes de contradiction par les moyens de l'arithmétique élémentaire.

C'est néanmoins la manière de voir de Hilbert qui s'est imposée dans la communauté mathématique au 20^{e} siècle. La théorie des ensembles (une fois réduite à un système axiomatique formel) était d'utilisation « trop agréable » pour que l'on y renonce facilement. Même si le programme de Hilbert était mort, on ferait donc comme s'il n'en était rien !

En fait, là n'est pas la fin de l'histoire. De manière un peu surprenante, ce sont les contradicteurs des mathématiques cantoriennes qui démontrent aujourd'hui une certaine validité du Programme de Hilbert, lorsque l'on remplace l'exigence de « démonstration en arithmétique finitiste à la Hilbert » que l'on trouvait dans ce Programme, par une exigence moins forte de « démonstration constructivement acceptable ». Mais ceci nous écarte un peu trop d'Alan Turing et nous renvoyons sur ce sujet à d'autres publications [1, 3, 4, 5, 6, 8, 9, 10, 12, 13, 19].

La Machine de Turing

La volonté de comprendre les mathématiques par l'étude de théories formelles rendant compte de « ce qui se passe quand on écrit des mathématiques » est une tentative de comprendre les mathématiques en termes d'un calcul purement mécanique. Il était donc naturel que nombre de ceux qui s'occupaient de logique formelle aient senti la nécessité de mettre au clair ce qu'est exactement « un calcul mécanisé », non seulement lorsque l'on pratique une théorie formelle, mais en toute généralité.

Un calcul mécanique qui reproduit les raisonnements de manière purement formelle est une manipulation de « mots » et de « phrases » selon des règles précises et objectives. Mais à vrai dire, il en est ainsi de tout calcul purement mécanique. Et par ailleurs une phrase n'est jamais qu'un « mot » très long comportant des signes de ponctuation.

Une fois qu'un alphabet a été fixé (y compris les signes de ponctuation), les mots et les phrases peuvent être codées par des entiers naturels, en choisissant par exemple un système de numération en base b, où b est supérieur au nombre de lettres dans l'alphabet. Via ce type de codage un calcul peut être vu comme une fonction de $\mathbb{N}^k$ vers $\mathbb{N}$ et « comprendre ce qu'est un calcul purement mécanique » revient à « comprendre ce qu'est une fonction calculable de $\mathbb{N}^k$ vers $\mathbb{N}$ par un procédé mécanique ».

Lorsque dans les années 1930 des mathématiciens et logiciens ont réfléchi à la manière de décrire en termes précis ce qu'est un calcul algorithmique, ils ont abouti à des résultats assez variés quant à la forme, mais identiques quant au fond. Tous les modèles élaborés ont abouti à la même notion de « fonction calculable de $\mathbb{N}^k$ vers $\mathbb{N}$ ». Citons notamment Gödel[1], Church [2], Post [15] et Turing.

Cependant, c'est Alan Turing [17] qui a emporté la conviction par la simplicité de son modèle, et par le caractère vraiment mécanique de ce dernier. Et ce n'est pas un hasard si les ordinateurs sont peu ou prou des Machines de Turing. Pour plus de détails nous renvoyons à l'ouvrage [16] où sont traduits et commentés l'article original de Turing et la rectification qu'il lui apporte six mois plus tard.

De plus, le caractère très élémentaire du fonctionnement abstrait de la Machine de Turing en a fait un candidat naturel, non seulement pour les questions de calculabilité théorique, mais également pour les questions de complexité, et en particulier pour la question de l'appréciation du temps et de l'espace nécessaires à l'exécution d'un algorithme.

`ENCADRÉ À PART` : **Machine de Turing et Programmation**

La Machine de Turing n'a pas à faire peur au néophyte. On peut donner un modèle équivalent à la Machine de Turing en termes de programmes exécutables, sans doute plus parlant pour quiconque a déjà écrit un programme informatique. On considère des programmes de nature très simple. Ils sont écrits en utilisant des variables entières $N_1, \ldots,$

1. On trouve dans Wikipedia l'information suivante, sans référence précise. Jacques Herbrand décrivit, dans une lettre adressée à Kurt Gödel en 1931, un modèle de calcul, des systèmes d'équations avec un mécanisme d'évaluation symbolique par valeur. Ce modèle fut précisé par Gödel lors d'exposés à Princeton en 1934 (cf. [7]). Les fonctions ainsi calculées furent appelées fonctions récursives générales.

N_r (les entiers sont supposés écrits en binaire) ou booléennes $B_1, \ldots, B_s$ ($\in \{0,1\}$). Un « programme élémentaire » n'est rien d'autre qu'une suite finie d'instructions numérotées de l'un des types suivants.

(A) Affectations
  (1) $B_j \leftarrow N_i \bmod 2$
  (2) $N_i \leftarrow N_i$ div 2
  (3) $N_i \leftarrow 2N_i + B_j$
  (4) $B_j \leftarrow 0$
  (5) $B_j \leftarrow 1$
(B) Branchements
  (1) Direct : aller à l'instruction n°...
  (2) Conditionnel booléen : si $B_j = 0$ aller à l'instruction n°...
  (3) Conditionnel entier : si $N_i = 0$ aller à l'instruction n°...
(S) Arrêt.

Les variables sont toutes initialisées à 0 sauf celles qui représentent les entrées du programme. Une des variables, précisée a priori, représente le résultat du calcul. Ce résultat est récupéré à la fin de l'exécution de l'algorithme.

Puisque les entiers sont écrits en binaire, on voit que chaque affectation ou branchement peut correspondre à un travail qui est réalisé en consommant un temps et une énergie indépendantes de l'état des variables. Le temps d'exécution est donc raisonnablement estimé comme étant le nombre d'instructions exécutées avant d'aboutir à l'arrêt.

`FIN ENCADRÉ`

Les nombres réels calculables

Dans son article fondateur, Turing prend prétexte du problème posé par la définition de « ce que c'est qu'un nombre réel calculable » pour proposer une définition beaucoup plus générale, celle de calcul mécanique, ou mécanisable.

Naturellement, il sait très bien qu'il ne peut y avoir de théorème mathématique proprement dit sur cette question[2], mais seulement une proposition raisonnable de définition, qui aboutit non à un théorème, mais à une thèse, la thèse dite aujourd'hui de Church-Turing :
Tout calcul mécanisable est susceptible d'être effectué par une Machine de Turing abstraite, qui disposerait d'autant de temps et d'espace qu'il serait nécessaire pour ce calcul.

Concernant les nombres réels calculables proprement dits, il propose que soit défini comme calculable un réel dont la suite des décimales est calculable (définition que l'on trouve par exemple dans l'Encylopedia Universalis).

Il est remarquable que la note rectificative qu'Alan Turing publie six mois plus tard [18], outre le fait de corriger des imperfections purement techniques de l'article initial, propose une révision de la notion même de nombre réel calculable. En effet, Alan Turing s'aperçoit qu'il se peut qu'un nombre réel x soit calculable au sens intuitif, c'est-à-dire que l'on sache calculer pour chaque entier n une approximation rationnelle de x avec la précision 10^{-n}, sans que pour autant l'on sache calculer la suite de ses décimales. Cela tient à ce que, lorsqu'un nombre réel est très proche d'un nombre décimal, il faut a priori

2. Dans la traduction de [16]. « Nous n'avons pas encore tenté de montrer que les nombres « calculables » incluent tous les nombres que l'on aurait naturellement tendance à considérer comme calculables. Les arguments que nous pouvons donner doivent, par principe, faire appel à l'intuition, et seront pour cette raison plutôt insatisfaisants, mathématiquement parlant. »

le connaître avec une précision infinie pour le situer précisément à droite (au sens large) ou à gauche (au sens strict) de ce nombre décimal. Voici ce qu'en dit Turing.

Des difficultés surviennent du fait de la manière spécifique dont nous avons défini les « nombres réels calculables ». En effet, si les nombres calculables doivent satisfaire à des exigences intuitives, on doit avoir :

> Soient (a_n) et (b_n) deux suites calculables de nombres rationnels. Si l'on a pour tout n, $a_n \leqslant a_{n+1} \leqslant b_{n+1} \leqslant b_n$ et $b_n - a_n \leqslant 2^{-n}$, alors il existe un nombre réel calculable α tel que, pour tout n, $a_n \leqslant \alpha \leqslant b_n$.

Nous pouvons donner une démonstration de cette proposition, valable suivant les standards des mathématiques en vigueur, mais faisant intervenir le principe du tiers exclu. La proposition suivante, par contre, est fausse :

> Avec les mêmes hypothèses, il existe une procédure générale qui permet de fabriquer le programme d'une machine qui calcule la suite des décimales du nombre α à partir des programmes des machines qui calculent les suites de rationnels (a_n) et (b_n).

...... Nous pouvons éviter cette situation désagréable en modifiant la manière dont les nombres réels calculables sont [codés] ......

Une définition intuitivement correcte des nombres réels calculables sera par exemple de dire que le réel α est calculable si, et seulement si, il peut être encadré par deux suites calculables de rationnels (a_n) et (b_n) qui satisfont les hypothèses dans la proposition fausse ci-dessus.

Ainsi, Alan Turing nous avertit que,

- d'une part, si l'on accepte « les standards mathématiques en vigueur », on peut démontrer en utilisant le principe du tiers exclu que la première définition proposée est conforme à l'intuition, mais que,
- d'autre part, cette définition n'est pas acceptable, car elle implique une proposition qui est contraire à l'intuition ; et qu'il faut donc la remplacer par une définition meilleure.

Cet avertissement de Turing concernant la contradiction qui apparaît entre l'usage du tiers exclu lorsque l'on suit les standards mathématiques en vigueur d'une part, et l'intuition d'autre part, n'est pas en général apprécié à sa juste valeur.

La question soulevée est en effet de savoir si c'est l'intuition ou les standards mathématiques en vigueur qui se trompent.

Qui se trompe ?

Je pense que, confirmant l'avertissement de Turing, la pratique montre que c'est l'intuition qui a raison contre les standards mathématiques en vigueur. Je n'en citerai que deux exemples.

Premier exemple, on a le fait bien établi suivant[3].

On sait définir une suite calculable $(x_k)_{k \in \mathbb{N}}$ de réels, au moyen d'approximations rationnelles $a_{n,k}/10^n$ $(a_{n,k} \in \mathbb{Z})$ à 10^{-n} près (c'est-à-dire $|x_k - a_{n,k}| \leqslant 10^{-n}$), la suite double $(n,k) \mapsto a_{n,k}$ étant calculable, et qui a la propriété suivante : la suite $n \mapsto$ signe de x_n n'est pas calculable.

En outre, si l'on accepte « les standards mathématiques en vigueur », pour parler comme Turing, on démontre en utilisant le principe du tiers exclu que dans la suite considérée, les x_k sont tous rationnels. Mais naturellement, puisque la suite des signes

3. Voir par exemple la proposition 11.2.2 page 173 dans [11]

n'est pas calculable, la suite calculable de réels $(x_k)_{k\in\mathbb{N}}$ n'est pas une suite calculable de nombres rationnels ! Le clash qui apparaît ici est dû à une seule raison : l'usage des standards en vigueur qui permet de démontrer que les x_k sont « certainement » rationnels, sans pour autant permettre d'expliciter des fractions de nombres entiers égales à ces x_k.

Deuxième exemple : on trouve, dans la littérature mathématique écrite selon les standards en vigueur, signés par de grands noms, des théorèmes du type suivant :
Si un polynôme a pour coefficients des nombres réels calculables (resp. calculables en temps polynomial), alors ses racines réelles sont des nombres calculables (resp. calculables en temps polynomial).

Cependant si vous prenez le polynôme $X^2 - \epsilon$, avec pour ϵ un réel calculable arbitraire, aucun algorithme ne permet de détecter si ce polynôme admet ou non une racine réelle. Que signifie alors le soi-disant théorème ci-dessus ? C'est la chose suivante : en acceptant le principe du tiers exclu, vous êtes capable de démontrer, au sein de la théorie formelle axiomatisée des ensembles ZF, qu'il existe un algorithme calculant les racines réelles de ce polynôme. Quelles peuvent bien être cette preuve et cet algorithme ? Vous inspectez la démonstration en détail et vous voyez apparaître la chose suivante.

– Si ϵ est < 0 l'algorithme tient en une ligne : Répondre : il n'y a pas de racine réelle.
– Si ϵ est $= 0$ l'algorithme tient en une ligne : Répondre : il y a une et une seule racine, 0.
– Si ϵ est > 0 l'algorithme consiste à calculer une approximation rationnelle r_n de $\sqrt{\epsilon}$ et à donner la réponse suivante. Répondre : il y a deux racines, et pour la précision 10^{-n} requise, elles sont données par r_n et $-r_n$.

Ainsi, on voit où mènent les standards en vigueur, démontrer qu'il existe un algorithme répondant à une question bien formulée, sans pouvoir écrire le moindre algorithme qui réponde à la question. Car en fait, vous n'avez pas **un** mais **trois** algorithmes, et il n'y a pas de procédure de décision pour décider lequel est le bon. Autrement dit, il n'y a pas d'algorithme général qui réalise correctement ce que le théorème prétend réaliser de manière algorithmique.

Le bon théorème est en fait le suivant. Il nous donne toute l'information intéressante contenue dans la démonstration du théorème fautif. Cette information est valable, non pas seulement dans le cadre de la théorie formelle ZF[4], mais dans le monde réel des calculs réels sur machines réelles.
Si un polynôme a pour coefficients des nombres réels calculables (resp. calculables en temps polynomial), et si son discriminant est clairement > 0 ou < 0, alors ses racines réelles sont des nombres calculables (resp. calculables en temps polynomial).

La moindre des choses serait de refuser les théorèmes qui vous disent qu'il existe un algorithme lorsque l'on ne vous donne pas l'algorithme. Mais les standards en vigueur étant ce qu'ils sont, on quitte manifestement le terrain du raisonnable ..., et personne ne croule sous le ridicule lorsqu'il publie un article très savant affirmant qu'« il existe un algorithme, que personne ne pourra jamais écrire, mais qui résout le problème concerné ».

La thèse de Church-Turing

La thèse de Church-Turing est généralement admise comme une vérité empirique, mais elle ne peut pas être démontrée.

Le fait que toute procédure algorithmique découverte jusqu'à maintenant pour résoudre n'importe quelle question mathématique se ramène toujours à une fonction mécaniquement calculable (au sens de Turing) ne signifie pas que ce sera toujours le cas.

4. Théorie formelle que l'on espère sans contradiction, mais sans avoir l'espoir de jamais le démonter.

Et, *plus important encore*, même si c'est pour le moment chaque fois le cas, on ne peut pas décrire un processus purement mécanique qui permette de passer de l'algorithme imaginé par le (ou la) mathématicien(ne) au programme qui réalisera cet algorithme. Ce passage qui consiste à transformer un algorithme intuitif en un programme ne peut certainement pas être réalisé par une machine, mais seulement par un être intelligent, pour qui les mathématiques ont « une signification ». Le rapport entre la sémantique (la signification) des mathématiques et leur syntaxe (leur mise en forme digérable par une machine) est un mystère que ne résolvent ni la thèse de Church, ni les mathématiques axiomatiques, ni la logique formelle. Ce mystère relève de la compréhension de ce que sont un collectif d'êtres intelligents, le langage, la pensée.

Nous essayons maintenant de préciser la discussion précédente dans un cadre plus directement mathématique.

Pour ceci, nous donnons deux explications opposées. La première serait celle d'un mathématicien constructif argumentant pour montrer que la calculabilité mécanique ne peut pas capturer la notion intuitive d'effectivité. La deuxième serait celle d'une mathématicienne classique défendant le thèse contraire.

L'argumentation du point de vue constructif

Nous, notons $\mathcal{E}(\mathbb{N},\mathbb{N})$ la classe des suites d'entiers effectives, c'est-à-dire les fonctions de $\mathbb{N}$ vers $\mathbb{N}$ qui peuvent être effectivement calculées.

Comme esquissé plus haut, nous distinguons a priori cette classe $\mathcal{E}(\mathbb{N},\mathbb{N})$ de la classe des fonctions mécaniquement calculables, que nous notons $\mathsf{Tu}(\mathbb{N},\mathbb{N})$.

La notion d'effectivité est plutôt une notion première (au même titre que la notion d'entier naturel). Intuitivement, cette notion se ramène à celle de construction : le résultat est effectif si l'on sait le construire à partir des données. Mais la notion de construction est aussi sujette à discussions. Certaines opérations sont clairement des constructions, et d'autres n'en sont clairement pas. Et il y a le domaine inconnu des mathématiques à venir, sur lequel tout pari est hasardeux. Ainsi, l'effectivité ne peut pas être définie stricto sensu, et elle n'a aucune raison de devoir être la même que celle d'effectivité « mécanique ».

D'autre part, la définition des suites mécaniquement calculables fait appel à la notion d'effectivité de manière apparemment incontournable, au moins lorsque l'on se place d'un point de vue constructif.

En effet, dans la définition d'une fonction mécaniquement calculable, il y a une utilisation inévitable de la notion d'effectivité intuitive (non définie). Cet utilisation de la notion d'effectivité ne peut pas être contournée. Lorsque l'on dit que pour toute entrée $m \in \mathbb{N}$, la Machine de Turing T donne un résultat, on dit que la machine aboutira « effectivement » à l'état final après un certain nombre d'étapes élémentaires de calcul. Autrement dit, sans le concept de base (non défini) d'effectivité, on ne peut comprendre de manière vraiment rationnelle la notion de fonction mécaniquement calculable.

En résumé, la définition même de la classe $\mathsf{Tu}(\mathbb{N},\mathbb{N})$ présuppose une connaissance intuitive de $\mathcal{E}(\mathbb{N},\mathbb{N})$, qui est une notion première et ne peut pas être définie. En conséquence, lorsqu'en mathématiques classiques on prétend capturer la notion d'effectivité à travers la notion de calculabilité mécanique, on commet un cercle vicieux.

L'argumentation du point de vue des standards implicites actuels

Il est vrai que l'on doit a priori distinguer $\mathcal{E}(\mathbb{N},\mathbb{N})$ et $\mathsf{Tu}(\mathbb{N},\mathbb{N})$, sinon nous dirions « Théorème de Church-Turing » au lieu de « Thèse de Church-Turing ».

Mais il n'est pas vrai que la définition de $\mathsf{Tu}(\mathbb{N},\mathbb{N})$ doive s'appuyer sur $\mathcal{E}(\mathbb{N},\mathbb{N})$.

L'affirmation « pour toute entrée m il existe un nombre d'étapes élémentaires de calcul, n, au bout duquel la machine est arrivée à l'état final », dont le mathématicien constructif semble mettre en doute qu'elle ait une signification objective claire, est certainement vraie, ou fausse, dans l'absolu, car l'ensemble des entiers naturels est un infini en acte au moins de manière idéale, c'est-à-dire au moins dans un monde des idées qui nous dépasse. Chaque fois que l'on démontre un théorème au sujet des entiers naturels, on ne fait que découvrir une vérité qui existait déjà de toute éternité, ou plutôt qui existe hors du temps et de l'espace, dans le monde des idées pures.

Ainsi, même s'il n'y a pas de procédé mécanique qui permette de savoir si une Machine de Turing T donnée calcule bien une fonction de $\mathbb{N}$ dans $\mathbb{N}$, il n'en reste pas moins vrai que la machine T, ou bien calcule une fonction de $\mathbb{N}$ dans $\mathbb{N}$, ou bien échoue à calculer au moins une fois la valeur de la fonction. En conséquence la notion de fonction effectivement calculable (au sens intuitif) est bien capturée par la notion (absolue) de fonction mécaniquement calculable au sens de Turing, au moins tant que de nouveaux procédés de calculs effectifs n'ont pas été mis à jour.

Références

[1] Bishop E. *Foundations of Constructive Analysis.* McGraw Hill, (1967). Réédition. Ishi Press International. New-York (2012). 2

[2] Church A. *A note on the Entscheidungsproblem.* Journal of Symbolic Logic, **1** (1), (1936), p. 40–41. 2

[3] Coquand T. *La contribution de Kolmogorov en logique intuitionniste.* dans : L'héritage de Kolmogorov en mathématiques. Charpentier E., Lesne A., Nikolski N. (eds). Belin, Paris (2004). 2

[4] Coquand T. *Recursive functions and constructive mathematics.* Dans : Calculability and constructivity, à paraître, Springer. 2

[5] Coquand T., Lombardi H. *A logical approach to abstract algebra.* (survey) Math. Struct. in Comput. Science **16** (2006), 885–900. 2

[6] Dowek G. *Les métamorphoses du calcul. Une étonnante histoire de mathématiques*, Le Pommier, (2007). 2

[7] Gödel K. *On undecidable propositions of formal mathematical systems.* Mimeographed lectures, Princeton, N. J. (1934) 2

[8] Largeault J.. *Intuition et intuitionisme.* VRIN, Paris (1993) 2

[9] Lombardi H. *Le programme de Hilbert et les mathématiques constructives.* Revue Repères IREM n°50 (janvier 2003) 85–103. 2

[10] Lombardi H. *Mathématiques Constructives.* Notes pour la Conférence aux journées APMEP de 2005 à La Rochelle. Paru dans le Bulletin de l'APMEP n°483 (2009) 449–466 2

[11] Lombardi H. *Épistémologie mathématique.* Ellipses (2011). 4

[12] Henri Lombardi & Claude Quitté. *Algèbre Commutative. Méthodes constructives.* Calvage&Mounet. 2011. 2

[13] Mines R., Richman F., Ruitenburg W. *A Course in Constructive Algebra.* Universitext. Springer-Verlag, (1988). 2

[14] Poincaré H. *La logique de l'infini.* Revue de Métaphysique et de Morale, **17**, p. 461–482, (1909). Réédité dans « Dernières pensées » Flammarion (1913). On peut le trouver sur le web. 1

[15] Post E. *Finite combinatory processes-formulation 1*. Journal of Symbolic Logic, **1** (3), (1936), p. 103–105. 2

[16] Turing A., Girard J.-Y. *La machine de Turing*. Le Seuil. Points Sciences, Paris, (1995). 2, 3

[17] Turing A. *On Computable Numbers, with an Application to the Entscheidungsproblem*. Proc. London Math. Soc., 2e série, **42**, (1937), p. 230–265.
Sur les nombres réels calculables, avec une application au problème de la décision. 2

[18] Turing A. *On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction*. Proc. London Math. Soc., 2e série, **43**, (1938), p. 544–546. 3

[19] Weyl H. *Le continu et autres écrits*. Traduits et commentés par Jean Largeault. Librairie VRIN (1994). 1, 2