Chapitre 2

Modèles de calcul et continuité

Avant de définir notre modèle, il nous a semblé utile de faire un tour d'horizon des modèles de calcul existant pour le situer.

Nous commençons par nous interroger sur le pourquoi et le comment du calcul. À partir de là, nous proposons une typologie des modèles de calcul de manière à discerner le continu[1] du discret. Nous illustrons ceci à l'aide de différents modèles de calcul.

2.1 Calculer

Nous considérons le calcul comme étant le moyen de parvenir, pas à pas, à un résultat à partir de données ou, de manière plus abstraite, comme le moyen d'extraire méthodiquement des informations d'autres informations.

Pourquoi calcule-t-on ? Pour compter (gestion, commerce), pour évaluer (prédiction, construction), pour apprécier (décision), pour mécaniser le raisonnement et traiter l'information (traduction, communication, et toutes les tâches de notre « société de l'information »). Dans ce tour d'horizon chronologique, se dessinent les différents sujets du calcul : nombres entiers puis décimaux (réels ?), les valeurs qualitatives et l'information au sens large (que nous ne tentons pas de définir).

Comment calcule-t-on ? En utilisant l'arithmétique, des abaques, des « recettes de cuisine », des procédés laborieux mais sûrs (algorithmes), des machines prévues à cet effet (bouliers, règles à calcul, ordinateurs). Au fur et à mesure des progrès technologiques, les demandes en calcul ont augmenté : données plus importantes, besoins de plus de rapidité, de précision, de robustesse... Conjointement, les éléments matériels pour calculer se sont aussi développés (roues crénelées, commutateurs électriques, lampes à vide, transistors). Alors que le calcul se mécanisait, philosophiquement, s'est posée la question de le définir.

Datons la définition formelle du calcul à 1936 avec les fonctions récursives de Gödel [Göd31], le λ-calcul de Church [Chu36b, Chu36a], les machines de Turing [Tur36] et les travaux de Post [Pos21, Pos36] et Kleene [Kle36b, Kle36a][2]. La première approche définit des ensembles de fonctions de manière inductive, algébrique, sans se soucier de

[1] Nous n'utilisons pas le terme « analogique », car nous le considérons trop restrictif en tant qu'antonyme de « digital ».

[2] Nous renvoyons à [Soa99] pour un historique de la notion de calculabilité.

leurs évaluations pratiques. Nous nommons cette approches *algébrique*. La seconde approche définit une classe de fonctions, mais fournit en plus des réductions (leur ordre étant laissé à l'utilisateur ou à un algorithme externe). En revanche, la troisième part de l'« activité humaine » : connaissance finie (états en nombre fini), quantité d'information non bornée (le ruban) qui n'est perçu que partiellement (case) mais est totalement accessible (par le mouvement de la tête). Une machine est un procédé *effectif* de calcul. Elle fait, de plus, référence implicitement au temps ; ce qui n'existe pas dans la première approche. Un *calcul* est une suite de configurations menant des données au résultat. Nous nommons cette approche *dynamique*.

Les modèles algébriques n'ont pas de dynamique : ils définissent des fonctions mais non leurs évaluations ; *e.g.* l'ordre d'évaluation des paramètres ou l'utilisation de la programmation dynamique ne font pas partie des fonctions récursives mais relève d'implantations.

Pour le λ-calcul, nous pouvons dire qu'il y a une dynamique dans le sens relation de *réécriture*. Le modèle prévoit des réécritures possibles et suppose des propriétés de confluence (théorèmes de Church-Rosser). En plus du λ-calcul, on peut classer parmi les modèles par réécriture les algorithmes de Markov, les machines de Kolmogorov et les systèmes de Post. À chaque fois, l'idée est de remplacer une partie de l'objet sur lequel porte le calcul (arbre syntaxique, mot ou graphe) en fonction de la partie remplacée et de règles. En λ-calcul, l'objet est un arbre, la réduction peut modifier un nombre non borné de feuilles. Dans les autre cas, le remplacement est local.

Si l'on considère une machine comme un système dynamique, un calcul est une *orbite*. Il est à noter que, d'une part, les notions de non déterminisme, de probabilisme et de parallélisme correspondent à cette approche et que, d'autre part, les définitions par système dynamique posent le problème de la définition des données et résultats : l'entrée (résultat) peut ne pas être présente dans la configuration initiale (finale) mais être fournie (engendrée) au cours du temps[3], ce qui pose un problème d'intégration entre la temporisation de l'entrée et le temps propre au système.

Parmi les systèmes dynamiques, nous voulons singulariser les systèmes par *propagation* finie du calcul. Ce sont plus que déterministes : le nombre d'opérations est fini et connu à l'avance ; de plus toute affectation est définitive, et l'ordre d'évaluation des opérations possibles ne change pas le résultat. Parmi ces modèles, se trouvent les circuits booléens, les réseaux de neurones non ré-entrants et le *straight line programming*.

2.2 Discret et continu

Ces deux notions sont relativement difficiles à cerner[4]. On considère sans contestation $\mathbb{N}$ et tout ensemble fini comme discrets et, $\mathbb{R}$ comme continu. Voici deux exemples limites. Le premier est l'ensemble $\mathbb{Q}$ muni de la topologie engendrée par l'ordre ; par l'intermédiaire des coupures de Dedekin, on retrouve toute la richesse de la topologie de $\mathbb{R}$. Le second est plus ambiguë : un ordinal dénombrable (ensemble dénombrable avec des points d'accumulation)

[3] C'est déjà le cas avec un système d'exploitation ou temps-réel ; car on ignore les entrées futures.

[4] Nous renvoyons à divers articles de [SS90]. Ce colloque a réuni de mathématiciens, physiciens et philosophes pour une réflexion sur le concept du continu. Dans divers communications, le continu est relié ou opposé au fini ou au discret.

peut être considéré comme continu avant chaque ordinal limite et, discret ailleurs, et donc, être *hybride* (à la fois l'un et l'autre selon l'élément). La distinction ne se fait donc pas sur le cardinal. Elle est topologique[5] ; mais il faudrait que la topologie soit toujours explicite. Notre point de vue est qu'il faut une connaissance intentionnelle (ou dynamique) pour pouvoir se prononcer. Regardons nos différentes classes de modèles.

Pour les modèles algébriques, par réécriture et par propagation, la distinction se fait par les arguments et valeurs possibles ainsi que par les fonctions présentes. Pour les modèles dynamiques, il faut regarder ce qui définit le système : l'échelle des temps, les ensembles de configurations et la règle de mise à jour. Si l'on approche le système par les orbites ou *diagrammes espace-temps* engendrés, il faut alors considérer les média ou *supports* pour le temps et l'espace, les valeurs locales des configurations et le critère de validité / correction des diagrammes (*e.g.* ce critère correspond-il à une formule logique du premier ou deuxième ordre ?).

Par exemple, pour une machine de Turing, le temps (itérations sur $\mathbb{N}$), l'espace (le ruban, $\mathbb{Z}$), les valeurs locales (états et symboles) et la fonction de transition (table finie) sont discrets. Inversement, un modèle basé sur des équations différentielles, sur des fonctions de $\mathbb{R}$ dans $\mathbb{R}$ serait, sans contestation, continu.

De l'observation de différents modèles de calcul dynamiques, on peut en extraire deux classes : ceux à *dynamique locale* et ceux à *dynamique globale*. Nous appelons locale toute dynamique dont la mise à jour de tout site du support ne dépend que des sites voisins (selon une topologie, éventuellement implicite, du support). Elle est globale si un site peut être influencé par un site arbitrairement éloigné. Cette classification n'est pas stricte : d'un côté, elle peut dépendre de la nature des configurations et, de l'autre, il existe des cas intermédiaires.

Notre classification des modèles est subjective et basée sur une compréhension elle aussi subjective de la nature de ces modèles.

Tous les modèles, sauf indication, sont capables de simuler toutes les machines de Turing et sont donc universels pour le calcul. Par ailleurs, certains modèles ayant des caractéristiques continues ont des capacités de calcul super-Turing. Une synthèse de cette classification, où sont inclus les automates cellulaires et les machines à signaux, est donnée dans les figures 2.1 (classification) et 2.2 (aspects continus / discrets).

2.2.1 Modèles algébriques

Nous ne revenons pas sur les fonctions récursives qui sont discrètes.

Fonctions analytiques récursives [Moo96] Cris Moore a défini un pendant analytique de la théorie de la récursion classique. Les fonctions de base sont de $\mathbb{R}^n$ dans $\mathbb{R}$ et les opérateurs sont adaptés (*e.g.* la récursion primitive devient l'intégration). Ce modèle est

[5] Un ensemble est continu en x si x appartient à la fermeture du complémentaire de $\{x\}$, discret sinon. Un ensemble est continu (resp. discret) s'il est continu (resp. discret) en chacun de ses éléments. Un ensemble ni continu ni discret est hybride (faute de meilleur terme, nous prenons celui des systèmes hybrides). Cela correspond à nos exemples et aux topologies discrètes (tous les ensembles sont ouverts) et s'applique aux sous-ensembles grâce aux topologies traces.

excessivement puissant : en plus des polynômes et des fonctions trigonométriques, on trouve aussi la fonction caractéristique de $\mathbb{Q}$. Des amendements sur les fonctions de base et les opérateurs ont été proposés pour remédier à cet excès et des liens ont été établis avec l'analyse récursive [CM01, Hai03]. Les valeurs manipulées sont continues.

Nous pouvons aussi placer ici les Machines de Matiyasevich, dont la seule opération est de dire si une équation diophantienne a ou non une solution ; ces équations pouvant coder les ensembles non récursifs.

2.2.2 Modèles dynamiques par réécriture

Nous ne revenons pas sur le λ-calcul qui est discret. Tous les modèles présentés ici sont discrets.

Algorithmes de Markov [MY78, Section 1.8] C'est un système de réécriture du type grammaires contextuelles. Les règles ont un ordre de priorité et certaines sont marquées comme terminales. Le modèle est déterministe.

Machines de Kolmogorov [US93] Ce modèle se veut le plus général possible pour englober tout ce qui a une dynamique locale. À l'inverse des systèmes de réécriture classiques, au lieu de considérer des structures de listes (*i.e.* les mots), elles travaillent sur des graphes (et non plus seulement des chaînes).

Systèmes de Post [Pos46] C'est un système de réécriture du type grammaires contextuelles. Toutes les règles sont à deux sens (*i.e.* de la forme $m \leftrightarrow m'$). Le système est totalement non déterministe (et le *problème du mot* est indécidable).

2.2.3 Modèles dynamiques par propagation

La puissance de calcul est très compliquée à définir pour les modèles suivants. En effet, chaque instance ne définit qu'une fonction élémentaire de n variables dans p, comme un ensemble d'opérations de base avec un ordre (pas forcément complet) d'évaluation. La question de la puissance se pose si l'on considère une famille de telles fonctions. Mais il faut encore définir des familles ; sans contraintes, tout est calculable. Les contraintes sont souvent définies à partir d'un autre modèle de calcul[6], ce qui ne rend pas la classification simple. Ceci est normal puisque ces modèles ont pour origine le calcul sur des ensembles finis et non sur $\mathbb{N}$ et, pour certains, la conception de circuits imprimés.

Tous ces modèles sont à temps discret, le nombre d'itérations étant déterminé à l'avance. Le support est discret (nombre fini de variables).

[6]Par exemple, les circuits engendrés en temps polynomial en fonction de 1^n caractérisent P.

Circuits booléens [Vol99, vL90, Chap. 14] Un circuit est un graphe orienté fini sans circuit (au sens de la théorie des graphes) dont les sommets internes sont étiquetés par des portes logiques. Les racines sont des entrées binaires à fournir et les feuilles, les sorties.

Nous ne présentons pas les versions probabilistes et quantiques [For03], qui n'apportent rien à notre classification.

Réseaux de neurones non-ré-entrants Ce sont aussi des graphes orientés sans circuit, mais les portes logiques sont remplacées par des fonctions seuils (ou des sigmoïdes, les valeurs étant alors réelles) en fonction de la valeur d'une forme linéaire sur les entrées.

Straight line programming La machine est une suite d'instructions de la forme numéro de ligne := opération({numéro de ligne}*). Les numéros opérandes devant être inférieurs au numéro courant.

Selon les cas, les valeurs et les opérations peuvent être discrètes ou continues.

2.2.4 Modèles à dynamique locale

Nous ne revenons pas sur les machine de Turing, foncièrement discrètes. Les modèles de cette classe sont majoritairement discrets.

Analyse récursive [Wei00] C'est le domaine des machines de Turing de type 2 : l'entrée et la sortie sont sur deux rubans distincts. L'entrée est infinie et on ne peut ni y écrire ni revenir en arrière. La sortie est produite caractère à caractère, on ne peut ni y lire ni y récrire ; elle est elle aussi de taille infinie. Les mots infinis inscrits sur ces rubans représentent des réels. Pour obtenir un résultat exact, il faut un nombre infini d'itérations de la machine ; pour une « approximation » un temps fini suffit.

Le temps est discret ; les configurations sont à support discret ($\mathbb{N}$) à valeurs discrètes (alphabet fini), manipulées discrètement (*i.e.* lettre à lettre). Malgré cela, nous serions tenté de dire qu'il est continu car les valeurs que l'on veut manipuler sont réelles et les fonctions correspondantes de $\mathbb{R}$ dans $\mathbb{R}$ sont continues pour la topologie usuelle de $\mathbb{R}$.

Automates temporisés [AD94] Dans le domaine de la vérification de systèmes temps-réel, on ne peut se contenter d'une simple relation de successeur ou de dates entières. Pour cela des automates finis temporisés manipulant des mots temporisés (*time-event*, une date est ajoutée à chaque entrée de l'automate) ou des *signaux* (fonctions constantes par morceaux) ont été introduits et intensivement étudiés [Rab98, Rab03, Tra01]. L'échelle des temps est continue, mais les changements d'états sont discrets. Les cas d'accumulation de ces dates (paradoxe de Zénon) sont généralement exclus, bien qu'une approche pour les traiter ait été proposée [BP00].

Ce modèle n'est pas Turing-universel, la mémoire des automates étant finie. Ils correspondent plus à la caractérisation d'un comportement qu'à un calcul.

On peut se demander, ici, si la donnée est continue (mot temporisé $(\Sigma \times \mathbb{R}_+)^*$ et les itérations discrètes ou, si les données sont discrètes mais le temps continu (selon la manière dont on considère la temporisation des mots).

Machine de Turing quantiques [Gru99, Hir01] Ce sont des machine de Turing, mais avec des opérateurs atomiques différents. Les valeurs des amplitudes sont complexes, les opérateurs sont des opérateurs linéaires unitaires. Les valeurs et la mise à jour sont donc continues.

Le modèle n'est pas tout à fait local car des cases arbitrairement éloignées et en nombre quelconque peuvent être corrélées[7]. Il se crée ainsi des liens de voisinage.

2.2.5 Modèles à dynamique globale

Automates à compteurs [Min67] Nous ne présentons pas ici ce modèle, car cela est fait dans la Sect. 4.1. Nous le considérons comme global car il peut, à tout moment, modifier n'importe quel compteur et donc agir à distance[8].

Continuous-space optical model [NW01] Dans ce modèle, les données sont des images carrées à deux couleurs (*i.e.* des fonctions caractéristiques de sous-ensembles de $[0,1]^2$), les opérateurs disponibles sont des redimensionnements et des filtres de décompositions en séries de Fourrier.

Nous l'avons inclus dans notre liste pour montrer la variété des modèles proposés. La dynamique est par itération, le support est discret ($\mathbb{Z}^2$) mais les valeurs sont continues. Pour chaque opération seulement deux parties finies de $\mathbb{Z}^2$ sont concernées ; mais comme la taille des redimensionnements n'est pas bornée et qu'il n'y a pas non plus de distance entre la donnée et le résultat, il faut le considérer comme global.

Équations aux dérivées partielles [Orp94, ŠO01] Ce sont des modèles où les configurations ont des valeurs continues, le temps est lui aussi continu et la dynamique est une équation différentielle. La trajectoire est le calcul ; le résultat la valeur limite. C'est un modèle continu bien que le nombre de variables soit fini.

Modèle de Blum, Shub et Smale [BSS89, BCSS98] Ce modèle peut être vu comme un programme comportant des évaluations de polynômes et des branchements. Il peut calculer une fonction de $\mathbb{A}^n$ dans $\mathbb{A}$ polynomiale par morceaux ($\mathbb{A}$ étant un anneau). Ces morceaux sont délimités par des polynômes et peuvent être en nombre infini. Ce modèle a une capacité de calcul super-Turing dès que l'on accepte des valeurs réelles quelconques car le modèle peut alors s'en servir comme oracles.

Le temps est discret ; les configurations ont un support discret (nombre de variables) mais des valeurs dans un ensemble discret ou continu (selon l'anneau, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ ou $\mathbb{C}$).

Ordinateurs analogiques [Sha41, Sha42, PE74] Avant le règne du tout digital, devant la faible capacité des ordinateurs disponibles et les types de problèmes à résoudre, on a conçu des ordinateurs analogiques, plus électriques qu'électroniques. Une théorie a été développée

[7]Typiquement, la *téléportation quantique* est instantanée quelle que soit la distance entre les éléments de la paire ERP (on peut se poser la question de savoir s'il n'y a pas altération de l'espace).

[8]Nous ne parlons pas du modèle RAM [AHU74, CR73] qui se classent comme les automates à compteurs.

sur ce modèle qui a été utilisé de la seconde guerre mondiale jusqu'à la fin des années soixante. Les seuls problèmes sont la précision et le bruit.

Système hybrides [Bra95] Ce sont des modèles à temps et configurations continus mais en partie discrets : l'évolution se faisant parfois par paliers discrets (*e.g.* dirigée par un automate fini, configurations dans $\mathbb{R}^n \times \mathbb{N}$).

Par exemple, le modèle à dérivée constante par morceaux (ces morceaux étant polygonaux et en nombre fini) a été étudié et des liens avec la hiérarchie analytique ont été établis [AM98, Bou99a, Bou99b]. Le système est à temps continu, mais les seuls instants importants sont ceux où l'on change de morceau. Ceux-ci sont discrets, mais ils peuvent s'accumuler (problème d'Achille et de la tortue) ce qu'utilise le modèle.

(Analog) recurrent neural networks (RNN et ARNN) [SS95] Ce sont des réseaux de neurones où le graphe d'interconnexion peut avoir des cycles. Les conditions de fin du calcul sont variées.

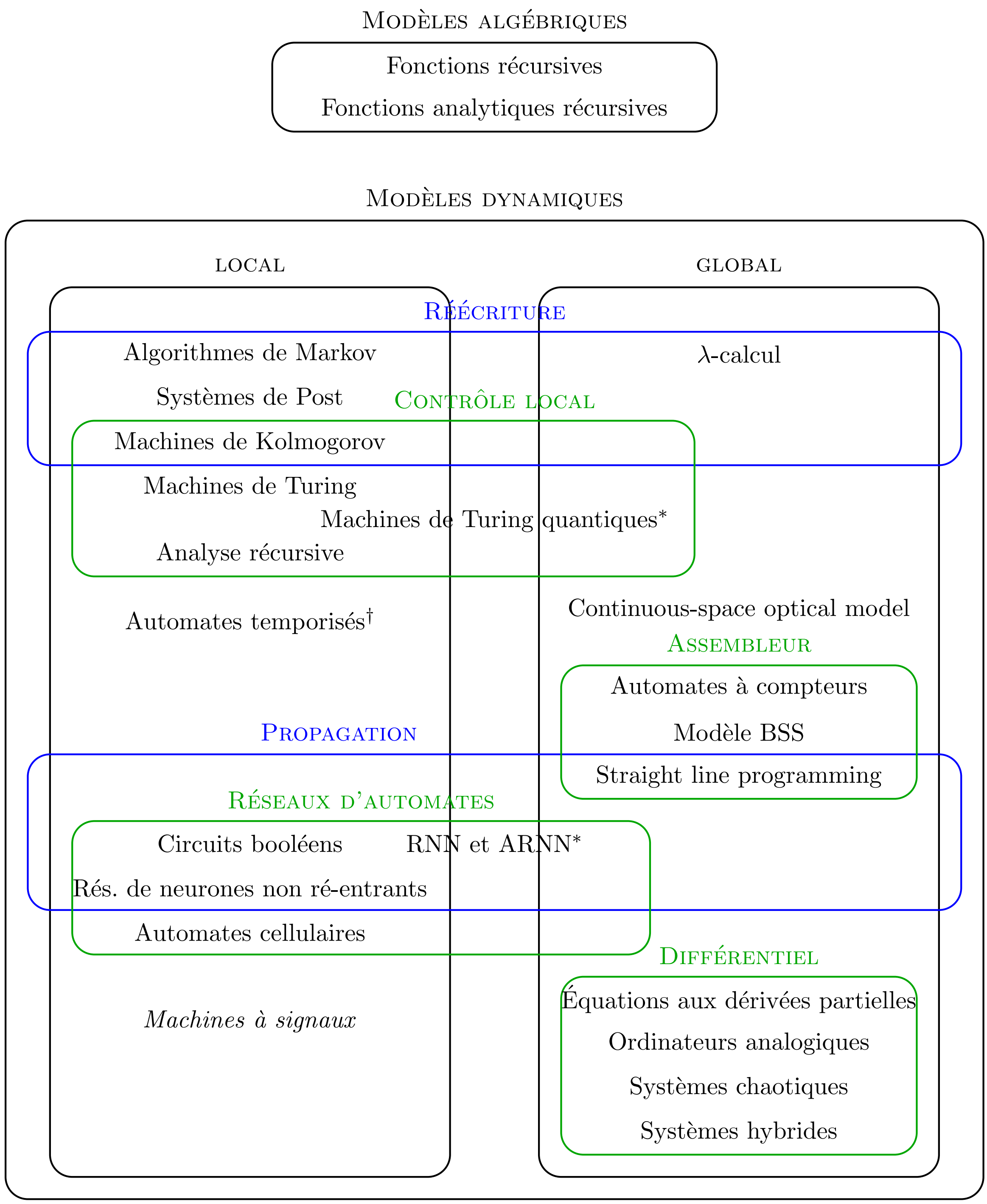
L'aspect local ou global d'un réseau de neurones n'est pas simple : si l'on considère les liaisons synaptiques comme réseau d'interconnexions, tout est local ; mais le modèle permet aussi que la valeur d'un neurone dépende de toutes les autres, ce qui est global.

Citons aussi la bio-informatique (*e.g.* ADN [PRS98] que l'on pourrait considérer comme continu en tant que grand fini), et les systèmes chaotiques [Moo91, Sie96] (foncièrement continus).

Cette classification peut être poursuivie, par exemple on peut définir les classes de modèles suivantes :
- *Assembleur*, *i.e.* suite d'instructions sur des variables : automates à compteurs, modèle BSS et straight line programming ;
- *Contrôle local* : ils déterminent localement, à l'endroit de la précédente modification, l'action à faire : machines de Kolmogorov, machines de Turing (dont quantiques) et analyse récursive ;
- *Différentiels* : ils utilisent des équations différentielles et ont donc besoin de temps continu : équations aux dérivées partielles, ordinateurs analogiques, systèmes chaotiques et hybrides ;
- *Réseaux d'automates* : circuits, tous les réseaux de neurones et les automates cellulaires.

Les machines à signaux sont donc des modèles à dynamique locale. C'est à notre connaissance, le seul modèle à temps et support continus mais à valeurs et mises à jour discrètes. C'est aussi le seul de nature géométrique.

Terminons ce chapitre en mentionnant que le continu pourrait ne pas exister et nous ferions alors fausse route. En effet la thèse d'une réalité totalement discrète (et même finie) existe [Ela01, Zei01].



* classification ambiguë, voir la description du modèle.

† ce n'est pas un modèle de calcul.

Fig. 2.1 – Classification des modèles.

Modèle	**Temps**	**Support**	**Valeurs**	**Mise à jour**
λ-calcul	discret	discret	discret	discret
Algorithmes de Markov	discret	discret	discret	discret
Machines de Kolmogorov	discret	discret	discret	discret
Systèmes de Post	discret	discret	discret	discret
Automates cellulaires	discret	discret	discret	discret
Analyse récursive	discret	discret	discret[1]	discret
Automates temporisés	cont./disc.[2]	continu	discret	discret
Machines de Turing	discret	discret	discret	discret
Machines à signaux	*continu*	*continu*	*discret*	*discret*
Circuits booléens	discret	discret	discret	discret
Réseaux de neurones non ré-entrants	discret	discret	discret	discret
ARNN	discret	discret	continu	continu
RNN	discret	discret	discret	discret
Machines de Turing quantiques	discret	discret	continu	continu
Automates à compteurs	discret	discret	discret	discret
Modèle BSS	discret	discret	cont./disc.[3]	discret
Straight line programming	discret	discret	cont./disc.[3]	discret
Continuous optical model	discret	discret	continu	continu
Équations aux dérivées partielles	continu	continu	continu	continu
Ordinateurs analogiques	continu	discret	continu	continu
Systèmes chaotiques	continu	continu	continu	continu
Systèmes hybrides	cont./disc.[4]	discret	cont./disc.[4]	cont./disc.[4]

1 : même si globalement c'est un réel qui est codé.
2 : évènements discrets mais sur un support continu.
3 : selon les éléments / l'anneau considéré.
4 : selon le modèle, continu ou discret peut disparaître d'une des cases.

FIG. 2.2 – Composantes des modèles dynamiques.