UNIVERSITÉ DU QUÉBEC À MONTRÉAL

EFFETS DE TRANSFERT ASSOCIÉS À L'APPRENTISSAGE DE LA PROGRAMMATION VERS LES DISCIPLINES SCOLAIRES TRADITIONNELLES AU SECONDAIRE

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

AU DOCTORAT EN ÉDUCATION

PAR

HUGO G. LAPIERRE

MAI 2024

REMERCIEMENTS

Cette thèse doctorale est l'aboutissement de plusieurs années de travail. De nombreuses personnes ont été présentes à différentes étapes de ce parcours.

Je tiens d'abord à remercier chaleureusement les membres de mon comité de direction. Merci à mon directeur, Patrick Charland, pour la confiance qu'il m'a accordée au fil des années, du début de la maîtrise jusqu'à aujourd'hui. Patrick, je te suis reconnaissant de ton soutien indéfectible à travers ce long processus. Je remercie également mon codirecteur Martin Riopel pour son écoute attentive et les nombreuses discussions enrichissantes à propos du milieu académique. Martin, merci de m'avoir partagé une vision inspirante de la recherche et de la science.

Mes remerciements vont également à l'Équipe de recherche en éducation scientifique et technologique, et en particulier à son directeur, Patrice Potvin, pour avoir su créer une équipe de recherche stimulante, propice à l'épanouissement de la relève étudiante, grâce à un programme de recherche à la fois exigeant et empreint de bienveillance.

Je désire aussi souligner l'apport de mes collègues du troisième étage, qui ont su insuffler dynamisme et camaraderie lors des innombrables journées consacrées à la rédaction. Jérémie Blanchette-Sarrasin, Amine Mahou, Alexandra Bolduc, Olivier Arvisais, Marie-Hélène Bruyère, Guillaume Malenfant-Robichaud et tous les autres, ces 21 sessions universitaires auraient paru beaucoup plus longues sans vous. Je tiens également à remercier spécialement Yannick Skelling-Desmeules, mon coloc de bureau, avec qui j'ai partagé un quotidien qui s'étirait souvent du matin au soir : du premier café au 5 à 7 tardif. Je n'aurais été coloc avec personne d'autre que toi!

Je remercie aussi mes amis de toujours, Antoine, Benjamin, Benoit, David, Étienne, Jérôme, Julien, Mathieu, Pierre-Luc, Simon et Amélie. Merci de m'avoir encouragé même si je sais que vous vous demandiez parfois ce que je faisais vraiment.

Merci à ma sœur Karine et à mon beau-frère Stéphane pour les nombreux repas cuisinés et pour votre soutien au quotidien. Viva Santa Maria!

Merci à mes beaux-parents Manon et Lucien pour leurs constants encouragements et pour le blitz de gardiennage des derniers mois.

Et bien entendu, merci à mes parents Claude et Guylaine pour leur support éternel. Je suis tellement chanceux de vous avoir et je ne vous remercierai jamais assez.

Finalement, merci à ma Lorie pour absolument tout, car tout est possible avec toi. Je t'aime.

À mon fils Laurence, qui est arrivé dans nos vies plus tôt que prévu et qui est devenu mon plus grand moteur d'inspiration dans le sprint final vers la fin de cette thèse.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Plusieurs organisations internationales, dont l'UNESCO (Claro et Castro-Grau, 2023) et l'OCDE (2023), mettent de l'avant la nécessité de mieux outiller les futurs citoyens afin qu'ils utilisent plus efficacement le numérique, et qu'ils parviennent de surcroît à contribuer à la création et à l'innovation technologique, notamment en matière d'intelligence artificielle. Conformément à ces aspirations, plusieurs systèmes éducatifs à travers le monde ont désormais rendu obligatoire l'apprentissage de la programmation pour tous les élèves du secondaire, et parfois même du primaire. S'inscrivant dans cette tendance internationale, le Gouvernement du Québec a récemment introduit l'apprentissage de la programmation au sein de la compétence numérique qui doit être développée de manière transversale chez tous les élèves du préscolaire, du primaire et du secondaire (Gouvernement du Québec, 2019).

Plusieurs raisons sont évoquées dans la littérature scientifique pour justifier la pertinence d'intégrer la programmation au cursus scolaire des apprenants (Blikstein et Moghadam, 2019). La plus souvent mentionnée concerne le rôle que jouerait l'apprentissage de la programmation dans le développement d'une *pensée informatique* (Lye et Koh, 2014). La pensée informatique réfère aux compétences en raisonnement de haut niveau, telles que la résolution de problèmes et l'abstraction, susceptibles d'être transférées à d'autres apprentissages scolaires (Shute *et al.*, 2017). Toutefois, de nombreux chercheurs (p. ex., Manches et Plowman, 2017) appellent à faire preuve de prudence, car les effets possibles de l'apprentissage de la programmation, par le biais du développement de la pensée informatique, n'ont pas encore été clairement démontrés de façon empirique.

C'est donc précisément pour répondre à ce vide sur le plan scientifique, mais également pour mieux éclairer les choix politiques et pédagogiques au regard de l'enseignement de la programmation, que **ce projet de recherche doctoral vise à examiner les effets de transfert potentiels associés à l'apprentissage de la programmation sur trois disciplines scolaires qui sont centrales au programme de formation du secondaire : le français, les mathématiques et les sciences**.

Pour s'intéresser à cette question, un devis longitudinal et comparatif a été employé. Une base de données massive a été constituée à partir de l'ensemble des résultats scolaires de fin d'année (français, mathématiques et sciences) de 440 élèves, pour l'entièreté de leur parcours au secondaire. Les données proviennent d'une même école secondaire ayant implanté une concentration scolaire portant spécifiquement sur l'apprentissage de la programmation depuis 2014. Deux autres concentrations (Arts et Sports) agissent à titre de groupe de contrôle actif afin de mieux isoler la valeur ajoutée de l'apprentissage de la programmation. Afin de comparer les résultats de fin d'année (années 1 à 5) des élèves pour chaque concentration (Programmation, Arts et Sports), une modélisation par trajectoires latentes multigroupe a été réalisée pour chaque discipline (français, mathématiques et sciences) à l'aide de Mplus (Muthén et Muthén, 2017), un logiciel de modélisation statistique permettant de réaliser des comparaisons longitudinales d'une variable à travers le temps.

Les résultats obtenus suggèrent que les élèves inscrits à la concentration Programmation présentent en moyenne de meilleurs résultats scolaires en mathématiques au fil du temps, comparativement à ceux des deux autres concentrations, suggérant ainsi un effet de transfert positif associé à l'apprentissage de la programmation pour cette discipline. Cet effet de transfert semble particulièrement notable pour la compétence mathématique liée à la résolution de problèmes. En revanche, l'appartenance à la

concentration Programmation n'est pas associée à de meilleurs résultats scolaires en sciences et en français, ce qui suggère une absence de transfert positif pour ces disciplines.

La principale interprétation de ces résultats est que les mathématiques présenteraient des similarités plus importantes avec la programmation, ce qui contribuerait à faire en sorte que ces deux domaines soient plus rapprochés, facilitant ainsi le transfert entre ceux-ci. Par ailleurs, des similarités moins marquées et possiblement moins directes entre les sciences et la programmation contribueraient à éloigner davantage ces domaines, ce qui pourrait expliquer en partie l'absence de transfert positif observé dans les résultats. Qui plus est, il est également possible que la plus grande hétérogénéité associée à la discipline des sciences, qui englobe une gamme de disciplines scientifiques, complique la détection des effets de transfert. En effet, les compétences et les connaissances développées par la programmation pourraient être plus directement liées à certaines disciplines scientifiques précises, comme la physique, et moins à d'autres, comme la biologie, ce qui pourrait contribuer à atténuer les résultats. Pour ce qui est du français, bien qu'il puisse exister quelques similarités avec la programmation, celles-ci apparaissent dans l'ensemble moins marquées que pour la discipline des mathématiques, et même celle des sciences. Il est donc probable que l'éloignement entre la programmation et le français soit trop important pour permettre un transfert positif significatif entre ces domaines.

Sur le plan scientifique, cette recherche représente un pas de plus vers une meilleure compréhension des effets de transfert associés à l'apprentissage de la programmation pour d'autres apprentissages scolaires. À notre connaissance, elle est la toute première à étudier de manière empirique, comparative et longitudinale les effets de transfert associés à l'apprentissage de la programmation, en documentant les performances scolaires d'élèves sur l'ensemble de leur parcours au secondaire. Cette recherche a ainsi permis d'obtenir des informations inédites et nécessaires concernant la possibilité de transfert associée à l'apprentissage de la programmation vers des apprentissages fondamentaux faisant partie du programme d'enseignement secondaire obligatoire. Les résultats obtenus amènent à nuancer l'idée très répandue selon laquelle l'apprentissage de la programmation faciliterait d'emblée un apprentissage transversal vers d'autres disciplines, à travers le développement de la pensée informatique. Ces premiers résultats permettent donc de mieux circonscrire et anticiper les retombées de l'intégration d'une telle concentration en programmation au sein des programmes d'études et apportent un éclairage supplémentaire quant à la place de la programmation en milieu scolaire. D'un point de vue pratique, cette compréhension plus fine des retombées associées à l'apprentissage de la programmation peut également fournir des pistes utiles aux enseignants et futurs enseignants afin d'orienter et appuyer leurs choix pédagogiques au regard de l'enseignement de la programmation. Cela pourrait notamment mener à préciser les pratiques d'interdisciplinarité entre les mathématiques, les sciences, le français et la programmation, pour favoriser les apprentissages scolaires et ultimement la réussite éducative des élèves.

Finalement, les résultats de ce projet ouvrent la voie à plusieurs pistes de recherches futures. Il serait d'abord pertinent d'identifier de manière plus précise les contenus disciplinaires qui pourraient bénéficier le plus de l'intégration de la programmation, tout en évaluant les effets de transfert pour ces contenus. Les résultats ouvrent également la porte à la réalisation d'études visant à mettre à l'essai des interventions éducatives qui établissent des liens multidisciplinaires explicites entre la programmation et les autres disciplines, ce qui permettrait d'établir des liens plus tangibles avec la pratique pédagogique.

Mots-clés : apprentissage de la programmation, apprentissages scolaires au secondaire, transfert des apprentissages, pensée informatique.

ABSTRACT

Many international organizations, including UNESCO (Claro and Castro-Grau, 2023), and the OECD (2023), emphasize the need to better equip future citizens to use digital technologies more effectively and contribute to technological creation and innovation, notably in the area of artificial intelligence (AI). In line with these aspirations, several educational systems worldwide have now made programming learning mandatory for all secondary school students, and sometimes even for primary school students. Aligning with this international trend, the Government of Quebec has recently made programming part of the digital competency, which is to be developed cross-sectionally across all students in preschool, primary, and secondary education (Government of Quebec, 2019).

Several reasons are discussed in the scientific literature to justify the relevance of integrating programming into learners' educational program (Blikstein & Moghadam, 2019). The most frequently cited reason concerns the role that programming learning plays in fostering *computational thinking* (Lye & Koh, 2014). Computational thinking encompasses high-level reasoning skills, such as problem-solving and abstraction, which can be transferred to other academic disciplines (Shute *et al.*, 2017). However, many researchers (e.g., Manches & Plowman, 2017) urge caution, as the potential effects of programming learning, through the development of computational thinking, have not yet been empirically demonstrated.

Therefore, precisely to address this scientific gap and to provide greater insight into policy and pedagogical decisions regarding programming education, **this doctoral research project aims to investigate the potential transfer effects associated with programming learning on three school disciplines central to the secondary school education program: French, mathematics, and sciences.**

To address this question, a longitudinal and comparative design was employed. A massive database was created from the end-of-year academic results (French, mathematics, and sciences) of 440 students throughout their entire secondary education. The data comes from a single secondary school that implemented an academic concentration specifically focused on programming learning since 2014. Two other concentrations (Arts and Sports) act as active control groups to better isolate the added value of programming learning. To compare the end-of-year results (years 1 to 5) of students for each concentration (Programming, Arts, and Sports), a multi-group latent trajectory modeling was conducted for each discipline (French, mathematics, and sciences) using Mplus (Muthén & Muthén, 2017), a statistical modeling software allowing longitudinal comparisons of a variable over time.

The results suggest that students in the Programming concentration show, on average, better academic results in mathematics over time, compared to those in the other two concentrations, suggesting a positive transfer effect associated with learning programming for this discipline. This transfer effect seems particularly notable for the mathematical skill of problem-solving. On the other hand, the Programming concentration is not associated with better academic results in science and French, which suggests a lack of positive transfer for these disciplines.

The main interpretation of these results is that mathematics would share greater similarities with programming, which would contribute to bringing these two fields closer together, thus facilitating transfer between them. On the other hand, less pronounced and possibly less direct similarities between science and programming would contribute to a greater distance between these two fields, which could

partly explain the lack of positive transfer observed in the results. It is also plausible that the greater heterogeneity associated with the science discipline, encompassing various scientific fields, complicates the detection of transfer effects. Indeed, the skills and knowledge developed through programming may be more directly linked to specific scientific disciplines, such as physics, and less to others, like biology, possibly contributing to the attenuation of the results. In the case of the French discipline, although there may be some similarities with programming, these seem overall less pronounced than for the discipline of mathematics, and even science. Consequently, it is likely that the distance between programming and French is too substantial to allow any significant positive transfer between these fields.

The primary interpretation of these results is that mathematics shares greater similarities with programming, contributing to the convergence of these two fields and facilitating knowledge transfer between them. On the other hand, less pronounced and possibly indirect parallels between science and programming contribute to a greater distance between these two domains, potentially explaining the observed lack of positive transfer in the results. It is also plausible that the greater heterogeneity associated with the science discipline, encompassing various scientific fields, complicates the detection of transfer effects. Indeed, skills and knowledge developed through programming may be more directly linked to specific scientific disciplines, such as physics, and less so to others, like biology, possibly contributing to the attenuation of the results. In the case of the French discipline, although there may be some similarities with programming, these seem overall less pronounced than in the field of mathematics, and even in science. Consequently, it is likely that the distance between programming and French is too substantial to facilitate any significant positive transfer between these fields.

Scientifically, this study represents a significant advancement in understanding the transfer effects associated with programming education on other academic domains. To our knowledge, it stands as the first empirical, comparative, and longitudinal exploration of such transfer effects, documenting students' academic trajectories throughout their secondary school years. This research has thus provided novel insights into the potential transferability of programming skills to fundamental components of the mandatory secondary school education program. The findings suggest a need to refine the widespread idea that learning programming would automatically facilitate cross-disciplinary learning in other disciplines through the development of computational thinking. Consequently, these initial findings offer valuable insights for planning and assessing the integration of programming concentrations within educational programs, shedding further light on the role of programming within educational settings. Additionally, from a practical point of view, this more detailed understanding of the spin-offs associated with learning to program can also provide useful leads for teachers and future teachers to guide and support their pedagogical choices with regard to teaching programming.This may entail refining interdisciplinary practices between mathematics, sciences, French, and programming to enhance students' academic learning and, ultimately, educational success.

Finally, the results of this project suggest several avenues for future research. It would be relevant to identify more precisely the disciplinary contents that could benefit most from the integration of programming, while evaluating the transfer effects for these contents. In addition, these results also open the door to conducting studies aiming to test educational interventions that establish explicit multidisciplinary links between programming and other disciplines, thereby establishing more tangible links with pedagogical practice.

Keywords : programming learning, secondary school education, transfer of learning, computational thinking.

INTRODUCTION

> « *Il y a un monde entre ce que les ordinateurs pourraient faire et ce que la société choisira de leur faire faire* » – Seymour Papert (1981)

Au cours des années 60 et 70, les ordinateurs étaient à un stade de développement rudimentaire. À cette époque, ces machines informatiques étaient fondamentalement des outils programmables dotés d'interfaces extrêmement limitées. Leurs capacités étaient bien loin de celles d'aujourd'hui, et leur utilisation était réservée à des experts en informatique. L'avènement de l'ordinateur fut accompagné de nombreux défis : les ordinateurs étaient volumineux, coûteux à produire et à maintenir, et leur puissance de calcul était très limitée par rapport aux normes actuelles. De plus, les interfaces utilisateur étaient très élémentaires, souvent basées sur des lignes de commande complexes et hermétiques, ce qui signifiait que pour accomplir une tâche, il fallait la programmer soi-même en utilisant des langages de programmation primitifs.

Cependant, les ordinateurs ont connu une évolution substantielle et rapide. Les années 80 et 90 ont ainsi été marquées par l'apparition d'ordinateurs personnels plus abordables et conviviaux, avec des interfaces graphiques permettant aux utilisateurs d'interagir de manière plus intuitive avec ces machines. Cela a ouvert la voie à une démocratisation de l'informatique : les ordinateurs sont devenus des outils plus accessibles et polyvalents destinés à un public plus large. C'est ainsi qu'ils ont graduellement fait leur entrée dans les salles de classe.

À mesure que les ordinateurs devenaient plus accessibles et qu'ils trouvaient leur place dans les milieux scolaires, la volonté d'enseigner aux élèves à programmer s'est développée en parallèle. Dans le domaine universitaire, cette nouvelle relation entre l'ordinateur et les apprenants a donné naissance au champ de recherche de la didactique de la programmation. Plusieurs études se sont d'abord penchées sur la manière dont les adultes souhaitant acquérir des compétences en programmation pouvaient le faire efficacement. Des questions fondamentales ont émergé, telles que les méthodes d'enseignement les plus efficaces, les concepts et compétences à développer chez les apprenants novices, et les caractéristiques des programmeurs experts. Ces questions ont trouvé des réponses dans la littérature scientifique de la didactique de la programmation chez les adultes. Toutefois, il a vite été constaté que l'apprentissage de la

programmation représentait un défi beaucoup plus important pour les jeunes apprenants, principalement les élèves du primaire et du secondaire.

Pour cette raison, des chercheurs comme Seymour Papert, un pionnier de l'éducation informatique, ont développé des environnements de programmation spécialement adaptés aux plus jeunes apprenants, afin de leur permettre de développer plus facilement leurs compétences en programmation. Ces nouveaux environnements ont ouvert de nouvelles perspectives sur l'apprentissage de la programmation et son potentiel pour le développement cognitif des élèves. Papert était en effet convaincu que la programmation pouvait aider les élèves à développer une pensée logique et procédurale, similaire au fonctionnement d'un ordinateur, un concept qu'il a désigné sous le terme de pensée informatique (*computational thinking*). Selon lui, et d'autres chercheurs par la suite, ce mode de pensée avait le potentiel de faciliter l'apprentissage des élèves dans de nombreuses disciplines. Ainsi, l'apprentissage de la programmation a été promu comme un outil permettant de favoriser l'apprentissage et la réussite des élèves dans d'autres disciplines scolaires, suscitant de ce fait un intérêt considérable.

Toutefois, au virage des années 2000, la programmation a pris une place moins importante dans le cursus des élèves du primaire et du secondaire. Les interfaces se sont raffinées et les applications numériques se sont multipliées; l'accent fut alors mis sur la maîtrise des outils technologiques plutôt que sur l'exploration et la compréhension de leur fonctionnement interne via l'apprentissage de la programmation. Vers 2010, de nombreux rapports (p. ex., « Shut down or restart », Royal Society, 2012; « Running on Empty », Association for Computing Machinery & Computer Science Teachers Association, 2010) mettent néanmoins de l'avant le fait que les jeunes élèves sont mauvais quant à l'utilisation des technologies, que les systèmes scolaires les préparent mal aux défis de l'avenir et que les milieux professionnels manquent énormément de travailleurs adéquatement qualifiés sur le plan technologique. En réponse à cette préoccupation croissante, certains gouvernements ont donc décidé d'introduire ou de réintroduire l'apprentissage de la programmation à leur cursus scolaire formel et ce mouvement se fait désormais sentir partout à travers le monde.

Au-delà des avantages pour le marché du travail, l'une des principales raisons évoquées pour justifier cette intégration renvoie directement à l'idée que les compétences développées par l'apprentissage de la programmation seraient susceptibles d'être transférées à d'autres disciplines scolaires, contribuant ainsi

à leur réussite. Or, ces bienfaits sur le plan du transfert des apprentissages demeurent à ce jour largement sous-documentés.

S'inscrivant dans le domaine de la didactique de la programmation, la présente recherche a donc pour objectif de vérifier s'il existe des effets de transfert associés à l'apprentissage de la programmation sur les disciplines scolaires traditionnelles.

Le premier chapitre de cette thèse introduit le contexte de la recherche, offrant un portrait de la situation étudiée. Il expose ainsi le problème de recherche, présente la question de recherche et justifie sa pertinence tant sur le plan scientifique que social. Le deuxième chapitre, le cadre théorique, établit les fondements théoriques de la recherche en abordant d'abord les deux concepts centraux, soit l'apprentissage de la programmation et le transfert des apprentissages. Puis, il présente un état des connaissances au regard des premiers résultats de recherche existant concernant les effets de transfert de l'apprentissage de la programmation vers les autres disciplines scolaires. Enfin, le chapitre se termine par la formulation des hypothèses de recherche. Le troisième chapitre expose la méthodologie de la recherche en détaillant le devis général du projet, le choix du terrain de recherche, la constitution de l'échantillon, la nature des données utilisées, les analyses réalisées ainsi que les considérations éthiques d'usage. Le quatrième chapitre présente les résultats obtenus à la suite de l'analyse des données. Finalement, le cinquième et dernier chapitre présente les principales interprétations de ces résultats, répondant ainsi à la question de recherche et revisitant les hypothèses initialement avancées. Il présente également quelques considérations pédagogiques relativement aux effets de l'apprentissage de la programmation sur les autres disciplines et à son intégration au sein des programmes d'études. Il aborde les forces et les limites du projet et identifie des pistes de recherches futures.

CHAPITRE 1
PROBLÉMATIQUE

Savoir programmer représente une compétence de plus en plus en demande sur le marché du travail actuel et qui le sera encore davantage dans le futur (Blikstein et Moghadam, 2019). Pourtant, au Québec notamment, mais aussi ailleurs, peu de jeunes sont encore amenés à développer cette compétence dans le cadre de leur cursus scolaire formel ou même à l'extérieur de l'école, et ce, malgré une tendance internationale croissante en faveur de l'enseignement de la programmation dans les écoles primaires et secondaires (Barma, 2019; Varenhold, 2017). Cet écart entre le besoin d'une main-d'œuvre présentant de bonnes compétences en programmation et l'absence de formation conséquente souligne la nécessité de mieux comprendre et documenter les bénéfices possibles pour les jeunes apprenants de développer des compétences en programmation, afin de mieux orienter les décisions et actions éducatives.

Ce premier chapitre vise à présenter la problématique de recherche. Dans un premier temps, il sera question du mouvement actuel et de l'enthousiasme, observés dans plusieurs pays, en faveur de l'intégration de la programmation au cursus scolaire des jeunes. À cet égard, une attention particulière sera portée aux contextes canadien et québécois. Dans un deuxième temps, puisqu'elles sont susceptibles de complexifier cette intégration, les difficultés associées à l'apprentissage de la programmation chez les jeunes seront discutées ainsi que les ressources importantes déployées pour surmonter ces difficultés. Compte tenu des défis associés à l'apprentissage de la programmation, il apparaît enfin important de discuter des bénéfices potentiels de cet apprentissage sur différents plans : employabilité future, équité et diversité sociale, développement de la pensée des apprenants et facilitation des apprentissages scolaires. Parmi ces bénéfices, l'effet de transfert possible associé à l'apprentissage de la programmation vers les autres disciplines scolaires sera approfondi, puisque cet élément est le plus évoqué pour justifier l'intégration de la programmation au cursus scolaire des jeunes et qu'il ne fait pourtant pas consensus à ce jour. Enfin, le chapitre se conclura par la présentation de la question de recherche sous-tendant ce projet, qui sera suivi d'une justification de sa portée sociale et scientifique.

1.1 Tendance internationale croissante vers l'apprentissage de la programmation chez les jeunes

La montée fulgurante de l'ère numérique, propulsée par la pandémie de COVID-19, a dessiné un paysage où une fracture technologique prononcée sépare désormais la population (Gaffield, 2020). Aux États-Unis, une étude révèle que 75 % des entreprises font face à des défis opérationnels liés aux technologies,

mettant en lumière une carence importante en compétences technologiques (Forrester, 2023). Un tiers des travailleurs américains ne présenterait pas les compétences numériques suffisantes, pourtant considérées de base, requises pour près de 92 % des emplois (National Skills Coalition, 2023). Au Canada, le tableau est également alarmant : un rapport révélait en 2019 qu'encore un quart de la population avait très peu ou pas du tout utilisé Internet, pourtant introduit depuis près de 30 années (Wavrock, 2019). Ces données assez récentes mettent en lumière qu'une portion considérable de la population canadienne demeure en marge de l'ère numérique, un décalage qui selon plusieurs (Wing, 2006; Wilson, 2010; Furber, 2012) appelle à une intervention urgente afin de favoriser le développement de meilleures compétences technologiques.

L'International Society for Technology in Education (2015) estime à cet égard qu'il est désormais crucial de savoir comment bien utiliser les technologies de l'information et de la communication (TIC) pour évoluer et fonctionner adéquatement dans la société du 21e siècle. Pour y parvenir, certains auteurs soutiennent depuis plusieurs années que l'apprentissage de la programmation représenterait une porte d'entrée privilégiée, car elle sous-tend toutes les réalisations numériques, tous les logiciels et tous les systèmes informatiques, offrant ainsi une base solide pour appréhender et comprendre les fondements technologiques (diSessa, 2000). Certains avancent même qu'il s'agirait d'un apprentissage aussi indispensable qu'apprendre à lire, écrire et compter (Wing, 2006), nécessaire pour contribuer aux défis majeurs de notre époque, notamment ceux relatifs à l'intelligence artificielle, qui occupent une place centrale dans les débats sociétaux actuels. Pour ces raisons, plusieurs soutiennent qu'il devient impératif de consentir des ressources importantes afin de former une nouvelle génération de citoyens en contrôle de la technologie, notamment via l'apprentissage de la programmation, afin qu'elle soit en mesure de s'attaquer à des problématiques d'avenir, indépendamment des actions et des décisions des générations précédentes (Rushkoff, 2010; Wing, 2006).

Ces aspirations vont donc au-delà de la simple maîtrise des outils numériques : il s'agit plutôt de permettre aux individus de comprendre les concepts sous-jacents à la culture numérique (ce que plusieurs appellent la « littératie numérique », Glister et Glister, 1997), notamment via l'apprentissage des concepts de base en informatique. Dans le même ordre d'idées, le Référentiel de compétences en technologies de l'information et de la communication (UNESCO, 2011) met de l'avant qu'un citoyen contemporain devrait non seulement être capable de trouver des informations via le numérique, mais également être en mesure de devenir un créateur numérique pouvant lui-même générer de nouvelles connaissances.

Conformément à ces aspirations, l'introduction de la programmation à l'école a connu une croissance notable à l'international dans les 10 dernières années et est parfois même intégrée très tôt dans le parcours scolaire des jeunes élèves (Grover et Pea, 2013). De nombreux pays ont ainsi fait le choix d'intégrer formellement la programmation ainsi que certains apprentissages fondamentaux en informatique (p. ex., le fonctionnement d'un ordinateur et les pièces qui le composent, la micro-informatique, la binarité, etc.) au sein de leur cursus scolaire primaire et/ou secondaire, et parfois même de manière obligatoire (Gal-Ezer et Stephenson, 2014; Syso, 2014; Knobelsdorf, 2015; Manches et Plowman, 2017).

Un rapport de l'Institut Brookings (2021), une organisation de recherche indépendante, a révélé que parmi 219 pays examinés, environ 20 % (44 pays) avaient rendu la programmation obligatoire ou facultative dans leur programme d'études; près de 7 % (15 pays) avaient introduit une forme d'éducation à l'informatique (« computer science ») dans certaines écoles et dans certaines juridictions régionales comme les états ou les provinces, mais qu'encore 73 % (160 pays) en étaient toujours aux premiers stades d'études curriculaires ou d'expérimentation quant à son intégration à leur cursus. La Figure 1.1 présente une carte du monde sur laquelle sont identifiés les choix réalisés par différents pays.

Bien que ces données témoignent d'un mouvement international en faveur de l'intégration de la programmation, celui-ci demeure relativement récent comparativement à d'autres disciplines scientifiques, technologiques, d'ingénierie et mathématiques (STIM), ce qui fait en sorte que les manières de réaliser cette intégration varient largement d'un pays à l'autre, menant ainsi à une variété de programmes éducatifs. Ceux-ci s'inscrivent au sein de différentes idéologies, approches et finalités relativement à son enseignement (Vahrenhold, 2017). Ces différences se reflètent notamment dans le choix des termes employés pour qualifier ces programmes : à titre d'exemple, certains pays comme le Royaume-Uni utilisent davantage le terme plus englobant des « sciences informatiques » (*computer sciences*), alors que d'autres, comme en Italie, parlent plutôt de la « computation » (*computing*) comme outil à l'apprentissage d'autres disciplines (Guzdial et DuBoulay, 2019). Malgré ces divergences quant aux termes employés, l'ensemble de ces programmes attribuent une place centrale à l'apprentissage de la programmation (Vahrenhold, 2017). Des systèmes éducatifs, tels que ceux de l'Estonie, de l'Australie, du Royaume-Uni et, plus près de nous, de la Colombie-Britannique, qui avaient déjà dans le passé intégré la programmation à leur cursus, mais l'avaient retirée, l'ont récemment réintégrée de manière obligatoire pour tous les élèves. Dans certains cas, cet apprentissage débute même très tôt, de telle sorte que même

des élèves de cinq ans sont parfois appelés à explorer les bases de la programmation à l'aide d'ordinateurs ou de robots (Manches et Plowman, 2017).

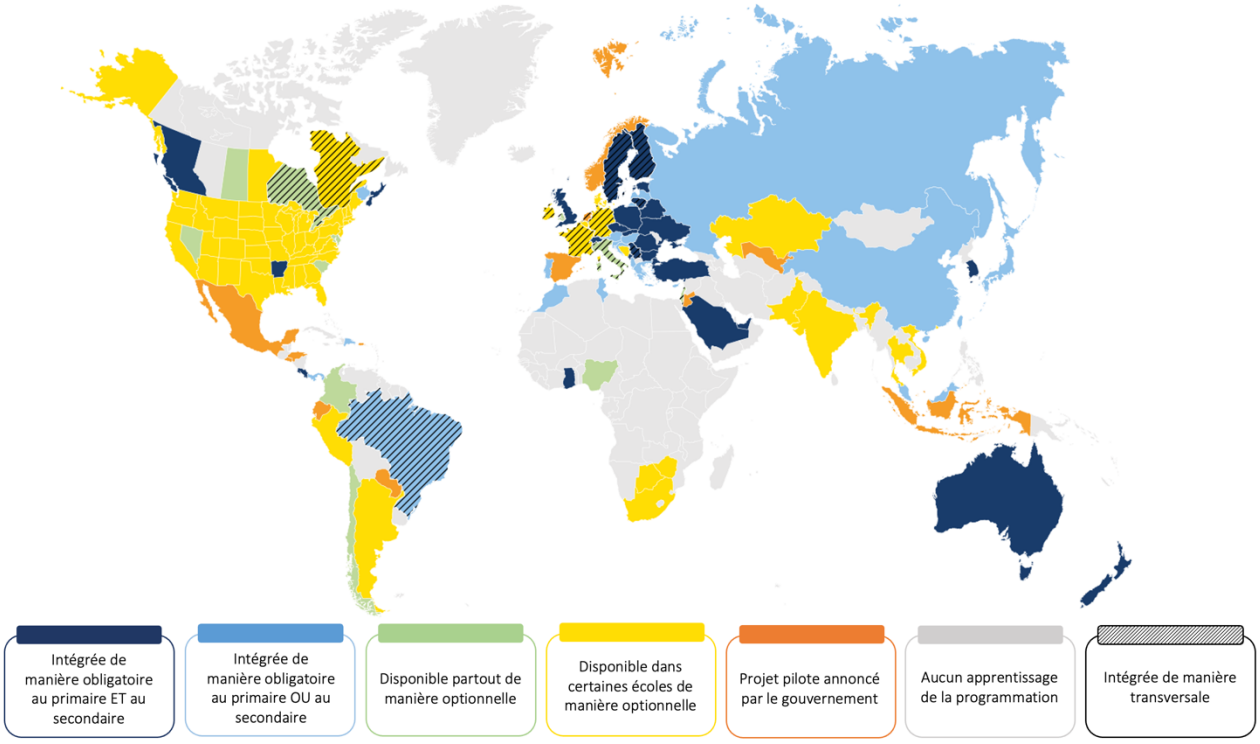

Figure 1.1 Choix de différents pays quant à l'intégration de la programmation au cursus scolaire des apprenants. Reproduite et adaptée de « Building skills for life : How to expand and improve computer science education around the worlds » par Vegas, Hansen, et Fowler, 2021, *The Brookings Institution*, p.16.

Qui plus est, il existe généralement deux grandes approches en matière d'intégration de la programmation au sein du cursus scolaire : 1- comme discipline à part entière ou 2- comme objet d'apprentissage transversal. En effet, certains pays optent pour la création d'une discipline spécifique dédiée à la programmation, comme le Royaume-Uni (Brown, 2012). D'ailleurs, d'autres comme Israël (Gal-Ezer et Stephenson, 2014), la Pologne (Syso, 2014) et l'Allemagne (Knobelsdorf, 2015), qui ont toujours intégré l'informatique et le numérique à leur cursus, octroient désormais à la programmation une place disciplinaire à part entière. D'autres choisissent plutôt de l'intégrer de manière transversale, en l'abordant à travers les autres disciplines scolaires ou encore par le biais d'une compétence transversale qui doit être développée par les élèves, telles que la Finlande et la Suède (Skolverket, 2017).

Enfin, dans les dernières années, de nombreux autres systèmes éducatifs ont mené des études curriculaires dans le but d'évaluer la pertinence d'intégrer la programmation au sein de leur programme

d'études du primaire ou du secondaire; c'est notamment le cas des États-Unis (Ericson, 2016), de la France (Baron, 2014), de l'Inde (Raman, 2015), de la Russie (Khenner et Semakin, 2014) et de la Suède (Rolandsson et Skogh, 2014). Le Québec n'est d'ailleurs pas en reste dans ce mouvement global : depuis 2018, plusieurs actions concertées ont été initiées afin de déterminer la place future que l'apprentissage de la programmation devrait prendre dans les programmes d'études québécois.

La section qui suit offre une présentation plus approfondie des spécificités et des implications de ces initiatives, non seulement dans le contexte québécois, mais aussi à l'échelle plus large du Canada.

1.1.1 Spécificités des contextes canadien et québécois

Au Canada, la forte volonté de mieux préparer les élèves à un monde numérisé et technologiquement avancé a récemment mené au développement du Cadre de référence pancanadien pour l'enseignement de l'informatique (Canada Learning Code, 2020) qui met l'accent de manière ciblée sur l'importance pour les élèves canadiens d'apprendre à programmer. Selon les recommandations de ce cadre de référence, les élèves canadiens devraient ainsi être en mesure de programmer un algorithme en utilisant divers concepts de base, y compris l'algèbre booléenne, les boucles et les tableaux de données, et ce, dès la fin de leurs études secondaires. Il existe toutefois une disparité importante entre les provinces et territoires canadiens dans la mise en application de ces recommandations : actuellement seulement sept des treize provinces et territoires canadiens ont intégré des apprentissages en programmation dans leurs programmes d'études au primaire et au secondaire, sans qu'il s'agisse toutefois d'apprentissages obligatoires (Vegas, 2021). À titre d'exemple, le Nouveau-Brunswick a agi comme précurseur en intégrant l'apprentissage de la programmation à son programme d'enseignement secondaire dès 2014.

En Colombie-Britannique, la programmation est intégrée au programme des élèves du primaire et du secondaire en tant que discipline obligatoire depuis 2016 (Gouvernement de la Colombie-Britannique, 2016). Plus récemment, en Ontario, la programmation a été intégrée au programme d'enseignement des mathématiques pour les élèves de la première année du primaire, dans l'objectif d'améliorer leurs compétences en résolution de problèmes et leur niveau de confort vis-à-vis de la technologie (Gouvernement de l'Ontario, 2020). Encore plus récemment, l'Ontario a aussi fait le choix d'ajouter l'apprentissage de la programmation aux cours de sciences et technologie du secondaire (Gouvernement de l'Ontario, 2022).

Au Québec cependant, l'apprentissage de la programmation ou même de l'informatique ne fait toujours pas partie du cursus obligatoire des élèves (Gouvernement du Québec, 2006). Toutefois, l'apprentissage de la programmation occupe visiblement une place croissante dans les réflexions gouvernementales, et fait partie intégrante du Plan d'action numérique mis sur pied afin d'accélérer la transition numérique de la province (Gouvernement du Québec, 2018). La mesure 2 de ce plan cible en effet « l'utilisation de la programmation informatique à des fins pédagogiques et didactiques pour soutenir les élèves dans la réalisation des apprentissages et le développement des compétences prévues au PFEQ » (Gouvernement du Québec, 2018, p. 27). Dans la lignée de ce plan d'action, un certain nombre d'études se sont penchées sur l'intégration de la programmation à l'école québécoise (Lapierre, 2019; Giroux, Girard et Gagnon, 2018; Allaire-Duquette, 2022; Bugman, 2018). À la lumière d'un rapport de recherche récent (Barma, 2019), des recommandations concrètes ont été formulées pour favoriser la croissance de l'usage pédagogique et didactique de la programmation dans les écoles du Québec. L'une des recommandations centrales est d'intégrer au PFÉQ une compétence transversale portant directement sur l'apprentissage de la programmation afin d'améliorer la mise en relation de la programmation et des autres disciplines scolaires prévues au programme.

Ce plan d'action numérique a aussi donné lieu, en 2019, au développement du Cadre de référence de la compétence numérique (Gouvernement du Québec, 2019), une mesure visant à valoriser l'utilisation du numérique pour l'apprentissage ainsi que le développement de la compétence numérique des jeunes et des adultes. Parmi les douze dimensions composant cette compétence numérique, la deuxième (« Développer et mobiliser ses habiletés technologiques ») cible principalement l'apprentissage de la programmation. Elle met en effet l'accent sur l'importance de la programmation comme moyen pour aider l'élève à développer une « pensée informatique », lui permettant ainsi de s'initier aux concepts clés en intelligence artificielle et pour différents phénomènes émergents liés au numérique (Gouvernement du Québec, 2019).

Plus récemment, des ressources ont également été consenties afin de dégager des recommandations consensuelles sur les arrimages possibles entre l'apprentissage de la programmation et le PFEQ (Barma, 2021). L'une de ces recommandations concerne la création d'une progression des apprentissages au regard de la compétence numérique, incluant des aspects spécifiques à l'apprentissage de la programmation. Concrètement, il est aussi suggéré de mettre en place des projets pilotes dans les écoles

primaires et secondaires du Québec afin d'identifier les concepts prescrits au sein des disciplines scolaires qui présentent des points d'ancrage pouvant faciliter l'introduction de la programmation.

L'ensemble de ces éléments soulignent la place croissante accordée à l'apprentissage de la programmation au sein des agendas gouvernementaux et éducatifs au Québec. Ils témoignent également de la décision du Québec d'intégrer la programmation de manière transversale dans son programme de formation, prévoyant ainsi le développement des compétences en programmation à travers toutes les matières scolaires.

Par ailleurs, outre le cursus scolaire formel, les dernières années ont aussi vu se dessiner de nombreuses initiatives provenant directement d'écoles primaires ou secondaires, ou encore de différents organismes éducatifs, qui visent directement l'exposition des élèves à l'apprentissage de la programmation. Par exemple, le projet « Code Mtl » piloté par conjointement par la Fondation pour les élèves de Montréal et par le Centre de services scolaire de Montréal (CSSDM), permet actuellement à plus de 10 600 élèves de 5 à 12 ans provenant de 117 écoles d'être initiés à la programmation à l'école (Code Mtl, 2023). De nombreuses écoles publiques et privées possèdent également des concentrations « robotique » ou « informatique » où la programmation est au cœur des apprentissages que réalisent les élèves (Barma, 2019). Une situation similaire est aussi observable aux États-Unis où, bien que la programmation ne soit pas un apprentissage obligatoire, plus de 50 % des écoles sont actuellement munis d'un programme scolaire qui expose leurs élèves à l'apprentissage de la programmation, tant au primaire qu'au secondaire (Roberts, Glennon, Weissman, Fletcher, Dunton, Baskin et Mak, 2022).

En somme, bien qu'il n'existe actuellement pas de programme d'études portant spécifiquement sur la programmation, il est indéniable que le Canada et le Québec suivent la tendance mondiale d'intégration de la programmation dans les salles de classe. Néanmoins, malgré cet enthousiasme partagé, de nombreux écrits mettent en évidence que l'apprentissage de la programmation peut s'avérer particulièrement difficile, surtout pour de jeunes apprenants du primaire et du secondaire. La section suivante mettra donc en lumière un aspect crucial et souvent sous-estimé de cette intégration : les difficultés associées à l'apprentissage de la programmation chez les jeunes.

1.1.2 Difficultés associées à l'apprentissage de la programmation, particulièrement chez les jeunes apprenants

La programmation est reconnue comme étant un apprentissage particulièrement difficile (Watson, Li et Godwin, 2014). La littérature en didactique de la programmation souligne en effet qu'apprendre à programmer représente un processus long et complexe, et ce, même pour des apprenants adultes de niveau universitaire (Robins, 2010). Il est généralement admis que la transition d'un programmeur novice à un programmeur expert peut prendre jusqu'à dix ans (Winslow, 1996). La nature même de la programmation est souvent avancée pour expliquer cette difficulté : il s'agit d'un apprentissage qui exige de l'apprenant qu'il comprenne des concepts abstraits (Lahtinen, Ala-Mutka et Järvinen, 2005) et qu'il construise et applique simultanément des compétences interdépendantes (Qian et Lehman, 2017). Cette complexité se traduit d'ailleurs par des taux d'échec particulièrement élevés dans les cours universitaires d'introduction à l'informatique (*Computer Science 101*, CS101), allant jusqu'au tiers des étudiants (Watson, Li et Godwin, 2014) et aurait aussi comme effet de miner l'intérêt des apprenants pour la programmation, en particulier les femmes qui demeurent largement sous-représentées dans le domaine de l'informatique (College Board, 2017). Ces constats préoccupants ont mené de nombreux chercheurs et décideurs politiques à suggérer qu'il serait bénéfique d'exposer plus tôt et de manière plus systématique tous les jeunes apprenants à la programmation (courant du *Computing for All*, Wing 2006; Sahami, 2018).

Toutefois, plusieurs écrits soulignent que l'apprentissage de la programmation pourrait s'avérer encore plus difficile pour de jeunes apprenants en raison des premiers apprentissages qui représentent le défi le plus important (Hermans et Aivaloglou, 2017; Meerbaum-Salant, 2013; Perkins, 2013). De plus, bien que la technologie suscite en général un attrait substantiel chez les jeunes, il semble que les premières expériences de programmation soient souvent considérées comme ennuyeuses et fastidieuses (Rankin, 2008), ce qui peut décourager les apprenants. Il apparaît donc impératif de s'assurer que les premières expériences de programmation se révèlent positives (Lapierre, 2023). La réussite des premiers apprentissages portant sur les concepts de base en programmation serait d'ailleurs déterminante des apprentissages subséquents (Robins, 2010).

Plusieurs raisons sont avancées afin d'expliquer les difficultés plus importantes rencontrées par les jeunes élèves. D'abord, certains auteurs soulignent que les langages de programmation traditionnels ne seraient pas adaptés au niveau scolaire des jeunes, puisqu'ils possèdent une syntaxe logico-mathématique inhabituelle qui est éloignée du langage naturel (Korkmaz, 2016). Il est donc probable que les élèves ne possèdent pas encore les référents nécessaires afin de comprendre le fonctionnement logique de la

programmation. Cette dernière peut alors paraître intimidante, hermétique et peu accessible, décourageant à la fois l'apprenant et l'enseignant. De plus, l'apprentissage de la programmation nécessite aussi d'être capable de faire preuve d'abstraction afin d'entrevoir la finalité d'un programme, ce qui peut représenter un défi supplémentaire pour de plus jeunes apprenants (Robins, Rountree et Rountree, 2003). En effet, le programmeur novice est habituellement limité à la surface d'un programme, puisqu'il l'approche ligne par ligne plutôt que de l'imaginer et de le considérer, dès le départ, comme un ensemble structuré (Lahtinen, 2005). Enfin, les rétroactions offertes par les logiciels de programmation traditionnels sont également limitées et peu ancrées dans la réalité physique du jeune apprenant. Il devient donc difficile pour l'enseignant de contextualiser les premiers apprentissages en programmation à l'aide des langages traditionnels, ce qui engendre souvent chez l'apprenant l'impression que la programmation est une discipline qui n'a pas de retombées tangibles dans son quotidien (Meerbaum-Salant, 2013).

Compte tenu de la difficulté particulière que représente l'apprentissage de la programmation pour les jeunes apprenants, des ressources substantielles ont été investies pour faciliter cet apprentissage et le rendre accessible à un plus grand nombre d'entre eux. La section suivante aborde plus en détail les initiatives et les moyens déployés pour surmonter ces défis et promouvoir la maîtrise de la programmation chez les jeunes.

1.1.3 Importance des ressources investies pour favoriser l'apprentissage de la programmation chez les plus jeunes apprenants

Au fur et à mesure de l'introduction de l'ordinateur au sein de l'école, des ressources considérables ont été investies afin de faciliter l'apprentissage de la programmation chez les jeunes. Cette préoccupation s'est exprimée de différentes façons, notamment par l'élaboration de langages et de logiciels de programmation spécifiquement adaptés aux jeunes apprenants (Kalelioglu et Gülbahar, 2014), le développement d'outils tangibles pour mieux contextualiser les apprentissages en programmation (Horn et Bers, 2019), la mise à l'essai et le déploiement de stratégies pédagogiques visant une meilleure introduction de cette discipline en classe (Waite et Sentance, 2021), ainsi que des investissements dans la formation du personnel enseignant pour qu'il soit adéquatement formé aux concepts clés de la didactique de la programmation (Brown, Sentance et Crick, 2014). De plus, des ressources importantes ont également été déployées en recherche : en effet, un nombre considérable d'études ont été menées pour explorer les effets de divers outils et interventions visant les jeunes dans le contexte de leur apprentissage de la programmation (Fincher, Tenenberg, Dorn, Hundhausen, McCartney et Murphy, 2019).

De nombreuses entreprises ou organismes ont ainsi consacré des efforts importants au développement d'outils spécialement conçus pour répondre aux besoins des jeunes. Par exemple, les dernières années ont été marquées par la prolifération de langages de programmation visuels, spécifiquement élaborés pour rendre l'apprentissage de la programmation plus accessible, incluant, seulement dans les 15 dernières années, plus de 35 langages visuels (Crick, 2017), dont les plus populaires sont Alice, Blockly, Edublocks et Scratch. Chacun de ces langages propose des interfaces et des fonctionnalités distinctes, visant à rendre la programmation accessible et engageante pour des apprenants novices.

Parmi cette multiplicité de logiciels, l'un des plus connus et parmi les plus populaires est Scratch, un logiciel développé par le Massachusetts Institute of Technology (MIT) dans le cadre d'un projet de recherche visant à accroître les compétences technologiques des enfants dès la maternelle (Resnick, 2009). Scratch utilise son propre langage de programmation graphique où l'apprenant clique et glisse des icônes visuelles représentant des opérations et fonctions informatiques, telles que des boucles, des expressions booléennes, des conditions et des variables, afin de générer du code informatique visuel. Cet environnement vise donc à éliminer les erreurs de syntaxe et à augmenter l'accessibilité de la programmation aux jeunes apprenants. De plus, le logiciel Scratch est muni d'une fenêtre de simulation qui permet une rétroaction visuelle immédiate sur le code généré. Une étude de Meerbaum-Salant (2013) a démontré que ce langage visuel semblait en effet faciliter l'apprentissage de concepts de base en programmation, tant pour des élèves du primaire (Flannery, 2013; Franklin, 2013; Bell, 2015; Papadakis, Kalogiannakis et Zanaris, 20016) que du secondaire (Grover, Cooper et Pea, 2014; Grover et Basu, 2017; Fields, Vasudevan et Kafai, 2015).

Cette effervescence dans le développement des langages de programmation visuels se reflète également par des investissements monétaires considérables. À titre d'exemple, l'organisme à but non lucratif Code.org, qui a pour mission d'offrir une occasion d'apprentissage de la programmation à tous les élèves et qui développe des activités gratuites en ligne utilisant Scratch, reçoit annuellement plus de 20 millions de dollars en donations provenant principalement de grandes compagnies privées américaines (Code.org, 2023).

Parallèlement, de multiples initiatives ont été consacrées au développement d'outils visant à permettre une meilleure contextualisation des apprentissages en programmation via une incarnation physique et une rétroaction tangible (pour une recension complète, voir Crick, 2017). Diverses entreprises ont ainsi investi de manière substantielle dans le développement d'une gamme impressionnante de robots

éducatifs préfabriqués et programmables, tels que Bee-Bot, Nao, Ozobot, et Sphero. Ces robots varient en complexité et permettent aux élèves d'interagir physiquement avec les concepts de programmation, le plus souvent via un langage de programmation visuel ou des interfaces tactiles.

En parallèle, de nombreux microcontrôleurs comme Arduino, Microbit, Codebug et des microprocesseurs comme Raspberry Pi ont également été développés afin d'introduire aux élèves des bases de l'électronique, tout en permettant une programmation de ces dispositifs. Ceux-ci donnent aux élèves la possibilité de concevoir des circuits électroniques et de programmer leur fonctionnement, créant ainsi une expérience d'apprentissage davantage contextualisée en utilisant de véritables composantes micro-informatiques. Enfin, une multitude de « kits » de robotique ont également vu le jour dans les dix dernières années, dont Lego Mindstorms, Picoboard et Makey Makey. Ces kits offrent des occasions d'apprentissages multidisciplinaires : ils permettent aux élèves de construire et de programmer des projets physiques, fusionnant ainsi les domaines de la science, de la technologie, de l'ingénierie mécanique et électronique, des arts et des mathématiques aux apprentissages clés en programmation.

Tous ces exemples soulignent ainsi la quantité importante d'outils développés spécifiquement pour faciliter l'apprentissage de la programmation chez les jeunes. Or, en plus des ressources engagées par le secteur des entreprises pour le développement de ces outils, les établissements scolaires souhaitant les acquérir ont également dû mobiliser des ressources considérables : d'une part, sur le plan financier en raison du coût souvent élevé de ces outils ainsi que pour assurer leur maintenance et les mises à jour nécessaires (Rich, 2022) et, d'autre part, pour former adéquatement le corps enseignant à leur utilisation, ce qui n'est généralement pas fait au sein des formations universitaires initiales (Spradling, Linville, Rogers et Clark, 2015).

À cet égard, plusieurs études récentes indiquent que les systèmes éducatifs manquent d'enseignants suffisamment qualifiés en informatique, ceux-ci déclarant souvent eux-mêmes avoir peu confiance dans leur compréhension de l'informatique et de la programmation (Ramen, 2015; Alfayez et Lambert, 2019; Royal Society, 2017, Gülbahar et Kalelioğlu 2017). Pour relever ce défi, de nombreux systèmes éducatifs ont ainsi mis en place des programmes de formation en programmation destinés aux enseignants. Ces formations (en présentiel ou à distance) prennent le plus souvent la forme d'ateliers axés sur les langages de programmation à base de blocs ou encore de séminaires visant à familiariser les enseignants avec le domaine de l'informatique et à les former pour accompagner les élèves dans l'acquisition de compétences en programmation (Liu, 2011; Spradling, 2015). Plusieurs entités en faveur de l'apprentissage de la

programmation recommandent même que les systèmes éducatifs lancent rapidement des programmes de certification destinés aux enseignants du primaire et du secondaire (Code.org, 2017; Mills, Coenraad, Ruiz, Burke et Weisgrau, 2021).

Qui plus est, le développement de ces outils a également exigé que les milieux de la pratique développent des interventions et activités pédagogiques intégrant ces nouvelles ressources (García-Peñalvo, 2016; Crick, 2017), ce qui a d'ailleurs donné lieu à une multitude d'études visant à mesurer l'efficacité et le potentiel de ces outils et interventions pour l'apprentissage des élèves. À titre d'exemple, le logiciel Scratch a fait l'objet à lui seul de 55 études dans les 10 dernières années, et cette tendance est encore croissante (Zhang et Nouri, 2019).

Ainsi, la volonté d'introduire les jeunes apprenants à la programmation et de simplifier leur apprentissage a entraîné une allocation substantielle de ressources, à la fois de la part des entreprises, des milieux de la pratique et de la recherche. Étant donné l'ampleur des ressources déployées à cette fin, il apparaît maintenant essentiel d'examiner et de discuter plus en détail des avantages potentiels pouvant découler de l'apprentissage de la programmation chez les jeunes.

1.2 Bénéfices pouvant découler de l'apprentissage de la programmation chez les jeunes

La littérature souligne trois arguments principaux qui justifient la nécessité d'intégrer la programmation au cursus scolaire des jeunes. Comme mentionné précédemment dans ce chapitre, le premier argument repose sur l'importance de la programmation pour le marché du travail actuel et futur. Le deuxième met en avant l'idée que l'apprentissage de la programmation peut contribuer à favoriser une plus grande équité sociale en initiant tous les élèves à l'informatique. Enfin, au-delà de ces considérations d'ordre économique et social, le troisième argument concerne les avantages de la programmation pour le développement de la pensée informatique, qui est susceptible d'apporter des bénéfices transversaux aux apprenants. Il convient de noter que ce dernier argument est le plus fréquemment évoqué pour justifier l'intégration de la programmation dans le cursus scolaire des élèves. Les sections suivantes examineront en détail chacun de ces arguments.

1.2.1 Apprendre à programmer pour répondre aux besoins du marché du travail

D'abord, l'évolution du marché mondial de l'emploi représente l'une des motivations au cœur de l'intégration de la programmation dans les programmes d'étude de plusieurs pays (Blikstein et Moghadam, 2019). Cette motivation, largement promue par les décideurs politiques, est essentiellement liée à la

nécessité de former un plus grand nombre de personnes possédant des compétences en programmation. En effet, le savoir technologique, en particulier dans le secteur des hautes technologies, stimule depuis plus de 20 ans la croissance économique en Amérique du Nord et ailleurs dans le monde (Avsec et Jamšek, 2016). Un nombre croissant d'emplois nécessite ainsi une compréhension approfondie des technologies (Rausch, 1998). Il est d'ailleurs prévu que ce nombre d'emplois augmente dans les prochaines années considérant que la science des données, l'intelligence artificielle et les technologies de décentralisation (p. ex., le *blockchain*) deviennent des domaines de plus en plus dominants du secteur économique (Akter, Michael, Uddin, McCarthy et Rahman, 2022). L'apprentissage de la programmation dès le plus jeune âge est ainsi fréquemment mis de l'avant comme un levier pouvant faciliter l'immersion et la performance de pays dans l'économie numérique (García-Peñalvo, 2016). De surcroît, certains experts argumentent qu'une initiation des jeunes à la programmation pourrait se révéler avantageuse même dans des professions moins technologiques. Un tel apprentissage permettrait aux individus d'aborder et de résoudre des problèmes avec une plus grande facilité, de comprendre les liens complexes entre diverses variables, et de traiter des problèmes techniques lorsque nécessaire, puisqu'ils auraient appris à identifier les questions pertinentes à poser, et surtout à déterminer l'ordre adéquat pour les poser, assurant ainsi des diagnostics plus précis (Nelson, 2009; Ornelas-Marques et Marques, 2012).

Finalement, certains auteurs (Grover, 2017) soutiennent qu'exposer les élèves à la programmation tôt dans le cursus scolaire pourrait avoir une incidence positive sur l'identité qu'ils développent au regard de ce domaine. Cette perspective prend en compte l'existence de nombreux stéréotypes associés à la programmation, notamment l'idée reçue que cette discipline est principalement destinée aux garçons. À cet égard, des arguments allant au-delà des retombées économiques peuvent ainsi être évoqués pour justifier l'intégration de la programmation au cursus scolaire. Cette idée est développée davantage dans la section qui suit.

1.2.2 Apprendre à programmer pour favoriser l'équité et la diversité sociale

Selon certains auteurs, une plus grande exposition à l'informatique par l'apprentissage de la programmation chez les jeunes pourrait en effet contribuer à promouvoir une plus grande équité sociale quant à la représentativité et à l'accès aux professions technologiques (Blikstein et Moghadam, 2019). D'une part, les compétences en sciences informatiques peuvent en effet donner accès à des emplois bien rémunérés, qui pourraient contribuer à assurer une plus grande stabilité financière pour des groupes qui n'ont pas eu la possibilité d'accumuler des richesses dans les dernières générations (Google et Gallup,

2015). D'autre part, exposer plus tôt tous les élèves à la programmation pourrait permettre une plus grande diversité par une participation accrue de personnes issues de groupes actuellement sous-représentés en informatique.

À cet égard, l'entreprise Google, en collaboration avec l'organisation Gallup, a mené une étude longitudinale (2016) sur les groupes sous-représentés aux États-Unis afin de comprendre les obstacles qu'ils rencontrent dans l'accès et l'apprentissage de l'informatique. Le rapport se concentre sur les obstacles structurels et sociaux auxquels sont confrontées les femmes ainsi que les personnes noires et hispaniques à l'école et dans la société, et qui peuvent affecter leur probabilité de poursuivre des études en informatique. Les principales conclusions de leur étude indiquent notamment que les étudiants noirs sont moins susceptibles que les étudiants blancs d'avoir des cours consacrés à la programmation dans leur école, et sont plus susceptibles de devoir apprendre l'informatique de leur propre initiative, en dehors de la classe, comme dans des clubs parascolaires. Ils soulignent également que les étudiants noirs et hispaniques sont moins susceptibles que les élèves blancs d'utiliser régulièrement un ordinateur à la maison, ce qui pourrait affecter leur confiance dans l'apprentissage de la programmation. Or, les élèves qui ne voient pas souvent des gens « faire de la programmation », en particulier des gens auxquels ils peuvent facilement s'identifier, pourraient avoir du mal à s'imaginer en train de faire eux-mêmes de la programmation. Finalement, il semble aussi que les étudiantes soient moins au courant des possibilités d'apprentissage de la programmation et moins susceptibles d'être intéressées à l'apprendre. Elles sont également souvent moins confiantes que les étudiants masculins dans leur capacité à apprendre les sciences informatiques. À l'inverse, les garçons sont plus susceptibles d'être encouragés par leurs parents et leurs enseignants à poursuivre des études en informatiques. Ces éléments supportent donc l'idée que l'intégration de la programmation au cursus scolaire de tous les jeunes élèves pourrait contribuer à une plus grande diversité au sein des professions technologiques.

Or, une meilleure diversité au sein des travailleurs dans le domaine des technologies conduirait à la conception de meilleurs produits (Hunt, 2018) destinés et accessibles à une plus grande partie des consommateurs sur le marché (K-12 Computer Science Framework, 2016). Une trop grande homogénéité au sein des travailleurs mène en effet à la conception de produits et de services qui répondent à un spectre relativement restreint d'individus et de problèmes, ce qui est susceptible de contribuer à renforcer les inégalités. Les chercheurs qui mettent de l'avant cet argument d'équité soutiennent d'ailleurs que si des mesures rapides et intentionnelles ne sont pas prises pour favoriser une plus grande diversité au sein des

professions technologiques, cela pourrait donner lieu à un « fossé numérique » beaucoup plus marqué dans les années à venir (Lewis, 2019). L'apprentissage de la programmation pour tous (*Computing for All*, Ladner et Israel, 2016) pourrait en ce sens représenter une mesure afin de diminuer ce fossé et exposer plus tôt tous les élèves à l'apprentissage de la programmation pourrait ainsi contribuer à une plus grande équité sociale.

Finalement, au-delà de ces bénéfices indéniables sur les plans de l'emploi et de l'équité sociale, le principal argument présenté dans la littérature, sur lequel se sont basés plusieurs systèmes éducatifs pour intégrer la programmation à leur cursus scolaire, réside dans les avantages potentiels que cet apprentissage peut apporter à l'échelle individuelle, c'est-à-dire pour les apprenants eux-mêmes. La prochaine section discute plus en détail de cette idée centrale.

1.2.3 Apprendre à programmer pour développer sa pensée et faciliter l'apprentissage de manière transversale

Ce troisième argument concerne spécifiquement le rôle que jouerait la programmation dans le développement d'une pensée informatique chez l'apprenant (Lye et Koh, 2014). Défini et popularisé par Wing (2006), le concept de pensée informatique réfère aux habiletés de « résolution de problèmes, de conception de systèmes et de compréhension du comportement humain s'appuyant sur les concepts fondamentaux de l'informatique » (p. 33). Plusieurs auteurs soutiennent que le développement d'une telle pensée informatique serait bénéfique pour l'apprenant, car elle lui permettrait notamment de développer des compétences en raisonnement de haut niveau susceptibles d'être transférées à d'autres apprentissages, tels que la résolution de problème, la créativité et l'abstraction (Shute, 2017). Pour ces raisons, la pensée informatique est souvent intégrée à même les nouveaux programmes d'apprentissage de la programmation (Bocconi, 2016), comme dans le cursus de l'Angleterre (Department for Education, 2013) où il est dit qu'« une éducation informatique de haute qualité équipe les élèves pour utiliser la pensée informatique et la créativité pour comprendre et changer le monde » (p.1). La pensée informatique est notamment considérée selon l'UNESCO (2011) comme étant une compétence critique nécessaire au citoyen contemporain afin qu'il puisse adéquatement s'adapter à la société et qu'il y développe une certaine autonomie. À cet égard, l'introduction de la programmation au cursus scolaire pourrait donc avoir un avantage pour tous les élèves, même ceux qui ne se destinent pas à une carrière technologique, car ils pourraient profiter du développement d'une pensée informatique dans leur quotidien d'une manière plus transversale.

La pensée informatique est parfois perçue, pour certains chercheurs, comme un « terme parapluie », englobant une série d'effets bénéfiques liés à l'apprentissage de la programmation. En fait, au sein même de la définition classique de la pensée informatique se trouve l'idée de transfert. Wing (2011) affirme à ce propos : « Les bénéfices éducatifs de la capacité à penser de manière informatique — commençant par l'utilisation d'abstractions — renforcent et améliorent les compétences intellectuelles, et peuvent ainsi être transférés à n'importe quel domaine » (p.7). En d'autres termes, la pensée informatique ne référerait pas à une compétence spécifique, mais bien à une gamme d'effets potentiellement transférables découlant de l'apprentissage de la programmation. La pensée informatique engloberait notamment les compétences suivantes : l'abstraction et la généralisation des modèles (Grover et Pea, 2013), le traitement systématique des informations (Shute, 2017), les systèmes de symboles et les représentations (Thorndike et Woodworth, 1901), la pensée algorithmique (Clements, 1995), la décomposition des problèmes (Grover et Pea, 2013) et la détection et la correction d'erreurs systématiques (Popat et Starkey, 2019). Ces compétences présentent des similitudes considérables avec celles nécessaires pour résoudre des problèmes généraux et spécifiques, et ce, pour plusieurs autres disciplines scolaires (Shute, 2017). Cette vision élargie de la pensée informatique suggère que l'apprentissage de la programmation pourrait influencer la manière d'aborder et de résoudre des problèmes dans diverses disciplines scolaires. Des chercheurs ont d'ailleurs fait valoir de manière théorique que l'intégration de la programmation dans le programme scolaire pourrait entraîner plusieurs apprentissages transférables (Kaleliolu, 2015; Moreno-León, 2016; Resnick, 2009; Wing, 2006).

Cependant, bien qu'un nombre croissant de recherches se soient ainsi intéressées aux effets de l'apprentissage de la programmation sur le développement de la pensée informatique des jeunes qui apprennent à programmer (Grover et Pea, 2013; Lye et Koh, 2014), un nombre plus restreint a examiné les effets de transfert de cet apprentissage pour des compétences ou des disciplines scolaires en dehors de l'informatique. C'est la raison pour laquelle la prochaine section poursuit cette réflexion sur le transfert potentiel résultant de l'apprentissage de la programmation.

1.3 Étude des effets de transfert associés à l'apprentissage de la programmation chez les jeunes : un corpus scientifique grandissant, mais encore contradictoire et limité

Au cours de la dernière décennie, le domaine de l'éducation s'est fortement orienté vers le développement des compétences du 21e siècle, notamment la pensée critique, la résolution de problèmes, la collaboration et la créativité (Pellegrino et Hilton, 2012). Ces compétences transversales, applicables à une variété de disciplines et de situations complexes, visent à préparer les élèves à relever les défis du

monde réel et à s'adapter à un environnement en constante évolution. En conséquence, le développement de ces compétences est devenu une priorité pour de nombreux enseignants (Kay et Greenhill, 2011). Les méthodes visant à encourager le développement de ces compétences et à faciliter leur application dans divers contextes et domaines de connaissances ont donc été de plus en plus explorées (Greiff, 2014).

C'est dans cette perspective que certains chercheurs ont formulé l'hypothèse que l'apprentissage de la programmation pourrait engendrer des avantages pour les apprenants en termes de transfert des apprentissages; les connaissances et compétences développées par la programmation étant susceptibles d'être réinvesties dans le cadre d'autres apprentissages scolaires (Grover et Pea, 2013; Liao et Bright, 1991; Pea et Kurland, 1984). Cette hypothèse repose d'une part sur le développement de la pensée informatique, dont il était question dans la section précédente, et d'autre part sur les similitudes existant entre les compétences requises en programmation, telles que la décomposition, l'abstraction, l'itération et la généralisation, et celles nécessaires pour résoudre des problèmes dans d'autres domaines, notamment en mathématiques et en sciences (Román-González, Pérez-Fernández et Jiménez-Fernández, 2017; Shute, Sun et Asbell-Clarke, 2017).

Cependant, malgré cette logique apparente sur le plan théorique, les recherches actuelles présentent des résultats divergents quant à la transférabilité des compétences acquises grâce à l'apprentissage de la programmation. Certains montrent ainsi des effets de transfert, tandis que d'autres n'en trouvent pas (Scherer *et al.*, 2019). En outre, ce manque de consensus en matière de transfert ne se limite pas à l'apprentissage de la programmation, mais s'étend également à d'autres domaines identifiés comme favorables au développement de compétences cognitives transversales, tels que l'enseignement des échecs, l'éducation musicale et l'entraînement de la mémoire de travail (Sala et Gobet, 2017a). Malgré une vaste littérature de recherche portant sur le transfert des apprentissages, les conclusions demeurent donc peu claires; et plusieurs auteurs soulignent qu'il y aurait une difficulté réelle à observer du transfert entre les apprentissages (Barnett et Ceci, 2002; Day et Goldstone, 2012; Lobato, 2006). Pour cette raison, certains auteurs vont jusqu'à remettre en question l'idée même du transfert (Denning, 2017).

Au regard de l'apprentissage de la programmation, les résultats de recherche existants (Liao et Bright, 1991; Liao, 2000, Scherer *et al.*, 2019, Starkey et Popat, 2019) révèlent des résultats mitigés, suggérant que la programmation aurait le potentiel d'avoir des effets de transfert globalement positifs, parfois neutres, et rarement négatifs; mais certaines limites méthodologiques importantes amènent toutefois à

considérer avec prudence ces premiers résultats. Une première méta-analyse (Liao et Bright, 1991) a en ce sens révélé que les compétences en programmation pouvaient être transférées à différents contextes de manière modérée, mais que cet effet de transfert serait influencé par des facteurs tels que le type de contextes, le niveau scolaire, les langages de programmation utilisés et la durée des interventions visant l'apprentissage de la programmation.

Par ailleurs, une deuxième méta-analyse de Liao (2000) a pour sa part démontré un effet de transfert global positif important de la programmation vers différentes habiletés cognitives (résolution de problèmes, pensée critique, habiletés spatiales, etc.). Néanmoins, en raison de l'inclusion d'études sans groupe de contrôle au sein de leur échantillon, certains auteurs (Scherer *et al.*, 2019) mettent en doute la validité des résultats obtenus et incitent à considérer ceux-ci avec prudence, l'absence d'un groupe de contrôle pouvant mener à un effet « gonflé », ce qui pourrait expliquer l'effet global très positif en contraste avec les résultats précédents. Cette deuxième méta-analyse a aussi souligné que les effets de transfert différaient selon le type de transfert. Ainsi, pour un transfert « de proximité » (où la situation de transfert est étroitement liée à la situation d'apprentissage initiale), les effets étaient plus prononcés. En revanche, pour un transfert « étendu » (où la situation de transfert est éloignée ou différente de la situation d'apprentissage initiale), les effets étaient nettement moindres.

Bien que ces résultats initiaux ne convergent pas de manière complète, ils suggèrent la possibilité qu'il puisse y avoir un transfert entre les compétences développées par la programmation et d'autres domaines d'apprentissage. Néanmoins, l'ensemble des études incluses dans ces deux méta-analyses ont été menées auprès d'apprenants adultes inscrits majoritairement dans des programmes universitaires visant spécifiquement l'apprentissage de la programmation. Il n'est donc pas certain que des effets de transfert similaires seraient observés pour des apprenants novices de niveau primaire ou secondaire.

À cet égard, une revue de la littérature récente (Popat et Starkey, 2019) s'est intéressée de manière spécifique à l'effet de transfert associé à l'apprentissage de la programmation pour des élèves de niveaux primaire et secondaire. À partir de cette revue, les auteurs ont identifié trois grandes catégories de transfert possibles associées à l'apprentissage de la programmation : 1- un transfert vers des compétences en raisonnement de haut niveau (principalement la résolution de problème et la pensée critique), 2- un transfert vers des compétences personnelles (principalement des compétences communicationnelles et de collaboration) ainsi que 3- un transfert vers des compétences en raisonnement de bas niveau (les

savoirs et compétences académiques). Pourtant, selon ces auteurs, cette dernière catégorie demeure à ce jour beaucoup moins documentée. En effet, bien que de premiers résultats prometteurs aient été obtenus pour les mathématiques et d'autres domaines liés à l'informatique (Jona, 2014; Wilensky, 2014), le corpus de recherches explorant la question du transfert vers d'autres disciplines scolaires chez de jeunes apprenants demeure plus limité.

À notre connaissance, il n'existe à l'heure actuelle qu'une seule méta-analyse s'étant directement intéressée aux effets de transfert découlant de l'apprentissage de la programmation chez les jeunes apprenants (Scherer *et al.*, 2019). Or, au terme de cette méta-analyse, il ressort d'abord que les effets de transfert vers les disciplines scolaires demeurent moins documentés –à l'exception de la discipline des mathématiques qui a fait l'objet d'une attention plus soutenue – mais surtout qu'il est difficile de conclure à des effets de transfert très convaincants.

Pourtant, considérant que l'un des arguments clés, voire le plus important, en faveur de l'intégration de la programmation au cursus des élèves du primaire et de secondaire repose sur le développement d'habiletés cognitives ayant des retombées positives dans divers contextes en dehors du domaine de l'informatique, établir de manière solide la présence d'effets de transfert bénéfiques pour d'autres disciplines scolaires s'impose comme une première étape cruciale. Dans cette optique, il apparaît d'autant plus essentiel de le faire pour des disciplines traditionnelles qui occupent une place centrale dans de nombreux programmes d'études, à savoir les mathématiques, la langue d'enseignement (au Québec, le français), et les sciences.

1.4 Question de recherche

La problématique à l'origine de cette recherche s'ancre en premier lieu dans la tendance mondiale à vouloir intégrer la programmation au cursus scolaire des apprenants. Cependant, il est reconnu que l'apprentissage de la programmation est particulièrement complexe et exigeant. Cette complexité atteint un niveau encore plus élevé pour les jeunes apprenants, ce qui pose un défi supplémentaire lorsqu'il s'agit de réfléchir à l'intégration de la programmation dans leur cursus scolaire. C'est pourquoi, au cours des dernières années, des ressources substantielles ont été investies pour faciliter cet apprentissage auprès des jeunes. Ces investissements ont été faits sur la base qu'apprendre tôt à programmer présenterait de nombreux bénéfices, tant sur le plan social qu'individuel. L'argument central avancé pour justifier l'intégration de la programmation au cursus scolaire est que cet apprentissage entraînerait des bénéfices

significatifs pour les élèves, allant bien au-delà du simple codage. Le bénéfice le plus anticipé réside dans le développement de la pensée informatique qui favoriserait le transfert des apprentissages développés par la programmation vers d'autres disciplines scolaires, contribuant ainsi à leur réussite.

Pourtant, en dépit de l'enthousiasme entourant cette perspective, la littérature existante ne permet pas de confirmer ces effets de transfert. Certains auteurs incitent même à la prudence, soulignant le manque de preuves empiriques solides pour appuyer cette idée. Il apparaît donc crucial de conduire des recherches plus approfondies sur ce sujet. C'est dans ce contexte que se pose la question de recherche de ce projet qui vise à examiner les effets de transfert possibles associés à l'apprentissage de la programmation vers les autres disciplines scolaires, en particulier les disciplines scolaires traditionnelles telles que les mathématiques, le français et les sciences.

Ce projet de recherche aspire donc à répondre à la question suivante :

> **Quels sont les effets de transfert associés à l'apprentissage de la programmation chez les jeunes apprenants sur les disciplines scolaires traditionnelles ?**

Sur le plan scientifique, ce projet de recherche contribuera à combler des lacunes importantes dans les connaissances actuelles au sein du domaine de la didactique de la programmation, spécifiquement en ce qui a trait aux effets de transfert associés à son apprentissage. L'importance de mener davantage de recherches sur l'effet de transfert de l'apprentissage de la programmation est évidente au vu des limites des études existantes. Ce projet permettra ainsi d'offrir un éclairage supplémentaire sur le processus complexe qu'est l'apprentissage de la programmation, en particulier sur les compétences mobilisées lors de celui-ci, et de ses impacts potentiels pour ses apprenants. Ces résultats apporteront donc des informations supplémentaires quant à la pertinence de l'intégration de la programmation au cursus scolaire et sont susceptibles d'éclairer les choix politiques et pédagogiques au regard de l'enseignement de la programmation.

Sur le plan éducatif, les résultats de cette recherche pourraient servir de référence pour guider les décisions quant à la place que pourrait prendre la programmation à l'école, qu'il s'agisse d'une intégration disciplinaire, interdisciplinaire ou transversale. Étant donné que le Québec évalue encore la place que pourrait occuper l'apprentissage de la programmation dans son programme scolaire, en particulier en tant qu'approche transversale, il est judicieux de mieux comprendre si des effets de transfert peuvent

réellement être attendus. De même, les résultats de cette étude pourront contribuer à préciser l'action pédagogique des enseignants s'intéressant à l'enseignement de la programmation. En effet, mieux comprendre quelles sont les disciplines scolaires possédant des éléments communs avec la programmation et pour lesquelles un transfert est possible pourrait permettre de mieux orienter la contextualisation des apprentissages afin de proposer des situations interdisciplinaires signifiantes aux élèves.

Sur le plan social, ce projet de recherche pourrait finalement contribuer à réduire certaines inégalités. Au Québec et ailleurs, d'importantes iniquités persistent quant à l'environnement éducatif au sein duquel évoluent les élèves. Or, de l'avis de plusieurs chercheurs (Lewis, 2019), une plus grande exposition à la programmation chez les jeunes pourrait contribuer à favoriser leur réussite éducative et à plus long terme, à promouvoir une plus grande équité, diversité et inclusion, notamment en ce qui concerne la représentativité et l'accès aux professions technologiques. Ce projet s'inscrit à cet égard dans une vision d'apprentissage de la programmation pour tous.

En somme, cette recherche présente une pertinence scientifique, éducative et sociale importante. Elle pourra contribuer à mieux identifier les bénéfices découlant de l'apprentissage de la programmation pour les apprenants, au regard des autres disciplines scolaires. Ces nouvelles connaissances sont susceptibles de fournir un éclairage nouveau quant aux points de convergence entre l'apprentissage de la programmation et les autres apprentissages qui sont attendus de l'élève. Les résultats obtenus seront donc de nature à bonifier les pratiques pédagogiques des enseignants, et à alimenter les réflexions au regard de l'intégration de la programmation dans le cursus scolaire du Québec et d'ailleurs.

CHAPITRE 2
CADRE THÉORIQUE

Ce projet de recherche s'intéresse aux effets de transfert pouvant découler de l'apprentissage de la programmation chez les jeunes. Les deux concepts centraux à ce projet sont donc : 1- l'apprentissage de la programmation et 2- le transfert des apprentissages. Dans un premier temps, ce chapitre aborde les origines et la signification de la programmation. Il est également question des principaux points d'ancrage théoriques relatifs à l'apprentissage de la programmation. Puis, le concept de pensée informatique permettant de faire le pont vers le transfert des apprentissages est ensuite présenté et discuté. La deuxième section du chapitre définit ensuite ce qu'est le transfert des apprentissages et en présente différentes typologies. L'accent est mis sur celles qui sont principalement utilisées dans le contexte du transfert associé à l'apprentissage de la programmation. À la suite de la présentation de ces concepts, une troisième section propose un état de connaissances en ce qui a trait aux effets de transfert de l'apprentissage de la programmation vers les apprentissages des autres disciplines scolaires traditionnelles. Finalement, le chapitre se termine par la présentation des objectifs spécifiques et des hypothèses de recherche.

2.1 Apprentissage de la programmation

Depuis les premiers jours de l'informatique, l'apprentissage de la programmation est au cœur du domaine. Cette section vise précisément à explorer les dimensions clés de l'apprentissage de la programmation. Dans un premier temps, il apparaît essentiel de présenter plusieurs définitions qui soulignent différents aspects du concept de la programmation et de le distinguer d'autres concepts apparentés.

2.1.1 Programmation : définitions et distinctions conceptuelles

Le terme « programmation » est polysémique et sa signification varie en fonction du domaine et du contexte d'application. En termes généraux, selon le dictionnaire Larousse, programmer signifie « établir à l'avance une suite d'opérations; planifier, déterminer à l'avance le moment et les modalités d'une action » (2023, en ligne). Cette définition générale permet ainsi d'utiliser le terme dans divers contextes, par exemple pour exprimer que quelqu'un a planifié (programmé) sa journée ou qu'une réunion a été prévue (programmée) pour le matin. De façon similaire, le Merriam-Webster (2023, en ligne) indique que programmer implique : « élaborer une séquence d'opérations à effectuer » (traduction libre). Ces deux

définitions convergent principalement vers l'idée de planification préalable d'une série d'opérations ou d'actions. Par ailleurs, le Larousse propose également une définition de la notion de « programmation » dans le contexte spécifique des sciences informatiques : « l'ensemble des activités liées à la définition, l'écriture, la mise au point et l'exécution de programmes informatiques; séquence des ordres auxquels doit obéir un dispositif » (2023, en ligne). Similairement, le Merriam-Webster indique pour sa part qu'il s'agit du : « processus de préparation d'un programme d'instructions pour un appareil tel qu'un ordinateur » (traduction, libre). Il semble donc que dans ce contexte précis, l'acte de programmer implique la création, la mise en œuvre, et la séquence des instructions ou des ordres destinés à un dispositif informatique. Ces deux définitions de la programmation informatique rejoignent ainsi celles plus générales de l'acte de programmer puisqu'elles impliquent également d'établir à l'avance une suite d'opérations, de planifier et de déterminer à l'avance le moment et les modalités d'une action.

Or, bien que le domaine de l'informatique soit d'intérêt dans le contexte de cette thèse, il importe de mentionner que le terme programmation peut parfois être utilisé dans d'autres disciplines. Par exemple, en génétique, la programmation peut faire référence à la modification de séquences d'ADN pour contrôler le comportement cellulaire (Endy, 2005). En psychologie, la programmation neurolinguistique (PNL) est une approche de la communication et du développement personnel (Esser, 2004). En économie, la programmation budgétaire réfère à la planification des dépenses et des recettes d'une organisation (Nafzaoui, 2021). La signification du terme programmation peut donc varier d'un domaine à l'autre, et il est souvent défini en fonction du contexte spécifique de chaque discipline. Dans le cadre de cette thèse, le terme programmation sera utilisé pour référer à la notion de programmation informatique.

En ce qui concerne la discipline de l'informatique, également connue sous les appellations anglophones de *Computer Science* ou *Computing*, il s'agit d'un domaine vaste et multidimensionnel. En Europe, le terme « informatics » est souvent utilisé comme synonyme, bien que le choix des termes employés puisse varier en fonction des régions et des contextes académiques. L'informatique englobe une variété de domaines, notamment la conception de systèmes logiciels (*software*) et de systèmes micro-informatiques (*hardware*), l'optimisation du fonctionnement des ordinateurs, l'analyse de données, l'intelligence artificielle, la réseautique, et bien d'autres. Parmi ces domaines, il est largement reconnu que la programmation occupe une place centrale (Guzdial, 2019).

D'un point de vue historique, bien que certains attribuent à Ada Lovelace la conceptualisation des premiers principes de programmation en 1840, la discipline de la programmation informatique en tant que telle n'a pris forme qu'au milieu du XX^{e} siècle (Randell, 1994). En effet, le terme « programmation informatique » est apparu au cours de la première moitié du XX^{e} siècle, avec le développement des premiers ordinateurs électroniques (Randell, 1994). L'utilisation de ces machines nécessitait la création de séquences d'instructions complexes pour effectuer des calculs et des traitements de données. C'est à cette époque que des pionniers de l'informatique tels que Alan Turing, Konrad Zuse, et John von Neumann ont commencé à développer des méthodes pour écrire des programmes informatiques (Wexelblat, 2014). L'un des premiers langages de programmation, le Fortran, a été créé par IBM dans les années 1950. Il a contribué à populariser la programmation informatique en permettant aux programmeurs d'exprimer des instructions en termes plus proches de l'algèbre que du langage machine (Wexelblat, 2014). Aujourd'hui, la programmation est utilisée dans une grande variété d'applications : développement de logiciels et d'applications mobiles, sites web, systèmes d'exploitation, intelligence artificielle, applications scientifiques et de recherche, jeux vidéo, automatisation et robotique, applications médicales, applications financières, sécurité informatique, et plusieurs autres.

Concrètement, programmer implique la création d'instructions précises pour un ordinateur, sous forme de lignes de code, en utilisant un langage de programmation, qui est un ensemble de règles et de symboles permettant de communiquer avec la machine (Meyer, 2022). Ces instructions forment un programme, c'est-à-dire un ensemble de lignes de code conçu pour effectuer des tâches spécifiques, de la simple addition à des opérations complexes, en donnant à l'ordinateur des étapes à suivre pour atteindre un objectif défini. En somme, programmer consiste à utiliser un langage de programmation pour développer un programme qui régit le comportement d'un ordinateur (Wexelblat, 2014).

Afin de donner une représentation plus concrète de ce à quoi ressemble la programmation, la Figure 2.1 illustre un exemple d'une série de lignes de code visant à réaliser une boucle, une fonction qui permet de répéter l'exécution d'une séquence d'instructions, selon A) un format textuel utilisant le langage JavaScript et B) un format visuel utilisant le langage Scratch destiné à de plus jeunes apprenants.

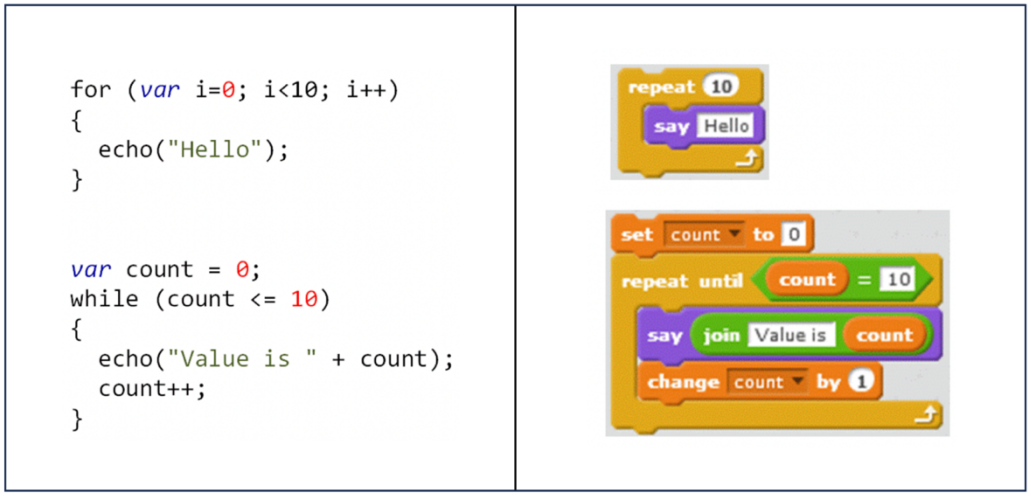

Figure 2.1 Exemples de lignes de code visant à réaliser une boucle selon A) un format textuel utilisant le langage JavaScript et B) un format visuel utilisant le langage Scratch.

Outre les définitions plus générales de la programmation mentionnées précédemment, de nombreux chercheurs dans ce domaine ont également avancé des définitions qui mettent en lumière divers aspects spécifiques de la programmation. Mills (2021) définit pour leur part la programmation comme « la pratique qui consiste à développer un ensemble d'instructions qu'un ordinateur peut comprendre et exécuter, ainsi qu'à déboguer, organiser et appliquer ce code dans des contextes de résolution de problèmes appropriés » (traduction libre, p.10). Cette définition met l'accent sur les aspects pratiques et techniques de la programmation, tels que le développement d'instructions, le débogage et l'organisation du code. L'accent est mis sur l'aspect fonctionnel de la programmation, dont l'objectif est de faire en sorte qu'un ordinateur exécute des instructions et résoudre des problèmes. De la même façon, dès 1963, l'Association for Computing Machinery (ACM) proposait dans son glossaire de définir la programmation comme : « L'art et la science de créer une séquence logique d'instructions pour guider les actions d'un système informatique » (Fritz, 1963, p. 156, traduction libre). Pour renforcer cette perspective, Turski (1978) affirme que « le but de la rédaction d'un programme est de communiquer des instructions complètes, détaillées et sans ambiguïté pour atteindre un objectif spécifique » (Turski et Wasserman, 1978, p. 6, traduction libre).

À cet égard, Tew et Gudzial (2010) ont identifié les dix concepts de base qui entrent en jeu lors de la programmation, à partir des manuels les plus populaires des cours d'introduction à la programmation. Ces concepts sont brièvement présentés dans le Tableau 2.1. Bien que technique, cette synthèse permettra au lecteur d'avoir une compréhension plus concrète de ce à quoi réfère l'acte de programmer.

Tableau 2.1 Présentation des dix concepts de base les plus centraux à la programmation informatique, tels qu'identifiés par Tew et Gudzial (2010)

Concepts de base en programmation	**Définitions**
Fondamentaux	Concepts de base tels que les variables, les types de données et les opérations arithmétiques.
Opérateurs logiques	Opérateurs tels que *and*, *or* et *not*, qui sont utilisés pour effectuer des opérations logiques.
Structures conditionnelles	Structures de contrôle, comme les instructions *if*, *else*, et *elseif*, qui permettent d'exécuter du code en fonction de certaines conditions.
Boucles définies	Boucles avec un nombre défini d'itérations, généralement réalisées avec des structures comme *for*.
Boucles indéfinies	Boucles sans un nombre prédéfini d'itérations, souvent mises en œuvre avec des structures comme *while*.
Tableaux	Structures de données qui stockent une collection ordonnée d'éléments de même type.
Paramètres de fonction	Variables dans la définition d'une fonction ou d'une méthode qui acceptent des valeurs lors de l'appel de la fonction.
Valeurs de retour de fonction	Valeurs que renvoie une fonction ou une méthode après son exécution.
Récursivité	Technique où une fonction s'appelle elle-même dans sa propre définition.
Bases de la programmation orientée objet	Concepts de base de la programmation orientée objet, comme les classes, les objets, l'héritage, et le polymorphisme.

Définitions inspirées de « Programming 101 : The How and Why of Programming Revealed Using the Processing Programming Language » par Meyer, 2018, Apress.

Par ailleurs, certains auteurs en didactique de la programmation élargissent cette définition classique de la programmation en intégrant des dimensions supplémentaires au concept. C'est notamment le cas de Guzdial (2019) qui définit la programmation comme « un moyen de communication, de partage, d'exploration d'idées, et un outil d'apprentissage et d'exécution de tâches ». (p. 596). Cette définition de Guzdial est plus large et souligne l'idée que la programmation n'est pas seulement une compétence technique, mais aussi un moyen de communication et un outil, tant pour explorer des idées que pour soutenir l'apprentissage. Cette définition rejoint ainsi l'idée de pensée informatique qui avait été brièvement abordée dans la problématique et qui fait référence à la capacité générale à résoudre des problèmes de manière logique et structurée en utilisant des compétences issues de l'informatique. Étant donné son importance, le concept de pensée informatique fait d'ailleurs l'objet d'une section à part entière du présent chapitre (section 2.1.3).

Similairement, Tricot (2020), aborde la programmation comme une fonction pédagogique dont l'intention peut être d'apprendre à programmer ou d'utiliser la programmation comme moyen d'apprendre autre chose. Tricot met pour sa part l'accent sur l'aspect éducatif de la programmation qui est considérée comme un outil d'enseignement et d'apprentissage, qu'il s'agisse de réaliser des apprentissages à propos de la programmation ou de l'utiliser comme outil pédagogique pour réaliser des apprentissages à propos d'un autre sujet. L'accent est donc mis sur le potentiel pédagogique de la programmation.

En somme, si les définitions générales de la programmation s'accordent sur le fait qu'elle désigne un ensemble d'instructions destinées à un tiers parti, la plupart du temps à un ordinateur, elles diffèrent parfois dans leur orientation. La première met en effet l'accent sur la programmation comme objet d'étude, tandis que les deux autres soulignent sa nature polyvalente et ses diverses finalités. Ce projet de recherche s'inscrit dans cette double perspective : la programmation y est non seulement considérée comme une discipline qui s'intéresse à la mise en place d'instructions qui sont interprétables par une machine, mais également comme un outil d'apprentissage qui pourrait s'avérer bénéfique pour ses apprenants.

Par ailleurs, dans les dernières années, de nombreuses organisations parascolaires ayant comme mission d'exposer les élèves du primaire et/ou du secondaire à la programmation ont fait le choix d'utiliser le terme « codage » plutôt que celui de « programmation ». C'est notamment le cas des très populaires organismes américains Code.org et It's hour of code, d'organismes européens tels que The EU code week,

et québécois comme Code Mtl. Il apparaît donc important de situer et distinguer le terme « codage » par rapport à celui de « programmation », ce dernier étant central au présent projet.

À cet égard, Martin (2017) propose une distinction entre ces deux concepts : le codage est l'action spécifique d'écriture de code à l'aide d'un langage précis, tandis que la programmation est l'action plus générale de réflexion, conception, création et mise à l'essai d'algorithmes. Pour illustrer cette différence, Martin (2017) utilise l'exemple d'une entreprise au sein de laquelle un programmeur aurait pour mandat de conceptualiser les étapes claires et partageables d'une solution à un problème ou à un besoin (parfois directement sous forme de langage codé, mais parfois aussi en langage usuel, c'est-à-dire en français ou en anglais). Dans cette même entreprise, un codeur possèderait un rôle plus technique de rédaction et de traduction d'un langage usuel à un langage informatique ou d'un langage informatique à un autre langage. De ce fait, le codeur possède souvent une connaissance syntaxique et fonctionnelle plus approfondie des langages informatiques. Il n'est toutefois pas rare qu'une seule et même personne effectue l'ensemble de la tâche considérant que les programmeurs sont suffisamment à l'aise avec le codage informatique.

De façon convergente, Robins et ses collaborateurs (2003) ainsi que Mannila et ses collaborateurs (2014) suggèrent que le codage est un élément de la programmation principalement axé sur l'écriture du code, tandis que la programmation constitue un processus plus holistique qui implique plusieurs savoirs et compétences interreliés, dont l'écriture du code. Certains auteurs proposent donc de considérer la programmation comme un « exercice beaucoup plus large qui englobe des activités de conception, d'écriture, de test et de maintenance; [qui] s'apparente autant, voire davantage, à la résolution de problèmes qu'à l'écriture d'un code » (Henri, 2014, p. 11).

Ces définitions et distinctions sont intéressantes, car elles permettent de mieux circonscrire le concept de programmation. Il apparaît néanmoins essentiel de mieux le situer sur le plan théorique et de comprendre et identifier ce que signifie plus concrètement l'acte même de programmer. À ce propos, des auteurs ont proposé un modèle qui permet d'appréhender de manière plus concrète les connaissances et compétences mobilisées par la programmation. Une meilleure compréhension de ces éléments permettra par la suite d'établir un lien plus clair avec ses effets potentiels en matière de transfert. La prochaine section présente donc les principaux ancrages théoriques propres à l'apprentissage de la programmation.

2.1.2 Principaux ancrages théoriques de l'apprentissage de la programmation

Dans le cadre de ce projet de recherche axé sur l'apprentissage de la programmation, il apparaît d'abord essentiel d'examiner la théorie du constructionnisme. Celle-ci revêt en effet une pertinence significative dans le contexte de ce projet, car elle souligne l'importance de l'apprentissage en faisant et en explorant, des principes qui apparaissent fondamentaux pour les apprenants qui cherchent à maîtriser la programmation informatique.

2.1.2.1 Constructionnisme : une théorie intrinsèquement liée à l'apprentissage de la programmation

La théorie du constructionnisme, développée par Seymour Papert (Papert, 1980), se révèle d'une grande pertinence dans le contexte de l'apprentissage de la programmation. Ce dernier a développé sa théorie du constructionnisme en s'appuyant sur les fondements du constructivisme, une théorie de l'apprentissage qui met l'accent sur le rôle actif de l'apprenant dans la construction de ses connaissances. Le constructivisme soutient que l'apprentissage se produit lorsque les apprenants interagissent avec leur environnement, manipulent des objets et construisent activement leur compréhension du monde. Cette perspective a été influencée par les travaux du psychologue Jean Piaget, qui a largement contribué à la théorie constructiviste de l'apprentissage.

Seymour Papert, un ancien élève de Piaget, a puisé dans ces idées constructivistes pour développer sa propre théorie du constructionnisme. Il a conçu le constructionnisme en insistant sur l'importance de l'apprentissage à travers la création et la matérialisation des idées. L'acte de construire quelque chose de concret, de tangible, est au cœur de cette théorie. Elle promeut l'idée que l'apprentissage se produit de manière plus significative lorsque les apprenants sont activement engagés dans des processus de création tangibles et partageables. Cela se traduit par la matérialisation des idées en objets concrets, qui peuvent être examinés, partagés, et critiqués (Papert et Harel, 1991). Papert affirme également que cet apprentissage est encore plus efficace lorsque ces créations sont destinées à un public, ce qui encourage les apprenants à réfléchir, à affiner et à améliorer leur compréhension.

L'accent mis par Papert sur la création d'une « entité publique » est l'une des caractéristiques distinctives du constructionnisme. Selon ce dernier, cette création peut prendre diverses formes, que ce soit la construction d'un robot, la programmation d'un ordinateur, la création d'un projet artistique, ou la rédaction d'un texte. Le point essentiel est que les apprenants matérialisent leurs idées, les rendent concrètes et partageables, ce qui selon lui favorise un apprentissage plus profond et significatif. Dans le

contexte de l'apprentissage de la programmation, le constructionnisme suggère que les étudiants acquièrent une compréhension plus approfondie des concepts informatiques en engageant activement des processus de résolution de problèmes, de débogage et de création de programmes. L'acte de programmer permet ainsi aux étudiants de matérialiser leurs idées en code, ce qui peut aider à clarifier leur pensée et à renforcer leur compréhension du matériel (Resnick, 2009). De plus, le constructionnisme avance l'idée que la programmation, par sa capacité à « créer » quelque chose de concret, peut servir d'outil pour faciliter l'apprentissage dans des disciplines où il est souvent difficile de matérialiser les concepts, comme les mathématiques. Les mathématiques sont en effet souvent considérées comme abstraites et théoriques, mais la programmation peut offrir un moyen de les rendre tangibles. Selon cette perspective constructionniste, la programmation peut ainsi servir d'« objet pour penser », aidant ainsi les apprenants à conceptualiser et à comprendre des idées complexes (Papert, 1980).

De ce fait, l'apprentissage de la programmation peut donc offrir des opportunités interdisciplinaires. Par exemple, les concepts mathématiques peuvent être intégrés dans des projets de programmation, permettant ainsi une application pratique et contextualisée des théories mathématiques (Noss et Hoyles, 1996). De même, la programmation peut être utilisée pour modéliser des phénomènes scientifiques, offrant ainsi une plateforme pour explorer des concepts en sciences naturelles (Sengupta, 2013).

Par ailleurs, les environnements de programmation modernes qui s'inspirent de l'approche constructionniste, tels que Scratch ou le kit de robotique éducative, encouragent l'exploration, la collaboration et la création. Ces environnements sont conçus pour être intuitifs et accessibles, permettant aux étudiants de se concentrer sur la logique et la résolution de problèmes plutôt que sur la syntaxe et l'acquisition de connaissances (Maloney, 2010).

En somme, la théorie du constructionnisme offre un cadre théorique solide pour comprendre les avantages potentiels de l'apprentissage de la programmation, notamment en termes d'engagement cognitif et de transfert interdisciplinaire de compétences. Le constructionnisme introduit également l'idée que la programmation n'est pas seulement un objet d'étude en soi, mais également un outil pédagogique pouvant faciliter l'apprentissage dans d'autres domaines ou disciplines, grâce à la création de quelque chose de concret et de significatif.

Après avoir exploré la théorie du constructionnisme et son lien avec l'apprentissage de la programmation, il semble désormais important de mieux appréhender ce qui se concrétise durant le processus de

programmation et quelles compétences doivent être développées pour y parvenir. À cet égard, les travaux de Robins et de son équipe (2003) ont mené à l'élaboration d'un modèle d'apprentissage de la programmation qui se base sur des principes cognitifs fondamentaux. La prochaine section présente et discute de ce modèle.

2.1.2.2 Modèle de l'apprentissage de la programmation

Depuis les années 60, le champ de la didactique de la programmation s'est surtout développé par l'étude d'apprenants adultes universitaires. En effet, ce sont principalement les membres du corps professoral des facultés d'informatique qui, à la lumière de leur enseignement des cours d'introduction à la programmation (CS1) et de leur contact avec les apprenants, ont jeté les bases de ce champ de recherche.

Le premier ouvrage documentant l'apprentissage de la programmation a été publié au début des années 1970, *The Psychology of Computer Programming* par Weinberg (1971) qui explore le domaine de la programmation en tant qu'« activité humaine complexe ». Par la suite, une collection d'articles rassemblés dans le livre *Studying the novice programmer* par Soloway et Spohrer (1989) a apporté une contribution significative à la discipline en s'intéressant de manière très précise aux caractéristiques des apprenants novices. Cela a ouvert la voie à d'autres auteurs, tels que Hoc (1990), qui ont cherché à approfondir notre compréhension du processus complexe de l'apprentissage de la programmation, en tirant parti des travaux antérieurs. Enfin, sur la base de ces travaux fondateurs et d'autres, Robins et ses collaborateurs (2003) ont réalisé une revue de la littérature portant spécifiquement sur les apprenants novices en programmation. Cette revue de littérature les a menés à concevoir un modèle théorique de l'apprentissage de la programmation, qui fait l'objet du Tableau 2.2 présenté un peu plus loin.

Selon Robins et ses collègues (2003), l'apprentissage de la programmation est un processus complexe qui nécessite l'adoption d'une approche multidimensionnelle afin d'en faire l'étude. Le modèle qu'ils proposent met en lumière cet aspect via la présentation de trois types d'apprentissages clés en programmation :

- **Les connaissances** : Il s'agit de la base théorique, des principes fondamentaux et des informations déclaratives qu'un programmeur doit acquérir, tels que l'algèbre booléenne et le concept de boucle.

- **Les compétences** : Les compétences représentent la capacité à appliquer ces connaissances dans des situations pratiques, telles que l'utilisation efficace d'une boucle ou la gestion des erreurs (compétence de débogage).
- **Les modèles mentaux** : Ce sont les structures mentales ou les cadres de pensée que les programmeurs développent pour comprendre et conceptualiser les problèmes, ainsi que pour structurer leurs solutions.

Afin d'expliciter les différentes connaissances, compétences et modèles mentaux impliqués dans l'apprentissage de la programmation, Robins *et al.* (2003) déclinent en trois phases ce processus d'apprentissage. Ces phases constituent les étapes essentielles qui sont mises en place lors de la programmation :

- **La conception** : Au cours de cette phase, les apprenants planifient leur programme en décomposant le problème en étapes logiques. Ils identifient les objectifs, les algorithmes, et les structures de données nécessaires pour résoudre le problème de manière efficace. La phase de conception implique une réflexion approfondie sur la stratégie de résolution.
- **La génération** : Après la phase de conception, les apprenants passent à la phase de génération, où ils traduisent leurs plans en code informatique concret. Ils écrivent le programme en utilisant le langage de programmation choisi, en suivant les structures logiques qu'ils ont définies pendant la phase de conception. Cette phase nécessite une compréhension précise de la syntaxe du langage.
- **L'évaluation** : Une fois le code écrit, les apprenants passent à la phase d'évaluation. Ils testent le programme pour identifier d'éventuelles erreurs, bogues ou incohérences. Les tests visent à s'assurer que le programme fonctionne correctement et répond aux objectifs fixés lors de la phase de conception. Les apprenants peuvent également optimiser leur code pendant cette phase.

À travers ces trois phases, ce modèle met ainsi en évidence le processus itératif de l'apprentissage de la programmation, où les apprenants conçoivent d'abord leur solution, génèrent ensuite le code correspondant, puis évaluent et améliorent leur programme.

En combinant ces trois types d'apprentissages et ces phases de réalisation, la programmation est présentée comme une activité holistique qui va au-delà de la simple écriture de code. Elle exige de nombreux allers-retours entre l'acquisition de connaissances (savoir ce qu'est quelque chose) et le

développement des compétences pratiques pour les utiliser. Par exemple, pour le concept de boucle *for*, il est d'abord nécessaire que l'apprenant comprenne comment cette boucle fonctionne et ce qui la distingue des autres types de boucles, mais il est ensuite important de savoir quand et comment l'utiliser efficacement. Le tableau qui suit présente une synthèse des principales composantes de ce modèle, à savoir des exemples de contenus associés à chaque type d'apprentissage clé, pour chacune des phases.

Tableau 2.2 Modèle de l'apprentissage de la programmation proposé par Robins *et al.* (2003)

	Apprentissages clés en programmation		
Phases	**Connaissances**	**Compétences**	**Modèles mentaux**
Conception	Des méthodes de planification, design d'algorithmes, méthodes formelles	Pour planifier, résoudre des problèmes, concevoir des algorithmes	Du domaine du problème, machine notionnelle
Génération	Du langage, bibliothèques, environnement/outils	Pour implémenter des algorithmes, coder, accéder aux connaissances	Des programmes souhaités
Évaluation	Des outils et méthodes de débogage	Pour tester, déboguer, suivre/tracer, réparer	Du programme réel

Adapté de « Learning and teaching programming : A review and discussion » par Robins, 2003, *Computer Science Education, 13*(2), p. 164. Tel que le suggèrent les auteurs, ce tableau doit principalement être lu par colonnes, c'est-à-dire, la connaissance des méthodes de planification (nécessaire pour concevoir un programme), la connaissance du langage (nécessaire pour générer un programme), la connaissance des outils et méthodes de débogage (nécessaire pour évaluer un programme), et ainsi de suite.

De manière plus approfondie que ce tableau-synthèse, Robins et ses collègues identifient également un ensemble de connaissances fondamentales que tout apprenant novice en programmation devrait acquérir. Dans la lignée des concepts ciblés par Tew et Gudzial (2010), cet ensemble de connaissances comprend, entre autres, les algorithmes et le séquençage, les opérateurs logiques, la sélection d'états et les conditions, la boucle définie et indéfinie, les tableaux (*arrays*), les fonctions de contrôle, les fonctions de retour de valeurs, les variables, la récursivité et la programmation orientée sur l'objet (*object-oriented programming*). Ils identifient également plusieurs compétences pour chacune des trois phases de leur modèle. Pour la phase de conception, il est essentiel de posséder des compétences en résolution de problèmes, de savoir utiliser des modèles et des analogies, et de pouvoir évaluer les facteurs liés à la tâche ou au langage qui influencent la structure d'un programme approprié. La phase de génération du

programme exige, quant à elle, des compétences visant à mettre en œuvre la conception du programme, ce qui nécessite de mobiliser les connaissances nécessaires et de les appliquer de manière appropriée. Enfin, la phase d'évaluation du programme peut nécessiter des compétences pour suivre, tester et déboguer le code. Ce modèle souligne donc l'ampleur des connaissances et compétences requises pour devenir un programmeur compétent, tout en mettant en lumière l'importance de maîtriser ces aspects à différentes étapes du processus de programmation.

Par ailleurs, la notion de « modèles mentaux » est pour sa part plus rare dans la littérature en didactique de la programmation. Inspiré de la littérature en sciences cognitives (Gentner, 2002; Gentner et Stevens, 2014; Johnson-Laird, 1983), les modèles mentaux sont des représentations cognitives internes de la manière dont un programme informatique fonctionne. Ces modèles sont formés à partir de l'expérience et sont essentiels pour devenir un programmeur efficace et compétent. Ils soutiennent en effet le programmeur afin qu'il soit en mesure de conceptualiser le problème, élaborer une solution, traduire cette solution en code, prévoir son exécution et résoudre les problèmes qui surviennent en cours de route. Selon Robins (2003), le modèle mental central à l'apprentissage de la programmation est celui de la « machine notionnelle », une expression popularisée par Du Boulay (1986).

La notion de machine notionnelle représente une abstraction mentale utilisée par les programmeurs pour comprendre comment un programme informatique fonctionne. Il s'agit d'un modèle mental simplifié de la manière dont un ordinateur interprétera et exécutera les instructions d'un programme. Cette représentation conceptuelle de l'ordinateur, tant au niveau du logiciel que du matériel, aide à expliciter comment les programmes sont exécutés de manière séquentielle, logique et systématique. À cet égard, Du Boulay (1986) emploie une analogie pour illustrer la complexité associée au fait de comprendre un programme : il compare cela à décrypter le fonctionnement d'un moteur à partir d'une modélisation sous forme de dessin. Il serait également possible de comparer le modèle mental de la machine notionnelle à un plan ou une carte : de la même manière qu'une carte simplifie la compréhension d'un territoire complexe, la machine notionnelle simplifie la compréhension du fonctionnement d'un programme informatique. Il semble ainsi que sans le développement d'une vision claire de cette machine notionnelle, les apprenants novices peuvent élaborer des interprétations erronées et éprouver des difficultés importantes en programmation (Mayer, 1989). La machine notionnelle permettrait en effet de simplifier la compréhension et la résolution de problèmes, tout en aidant les programmeurs à développer d'autres modèles mentaux clairs. Robins (2003) identifient d'ailleurs plusieurs autres modèles mentaux tels que le

modèle mental de la solution, qui correspond à la représentation mentale de la solution au problème, c'est-à-dire le programme informatique lui-même. Ce modèle mental de la solution se développe en concevant l'algorithme, la structure des données et le code source pour résoudre le problème. Un autre exemple est le modèle mental du langage de programmation. Il s'agit de la manière dont le programmeur comprend et interprète le langage qu'il utilise. Cela inclut la syntaxe, les structures de contrôle, les fonctions, les bibliothèques, etc. Un autre exemple encore est le modèle mental de l'exécution qui correspond à la manière dont le programme s'exécutera. Il permet de prévoir comment les instructions seront traitées par l'ordinateur et comment les données seront manipulées.

Qui plus est, le modèle théorique de Robins (2003) présente les connaissances comme une nécessité évidente pour l'apprentissage de la programmation. Toutefois, au terme de leur revue de la littérature, ils soutiennent que de nombreux manuels d'introduction et cours de programmation sont presque exclusivement axés sur l'acquisition des connaissances, principalement celles associées à la phase de génération (rédaction du code). Or, bien que certains auteurs soulignent que ces connaissances doivent faire l'objet d'une attention particulière auprès des apprenants novices parce qu'elles sont souvent « fragiles », trop rapidement « oubliées », ou encore « utilisées de manière inappropriée » (Perkins et Martin, 1986), Robins et ses collègues (2003) mettent pour leur part de l'avant qu'il importe d'aller au-delà de l'acquisition de celles-ci afin que les apprenants réalisent des apprentissages leur permettant de réellement concrétiser la programmation. Il semble d'ailleurs que les aspects les plus souvent négligés par le champ de la didactique de la programmation soient le développement de compétences et la construction des modèles mentaux.

En somme, le modèle théorique proposé par Robins et des collaborateurs met de l'avant l'idée que l'apprentissage de la programmation fait interagir plusieurs composantes. Il ne se résume pas à l'acquisition unique de connaissances ou encore à la maîtrise d'un langage « codé ». Il nécessite aussi le développement de compétences et même la construction de modèles mentaux plus larges qui permettent de simplifier et d'organiser la compréhension d'un programme informatique. Ce modèle théorique offre ainsi une vision plus approfondie de ce que signifie réellement d'apprendre à programmer. Il propose également une décomposition détaillée des éléments impliqués dans l'apprentissage de la programmation. Cette décomposition permet, par extension, d'identifier les éléments qui pourraient être réutilisés dans le cadre d'autres apprentissages. La prochaine section explore d'ailleurs le concept de la pensée

informatique, qui apparaît central à l'idée du transfert des apprentissages de la programmation vers d'autres domaines.

2.1.3 Pensée informatique

L'expression « pensée informatique » semble avoir été utilisée pour la première fois dans un livre de Seymour Papert (*Mindstorms : Children, Computers, and Powerful Ideas*) en 1980, où il présentait la pensée informatique comme la relation entre l'apprentissage de la programmation et ses effets potentiels sur la capacité de réflexion des apprenants. Dans la lignée de sa théorie du constructionnisme, Papert était en effet convaincu que, grâce à la programmation, les élèves pouvaient développer de façon générale une pensée plus logique et procédurale, semblable au fonctionnement d'un ordinateur, et que cette pensée serait applicable et bénéfique à l'apprentissage de multiples disciplines. Ces dernières années, un nombre croissant de systèmes éducatifs ont intégré la notion de pensée informatique dans leur programme d'enseignement obligatoire, le plus souvent en relation avec l'apprentissage de la programmation (Heintz et Mannila, 2018; Mannila *et al.*, 2014).

De l'avis de plusieurs auteurs, apprendre à programmer favoriserait donc le développement de la pensée informatique chez les apprenants (Grover and Pea, 2013; Lye et Koh, 2014). Certains considèrent d'ailleurs la pensée informatique comme un terme parapluie englobant les bienfaits collatéraux généraux associés à l'apprentissage de la programmation (Nouri *et al.*, 2020). Plusieurs études ont d'ailleurs permis de confirmer les effets de l'apprentissage de la programmation sur le développement de la pensée informatique, dont la méta-analyse de Zhang et Nouri (2019) qui a mis en évidence que l'utilisation de Scratch pour apprendre la programmation contribuait au développement de la pensée informatique chez des élèves de 6 à 15 ans (K-9).

Dans les années 2000, c'est Wing (2006) qui a porté l'expression « pensée informatique » (*Computational Thinking*) à l'attention de la communauté informatique et popularisé le terme dans son article fondateur où elle décrivait la pensée informatique comme le fait de « penser comme un programmeur » (cet article compte actuellement plus de 6500 citations sur Google Scholar). À partir de 2006, un corpus considérable de littérature a été produit afin de parvenir à une meilleure définition de ce concept et de fournir des outils et des cadres pour introduire et évaluer la pensée informatique dans l'enseignement primaire et secondaire (Lodi, 2020). En 2011, Wing a actualisé sa première définition en une définition plus formelle de la pensée informatique comme étant « les processus de pensée impliqués dans la formulation des

problèmes et de leurs solutions, de sorte que les solutions soient représentées sous une forme qui peut être efficacement mise en œuvre par un agent de traitement de l'information » (traduction libre, p. 1). Néanmoins, il est important de souligner que la notion de pensée informatique n'est pas définie de manière univoque et consensuelle (Weintrop, 2016). En effet, de nombreux chercheurs ont élaboré leur propre définition, reflétant ainsi la richesse et la complexité de ce concept. Plusieurs synthèses des définitions existantes ont également fait l'objet d'articles publiés (voir par exemple Corradini, 2017 et Juškevičienė et Dagienė, 2018). Dans sa revue de littérature récente, Lodi (2020), en recense et en analyse près d'une vingtaine.

Parmi les définitions existantes, certains soutiennent que la pensée informatique correspond à : « une activité associée à la résolution de problèmes, sans y être limitée. Il s'agit d'un processus cognitif ou de réflexion qui reflète la capacité à penser en abstractions, à penser en termes de décomposition, à penser algorithmiquement, à penser en termes d'évaluations, et à penser en généralisations » (Selby et Woolard, traduction libre, p. 5) ou encore à « utiliser des schémas et des processus de réflexion pour poser et résoudre des problèmes ou préparer des programmes » (Krauss et Prottsman, 2016, traduction libre, p. 4). D'autres soulignent que la pensée informatique renverrait aux « compétences de réflexion d'ordre supérieur qui peuvent nécessiter d'utiliser la puissance des capacités cognitives humaines et d'accepter le soutien des machines pour penser et résoudre les problèmes » (Kalelioğlu, 2016, traduction libre, p. 593) ou à « la base conceptuelle nécessaire pour résoudre des problèmes de manière efficace et efficiente (c'est-à-dire de manière algorithmique, avec ou sans l'aide d'ordinateurs) avec des solutions réutilisables dans différents contextes » (Shute, 2017, traduction libre, p 151).

À la lumière de ces définitions, il apparaît clair que la résolution de problèmes occupe une place de premier plan au sein de ce concept de pensée informatique. Les individus qui pensent de manière informatique seraient en effet capables d'analyser un problème, d'identifier les informations pertinentes, de concevoir une solution et de la mettre en œuvre. Plusieurs auteurs sont même d'avis que la capacité à résoudre des problèmes, avec ou sans l'aide de la technologie, serait ce qui représente le mieux le concept de pensée informatique. À cet égard, Lodi (2020) avance d'ailleurs qu'il serait selon lui plus juste de parler d'un « processus de résolution de problèmes informatique » afin de mettre davantage l'accent sur cet aspect, étant donné qu'il semble s'agir de celui qui fait le plus consensus.

Par ailleurs, en plus de ces définitions, plusieurs auteurs ont également souhaité mieux caractériser la pensée informatique en la morcelant en diverses composantes, les plus fréquemment évoquées étant l'abstraction, les algorithmes, les données, la décomposition du problème, le parallélisme, le débogage et les tests (Rose, Habgood, et Jay, 2017). Dans sa revue de la littérature, Lodi (2020) identifie ainsi les différentes composantes de la pensée informatique présentes dans la littérature scientifique et classe ces dernières en quatre catégories : 1- les processus mentaux mobilisés pour résoudre des problèmes, 2- les méthodes utilisées par les programmeurs, 3- les pratiques typiques utilisées dans la mise en œuvre de solutions basées sur des machines informatiques et 4- les compétences transversales. Le Tableau 2.3 présente une synthèse des composantes de la pensée informatique recensées par Lodi (2020). Afin de donner une idée de l'importance relative de chacune de ces composantes, celles-ci ont ici été ordonnancées par fréquence d'occurrence dans la littérature (tel que rapporté dans Lodi, 2020), au sein de chaque catégorie.

Tableau 2.3 Synthèse des composantes de la pensée informatique identifiées par Lodi (2020)

Catégories	**Composantes de la pensée informatique**
Processus mentaux	• Pensée algorithmique • Abstraction • Décomposition de problèmes • Reconnaissance de schémas/tendances • Généralisation • Pensée logique
Méthodes	• Modélisation et simulation • Automatisation • Collecte, analyse et représentation des données • Analyse et évaluation • Parallélisation • Programmation
Pratiques	• Testage et débogage • Expérimentation, itération et bricolage • Réutilisation et remix
Compétences transversales	• Conception et création • Réflexion, apprentissage, méta-réflexion, compréhension du monde de manière informatique • Persévérance dans le traitement de problèmes complexes • Communication et collaboration • Tolérance à l'ambiguïté

Adapté de « Informatical Thinking » par Lodi, 2020, *Olympiads in Informatics, 14,* p. 121-123

Cette synthèse met en évidence que plusieurs composantes de la pensée informatique sont identifiées et discutées dans la littérature. Incluse dans la catégorie des processus mentaux, la pensée algorithmique apparaît parmi les plus centrales. Les algorithmes sont des séquences d'étapes clairement définies pour résoudre un problème. La pensée informatique consisterait donc notamment à développer des algorithmes pour résoudre des tâches, que ce soit pour trier des données, effectuer des calculs mathématiques complexes ou automatiser des processus (Barr et Stephenson, 2011). L'abstraction constitue un autre exemple d'une composante largement discutée et qui semble au cœur de la pensée informatique. Elle consiste à se débarrasser des détails inutiles pour se concentrer sur les informations ou les idées les plus importantes. Toujours dans la catégorie des processus mentaux, la décomposition de problèmes permet pour sa part de simplifier un problème en le décomposant en sous-problèmes plus facilement gérables. Cette compétence est souvent utilisée pour concevoir des algorithmes et des programmes informatiques. Par exemple, lors de la création d'une application mobile, un développeur doit abstraire la logique de l'application en différents éléments, tels que l'interface utilisateur, la gestion des données et les fonctions de traitement (Grover et Pea, 2013). Dans la catégorie des méthodes, on retrouve notamment la modélisation, qui renvoie à la capacité à représenter des données et des processus sous la forme de modèles. Dans la catégorie des pratiques, la composante du testage/débogage, par exemple, représente la capacité à vérifier qu'une solution fonctionne réellement en la testant et en réglant les problèmes (bogues) qui surviennent, le cas échéant. Finalement, la catégorie des compétences transversales regroupe plusieurs composantes qui renvoient à des manières plus générales de voir et d'agir dans le monde et qui seraient favorisées par le fait de penser comme des programmeurs, notamment la capacité à concevoir et construire des objets et des artefacts informatiques, ainsi qu'à utiliser l'informatique pour être créatif et s'exprimer. Cette synthèse proposée par Lodi (2020) fournit donc une vue d'ensemble des principaux éléments discutés dans la littérature qui composent la pensée informatique, et de l'importance relative de chacun de ceux-ci.

Enfin, Brennan et Resnick (2012) proposent également un modèle de la pensée informatique, qui est à ce jour le plus cité et le plus utilisé dans les écrits en didactique de la programmation. Ce modèle examine comment la pensée informatique se développe chez un apprenant novice par l'apprentissage de la programmation. Plusieurs des éléments de ce modèle rejoignent ceux évoqués dans le modèle de l'apprentissage de la programmation développé par Robins (2003) dont il était question un peu plus tôt. Brennan et Resnick décrivent pour leur part la pensée informatique à travers trois grandes composantes : 1- les concepts informatiques, 2- les pratiques informatiques et 3- les perspectives informatiques.

Les concepts informatiques réfèrent aux idées de base que les apprenants manipulent lorsqu'ils programment. Ces concepts ressemblent beaucoup à ceux identifiés par Robins (2003), soit :

- Suivre une séquence d'instructions
- Utiliser des boucles pour répéter des actions
- Faire plusieurs choses à la fois (parallélisme)
- Réagir à des événements ou des déclencheurs
- Prendre des décisions basées sur des conditions
- Utiliser des opérateurs pour effectuer des calculs
- Gérer et manipuler des données

Les pratiques informatiques représentent quant à elles les méthodes que les concepteurs développent en programmant. Ces pratiques présentent également plusieurs similarités avec les compétences qui avaient été identifiées par Robins et ses collègues (2003) :

- Travailler étape par étape et réaliser une amélioration progressive
- Vérifier et corriger les erreurs
- Utiliser des éléments existants et les adapter à de nouveaux besoins
- Simplifier les problèmes en les décomposant en parties gérables

Enfin, les perspectives informatiques renvoient de manière plus générale aux façons dont les concepteurs appréhendent le monde autour d'eux, à travers le prisme de la pensée informatique. Elles comprennent :

- Exprimer des idées à travers la programmation.
- Se connecter avec d'autres et partager des idées.
- Poser des questions et être curieux sur le fonctionnement des choses.

Dans les modèles présentés, qu'il s'agisse de celui de Lodi (2020) émergeant de sa revue de littérature ou de celui de Brennan et Resnick (2012), la programmation et la pensée informatique apparaissent profondément interconnectées. En effet, dans le modèle de Brennan et Resnick (2012), les concepts et pratiques informatiques sont intrinsèquement liés aux apprentissages en programmation. En apprenant à programmer, les apprenants acquièrent donc non seulement des savoirs et compétences nichés en programmation, mais développent également une manière structurée et logique de penser associée à des compétences d'ordre plus général : la pensée informatique. C'est d'ailleurs cette pensée informatique qui, de l'avis de plusieurs chercheurs, pourrait être mobilisée pour faciliter les apprentissages au sein d'autres disciplines scolaires (Shute, 2017; Weintrop, 2016).

Certains auteurs proposent également des modèles théoriques permettant d'établir des liens entre la programmation, la pensée informatique et les autres disciplines scolaires. Parmi ces modèles, on retrouve notamment celui de Weintrop (2016) ainsi que Barr et Stephenson (2011). Celui de Barr et Stephenson (2011) apparaît particulièrement intéressant dans le contexte de cette thèse, puisqu'il va au-delà des liens qui sont habituellement tissés entre la programmation, les mathématiques et les sciences, en incluant également les domaines des langues et des sciences sociales. Qui plus est, ce modèle a été élaboré dans le cadre d'une initiative chapeautée par deux organisations américaines : le Computer Science Teachers Association (CSTA) et l'International Society for Technology in Education (ISTE), qui a mobilisé la participation conjointe de chercheurs et d'éducateurs. Cette initiative visait à opérationnaliser comment la pensée informatique pouvait être intégrée au programme d'études de la maternelle à la 12^{e} année (*K-12 curriculum*). Ce projet a ainsi mené à l'identification de liens interdisciplinaires entre les principales composantes de la pensée informatique et différentes disciplines scolaires, principalement via la résolution de problèmes et les concepts de bases en programmation.

Ces relations sont présentées au Tableau 2.4. À titre d'exemple, la capacité d'abstraction pourrait être mobilisée en mathématiques, notamment lors de l'utilisation de variables en algèbre, ou encore dans le domaine des langues, pour employer des figures de style comme la métaphore. La composante de la décomposition de problèmes pourrait quant à elle être mobilisée en sciences pour soutenir la classification des espèces, ou en langues, dans le contexte de la rédaction du plan d'un texte. La composante de l'analyse des données pourrait également être déployée en sciences sociales pour identifier des tendances à partir de statistiques ou encore en langues afin d'identifier des modèles pour différents types de phrases.

Tableau 2.4 Synthèse des composantes de la pensée informatique pouvant être mobilisées dans différentes disciplines telles qu'identifiées par Barr et Stephenson (2011)

Composantes de la pensée informatique	Programmation	Autres disciplines scolaires			
		Mathématiques	**Sciences**	**Sciences sociales**	**Langues**
Algorithmes et procédures	Étudier des algorithmes classiques; implémenter un algorithme	Effectuer une division longue, une factorisation; gérer les retenues lors de l'addition ou de la soustraction	Réaliser une procédure expérimentale	---	Rédiger des instructions
Abstraction	Utiliser des procédures pour un ensemble de commandes fréquemment répétées (p. ex., des boucles)	Utiliser des variables dans l'algèbre; étudier les fonctions essentielles d'un problème par rapport aux fonctions en programmation	Construire un modèle d'une entité physique	Résumer des faits; déduire des conclusions des faits	Utiliser des figures de style, comme la métaphore; écrire une histoire avec des ramifications
Décomposition de problèmes	Définir des objets et des méthodes; définir des fonctions principales	Appliquer l'ordre des opérations dans une expression	Effectuer une classification d'espèces	---	Rédiger un plan
Simulation	Animation d'algorithmes, balayage de paramètres	Tracer une fonction dans un plan cartésien et modifier les valeurs des variables	Simuler le mouvement du système solaire	Participer à des jeux de société ou vidéo à thème historique	Effectuer une reconstitution à partir d'une histoire

Automatisation	---	Utiliser des outils permettant de tracer des figures géométriques ou des extraits de code Python	Utiliser du matériel de laboratoire	Utiliser un tableur comme Excel	Utiliser un vérificateur d'orthographe
Parallélisation	Effectuer un traitement parallèle, diviser une tâche pour qu'elle soit traitée en parallèle	Résoudre des systèmes linéaires, multiplication de matrices	Exécuter simultanément des expériences avec différents paramètres	---	---
Collecte des données	Trouver une source de données pour un domaine particulier	Trouver une source de données pour un domaine particulier (p. ex., en lançant des dés)	Collecter des données à partir d'une expérience	Étudier les statistiques ou les données démographiques	Réaliser une analyse linguistique des phrases
Analyse des données	Écrire un programme pour effectuer des calculs statistiques de base sur un ensemble de données	Compter les occurrences de lancers de dés et analyser les résultats	Analyser les données d'une expérience	Identifier des tendances à partir de statistiques	Identifier des modèles pour différents types de phrases
Représentation des données	Utiliser des structures de données, comme un tableau, une liste, un graphique	Utiliser un histogramme ou un diagramme circulaire pour représenter des données; utiliser des ensembles, des listes ou des graphiques pour contenir des données	Résumer les données d'une expérience	Résumer et représenter des tendances	Représenter des modèles de différents types de phrases

Adapté de « Bringing computational thinking to K-12 : What is involved and what is the role of the computer science education community?» par Barr et Stephenson, 2011, *ACM Inroads, 2(1),* p.52

Selon Barr et Stephenson (2011), un lien organique émerge ainsi entre la programmation, la pensée informatique et les disciplines scolaires plus traditionnelles. La programmation peut donc servir d'outil pour développer différentes composantes de la pensée informatique, qui peuvent être réinvesties afin de réaliser des apprentissages dans d'autres disciplines. Le modèle de Barr et Stephenson permet donc déjà de réfléchir de manière théorique aux relations possibles entre la programmation et les autres disciplines scolaires, à travers le développement de la pensée informatique. Il ouvre ainsi la porte à une réflexion plus poussée sur les effets de transfert pouvant découler de l'apprentissage de la programmation. Cette relation envisagée entre l'apprentissage de la programmation, le développement de la pensée informatique et l'apprentissage d'autres disciplines scolaires est illustrée à la Figure 2.2 ci-bas. La flèche pointillée représente l'effet de transfert potentiel entre l'apprentissage de la programmation et les autres disciplines. L'apprentissage de la programmation favoriserait ainsi le développement de la pensée informatique, un ensemble de compétences cognitives générales telles que la résolution de problèmes, la décomposition de tâches complexes, et l'abstraction, qui pourraient être réutilisées pour réaliser des apprentissages dans d'autres disciplines scolaires. La pensée informatique agirait donc comme une variable médiatrice entre l'apprentissage de la programmation et l'apprentissage d'autres disciplines scolaires.

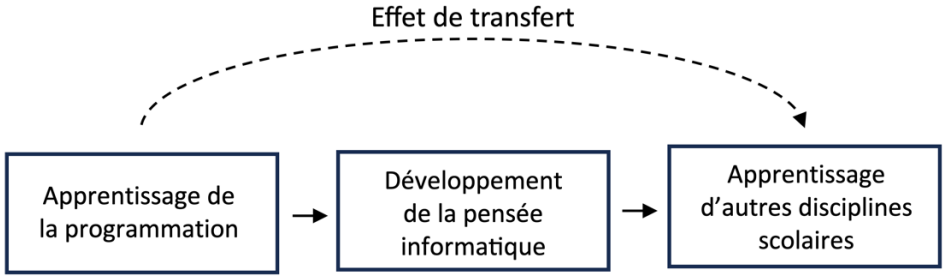

Figure 2.2 Représentation de l'effet de transfert potentiel de l'apprentissage de la programmation vers l'apprentissage d'autres disciplines scolaires, à travers le développement de la pensée informatique.

Qui plus est, le Tableau 2.4 synthétisant les composantes de la pensée informatique susceptibles d'être mobilisées par différentes disciplines, laisse aussi entrevoir que certaines disciplines apparaissent plus « proches » de la programmation, telles les mathématiques et dans une moindre mesure les sciences, et que d'autres apparaissent possiblement plus « éloignées », comme l'apprentissage d'une langue. Cette notion de « distance » entre les disciplines est un aspect souvent discuté quand il est question du transfert des apprentissages. La section qui suit aborde donc le deuxième concept central à ce projet, soit celui du transfert des apprentissages.

2.2 Transfert des apprentissages

Le concept de transfert des apprentissages apparaît central à ce projet de recherche qui vise à déterminer si l'apprentissage de la programmation est susceptible de faciliter les apprentissages pour d'autres disciplines scolaires. Il convient de noter que le transfert des apprentissages est un sujet qui a été profondément exploré au sein d'une littérature de recherche vaste et établie, ayant fait l'objet d'études depuis de nombreuses années. Cette section se consacrera tout d'abord à l'exploration et à la définition de ce concept, afin d'en assurer une compréhension claire et précise.

2.2.1 Définitions et survol historique de l'évolution du concept de transfert des apprentissages

L'objectif fondamental de tout apprentissage est qu'il puisse se manifester dans une gamme diversifiée de situations (Hattie, 2016; Richard et Bissonnette, 2012, Scherer *et al.*, 2019). Dans cette optique, la littérature propose de nombreuses définitions du transfert des apprentissages. En éducation, Legendre (2005) le définit comme : « l'usage fait de connaissances acquises dans une situation nouvelle. Influence, impact sur un apprentissage subséquent » (p. 1402). Cette définition met de l'avant une idée centrale au transfert, soit celle de réutiliser des connaissances dans une autre situation que celle dans laquelle elles ont été initialement acquises. Bransford (2000) reprend cette idée et le définit simplement comme « la capacité d'étendre à de nouveaux contextes ce qui a été appris dans un contexte donné » (2000, p. 51). De façon tout à fait convergente, l'Office québécois de la langue française (2023, en ligne) propose une définition lexicographique du transfert des apprentissages qui serait un : « mécanisme qui permet à une personne de faire appel, dans une situation nouvelle, à des savoirs ou des habiletés acquis dans une autre classe de situations ». Elle mentionne également que ce transfert : « […] permet, d'une part, d'étendre un apprentissage à des domaines différents et plus larges, et, d'autre part, d'accéder à *des niveaux plus élevés de compétence* », ce qui souligne également que le transfert des apprentissages peut entraîner un changement dans la performance, générant ainsi potentiellement un avantage pour les apprentissages résultant du transfert. Dans la même veine, pour Perkins et Salomon (1992), le transfert des apprentissages « survient lorsque l'apprentissage qui est réalisé dans un contexte précis ou avec du matériel particulier *influence la performance* dans un autre contexte ou avec d'autre matériel » (traduction libre, p.6454). Cette définition ne suggère toutefois pas que l'influence soit forcément de nature à faciliter ou améliorer la performance. De façon encore plus explicite, Péladeau et ses collaborateurs (2006) définissent le transfert comme : « toute influence, *positive ou négative*, que peut avoir l'apprentissage ou la pratique d'une tâche sur les apprentissages ou les performances subséquentes » (p.17). De l'avis de ces

auteurs, cette définition se veut consensuelle, assemblant ainsi les diverses formulations proposées par de nombreux auteurs, quelle que soit leur orientation théorique. Cette perspective ouvre la voie à la compréhension des effets positifs et négatifs du transfert.

S'inscrivant dans une perspective assez classique du transfert, ces définitions apparaissent ainsi très semblables : elles convergent toutes vers l'idée que le transfert des apprentissages implique l'application de connaissances, de compétences ou d'attitudes acquises dans un contexte donné à une situation nouvelle.

L'origine du concept de transfert des apprentissages remonte aux toutes premières années du 20e siècle. La notion d'effet d'expansion (*expansion effect*) est alors employée pour décrire l'idée de transfert (Thorndike et Woodworth, 1901). Il est suggéré que le transfert se produit lorsque l'apprentissage d'une tâche ou d'une compétence dans un contexte donné influence positivement la performance dans une tâche similaire. L'une des idées fondamentales de Thorndike qui a contribué à la compréhension du transfert était sa conviction que l'apprentissage d'une matière ou d'une compétence pouvait influencer positivement la performance dans des domaines apparentés. Plus spécifiquement, il était d'avis qu'apprendre le latin pouvait avoir un effet d'expansion sur l'apprentissage d'autres langues ou de sujets connexes. Il suggérait en effet que les compétences cognitives développées lors de l'apprentissage du latin, telles que la compréhension de la structure grammaticale et la résolution de problèmes linguistiques (souvent rassemblées sous l'idée de la pensée rationnelle), pouvaient être transférées et appliquées avec succès à l'apprentissage d'autres langues, comme l'anglais, ou de disciplines exigeant une réflexion similaire, telles que les mathématiques. Il ne parvint toutefois pas à démontrer cette hypothèse, mais avança que pour que l'effet d'expansion puisse se produire, les éléments de la situation initiale et de la nouvelle situation d'apprentissage doivent être identiques (Cox, 1997). S'inscrivant dans le courant behavioriste de la psychologie, ces idées ont également donné naissance à ce que l'on connaît aujourd'hui sous le nom de « lois du transfert ». Ces lois énonçaient que le transfert dépendait de la similitude entre les situations d'apprentissage et de transfert. En d'autres termes, plus les deux situations étaient semblables, plus le transfert était probable. Cette idée de similitude a jeté les bases de nombreuses recherches ultérieures sur le transfert des apprentissages et a ouvert la voie à une famille de théories sur le transfert des apprentissages s'articulant autour de l'idée des éléments communs (*common element theories*, Loboto, 2006) qui a dominé les recherches sur la question au 20e siècle (Cox, 1997). Les travaux de Thorndike ont ainsi mené d'autres chercheurs à explorer en détail les facteurs qui influencent le

transfert, notamment la nature des compétences acquises, le contexte de l'apprentissage, et la manière dont les connaissances sont appliquées dans de nouvelles situations.

Vers les années 1910-1920, la psychologie de la Gestalt a aussi influencé l'évolution du concept de transfert des apprentissages auquel était alors associé le terme de « transposition » (Cyr, 2012). La transposition est un concept qui s'apparente au transfert dans le sens où il implique le déplacement ou la réutilisation de connaissances ou de compétences d'une situation à une autre. La psychologie de la Gestalt a exploré la manière dont les individus perçoivent et comprennent le monde en organisant les informations en « formes » ou « figures » perceptuelles. Le terme transposition tel qu'il est compris dans la théorie de la Gestalt concerne principalement la capacité de percevoir des structures ou des formes similaires dans des contextes différents. Un exemple concret de transposition dans cette perspective peut être illustré par l'illusion d'optique du « triangle de Penrose » (Desolneux, 2008). Le triangle de Penrose est un dessin tridimensionnel qui crée l'illusion d'un objet impossible, un triangle qui semble se connecter en boucle, formant un objet sans fin qu'il serait impossible de construire dans un espace tridimensionnel réel. Le cerveau perçoit néanmoins la continuité du triangle, bien que les angles et les connexions ne correspondent pas à la réalité. Cette illusion illustre la capacité du cerveau à transposer des relations spatiales d'un contexte à un autre, même lorsque ces relations ne sont pas physiquement réalisables. La transposition dans le cadre de la Gestalt est donc liée à la manière dont les individus peuvent réorganiser leurs perceptions et leurs compréhensions pour les adapter à de nouvelles situations, ce qui est similaire à l'idée de transfert des apprentissages. Qui plus est, les principes de proximité et de similitude, également mis de l'avant par les psychologues de la Gestalt, soulignent l'importance de la ressemblance entre les éléments d'apprentissage et de transfert pour faciliter le transfert des connaissances et des compétences.

Vers 1930-1940, le psychologue behavioriste Skinner a également joué un rôle prépondérant dans l'étude du transfert des apprentissages en introduisant le concept de « généralisation » au domaine de l'apprentissage (Cyr, 2012). Il a montré que les organismes apprennent des comportements dans un contexte donné et qu'ils ont tendance à généraliser ces comportements à des situations similaires. Cela a jeté les bases de la compréhension du transfert comme une forme de généralisation. Skinner est célèbre pour ses expériences sur l'apprentissage, en particulier l'apprentissage par conditionnement opérant. L'une de ses expériences les plus connues qui illustre la généralisation et a des implications pour l'idée du transfert est celle de la boîte de Skinner, également appelée la « boîte de conditionnement opérant ». Dans cette expérience, des pigeons étaient utilisés comme sujets d'étude (voir p. ex., Morse et Skinner,

1958). Ceux-ci étaient placés dans une boîte spéciale où ils pouvaient effectuer une action (appuyer sur un levier) pour recevoir une récompense, généralement de la nourriture. L'expérience a révélé que les pigeons apprenaient rapidement à appuyer sur le levier pour obtenir de la nourriture, ce qui est un exemple de conditionnement opérant. Cependant, l'aspect de généralisation est devenu évident lorsque Skinner a modifié l'expérience. Il a introduit des variations dans la couleur, la forme ou la taille du levier, et les pigeons ont continué à appuyer sur ces leviers modifiés pour obtenir de la nourriture, même s'ils n'avaient jamais rencontré ces variations auparavant. Cela illustre le phénomène de généralisation, où les pigeons ont généralisé (c'est-à-dire transféré) leur apprentissage initial de l'action (appuyer sur le levier) à des situations légèrement différentes. Dans ce cas précis, le transfert se manifeste sous la forme d'une généralisation des réponses apprises à des stimuli similaires. Bien que Skinner n'ait pas directement utilisé le terme « transfert » pour qualifier le résultat de cette expérience, cela illustre le concept de généralisation des apprentissages d'une situation à une autre, une idée qui est essentielle dans la compréhension du transfert des apprentissages. Cette expérience a ainsi contribué à éclairer la manière dont les compétences et les comportements peuvent être transférés à des situations similaires, mais différentes, un aspect clé de l'étude du transfert des apprentissages.

L'ensemble de ces courants de recherche insistent donc sur le fait que les composantes de la situation d'apprentissage initiale doivent présenter une forte ressemblance avec celles de la situation dans laquelle se produit le transfert afin que celui-ci puisse avoir lieu. Le concept de similarité apparaît de ce fait comme une dimension incontournable du transfert des apprentissages. Certains chercheurs avancent même que sans ce concept, il n'y aurait pas de théorie du transfert (Butterfield et Belson, 1989). Les courants de recherche qui viennent d'être discutés ont eu tendance à accorder une grande importance à la similarité dite « de surface » (Bracke, 1998) qui fait référence à des caractéristiques externes ou apparentes des tâches ou des situations d'apprentissage. Cela concerne des éléments tels que la présentation visuelle, le contexte ou le format, qui peuvent sembler similaires ou différents entre la situation initiale et la situation de transfert. Le postulat partagé entre ces courants était que des tâches ou des situations d'apprentissage similaires en apparence favoriseraient le transfert. Cette perspective était largement influencée par les théories du courant behavioriste qui se concentraient sur les caractéristiques extérieures des stimuli. Il semble par ailleurs que la question de la définition de ce qu'est la similarité entre deux situations d'apprentissage ou entre deux tâches demeure d'actualité (Fleishman, 2014).

Cette notion de similarité est toujours présente dans les théories plus contemporaines du transfert (Bracke, 1998; Tardif, 1999). Différents types de similarité ont été proposés par différents auteurs; les situations d'apprentissage pouvant en effet être similaires (ou, à l'inverse, différentes) à plusieurs égards (pour une synthèse, voir Bracke, 1998). Il est notamment possible d'évaluer cette similarité au regard de caractéristiques inhérentes aux situations elles-mêmes, indépendamment des apprenants, telles que leur degré de complexité, les exigences en matière de précision de la réponse à fournir, leur objectif, les conditions expérimentales au sein desquelles elles se déroulent, etc. Il est aussi possible d'évaluer leur similarité au regard des habiletés qu'elles nécessitent de la part de l'apprenant qui les réalise, par exemple la compréhension verbale, la coordination, ou encore des compétences qu'elles mobilisent telles que la résolution de problème, la créativité, etc. (Fleishman, 2014; Lobato, 2006). Encore aujourd'hui, ces ambiguïtés concernant la nature de la similarité contribuent largement à la variabilité observée dans la mesure et l'évaluation du transfert (Bracke, 1998).

Dans les années 1960, le courant cognitiviste a pris de l'ampleur et a lui aussi apporté une contribution particulière à l'étude du transfert des apprentissages. En réaction au courant behavioriste de l'époque, lequel repose sur l'idée que le transfert est largement dicté par des stimuli externes (Haskell, 2001), le cognitivisme a privilégié l'exploration des mécanismes sous-jacents à la manière dont les individus acquièrent de nouvelles compétences et connaissances et les appliquent dans des contextes variés; l'attention s'est donc recentrée sur les processus cognitifs mobilisés par une tâche ou une situation (Cyr, 2012). Plusieurs auteurs ont ainsi présenté le transfert des apprentissages comme étant en étroite relation avec le raisonnement analogique : pour que le transfert ait lieu, il est nécessaire que les représentations mentales des connaissances ou compétences acquises dans un contexte puissent se superposer ou présenter des analogies avec celles requises dans une nouvelle situation (Day et Goldstone, 2012). En d'autres termes, le transfert est favorisé lorsque les éléments structurels des tâches ou des situations d'apprentissage sont similaires, indépendamment des ressemblances superficielles. Cependant, ce type de transfert plus profond nécessite que l'individu détienne une expertise suffisante pour mobiliser les processus cognitifs nécessaires et développer des représentations mentales appropriées à la résolution des deux situations d'apprentissage (Bracke, 1998; Cyr, 2012). Par conséquent, l'étude des dimensions cognitives du transfert a conduit à une réorientation de la notion de similarité, mettant davantage l'accent sur la similarité dite « de structure » (Bracke, 1998), c'est-à-dire qui repose sur les structures internes aux apprentissages plutôt que sur des caractéristiques de surface. Cette similarité structurelle considère notamment comment les connaissances, les compétences ou les principes sous-jacents sont organisés et

interconnectés et va au-delà de l'apparence externe pour examiner les similitudes dans les schémas, les relations et les concepts essentiels impliqués dans les situations d'apprentissage. En d'autres termes, elle porte sur la manière dont l'information est organisée et traitée en profondeur, indépendamment des aspects superficiels.

Plusieurs auteurs clés ont exploré cette perspective cognitive du transfert et ont mis en évidence des principes importants. Par exemple, Ausubel, a développé la théorie de l'apprentissage significatif (*Meaningful Learning Theory*, 1963), qui insiste sur la connexion des nouvelles informations à des connaissances préexistantes. Cette connexion renforce le transfert en permettant aux apprenants de relier leurs nouvelles compétences ou connaissances à celles déjà acquises, créant ainsi un « pont cognitif » pour le transfert. Les travaux d'Ausubel soulignent l'importance de l'ancrage des nouvelles informations dans un contexte cognitif préexistant pour maximiser le transfert. Cette connexion avec des connaissances antérieures est cruciale, car elle permet aux apprenants de relier leurs nouvelles compétences ou connaissances à celles déjà acquises.

Bransford (2000) a pour sa part étudié les mécanismes de transfert en examinant comment les individus organisent et restructurent leurs connaissances pour les appliquer dans de nouvelles situations. Ses travaux ont mis en évidence l'importance de la réorganisation cognitive dans le transfert des compétences, en mettant l'accent sur la compréhension conceptuelle comme facteur clé. D'autres encore ont cherché à identifier les opérations cognitives sous-jacentes au transfert des apprentissages (p. ex., Tardif, 1992; Bastien, 1997).

À cet égard, Presseau (2000) souligne le rôle du processus d'encodage dans le transfert : « Les connaissances encodées, emmagasinées et organisées servent en quelque sorte de base au transfert puisque ce sont celles-ci qui, ultérieurement, seront rappelées et activées lors de l'accomplissement d'une nouvelle tâche, en l'occurrence la tâche cible […] en fonction de la similarité perçue entre tâche source et tâche cible […] » (p. 518). En d'autres termes, lorsque l'apprenant reconnaît clairement les ressemblances entre les deux tâches, il peut réutiliser les connaissances acquises dans la tâche initiale tout en les adaptant à la nouvelle tâche.

Toujours d'un point de vue cognitif, le transfert des apprentissages peut également être appréhendé comme un « processus économique et adaptatif » (Bracke, 1998, p.238). En effet, en raison des limites du système cognitif humain et du nombre considérable de possibilités qui peuvent être envisagées pour

s'adapter à une nouvelle situation d'apprentissage, le cerveau humain développe des stratégies économiques qui réduisent la quantité de traitements conscients à réaliser (Dörner et Wearing, 1995) tels que l'organisation d'informations en unités plus larges (Miller, 1956) ou encore le développement de stratégies permettant de réaliser des recherches sélectives et organisées d'informations en mémoire à long terme (Simon, 1990). Dans cette optique, le transfert des apprentissages peut être défini comme : « une stratégie approximative qui s'appuie sur certaines de ces habiletés. Lorsqu'un individu effectue un transfert, il ne se contente pas de se référer à n'importe quelle connaissance antérieure, mais plutôt à un ensemble de connaissances de haut niveau adaptables dans un nouveau contexte et qui sont modifiées à cette fin. » (Bracke, 1998, p.240).

Après avoir examiné les diverses définitions et retracé l'évolution conceptuelle du transfert des apprentissages, la section qui suit présente les principales typologies du transfert qui ont émergé dans la littérature. Ces typologies permettront de mieux cerner les différentes dimensions du transfert et d'explorer comment il a été catégorisé et étudié au fil du temps.

2.2.2 Principales typologies du transfert des apprentissages

La conceptualisation du transfert diffère souvent d'un auteur à l'autre, voire d'une tradition théorique à l'autre, si bien qu'« il serait bien difficile d'arriver à une typologie exhaustive du transfert qui susciterait un consensus réel chez les chercheurs » (Péladeau, 2005, p.188). La littérature de recherche fait ainsi état de plusieurs typologies du transfert qui font émerger certaines distinctions importantes (pour une revue exhaustive, voir Haskell, 2001 ou Toupin, 1995). Ces typologies proviennent de différentes approches théoriques et empiriques dans l'étude du transfert. Étant donné qu'il ne serait pas possible de réviser l'ensemble de la littérature sur le transfert et de discuter de l'ensemble des typologies existantes dans le cadre de cette thèse, il sera ici question des typologies les plus souvent discutées dans la littérature, qui permettent de faire ressortir des distinctions pertinentes au regard de la question de recherche de ce projet. Il convient de noter que ces typologies ne sont pas mutuellement exclusives (Haskell, 2001), et qu'un même transfert peut ainsi appartenir à plusieurs d'entre elles en fonction de ses caractéristiques spécifiques.

2.2.2.1 Transfert positif et négatif

Une première typologie permet de distinguer le transfert positif du transfert négatif (Cormier et Hagman, 1987; Ellis, 1965; Perkins et Salomon, 1992). Dans sa théorie du transfert des apprentissages, Ellis (1965)

propose qu'un transfert positif se produit lorsque des compétences, des connaissances ou des stratégies acquises dans un contexte donné améliorent ou facilitent la performance dans un nouveau contexte. En revanche, le transfert négatif survient lorsque les acquis antérieurs entravent la performance dans un nouveau contexte, souvent en raison de la discordance entre les deux situations, qui peut entraîner des interférences ou des erreurs (Bartolotti, 2015). L'influence de ce premier apprentissage sur le second est donc négative. Cette typologie du transfert positif et négatif met ainsi en évidence l'importance de la similarité entre les situations d'apprentissage antérieures et celles où le transfert est souhaité. Elle postule que plus les contextes sont similaires, plus le transfert positif est susceptible de se produire. À l'inverse, lorsque les contextes diffèrent considérablement, le transfert négatif est plus susceptible de survenir. En éducation, le transfert souhaité est donc évidemment un transfert positif, le but principal de l'enseignement étant de permettre aux apprenants d'appliquer leurs connaissances et leurs compétences dans de nouveaux contextes de manière efficace et adaptative. Il est donc souhaitable que les apprentissages qu'ils réalisent dans un contexte donné puissent faciliter les apprentissages subséquents.

2.2.2.2 Transfert spécifique et général

La catégorisation du transfert des apprentissages en transfert spécifique et général a été proposée par plusieurs chercheurs dans le domaine de la psychologie de l'apprentissage et de l'éducation (voir p. ex., Cormier et Hagman, 1987; Malcuit *et al.*, 1995; Toupin, 1995). Le transfert spécifique apparaît dans une situation où il y a une similarité claire entre les éléments de la situation d'apprentissage originale et ceux de la situation d'apprentissage transférée. Dans ce cas, les éléments clés de l'apprentissage initial correspondent étroitement à ceux de la situation de transfert; les ressemblances sont manifestes. Lorsque le transfert est plutôt réalisé sur la base de caractéristiques générales, et donc que les connaissances ou les compétences acquises dans un contexte donné sont applicables à une variété de situations différentes, le terme transfert général est employé. Il permet ainsi de qualifier les effets d'un apprentissage précédent sur un nouvel apprentissage où il n'existe aucun partage évident.

2.2.2.3 Transfert vertical et latéral

La typologie de transfert vertical et latéral a été introduite par Gagné (1965). Elle s'inscrit dans la continuité de ses précédents travaux portant sur l'enseignement programmé (Gagné, 1962). Cette typologie postule qu'il existerait une hiérarchie des apprentissages et suggère que la maîtrise de connaissances de base et d'habiletés préalables constituerait la base nécessaire pour l'acquisition de compétences de niveau supérieur. Le transfert vertical se manifeste lorsque des compétences ou des connaissances acquises à un

niveau donné d'apprentissage sont déplacées vers un niveau ultérieur ou plus avancé. En d'autres termes, le transfert vertical implique une progression naturelle des compétences et des connaissances, allant d'une étape de moindre complexité à une étape supérieure. Qui plus est, ce transfert vertical n'implique pas seulement un rôle de facilitation des apprentissages complexes, mais sous-tend l'idée qu'un individu ne peut maîtriser une compétence complexe sans d'abord maîtriser ses éléments constitutifs. La deuxième composante de cette typologie, le transfert latéral, diffère du transfert vertical en ce qu'il se produit entre des domaines ou des contextes différents, mais de niveau similaire ou équivalent. Le transfert latéral implique l'application d'apprentissages d'un domaine à un autre domaine qui partage des similitudes conceptuelles ou des compétences transposables. Cette typologie apparaît donc particulièrement pertinente considérant que ce projet de recherche s'intéresse au transfert des apprentissages de la programmation vers d'autres disciplines.

2.2.2.4 Transfert proche et éloigné

Une autre typologie de transfert concerne le fait que ce dernier soit qualifié de proche ou éloigné. Cette distinction repose sur la distance, c'est-à-dire la proximité ou l'éloignement, entre la situation d'apprentissage initiale et la situation dans laquelle le transfert est sollicité. Cette idée a été abordée par plusieurs auteurs. Par exemple, selon Mayer (1975), le transfert proche, également appelé transfert rapproché, se produit lorsque les apprenants appliquent ce qu'ils ont appris dans une situation qui est étroitement similaire à la situation d'apprentissage d'origine. En d'autres termes, les éléments clés de la situation d'apprentissage et de la situation de transfert sont très semblables. À l'inverse, le transfert éloigné, parfois appelé transfert lointain, se produit lorsque les apprenants appliquent ce qu'ils ont appris dans une situation qui est notablement différente de la situation d'apprentissage initiale. Les éléments clés de la situation de transfert sont moins semblables à ceux de la situation d'apprentissage d'origine. Mayer (1975) réfère à ces éléments clés par l'expression « complexe de stimuli » qui renvoie à l'ensemble des caractéristiques, des éléments ou des indices présents dans la situation d'apprentissage d'origine. Ces caractéristiques incluent tous les éléments qui sont perçus et pris en compte par l'apprenant lors de l'acquisition de connaissances ou de compétences dans un contexte particulier. Ils peuvent inclure des composants visuels, auditifs, textuels, conceptuels ou sensoriels qui contribuent à la compréhension et à l'exécution de la tâche. Par exemple, dans le contexte précis d'une tâche de résolution de problèmes mathématiques, le complexe de stimuli pourrait inclure les chiffres, les symboles mathématiques, les énoncés de problèmes, les diagrammes, les graphiques, etc. Une similitude entre les complexes de stimuli favorise le transfert rapproché, tandis qu'une grande différence favorise le transfert éloigné. De la même

façon, pour Perkins et Salomon (1992), le transfert proche se produit lorsque les connaissances ou les compétences acquises dans une situation d'apprentissage sont directement et immédiatement applicables à une situation similaire, alors que le transfert éloigné se produit lorsque les connaissances ou les compétences acquises dans une situation d'apprentissage sont appliquées dans un contexte qui diffère considérablement de la situation d'origine. Les éléments clés du contexte d'apprentissage initial ne sont pas directement transférables à la nouvelle situation, ce qui rend le transfert plus complexe. Le transfert éloigné nécessite de ce fait souvent une réflexion, une généralisation ou une adaptation plus approfondie de ce qui a été appris. Par ailleurs, ces auteurs soulignent qu'il n'existe pas de définition universelle de ce qui doit être considéré comme proche ou éloigné (Perkins et Salomon, 1992). Par conséquent, la catégorisation du transfert comme étant proche ou éloigné peut considérablement varier selon les auteurs et les études, ce qui a des implications importantes pour la façon dont le transfert est mesuré et évalué (Barnett et Ceci, 2002). D'ailleurs, pour Royer (1979), un transfert est plutôt qualifié de proche ou éloigné en fonction du contexte environnemental dans lequel il se produit. Un transfert proche désigne ainsi le transfert d'une situation d'apprentissage qui a lieu à l'école à une autre situation d'apprentissage se déroulant dans le même contexte, soit l'école. Un transfert éloigné désigne plutôt le transfert d'une situation d'apprentissage se déroulant à l'école vers une situation d'apprentissage qui a lieu à l'extérieur de l'école. Cette distinction apparaît toutefois moins employée, la plus courante étant celle présentée dans le paragraphe précédent.

***

À la lumière de ces différentes typologies, il apparaît pertinent de circonscrire et de qualifier le transfert faisant l'objet de ce projet, soit celui entre la programmation et les autres disciplines scolaires traditionnelles (mathématiques, sciences et français). D'abord, le transfert souhaité d'un point de vue éducatif et qui sera vérifié dans le cadre de cette recherche en est un *positif* puisqu'il est question d'un bénéfice probable découlant de l'apprentissage de la programmation pour les autres disciplines scolaires. Ensuite, ce transfert est également *général*, car il suppose que les apprentissages réalisés en programmation puissent être applicables au sein de contextes différents propres aux autres disciplines. Dans la section précédente de ce chapitre, la pensée informatique a d'ailleurs été présentée comme une compétence générale qui se développerait par la programmation, mais pourrait être mobilisée dans d'autres contextes. Par ailleurs, le transfert dont il est question dans cette étude peut aussi être qualifié de *latéral*, puisqu'il se produit entre des disciplines différentes, mais pour lesquelles certaines

compétences peuvent être partagées. Finalement, à la lumière du croisement entre l'apprentissage de la programmation, la pensée informatique et les autres disciplines scolaires réalisé par Barr et Stephenson (2011, voir Tableau 2.4), il semble que le transfert entre la programmation et les autres disciplines puisse être relativement proche ou éloigné, en fonction de la discipline cible. En effet, certains auteurs sont d'avis qu'en raison du plus grand nombre de compétences partagées par la programmation et les mathématiques, le transfert entre ces disciplines serait plus rapproché (Scherer *et al.*, 2019), alors que le transfert de la programmation vers les sciences ou les langues serait plus éloigné.

Cependant, certains chercheurs sont d'avis qu'il apparaît difficile de classer l'ensemble des transferts possibles dans l'un de ces extrêmes (Barnett et Ceci, 2002; Haskell, 2001) et que les tentatives pour catégoriser le transfert de manière strictement binaire en termes de « proche » ou « éloigné » sont souvent insuffisantes pour rendre compte de la réalité du phénomène de transfert. Barnett et Ceci (2002) ont ainsi proposé une typologie plus exhaustive afin de mieux caractériser le degré d'éloignement entre deux situations d'apprentissage. Cette typologie permet de prendre en considération différents aspects méthodologiques pour réaliser une opérationnalisation plus précise du transfert. Elle apparaît particulièrement intéressante pour ce projet, étant donné qu'il s'intéresse au transfert de la programmation vers plusieurs disciplines scolaires qui sont susceptibles de présenter des degrés d'éloignement différents.

2.2.2.5 Typologie du transfert éloigné de Barnett et Ceci (2002)

Ces auteurs ont développé une classification en neuf facteurs permettant de caractériser une situation d'apprentissage (voir Tableau 2.5). Chacun de ces facteurs peut contribuer à établir la proximité ou l'éloignement entre deux tâches, en permettant de mieux établir un degré d'éloignement pour chaque facteur individuellement (sur un continuum allant de proche à éloigné), plutôt que de procéder à une classification générale du transfert en tant que proche ou éloigné. Ces facteurs se divisent en deux dimensions principales : 1- le contenu transféré et 2- le cadre contextuel du transfert. Il est à noter que cette typologie se concentre davantage sur la caractérisation du transfert éloigné, les auteurs postulant que le minimum de transfert possible est nécessairement du transfert proche.

Tableau 2.5 Typologie du transfert de Barnett et Ceci (2002) : catégorisation des facteurs en deux dimensions

Contenu transféré	• Compétence apprise • Changement dans la performance évaluée • Demandes en mémoire liées à la tâche de transfert
Cadre contextuel du transfert	• Domaine de connaissance • Contexte physique de réalisation • Contexte temporel • Contexte fonctionnel • Contexte social d'acquisition • Modalité d'acquisition

Pour ce qui est du contenu transféré, Barnett et Ceci (2002) examinent d'une part la compétence acquise pour déterminer si elle est de nature spécifique ou générale : une procédure est ainsi plus spécifique, alors qu'un principe est plus général. Une procédure spécifique peut être caractérisée comme un ensemble d'étapes particulières décrites en termes de caractéristiques superficielles, tandis qu'un principe général peut être caractérisé comme une compréhension plus profonde, structurelle ou causale. Par exemple, ils considèrent que l'apprentissage d'une équation pour réaliser un calcul de proportions est une compétence spécifique, tandis que le développement de la capacité à raisonner selon des principes statistiques représente une compétence générale. D'autre part, ils évaluent tout changement dans la performance, à savoir si la vitesse d'exécution, l'exactitude ou l'approche employée (c'est-à-dire faire ou ne pas faire quelque chose de particulier) ont changé d'une situation d'apprentissage à l'autre. Enfin, ils prennent en compte les demandes en termes de mémoire liées à la tâche de transfert. Pour ce facteur relatif aux demandes de la mémoire, il s'agit en fait d'évaluer si le transfert exige que l'apprenant soit simplement capable d'exécuter une activité apprise, à l'aide d'indications sur la procédure correcte à appliquer, ou s'il exige que les apprenants sélectionnent également l'approche appropriée. En d'autres termes, les demandes de la mémoire associées au transfert sont susceptibles d'être plus ou moins importantes selon que les participants ont seulement à mettre à exécution une compétence bien identifiée (*prompted*) ou à reconnaître et rappeler la compétence qui doit être mobilisée. Par exemple, si une première situation consiste en la résolution d'une addition déjà posée, et que la seconde situation exige simplement la répétition de cette même procédure d'addition avec des chiffres différents, cela relève de l'exécution. Si par ailleurs la seconde situation requiert l'utilisation de l'addition, mais que celle-ci n'est pas déjà posée, il s'agit alors de reconnaissance. Enfin, si la seconde tâche consiste en la résolution d'un problème qui n'inclut aucune mention de l'addition, mais nécessite l'utilisation de cette compétence, il faudra alors

réaliser un rappel, en plus de reconnaître et exécuter. L'ensemble de ces facteurs propres au contenu transféré sont présentés au Tableau 2.6.

Tableau 2.6 Typologie du transfert de Barnett et Ceci (2002) : le contenu transféré

Contenu : Qu'est-ce qui est transféré ?			
Compétence apprise	**Transfert spécifique**	⟷	**Transfert général**
	Procédure	Représentation	Principe, heuristique
Changements dans la performance	Vitesse	Exactitude	Approche
Demandes en mémoire	Exécuter	Reconnaître et exécuter	Rappeler, reconnaître et exécuter

Traduit de « When and where do we apply what we learn? : A taxonomy for far transfer » par Barnett et Ceci, 2002, *Psychological Bulletin, 128*(4), p.621.

En ce qui concerne le cadre contextuel, Barnett et Ceci (2002) suggèrent d'évaluer le degré d'éloignement entre les deux situations ou tâches en examinant différents facteurs tels que le domaine de connaissances concerné par les deux situations, le contexte physique (le contexte au sein duquel se déroulait chacune des situations d'apprentissage), le contexte temporel (l'intervalle de temps entre les deux situations), le contexte fonctionnel (la fonction ou finalité pour laquelle la compétence est utilisée), le contexte social d'acquisition (situation d'apprentissage en groupe ou en solitaire), et la modalité d'acquisition (type et format d'une tâche). Cette classification permet ainsi d'identifier un plus grand spectre de variations sur le continuum proche/éloigné entre deux situations d'apprentissage. Le Tableau 2.7. présente cette classification en donnant des exemples permettant d'illustrer la gradation de proche à éloigné pour chacun des facteurs identifiés par les auteurs.

Tableau 2.7 Typologie du transfert de Barnett et Ceci (2002) : le cadre contextuel

Cadre contextuel : Où et quand est-ce transféré ?					
	Transfert proche	⟷			**Transfert éloigné**
Domaines de connaissance	Souris vs rat	Biologie vs botanique	Biologie vs économie	Sciences vs histoire	Sciences vs arts
Contextes physiques	Même salle dans l'école	2 salles différentes à l'école	École vs laboratoire	École vs maison	École vs plage
Contextes temporels	Même session	Lendemain	Semaines plus tard	Mois plus tard	Années plus tard
Contextes fonctionnels	Identiques	Tous les deux académiques, mais un sans évaluation	Académique vs tâche administrative	Académique vs questionnaire informel	Académique vs jeu
Contextes sociaux	Tous les deux individuels	Individuel vs paire	Individuel vs petit groupe	Individuel vs grand groupe	Individuel vs société
Modalités	Tous les deux écrits, même format	Choix de réponses vs réponses à développement	Lecture d'un livre vs examen oral	Lecture vs dégustation de vin	Lecture vs sculpture sur bois

Traduit de « When and where do we apply what we learn? : A taxonomy for far transfer » par Barnett et Ceci, 2002, *Psychological Bulletin, 128*(4), p.621.

Cette analyse plus fine des facteurs propres au contenu transféré ainsi qu'au cadre contextuel du transfert que propose la typologie de Barnett et Ceci (2002) permet d'entrevoir plusieurs éléments qu'il pourrait être intéressant de considérer afin de réfléchir au transfert possible des apprentissages de la programmation vers les autres disciplines scolaires. Cette typologie offre en effet un cadre pour décomposer les nombreuses variables associées au transfert, susceptibles d'influencer le degré d'éloignement entre deux situations d'apprentissage. L'un des facteurs identifiés dans cette typologie concerne les changements qui peuvent être observés dans la performance des apprenants pour différentes situations d'apprentissage. La section qui suit vise à présenter différentes façons de faire l'étude du transfert des apprentissages, dont la plus courante est justement de s'intéresser à la performance des apprenants pour évaluer la présence de transfert.

2.2.3 Différentes façons de faire l'étude du transfert des apprentissages

Le transfert des apprentissages est un sujet de recherche abondamment exploré, et son étude a été abordée selon diverses approches méthodologiques. Certaines recherches ont ainsi adopté une approche qualitative, par exemple pour explorer les facteurs pouvant influencer le transfert (Peters *et al.*, 2018), les défis relatifs au transfert des apprentissages (Thianthai et Sutamchai, 2022), les perspectives d'éducateurs sur leur capacité à transférer certaines compétences à de nouveaux contextes, ou encore les stratégies pour favoriser le transfert (Brion et Cordeiro, 2018). Cette approche qualitative a l'avantage de mettre en lumière les facteurs contextuels, les attitudes, les croyances et les expériences des individus liés au transfert. Elle offre ainsi une compréhension détaillée et nuancée du phénomène.

Plus récemment, d'autres études ont opté pour une approche neuroscientifique dans l'étude du transfert, en comparant notamment les processus cognitifs et cérébraux mobilisés pour différentes situations d'apprentissage (Cartwright, 2012; Ivanova *et al.*, 2020). Ce type d'étude permet de s'intéresser aux régions cérébrales qui sont activées pour ces différentes situations, ce qui contribue à renseigner sur la possibilité de transfert entre celles-ci. Il permet aussi d'examiner si certains entraînements cognitifs peuvent engendrer un transfert des apprentissages, et s'il est possible d'observer ce transfert sur le plan cérébral (p. ex., Wexler *et al.*, 2016).

Par ailleurs, la façon la plus courante d'évaluer et de mesurer le transfert, et plus précisément *les effets* de transfert, est par le biais de devis de recherche expérimentaux ou quasi expérimentaux (Péladeau, 2005). Le devis de recherche le plus typique pour évaluer le transfert des apprentissages d'une situation A à une situation B implique une comparaison de deux groupes : un groupe expérimental prend ainsi part à une situation d'apprentissage A, puis à une seconde situation d'apprentissage B. Le plus souvent ces situations sont des tâches que les participants réalisent, la tâche A étant la tâche source ou la tâche initiale et la tâche B étant la tâche cible. Un groupe de contrôle prend aussi part à la situation d'apprentissage B, sans l'apprentissage préalable de la tâche A (mais si le groupe de contrôle est actif, il prend néanmoins part à une autre situation d'apprentissage A′). Si la performance du groupe expérimental à la tâche B est supérieure à celle du groupe de contrôle, on suppose généralement qu'il y a eu un transfert positif de la tâche A vers la tâche B. Si, au contraire, la performance du groupe expérimental est inférieure à celle du groupe de contrôle, le transfert est considéré négatif. Enfin, si la performance des deux groupes est équivalente, on conclut alors qu'il n'y a pas eu de transfert des apprentissages (transfert nul). Dans ce type de devis, la nature de l'effet de transfert (positif, négatif ou nul) est donc inférée en analysant l'influence

de la tâche A sur la performance à la tâche B d'un groupe expérimental, comparativement à un groupe de contrôle. Le transfert en tant que tel n'est donc pas directement observé : il est plutôt déduit des performances qui sont mesurées pour différentes situations. Cette observation met en lumière une dimension importante dans la définition du transfert des apprentissages : il s'agit d'un construit qu'il n'est pas possible de mesurer directement. La Figure 2.3 illustre ce devis le plus traditionnellement utilisé.

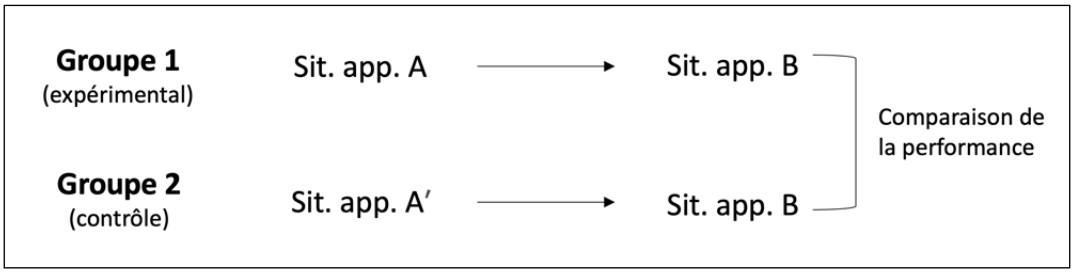

Figure 2.3 Devis de recherche le plus typique pour évaluer le transfert des apprentissages d'une situation d'apprentissage A à une situation d'apprentissage B.

Cette impossibilité de mesurer directement le transfert amène certains auteurs à considérer ce dernier comme un construit latent, c'est-à-dire un construit théorique qu'il n'est pas possible d'observer directement et qui doit donc être inféré à partir de plusieurs indicateurs ou variables manifestes (Bollen, 2002). Ces indicateurs sont le plus souvent les performances à des tâches (une tâche initiale et une tâche cible). Or, les tâches sont toujours des indicateurs imparfaits d'une capacité cognitive (Noack *et al.*, 2014). Dans le cas précis du transfert des apprentissages associé à la programmation, il semble donc que cette dimension latente puisse être en partie associée aux capacités cognitives relevant de la pensée informatique, qui, de l'avis de plusieurs chercheurs (Lye et Kho, 2014; Wing, 2006), sous-tend le processus de transfert de la programmation vers d'autres disciplines. Pour cette raison, certains auteurs soulignent que l'étude des effets de transfert devrait être réalisée par le biais de méthodes permettant de reconnaître le caractère latent de ce construit (McArdle et Prindle, 2018; Melby-Lervåg, 2018; Noack *et al.*, 2014). Cette considération théorique mérite d'être mise de l'avant, car elle orientera certains des choix méthodologiques qui seront présentés au chapitre 3 de cette thèse.

Les effets de transfert pouvant découler de l'apprentissage de la programmation et du développement d'une pensée informatique font d'ailleurs l'objet d'un modèle théorique qui est présenté dans la section qui suit.

2.2.4 Modèle du transfert des apprentissages associé à l'apprentissage de la programmation

Le modèle de Popat et Starkey (2019) émerge d'une revue systématique de la littérature comprenant dix études clés, visant toutes à mesurer les effets de transfert découlant de l'apprentissage de la programmation chez les jeunes. Ces articles présentaient soit des données quantitatives provenant d'élèves âgés de 5 à 17 ans, ou des données qualitatives provenant d'élèves et d'enseignants du primaire. Ces études ont été menées dans différents pays, notamment les États-Unis, la Nouvelle-Zélande, la Grèce, l'Irlande, la Turquie et l'Espagne, et ont été publiées entre 1988 et 2017. Sur la base des résultats présentés dans ces articles, Popat et Starkey ont réalisé une analyse thématique de contenu basée sur la taxonomie révisée de Bloom, ce qui les a menés à proposer un modèle (voir Figure 2.4) synthétisant les effets de transfert possibles associés à l'apprentissage de la programmation. Il est à noter que les auteurs n'utilisent pas directement le terme « effets de transfert » dans ce modèle, parlant plutôt des effets sur le plan éducatif (*educational outcomes*), mais ceux-ci nous apparaissent clairement relever du processus de transfert.

Au centre de ce modèle se situe l'apprentissage du code puisqu'il s'agit de l'effet premier de l'apprentissage de la programmation. Le modèle inclut également une dimension qui concerne le curriculum et le design pédagogique (en gris dans la figure), qui fait référence à la conception des programmes d'études et des méthodes pédagogiques employées et à la façon dont ils intègrent l'apprentissage de la programmation. Cette dimension est en effet susceptible d'influencer la capacité des apprenants à transférer leurs compétences ou connaissances d'une situation d'apprentissage à une autre. En périphérie, le modèle présente ensuite trois catégories centrales d'effets de transfert pouvant découler de l'apprentissage de la programmation, allant au-delà du codage : 1- les compétences en raisonnement de haut niveau (*high order thinking skills;* en bleu dans la figure), 2- les compétences personnelles (en vert dans la figure) et 3- les compétences en raisonnement de bas niveau (*low order thinking skills,* en jaune dans la figure)

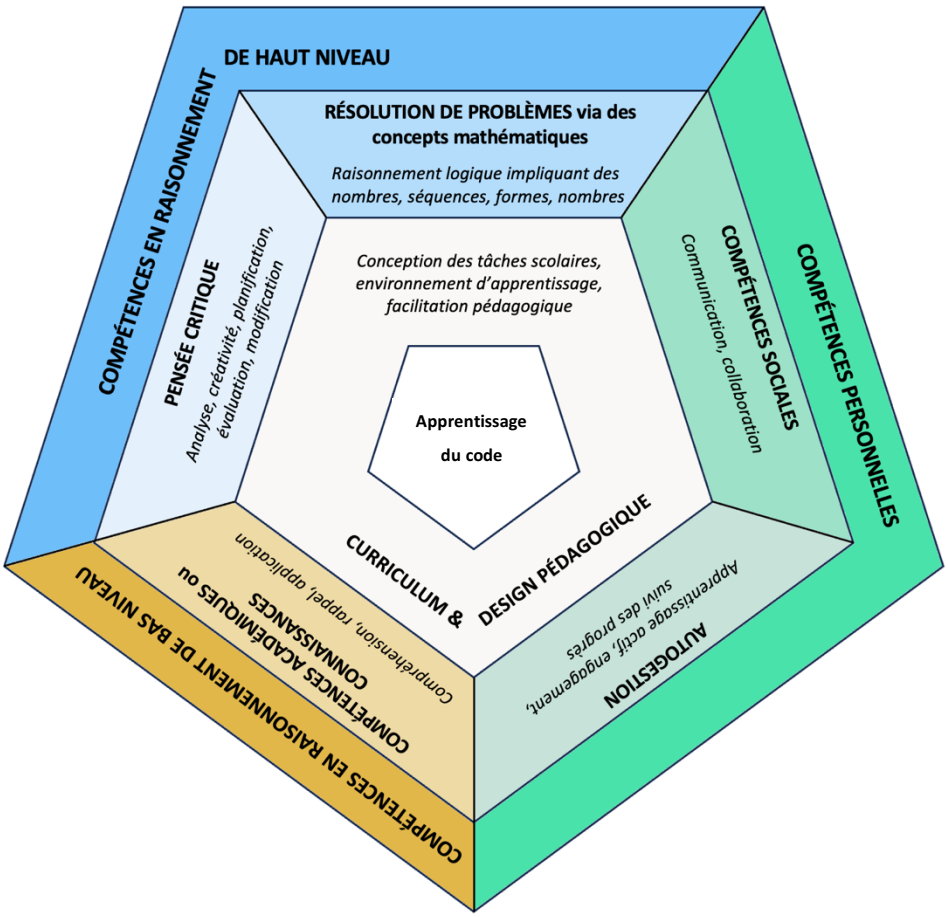

Figure 2.4 Modèle décrivant les effets possibles de l'apprentissage de la programmation sur différents plans. Reproduit et adapté de « Learning to code or coding to learn? A systematic review » par Popat et Starkey, 2019, *Computers & Education*, *128*, p.370.

Ces trois catégories centrales englobent des compétences plus ciblées qui, selon les auteurs, sont interdépendantes. Les compétences en raisonnement de haut niveau comprennent ainsi des compétences associées à la pensée critique, telles que l'analyse et la créativité, et à la résolution de problèmes, telles que le raisonnement logique (en bleu pâle dans la figure), qui s'influencent mutuellement (la pensée critique étant nécessaire pour résoudre des problèmes). Les compétences personnelles comprennent pour leur part des compétences sociales comme la communication et la collaboration ainsi que des compétences en autogestion telles que l'engagement (en vert pâle dans la figure). Finalement, les

compétences en raisonnement de bas niveau incluent des compétences académiques pouvant toucher à différentes disciplines, telles que la compréhension, le rappel et l'application (en jaune pâle dans la figure).

Dans le cadre du présent projet, le modèle de Popat et Starkey (2019) revêt une importance particulière puisqu'il s'agit du seul cadre théorique portant spécifiquement sur les effets de transfert associés à l'apprentissage de la programmation chez les jeunes. Ce modèle revêt également une pertinence élevée étant donné qu'il s'appuie sur les données empiriques des études faisant partie de la revue systématique menée par les auteurs. Les sections qui suivent abordent plus en détail les effets de transfert documentés pour chacune des trois catégories centrales de ce modèle.

2.2.4.1 Compétences en raisonnement de haut niveau

Parmi les trois catégories de ce modèle, la catégorie des compétences en raisonnement de haut niveau est celle qui a été le plus largement étudiée dans la littérature scientifique. Dans la revue de la littérature de Popat et Starkey (2019), l'ensemble des études se sont intéressées aux effets de la programmation sur les compétences en raisonnement de haut niveau.

Les compétences en raisonnement de haut niveau sont des compétences cognitives qui vont au-delà de la simple mémorisation ou de l'application mécanique de connaissances (Merta, 2017). Dans la littérature, elles incluent le plus souvent la résolution de problèmes, la pensée critique, la pensée créative et la prise de décisions, qui sont généralement présentées comme des compétences transférables et utiles dans divers contextes académiques et professionnels (Robins, 2019). Dans leur modèle, Popat et Starkey (2019) incluent uniquement la résolution de problème et la créativité, puisque leur revue systématique de la littérature ne leur a permis de relever des effets de transfert significatifs que pour ces deux composantes.

Comme cela a été mis de l'avant un peu plus tôt dans ce cadre théorique, la résolution de problème est centrale à l'acte de programmer. De ce fait, le transfert de la programmation vers la résolution de problème est l'effet qui est le plus souvent attendu ou même « promis » d'un point de vue théorique (Scherer *et al.*, 2019). À cet égard, l'ensemble des dix articles compris dans la revue de la littérature de Popat et Starkey (2019) ont abordé la résolution de problèmes, tous pour des contextes de résolution de problèmes en mathématiques. Des études comme celles de Fessakis (2013), Falloon (2016) et Kalelioğlu (2015) arrivent toutes à la conclusion que le fait d'utiliser des activités mobilisant la programmation pourrait contribuer aux compétences en résolution de problèmes chez les jeunes élèves, parfois aussi

jeunes que cinq ans. De la même façon, des études antérieures menées par Bernardo et Morris (1994) ainsi que Palumbo et Reed (1991) auprès d'élèves de 14 à 18 ans ont constaté que l'apprentissage de la programmation améliorait de manière significative les compétences de résolution de problèmes en mathématiques. Par ailleurs, d'autres études arrivent à des résultats qui ne sont pas toujours aussi convergents : Miller (1988) a ainsi trouvé que seuls les garçons du groupe expérimental prenant part à des interventions visant l'apprentissage de la programmation présentaient de meilleures compétences en résolution de problèmes, comparativement au groupe de contrôle. Dans le cadre d'une étude quasi expérimentale, Psycharis et Kallia (2017) ont constaté que l'utilisation de la programmation améliorait de manière significative la résolution de problèmes mathématiques d'élèves de 16-17 ans, bien que l'amélioration constatée ne soit pas significativement supérieure à celle d'un groupe témoin qui n'utilisait pas la programmation. D'autres études ont également remis en question la pertinence de la programmation pour améliorer les compétences en résolution de problèmes : c'est notamment le cas de Falloon (2016) ainsi que Kalelioğlu et Gülbahar (2014) qui n'ont constaté aucune amélioration significative des compétences en résolution de problèmes à la suite de l'apprentissage de la programmation via le logiciel Scratch, ainsi que de Hayes et Stewart (2016) qui ont conclu que l'apprentissage de la programmation semblait moins efficace qu'une intervention éducative visant spécifiquement la résolution de problèmes.

Au-delà de la revue menée par Popat et Starkey (2019) chez les jeunes apprenants, une méta-analyse de Scherer *et al.* (2019), cette fois principalement chez les apprenants adultes, fournit également des résultats quant au transfert découlant de la programmation vers les compétences en résolution de problèmes. Ces auteurs rapportent ainsi que l'apprentissage de la programmation pourrait influencer significativement les compétences en résolution de problèmes; la taille d'effet rapportée étant de 0,47, la magnitude de cet effet serait donc modérée selon les balises établies par Cohen (2013). De façon convergente, Liao (2000) avait pour sa part rapporté dans sa méta-analyse portant sur les effets de programmation une taille d'effet un peu plus importante, soit de 0,58, ce qui l'avait mené à conclure que l'apprentissage de la programmation pouvait réellement contribuer à l'amélioration des compétences en résolution de problèmes. Il apparaît toutefois important d'aborder cette conclusion avec une certaine prudence étant donné la variabilité et l'hétérogénéité des études incluses dans cette méta-analyse. Cependant, ces résultats soutiennent l'idée que l'apprentissage de la programmation pourrait avoir des implications au-delà du domaine de la programmation lui-même et qu'il est possible qu'il puisse faciliter

l'apprentissage dans d'autres disciplines, en particulier celles où la résolution de problèmes joue un rôle central.

La deuxième composante de la catégorie des compétences en raisonnement de haut niveau, la pensée critique, a pour sa part été étudiée dans cinq des dix études examinées par Popat et Starkey (2019). Dans ces études, la pensée critique englobe diverses compétences, notamment l'analyse d'une tâche, l'interprétation de l'information, la création et la mise en œuvre d'un plan, l'évaluation des résultats, et l'ajustement des performances pour atteindre les objectifs souhaités. Falloon (2016) a notamment constaté que des élèves de 5 et 6 ans utilisant Scratch Junior pour apprendre les formes de base dans le cadre d'une activité en mathématiques déployaient des compétences d'évaluation afin de prédire le résultat attendu avant d'exécuter leur code. Kalelioğlu (2015) a quant à lui observé une légère amélioration, toutefois non significative, pour certaines compétences de la pensée critique (notamment l'analyse et l'évaluation), à la suite d'une activité de programmation réalisée par des élèves de dix ans sur la plateforme Code.org. Finalement, Palumbo et Reed (1991) ont utilisé un test standardisé permettant d'évaluer la pensée critique et ont observé une meilleure performance pour la compétence d'interprétation de l'information chez des élèves de 14 à 18 ans apprenant la programmation, comparativement à un groupe de contrôle actif.

Bien que très variables dans leur façon d'évaluer la pensée critique, ces études suggèrent dans l'ensemble que l'apprentissage de la programmation pourrait favoriser le développement de compétences en pensée critique chez de jeunes apprenants. Qui plus est, la méta-analyse de Liao (2000) portant sur les effets de transfert associés à l'apprentissage de la programmation chez les adultes s'est également intéressée aux incidences sur la pensée critique et a rapporté une taille d'effet moyenne de 0,48.

Néanmoins, malgré ces résultats assez encourageants, certains auteurs rappellent que les effets de transfert pouvant découler de l'apprentissage de la programmation vers des compétences en raisonnement de haut niveau font encore l'objet de débats scientifiques (Lodi, 2020). Il existe d'ailleurs un corpus de recherches qui remet en question la possibilité d'améliorer ces compétences de façon générale à travers d'autres apprentissages. Des études ont en effet montré que ces compétences ne sont pas facilement enseignées ni même transférées en dehors du contexte expérimental (Brown, 1992). Cette idée a d'ailleurs été avancée dès les premières études de Thorndike et ses collègues, au début du 20^{e} siècle (Thorndike et Woodworth, 1901; Thorndike, 1924). Ceux-ci concluaient que l'amélioration des

« compétences générales de la pensée » par l'étude d'une discipline particulière n'est pas aussi facile que l'on pourrait l'imaginer. À cet égard, malgré les résultats positifs que souligne leur méta-analyse, Scherer et ses collègues (2019) se montrent eux-mêmes très prudents concernant le transfert de la programmation vers des compétences en raisonnement de haut niveau.

2.2.4.2 Compétences personnelles

Bien que cette catégorie des compétences personnelles puisse a priori paraître un peu différente des autres, il n'est pas si surprenant qu'elle ait intégré ce modèle. Elle trouve en effet un écho dans la définition plus large de la programmation avancée par Guzdial (2019), qui considère la programmation comme bien plus qu'une simple compétence technique, la voyant également comme un moyen de communication et un outil d'apprentissage. De plus, plusieurs aspects de la pensée informatique, tels que « Communiquer et collaborer » ou « Faire preuve de persévérance face aux problèmes complexes » s'alignent naturellement avec ces compétences personnelles. Par ailleurs, l'idée selon laquelle les apprenants puissent acquérir une forme d'autonomie par la programmation s'accorde avec la perspective constructionniste de Papert, selon laquelle les étudiants prennent en main leur propre processus d'apprentissage (Papert, 1980). Ainsi, l'inclusion de cette catégorie au sein du modèle de Popat et Starkey reflète la complexité et de la polyvalence des compétences qui apparaissent associées à l'apprentissage de la programmation.

Néanmoins, peu d'études ont documenté le développement de ces compétences personnelles par l'apprentissage de la programmation. Parmi celles faisant partie de la revue de la littérature de Popat et Starkey (2019), quatre ont étudié ces compétences personnelles et elles l'ont souvent fait de manière plus périphérique, sans qu'il s'agisse de l'objectif principal de la recherche. Kalelioğlu (2015) a ainsi fait valoir que les élèves de dix ans partageaient leur code avec d'autres qui avaient besoin d'aide, indiquant, selon Popat et Starkey (2019), un certain développement de compétences sociales. Similairement, Fessakis (2013) a observé l'établissement d'un dialogue entre les élèves dans le cadre d'une tâche en programmation. Ces observations suggèrent que la programmation pourrait favoriser une interaction entre les élèves et de ce fait le développement de compétences sociales. Ces mêmes auteurs ont également observé que certains des élèves progressaient par essai et erreur, tandis que d'autres devaient demander de l'aide, ce qui nécessitait des compétences d'autorégulation. De la même façon, Kalelioğlu (2015) a aussi observé que les élèves étaient plus enclins à suivre leur propre rythme lors de l'utilisation de la plateforme Code.org, ce qui suggère également un certain degré d'autorégulation.

Ces résultats suggèrent donc que l'apprentissage de la programmation pourrait contribuer au développement de compétences personnelles. Néanmoins, ils reposent pour la plupart sur des observations rapportées dans le cadre d'études dont le but premier n'était pas de documenter les compétences personnelles des participants; le devis de recherche n'était donc pas optimisé à cette fin. Ces résultats, bien qu'intéressants, sont donc à appréhender avec prudence.

2.2.4.3 Compétences en raisonnement de bas niveau

Selon le modèle de Popat et Starkey (2019), les compétences en raisonnement de bas niveau reposent sur la capacité à comprendre et à appliquer du contenu. Ces compétences comprennent principalement la mémorisation, la compréhension, et l'application des connaissances disciplinaires externes à la programmation et aux mathématiques.

Dans cette catégorie et au terme de leur revue de la littérature, Popat et Starkey (2019) mentionnent explicitement qu'ils ont identifié très peu d'études qui ont examiné les effets de l'apprentissage de la programmation sur les compétences académiques de jeunes apprenants dans des disciplines scolaires traditionnelles autres que les mathématiques et l'informatique. Certaines, comme Hayes et Stewart (2016), ont constaté une amélioration des compétences en lecture et en orthographe chez des élèves âgés de 10 à 11 ans qui avaient pris part à une intervention en programmation via Scratch. De même, Sáez-López (2016) a observé que l'intégration de la programmation dans le programme d'histoire de l'art semblait avoir aidé les élèves à mieux comprendre certains concepts liés à cette discipline. Considérant ce vide important dans les connaissances, Popat et Starkey (2019) insistent sur la nécessité de conduire des études supplémentaires afin d'explorer ces effets plus en détail.

Par ailleurs, la méta-analyse de Scherer *et al.* (2019), qui inclut presque exclusivement des études menées chez les enfants, apporte des informations supplémentaires. Leurs résultats indiquent que l'efficacité du transfert des apprentissages associé à la programmation varierait en fonction de la discipline ciblée. Par exemple, leur méta-analyse met en évidence un transfert positif de la programmation vers les mathématiques, avec une taille d'effet notable de 0,57. En revanche, en ce qui concerne le transfert vers l'apprentissage d'une langue, leur méta-analyse ne suggère aucun effet de transfert significatif (taille d'effet nulle). Enfin, cette même méta-analyse indique que le transfert vers d'autres disciplines scolaires (sciences, sciences sociales, arts) apparaît possible et même positif, mais moins important; la taille d'effet est à cet égard plus faible (0,28). Il est toutefois à noter que le transfert de la programmation vers ces

autres disciplines scolaires représentait la catégorie contenant le plus petit échantillon d'études de leur méta-analyse : sur un total de 105 études, seulement cinq s'intéressaient au transfert vers des disciplines scolaires autres que les mathématiques et les langues.

Bien que limités, ces premiers résultats apparaissent particulièrement pertinents au regard de la question de recherche sous-tendant ce projet qui s'intéresse aux effets de transfert de la programmation vers d'autres disciplines scolaires. Ils soulignent notamment l'intérêt de mieux comprendre les similitudes entre la discipline de la programmation et les autres disciplines scolaires, ainsi que leur degré de proximité, afin d'anticiper les possibles effets de transfert. Étant donné l'importance des études portant spécifiquement sur le transfert de la programmation vers les autres disciplines scolaires, la prochaine section est entièrement consacrée à réaliser un état des connaissances approfondi sur la question. Cet état des connaissances sera présenté pour les trois disciplines scolaires centrales que sont les mathématiques, les sciences et les langues, pour lesquelles la littérature existante, bien que lacunaire, présente de premiers résultats permettant d'asseoir les hypothèses de ce projet.

2.3 État des connaissances sur le transfert des apprentissages de la programmation vers les disciplines scolaires traditionnelles auprès des jeunes apprenants

Étant donné que les études discutées dans la présente section sont plus étroitement liées à la question de recherche sous-jacente à ce projet, une attention particulière sera accordée à leur méthodologie. Cela permettra ensuite de mieux justifier les choix méthodologiques de ce projet.

2.3.1 Transfert des apprentissages de la programmation vers les mathématiques

La discipline des mathématiques est celle pour laquelle les effets de transfert ont été le mieux documentés. Cette exploration a d'ailleurs commencé dès les débuts de l'ère informatique, alors que Papert s'est intéressé à la manière dont les apprentissages en mathématiques pouvaient bénéficier de l'apprentissage de la programmation, ce qui a ultimement conduit au développement de sa théorie du constructionnisme (1980). La littérature scientifique compte aujourd'hui un grand nombre d'études portant sur le transfert des apprentissages de la programmation vers les mathématiques. Ces études adoptent une multitude de formes, couvrant un large éventail de méthodologies, des approches qualitatives aux approches quantitatives, en passant par des devis quasi expérimentaux ou corrélationnels. Elles se concentrent généralement sur les effets d'interventions de courte durée, font l'étude de la programmation à travers une diversité d'outils et de langages, et examinent les effets sur des contenus mathématiques très variés

allant du préscolaire à la fin du secondaire. Certaines de ces études rapportent des effets de transfert positifs et significatifs, tandis que d'autres non. Parmi cette abondance de recherches, caractérisées par une variabilité notable dans leurs choix méthodologiques, il apparaît donc complexe de dégager une vision claire du transfert entre ces deux disciplines. Cette section n'a donc pas pour ambition d'examiner et de présenter de manière exhaustive chacune des études existantes sur ce sujet, ce qui ne serait d'ailleurs pas pertinent. Nous réaliserons plutôt l'étude de la question en privilégiant la méta-analyse, laquelle permet de synthétiser les résultats de multiples études pour obtenir une perspective globale et plus robuste sur le sujet.

À cet égard, l'étude la plus récente et exhaustive s'étant directement intéressée aux effets de transfert entre la programmation et les mathématiques a été menée à travers la méta-analyse de Scherer *et al.* (2019) qui est la seule méta-analyse portant sur les effets de transfert découlant de l'apprentissage de la programmation sur différentes habiletés cognitives ainsi que sur les disciplines scolaires ayant inclus des études menées auprès de jeunes apprenants. Ces potentiels effets de transfert ont été catégorisés en termes de transfert proche (*near transfer*) et transfert éloigné (*far transfer*), les disciplines scolaires faisant partie de cette dernière catégorie de transfert. Cette méta-analyse comprend 33 études (sur un échantillon total de 105) traitant spécifiquement du transfert des apprentissages de la programmation vers les mathématiques chez les jeunes, la majorité d'entre elles rapportant des effets de transfert positifs. Il convient de noter que cette méta-analyse englobe la plupart des études précédemment rapportées au sein de revues de la littérature, à savoir celles de Grover et Pea (2013), Moreno-Léon et Robles (2016) ainsi que Hickmott (2017), qui ne s'étaient pas seulement ou directement concentrées sur les effets de transfert, mais sur l'intégration de la programmation au cursus scolaire de manière plus générale. Nous articulerons donc notre analyse actuelle autour de la méta-analyse la plus récente conduite par Scherer et son équipe en 2019, laquelle va au-delà de la simple revue de la littérature en proposant une analyse plus approfondie et mesurable des effets associés à chaque étude. Nous détaillerons plusieurs des études incluses dans cette méta-analyse afin de faire ressortir des exemples saillants permettant d'illustrer comment l'étude du transfert entre la programmation et les mathématiques a été réalisée. Certaines études supplémentaires seront également abordées afin de souligner les principaux points de convergence dans les résultats obtenus.

Parmi les études rapportant des effets de transfert positifs entre les deux disciplines, on compte notamment celle de Brown (2008) qui a été menée auprès d'élèves de 5^{e} et 6^{e} années du primaire aux

États-Unis (total de 113 élèves âgés de 10 à 12 ans). Ces chercheurs ont examiné l'impact de l'apprentissage de la programmation sur les compétences en résolution de problèmes en mathématiques. À cet effet, un groupe expérimental composé de 73 élèves a pris part à une intervention constituée de quatre séances de 45 minutes portant sur la programmation avec Scratch, tandis qu'un groupe de contrôle passif composé de 40 élèves ne prenait pas part à cette intervention en programmation. Pour évaluer l'impact de l'apprentissage de la programmation sur les compétences en résolution de problèmes, les élèves des deux groupes étaient invités à résoudre divers problèmes mathématiques à deux reprises, soit avant et après l'intervention. Les résultats de l'étude ont révélé que les élèves ayant suivi la formation en programmation avec Scratch parvenaient mieux à décomposer les concepts mathématiques et à aborder les problèmes qui leur étaient soumis en étapes simples, comparativement au groupe de contrôle. Cette différence était associée à une taille d'effet importante (g = 0,968), telle que calculée par Scherer *et al.* (2019). Ces observations ont été interprétées par les auteurs comme une amélioration des compétences en résolution de problèmes. En conséquence, les auteurs ont conclu que l'apprentissage de la programmation, en particulier au travers d'environnements comme Scratch, peut constituer un moyen efficace pour renforcer les compétences en résolution de problèmes des élèves en mathématiques. Ces résultats convergent avec les résultats d'autres études, notamment celle de Kazakoff et ses collègues (2013) qui a montré que des enfants d'âge préscolaire qui prenaient part à un atelier intensif portant sur la robotique et la programmation présentaient une amélioration dans leur capacité à séquencer des événements, une compétence fondamentale en mathématiques. Une étude de Clements et Sarama (2009) également menée auprès d'enfants d'âge préscolaire a pour sa part permis d'observer une amélioration significative de leur compréhension de concepts de base en mathématiques, au terme d'une intervention intégrant la technologie (dont certains éléments de programmation) au jeu, dans le contexte des mathématiques.

De façon similaire, une autre étude menée par Calao et ses collaborateurs (2015) a également révélé des effets de transfert positifs. L'échantillon de cette étude était constitué de 42 élèves de 6^{e} année en Colombie, répartis en un groupe expérimental de 24 élèves et un groupe de contrôle de 18 élèves. Le groupe expérimental réalisait un apprentissage structuré de la programmation avec le logiciel Scratch, sur une période de trois mois. En comparaison, le groupe de contrôle a continué à réaliser les activités pédagogiques traditionnelles sans apprendre la programmation avec Scratch. Les effets de l'intervention ont été mesurés sur les compétences des élèves pour quatre processus mathématiques clés, soit la modélisation, le raisonnement, la résolution de problèmes et l'exécution de procédures et d'algorithmes.

Les résultats indiquent que le groupe expérimental a montré une amélioration significativement plus grande que le groupe de contrôle pour l'ensemble des processus mathématiques évalués, l'exécution de procédures et d'algorithmes étant le processus pour lequel l'amélioration était la plus importante. Qui plus est, la taille d'effet associée à cette différence était importante (g = 2,2). Selon les auteurs de l'étude, les gains observés dans le groupe expérimental suggèrent donc que l'apprentissage de la programmation avec Scratch peut avoir un effet bénéfique sur l'acquisition de compétences mathématiques chez les élèves. Ces résultats rejoignent ceux de l'étude de Bers (2014) menée auprès d'enfants du préscolaire qui a révélé une amélioration significative des compétences en résolution de problèmes mathématiques chez les enfants d'un groupe expérimental ayant pris part à un programme en robotique éducative, comparativement à un groupe de contrôle. Ils convergent également avec les résultats de Sáez-López (2016) qui a mesuré les effets de l'intégration de la programmation avec Scratch au sein du cursus scolaire des élèves de niveau primaire. Les chercheurs de cette étude rapportent que cette intégration avait eu un impact positif et significatif sur les résultats scolaires des élèves en mathématiques.

Par ailleurs, il convient de noter que certaines de ces études présentaient une limite importante au regard de la qualité de la mesure de contrôle utilisée. En effet, certains des groupes de contrôle étaient de type passif ou ne prenaient part à aucune intervention comparable à celle du groupe expérimental. Cela fait en sorte qu'il n'est pas possible de déterminer si l'intervention portant sur la programmation avait réellement un effet spécifique par rapport à d'autres interventions ou traitements similaires. À ce propos, deux études qui ont utilisé un groupe de contrôle actif mettent en perspective les résultats des études précédentes.

Dans celle de Hayes et Stewart (2016), deux groupes d'enfants âgés de 10 à 12 ans ont été soumis à des interventions différentes : le premier groupe (groupe de contrôle actif) a pris part à une intervention informatisée visant le développement de compétences en résolution de problèmes, tandis que le second groupe (groupe expérimental) prenait part à une intervention en programmation utilisant Scratch. L'objectif de l'étude était d'évaluer l'impact de ces interventions sur diverses mesures cognitives et académiques. Les résultats ont révélé que les deux groupes ont montré des améliorations dans plusieurs domaines, le groupe de contrôle actif présentant toutefois des gains significativement plus importants, notamment pour les compétences en mathématiques (telles qu'évaluées par le Drumcondra Primary Mathematics Test), qui incluaient notamment des questions en arithmétique, géométrie et statistiques. Ce résultat s'est traduit par une taille d'effet de g = -0,241, telle que rapportée par Scherer et ses collègues (2019). Cette étude suggère donc que d'autres types d'intervention ciblant plus directement certaines

compétences en mathématiques pourraient s'avérer plus efficaces pour l'apprentissage des mathématiques que des interventions utilisant la programmation, ce qui, selon les auteurs, nuance l'efficacité du transfert potentiel entre la programmation et les mathématiques. Qui plus est, une étude de Ortiz (2015) a pour sa part examiné l'impact de l'intégration de la programmation et de la robotique LEGO dans l'enseignement des notions de ratios et de proportions chez des élèves de 5e année aux États-Unis. Un groupe expérimental composé de 15 élèves suivait un programme d'intervention d'une semaine qui intégrait la robotique LEGO dans l'enseignement des contenus mathématiques portant sur les ratios et les proportions. Ce programme était dispensé en cinq séances totalisant 15 heures. Un groupe de contrôle recevait pour sa part un enseignement plus traditionnel basé sur des manuels, également axé sur les ratios et les proportions. Les deux groupes étaient évalués à l'aide d'un examen maison à trois moments différents afin de mesurer l'évolution de leur compréhension des ratios et des proportions. Les résultats obtenus indiquent que le groupe expérimental ne présentait pas une amélioration significativement différente de celle observée dans le groupe de contrôle, ce qui suggère également qu'une intervention en programmation n'est pas nécessairement plus efficace au regard de l'apprentissage de contenus mathématiques. La taille d'effet rapportée par Scherer *et al.* (2019) pour cette étude était petite, soit de $g = 0{,}316$.

La méta-analyse de Scherer et ses collègues (2019) semble par ailleurs avoir omis d'inclure l'étude de Boylan (2018) utilisant ScratchMaths, qui est l'une des rares à bénéficier d'un protocole solide pour évaluer de manière écologique les effets de la programmation sur l'apprentissage des mathématiques. Cette étude randomisée contrôlée a évalué les effets d'une intervention réalisée pendant deux années en Angleterre auprès de 6 232 élèves âgés de 9 à 11 ans. Elle a examiné à la fois l'influence de la programmation sur les compétences liées à la pensée informatique et sur les compétences en mathématiques. Les compétences mathématiques ont été évaluées à deux moments distincts, soit deux ans avant le début de l'intervention et immédiatement après l'intervention, en se basant sur les résultats des évaluations nationales en mathématiques. Les résultats de cette étude n'ont pas permis d'observer un effet de transfert de la programmation vers les mathématiques (taille d'effet de 0,03). Par ailleurs, étant donné qu'une période de quatre années s'est écoulée entre les deux moments de mesure, il est possible que les enseignants aient mis en œuvre des activités supplémentaires en programmation, tant pour le groupe expérimental que pour le groupe de contrôle, ce qui aurait pu contribuer à réduire la capacité à détecter une différence entre les deux groupes.

Outre ces articles, une communication de Zavala *et al.* (2013) a présenté les résultats d'une étude visant à tester les effets de l'utilisation de Scratch sur les capacités d'élèves de 8 et 9 ans à conceptualiser la signification des nombres et leurs relations. Ces chercheurs ont ainsi comparé deux groupes : le groupe expérimental, composé de 27 élèves, bénéficiait d'une intervention avec Scratch comprenant trois séances de 30 minutes réparties sur deux semaines. Cependant, les auteurs ne précisent pas la nature des activités qui étaient réalisées par les élèves du groupe de contrôle. Les résultats indiquent que le groupe expérimental présentait une meilleure performance au regard de la signification des nombres et de l'ordonnancement de séquences de nombres, comparativement au groupe de contrôle. La validité de ces résultats dépend néanmoins de l'équivalence du groupe de contrôle.

Finalement, une étude corrélationnelle (Lewis et Shah, 2012) a aussi été conduite auprès de 47 enfants de 6^{e} année, participant à un programme visant l'apprentissage des langages de programmation Scratch, Snap et LOGO, lors d'une colonie de vacances (36 heures réparties sur 12 jours). Ces enfants répondaient à des jeux-questionnaires visant à tester leurs connaissances en programmation du 2^{e} au 10^{e} jours du programme. Les résultats de l'étude ont révélé une forte corrélation entre les performances en mathématiques (résultats aux tests nationaux partagés par 22 élèves de l'échantillon) et les scores obtenus aux jeux-questionnaires (t = 3,46, p = 0,002). En raison de sa nature corrélationnelle ainsi que de la petite taille de son échantillon, il convient toutefois d'appréhender les résultats de cette étude avec prudence.

Pour leur part, à l'issue de leur méta-analyse, Scherer et ses collègues (2019) ont mis en lumière un effet de transfert positif de l'apprentissage de la programmation vers les mathématiques, illustré par une taille d'effet significative de 0,57. À la lumière de ces données, il semble donc que le transfert de la programmation vers les mathématiques soit quelque chose de réalisable, de nombreuses études montrant en effet des améliorations pour diverses notions et compétences mathématiques, des suites de l'apprentissage de la programmation.

Cependant, il est essentiel de reconnaître que plusieurs de ces études s'accompagnent de limites méthodologiques importantes pouvant entraver la validité des résultats. Par exemple, certaines études ont complètement omis de mettre en place des groupes de contrôle ou alors ont utilisé des groupes de contrôle passifs, et certaines autres présentaient des échantillons de taille très restreinte. Cette situation a d'ailleurs conduit Hickmott et ses collègues à souligner au terme de leur revue de la littérature (2017)

qu'il existe un manque de recherches rigoureuses permettant d'établir un lien entre l'apprentissage de la programmation et l'apprentissage en mathématiques, tel que reflété dans les performances scolaires des élèves. La section qui suit présente un état des connaissances sur le transfert des apprentissages de la programmation vers la discipline des sciences.

2.3.2 Transfert des apprentissages de la programmation vers les sciences

Ces dernières années, il est de plus en plus courant d'observer une intégration de l'enseignement de la programmation au domaine des sciences et technologie, ingénierie et mathématiques (STIM, Lee, 2020; Li, 2020). Néanmoins, bien qu'il existe de nombreux parallèles entre l'apprentissage de la programmation et l'apprentissage des sciences, notamment sur le plan des compétences en résolution de problèmes, pensée critique et modélisation (Khine, 2018), très peu d'études se sont spécifiquement intéressées aux effets de transfert entre ces disciplines (Scherer *et al.*, 2019; Wang 2022), et encore moins chez les jeunes apprenants. En comparaison des mathématiques et des langues, les sciences ont en effet beaucoup moins été étudiées : aucune des dix études incluses dans la revue systématique de Popat et Starkey (2019) ne portait sur le transfert de la programmation vers les sciences et des 105 études faisant partie de la méta-analyse de Scherer *et al.* (2019), seulement deux portaient sur cet aspect et impliquaient de jeunes apprenants. Ces études sont discutées ci-bas.

L'étude de Nugent (2010) a été menée auprès d'élèves du secondaire âgés de 10 à 15 ans et a employé un devis quasi expérimental prétest/posttest pour évaluer l'impact de deux types d'interventions sur l'apprentissage et les attitudes des élèves en matière de STIM. La première intervention mobilisait 147 élèves qui prenaient part à un camp d'été intensif d'une semaine (total de 40 heures) axé sur la programmation à l'aide de la robotique (plateforme LEGO Mindstorms NXT) et les technologies géospatiales, tandis que la deuxième faisait participer 141 élèves et consistait en une introduction de trois heures à ces mêmes technologies. Les résultats ont montré que seule la première intervention de plus grande durée avait un impact significatif sur l'apprentissage en STIM, tel que mesuré à l'aide d'un test maison comportant 37 questions, en comparaison à un groupe de contrôle sans intervention (taille d'effet de g = 0,337, telle que rapportée par Scherer *et al.*, 2019). Cet impact semblait particulièrement important pour les domaines de l'ingénierie et des technologies géospatiales. Par ailleurs, l'intervention qui était plus courte, bien que n'ayant pas d'impact significatif sur l'apprentissage, avait néanmoins un effet positif significatif sur les attitudes et la motivation des élèves envers les STIM.

Une autre étude menée par Park (2015) a pour sa part examiné les effets d'un programme d'investigation scientifique mobilisant la robotique éducative sur la motivation et la réussite en sciences, dans le cadre du programme formel d'enseignement des sciences de la Corée du Sud. Ce programme d'une durée de dix semaines était mis en place auprès d'élèves de 4e et 5e années du primaire et touchait à plusieurs notions variées : le monde animal, le cycle de vie des plantes, la dissolution, le système solaire, les circuits électriques, etc. Le groupe expérimental était composé de 63 élèves, tandis que le groupe de contrôle était constitué de 60 élèves qui voyaient les mêmes notions, mais selon une méthode d'enseignement traditionnelle. Les résultats de l'étude ont montré une amélioration significative de la motivation et de la réussite scolaire dans le groupe expérimental par rapport au groupe témoin. De plus, les chercheurs rapportent que les perceptions des élèves concernant l'utilisation de la robotique comme outil d'apprentissage basé sur l'investigation étaient positives. Ces résultats suggèrent donc que la robotique pourrait présenter une plus-value pour l'apprentissage des élèves dans le cadre de l'éducation scientifique formelle. Scherer et ses collègues (2019) rapportent à ce propos une taille d'effet de g = 0,323.

Il convient de noter que ces deux études ont utilisé la robotique, mettant ainsi en évidence l'utilisation fréquente de cette technologie pour contextualiser les apprentissages en sciences (Ferrada-Ferrada, 2020; Spôlaôr et Benitti, 2017). La robotique s'inscrit directement dans la lignée de la théorie du constructionnisme de Papert (1980) en favorisant l'utilisation de matériel concret pour faciliter l'apprentissage scientifique. Cependant, il est pertinent de noter qu'aux yeux de certains auteurs, cette approche peut parfois négliger ou minimiser l'enseignement de la programmation, voire l'ignorer complètement (Guzdial, 2015). Il s'agit donc d'une limite importante à ces deux études au regard de notre question d'intérêt : elles ont examiné le transfert de la programmation vers les sciences, uniquement à travers l'utilisation de la robotique éducative, qui plus est pour des activités entièrement ancrées en sciences. En conséquence, il devient difficile de réellement isoler les effets de l'apprentissage de la programmation.

Jusqu'à présent, la recherche s'est donc principalement concentrée sur l'identification des similitudes conceptuelles entre les disciplines de la programmation et des sciences. Par exemple, les modèles de Weintrop (2016) et de Barr et Stephenson (2011), abordés précédemment, ont mis en lumière les composantes de la pensée informatique qui émergent de l'apprentissage de la programmation et qui peuvent être réinvesties dans d'autres disciplines, incluant les sciences. Même une récente revue de la littérature menée par Wang et ses collègues (2022), qui a exploré les méthodes d'enseignement

interdisciplinaire entre les sciences et la programmation afin de réfléchir à l'intégration de la pensée informatique aux programmes de sciences, n'a pas abordé la question des éventuels effets positifs de transfert entre l'apprentissage de la programmation et celui des sciences. Cette omission est d'autant plus marquante que l'idée de transfert entre ces domaines est à la base de la décision d'intégrer la pensée informatique dans les programmes de sciences. Ainsi, il apparaît qu'il existe un manque significatif de recherches approfondies concernant les effets de transfert entre l'apprentissage de la programmation et celui des sciences.

La section qui suit poursuit la présentation et la discussion de l'état des connaissances, cette fois pour la discipline des langues.

2.3.3 Transfert des apprentissages de la programmation vers les langues

L'idée selon laquelle il existerait des similitudes entre l'apprentissage de la programmation et l'apprentissage d'une langue n'est pas nouvelle (Bers, 2019; Papert, 1980; Robins, 2003). Ces parallèles reposent principalement sur le fait que ces deux types d'apprentissages font appel à des langages et, par conséquent, sollicitent des compétences similaires, notamment en matière de lecture et d'écriture. Par exemple, l'écriture d'un code et l'écriture d'un texte partagent des caractéristiques fondamentales, telles que la nécessité de respecter des règles grammaticales et syntaxiques pour exprimer des idées de manière cohérente et logique. Dans les deux cas, une séquence incorrecte ou un élément mal placé peut entraîner des erreurs de compréhension ou des dysfonctionnements (Bers, 2019; Knuth, 1992). De plus, à la fois en programmation et en rédaction linguistique, la maîtrise des nuances, comme le choix des mots ou des fonctions, peut avoir un impact significatif sur la précision et l'efficacité de la communication (Bers, 2019).

Récemment, Hassenfeld et Bers (2020) ont présenté un modèle qui élargit la comparaison entre l'apprentissage des langues et celui de la programmation au-delà des similitudes courantes concernant la grammaire et la syntaxe. Leur modèle met en lumière plusieurs points de convergence entre ces deux disciplines. Tout d'abord, ils établissent un parallèle au niveau de la planification et de la pré-écriture. Ils notent que la programmation, tout comme la rédaction d'un texte, nécessite une phase préliminaire de planification. Ensuite, ils mettent en évidence le rôle de la création et de la rédaction, en expliquant que les deux activités impliquent une phase créative où le produit final, qu'il s'agisse de code ou de texte, est généré. Troisièmement, ils soulignent l'importance des tests et de l'évaluation, des étapes nécessaires à la fois pour la maîtrise d'une langue et pour la programmation. L'individu doit en effet prendre du recul

pour évaluer la qualité et l'efficacité de son travail. Enfin, ils discutent du débogage et de la révision comme des étapes où les erreurs sont identifiées et corrigées, que ce soient des erreurs mécaniques en programmation, telles qu'une syntaxe incorrecte, ou des ajustements stylistiques dans la rédaction visant à améliorer la clarté d'un texte. Ce modèle enrichit donc la compréhension des compétences et des activités cognitives partagées entre les disciplines de la programmation et des langues. Il rejoint également plusieurs des compétences identifiées par Robins *et al.* (2003) dans leur modèle sur l'apprentissage de la programmation ainsi que par Brennan et Resnick (2009) dans leur modèle de la pensée informatique.

Cependant, malgré ces parallèles apparents et prometteurs, peu d'études ont réussi à documenter des effets de transfert significatifs et positifs entre l'apprentissage de la programmation et celui des langues (Scherer *et al.*, 2019). En réalité, les résultats concernant le transfert des apprentissages en programmation vers les langues, comme le souligne Scherer et ses collègues dans leur méta-analyse (2019), sont mitigés : parfois positifs, parfois nuls, et parfois même négatifs.

Parmi les études qui révèlent des effets de transfert positifs, celle de Jenkins (2015) mérite une mention particulière. Cette étude a été menée auprès d'élèves du secondaire (8^{e} année; élèves âgés de 12 à 13 ans) au Royaume-Uni et a utilisé Scratch dans le cadre d'un cours de poésie. Un groupe expérimental composé de 13 élèves utilisait Scratch afin de générer automatiquement des poèmes durant trois séances de une heure en salle de classe. Un groupe de contrôle constitué de 14 élèves réalisait des activités plus classiques de création de poèmes pour une durée équivalente de trois heures. Les résultats ont mis en évidence une différence positive et significative au regard de la compréhension des structures de phrases et des familles de mots entre les élèves du groupe expérimental et ceux du groupe de contrôle, selon un devis de type prétest/posttest, avec une taille d'effet de g = 0,435 (Scherer *et al.*, 2019). Une autre étude a également été menée par Clements (1986), cette fois auprès d'élèves de 3^{e} année du primaire (élèves âgés d'environ 8 ans) aux États-Unis. Dans cette étude, un premier groupe de 12 élèves prenait part chaque semaine à des séances de 45 minutes d'apprentissage de la programmation avec le langage LOGO, pour un total de 22 semaines. Un deuxième groupe de 12 élèves réalisait pour sa part des activités classiques d'initiation à la programmation sur un ordinateur pour une durée équivalente. Finalement, un troisième groupe de 12 élèves réalisait des activités traditionnelles en salle de classe qui n'étaient pas en programmation. Un devis comparatif de prétest/posttest a mis en évidence une différence entre les deux groupes les plus contrastés, soit le premier et le troisième; le premier présentant une plus grande amélioration en lecture que le troisième (tel que mesuré par le *Metropolitan Readiness Test*) au terme des interventions. Il

convient cependant de noter que cette différence entre les groupes n'était pas significative. À partir de ces données, Scherer et ses collègues (2019) ont néanmoins calculé une taille d'effet positive moyenne de g = 0,329. (Scherer *et al.*, 2019).

En contraste, certaines études suggèrent un effet de transfert nul ou minime de l'apprentissage de la programmation vers l'apprentissage de la lecture. Par exemple, l'étude de Owston (2009), menée auprès de 311 élèves de niveau primaire (4e année, élèves âgés d'environ 8-9 ans) au Canada, utilisait un environnement de programmation en langage HTML comme outil pour amener les élèves à développer des jeux vidéo dans le cadre d'apprentissages en sciences sociales. Les élèves du groupe expérimental devaient donc apprendre à programmer via ce langage pour être en mesure de développer leur jeu, tandis que les élèves du groupe de contrôle suivaient simplement les activités régulières de classe. Les activités du groupe expérimental étaient réalisées en dehors des heures de classe (en plus des activités régulières), à raison d'une heure par semaine durant dix semaines. Les chercheurs mesuraient également les compétences en langues (*Group Reading Assessment and Diagnostic Evaluation* et *Test of Written Language*) des élèves avant et après ces activités. Aucune différence significative en vocabulaire et en compréhension de texte n'a été mesurée pour le groupe expérimental comparativement au groupe de contrôle, tandis qu'une légère différence positive fut observée pour la rédaction de texte des élèves qui ont appris à programmer (Owtson, 2009); associée à une taille d'effet presque négligeable de g = 0,014 (Scherer *et al.*, 2019).

Bien que cela puisse paraître étonnant, il existe également des données indiquant un effet de transfert négatif de l'apprentissage de la programmation vers les compétences en lecture. Par exemple, l'étude de Seidman (1981) menée aux États-Unis s'est intéressée aux effets de transfert de la programmation vers les compétences en lecture d'élèves de niveau primaire. Dans cette étude, un groupe expérimental composé de 20 élèves apprenaient à programmer avec le langage LOGO, tandis qu'un groupe de contrôle de 21 élèves ne prenait part à aucune intervention particulière. Les résultats indiquent que seuls les participants du groupe de contrôle présentaient une amélioration de leurs compétences en lecture, comparativement au groupe expérimental. Une taille d'effet négative de g = −0,539 a été rapportée par Scherer et ses collègues (2019). Lehrer et Randle (1987) ont aussi constaté un effet négatif sur les compétences en écriture (composition) d'élèves de 1re année du primaire (âgés de 6-7 ans), associé à une taille d'effet de g = -0,469 (Scherer *et al.*, 2019). Cette étude impliquait un groupe expérimental (13 élèves) qui apprenait aussi à programmer avec LOGO, à raison de 2 séances d'environ 20 minutes par semaine

durant cinq mois, tandis qu'un groupe de contrôle (13 élèves) employait un autre logiciel visant spécifiquement la composition et la résolution de problèmes (une seule séance de 45 minutes par semaine durant 5 mois). Les deux groupes étaient comparés à une condition sans traitement (13 élèves). Les résultats indiquent qu'il n'y avait pas de différence significative entre les groupes au regard de leurs compétences en composition, entre le prétest et le posttest. Plus récemment, l'étude de Hayes et Stewart (2016), dont la méthodologie a été présentée en détail dans la section portant sur les mathématiques, mesurait également les effets de transfert pour d'autres apprentissages, dont les langues. Celle-ci a révélé un effet de transfert négatif des apprentissages en programmation réalisés avec le logiciel Scratch sur les apprentissages en orthographe (g = -1,033).

Des 105 études incluses dans la méta-analyse de Scherer *et al.* (2019), seulement onze études ont examiné les effets de transfert de la programmation vers les langues. Bien qu'individuellement certaines des études observent un effet de transfert significatif, les résultats de la méta-analyse englobant l'ensemble des études s'étant intéressées à la question entre 1965 et 2017 indiquent qu'il ne semble pas possible de conclure à un effet de transfert significatif de l'apprentissage de la programmation vers les langues (g = -0,02). Une explication possible de ce résultat pourrait être en lien avec la distance importante entre les compétences spécifiques évaluées pour le domaine des langues et celles développées par la programmation. Cette idée est mise de l'avant dans plusieurs des typologies du transfert des apprentissages (p. ex., Barnett et Ceci, 2002; Ellis, 1979; Mayer, 1975). Des conclusions similaires ont d'ailleurs été tirées dans la revue de la littérature menée par Sala et Gobet (2017a), qui examinait pour sa part les effets du transfert de l'apprentissage de la musique vers l'apprentissage des langues (g = 0,07), deux disciplines pouvant également être considérées comme assez éloignées l'une de l'autre.

D'autres hypothèses explicatives sont également avancées dans la littérature quant à cette absence de transfert. Par exemple, une étude d'Ivanova *et al.* (2020) a utilisé l'imagerie par résonance magnétique fonctionnelle (IRMf) afin d'identifier les réseaux neuronaux impliqués dans la lecture et la compréhension du code en programmation. Cette étude montre que la lecture et la compréhension du code informatique sollicitent principalement des régions bilatérales du cerveau qui sont habituellement mobilisées durant la résolution de problèmes en mathématiques, en logique ainsi que pour des tâches exécutives, et ne sollicitent pas celles spécifiquement dédiées au traitement du langage (latéralisées dans l'hémisphère gauche. Cela pourrait donc contribuer à expliquer pourquoi le transfert entre la programmation et les langues s'avère difficile étant donné que chacune mobilise des régions cérébrales distinctes.

***

Cet état des connaissances sur les effets de transfert de la programmation vers les disciplines scolaires que sont les mathématiques, les sciences et les langues a principalement été réalisé à travers la méta-analyse de Scherer et ses collègues (2019). Au terme de celle-ci, il est d'abord possible de constater que les effets de transfert entre la programmation et les disciplines scolaires sont le plus souvent inférés à partir de mesures associées à la performance scolaire des élèves. Les auteurs de cette méta-analyse font également émerger certains constats plus généraux. Il semble d'abord que le transfert associé à l'apprentissage de la programmation soit globalement positif avec une taille d'effet modérée (g = 0,49), ce qui suggère que l'apprentissage de la programmation pourrait présenter certains bénéfices sur le plan cognitif. Or, tel que cela a été mentionné plus tôt, cette méta-analyse ne portait pas uniquement sur les effets de transfert vers les disciplines scolaires (qui étaient catégorisées comme étant du transfert éloigné); elle s'intéressait également aux effets de transfert sur d'autres habiletés cognitives susceptibles de partager davantage de similarités avec la programmation (qui étaient majoritairement rassemblées sous la catégorie du transfert proche). Lorsque ces catégories sont départagées, il ressort que le transfert proche est globalement positif et est associé à une taille d'effet importante (g = 0,75). Par ailleurs, le transfert éloigné qui englobe notamment les disciplines scolaires a quant à lui un effet positif modéré (g = 0,47), les mathématiques étant la discipline faisant partie du transfert éloigné apparaissant comme étant la moins éloignée (g = 0,57).

Néanmoins, cet état des connaissances a également révélé certaines limites méthodologiques importantes au sein de ce corpus de recherche, qu'il convient d'approfondir. La prochaine section poursuit donc cet objectif et aspire à mieux circonscrire le vide dans les connaissances qui demeure quant aux effets de transfert de la programmation vers les disciplines scolaires.

2.3.4 Principales limites méthodologiques relevées au sein des recherches existantes

À la lumière des études examinées et discutées précédemment, il apparaît que quatre limites méthodologiques importantes se dégagent. Par conséquent, il apparaît pertinent de mieux comprendre les inconvénients qui y sont associés, et d'identifier des pistes permettant de les surmonter dans le cadre de recherches futures.

L'une des principales limites mises en évidence dans la littérature (Popat et Starkey, 2019) est l'absence fréquente de comparaison avec un groupe de contrôle. Par exemple, sur les 708 études incluses à la suite

de la recherche documentaire menée par Scherer *et al.* (2019) pour leur méta-analyse, 335 études ont été exclues pour diverses raisons, dont l'absence d'un groupe de contrôle. Cette méta-analyse repose ainsi exclusivement sur des études ayant utilisé un groupe de contrôle, bien que ce groupe de contrôle soit parfois passif. Or, les groupes de contrôle passifs présentent un risque plus élevé de biaiser les résultats, car ils ont tendance à surestimer l'effet de l'intervention. À cet égard, Scherer *et al.* (2019) soulignent à la fin de leur méta-analyse que les études ayant utilisé un groupe de contrôle passif (p. ex., l'étude de Hayes et Stewart, 2016) affichaient des tailles d'effet plus importantes que celles ayant utilisé un groupe de contrôle actif, laissant ainsi entendre que l'absence d'un groupe de contrôle actif peut mener à une inflation de l'impact de l'intervention. En plus des études intégrées à la méta-analyse de Scherer *et al.* (2019), d'autres recherches ne comportant pas de groupe de contrôle se sont intéressées au transfert de la programmation vers d'autres disciplines (p. ex., Ke, 2014). Cependant, l'absence d'un groupe de contrôle rend impossible la détermination de l'origine des changements observés en termes de transfert des apprentissages. Un groupe de contrôle actif apparaît donc essentiel afin d'évaluer la plus-value de l'apprentissage de la programmation en ce qui concerne le transfert des apprentissages vers d'autres domaines scolaires (Hickmott *et al.*, 2017).

Une seconde limite relevée par plusieurs auteurs concerne le fait que plusieurs études ont été menées auprès d'échantillons de petite taille. À cet égard, Scherer et ses collègues (2019) soulignent que plusieurs des recherches incluses dans leur méta-analyse présentent des tailles d'échantillons relativement petites pour les groupes de traitement et de contrôle. Or, cette limite peut avoir des implications importantes au regard de la généralisabilité des effets observés. En effet, il est plus probable de détecter des effets de grande ampleur lorsque les échantillons sont petits, car ces effets peuvent être suffisamment forts pour émerger malgré la variabilité inhérente à un petit échantillon. En revanche, des effets plus faibles ou de taille moyenne peuvent ne pas être détectés, car la variabilité au sein d'un petit échantillon peut masquer ces effets plus subtils. Cependant, il est important de noter que la détection d'un effet de grande ampleur dans un petit échantillon peut également être associée à une surestimation de la taille de cet effet. En d'autres termes, la taille de l'effet peut apparaître plus grande qu'elle ne l'est réellement dans la population globale, en raison de l'erreur d'échantillonnage accrue associée à des échantillons plus petits. Ce phénomène est souvent référé comme l'inflation de la taille de l'effet (Braver, 2010). Par conséquent, Scherer et ses collègues (2019) appellent à faire preuve de prudence dans l'interprétation des tailles d'effet de leur propre méta-analyse et recommandent que des études futures soit menées auprès

d'échantillons plus grands afin d'améliorer la fiabilité et la généralisabilité des résultats concernant les effets de transfert associés à l'apprentissage de la programmation.

En troisième lieu, une autre limite importante réside dans la courte durée des interventions mises en place dans de nombreuses études. Par exemple, Scherer et ses collègues (2019) révèlent que la durée moyenne des interventions au sein de leur méta-analyse est de seulement 45 minutes. Or, des interventions de courte durée risquent de ne pas offrir suffisamment de temps pour qu'un apprentissage en profondeur et significatif puisse se réaliser (Liu et Pásztor, 2022). Pourtant, les compétences complexes, telles que la programmation, requièrent souvent un investissement temporel considérable pour être pleinement maîtrisées et pour que les apprenants puissent transférer efficacement ces compétences vers d'autres domaines (Liu et Pásztor, 2022; Robins, 2010). Afin de pallier cette limite, plusieurs auteurs (Popat et Starkey, 2019; Scherer *et al.*, 2019) recommandent d'adopter des devis longitudinaux qui permettraient d'étudier les effets de transfert à travers le temps. À cet égard, des interventions de plus longue durée permettraient de suivre les participants sur une période prolongée, offrant ainsi une opportunité pour évaluer la stabilité des effets de transfert au fil du temps. Des études longitudinales seraient donc plus susceptibles de permettre une analyse approfondie des mécanismes de transfert et de la manière dont ils peuvent évoluer avec une exposition prolongée à l'apprentissage de la programmation.

Finalement, une quatrième limite notable concerne le fait que plusieurs études ont évalué les effets d'interventions où l'apprentissage de la programmation était intégré au sein d'autres disciplines (par exemple, une intervention dans laquelle les bases de la programmation étaient enseignées dans le contexte d'une activité de science). Cela soulève un problème potentiel de contamination des résultats, car il devient plus probable de mesurer des effets d'entraînement direct de cette discipline plutôt que de mesurer des effets de transfert résultant de l'apprentissage de la programmation vers cette discipline (Melby-Lervåg *et al.*, 2016). Qui plus est, la quantité de programmation que les élèves acquièrent est alors limitée à ce qui est strictement nécessaire pour cette discipline (Gudzial, 2015). Afin d'isoler et de mesurer les effets de transfert spécifiques à la programmation, il apparaît donc essentiel de concevoir des devis permettant de faire la distinction entre l'effet propre à la programmation et les effets qui pourraient découler d'autres types d'activités incluses dans l'intervention. Le Tableau 2.8 propose une synthèse de ces principales limites et des inconvénients qui leur sont associés.

Tableau 2.8 Synthèse des principales limites identifiées au regard de la littérature existante sur le transfert des apprentissages de la programmation vers les disciplines scolaires

Limites	Inconvénients associés
Absence d'un groupe de contrôle ou groupe de contrôle passif	• Il est difficile de déterminer si les résultats observés dans le groupe expérimental sont réellement dus à l'intervention ou si d'autres facteurs ont influencé les résultats.
Petite taille des échantillons	• Il est moins probable de détecter des effets de transfert plus faibles en raison de la variabilité au sein d'un petit échantillon. • Par ailleurs, une petite taille d'échantillon augmente l'erreur d'échantillonnage, ce qui favorise l'inflation du résultat obtenu. Autrement dit, le niveau de transfert réel aura des chances d'être inférieur à celui qui aura été identifié.
Interventions de courte durée	• Les interventions de courte durée sont moins susceptibles de mener à des apprentissages en programmation, ce qui diminue les chances de détecter des effets de transfert. • Les interventions de courte durée ne permettent pas d'observer des effets à long terme et la persistance des effets dans le temps.
Interventions dans lesquelles l'apprentissage de la programmation est intégré au sein des disciplines	• Il y a un effet de contamination; il est plus probable de mesurer des effets d'entraînement direct et non des effets de transfert découlant de l'apprentissage de la programmation.

2.4 Hypothèses de recherche

La question de recherche générale sous-tendant ce projet est la suivante : Quels sont les effets de transfert associés à l'apprentissage de la programmation chez les jeunes apprenants sur les disciplines scolaires traditionnelles ? L'état des connaissances scientifiques exposé dans ce cadre théorique a permis de mettre en évidence des résultats mitigés au regard de cette question. Il semble que le transfert des apprentissages de la programmation ait à ce jour été principalement documenté pour les disciplines des mathématiques, des sciences et des langues, les conclusions des études variant selon la discipline, suggérant ainsi une grande hétérogénéité des résultats. De plus, de nombreuses lacunes méthodologiques ont été identifiées dans les études existantes sur ce sujet. L'incapacité à obtenir une perspective claire sur la faisabilité du transfert des apprentissages en programmation vers d'autres disciplines scolaires suggère qu'il subsiste

un vide important dans les connaissances, soulignant ainsi le besoin de mener davantage de recherches sur le sujet.

Pour combler ce vide et dans le but de dépasser les limites précédemment identifiées, cette recherche doctorale prévoit employer un devis comparatif longitudinal auprès de jeunes apprenants afin d'approfondir la compréhension des effets de transfert de la programmation vers les trois disciplines pour lesquelles un état des connaissances a été réalisé. Malgré leurs limites respectives, les recherches examinées dans ce cadre théorique ont établi des bases nous permettant de formuler des hypothèses de recherche distinctes, au regard de chacune des disciplines. Ces hypothèses sont présentées ci-bas.

Hypothèse 1 - Mathématiques

Considérant que :

- les mathématiques représentent la discipline qui a été la plus étudiée au regard du transfert possible depuis la programmation;
- des auteurs ont identifié de nombreux liens disciplinaires entre la programmation et les mathématiques, via le développement de la pensée informatique (Barr et Stephenson, 2011; Weintrop *et al.*, 2016);
- bien que le transfert entre les mathématiques et la programmation puisse être considéré comme étant éloigné (Barnett et Ceci, 2002), les mathématiques seraient, de l'avis de plusieurs auteurs (voir p. ex., Popat et Starkey, 2019), la discipline scolaire la plus « proche » de la programmation, ce qui augmente la probabilité d'un transfert entre celles-ci;
- il existe une vaste littérature de recherche qui tend à indiquer des effets positifs de l'apprentissage de la programmation sur les apprentissages en mathématiques;
- les conclusions de la méta-analyse menée par Scherer et ses collègues (2019) ont mis en lumière un effet de transfert positif et significatif d'une ampleur moyenne de 0,57 :

nous faisons l'hypothèse (H1) que des *effets de transfert positifs* seront observés entre l'apprentissage de la programmation et la performance scolaire en mathématiques. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à de *meilleurs résultats scolaires en mathématiques*, comparativement à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

Hypothèse 2 - Sciences

Considérant que :

- il existe peu d'études portant sur les effets de transfert de l'apprentissage de la programmation vers les sciences;
- le transfert entre les sciences et la programmation est généralement considéré comme étant éloigné (Barnett et Ceci, 2002);
- des auteurs ont identifié certains liens disciplinaires entre la programmation et les sciences, via le développement de la pensée informatique (Barr et Stephenson, 2011; Weintrop *et al.*, 2016);
- les quelques études existantes, dont celles faisant partie de la méta-analyse de Scherer *et al.* (2019), tendent à indiquer des effets positifs, mais de petite taille, de l'apprentissage de la programmation sur les apprentissages en sciences;

nous faisons l'hypothèse (H2) que des *effets de transfert positifs, mais de moindre ampleur*, seront observés entre l'apprentissage de la programmation et la performance scolaire en sciences. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à des résultats scolaires en sciences *légèrement supérieurs* à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

Hypothèse 3 - Français

Considérant que :

- des auteurs ont identifié quelques liens disciplinaires entre la programmation et les langues, via le développement de la pensée informatique (Barr et Stephenson, 2011), mais ceux-ci sont beaucoup moins nombreux que pour les sciences et les mathématiques;
- le domaine des langues apparaît possiblement plus « éloigné » de la programmation, ce qui diminue les probabilités de transfert (Barnett et Ceci, 2002; Scherer *et al.*, 2019);
- il existe un certain nombre d'études portant sur les effets de transfert de l'apprentissage de la programmation vers les langues, mais les résultats de celles-ci sont plus hétérogènes et apparaissent moins concluants;
- les conclusions de la méta-analyse menée par Scherer et ses collègues (2019) ont mis en lumière un effet de transfert nul d'une ampleur de - 0,02;

- les résultats d'une étude récente (Ivanova *et al.*, 2020) suggèrent que les langues et la programmation mobilisent des régions cérébrales distinctes;

nous faisons l'hypothèse (H3) qu'il n'y aura *pas d'effets de transfert positifs* entre l'apprentissage de la programmation et la performance scolaire en français. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à des résultats scolaires en français qui seront *similaires* à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

CHAPITRE 3
MÉTHODOLOGIE

Cette recherche a pour objectif de vérifier quels sont les effets de l'apprentissage de la programmation sur les disciplines scolaires traditionnelles, soit les mathématiques, les sciences et le français. Dans un premier temps, ce chapitre présente le devis général du projet de recherche afin de permettre au lecteur d'avoir une vue d'ensemble du déroulement du projet. Puis, le choix d'un devis longitudinal comparatif et de nature observationnelle est justifié. Les points suivants sont ensuite abordés : le choix du terrain de recherche (critères de sélection, description des concentrations), la constitution de l'échantillon (obtention des données et présentation de l'échantillon) et la nature des données utilisées. Les analyses statistiques envisagées sont enfin présentées et discutées, notamment au regard du choix des analyses de trajectoires latentes et des étapes de prétraitements des données. Finalement, les considérations éthiques d'usage (consentement et confidentialité) sont abordées.

3.1 Devis général du projet de recherche

Cette section vise à fournir un aperçu du déroulement général du projet de recherche. Chacun des aspects du projet sera par la suite détaillé dans les sections qui suivent. Afin de répondre à la question de recherche et de vérifier les hypothèses avancées, un devis longitudinal comparatif a été adopté. La Figure 3.1 présente ce devis. Ce dernier consiste en l'étude d'une cohorte unique de 440 élèves fréquentant une école secondaire de Montréal. Cette cohorte a été suivie sur une période de cinq ans, soit de la 1re année du secondaire (T1) à la 5e année (T5); ce devis comporte donc cinq temps de mesure correspondant à chaque année du parcours scolaire secondaire.

Dès le début de ce parcours, chaque élève est invité par l'établissement scolaire à choisir obligatoirement une concentration scolaire parmi les trois suivantes : Programmation (en jaune dans la figure), Arts (en vert) et Sports (en bleu). Ces concentrations sont mises en place pour les trois premières années du parcours scolaire des élèves (T1 à T3), durant lesquelles l'ensemble des élèves suivent le même programme d'études, à l'exception de leur concentration. À partir de la 4e année (T4), le cheminement par concentrations prend fin, et ce jusqu'à la fin du secondaire (T5). Cette spécificité est également représentée sur la figure par l'utilisation d'un ombragé plus pâle aux temps 4 et 5. Il est donc possible de comparer dans le temps (T1 à T5) les effets de chacune des concentrations sur les résultats scolaires des

élèves pour les autres disciplines scolaires, les concentrations Arts et Sports agissant à titre de groupes de contrôle actifs. Comme illustré à la Figure 3.1, les données utilisées pour mesurer les effets des concentrations sur les disciplines scolaires sont les résultats scolaires individuels extraits des bulletins de fin d'année pour les mathématiques, les sciences et le français, et ce, pour les cinq années du secondaire. Tel que cela a été constaté au terme de la discussion de la méta-analyse de Scherer et ses collègues (2019), il semble que les résultats scolaires soient le plus souvent employés afin de s'intéresser aux effets de transfert d'un contexte à un autre (voir par exemple l'étude de Hayes et Stewart, 2016). En plus des considérations théoriques avancées dans le chapitre précédent, le choix de ces disciplines s'avère également pertinent d'un point de vue méthodologique puisque ce sont celles qui occupent la place la plus importante dans le programme d'études des élèves (cet aspect est détaillé à la section 3.2.2). Ce devis permet ainsi une analyse longitudinale comparative des résultats scolaires en fonction de la concentration choisie.

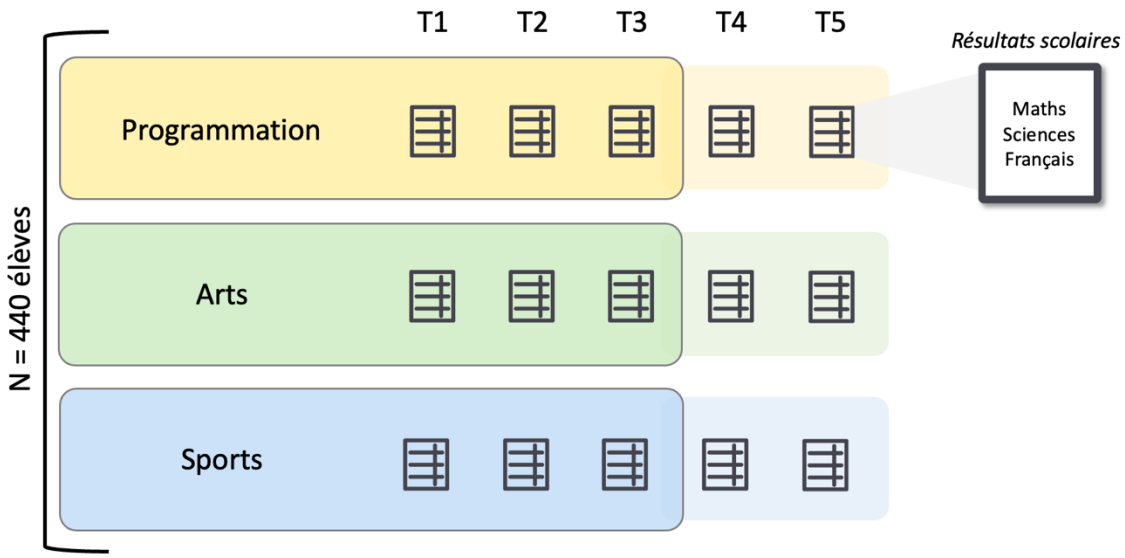

Figure 3.1 Devis général du projet de recherche.

Ce projet de recherche s'appuie donc sur un devis observationnel en grappes (*clustered observational study*; Page, 2020) et mobilise des données secondaires. Ces données sont appelées « secondaires », car elles sont initialement collectées à d'autres fins que celle poursuivie par le projet de recherche, mais peuvent être réutilisées afin de répondre à des questions de recherche originales (Dionne et Fleuret, 2016). Ce choix méthodologique s'explique principalement par le fait que ce type de devis soutient des

méthodologies longitudinales, s'échelonnant sur cinq ans dans le cadre de ce projet, et comparative, qui fait dans ce cas-ci intervenir deux groupes de contrôle actifs.

Les études observationnelles sont un type de recherche quantitative qui visent à observer les phénomènes à l'étude sans manipuler de variables (Rubin, 2007). Ce type de recherche nécessite d'« extraire » des données qui sont issues de situations écologiques (Karsenti et Savoie-Zajc, 2018). L'exploitation de ces données présente l'avantage de réduire les coûts et le temps que nécessiterait la réalisation d'une nouvelle collecte de données, tout en offrant la possibilité d'observer un contexte de manière authentique (Turgeon et Bernatchez, 2009), ce qui s'avère particulièrement intéressant dans le cadre d'un projet doctoral pour lequel les ressources et le temps sont limités. Les études observationnelles peuvent être de nature comparative, c'est-à-dire qu'elles offrent la possibilité de comparer deux groupes ou plus au regard d'une ou plusieurs variables d'intérêt (Page, 2020).

Le terme « grappe » réfère généralement à un groupe d'unités (dans le cas des études en éducation, il s'agit le plus souvent de participants) qui partagent des caractéristiques communes et qui sont étudiées ensemble au sein de ce groupe. Dans le cas de la présente recherche, les grappes réfèrent donc aux concentrations. Dans un devis observationnel comparatif en grappes, l'attribution du traitement n'est pas aléatoire; il se fait au niveau de la grappe plutôt qu'au niveau individuel. Par exemple, dans un contexte éducatif, une grappe peut être une école, une classe ou, comme cela est le cas dans ce projet, une concentration scolaire. Les unités sont donc le plus souvent les élèves au sein de ces grappes (Page, 2020). Les analyses demeurent centrées sur les résultats au niveau des unités, mais prennent en compte la structure de la grappe (Page, 2020).

Ce type de devis est particulièrement utile pour les études qui cherchent à comprendre les effets contextuels ou les effets d'interventions qui sont naturellement administrées au niveau de la grappe (Page, 2020). Il soutient également des méthodologies longitudinales et comparatives, ce qui le rend idéal pour des études s'étendant sur une période prolongée et impliquant plusieurs groupes, ce qui est le cas de ce projet. En somme, en raison du regroupement des données en grappes, il devient possible de les comparer. Dans la mesure où ces grappes sont équivalentes et se distinguent uniquement au regard de la concentration, le devis permet de distinguer les effets respectifs de chacune de ces grappes (ici les concentrations) à travers le temps.

3.2 Choix du terrain de recherche

Le choix du terrain d'étude est un élément crucial dans la conception d'une étude observationnelle (Rubin, 2007). Ce choix détermine en grande partie la qualité et la validité des données recueillies, ainsi que la capacité à minimiser les biais et à contrôler au mieux les variables confondantes. Rubin (2007) souligne également que les études observationnelles devraient être conceptualisées le plus possible comme des études expérimentales randomisées, pour lesquelles il est nécessaire d'adopter une posture déductive plutôt qu'inductive. Ainsi, le devis, le choix du terrain et des analyses envisagées doivent être déterminés avant de considérer toute donnée, dans le but de réaliser les principaux choix méthodologiques de manière objective, sans qu'ils soient teintés par la nature des données collectées.

Par ailleurs, la réalisation d'un projet de recherche portant sur les effets de transfert de l'apprentissage de la programmation vers les disciplines scolaires traditionnelles comporte plusieurs contraintes logistiques qui complexifient le choix d'un terrain approprié.

D'abord, le manque de matériel adapté à l'enseignement de la programmation dans de nombreux milieux scolaires limite la mise en place d'un programme efficace et restreint le bassin d'écoles susceptibles de prendre part au projet. Qui plus est, étant donné que l'introduction de la programmation dans les milieux scolaires est relativement récente, peu d'enseignants sont adéquatement formés. De ce fait, trouver un terrain où les enseignants détiennent les compétences et l'expérience nécessaires pour enseigner la programmation peut s'avérer difficile. De plus, l'absence de recommandations gouvernementales relativement à l'intégration de la programmation en milieu scolaire ajoute un niveau de complexité supplémentaire, car le choix d'introduire ou non la programmation revient à chaque milieu et la façon de le faire également. Finalement, la nature longitudinale de l'étude qui exige un suivi continu des élèves sur plusieurs années représente également un défi en termes de cohérence, de suivi et de gestion des données. Ces contraintes, combinées, peuvent contribuer à expliquer pourquoi un projet de recherche de cette nature n'a encore jamais été mené.

Dans le cadre de cette étude, nous avons stratégiquement fait le choix d'approcher une école secondaire qui comporte une concentration scolaire spécifiquement axée sur l'apprentissage de la programmation. Ce choix nous permet de surmonter bon nombre des défis logistiques mentionnés ci-dessus. Il s'agit en effet d'une rare opportunité d'obtenir des données écologiques permettant de documenter le potentiel de l'apprentissage de la programmation en matière de transfert vers les apprentissages des autres

disciplines scolaires, et ce, d'une manière longitudinale et comparative. Les critères qui ont été employés pour sélectionner cette école sont détaillés ci-bas.

3.2.1 Critères de sélection du terrain de recherche

Trois critères de sélection centraux visant à répondre aux principales limites des études existantes ont guidé le choix du terrain de recherche. Ces critères visaient à sélectionner une école :

1) de bonne taille, permettant de constituer un échantillon suffisamment robuste;
2) où un programme d'apprentissage de la programmation est mis en place depuis plusieurs années, permettant de réaliser un suivi longitudinal;
3) où le programme d'études offre la possibilité d'avoir un groupe de contrôle actif.

L'école qui a été approchée est une école secondaire privée de Montréal à laquelle sont inscrits plus de 2000 élèves, qui propose depuis 2014 une concentration scolaire portant spécifiquement sur l'apprentissage de la programmation. Il s'agit d'une école secondaire d'une taille assez importante, de sorte que chaque cohorte d'élèves est constituée d'environ 450 élèves. Ce nombre d'élèves répond à un premier critère relatif à la taille de l'échantillon, que nous voulions suffisamment importante afin d'assurer une puissance statistique intéressante au regard des analyses statistiques envisagées (Fan et *al.*, 2003; Soper, 2023). De plus, considérant que cette concentration scolaire est existante depuis 2014, il est possible de suivre les possibles effets de transfert qui pourraient découler de celle-ci sur l'ensemble du parcours scolaire au secondaire d'une cohorte d'élèves (de 2014 à 2019), ce qui répondait à notre deuxième critère de sélection.

Notre troisième critère de sélection concernait la nécessité de pouvoir inclure un groupe de contrôle actif afin de comparer les effets observés pour la concentration Programmation à un contrôle équivalent. L'école sélectionnée permettait de répondre à ce critère, car elle propose plusieurs concentrations aux élèves qui la fréquentent et toutes ces concentrations sont « équivalentes » en ce qui concerne le temps de classe alloué (ces concentrations sont présentées plus en détail à la section 3.2.2.1). En effet, tel que cela a été mentionné plus tôt, chaque élève choisit dès le début de son parcours une concentration parmi les options Programmation, Arts ou Sports. Chaque concentration est associée à un même nombre d'heures de cours par semaine, permettant ainsi une comparaison entre la concentration d'intérêt (Programmation) et les autres concentrations (Arts et Sports) qui agissent de ce fait comme conditions de contrôle actives. Ce terrain de collecte permet donc de mettre en place un devis méthodologique

rigoureux et contrôlé. Le choix de cette école permet ainsi de respecter les principales et plus récentes recommandations mises de l'avant par différents auteurs en didactique de la programmation (Popat et Starkey, 2019; Scherer *et al.*, 2019), soit de prioriser l'emploi d'un devis de recherche comprenant des interventions en programmation qui sont de longue durée, de prévoir plusieurs temps de collecte sur une période prolongée, et d'inclure un groupe de contrôle actif.

Qui plus est, d'autres éléments allant au-delà de ces critères rendent le choix de ce terrain encore plus intéressant. D'une part, étant donné que le cheminement par concentrations scolaires n'est mis en place que durant les trois premières années du parcours scolaire des élèves, cela fait en sorte que le cursus scolaire de l'ensemble des élèves redevient plus homogène pour les 4e et 5e années. Il est donc également possible d'observer les effets différés associés aux différentes concentrations scolaires aux temps 4 et 5. Le programme d'études est présenté plus en détail à la section 3.2.2 ci-bas. D'autre part, la majorité des évaluations qui sont réalisées pour les disciplines des mathématiques, des sciences et du français, sont standardisées et uniformisées pour toute l'école, ce qui limite le bruit qui aurait pu être associé aux évaluations de la performance scolaire des élèves. Cet aspect sera élaboré davantage à la section 3.4 portant sur la nature des données secondaires utilisées.

Par ailleurs, la formation du personnel enseignant en programmation constitue un enjeu de taille (Brown, 2014; Royal Society, 2017). Non seulement il est difficile de trouver des enseignants suffisamment formés pour introduire cette discipline auprès des élèves, mais il a également été démontré que des enseignants qui s'y connaissent peu en programmation sont moins susceptibles de mener à des apprentissages significatifs en programmation et sont, à l'inverse, plus susceptibles de contribuer à un désengagement des élèves envers cette discipline (Schulte, 2012; Voolsted, 2007). Or, les deux enseignants qui chapeautent la concentration Programmation de l'école sélectionnée ont été rigoureusement formés et sont aussi responsables du programme parascolaire en programmation qui existe à l'école depuis 2010. Ces enseignants détiennent donc une expertise suffisante relativement à l'enseignement de la programmation, ce qui laisse supposer que le programme d'enseignement associé à cette concentration a été bien réfléchi et qu'il intègre de bonnes pratiques pédagogiques.

La section qui suit présente de manière plus détaillée le programme d'études de cette école secondaire afin d'offrir une vue d'ensemble de son fonctionnement et de son contenu, permettant ainsi de mieux situer la place des concentrations au sein de ce dernier.

3.2.2 Description du programme d'études

Le programme d'études de l'école secondaire sélectionnée se décline selon un cycle de 18 jours. Le programme des élèves de 1re et 2e secondaire met un accent particulier sur les disciplines que sont le français, les mathématiques, et les sciences et technologies. Un total de 20 périodes d'enseignement sont allouées au français, 15 aux mathématiques, et 10 aux sciences, par cycle de 18 jours. Parallèlement à ces disciplines centrales, chacune des concentrations est associée à un total de 5 périodes d'enseignement par cycle. De manière complémentaire et dans une moindre mesure, le programme englobe également des cours en éducation physique et à la santé (5 périodes), en éthique et culture religieuse (5 périodes), ainsi qu'en géographie (5 périodes), histoire (5 périodes) et éducation à la citoyenneté (5 périodes).

Pour ce qui est du programme d'études des élèves de 3e secondaire, les disciplines du français, des mathématiques et des sciences comprennent toutes 15 périodes. Chacune des trois concentrations se voit toujours allouer de manière équivalente un total de 5 périodes par cycle. Les autres matières complétant le programme d'études sont l'anglais (10 périodes), l'espagnol (10 périodes), l'éducation physique et à la santé (5 périodes), l'histoire du Québec et du Canada (10 périodes) et les arts plastiques (5 périodes).

En 4e secondaire, le français et les mathématiques continuent d'occuper 15 périodes tandis que les sciences occupent désormais une place plus importante et comptent un total de 20 périodes par cycle. À ce stade, il est à noter que le cheminement par concentrations n'est plus actif. Les autres matières qui complètent le programme gagnent donc en importance : anglais (10 périodes), espagnol (5 périodes), éducation physique et à la santé (4 périodes), éthique et culture religieuse (8 périodes) et histoire du Québec et du Canada (10 périodes).

Enfin, le programme d'études de 5e secondaire comporte 15 périodes de français et de mathématiques. À ce niveau, les élèves ont également la possibilité de choisir un profil « sciences pures », qui comprend des cours en chimie (12 périodes) et en physique (12 périodes), ou un profil « sciences humaines », comprenant des cours en entrepreneuriat et monde des affaires (12 périodes) ainsi qu'en histoire du XXe siècle (12 périodes). Outre ces disciplines, le programme d'études compte aussi des cours en anglais (10 périodes), en éducation physique et à la santé (6 périodes), en éthique et culture religieuse (5 périodes), en éducation financière (5 périodes), et en monde contemporain (5 périodes).

Étant donné leur importance dans ce projet, la section qui suit présente plus en détail chacune des concentrations.

3.2.2.1 Description des concentrations

3.2.2.1.1 Concentration Programmation

La concentration Programmation propose une introduction approfondie aux concepts de base et compétences en programmation, principalement à l'aide de langages visuels. Le cursus est conçu pour progresser de manière graduelle via une démarche d'apprentissage par projets, permettant aux élèves d'acquérir des compétences pratiques et théoriques qui vont au-delà de la simple écriture de code. Le profil de sortie des finissants de la concentration Programmation prévoit qu'ils auront développé une bonne maîtrise et une autonomie au regard des concepts essentiels en programmation, développé les habiletés nécessaires pour réaliser leur propre programme (tant en langage visuel que textuel) et créer des applications web à l'aide de programmes, appliqué la programmation à différents contextes (création de logiciels, dessin 3D, robotique, utilisation de microprocesseurs, etc.), développé leur esprit critique à l'égard d'enjeux contemporains liés aux technologies et au numérique et participé à des activités compétitives de programmation.

3.2.2.1.2 Concentration Arts

La concentration Arts prévoit que les élèves ont la possibilité de choisir entre différentes sous-disciplines, notamment la musique, où ils peuvent se spécialiser en cordes, vents ou percussions; la danse classique; ou les arts plastiques et multimédias. Chaque sous-discipline est structurée de manière à offrir une formation complète qui englobe à la fois la technique et la théorie, favorisant ainsi le développement de compétences artistiques et créatives diversifiées. Les élèves sont ainsi amenés à explorer plusieurs médiums, à vivre le processus de création, à expérimenter l'art numérique, à découvrir l'histoire de l'art, et à réaliser des créations personnelles. Le profil de sortie des finissants de la concentration Arts implique qu'ils auront développé une bonne connaissance des métiers liés aux arts, exploré des enjeux contemporains à travers différents médiums, réalisé des expositions ou des concerts, créé des œuvres en s'appropriant le processus de création, et développé leur autonomie et leur esprit critique en produisant des œuvres matures et en partageant leur vision.

3.2.2.1.3 Concentration Sports

La concentration Sports ne se limite pas à l'aspect physique de l'activité; elle inclut également un volet théorique centré sur les bases et les stratégies de diverses disciplines sportives. Ce double objectif permet aux élèves de développer à la fois leurs capacités physiques et leur compréhension stratégique des sports, tout en faisant la promotion de l'importance du travail d'équipe et de la discipline personnelle. Plus précisément, la concentration vise à développer des habiletés physiques inhérentes à toute pratique sportive; à approfondir les connaissances sur la condition physique, le corps humain et la santé de chaque élève. Pour ce faire, les élèves sont amenés à travailler une multitude de disciplines sportives individuelles et collectives, intérieures et extérieures. Le profil de sortie des finissants de la concentration Sports implique qu'ils auront développé leurs habiletés collaboratives et de leadership, consolidé leurs habiletés physiques fondamentales, élargi leurs connaissances des sports variés, intériorisé les bénéfices de l'activité sportive régulière pour réguler leurs émotions et évacuer le stress et adopté un mode de vie sain.

Au-delà des contenus d'apprentissage qui diffèrent d'une concentration à l'autre, plusieurs caractéristiques apparaissent équivalentes entre ces concentrations, ce qui minimise les biais qui pourraient découler de variables potentiellement confondantes et favorise une meilleure comparabilité. Le Tableau 3.1 présente une synthèse de ces caractéristiques.

Tableau 3.1 Synthèse des principales caractéristiques équivalentes entre les concentrations

Caractéristiques	**Explications**
Nombre de périodes d'enseignement par cycle de 18 jours	5 périodes d'enseignement/cycle
Nombre total d'heures par année scolaire	70 heures annuellement
Répartition des périodes d'enseignement dans la grille-horaire	Les périodes consacrées à toutes les concentrations sont réparties de la même façon dans la grille-horaire
Nombre d'enseignants/concentration	Deux enseignants pour chaque concentration
Critères de sélection des élèves	Aucun critère de sélection; le choix se fait uniquement à partir des intérêts des élèves
Taille des groupes	Entre 32 et 36 élèves par groupe
Stratégies pédagogiques privilégiées	Stratégies pédagogiques actives principalement axées sur l'apprentissage par projets

Ainsi, l'ensemble des concentrations sont associées à un même nombre de périodes d'enseignement par cycle de 18 jours, du T1 au T3, soit 5 périodes par cycle, ce qui fait en sorte que le nombre total d'heures d'enseignement allouées aux concentrations pour une année scolaire est également équivalent (total de 70 heures). De plus, étant donné que tous les élèves suivent une même grille-horaire pour l'ensemble des disciplines prévues au programme d'études, les périodes allouées à toutes les concentrations occupent la même place dans cette grille-horaire. Elles sont donc réparties de la même façon à travers le temps, ce qui limite la différence d'espacement des apprentissages, qui représente un facteur pouvant avoir une incidence importante sur la rétention et la consolidation des apprentissages (Walsh *et al.*, 2018). Chaque concentration est également chapeautée par deux enseignants; aucune d'entre elles n'est donc associée à un seul enseignant, ce qui limite également le risque que l'une des concentrations soit plus impactée par un potentiel « effet enseignant » (Hattie, 2008). Aucune ne présente de critères visant à guider la sélection des élèves; la répartition des élèves au sein de ces concentrations se fait donc uniquement sur la base de leur intérêt personnel. Qui plus est, la taille des groupes au sein de chaque concentration est également similaire, ceux-ci comptant entre 32 et 36 élèves. Finalement, l'ensemble des concentrations privilégient l'utilisation de stratégies pédagogiques actives (action et création), et utilisent peu d'enseignement magistral, ce qui maximise la comparabilité entre les concentrations et permet de mieux isoler l'effet du contenu (Programmation, Arts ou Sports) en minimisant l'impact des différences qui pourraient provenir des stratégies pédagogiques employées.

3.3 Constitution de l'échantillon

3.3.1 Processus d'obtention des données

La réalisation de cette étude a nécessité l'accès à des données de performance scolaire détaillées. Le processus d'obtention de ces données a été initié par le chercheur en établissant un contact direct avec la direction pédagogique de l'école concernée afin de valider leur intérêt pour le projet. Une fois cette validation obtenue, une demande de certification éthique a été déposée et approuvée par le Comité institutionnel d'éthique de la recherche avec des êtres humains (CIEREH) de l'Université du Québec à Montréal (N^{o} du certificat éthique 4759_e_2021). L'ensemble des données ont été soumises à un processus d'anonymisation, réalisé par l'institution avant d'être transmises au chercheur, afin d'éliminer tout risque d'identification des élèves et des enseignants. Pour ce faire, le chercheur a communiqué avec la technicienne en organisation scolaire de l'école, qui avait été mandatée pour procéder à cette anonymisation, afin de s'entendre sur la façon dont cette procédure serait réalisée. Tous les noms des

élèves et des enseignants ont ainsi été retirés, de même que d'autres informations sensibles qui auraient pu permettre de les identifier (p. ex., date de naissance, numéros des groupes-classes, etc.).

Il convient de souligner que ces données ont été recueillies annuellement par les enseignants entre 2014 et 2019 et servaient initialement à la création des bulletins scolaires. L'extraction des données a donc été effectuée à partir de ces bulletins par la technicienne administrative de l'école qui a fourni l'ensemble des informations contenues dans les bulletins finaux de chaque année scolaire pour toutes les disciplines. Après la réception des données, une vérification de leur intégrité a été effectuée par le chercheur pour s'assurer que la base de données était fonctionnelle et complète. Ce processus méticuleux a permis de garantir que celle-ci était fiable et permettait la réalisation des analyses longitudinales et comparatives.

3.3.2 Procédure pour former l'échantillon

À partir des extractions brutes des données réalisées, plusieurs étapes de prétraitement et de sélection des données ont été réalisées en vue de la constitution de l'échantillon final qui a été utilisé pour mener les analyses envisagées. La cohorte ciblée par cette étude, soit celle allant de 2014 à 2019, a été choisie puisqu'il s'agit de la toute première cohorte de la concentration Programmation et la seule à avoir complété l'entièreté de son parcours au secondaire au moment de l'obtention du certificat éthique.

Le formatage initial de la base de données anonymisée s'est avéré une étape complexe. Initialement dispersée dans de nombreux fichiers Excel distincts (une extraction Excel par niveau scolaire par année, totalisant 25 fichiers individuels), la base a été restructurée de manière à ce que chaque ligne représente un élève de la cohorte 2014 et que chaque colonne représente un résultat scolaire obtenu dans une discipline spécifique au fil des années. Cette réorganisation importante a été suivie par la sélection des trois disciplines d'intérêt susceptibles de profiter d'un transfert associé à l'apprentissage de la programmation, à savoir les mathématiques, les sciences et le français. Enfin, une variable de regroupement a été générée en fonction de la concentration choisie par chaque élève.

3.3.3 Description de l'échantillon constitué

La base de données constituée englobait ainsi une cohorte totale de 467 élèves de 1re secondaire, répartis en 12 groupes-classes. Parmi ces élèves, 27 élèves n'étaient pas inscrits dans l'une des trois concentrations (Programmation, Arts ou Sports) en raison de difficultés langagières importantes. Ces élèves étaient donc

plutôt attitrés à une concentration axée spécifiquement sur l'amélioration de leurs compétences langagières. Pour cette raison, ces 27 élèves ont été retirés de l'échantillon final.

L'échantillon final au T1 était ainsi composé de 440 élèves dont la moyenne d'âge était de 12,70 ans (ET = 0,35). Cette taille d'échantillon respecte les recommandations de statisticiens experts pour les analyses envisagées (Fan *et al.*, 2003; Shi *et al.*, 2021) et se situe également dans les balises permettant de détecter un effet modéré avec une puissance statistique de 0,8, soit entre 288 et 489 participants, tel que calculé par le biais d'une analyse de puissance a priori réalisée à l'aide d'un calculateur de taille d'échantillon pour les modèles d'équations structurelles (Soper, 2023).

Le Tableau 3.2 présente la répartition de cet échantillon selon les concentrations scolaires, ainsi que l'âge moyen et la proportion de garçons et de filles pour chacune des concentrations. Il est à noter que la proportion de garçons et de filles n'est pas équivalente, tant au sein de l'échantillon final (42 % de garçons et 58 % de filles) qu'au sein de chacune des concentrations.

Tableau 3.2 Description de l'échantillon final au temps 1 (*n* = 440)

Concentrations	Nombre de participants	Nombre de garçons (%)	Nombre de filles (%)	Âge (années)
Toutes	440	185 (42 %)	255 (58 %)	12,70 (ET = 0,35)
Programmation	127	92 (72 %)	35 (28 %)	12,70 (ET = 0,34)
Arts	193	57 (19 %)	157 (81 %)	12,71 (ET = 0,33)
Sports	120	36 (48 %)	63 (52 %)	12,68 (ET = 0,40)

Les 127 élèves faisant partie de la concentration Programmation présentent une moyenne d'âge de 12,70 ans (ET = 0,34). La proportion de garçons est plus importante (92 garçons pour 35 filles), ceux-ci représentant 73 % de cet échantillon. La concentration Arts regroupe pour sa part 193 élèves dont la moyenne d'âge est de 12,71 ans (ET = 0,33). Pour cette concentration, la proportion de filles est plus importante (157 filles pour 57 garçons), celles-ci représentant 81 % de l'échantillon. Enfin, la concentration Sports inclut 120 élèves dont la moyenne d'âge est de 12,68 ans (ET = 0,40). Contrairement aux autres, la proportion de garçons et de filles pour cette concentration est plus équilibrée, avec 57 garçons (48 %) et 63 filles (52 %).

Une analyse de variance (ANOVA à un facteur) a révélé qu'il n'y avait pas de différence significative entre les concentrations en ce qui concerne l'âge des participants, $F(2, 437) = 0{,}22$, $p = 0{,}803$. Un test de Chi-carré a été réalisé pour vérifier l'équivalence des concentrations. Considérant les informations présentées au Tableau 3.1, il était en effet probable que la distribution des participants à travers les concentrations ne soit pas uniforme, ce qu'a effectivement confirmé ce test, $\chi^2 (2, 440) = 22{,}12$, $p < 0{,}001$. Un autre test de Chi-carré a pour sa part révélé une association significative entre la concentration et le sexe, $\chi^2 (2, 440) = 92{,}96$, $p < 0{,}001$, indiquant que l'échantillon n'est pas homogène au regard de la distribution des garçons et des filles au sein des différentes concentrations. Il sera donc judicieux de prendre en compte ces inégalités lors de l'interprétation des résultats et de discuter des limites liées à la composition de l'échantillon dans la section de la discussion. Qui plus est, il apparaît important de garder à l'esprit que des effectifs plus faibles (ce qui est le cas pour les concentrations Programmation et Sports) peuvent limiter la puissance statistique, rendant ainsi plus difficile la détection d'effets significatifs.

En somme, cet échantillon présente une diversité quant à la distribution du nombre de participants au sein des concentrations et une distribution par sexe qui varie de manière significative selon la concentration. La concentration Programmation est majoritairement masculine, la concentration Arts est majoritairement féminine, tandis que la concentration Sports présente une proportion de garçons et de filles plus équilibrée.

3.4 Nature des données utilisées

Tel que cela a été mentionné, le devis de cette étude repose sur l'utilisation de données secondaires provenant des bulletins scolaires de fin d'année des élèves. Ces données ont été initialement collectées par l'établissement scolaire dans une visée d'évaluation des apprentissages. Pour chaque élève, les données suivantes ont été considérées :

- Performance pour chaque discipline : il s'agit d'une note globale sur 100 qui représente une moyenne calculée à partir de l'ensemble des évaluations sommatives réalisées par l'élève durant l'année scolaire, en mathématiques, en sciences et en français.

Qui plus est, il importe de spécifier que la performance globale pour chaque discipline scolaire est calculée à partir de la performance obtenue pour chacune des compétences faisant l'objet d'une évaluation au bulletin. Pour chaque discipline, les compétences évaluées se basent sur les compétences ciblées dans le Programme de formation de l'école québécoise (PFEQ). Par exemple, en mathématiques, trois

compétences sont identifiées dans le PFEQ (1- « Résoudre une situation-problème », 2- « Déployer un raisonnement mathématique », et 3- « Communiquer à l'aide du langage mathématique »), mais seules les deux premières sont évaluées au sein du bulletin. Pour la discipline des sciences, le bulletin scolaire présente un volet d'évaluation pratique et un volet d'évaluation théorique. Ces deux volets sont respectivement associés de manière centrale aux compétences « Chercher des réponses ou des solutions » et « Mettre à profit ses connaissances ». Les compétences et leur pondération respective varient pour chaque discipline. Le Tableau 3.3 ci-bas présente la répartition des compétences évaluées et leur pondération respective pour chaque discipline scolaire.

Tableau 3.3 Présentation des compétences évaluées pour chaque discipline et leur pondération respective

	Disciplines						
	Mathématiques		**Sciences***		**Français**		
Pondération	30 %	70 %	40 %	60 %	40 %	40 %**	20 %**
Compétences évaluées au bulletin	Résoudre une situation-problème	Déployer un raisonnement mathématique	Chercher des réponses ou des solutions (volet Pratique)	Mettre à profit ses connaissances (volet Théorie)	Lire	Écrire	Communiquer oralement

* Au Québec, la discipline des sciences porte le libellé «Science et technologie » au sein du PFEQ et intègre donc également des contenus interdisciplinaires portant sur la technologie. Étant donné qu'il s'agit d'une particularité propre au programme d'enseignement québécois, et que les résultats des études discutées dans l'état des connaissances concernent les sciences de manière générale, il a été décidé d'utiliser l'appellation « Sciences » de manière plus large afin de faciliter la compréhension du lecteur. Il est aussi à noter que l'évaluation des deux volets en sciences (volets Pratique et Théorie) inclut également une 3e compétence « Communiquer à l'aide des langages utilisés en science et technologie » qui peut, dans une moindre mesure, teinter la performance obtenue. Seules les deux compétences centrales permettant de distinguer chacun des volets ont donc été retenues et présentées dans le tableau.

** Il est à noter que la pondération est légèrement différente en 5e secondaire : pour cette année, la compétence « Écrire » est associée à une pondération de 50 % et la compétence « Communiquer oralement » est associée à une pondération de 10 %.

Afin d'avoir une compréhension plus fine des potentiels effets de transfert pour chaque discipline, il apparaît intéressant de considérer les compétences qui composent ces disciplines, ce qui permet d'obtenir un degré de précision supplémentaire allant au-delà de la discipline globale. Tel que cela a été discuté dans le cadre théorique, plusieurs auteurs ont en effet identifié des compétences susceptibles d'être développées par l'apprentissage de la programmation et réinvesties dans d'autres disciplines scolaires (voir p. ex., Popat et Starkey, 2019 ainsi que Scherer *et al.*, 2019). Cet aspect est discuté plus en détail dans la section suivante qui présente les analyses envisagées.

Il apparaît également important de mentionner que les évaluations menées au sein de chaque discipline pour un niveau scolaire donné étaient uniformes et standardisées. Bien que cette standardisation n'ait pas été contrôlée de manière expérimentale, elle permet une comparaison plus fiable des résultats scolaires entre les différentes concentrations et au fil du temps. L'uniformité des méthodes d'évaluation minimise également l'impact possible de variables confondantes telles que l'effet « enseignant » (Hattie, 2009), ce qui renforce la validité de l'analyse longitudinale des données.

À la lumière des données utilisées, la base de données finale comportait ainsi un total de :

- 8 800 observations qui ont été considérées pour réaliser les analyses principales menées sur la performance globale pour chaque discipline : 440 élèves * 4 disciplines (3 disciplines traditionnelles et 1 concentration d'appartenance) * 5 années scolaires
- 24 200 observations qui ont été considérées pour réaliser les analyses spécifiques incluant en plus les compétences évaluées pour chaque discipline : 440 élèves * 11 résultats (les résultats aux 7 compétences ainsi que le résultat global pour les 3 disciplines et la concentration d'appartenance) * 5 années scolaires.

3.5 Analyse des données

Afin de vérifier les hypothèses de recherche avancées, des analyses provenant du domaine de la modélisation par équations structurelles (*structural equations models*; *SEM*) ont été réalisées (Caron, 2018; Girard et Béland, 2017). Cette approche statistique, moins fréquemment utilisée en recherche en sciences de l'éducation, est particulièrement adaptée pour s'intéresser aux relations complexes entre des variables observées et des variables latentes sur plusieurs temps de mesure (Caron, 2018).

Une variable latente peut être définie comme une variable non observable que l'on cherche à mesurer indirectement à travers une série de variables manifestes (Muthén, 2004). En d'autres termes, contrairement aux variables qui peuvent être directement observées (par exemple, la performance scolaire), les variables latentes permettent de s'intéresser à des construits théoriques latents qui ne peuvent pas être mesurés directement, comme l'attitude, la motivation ou, dans le cas de cette étude, le transfert (Muthén, 2004; Perez *et al*., 2018). Les variables latentes sont cruciales à la modélisation par équations structurelles, car elles permettent de distinguer et de mesurer les tendances dans les données longitudinales, qui ne sont pas immédiatement évidentes. Dans cette étude, trois variables latentes permettent de caractériser comment les élèves de chaque concentration progressent au fil du temps, pour

chacune des disciplines d'intérêt, contribuant ainsi à mettre en évidence l'existence et l'ampleur d'un éventuel transfert des apprentissages. Ces variables sont détaillées à la section 3.5.1 ci-bas.

L'approche de modélisation par équations structurelles permet également de tester simultanément des effets directs et indirects entre les variables, tout en limitant les effets de grappes dans les données. Plus spécifiquement, un type d'analyses propre à la modélisation par équations structurelles a été employé : les analyses de trajectoires latentes (*latent growth analysis*; *LGA*) par groupes multiples (*multiple-group analysis*; *MGA*). Ces analyses ont été réalisées en quatre grandes étapes, telles qu'illustrées à la Figure 3.2, à l'aide du logiciel Mplus (Muthén et Muthén, 2017) : (1) établissement du modèle de base des trajectoires latentes pour chaque discipline scolaire; (2) établissement du modèle multigroupe (incluant les concentrations) pour chaque discipline scolaire; (3) comparaison de l'ajustement des modèles et (4) comparaison des trajectoires des concentrations.

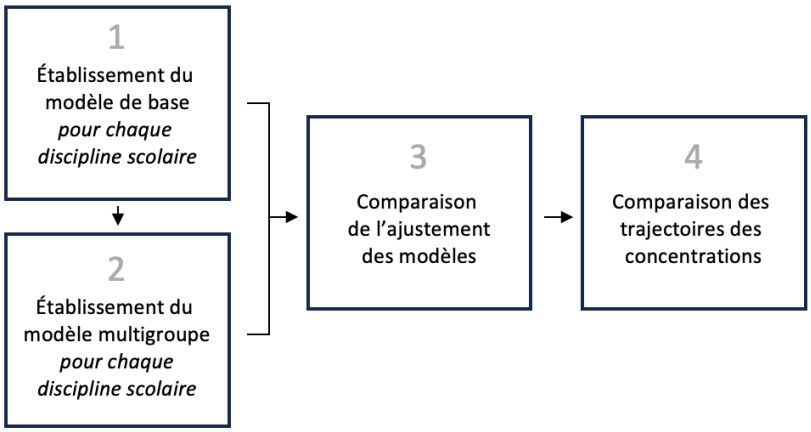

Figure 3.2 Synthèse des principales étapes d'analyse.

3.5.1 Étape 1 : établissement du modèle de base pour chaque discipline scolaire

L'analyse de trajectoires latentes est une méthode privilégiée pour examiner les trajectoires longitudinales au sein d'un échantillon. Cette technique nécessite au minimum trois temps de mesure et permet d'estimer les changements au sein de l'échantillon en fonction de deux ou trois variables latentes, selon qu'il s'agisse d'une relation linéaire ou quadratique (Caron, 2018). Ces variables latentes sont la pente (S et Q) et l'ordonnée à l'origine (I).

Le paramètre S (et le paramètre Q dans le cas d'une relation quadratique) est un indicateur de la tendance (augmentation ou diminution) de la variable mesurée dans le temps, tandis que le paramètre I représente le point de départ de l'échantillon. Chacun de ces paramètres est associé à une moyenne et une variance, permettant par la suite de comparer l'homogénéité de la croissance ou de la décroissance au sein de l'échantillon au fil du temps, selon une variable de regroupement (Caron, 2018).

Dans un premier temps, une analyse de trajectoires latentes a d'abord été effectuée de façon séparée pour chacune des disciplines scolaires (mathématiques, sciences, et français), en utilisant les cinq temps de mesure disponibles (secondaire 1 à 5). Un modèle de base sans les concentrations a donc été mis au point pour chaque discipline, menant ainsi à l'obtention de trois modèles. La Figure 3.3 illustre le modèle de base destiné à évaluer les relations entre les variables dépendantes (résultats scolaires globaux pour chaque discipline) aux cinq points temporels (T1 à T5) et les trois variables latentes (« Ordonnée à l'origine », « Pente linéaire », et « Pente quadratique »). Ces variables latentes représentent respectivement le niveau initial de l'échantillon, la tendance linéaire du changement au fil du temps (pente linéaire), et la tendance du changement quadratique (pente quadratique).

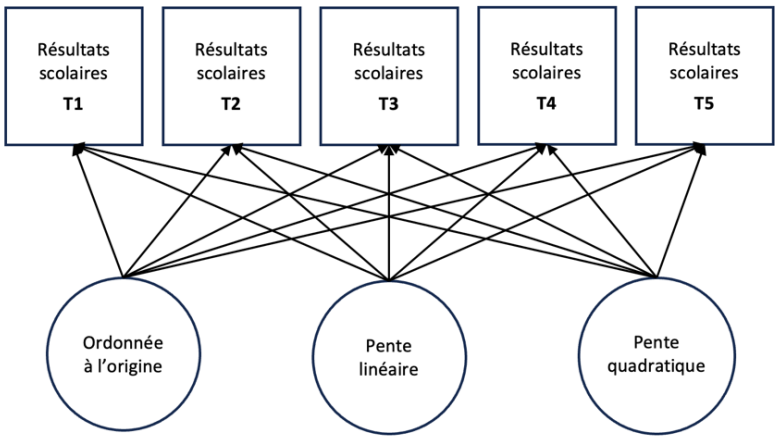

Figure 3.3 Modèle de base des trajectoires latentes qui ne comprend pas la variable de regroupement. Un modèle est établi de manière distincte pour chaque discipline scolaire.

Ce modèle de base permet donc de rendre compte de la trajectoire longitudinale de la performance pour chaque discipline. Dans le cadre de cette recherche, aucune hypothèse n'était préalablement formulée quant à la trajectoire de ce modèle de base pour chaque discipline; la performance en mathématiques, en

sciences et en français pouvant varier positivement ou négativement à travers le temps, de manière tant linéaire que quadratique. Les modèles de base constituent simplement des points de comparaison pour les modèles multigroupes qui, eux, incluent les concentrations (Programmation, Arts ou Sports).

De plus, tel que cela a été mentionné plus tôt, en plus des modèles de base pour la performance globale pour chaque discipline, des analyses plus spécifiques ont également été réalisées afin d'évaluer les relations entre les résultats scolaires obtenus pour chacune des compétences évaluées pour chaque discipline aux cinq temps de mesure, et les trois variables latentes. Ces analyses reprennent exactement la même procédure de modélisation par trajectoires latentes détaillée ci-haut, ce qui a mené à l'obtention de sept modèles statistiques de base supplémentaires (deux compétences en mathématiques, deux compétences en sciences et trois compétences en français), pour un total de dix modèles de base.

3.5.2 Étape 2 : établissement du modèle multigroupe pour chaque discipline scolaire

La deuxième étape consistait en l'établissement du modèle multigroupe pour chaque discipline scolaire et chaque compétence, c'est-à-dire du modèle incluant la variable de regroupement des concentrations. Dans le cadre de cette étude, il est important de rappeler que les groupes correspondent aux concentrations scolaires; le terme « multigroupe » est donc employé pour référer à ces concentrations, puisqu'il s'agit de l'appellation typiquement associée aux analyses de trajectoires latentes qui comportent plusieurs groupes. Ce modèle est présenté à la Figure 3.4 ci-bas.

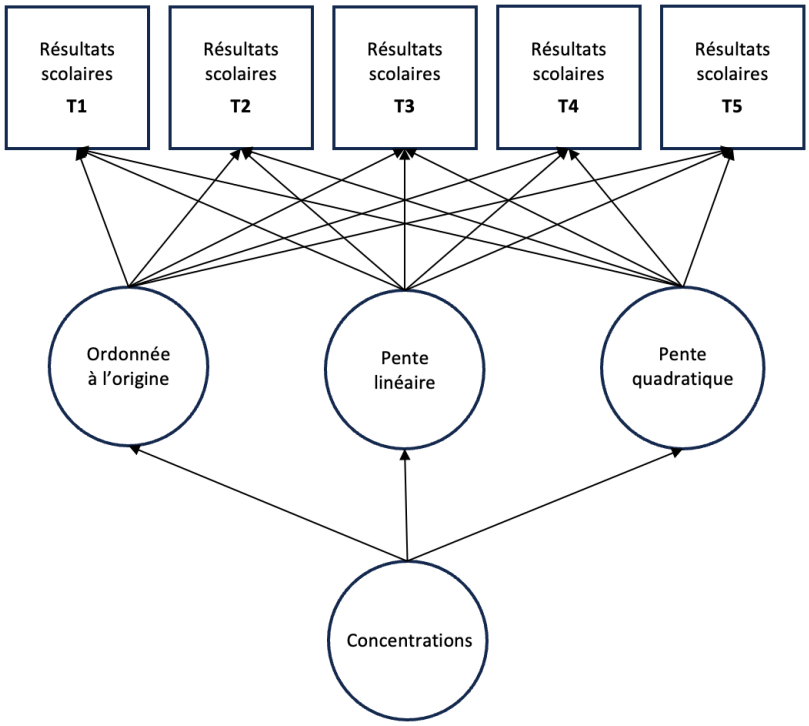

Figure 3.4 Modèle multigroupe des trajectoires latentes qui inclut la variable de regroupement des concentrations. Un modèle est établi de manière distincte pour chaque discipline scolaire.

Les modèles multigroupes permettent donc de représenter la trajectoire longitudinale associée à chaque concentration. Comme pour les modèles de base, un total de dix modèles multigroupes ont été établis.

3.5.3 Étape 3 : comparaison de l'ajustement des modèles

Après avoir établi le modèle de base et le modèle multigroupe pour chaque discipline, l'ajustement (en anglais, le *fit*) de ces modèles a été comparé. La qualité de l'ajustement des modèles a été évaluée à l'aide de plusieurs tests statistiques complémentaires recommandés dans la littérature (Hoyle, 1995; Bentler, 1999). Ces tests statistiques incluent le calcul du Chi-carré (χ^2), de l'indice de comparaison de l'ajustement (*Comparative Fit Index*, CFI), de l'indice de Tucker-Lewis (*Tucker-Lewis Index*, TLI) et de l'erreur quadratique moyenne de l'approximation (*Root Mean Square Error of Approximation*, RMSEA), dont les fonctions respectives sont présentées dans les lignes qui suivent. Ces métriques sont utilisées, d'une part, pour évaluer la qualité de l'ajustement des modèles de manière individuelle, et d'autre part pour comparer les ajustements de différents modèles.

Le test de Chi-carré permet de vérifier l'adéquation entre les covariances prédites par le modèle et celles observées dans les données. Un résultat non significatif indique que le modèle ajusté reflète fidèlement les relations dans les données (Caron, 2018). Le CFI compare pour sa part la performance du modèle ajusté à celle du modèle de base; des valeurs au-delà de 0,90 suggérant un bon ajustement du modèle (Bentler, 1990). Le TLI mesure l'amélioration relative de l'ajustement du modèle par rapport au modèle de base; des valeurs approchant ou excédant 0,95 reflètent un ajustement jugé supérieur (Kline, 2005). Finalement, le RMSEA fournit une estimation de l'erreur d'approximation du modèle dans la population globale; des valeurs en dessous de 0,05 sont jugées excellentes, des valeurs entre 0,05 et 0,08 sont généralement interprétées comme étant bonnes (Browne et Cudeck, 1993).

La combinaison de ces tests permet d'obtenir une appréciation complète de la qualité de l'ajustement des modèles. Il est ainsi possible de déterminer si l'inclusion de la variable de regroupement des concentrations scolaires au sein des modèles de chaque discipline mène à une amélioration significative de l'ajustement, ce qui suggérerait qu'il existe une relation entre l'appartenance aux concentrations et les résultats scolaires aux disciplines. À l'inverse, si le modèle de base indique une meilleure capacité à expliquer les données, cela signifie l'inclusion des concentrations dans le modèle multigroupe ne permet pas de mieux représenter les données. En d'autres termes, cela veut dire que la variable de regroupement utilisée dans le modèle multigroupe ne contribue pas à améliorer de manière significative l'ajustement des données comparativement à la trajectoire globale offerte par le modèle de base.

Cette troisième étape permet donc de vérifier a priori si l'appartenance à une concentration influence les données des résultats scolaires. La quatrième étape, visant à comparer statistiquement la trajectoire des concentrations entre elles, n'est donc possible que dans la mesure où le modèle multigroupe présente un meilleur ajustement que le modèle de base. Si ce n'est pas le cas pour une discipline scolaire, cela signifie qu'il n'existe pas de relation entre l'appartenance aux concentrations et les résultats scolaires obtenus à cette discipline. Dans le cas de la présente étude, l'absence de cette relation suggère que les résultats scolaires des élèves ne se distinguent pas sur la base des concentrations et donc qu'il n'y aurait pas une concentration particulièrement propice au transfert des apprentissages pour cette discipline.

3.5.4 Étape 4 : comparaison des trajectoires des concentrations

Pour les disciplines scolaires où le modèle multigroupe indique un meilleur ajustement que le modèle de base, l'étape finale consiste donc à comparer les trajectoires des concentrations entre elles, via la

comparaison de leurs variables latentes. Tel que discuté, cette étape ne peut être réalisée que lorsque le modèle multigroupe présente un meilleur ajustement aux données que le modèle de base (Caron, 2018); autrement, la comparaison statistique des trajectoires peut non seulement s'avérer superflue, mais également induire en erreur en suggérant des différences de groupe significatives qui n'existent pas. Pour comparer les trajectoires des concentrations, des tests de Wald ont été réalisés avec le logiciel Mplus, en utilisant la commande *Model Test* (Caron, 2018; Muthén et Muthén, 2009). Ce type de test statistique permet de comparer directement les variables latentes du modèle d'intérêt entre les différents groupes (concentrations) et mène à l'obtention d'une valeur de Wald (W), qui représente la magnitude de la différence observée. La statistique de Wald s'accompagne d'une valeur de *p*. Pour chaque test, si la valeur de *p* est inférieure à un seuil prédéterminé (0,05), il est alors possible de conclure que les paramètres diffèrent significativement entre les groupes. Une différence significative relativement à l'ordonnée à l'origine signifie que les concentrations présentent une performance scolaire initiale distincte. Une différence statistique significative entre les concentrations relativement aux pentes (S et Q) signifie que la performance des élèves varie différemment à travers le temps en fonction de leur concentration d'appartenance. C'est donc la comparaison de ces derniers paramètres (S et Q) qui permet de vérifier les hypothèses, considérant que le devis de recherche ne comportait pas une attribution aléatoire des élèves aux concentrations et qu'il est donc attendu que la performance initiale de chacune soit différente.

Par la comparaison des trajectoires des concentrations, cette quatrième étape permet donc de vérifier les hypothèses de recherche formulées au regard de l'effet de transfert de la programmation vers les disciplines scolaires des mathématiques, des sciences et du français.

3.5.5 Justification du choix des analyses de trajectoires latentes

Les méthodes de comparaison statistique traditionnelles, telles que l'ANOVA à mesures répétées, sont souvent bien adaptées aux devis de recherche expérimentaux où les chercheurs ont un contrôle direct sur la manipulation des variables indépendantes et l'attribution aléatoire des participants aux groupes (Gaudreau, 2011; Field, 2013). Ces méthodes sont en effet conçues pour évaluer les effets causaux d'une ou plusieurs variables indépendantes sur une variable dépendante, dans un contexte où les conditions expérimentales sont rigoureusement contrôlées (Field, 2013). En revanche, ce type d'analyse peut être moins efficace pour des devis observationnels où il existe une multitude de variables confondantes et où l'attribution des participants aux conditions ne peut être réalisée de manière aléatoire. Dans de tels cas, des techniques statistiques plus avancées comme la modélisation par équations structurelles apparaissent

plus appropriées (Duncan et Duncan, 2009). Par exemple, dans l'éventualité où des groupes comparés ne présenteraient pas une performance initiale équivalente, des analyses par trajectoires latentes permettent tout de même de comparer la variation de la performance des groupes à travers le temps via la comparaison de variables latentes, telles que les pentes (S et Q) des trajectoires de leur performance dans le temps.

Qui plus est, le transfert des apprentissages est un construit qu'il apparaît difficile de mesurer directement, car il implique de réinvestir ou réutiliser certaines capacités cognitives qui ne sont pas observables; c'est la raison pour laquelle le transfert des apprentissages est souvent considéré comme un construit latent (Noack, 2014). Pour le quantifier, il apparaît donc approprié d'utiliser des analyses de trajectoires latentes qui permettent de modéliser les relations entre les différentes variables et de mesurer de manière indirecte l'effet de transfert. Ce type d'analyse apparaît particulièrement utile pour analyser les effets de transfert, car elle peut modéliser à la fois le niveau initial d'un concept (c'est-à-dire l'ordonnée à l'origine) et le changement dans le temps (c'est-à-dire la pente) qui permet d'estimer l'ampleur de l'effet de transfert.

Les analyses de trajectoires latentes permettent ainsi d'explorer comment les compétences ou les connaissances acquises dans un contexte donné peuvent influencer la performance dans un contexte différent, en prenant en compte des variables latentes non observées qui sous-tendent le transfert. Elles permettent donc de modéliser comment les variables d'intérêt évoluent au fil du temps. Il est ainsi possible d'étudier comment l'effet de transfert se manifeste progressivement à travers différentes périodes ou points de mesure, tel que cela est le cas dans ce projet.

3.5.6 Considérations statistiques supplémentaires

En plus des étapes qui ont été détaillées précédemment, certaines considérations statistiques supplémentaires méritent d'être présentées, soit la gestion de l'attrition des données ainsi que la correction du nichage des données.

3.5.6.1 Gestion de l'attrition des données

L'attrition des données, c'est-à-dire la perte de données au fil du temps, est un problème courant dans les recherches longitudinales qui peut affecter la validité interne d'une étude. Dans le contexte de ce projet,

nous avons examiné le taux d'attrition au sein de chaque concentration. Le Tableau 3.4 décrit l'attrition expérimentale du T1 au T5.

Tableau 3.4 Présentation de l'attrition expérimentale du T1 au T5

Année scolaire	Nombre de participants	Attrition nominale	Attrition annuelle (%)	Attrition depuis le T1 (%)
Secondaire 1 (T1)	440	0	0	0
Secondaire 2 (T2)	414	26	5,9	5,9
Secondaire 3 (T3)	388	52	6,3	11,8
Secondaire 4 (T4)	372	68	4,1	15,5
Secondaire 5 (T5)	365	75	1,9	17,1

De l'avis de certains auteurs, les taux acceptables d'attrition pour des devis de recherche longitudinaux varient de 5 % à 20 % (Bennett, 2001; Peng, 2006; Schlomer, 2010), bien qu'un consensus généralement accepté par la plupart des chercheurs soit de 10 % (Langkamp, 2010). Considérant que le pourcentage d'attrition totale atteint 17,1 % (souligné dans le tableau précédent), un test de Little MCAR (1988) a été réalisé sur les résultats scolaires afin de vérifier si cette attrition était aléatoire. Le test a permis de conclure que les données manquantes étaient distribuées de manière aléatoire ($p > 0,05$) suivant un modèle MCAR (*Missing Completely At Random*). Il est donc raisonnable de croire que l'attrition observée est due à des causes externes aux concentrations scolaires à l'étude. Par ailleurs, il convient de noter que les analyses de trajectoires latentes permettent de gérer l'attrition de manière plus robuste, comparativement aux analyses univariées, en utilisant l'ensemble des données disponibles.

3.5.6.2 Correction du nichage des données

Dans les études en éducation, il est fréquent que les données soient structurées de manière hiérarchique, c'est-à-dire structurées en grappes (Page, 2020). Ces données sont alors souvent qualifiées de données « nichées » (Bauer, 2020). Cette structure peut être prise en compte dans les analyses en utilisant des modèles linéaires mixtes, également connus sous le nom de modèles hiérarchiques linéaires (*linear hierarchical models*).

Le modèle linéaire mixte permet d'inclure à la fois des effets fixes, qui sont constants à travers les unités d'analyse (ici, les élèves), et des effets aléatoires, qui peuvent varier entre les groupes (ici, les

concentrations et les groupes-classes). Ce type de modélisation permet donc de tenir compte d'une possible corrélation intragroupe (p. ex., l'interdépendance des résultats scolaires au sein des concentrations), qui caractérise généralement les données nichées, afin de diminuer les biais associés à ces regroupements. L'intégration d'effets aléatoires dans le modèle permet de reconnaître que la relation entre les variables d'intérêt peut être différente selon les concentrations. Par exemple, l'effet de la programmation sur la performance en mathématiques peut être différent de son effet sur la performance en sciences ou en français. De même, cet effet peut varier en fonction des caractéristiques spécifiques des élèves au sein de chaque concentration.

Étant donné que les données sont imbriquées au sein de grappes (les élèves sont regroupés au sein de groupes-classes fixes et ces groupes-classes sont regroupés au sein de concentrations), des modèles hiérarchiques linéaires ont été utilisés pour ajuster les estimations des variables latentes et tenir compte de cette structure des données nichées (Caron, 2018). Cette étape est cruciale pour assurer l'exactitude des inférences statistiques. La fonction « *Type = complex »* qui permet de limiter les effets de grappes (*clustering*) dans les données (Caron, 2018) a donc également été appliquée à l'ensemble des analyses envisagées à l'aide du logiciel Mplus, qui est spécifiquement conçu pour mener ce type d'analyse longitudinale (Muthén, 1997). La correction du nichage des données, par l'utilisation des modèles hiérarchiques linéaires et de la fonction « *Type = complex »*, a ainsi permis de contrôler les effets intragroupes et intergroupes, ce qui améliore le pouvoir causal de l'étude en réduisant les biais potentiels découlant de la non-randomisation des concentrations scolaires.

3.6 Considérations éthiques

Ce projet de recherche a reçu l'approbation du Comité institutionnel d'éthique de la recherche avec des êtres humains (CIEREH) de l'Université du Québec à Montréal (No du certificat éthique 4759_e_2021). Les données secondaires utilisées dans le cadre de cette recherche sont confidentielles. Une procédure d'anonymisation a été réalisée par l'établissement scolaire avant même que les données ne soient transmises au chercheur : les participants ont ainsi été identifiés par un numéro de code et la clé du code reliant le nom du participant à ses données personnelles n'est pas connue du chercheur. Cette procédure garantit également qu'aucune publication ou communication scientifique ne renfermera d'éléments d'information pouvant permettre d'identifier un participant ou un enseignant.

Il n’a pas été nécessaire d’obtenir le consentement des parents, tuteurs ou élèves, compte tenu de l’anonymisation complète des données. Par ailleurs, il a été convenu que l’utilisation des données se limiterait à ce projet de recherche, à la demande de l’établissement scolaire.

CHAPITRE 4
RÉSULTATS

Ce quatrième chapitre vise à présenter les principaux résultats obtenus à la suite de l'analyse des données. L'objectif principal de cette recherche est de vérifier s'il existe un effet de transfert de l'apprentissage de la programmation vers les disciplines scolaires traditionnelles. Ce chapitre est donc structuré en quatre sections distinctes, afin d'aborder de manière individuelle les résultats propres à chaque discipline, dans l'ordre suivant : mathématiques, sciences et français, puis de réaliser une synthèse de ces résultats pour en offrir un portrait plus global. Pour chacune des disciplines, les résultats relatifs à la performance globale sont d'abord présentés. Puis, les résultats pour chacune des compétences évaluées au sein de la discipline sont abordés de manière distincte. À chaque fois, la comparaison de l'ajustement des deux modèles est d'abord présentée et, lorsque cela est possible, les trajectoires des concentrations sont ensuite comparées afin de pouvoir vérifier les hypothèses de recherche. L'ensemble de ces résultats seront par la suite discutés et interprétés dans le prochain chapitre au regard des hypothèses préalablement formulées et de la littérature scientifique existante.

4.1 Mathématiques

Tel que présenté au Tableau 3.3, deux compétences en mathématiques font partie de l'évaluation au sein des bulletins scolaires, soit la compétence 1 « Résoudre une situation-problème » et la compétence 2 « Déployer un raisonnement mathématique ». La performance globale en mathématiques équivaut donc à la moyenne pondérée des résultats de fin d'année scolaire obtenus pour la compétence 1 (pondération de 30 %) et la compétence 2 (pondération de 70 %).

4.1.1 Performance globale

Cette section vise dans un premier temps à comparer l'ajustement des modèles relatifs à la performance globale en mathématiques. Pour ce faire, les indices d'ajustement du modèle de base sont d'abord présentés, ainsi que les paramètres permettant d'illustrer la trajectoire de la performance globale des élèves du T1 au T5. De la même façon, les indices d'ajustement du modèle multigroupe (modèle d'intérêt incluant les concentrations) sont ensuite présentés, de même que les paramètres des trajectoires de chacune des concentrations. Les indices d'ajustement des deux modèles sont comparés afin de déterminer si la variable de regroupement des concentrations permet d'améliorer l'ajustement du modèle aux

données. Dans un deuxième temps, si le modèle multigroupe présente un meilleur ajustement que le modèle de base, une comparaison des paramètres des trajectoires des concentrations est finalement réalisée à l'aide de tests comparatifs de Wald afin de déterminer s'il existe des différences significatives entre ces trajectoires.

4.1.1.1 Comparaison de l'ajustement des modèles

Tel que discuté dans la section 3.5 de la méthodologie, l'ajustement du modèle de base ainsi que l'ajustement du modèle multigroupe ont d'abord été testés à l'aide de quatre tests statistiques usuels et complémentaires qui permettent ensemble d'obtenir une appréciation complète de la qualité de l'ajustement des modèles (voir section 3.5.3 pour une description plus détaillée de chacun des tests). D'abord, pour ce qui est du modèle de base, le test de Chi-Carré a révélé un résultat significatif $\chi^2(11) = 116{,}966$, $p < 0{,}001$, qui suggère une divergence notable entre les données observées et celles prédites par le modèle, ce qui reflète un mauvais ajustement du modèle de base. De manière convergente, le *Root Mean Square Error of Approximation* (RMSEA), un indice mesurant l'erreur d'ajustement du modèle, est de 0,148. Cette valeur, qui dépasse largement le seuil standardisé de 0,08, reflète un ajustement faible, indiquant que le modèle de base n'est possiblement pas en mesure de saisir toutes les nuances des données. Par ailleurs, le *Comparative Fit Index* (CFI) et le *Tucker-Lewis Index* (TLI), qui évaluent tous deux la qualité de l'ajustement du modèle statistique aux données observées, sont respectivement de 0,942 et 0,921. Pour ces deux tests, le seuil de 0,95 est généralement considéré comme un bon ajustement; les valeurs obtenues, bien que relativement élevées, ne rencontrent donc pas le standard habituel et ne suggèrent donc pas un ajustement optimal. L'ensemble de ces indices souligne donc que le modèle de base ne parvient pas à représenter de manière satisfaisante la structure sous-jacente des données, selon les seuils standardisés usuels. Cela peut être dû à la nature des données, à la spécificité des variables incluses dans le modèle, à la dynamique entre ces variables, ou encore à d'autres facteurs exogènes qui ne sont pas pris en compte dans le modèle actuel.

Ce modèle de base a estimé les variables latentes suivantes : l'ordonnée à l'origine (I), la pente linéaire (S), et la pente quadratique (Q). Les estimations des moyennes pour ces variables étaient respectivement de I : 78,784 (Erreur standard, ES = 0,535, $p < 0{,}001$), S : -0,901 (ES = 0,324, $p = 0{,}005$), et Q : -0,378 (ES = 0,083, $p < 0{,}001$). La Figure 4.1 illustre graphiquement la trajectoire estimée du modèle de base pour les variables I, S, et Q de la performance globale en mathématiques. Il est possible de constater une

tendance initiale positive en mathématiques (I), suivie d'une légère baisse linéaire (S) et d'une accélération de cette baisse de manière quadratique (Q) au fil du temps.

Il apparaît important de rappeler qu'aucune hypothèse n'était formulée quant à la trajectoire du modèle de base. Cette trajectoire représente en effet la trajectoire de la performance globale en mathématiques pour l'ensemble des élèves inclus dans l'étude, pour chacune des années scolaires. Puisqu'il ne s'agit pas d'une étude expérimentale, le devis de recherche ne prévoyait pas de sélectionner des élèves sur la base de leur performance, ou encore de répartir les élèves aléatoirement au sein de différents groupes. La présente étude emploie un devis de type observationnel qui prévoit simplement utiliser et modéliser les données secondaires collectées pour l'ensemble des élèves. La trajectoire du modèle de base permet donc de constater la tendance qui émerge au regard de la performance globale des élèves en mathématiques, en raison de variables qui ne sont pas contrôlées. Dans ce cas-ci, le déclin de la performance observé pourrait découler de différentes variables : une augmentation de la complexité des apprentissages à réaliser, des évaluations de plus en plus difficiles, des élèves de moins en moins engagés, etc.

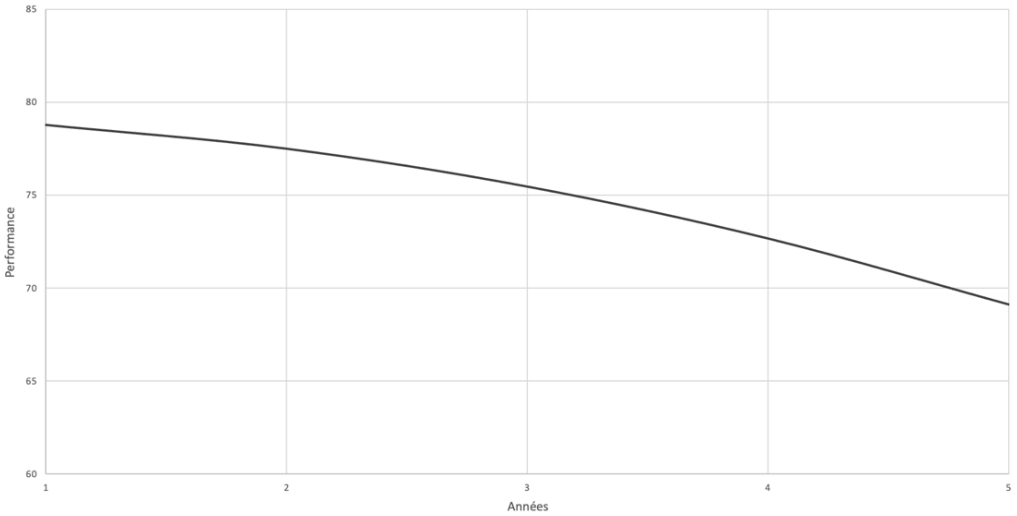

Figure 4.1 Trajectoire estimée du modèle de base pour la performance globale en mathématiques.

Ensuite, l'ajustement du modèle multigroupe a également été examiné à l'aide des mêmes mesures d'ajustement : χ^2 (18) = 128,86, p = 0,000, RMSEA = 0,125; CFI = 0,959; TLI = 0,938. Ces indices suggèrent un meilleur ajustement aux données que le modèle de base, bien qu'ils ne reflètent pas tous un ajustement élevé selon les standards recommandés.

Au terme de ces analyses, les deux modèles présentent donc un ajustement moyen, ce qui indique que la complexité des données n'est pas entièrement capturée et représentée par ces derniers. Toutefois, un meilleur ajustement dans le modèle multigroupe par rapport au modèle de base semble indiquer que l'ajout de la variable de regroupement (les concentrations) contribue à une meilleure représentation des données.

Les paramètres de chaque groupe (concentration) du modèle multigroupe ont été estimés. Pour la concentration Programmation, les moyennes estimées pour I, S, et Q sont respectivement de 78,987 (ES = 1,117, $p < 0,001$), -2,671 (ES = 0,759, $p < 0,001$), et 0,147 (ES = 0,183, $p = 0,420$). Pour la concentration Arts, ces valeurs sont de 80,721 (ES = 0,690, $p < 0,001$), -0,410 (ES = 0,435, $p = 0,346$), et -0,504 (ES = 0,108, $p < 0,001$). Pour la concentration Sports, elles sont de 75,163 (ES = 1,083, $p < 0,001$), -0,204 (ES = 0,687, $p = 0,766$), et -0,618 (ES = 0,192, $p < 0,01$). La Figure 4.2 fournit une représentation visuelle des trajectoires des trois concentrations pour la performance globale en mathématiques.

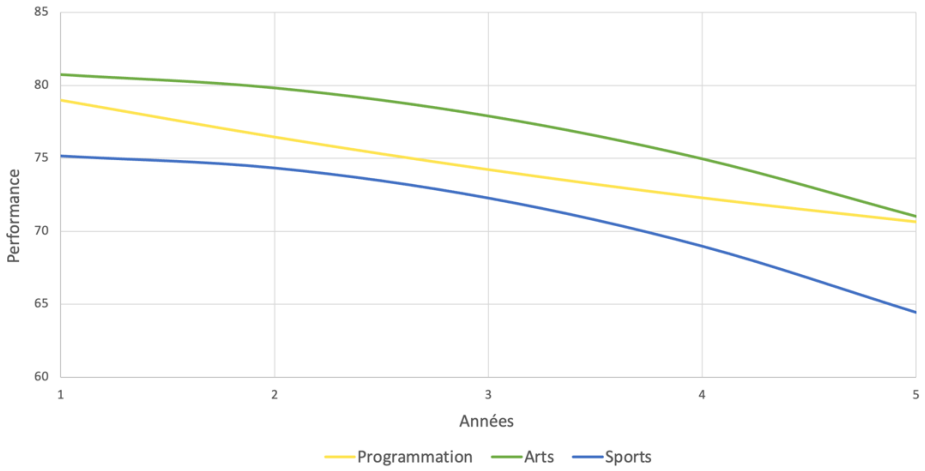

Figure 4.2 Trajectoires des trois concentrations pour la performance globale en mathématiques.

Cette figure permet globalement de constater une tendance vers le bas de la performance à travers les cinq années, et ce, pour les trois concentrations. De manière qualitative, la trajectoire de la concentration Programmation semble différer des deux autres. La section qui suit vise à comparer statistiquement ces trajectoires de manière à vérifier si elles se distinguent au regard de la performance globale des élèves en mathématiques.

4.1.1.2 Comparaison des trajectoires des concentrations

Considérant que les indices d'ajustement du modèle multigroupe sont meilleurs que ceux du modèle de base, des tests de Wald ont été réalisés afin d'évaluer les différences entre les concentrations pour les variables latentes I, S, et Q. Une valeur de *p* inférieure à 0,05 indique une différence statistiquement significative entre les variables comparées. Le Tableau 4.1 détaille les valeurs de la statistique de Wald (W) et les valeurs *p* correspondantes pour chaque comparaison de groupes, offrant ainsi une vision exhaustive des différences observées.

Tableau 4.1 Résultats des tests comparatifs de Wald entre les paramètres des trajectoires des concentrations pour la performance globale en mathématiques

Variables latentes	**Comparaisons**	**Statistique de Wald (W)**	***p***
Ordonnée à l'origine (I)	Prog. vs Arts	-1,32	0,187
	Prog. vs Sports	2,46	0,014*
	Arts vs Sports	4,33	< 0,001*
Pente linéaire (S)	Prog. vs Arts	-2,58	0,010*
	Prog. vs Sports	-2,41	0,016*
	Arts vs Sports	-0,25	0,800
Pente quadratique (Q)	Prog. vs Arts	3,06	0,002*
	Prog. vs Sports	2,88	0,004*
	Arts vs Sports	0,52	0,605

*Différence statistiquement significative à un seuil de $p < 0,05$.

À la lumière de ce tableau, il est d'abord possible de constater qu'il existe des différences significatives au niveau de l'ordonnée à l'origine (performance initiale) entre les concentrations. Ces différences significatives entre les concentrations Programmation et Sports ($p = 0,014$) et entre Arts et Sports ($p < 0,001$) révèlent que la concentration Sports était initialement (T1) moins performante que les autres en mathématiques. Le tableau met également en évidence qu'il existe des différences significatives au niveau des pentes (S et Q), ce qui paraît davantage d'intérêt au vu de l'objectif de la recherche. Ces résultats indiquent en effet des différences statistiques au regard des trajectoires des concentrations. Tant la pente linéaire (S) que quadratique (Q) sont différentes pour la concentration Programmation, comparativement à Arts (S : $p = 0,010$; Q : $p = 0,002$) et Sports (S : $p = 0,016$; Q : $p = 0,004$). Ces différences

sur le plan de la trajectoire soulignent que la concentration Programmation est associée à une trajectoire distincte au regard de la performance globale des élèves en mathématiques, lorsque comparée aux deux autres concentrations : le déclin initial de la performance apparaît plus prononcé pour la concentration Programmation, mais l'accélération de ce déclin entre le T1 et le T5 apparaît cependant moins importante.

Considérant que les différences observées entre les ordonnées à l'origine nous intéressent peu puisque le devis ne mobilisait pas une attribution aléatoire des élèves au sein des concentrations, la Figure 4.3 présente les trajectoires rapportées à une même ordonnée à l'origine afin de faciliter l'observation des différences pour les paramètres S et Q. Ce graphique permet de constater que le déclin linéaire (paramètre S) est initialement plus prononcé pour la concentration Programmation, mais que ce déclin s'accélère davantage (paramètre Q) pour les concentrations Arts et Sports. Lorsque rapportée à un même point de départ, la concentration Programmation est donc celle associée au plus petit déclin et donc à la meilleure performance au T5.

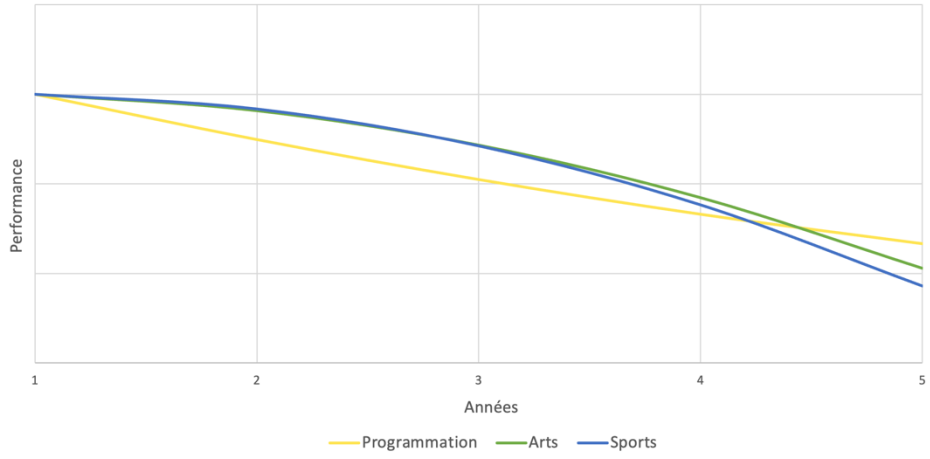

Figure 4.3 Trajectoires des trois concentrations pour la performance globale en mathématiques, représentées à partir d'une même ordonnée à l'origine.

Cette première analyse de la performance globale en mathématique permet d'obtenir un portrait général de la façon dont la performance des élèves évolue dans le temps selon leur concentration d'appartenance, pour la discipline des mathématiques. L'analyse plus fine de la performance aux deux compétences évaluées en mathématiques pourrait toutefois offrir plus d'informations quant à la provenance des différences observées entre les trajectoires des concentrations. C'est ce que présentent les paragraphes qui suivent.

4.1.2 Mathématiques : Compétence 1 « Résoudre une situation-problème »

4.1.2.1 Comparaison de l'ajustement des modèles

Encore une fois, plusieurs indices statistiques ont été utilisés afin d'évaluer l'ajustement du modèle de base pour la compétence 1 en mathématiques. L'ensemble de ces indices souligne un faible ajustement du modèle de base : χ^2 (11) = 165,859, p = 0,000, RMSEA = 0,179; CFI = 0,829; TLI = 0,766.

Les estimations des paramètres de ce modèle étaient respectivement de I : 84,054 (ES = 0,466, p < 0,001), S : -2,460 (ES = 0,451, p < 0,001), et Q : -0,214 (ES = 0,120, p = 0,075), indiquant une compétence initiale relativement élevée (I), une diminution linéaire (S), et une variation quadratique (Q) non significative. La Figure 4.4 illustre la trajectoire de la performance à la compétence 1 issue de ces paramètres pour les cinq temps de mesure.

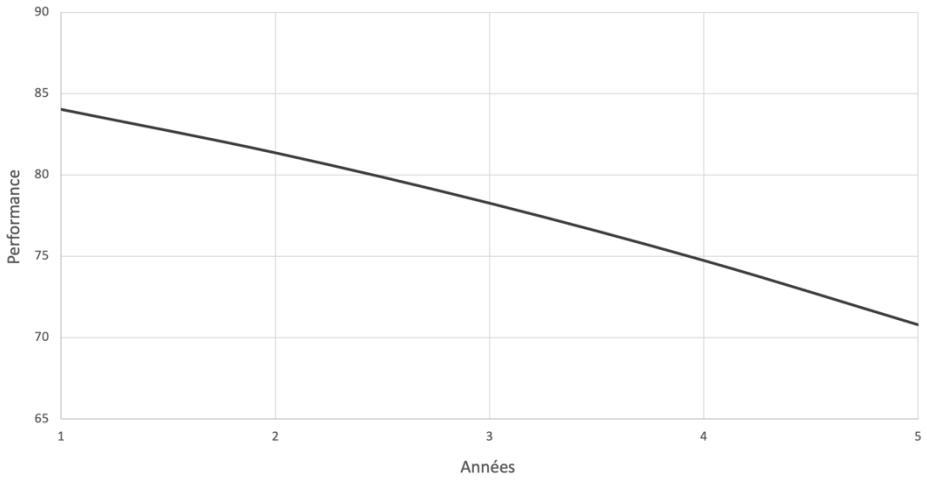

Figure 4.4 Trajectoire estimée du modèle de base pour la performance à la compétence 1 « Résoudre une situation-problème » en mathématiques.

L'ajustement du modèle multigroupe a également été évalué en utilisant les mêmes indices : χ^2 (18) = 147,172, p = 0,000, RMSEA = 0,101; CFI = 0,921; TLI = 0,905. L'ensemble de ces mesures montre que le modèle multigroupe présente un meilleur ajustement que le modèle de base, ce qui indique que l'ajout de la variable de regroupement contribue à une meilleure représentation des données. Cela suggère que donc que la variable de regroupement améliore l'ajustement du modèle avec les données.

Les paramètres I, S et Q ont été estimés pour chaque concentration : en Programmation, le I présente une moyenne estimée à 82,052 (SE = 0,959, $p < 0,001$), le S à -1,466 (SE = 0,342, $p = 0,019$) et le Q à -0,131 (SE = 0,262, $p = 0,617$). En Arts, les paramètres sont les suivants : I : 86,155 (SE = 0,599, $p < 0,001$); S : - 2,114 (SE = 0,620, $p = 0,001$), et Q : -0,370 (SE = 0,161, $p = 0,021$). Enfin, pour la concentration Sports, le I est estimé à 82,836 (SE = 0,938, $p < 0,001$), le S à -4,269 (SE = 0,889, $p < 0,001$), et le Q à 0,038 (SE = 0,246, $p = 0,878$). La Figure 4.5 illustre les trajectoires issues de ces paramètres pour chaque concentration au regard de la performance à la compétence 1 en mathématiques.

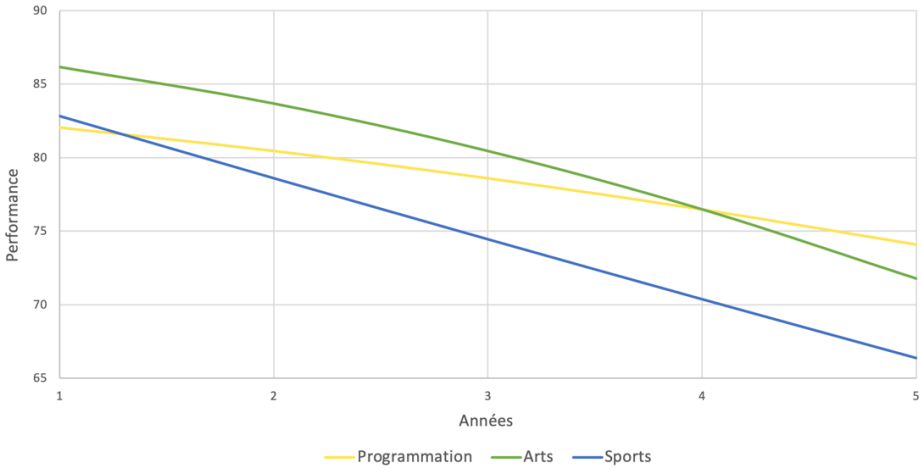

Figure 4.5 Trajectoires des trois concentrations pour la performance à la compétence 1 « Résoudre une situation-problème » en mathématiques.

Cette figure permet de constater qualitativement que la trajectoire en programmation semble associée à un plus petit déclin de la performance à travers le temps que les deux autres concentrations. De plus, malgré que la performance au T1 (ordonnée à l'origine) de la concentration Programmation soit plus basse que pour les deux autres concentrations, la performance observée au T5 est par ailleurs plus élevée. Considérant que l'ajustement du modèle multigroupe est supérieur au modèle de base, la section suivante vise à valider statistiquement ces différences.

4.1.2.2 Comparaison des trajectoires des concentrations

Puisque le modèle multigroupe présente un meilleur ajustement aux données que le modèle de base, des tests de Wald ont pu être réalisés afin de révéler si des différences significatives entre les paramètres des concentrations étaient observables. Le Tableau 4.2 détaille les valeurs de la statistique de Wald (W) et les valeurs *p* correspondantes pour l'ensemble des comparaisons réalisées.

Tableau 4.2 Résultats des tests comparatifs de Wald entre les paramètres des trajectoires des concentrations pour la performance à la compétence 1 «Résoudre une situation-problème » en mathématiques

Variables latentes	Comparaisons	Statistique de Wald (W)	*p*
Ordonnée à l'origine (I)	Prog. vs Arts	13.17	0.000*
	Prog. vs Sports	0.34	0.559
	Arts vs Sports	8.89	0.003*
Pente linéaire (S)	Prog. vs Arts	0.33	0.566
	Prog. vs Sports	4.68	0.030*
	Arts vs Sports	3.95	0.047*
Pente quadratique (Q)	Prog. vs Arts	1.136	0.041*
	Prog. vs Sports	0.22	0.638
	Arts vs Sports	1.146	0.026*

*Différence statistiquement significative à un seuil de $p < 0,05$.

Pour l'ordonnée à l'origine (I), des différences statistiquement significatives ont été observées entre les concentrations Programmation et Arts (W = 13,17, $p = 0,000$), ainsi qu'entre les concentrations Arts et Sports (W = 8,89, $p = 0.003$). Cependant, aucune différence significative n'a été détectée entre les concentrations Programmation et Sports (W = 0,34, $p = 0,559$), révélant que la concentration Arts était initialement plus performante à la compétence 1 que les deux autres concentrations. Pour la pente linéaire (S), des différences significatives ont été identifiées entre les concentrations Programmation et Sports (W = 4,68, $p = 0,030$), et entre Arts et Sports (W = 3,95, $p = 0,047$). Enfin, des différences significatives ont été identifiées pour la pente quadratique (Q) entre les concentrations Programmation et Arts (W = 1,136, p = 0,041) et entre Arts et Sports (W = 1,146, p = 0,026). Ces tests ont permis de confirmer que la concentration Programmation est associée à un déclin linéaire (S) statistiquement moins grand qu'en Sports, ainsi qu'à une accélération de ce déclin (Q) statistiquement moins importante qu'en Arts.

Encore une fois, considérant que les différences observées entre les ordonnées à l'origine des concentrations n'étaient pas d'intérêt, la Figure 4.6 présente les trois trajectoires rapportées à une même ordonnée à l'origine. Ce graphique permet de mettre davantage en évidence que le déclin linéaire (S) est moins prononcé pour la concentration Programmation que la concentration Sports. De plus, la différence relativement à la pente quadratique (Q) entre les concentrations Programmation et Arts est aussi plus

facile à observer sur cette figure où l'on voit clairement que l'accélération du déclin est beaucoup plus prononcée pour la concentration Arts comparativement à Programmation. Enfin, bien que la Figure 4.5 permettait déjà de constater que la concentration Programmation était celle qui terminait au T5 avec la meilleure performance à la compétence 1, la Figure 4.6 qui rapporte à un même point de départ la performance au T1 met davantage en évidence que la concentration Programmation est celle qui est associée au plus faible déclin observé au fil du temps et à la meilleure performance à la compétence 1 au T5. Qui plus est, il est intéressant de constater qu'au T3, qui marque la fin de l'activité des concentrations, des différences semblent présentes entre les concentrations, mais ces différences s'accentuent davantage entre la programmation et les autres concentrations jusqu'au T5, soit jusqu'à deux ans après la fin de l'activité des concentrations.

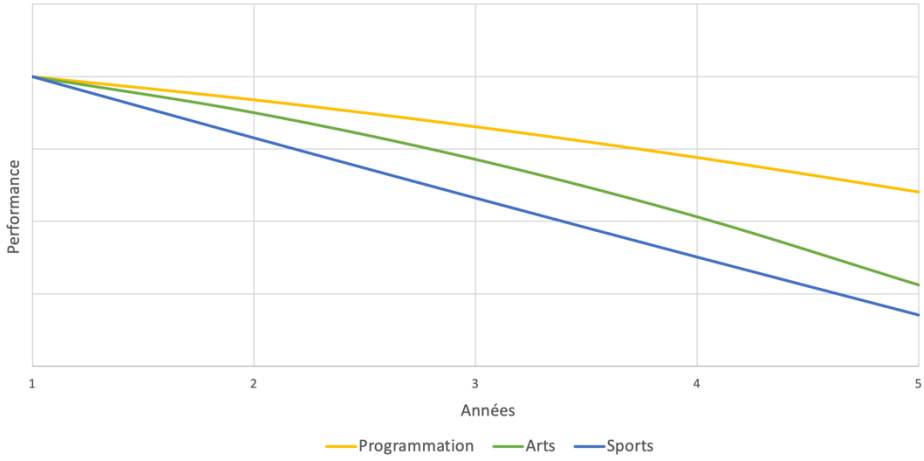

Figure 4.6 Trajectoires des trois concentrations pour la performance à la compétence 1 « Résoudre une situation-problème » en mathématiques, représentées à partir d'une même ordonnée à l'origine.

4.1.3 Mathématiques : Compétence 2 « Déployer un raisonnement mathématique »

4.1.3.1 Comparaison de l'ajustement des modèles

Plusieurs indices statistiques ont de nouveau été utilisés afin d'évaluer l'ajustement du modèle de base pour la compétence 2. L'ensemble de ces indices souligne un faible ajustement du modèle de base : χ^2 (11) = 126,856, p = 0,000, RMSEA = 0,155; CFI = 0,936; TLI = 0,912.

Les estimations des paramètres de ce modèle de base étaient respectivement de I : 76,466 (SE = 0,600, p < 0,001), S : 0,284 (SE = 0,352, p = 0,420), et Q : -0,570 (SE = 0,093, p < 0,001). La Figure 4.7 ci-dessous

représente graphiquement la trajectoire issue des paramètres du modèle de base pour la performance à la compétence 2 en mathématiques, où il est possible de constater une accélération du déclin de la performance au fil du temps.

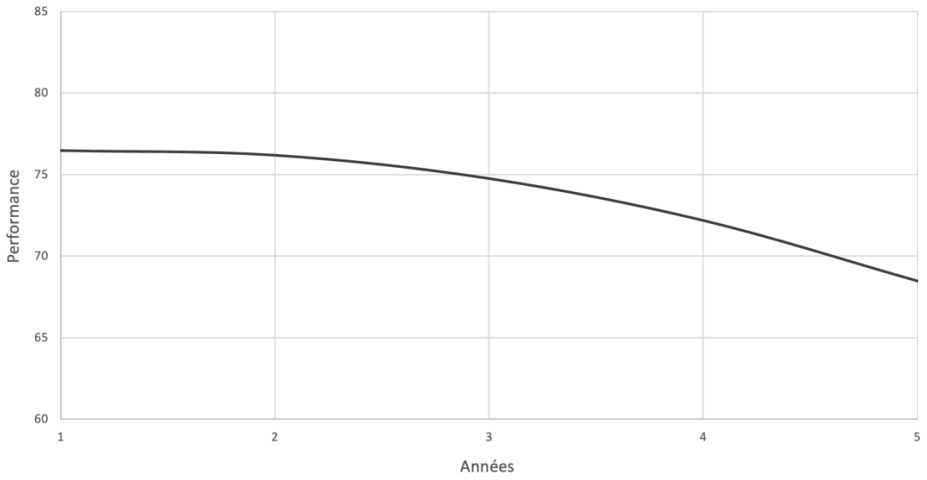

Figure 4.7 Trajectoire estimée du modèle de base pour la performance à la compétence 2 « Déployer un raisonnement mathématique ».

L'ajustement du modèle multigroupe a été évalué en utilisant les mêmes indices : χ^2 (18) = 121,833, p = 0,000, RMSEA = 0,198; CFI = 0,922; TLI = 0,903. L'ensemble de ces indices indique un ajustement global moins favorable que celui du modèle de base, ce qui signifie que la variable de regroupement des concentrations ne permet pas de mieux représenter les données.

Concernant les estimations des paramètres au sein des différents groupes, pour la concentration Programmation, les moyennes estimées pour l'ordonnée à l'origine (I), la pente linéaire (S) et la pente quadratique (Q) sont respectivement de 77,388 (ES = 1,279, p < 0,001), -1,286 (ES = 0,867, p = 0,138) et - 0,182 (ES = 0,217, p = 0,401). En Arts, ces valeurs sont respectivement de 78,433 (ES = 0,769, p < 0,001), 0,507 (ES = 0,463, p = 0,274) et -0,606 (ES = 0,119, p < 0,001). Enfin, en Sports, ces valeurs sont de 71,808 (ES = 1,194, p < 0,001), 2,357 (ES = 0,731, p = 0,001), et -1,076 (ES = 0,202, p < 0,001). La Figure 4.8 illustre les trajectoires issues de ces paramètres.

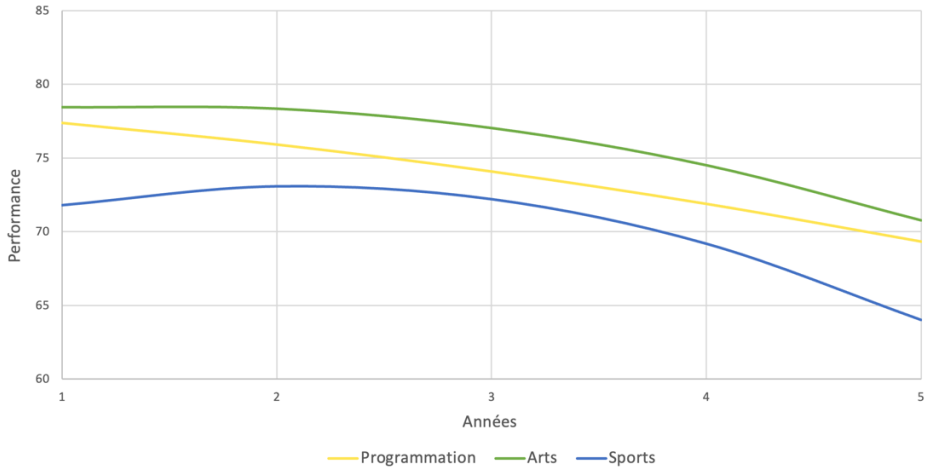

Figure 4.8 Trajectoire des trois concentrations pour la performance à la compétence 2 « Déployer un raisonnement mathématiques ».

À première vue, la figure semble suggérer que la concentration Programmation est associée à un déclin plus linéaire que les deux autres concentrations et que les trajectoires des trois concentrations sont suffisamment distinctes. Néanmoins, considérant que le modèle multigroupe ne présente pas un meilleur ajustement que le modèle de base, il n'est pas souhaitable de réaliser les tests comparatifs de Wald. En effet, si le modèle de base indique une meilleure capacité à expliquer les données, il en résulte logiquement que les distinctions entre les groupes dans le modèle multigroupe ne sont probablement pas statistiquement significatives. Les analyses par trajectoires latentes multigroupe comportent en effet un mécanisme de correction relatif au nombre de paramètres inclus dans le modèle. Ainsi, l'introduction de variables supplémentaires qui n'ont pas d'effet significatif sur l'explication de la variation des données est pénalisée, entraînant des indices d'ajustement moins favorables. Ce principe, souvent désigné sous le terme de « parcimonie du modèle » (Burnhame et Anderson, 2002), favorise les modèles plus simples et efficaces en terme explicatif, tout en évitant le surajustement potentiel induit par l'ajout de paramètres superflus. Par conséquent, si l'introduction de la variable de regroupement des concentrations dans le modèle multigroupe ne conduit pas à une amélioration significative de l'ajustement par rapport au modèle de base, cela peut indiquer que la variable en question n'apporte pas d'information utile pour expliquer ou prédire la variance observée dans les données. Cela peut résulter en des indices d'ajustement relativement moins favorables pour le modèle multigroupe, reflétant ainsi la préférence pour un modèle plus simple et économique en termes de nombre de paramètres. En d'autres termes, dans ce cas-ci, la variable de regroupement utilisée dans le modèle multigroupe ne contribue pas à améliorer de manière significative l'ajustement des données comparativement à la trajectoire globale offerte par le modèle de

base. La réalisation de tests de Wald dans ce contexte pourrait non seulement s'avérer superflue, mais également induire en erreur en suggérant des différences de groupe significatives là où elles n'existent pas.

4.2 Sciences

Tel que présenté au Tableau 3.3, deux volets font partie de l'évaluation au sein des bulletins scolaires, soit le volet Pratique et le volet Théorie. Ces deux volets sont respectivement associés à la compétence 1 « Chercher des réponses ou des solutions à des problèmes d'ordre scientifique ou technologique » et à la compétence 2 « Mettre à profit ses connaissances scientifiques et technologiques ». La performance globale en sciences équivaut à la moyenne pondérée des résultats de fin d'année scolaire obtenus pour la compétence 1 (pondération de 40 %) et la compétence 2 (pondération de 60 %).

4.2.1 Performance globale

4.2.1.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la performance globale en sciences a d'abord été conduite : χ^2 (11) = 263,585, p < 0,0000; RMSEA = 0,228, CFI = 0,858, TLI = 0,807. L'ensemble de ces indices indique un faible ajustement entre les données observées et celles prédites par le modèle de base.

Les paramètres (I, S et Q) du modèle ont été estimés. L'estimation pour I est de 75,009 (SE = 0,451, p < 0,0000), reflétant le niveau initial en sciences, pour S de 1,218 (SE = 0,348, p < 0,0000), indiquant une augmentation linéaire de la performance au fil du temps, et pour Q de -0,267 (SE = 0,102, p = 0,009), suggérant un ralentissement de cette augmentation de la performance en sciences au fil du temps. La Figure 4.9 illustre la trajectoire estimée issue de ces paramètres pour la performance globale en sciences. À première vue, il semble que la performance globale en sciences demeure relativement stable à travers les cinq années.

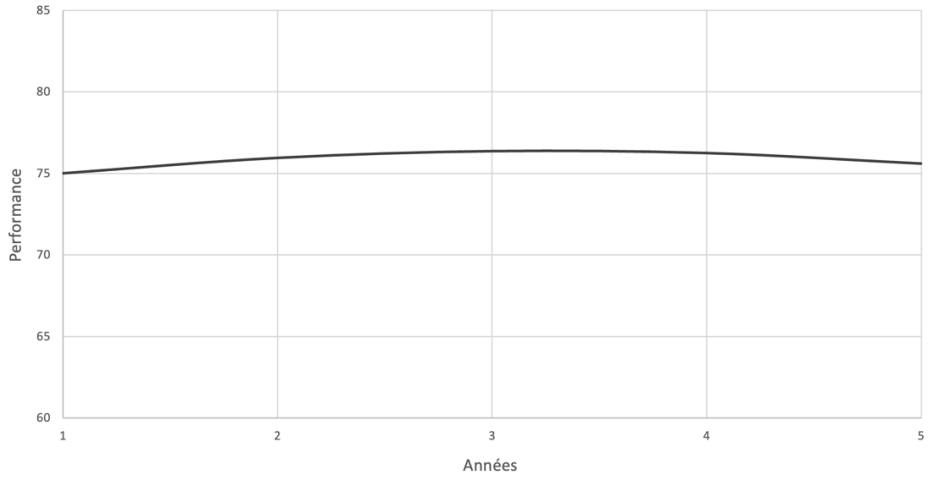

Figure 4.9 Trajectoire estimée du modèle de base pour la performance globale en sciences.

L'ajustement du modèle multigroupe a ensuite été évalué à l'aide des mêmes indices d'ajustement : χ^2 (18) = 255,767, p < 0,0000; RMSEA = 0,300, CFI = 0,864, TLI = 0,774. L'ensemble de ces indices indique une divergence encore plus grande que le modèle de base entre les données observées et celles prédites par le modèle multigroupe, ce qui signifie que la variable de regroupement des concentrations ne permet pas de mieux modéliser les données de performance globale en sciences.

Pour la concentration Programmation, le I est estimé à 73,978 (SE = 0,835, p < 0,001), le S est estimé à 2,946 (SE = 0,580, p < 0,001), et le Q à -0,689 (SE = 0,161, p < 0,001). Pour la concentration Arts, le I est estimé à 79,009 (SE = 0,550, p < 0,001), le S à 0,929 (SE = 0,456, p = 0,042), et le Q à -0,384 (SE = 0,131, p = 0,003). Enfin, pour la concentration Sports, le I est estimé à 69,667 (SE = 0,879, p < 0,001), le S à 1,723 (SE = 0,470, p = 0,028), et le Q à -0,456 (SE = 0,170, p = 0,004). La Figure 4.10 illustre les trajectoires issues de ces paramètres pour chaque concentration au regard de la performance globale en sciences.

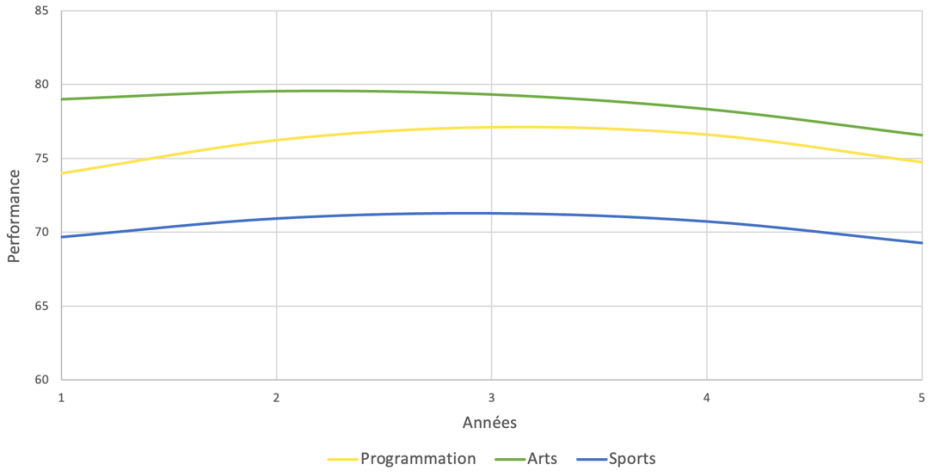

Figure 4.10 Trajectoires des trois concentrations pour la performance globale en sciences.

Cette figure permet de constater que les trajectoires apparaissent assez similaires; elles évoluent en effet de manière relativement parallèle à travers le temps, ce qui illustre que les concentrations ne semblent pas associées à une évolution différente de la performance globale en science au fil du temps. De nouveau, considérant que le modèle de base présente un meilleur ajustement que le modèle multigroupe, il n'a pas été possible de procéder à la comparaison des trajectoires des concentrations à l'aide des tests de Wald.

4.2.2 Sciences : Compétence 1 « Chercher des réponses ou des solutions à des problèmes »

4.2.2.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la compétence 1 en sciences a d'abord été conduite : χ^2 (11) = 112,511, p < 0,0000; RMSEA = 0,145, CFI = 0,874, TLI = 0,828. L'ensemble de ces indices indique un faible ajustement entre les données observées et celles prédites par le modèle de base.

Les paramètres (I, S et Q) du modèle ont été estimés. L'estimation pour I est de 76,741 (SE = 0,381, p < 0,000), reflétant le niveau initial en sciences, pour S de 3,511 (SE = 0,290, p < 0,0000), indiquant une augmentation linéaire de la performance au fil du temps, et pour Q de -0,975 (SE = 0,081, p = 0,000), suggérant un déclin de cette augmentation au fil du temps. La Figure 4.11 illustre la trajectoire estimée issue de ces paramètres pour la compétence 1 en sciences.

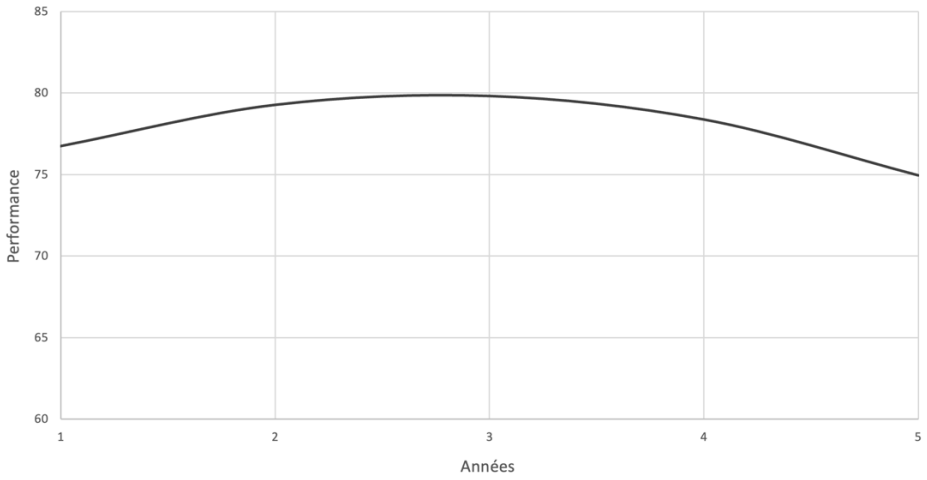

Figure 4.11 Trajectoire estimée du modèle de base pour la performance à la compétence 1 « Chercher des réponses ou des solutions à des problèmes » en sciences.

Tout comme pour la performance globale en science, cette trajectoire semble à première vue indiquer que la performance à la compétence 1 en sciences est demeurée relativement stable à travers les 5 années, avec une augmentation plus prononcée de la performance à cette compétence vers la fin de la 3^{e} année du secondaire, comparativement à la performance globale.

L'ajustement du modèle multigroupe a été évalué avec les mêmes mesures d'ajustement : χ^2 (18) = 125,151, p < 0,000; RMSEA = 0,201, CFI = 0,861, TLI = 768. L'ensemble de ces indices indique une divergence entre les données observées et celles prédites par le modèle multigroupe encore plus importante que celle du modèle de base, ce qui signifie que l'ajout de la variable de regroupement des concentrations n'améliore pas significativement la prédiction du modèle pour la compétence 1 en sciences.

Concernant les estimations des paramètres au sein des différents groupes, pour la concentration Programmation, les moyennes estimées pour l'ordonnée à l'origine (I), la pente linéaire (S) et la pente quadratique (Q) sont respectivement de 76,733 (SE = 0,662, p < 0,001), 3,526 (SE = 0,470, p < 0,001) et - 0,992 (SE = 0,130, p < 0,001). En Arts, ces valeurs sont respectivement de 79,667 (SE = 0,515, p < 0,001), 2,206 (SE = 0,450, p < 0,001) et -0,711 (SE = 0,122, p < 0,001). Enfin, en Sports, ces valeurs sont de 72,449 (SE = 0,719, p < 0,001), 5,139 (SE = 0,593, p < 0,001), et -1,258 (SE = 0,184, p < 0,001). La Figure 4.12 illustre les trajectoires issues de ces paramètres.

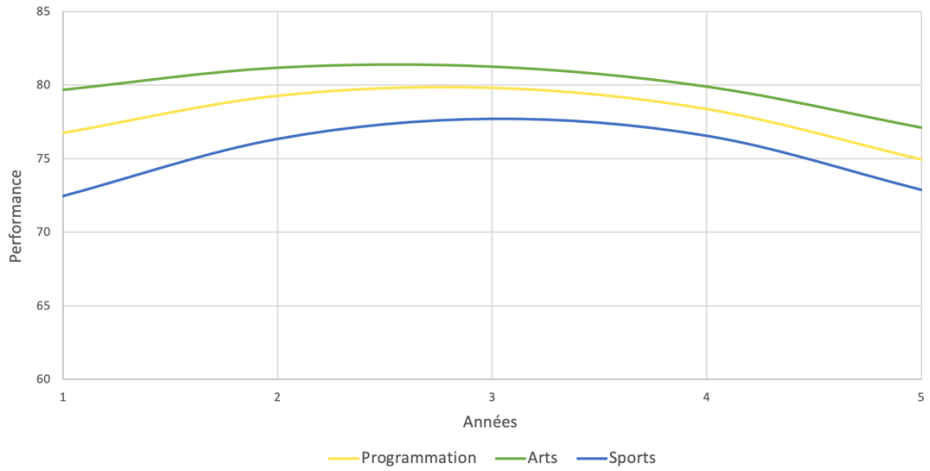

Figure 4.12 Trajectoires des trois concentrations pour la performance à la compétence 1 « Chercher des réponses ou des solutions à des problèmes » en sciences.

Cette figure permet de mettre en évidence que les trajectoires des concentrations apparaissent très similaires, et qu'il est donc peu probable que des différences significatives soient observables entre celles-ci. Cette grande uniformité entre les trajectoires des concentrations permet de mieux comprendre pourquoi le modèle multigroupe ne propose pas un meilleur ajustement que le modèle de base.

4.2.3 Sciences : Compétence 2 « Mettre à profit ses connaissances scientifiques »

4.2.3.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la compétence 2 en sciences a d'abord été conduite : χ^2 (11) = 423,027, p < 0,000; RMSEA = 0,292, CFI = 0,761, TLI = 0,675. L'ensemble de ces indices indique un faible ajustement entre les données observées et celles prédites par le modèle de base.

L'estimation des paramètres indique que le I est de 73,623 (SE = 0,539, p < 0,000), reflétant le niveau initial en sciences, le S de 1,311 (SE = 0,463, p = 0,005), indiquant une légère augmentation linéaire de la performance au fil du temps, et le Q de -0,208 (SE = 0,142, p = 0,143), suggérant un très faible ralentissement (non significatif) de la performance au fil du temps. La Figure 4.13 illustre la trajectoire estimée issue de ces paramètres pour la performance à la compétence 2 en sciences.

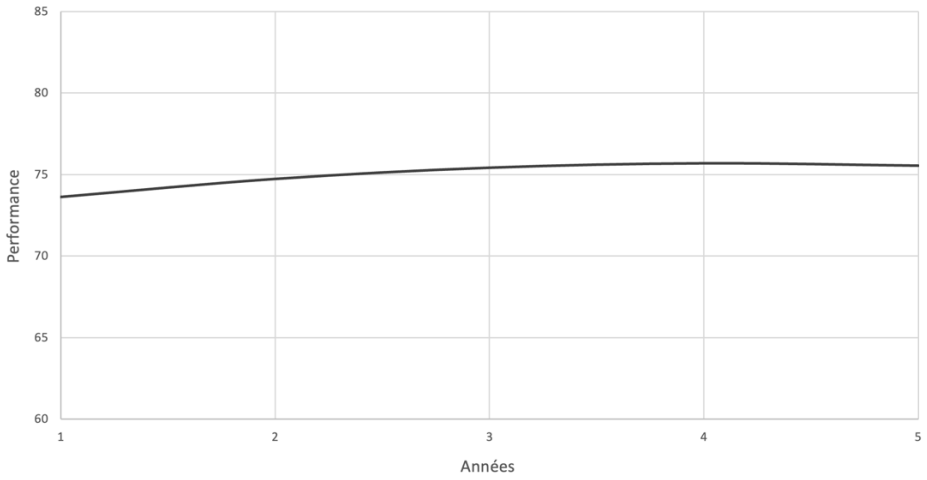

Figure 4.13 Trajectoire estimée du modèle de base pour la performance à la compétence 2 « Mettre à profit ses connaissances scientifiques ».

L'ajustement du modèle multigroupe a ensuite été évalué avec les mêmes mesures d'ajustement : χ^2 (18) = 387,754, p < 0,0000; RMSEA = 0,374, CFI = 0,781, TLI = 635. L'ensemble de ces indices indique un ajustement encore plus faible du modèle multigroupe que celui du modèle de base entre les données observées et celles prédites par le modèle multigroupe, ce qui signifie que la variable de regroupement des concentrations ne permet pas de mieux représenter les données.

Les paramètres I, S et Q ont été estimés pour chaque concentration. Pour la concentration Programmation, le I est estimé à 72,120 (SE = 1,020, p < 0,001), le S à 4,431 (SE = 0,741, p < 0,001), et le Q à -0,999 (SE = 0,208, p < 0,001). Pour la concentration Arts, le I est estimé à 78,012 (SE = 0,646, p < 0,001), le S à 2,079 (SE = 0,580, p < 0,001), et le Q à -0,741 (SE = 0,172, p < 0,001). Enfin, pour la concentration Sports, le I est estimé à 68,114 (SE = 1,071, p < 0,001), le S à 2,301 (SE = 0,491, p = 0,005), et le Q à -0,622 (SE = 0,135, p < 0,001). La Figure 4.14 illustre les trajectoires issues de ces paramètres pour chaque concentration pour la performance à la compétence 2 en sciences.

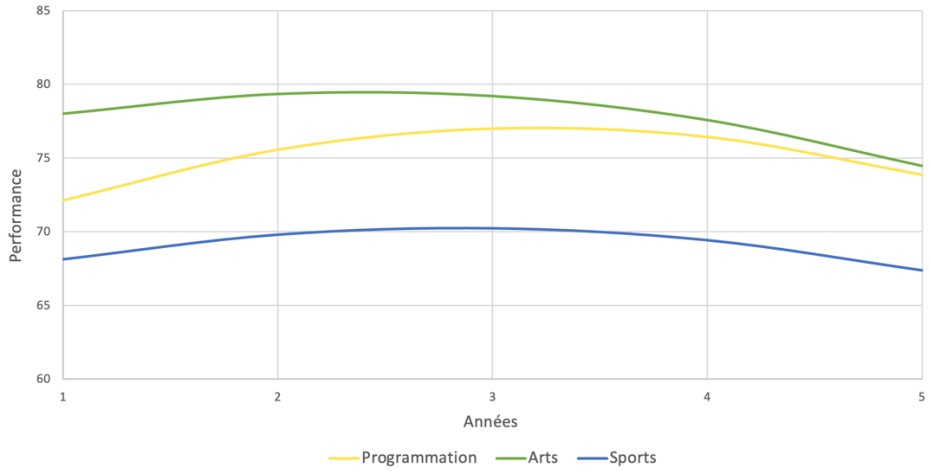

Figure 4.14 Trajectoires des trois concentrations pour la performance à la compétence 2 « Mettre à profit ses connaissances scientifiques ».

Globalement, cette figure permet de constater que les trajectoires des concentrations sont assez similaires, bien que celle de la concentration Programmation paraisse à première vue associée à un déclin moins important entre le T3 et le T5. Cependant, considérant que le modèle de base présente un meilleur ajustement que le modèle multigroupe, et donc que ces trajectoires représentent moins bien les données que la trajectoire du modèle de base, aucun test de Wald n'a pu être réalisé afin de comparer statistiquement ces trajectoires entre elles.

4.3 Français

Tel que présenté au Tableau 3.3, trois compétences en français font partie de l'évaluation au sein des bulletins scolaires, soit la compétence 1 « Lire », la compétence 2 « Écrire », et la compétence 3 « Communiquer oralement ». La performance globale en français équivaut à la moyenne pondérée des résultats de fin d'année scolaire obtenus pour la compétence 1 (pondération de 40 %), la compétence 2 (pondération de 40 %) et la compétence 3 (pondération de 20 %).

4.3.1 Performance globale

4.3.1.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la performance globale en français a d'abord été conduite : χ^2 (11) = 96,009, p < 0,000; RMSEA = 0,133, CFI = 0,950, TLI = 0,932. Bien que l'indice CFI se situe

au-dessus du seuil de 0,95, la majorité des indices suggère un faible ajustement entre les données observées et celles prédites par le modèle de base.

Les paramètres (I, S et Q) du modèle ont été estimés. L'estimation pour I est de 76,418 (ES = 0,416, $p < 0,000$), reflétant le niveau initial en français, pour S de -3,033 (ES = 0,247, $p < 0,000$), indiquant une diminution linéaire de la performance au fil du temps, et pour Q de 0,677 (ES = 0,060, $p < 0,000$), suggérant un léger ralentissement du déclin de la performance au fil du temps. La Figure 4.15 illustre la trajectoire estimée issue de ces paramètres pour la performance globale en français. À première vue, cette figure permet de constater que la performance globale en français demeure relativement stable à travers les cinq années, mais présente une légère diminution passagère de la performance vers la 3e année.

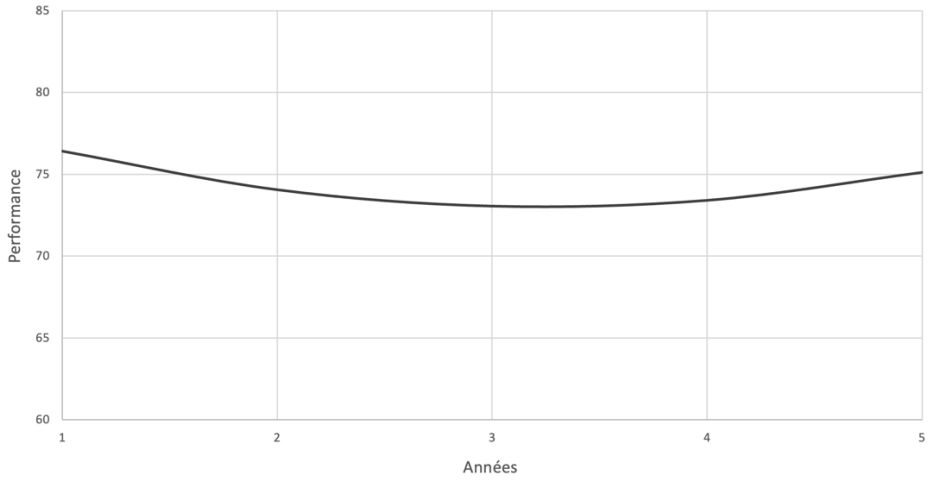

Figure 4.15 Trajectoire estimée du modèle de base pour la performance globale en français.

Les indices d'ajustement suivant ont été obtenus pour le modèle multigroupe : χ^2 (18) = 95,107, $p < 0,0000$; RMSEA = 0,171, CFI = 0,955, TLI = 924. Bien que le CFI soit légèrement plus haut que celui du modèle de base, le test du Chi-carré demeure toujours significatif, le RMSEA est plus haut et le TLI est plus bas, ce qui indique un ajustement légèrement plus faible du modèle multigroupe comparé à celui du modèle de base, ce qui signifie que l'ajout de la variable de regroupement des concentrations ne permet pas de mieux représenter les données.

Pour la concentration Programmation, la variable I a été estimée à 74,693 (ES = 0,799, $p < 0,001$), le S à - 2,457 (EES = 0,519, $p < 0,001$), et le Q à 0,516 (ES = 0,125, $p < 0,001$). Pour la concentration Arts, le I est

estimé à 77,735 (ES = 0,596, $p < 0,001$), le S à -1,995 (ES = 0,302, $p < 0,001$), et le Q à 0,506 (ES = 0,075, $p < 0,001$). Enfin, pour la concentration Sports, le I est de 76,043 (ES = 0,803, $p < 0,001$), le S de -5,603 (ES = 0,509, $p < 0,001$), et le Q de 1,181 (ES = 0,131, $p < 0,001$). La Figure 4.16 illustre les trajectoires issues de ces paramètres pour chaque concentration pour la performance globale en français.

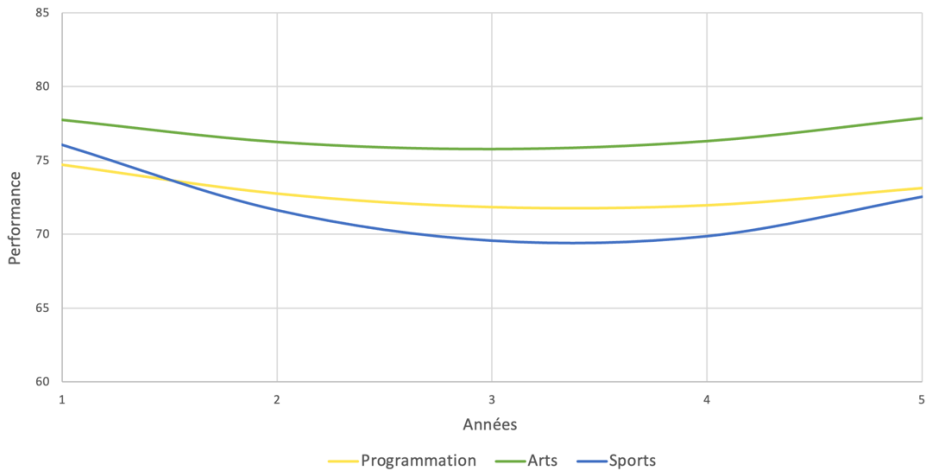

Figure 4.16 Trajectoires des trois concentrations pour la performance globale en français.

Considérant que le modèle de base présente un meilleur ajustement aux données que le modèle multigroupe, il n'a pas été possible de procéder à une comparaison statistique des trajectoires des concentrations à l'aide des tests de Wald.

4.3.2 Français : Compétence 1 « Lire »

4.3.2.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la compétence 1 en français a révélé les indices d'ajustement suivants : χ^2 (11) = 63,588, $p < 0,000$; RMSEA = 0,104, CFI = 0,946, TLI = 0,927. L'ensemble de ces indices indique un faible ajustement entre les données observées et celles prédites par le modèle de base.

L'estimation des paramètres du modèle révèle que le I est de 73,831 (ES = 0,530, $p < 0,000$), reflétant le niveau initial en français, le S de -2,178 (ES = 0,388, $p < 0,000$), indiquant une diminution linéaire de la performance au fil du temps, et le Q de 0,613 (ES = 0,090, $p < 0,000$), suggérant un ralentissement du

déclin de la performance au fil du temps. La Figure 4.17 illustre la trajectoire estimée issue de ces paramètres pour la performance à la compétence 1 en français.

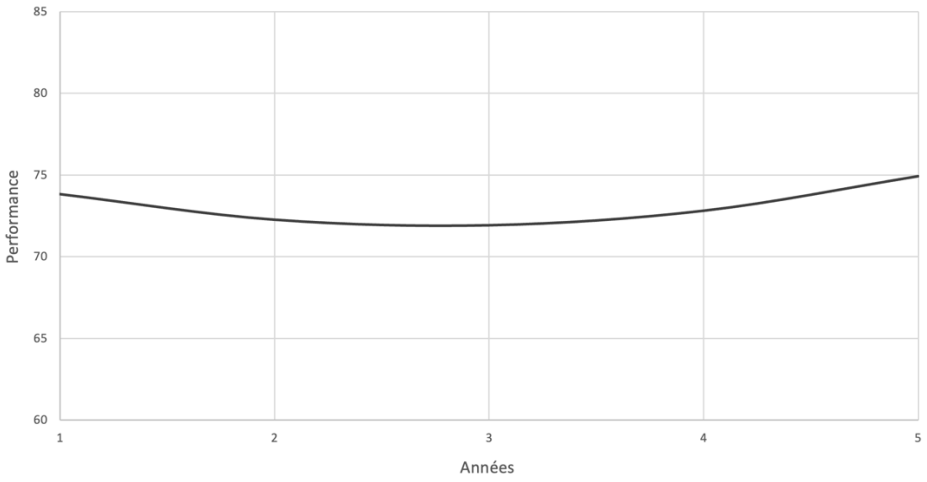

Figure 4.17 Trajectoire estimée du modèle de base pour la performance à la compétence 1 « Lire » en français.

L'ajustement du modèle multigroupe a ensuite été évalué avec les mêmes mesures d'ajustement : χ^2 (18) = 71,756, p < 0,0000; RMSEA = 0,143, CFI = 0,945, TLI = 909. L'ensemble de ces indices indique un ajustement légèrement plus faible du modèle multigroupe que celui du modèle de base, ce qui signifie que la variable de regroupement des concentrations ne permet pas de mieux modéliser les données de performance pour cette compétence.

Pour la concentration Programmation, le I a été estimé à 72,929 (ES = 1,059, p < 0,001), le S à -2,629 (EES = 0,751, p < 0,001), et le Q à 0,643 (ES = 0,173, p < 0,001). Pour la concentration Arts, le I est estimé à 74,697 (ES = 0,712, p < 0,001), le S est estimé à -0,740 (ES = 0,465, p = 0,112), et le Q est estimé à 0,365 (ES = 0,110, p = 0,001). Enfin, pour la concentration Sports, le I est de 73,685 (ES = 1,095, p < 0,001), le S est de -4,351 (ES = 0,907, p < 0,001), et le Q est de 0,984 (ES = 0,217, p < 0,001). La Figure 4.18 illustre les trajectoires issues de ces paramètres pour chaque concentration pour la compétence 1 en français.

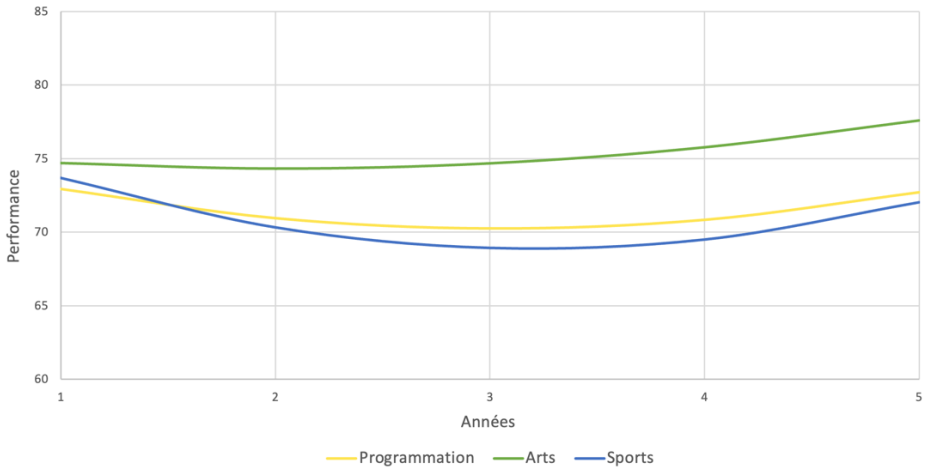

Figure 4.18 Trajectoires des trois concentrations pour la performance à la compétence 1 « Lire » en français.

De nouveau, considérant que le modèle de base présente un meilleur ajustement que le modèle multigroupe, il n'a pas été possible de conduire les tests de Wald visant à comparer statistiquement les trajectoires des concentrations.

4.3.3 Français : Compétence 2 « Écrire »

4.3.3.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la compétence 2 en français a mené à l'obtention des indices d'ajustement suivants : χ^2 (11) = 275,159, p < 0,000; RMSEA = 0,234, CFI = 0,847, TLI = 0,792. L'ensemble de ces indices suggère un faible ajustement entre les données observées et celles prédites par le modèle de base.

Les paramètres de base ont été estimés de la manière suivante : I : 76,167 (ES = 0,541, p < 0,000), reflétant le niveau initial de la compétence 1 en français; S : -3,795 (ES = 0,309, p < 0,000), indiquant une diminution linéaire de la performance au fil du temps, et Q : 0,806 (ES = 0,076, p < 0,000), suggérant un ralentissement du déclin de la performance au fil du temps. La Figure 4.19 illustre la trajectoire estimée issue de ces paramètres pour la performance associée à la compétence 2 en français.

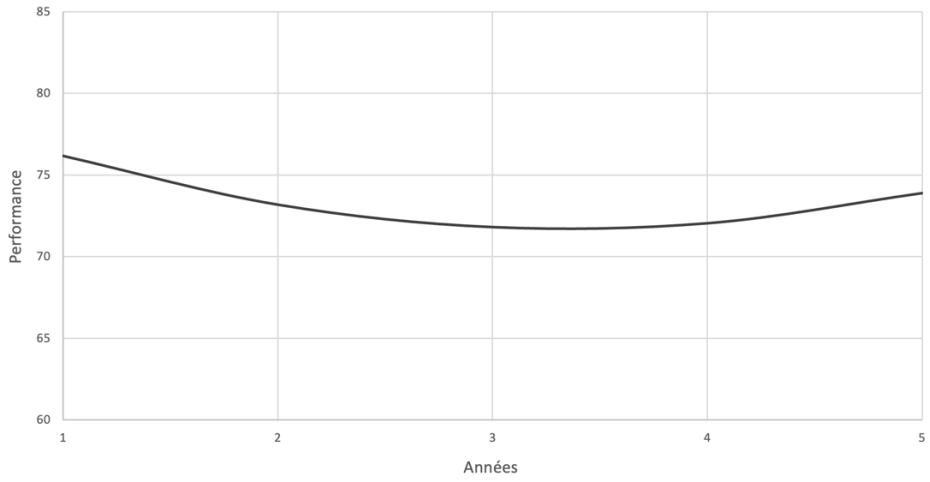

Figure 4.19 Trajectoire estimée du modèle de base de la compétence 2 « Écrire » en français.

Les analyses d'ajustement du modèle multigroupe ont pour leur part révélé les indices d'ajustement suivants : χ^2 (18) = 254,088, p < 0,0000; RMSEA = 0,299, CFI = 0,861, TLI = 768. Les indices indiquent un ajustement globalement plus faible que celui du modèle de base entre les données observées et celles prédites par le modèle multigroupe, ce qui signifie que l'ajout de la variable de regroupement des concentrations ne permet pas de mieux modéliser les données.

Les paramètres I, S et Q ont été estimés pour chaque concentration. Pour la concentration Programmation, le I est estimé à 72,598 (ES = 1,001, p < 0,001), le S à -1,744 (EES = 0,638, p = 0,006), et le Q à 0,350 (ES = 0,162, p = 0,031). Pour la concentration Arts, le I est de 78,685 (ES = 0,812, p < 0,000), le S de -3,328 (ES = 0,425, p < 0,000), et le Q de 0,727 (ES = 0,103, p < 0,000). Enfin, pour la concentration Sports, le I est estimé à 74,593 (ES = 0,953, p < 0,001), le S à -5,969 (ES = 0,566, p < 0,001), et le Q à 1,298 (ES = 0,144, p < 0,001). La Figure 4.20 illustre les trajectoires issues de ces paramètres pour chaque concentration au regard de la compétence 2 en français.

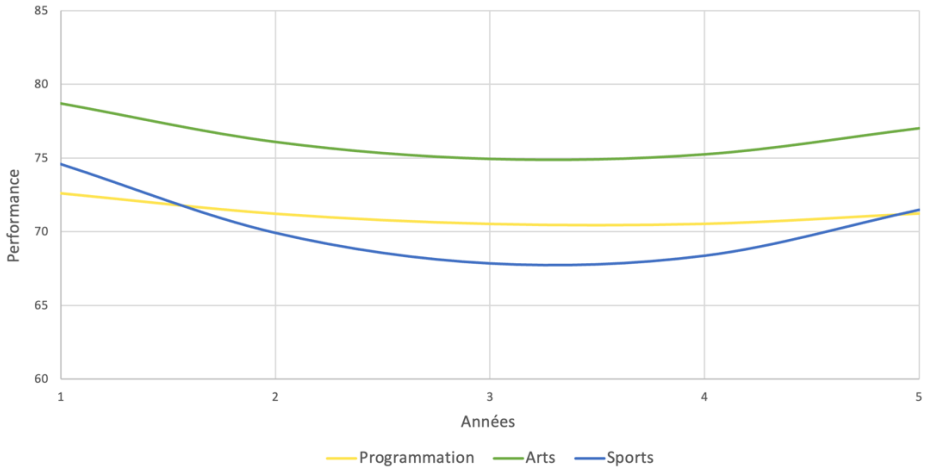

Figure 4.20 Trajectoires des trois concentrations pour la performance à la compétence 2 « Écrire » en français.

À la lumière des analyses réalisées pour comparer les ajustements des modèles, il ressort que le modèle de base présente un meilleur ajustement que le modèle multigroupe, ce qui ne permet donc pas de procéder à une comparaison statistique des trajectoires des concentrations à l'aide des tests de Wald.

4.3.4 Français : Compétence 3 « Communiquer oralement »

4.3.4.1 Comparaison de l'ajustement des modèles

L'évaluation de l'ajustement du modèle de base pour la compétence 3 en français a d'abord été conduite : χ^2 (11) = 81,610, p < 0,000; RMSEA = 0,121, CFI = 0,878, TLI = 0,834. L'ensemble de ces indices indique un faible ajustement entre les données observées et celles prédites par le modèle de base.

Les paramètres (I, S et Q) du modèle sont de I : 82,307 (ES = 0,332, p < 0,000), reflétant le niveau initial en français, S : -3,774 (ES = 0,388, p < 0,000), indiquant une diminution linéaire de la performance au fil du temps, et Q : 1,033 (ES = 0,100, p < 0,000), suggérant un ralentissement du déclin de la performance au fil du temps. La Figure 4.21 illustre la trajectoire estimée issue de ces paramètres pour la performance à la compétence 3 en français.

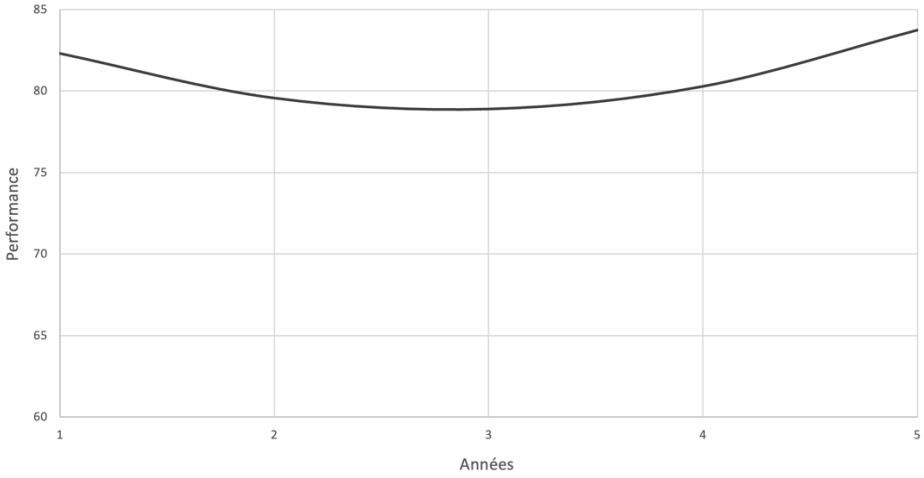

Figure 4.21 Trajectoire estimée du modèle de base pour la performance à la compétence 3 « Communiquer oralement » en français.

L'ajustement du modèle multigroupe a ensuite été évalué à l'aide des tests usuels et à mené aux indices suivants : χ^2 (18) = 106,361, $p < 0,0000$; RMSEA = 0,183, CFI = 0,847, TLI = 745. L'ensemble de ces indices indique un ajustement encore plus faible que celui du modèle de base entre les données observées et celles prédites par le modèle multigroupe, ce qui signifie que la variable de regroupement des concentrations ne contribue pas à une meilleure représentativité des données.

Concernant les estimations des paramètres au sein des différents groupes, pour la concentration Programmation, les moyennes estimées pour l'ordonnée à l'origine (I), la pente linéaire (S) et la pente quadratique (Q) sont respectivement de 81,095 (ES = 0,661, $p < 0,001$), -2,844 (ES = 0,809, $p < 0,000$) et 0,740 (ES = 0,220, $p = 0,001$). En Arts, ces valeurs sont respectivement de 82,012 (ES = 0,478, $p < 0,000$), - 2,196 (ES = 0,508, $p < 0,000$) et à 0,760 (ES = 0,128, $p < 0,000$). Enfin, en Sports, ces valeurs sont de 84,307 (ES = 0,675, $p < 0,001$), -7,595 (ES = 0,933, $p < 0,001$), et 1,882 (ES = 0,257, $p < 0,001$). La Figure 4.22 représente les trajectoires issues de ces paramètres pour chaque concentration pour la performance à la compétence 3 en français.

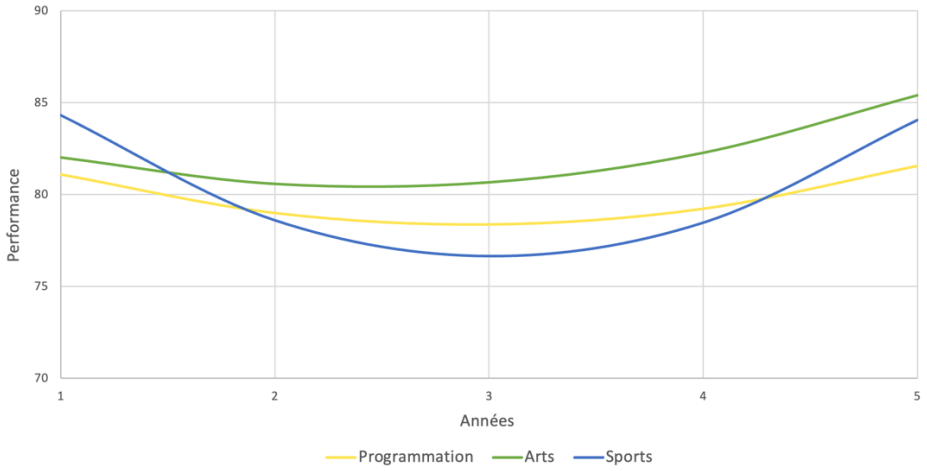

Figure 4.22 Trajectoires des trois concentrations pour la performance à la compétence 3 « Communiquer oralement » en français.

Les résultats des analyses comparatives d'ajustement des modèles ayant indiqué que le modèle de base présentait un meilleur ajustement aux donnés que le modèle multigroupe, la comparaison statistique des trajectoires des différentes concentrations par le biais des tests de Wald n'est donc pas justifiée.

4.4 Synthèse des résultats

En somme, les analyses réalisées dans cette étude ont permis de déterminer que la qualité de l'ajustement du modèle multigroupe s'avérait meilleure que celle du modèle de base uniquement pour la discipline des mathématiques. Pour cette dernière, le modèle multigroupe présentait en effet un ajustement supérieur au modèle de base pour les données relatives à la performance globale, ainsi qu'à la compétence 1 « Résoudre une situation-problème ». La comparaison des trajectoires des concentrations à l'aide des tests de Wald n'a donc été possible que pour ces données. Pour la performance globale, les trajectoires de l'ensemble des concentrations indiquent un déclin de la performance entre le T1 et le T5. Comparativement aux deux autres concentrations, bien que la concentration Programmation présente un déclin initial de la performance légèrement plus marqué, l'accélération de ce déclin entre le T1 et le T5 apparaît cependant moins importante. Pour la compétence 1, les résultats indiquent que la concentration Programmation est associée à un déclin linéaire statistiquement moins grand qu'en Sports, ainsi qu'à une accélération de ce déclin statistiquement moins importante qu'en Arts. Enfin, pour la compétence 2 en mathématique, le modèle multigroupe ne présentait pas un meilleur ajustement que pour le modèle de base, suggérant que l'ajout de la variable des concentrations ne permettait pas de mieux représenter la

trajectoire de la performance des élèves pour cette compétence. Cela souligne donc que le meilleur ajustement du modèle multigroupe pour la performance globale en mathématiques est principalement dû aux données de la compétence 1, les données de la compétence 2 ajoutant probablement du « bruit » à ce modèle.

Pour l'ensemble des données en sciences et en français, qu'il s'agisse de la performance globale ou des compétences évaluées pour chacune des disciplines, le modèle de base présentait toujours un ajustement supérieur au modèle multigroupe, suggérant que l'ajout de la variable de regroupement des concentrations ne permettait pas de mieux représenter les données. Étant donné que pour ces disciplines la modélisation par concentrations ne permettait pas de mieux représenter la trajectoire des données, il n'était pas justifié de comparer les différences entre les trajectoires des concentrations.

Le chapitre qui suit présente une discussion de ces résultats, et particulièrement des résultats les plus significatifs en mathématiques. Des pistes d'interprétation seront proposées en lien avec la littérature scientifique portant sur l'apprentissage de la programmation et le transfert des apprentissages.

CHAPITRE 5
DISCUSSION

Ce projet de recherche avait pour objectif de répondre à la question suivante : Quels sont les effets de transfert associés à l'apprentissage de la programmation chez les jeunes apprenants sur les disciplines scolaires traditionnelles ? Il visait spécifiquement à approfondir la compréhension des effets de transfert de la programmation vers les trois disciplines scolaires les plus centrales, soit les mathématiques, les sciences et le français. Pour ce faire, les résultats scolaires d'élèves du secondaire prenant part à une concentration scolaire en programmation ont été comparés à ceux d'élèves prenant part à deux autres concentrations scolaires (Arts et Sports), et ce, pour l'ensemble de leur parcours scolaire au secondaire. Les hypothèses formulées étaient que l'apprentissage de la programmation, comparativement aux deux autres concentrations, serait associé à un effet de transfert positif vers les mathématiques, à un effet de transfert positif, mais de moindre ampleur vers les sciences, et qu'aucun effet de transfert ne serait observé vers le français.

Ce chapitre de la thèse vise à proposer une discussion des résultats présentés au chapitre précédent, de façon à mettre en lumière les principaux éléments de réponse à la question de recherche. Dans un premier temps, un retour sur les principaux résultats est donc réalisé au regard des trois hypothèses formulées. Puis, ces résultats sont abordés, d'une part, en lien avec la littérature scientifique en didactique de la programmation et, d'autre part, en lien avec la littérature portant sur le transfert des apprentissages, de façon à dégager et discuter des pistes d'interprétation qui apparaissent les plus centrales. À la suite de l'interprétation des résultats, les principales limites de cette étude sont identifiées et détaillées de façon à circonscrire leur influence sur les résultats obtenus. Puis, différentes considérations éducatives sont abordées et discutées à la lumière des résultats. Des perspectives de recherches futures sont finalement présentées.

5.1 Retour sur la question et les hypothèses de recherche

Les résultats du projet permettent de fournir des éléments de réponse à la question de recherche : il semble en effet que le transfert découlant de l'apprentissage de la programmation varie selon la discipline scolaire cible. Les résultats obtenus confirment en partie certaines des hypothèses qui avaient été avancées. Ces hypothèses s'appuyaient d'une part sur différents ancrages théoriques relativement à la didactique de la programmation et au transfert des apprentissages. D'autre part, ils s'appuyaient

également sur les premiers résultats existants au regard du transfert de la programmation vers d'autres disciplines scolaires (Scherer *et al.*, 2019). À la lumière de ces éléments, les hypothèses suivantes avaient été formulées.

Pour les mathématiques, la littérature met en évidence plusieurs liens disciplinaires entre cette discipline et la programmation, notamment via le développement de la pensée informatique (Barr et Stephenson, 2011; Weintrop *et al.*, 2016). Qui plus est, il s'agit de la discipline scolaire la plus étudiée au regard du transfert possible depuis la programmation. De nombreuses études, incluant la méta-analyse de Scherer et ses collègues (2019) ont mis en lumière un transfert positif et significatif entre la programmation et les mathématiques. Pour ces raisons, l'hypothèse formulée était la suivante :

H1 - Des *effets de transfert positifs* seront observés entre l'apprentissage de la programmation et la performance scolaire en mathématiques. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à de *meilleurs résultats scolaires en mathématiques*, comparativement à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

⇒ Hypothèse **confirmée**

Les résultats obtenus tendent à confirmer cette première hypothèse. En effet, ces résultats indiquent d'abord que la modélisation incluant les concentrations permet de mieux représenter les données de performance scolaire en mathématiques, d'une part pour la performance globale, mais en particulier pour la compétence 1 « Résoudre une situation-problème ». Cela signifie donc que l'appartenance à une concentration influence les données de performance scolaire en mathématiques. En revanche, ce n'est pas le cas pour la compétence 2 « Déployer un raisonnement mathématique », ce qui suggère que le meilleur ajustement observé pour la performance globale en mathématiques est majoritairement dû aux résultats relatifs à la performance pour la compétence 1, même si celle-ci détient une pondération moins importante. Cela indique que l'appartenance à une concentration semble exercer une influence significative sur les données de performance scolaire en mathématiques, particulièrement pour la compétence 1. Qui plus est, pour cette compétence, les différences entre la concentration Programmation et les deux autres concentrations s'accentuent entre la fin de la troisième année (T3), qui marque la fin de l'activité des concentrations, et la fin de la cinquième année (T5), ce qui suggère que l'effet de transfert

positif associé à l'apprentissage de la programmation perdure jusqu'à deux ans après l'exposition à la concentration pour cette compétence.

Pour cette compétence précise, les comparaisons réalisées entre les trajectoires des différentes concentrations ont mené à l'obtention de différences statistiquement significatives qui vont dans le sens de l'hypothèse formulée. Bien que l'ensemble des concentrations soit associé à un déclin de la performance en mathématiques à travers le temps, la concentration Programmation est celle qui présente le déclin le moins prononcé. Ces différences au niveau de la trajectoire de la performance permettent de confirmer que la concentration Programmation est donc associée à de meilleurs résultats scolaires en mathématiques, comparativement à ceux d'élèves ayant participé aux deux autres concentrations scolaires, au regard de la compétence 1 en résolution de problèmes. Il semble donc que l'appartenance à la concentration Programmation mène à de meilleurs résultats scolaires au fil du temps, ce qui suggère un effet de transfert positif associé à l'apprentissage de la programmation pour la discipline des mathématiques. Ces résultats corroborent les conclusions de diverses études, notamment celle menée par Brown (2008), qui a constaté une amélioration des compétences en résolution de problèmes mathématiques chez des élèves âgés de 10 à 12 ans ayant participé à une intervention en programmation. Ils concordent également avec les conclusions de l'étude de Sáez-López (2016), qui a observé une amélioration des performances scolaires en mathématiques après l'intégration de la programmation dans le cursus scolaire des élèves du primaire. De plus, ils s'alignent sur les résultats de la méta-analyse menée par Scherer *et al.* (2019), indiquant une taille d'effet significative de $g = 0{,}57$ pour le transfert de la programmation vers les mathématiques.

Pour les sciences, les liens disciplinaires avec l'apprentissage de la programmation sont moins présents dans la littérature et les effets de transfert potentiels associés à l'apprentissage de la programmation sont également moins documentés. Néanmoins, quelques études, dont celles faisant partie de la méta-analyse de Scherer *et al.* (2019), tendent à indiquer des effets positifs, mais de petite taille, de l'apprentissage de la programmation sur les apprentissages en sciences. L'hypothèse initialement formulée était donc que :

H2 – Des *effets de transfert positifs, mais de moindre ampleur*, seront observés entre l'apprentissage de la programmation et la performance scolaire en sciences. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à des résultats scolaires en sciences *légèrement supérieurs* à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

⇒ Hypothèse **infirmée**

Or, les résultats obtenus pour la discipline des sciences ne vont pas dans le sens de cette deuxième hypothèse. En effet, la modélisation incluant les concentrations ne permet pas de mieux représenter les données de performance scolaire en sciences, qu'il s'agisse de la performance globale ou de la performance à chacune des compétences évaluées. Cela signifie que l'appartenance à une concentration n'influence pas les données de performance scolaire en sciences. En d'autres mots, les concentrations ne semblent pas avoir d'effet sur les résultats scolaires pour cette discipline. Contrairement à ce qui avait été anticipé, l'appartenance à la concentration Programmation ne mène donc pas à des résultats scolaires en sciences légèrement supérieurs, comparativement à ceux des élèves des deux autres concentrations, ce qui suggère qu'il n'y a pas d'effet de transfert positif associé à l'apprentissage de la programmation pour la discipline des sciences. Ainsi, ces résultats se distinguent de ceux obtenus dans le cadre d'études antérieures, en particulier celles de Nugent (2010) et de Park (2015), lesquelles avaient toutes deux constaté que des interventions intégrant la programmation avaient des effets positifs sur différents apprentissages en sciences, tant au niveau primaire que secondaire.

Pour ce qui est du français, la littérature scientifique met en évidence quelques liens disciplinaires entre la programmation et les langues (voir p. ex., Barr et Stephenson, 2011), mais ceux-ci sont beaucoup moins nombreux que pour les sciences et les mathématiques. Les résultats des études portant sur les effets de transfert de l'apprentissage de la programmation vers les langues présentent des résultats hétérogènes et moins concluants; la méta-analyse de Scherer et ses collègues ayant pour leur part mis en lumière un effet de transfert nul. Pour ces raisons, l'hypothèse suivante avait été formulée :

H3 – Il n'y aura *pas d'effets de transfert positifs* entre l'apprentissage de la programmation et la performance scolaire en français. De manière opérationnelle, cette hypothèse prévoit qu'à travers cinq années d'observation, la participation à un programme d'apprentissage de la programmation sera associée à des résultats scolaires en français qui seront *similaires* à ceux d'élèves ayant participé à un programme d'apprentissage portant sur un autre domaine que la programmation (groupe de contrôle).

⇒ Hypothèse **confirmée**

Les résultats obtenus confirment cette troisième hypothèse. En effet, de la même façon que pour les résultats obtenus en sciences, la modélisation incluant les concentrations ne permet pas de mieux représenter les données de performance scolaire en français, qu'il s'agisse de la performance globale ou de la performance à chacune des compétences évaluées. Cela signifie que l'appartenance à une concentration n'influence pas les données de performance scolaire en français. En d'autres mots, les concentrations ne semblent pas avoir d'effet sur les résultats scolaires pour cette discipline. Conformément à l'hypothèse formulée, il semble donc que l'appartenance à la concentration Programmation mène à des résultats scolaires en français qui sont similaires à ceux des élèves des autres concentrations, ce qui suggère qu'il n'y a pas d'effet de transfert positif associé à l'apprentissage de la programmation pour la discipline du français. Ces résultats concordent ainsi avec ceux d'autres études qui n'avaient pas observé d'effet de transfert de la programmation vers les langues, telle que celle de Owston (2009) menée auprès d'élèves de niveau primaire ayant pris part à une intervention en programmation. Dans cette étude, aucune différence significative n'a été observée quant à leurs compétences en langage (vocabulaire et compréhension de texte) au terme de l'intervention. De même, les résultats sont en cohérence avec la méta-analyse de Scherer *et al.* (2019), qui suggère l'absence d'un effet de transfert significatif de l'apprentissage de la programmation vers les langues (g = -0,02).

La section qui suit propose d'abord d'aborder et de discuter ces résultats en les analysant à la lumière des concepts, théories et modèles abordés dans le cadre théorique, ainsi qu'au regard de la littérature scientifique portant sur le transfert des apprentissages

5.2 Piste d'interprétation centrale des résultats concernant la nature du transfert entre la programmation et les disciplines scolaires étudiées

Parmi les différentes dimensions à considérer pour discuter des résultats obtenus, l'une apparaît comme étant particulièrement centrale : celle liée à la nature du transfert entre l'apprentissage de la

programmation et chacune des disciplines d'intérêt. Tel que cela a été abordé dans le chapitre de cette thèse présentant le cadre théorique, il existe plusieurs typologies du transfert des apprentissages. Dans le cadre de ce projet, le transfert entre l'apprentissage de la programmation et l'ensemble des disciplines scolaires étudiées est *général* (Cormier et Hagman, 1987), car il suppose que les connaissances acquises en programmation soient applicables dans des contextes différents propres à d'autres disciplines, via le développement de la pensée informatique. Il s'agit également d'un transfert *latéral* (Gagné, 1965), opérant entre des domaines différents (par exemple, programmation et mathématiques), pour lesquels certaines compétences pourraient être partagées. Par ailleurs, il semble que selon la discipline, le transfert à réaliser puisse être plus ou moins proche ou éloigné. À cet égard, Scherer et ses collègues avancent l'idée selon laquelle, en raison du plus grand nombre de compétences qui seraient partagées par la programmation et les mathématiques, le transfert entre ces disciplines serait plus rapproché (Scherer *et al.*, 2019), alors que le transfert de la programmation vers les sciences ou les langues serait plus éloigné. Du fait que la nature du transfert entre la programmation et les disciplines d'intérêt peut varier en fonction du degré de proximité ou d'éloignement entre celles-ci, cet élément revêt une pertinence particulière pour l'interprétation des résultats obtenus. La section qui suit discute donc plus en détail de cette idée et des pistes d'interprétation qui lui sont associées.

5.2.1 Continuum proche/éloigné du transfert comme fenêtre d'analyse privilégiée

D'abord, nos résultats convergent globalement avec l'un des constats centraux découlant de la méta-analyse réalisée par Scherer et ses collègues (2019) : catégorisant le transfert entre l'apprentissage de la programmation et les disciplines scolaires comme étant globalement éloigné, ceux-ci ont toutefois mis de l'avant que parmi les disciplines scolaires, celle des mathématiques apparaissait comme étant la moins éloignée et était associée à la taille d'effet la plus importante (g = 0,57).

D'un point de vue théorique, le modèle du transfert des apprentissages proposé par Barnett et Ceci (2002) fournit également une perspective intéressante pour évaluer le niveau de proximité/éloignement entre la programmation et les autres disciplines scolaires. Ces auteurs identifient en effet plusieurs facteurs susceptibles d'influencer la nature du transfert entre deux situations d'apprentissage sur un continuum proche/éloigné (voir Tableau 2.7). Ces facteurs concernent entre autres le contexte physique (lieu du transfert), le contexte temporel (moment du transfert) et le contexte social (p. ex., en groupe ou de manière individuelle). La plupart de ces facteurs apparaissent équivalents pour l'ensemble des disciplines. Par ailleurs, l'un de ces facteurs concerne spécifiquement les domaines de connaissance associés aux deux

situations d'apprentissage, c'est-à-dire les champs spécifiques dans lesquels les apprentissages sont réalisés, et qui peuvent être plus ou moins proches ou éloignés. À titre d'exemple, Barnett et Ceci (2002) considèrent ainsi que le transfert des apprentissages entre les sciences et les arts serait éloigné étant donné qu'il s'agit de deux domaines de connaissance bien distincts. Cela suggère que, selon ce modèle, le transfert des apprentissages entre les situations d'intérêt de ce projet, à savoir la programmation et chacune des disciplines scolaires, serait globalement caractérisé comme étant éloigné. Cependant, il est raisonnable de postuler que toutes les disciplines ne présentent pas nécessairement le même degré d'éloignement sur ce continuum.

D'ailleurs, tel que cela a été abordé dans le cadre théorique, ce continuum du transfert des apprentissages, allant de proche à éloigné, est étroitement lié au fait que la présence d'éléments communs, c'est-à-dire de similiarités entre deux situations d'apprentissage, peut contribuer à rapprocher ces situations et à favoriser le transfert entre elles (Bracke, 1998; Day et Goldstone, 2012; Lobato, 2006; Thorndike et Woodworth, 1901). Ces similarités sont souvent qualifiées comme étant « de surface » ou encore « structurelle » (Bracke, 1998). La similarité de surface suggère que plus les éléments visibles ou tangibles des situations d'apprentissage sont semblables, plus le transfert est susceptible de se produire. En d'autres termes, des caractéristiques apparentes communes entre deux situations, telles que le format de présentation, l'organisation visuelle ou le langage utilisé, renforcent la probabilité d'un transfert des compétences acquises d'une situation à l'autre (Franks *et al.*, 2000; Lightbown, 2008). La similarité structurelle concerne pour sa part les aspects plus profonds et conceptuels des tâches. Elle renvoie davantage aux structures sous-jacentes des situations d'apprentissage, notamment aux compétences qu'elles mobilisent et ont en commun (Bracke, 1998; 2004; Day et Goldstone, 2012). Tel que cela a été vu, tant la similarité de surface que la similarité structurelle peuvent influencer le transfert des apprentissages (Bracke, 1998; 2004; Cyr, 2012; Gentner *et al.*, 1993; 2003).

Afin d'approfondir notre compréhension du degré d'éloignement entre la programmation et chacune des disciplines d'intérêt, en vue de mieux interpréter les résultats obtenus, il est donc pertinent d'analyser de manière plus approfondie les similarités entre la programmation et les disciplines scolaires. Ces éléments pourraient effectivement contribuer à expliquer les variations observées dans le transfert des apprentissages entre ces domaines spécifiques, soit la présence d'un transfert positif pour les mathématiques, et une absence de transfert pour les sciences et le français.

5.2.1.1 Présence de transfert positif pour la discipline des mathématiques

Bien que le transfert entre la programmation et les mathématiques puisse être considéré comme éloigné selon le modèle de Barnett et Ceci (2002), il est généralement admis que les mathématiques sont la discipline scolaire la plus proche de la programmation (Popat et Starkey, 2019; Scherer *et al*., 2019). À cet égard, plusieurs modèles et concepts présentés dans le cadre théorique contribuent à soutenir cette idée, particulièrement en ce qui concerne la similarité structurelle.

D'abord, le modèle de l'apprentissage de la programmation proposé par Robins *et al.* (2003) dont il a été question à la section 2.1.2.2 permet d'identifier des points de convergence entre la programmation et les mathématiques, tant sur le plan des connaissances que des compétences. L'un des principaux points de convergence concerne sans contredit la compétence de résolution se problèmes, qui apparaît centrale à l'apprentissage de la programmation, et qui représente précisément l'une des compétences évaluées pour la discipline des mathématiques (la compétence 1). Or, c'est justement pour cette compétence que nos résultats indiquent que l'appartenance à la concentration Programmation était associée à une meilleure performance. Parmi les autres compétences identifiées dans le modèle de Robins, le débogage, c'est-à-dire la capacité à réviser sa démarche afin de trouver des erreurs, apparaît également susceptible d'être mobilisé en mathématiques. Il existe donc plusieurs recoupements sur le plan des compétences entre les mathématiques et la programmation, tels qu'identifiés dans le modèle de Robins *et al.* (2003) qui suggèrent qu'il existerait une similarité structurelle assez importante entre ces domaines.

Toujours sur le plan de la similarité structurelle, il est aussi envisageable que les mathématiques entretiennent une proximité accrue avec la programmation, en raison du développement de la pensée informatique qui serait favorisé par l'apprentissage de la programmation. En effet, de l'avis de plusieurs auteurs (Barr et Stephenson, 2011; Weintrop, 2016), les composantes de cette pensée informatique trouvent des liens substantiels avec les mathématiques. Par exemple, la composante relative à la capacité d'abstraction serait nécessaire lors de l'utilisation de variables en algèbre, ou encore la composante relative à la décomposition des problèmes interviendrait pour appliquer l'ordre des opérations dans une expression mathématique. D'ailleurs, l'un des éléments fondamentaux partagés par les nombreuses définitions de la pensée informatique réside dans l'importance accordée à la résolution de problèmes. Cette prépondérance est telle que Lodi (2020) suggère même qu'il serait plus précis de parler d'un « processus de résolution de problèmes » informatique plutôt que de simplement évoquer une « pensée » informatique, étant donné que cette dimension occupe une place de premier plan. Ce dernier a également

identifié plusieurs composantes de la pensée informatique qui apparaissent étroitement liées aux mathématiques, notamment la pensée algorithmique, l'abstraction, la décomposition de problèmes (une présentation exhaustive de ces éléments a été réalisée au Tableau 2.3). Ces éléments pourraient contribuer à expliquer pourquoi la programmation, en favorisant le développement de la pensée informatique, contribuerait aux apprentissages réalisés en mathématiques, et plus particulièrement en résolution de problèmes mathématiques, ce qui est observé dans nos résultats.

Il est également intéressant de considérer la relation entre la programmation et les mathématiques à la lumière du modèle de Popat et Starkey (2019) qui, au terme de leur revue de la littérature, ont identifié trois grandes catégories d'effets de transfert documentés pouvant découler de l'apprentissage de la programmation, soit des effets sur : 1- les compétences en raisonnement de haut niveau, 2- les compétences personnelles, et 3- les compétences en raisonnement de bas niveau (voir Figure 2.2). Parmi les trois catégories de ce modèle, celle des compétences en raisonnement de haut niveau est la plus documentée dans la littérature scientifique. Elle inclut à la fois les compétences associées à la pensée critique ainsi qu'à la résolution de problèmes, précisément dans le contexte de problèmes en mathématiques. Ce modèle mettait donc déjà de l'avant la place privilégiée des mathématiques dans le processus de transfert depuis la programmation. Les résultats de la présente étude appuient donc cette idée d'un transfert de l'apprentissage de la programmation, vers des compétences de haut niveau en résolution de problèmes mathématiques.

À cet égard, les résultats qui ont été obtenus en mathématiques sont également convergents avec certaines des études incluses dans la revue de littérature de Popat et Starkey (2019) qui ont établi un lien direct entre l'apprentissage de la programmation et l'amélioration des compétences en résolution de problèmes mathématiques. Par exemple, Fessakis (2013) avait montré que le fait de prendre part à des activités en programmation peut contribuer positivement aux capacités de résolution de problèmes mathématiques chez de jeunes élèves. Similairement, les études de Bernardo et Morris (1994) et Palumbo et Reed (1991) ont observé que l'apprentissage de la programmation favorisait le développement des compétences en résolution de problèmes en mathématiques chez des élèves du secondaire. Ces auteurs avançaient l'idée que ce transfert découlait probablement de la similitude des processus cognitifs engagés à la fois dans la programmation et dans la résolution de problèmes mathématiques, englobant des aspects tels que l'analyse des problèmes, la planification, et l'exécution logique des solutions, ce qui rejoint donc l'idée d'une similarité structurelle importante entre la programmation et les mathématiques.

Par ailleurs, bien que cela ne soit pas identifié de manière explicite dans la littérature, nous avançons l'idée qu'il est possible que les mathématiques et la programmation partagent également, en plus d'une forte similarité structurelle, un certain niveau de similarité de surface, et ce, possiblement de manière plus importante que pour d'autres disciplines. En effet, à la lumière des principaux concepts identifiés par Tew et Gudzial relativement à l'apprentissage de la programmation (ceux-ci ont été présentés au Tableau 2.1), il appert que certains de ceux-ci sont partagés par les mathématiques, par exemple les variables, les opérations sur les variables ainsi que le type de données. En programmation, les variables sont utilisées pour stocker des données qui peuvent être modifiées au cours de l'exécution d'un programme. Elles sont fondamentales pour le stockage et la manipulation des informations. En mathématiques, plus précisément en algèbre, les élèves apprennent à utiliser des lettres pour représenter des nombres inconnus ou des quantités qui peuvent changer. La programmation utilise également des opérations arithmétiques pour effectuer des calculs sur les variables. Ces opérations de base incluent l'addition, la soustraction, la multiplication et la division. En mathématiques, ces opérations arithmétiques sont omniprésentes, y compris l'ordre des opérations et les propriétés des opérations comme la commutativité, l'associativité et la distributivité. En programmation, les types de données définissent la nature des valeurs que peuvent prendre les variables, comme des nombres entiers, des nombres à virgule flottante, ou des chaînes de caractères. En mathématiques, les élèves étudient différents types de nombres (entiers, décimaux, rationnels) et apprennent comment les manipuler. Cette compétence est analogue à la compréhension de l'importance des types de données en programmation pour effectuer des calculs. Or, cette similarité relative au contenu va au-delà de la similarité structurelle. En effet, nous postulons qu'une similarité de surface entre la programmation et les mathématiques peut également être observée dans la façon dont ces concepts sont exprimés et utilisés concrètement. Par exemple, l'utilisation partagée de variables peut contribuer à rendre les tâches plus similaires sur le plan de la présentation visuelle. Il en va de même pour l'utilisation des opérations arithmétiques et du type de données. Qui plus est, d'autres exemples pourraient concerner la syntaxe des instructions de programmation qui peut rappeler la notation mathématique; la représentation graphique des données, telle que l'utilisation de graphiques et de diagrammes présente pour les deux domaines ou encore les commentaires descriptifs, fréquemment utilisés en programmation pour expliquer le fonctionnement du code, qui peuvent être comparés aux étapes détaillées dans la résolution de problèmes mathématiques. La manière dont les boucles et les conditions sont exprimées, qu'il s'agisse de boucles *for* et *while* en programmation ou d'itérations mathématiques, représente un autre exemple de similarité de surface.

À la lumière de ces éléments, il semble donc que la programmation et les mathématiques présentent à la fois des similarités sur le plan de la structure et de la surface, ce qui pourrait contribuer à faire en sorte que ces deux domaines soient plus « rapprochés ». Les similarités identifiées sur le plan de la structure concernent principalement des compétences qui sont partagées par les deux domaines et qui semblent étroitement liées aux composantes de la pensée informatique. En particulier, la compétence en résolution de problèmes semble représenter un point de similarité structurelle pour le moins central. Il est d'ailleurs possible que les éléments de similarité de surface qui ont été identifiés et discutés précédemment rejoignent plus directement cette compétence de résolution de problèmes, créant une forme d'effet additif quant à la similarité existante, ce qui contribuerait aussi à expliquer pourquoi l'effet de transfert a été observé pour la compétence 1 « Résoudre une situation-problème » et non pour la compétence 2 « Déployer un raisonnement mathématique », dans le cadre de ce projet. Cette distinction revêt un intérêt particulier, car la façon dont ces deux compétences sont évaluées dans la pratique diffère à plusieurs égards (Gouvernement du Québec, 2011). La compétence 1 est en effet souvent évaluée par le biais de mises en situations contextualisées et plus complexes, impliquant davantage de texte, qui nécessitent de faire appel à plusieurs notions. Cette compétence vise à déterminer si l'élève sait *quand* mobiliser différentes techniques. La compétence 2 est pour sa part évaluée par le biais de problèmes plus simples qui ne font pas appel à une mise en situation complexe et qui réfèrent à un nombre limité de notions ciblées; il s'agit davantage de mises en application. Elle permet de déterminer si l'élève sait *comment* appliquer une technique en particulier. Il est donc intéressant de souligner que les résultats suggèrent un effet de transfert uniquement pour la compétence 1. Cela suggère que les compétences partagées par la programmation et les mathématiques, incluant la résolution de problèmes, sont possiblement davantage mobilisées lors des situations d'évaluation associées à la compétence 1 et que l'apprentissage de la programmation est susceptible de contribuer davantage à ces situations d'évaluation plus complexes, probablement en raison des compétences générales et de haut niveau qu'il permet de développer.

5.2.1.2 Absence de transfert positif pour la discipline des sciences

Afin de discuter de l'absence de transfert positif constatée en sciences, les paragraphes qui suivent examinent également les principales similarités entre la programmation et les sciences, de manière à mieux appréhender le degré d'éloignement entre ces domaines.

D'abord, sur le plan de la similarité structurelle, il apparaît probable que les sciences entretiennent une certaine proximité avec la programmation. Par exemple, tout comme les mathématiques, la résolution de

problème est une compétence essentielle en sciences. À cet égard, les attentes de fin de cycle de la compétence 1 mentionnent que : « À la fin du deuxième cycle du secondaire, l'élève est en mesure de mettre en œuvre un processus de résolution de problèmes. Il s'approprie le problème en dégageant le but à atteindre ou le besoin à cerner ainsi que les conditions à respecter. Il formule ou reformule des questions qui s'appuient sur des données issues du problème. Il propose des hypothèses vraisemblables ou des solutions possibles, qu'il est en mesure de justifier » (Gouvernement du Québec, 2007, p.14). Ces éléments rejoignent les compétences de planification et de résolution de problèmes, telles que décrites dans le modèle de Robins *et al.* (2003).

De plus, cet extrait des attentes de fin de cycle de la compétence 1 en sciences met aussi en évidence l'importance du processus de formulation et la vérification d'hypothèses. Une hypothèse est une proposition testable, soumise à des expériences pour validation ou invalidation en fonction des résultats obtenus. Ce processus implique souvent d'analyser si plusieurs conditions sont satisfaites pour considérer une hypothèse comme vérifiée. Or, en programmation, les opérateurs logiques tels que *and*, *or* et *not* sont utilisés pour construire des expressions complexes évaluant la véracité de multiples conditions. Par exemple, l'opérateur logique *and* permet d'exécuter un bloc de code uniquement si plusieurs conditions sont simultanément satisfaites. De même, *or* permet l'exécution d'un bloc de code si au moins l'une des conditions est remplie. Une similarité apparaît donc entre les deux domaines quant au raisonnement nécessaire pour parvenir à des conclusions. En sciences, un chercheur peut ainsi formuler une hypothèse exigeant que plusieurs observations soient valides pour être confirmée. Par exemple, pour qu'une hypothèse soit jugée plausible, il peut être nécessaire que les observations A ET B soient toutes deux valides (de manière analogue à l'utilisation de l'opérateur *and* en programmation). Alternativement, il suffit que l'observation A OU B soit valide pour considérer une autre hypothèse comme probable (de manière analogue à l'utilisation de l'opérateur *or* en programmation).

Il est également pertinent d'établir des liens entre la programmation et les sciences à travers le concept de la pensée informatique, comme souligné par les travaux de Barr et Stephenson (2011) et de Lodi (2020). Par exemple, Barr et Stephenson (2011) ont identifié plusieurs composantes de la pensée informatique susceptibles d'être mobilisées à la fois en programmation et en sciences, notamment la collecte, l'analyse et la représentation des données. En sciences, dans le contexte d'une expérimentation en laboratoire par exemple, la collecte, l'analyse et la représentation des données jouent un rôle central afin de vérifier une hypothèse. En programmation, l'utilisation et la manipulation des données passent par l'interaction avec

le programme. Concrètement, un programmeur pourrait par exemple rédiger un code afin qu'un utilisateur fournisse certaines informations (collecte des données), que le programme traite ces informations (analyse) et affiche ensuite le résultat de ce traitement à l'utilisateur (représentation). De manière convergente, Lodi (2020) intègre également la collecte, l'analyse et la représentation des données dans son modèle de la pensée informatique. Il apparaît donc que l'utilisation des données, qui constitue une composante de la pensée informatique, représente un autre point de similarité structurelle entre les domaines de la programmation et des sciences.

Par ailleurs, tout comme en mathématiques, nous avançons qu'il est possible que les sciences et la programmation partagent également un certain niveau de similarité de surface. Toujours à la lumière des principaux concepts identifiés par Tew et Gudzial (2010) relativement à l'apprentissage de la programmation (présentés au Tableau 2.1), il apparaît que certains présentent des similarités, tels que les variables et les opérations sur les variables. En sciences, les variables sont fréquemment utilisées pour représenter des quantités physiques qui peuvent varier dans le contexte d'une expérience ou d'un problème spécifique. De manière concrète, la Progression des apprentissages en sciences au secondaire du PFEQ (Gouvernement du Québec, 2011) identifie d'ailleurs de nombreuses formules (p. ex., $E_p = mgh$; $U = RI$; $P = UI$; $E = P\Delta t$; $F = kq_1q_2/r^2$; $PV = nRT$; $F = ma$) qui doivent être maîtrisées par les élèves afin de comprendre et d'appliquer les principes fondamentaux régissant les phénomènes physiques et chimiques. Ces formules impliquent toutes l'utilisation de variables (telles que m pour la masse, g pour l'accélération gravitationnelle, R pour la résistance, I pour l'intensité du courant, etc.) pour modéliser et résoudre des problèmes scientifiques. De plus, ces variables peuvent prendre différentes valeurs selon les conditions expérimentales, illustrant leur nature dynamique et modifiable. Fréquemment utilisées en sciences pour représenter des quantités physiques, elles trouvent leur équivalent dans la programmation, où elles servent à stocker et manipuler des données tout au long de l'exécution d'un programme. Un parallèle existe donc entre les sciences et la programmation quant à l'utilisation de variables qui permettent une représentation abstraite de données concrètes, facilitant ainsi le traitement, l'analyse et la résolution de problèmes complexes. De même, les opérateurs arithmétiques jouent un rôle clé dans la manipulation de ces variables tant en sciences qu'en programmation pour calculer de nouvelles valeurs à partir de celles qui sont déjà connues. Ainsi, la similarité de surface entre la programmation et les sciences pourrait ainsi résider dans le fait que ces deux domaines partagent un langage commun constitué de variables et d'opérations arithmétiques.

Toutefois, malgré ces similarités de structure en ce qui concerne la résolution de problèmes, l'utilisation du raisonnement logique et la collecte et la manipulation des données, en plus des similarités de surface probables qui viennent d'être abordées, il est important de rappeler que les résultats scolaires en sciences observés dans notre étude ne laissent pas supposer un transfert positif à la suite de l'apprentissage de la programmation. Cela peut sembler surprenant étant donné les similarités qui ont été identifiées.

Une première hypothèse explicative pourrait être que bien que les sciences semblent partager des similarités avec la programmation, celles-ci sont tout de même moins présentes et moins centrales qu'en mathématiques. Par exemple, bien que les deux disciplines nécessitent des compétences en résolution de problèmes, les concepts mathématiques sont souvent plus abstraits et formels, ce qui pourrait les rendre plus facilement transposables en matière de logique de programmation. En revanche, les concepts scientifiques peuvent être plus concrets et spécifiques à des domaines particuliers, ce qui pourrait rendre la transposition vers la programmation moins directe. Qui plus est, les similarités de surface identifiées entre les sciences et la programmation entrent en jeu principalement en fin de parcours au secondaire (4^{e} et 5^{e} années), lorsqu'une compréhension quantitative des phénomènes scientifiques est introduite. Cela signifie que les analyses effectuées n'ont peut-être pas été en mesure de capturer pleinement l'impact potentiel de l'apprentissage de la programmation sur les compétences en sciences, étant donné que ces similarités deviennent plus manifestes uniquement dans les dernières années du parcours scolaire. En effet, le devis longitudinal de l'étude, couvrant plusieurs années, ne permettait pas d'isoler spécifiquement les effets sur les temps de mesure où les similarités de surface étaient plus susceptibles d'influencer positivement les apprentissages en sciences, ce qui a pu limiter la capacité à détecter un transfert positif pour l'ensemble des données recueillies.

De plus, il paraît intéressant de noter que les similarités de surface identifiées entre les sciences et la programmation sont particulièrement liées aux concepts mathématiques, et plus précisément à l'application de formules mathématiques en contexte de résolution de problèmes scientifiques. Or, l'intégration des mathématiques dans les sciences devient également plus explicite vers la fin du parcours scolaire au secondaire. C'est en effet à ce stade que les élèves doivent mettre en relation la compréhension qualitative d'un concept, comme le principe de conservation de la matière, et l'application quantitative des formules mathématiques correspondantes, telles que celles utilisées en stœchiométrie pour équilibrer des équations. Contrairement aux mathématiques, où la relation avec la programmation apparaît plus directe et intégrée tout au long du parcours, en sciences, il semble probable que cette connexion transite

davantage par les mathématiques, et qu'elle prenne place davantage en fin de parcours. Des similarités moins marquées et moins directes entre les sciences et la programmation, notamment en termes de similarités de surface, pourraient donc contribuer à éloigner davantage ces domaines, ce qui pourrait expliquer en partie l'absence de transfert positif observé dans les résultats.

Une seconde hypothèse explicative concerne le fait que la discipline des sciences au secondaire apparaît plus hétérogène que celle des mathématiques, englobant une gamme de disciplines scientifiques telles que la biologie, la physique, la chimie, l'écologie, la géologie et l'astronomie au sein d'une même évaluation. Or, cette hétérogénéité disciplinaire pourrait compliquer la détection des effets de transfert de l'apprentissage de la programmation, car les compétences et les connaissances développées par la programmation pourraient être plus directement liées à certaines disciplines scientifiques précises, comme la physique, où la manipulation de variables et l'application d'opérations arithmétiques sont plus centrales. En revanche, ces liens semblent moins évidents pour des disciplines comme la géologie, pour laquelle l'étude des processus terrestres, tels que la formation des roches, l'érosion, ou les mouvements tectoniques, ne présente pas de similarités aussi importantes avec la programmation. Ainsi, malgré certaines similarités structurelles et de surface entre la programmation et les sciences, il est possible que cette hétérogénéité disciplinaire puisse contribuer à expliquer les résultats obtenus. En effet, considérant que les résultats scolaires utilisés dans les analyses portent sur deux compétences qui couvrent l'ensemble des disciplines scientifiques, il est plausible que la portion des résultats associés aux disciplines qui sont moins étroitement liées à la programmation ait atténué l'effet qui aurait pu être observé. Par exemple, si la programmation a un impact sur les apprentissages en physique, mais que ces contenus disciplinaires représentent une part relativement faible des évaluations par rapport à d'autres disciplines scientifiques moins connectées à la programmation, cela pourrait avoir réduit la manifestation d'un transfert positif dans les résultats de chaque compétence ainsi que dans les résultats globaux. En d'autres termes, la diversité des disciplines scientifiques évaluées pourrait avoir dilué l'effet potentiellement positif de la programmation sur les performances en sciences.

Enfin, il est essentiel de rappeler qu'à la différence des mathématiques, il existe très peu d'études documentant les effets de transfert de l'apprentissage de la programmation vers les sciences; d'ailleurs, seulement deux faisaient partie de la méta-analyse de Scherer *et al.* (2019). À la lumière des résultats obtenus dans le cadre de ce projet, il est aussi possible que cette rareté d'études puisse partiellement s'expliquer par un biais de publication. Les recherches ne montrant pas d'effets significatifs ou présentant

des résultats moins concluants ont en effet tendance à être moins publiées, créant ainsi un biais de publication qui favorise la dissémination de recherches avec des résultats positifs et significatifs. Il est donc possible que ce biais ait influencé notre hypothèse de recherche (H2), expliquant ainsi pourquoi les résultats obtenus vont à l'encontre des attentes initiales. L'absence de transfert significatif en sciences observée dans cette étude pourrait ainsi non seulement représenter une tendance réelle dans les données, mais également suggérer une sous-représentation des résultats non significatifs dans les recherches publiées sur le sujet.

5.2.1.3 Absence de transfert positif pour la discipline du français

Contrairement aux résultats en sciences, l'absence de transfert positif de la programmation vers le français observé dans cette étude était attendue, étant donné les résultats des quelques études existantes s'étant intéressées à la question (notamment Hayes et Stewart, 2016). D'emblée, il semble que la programmation et le français soient généralement considérés comme étant plus éloignés (Scherer *et al.*, 2019). Intuitivement, les compétences mobilisées en français telles que la lecture, l'écriture et la communication orale semblent en effet plus distantes des compétences techniques et logiques qui sont développées en programmation, à l'exception peut-être de la rédaction du code qui, de manière assez apparente, rejoint le français via l'écriture d'un langage. Qu'il s'agisse d'un langage de programmation informatique ou de la langue française, les deux constituent des systèmes de représentation symboliques, avec une grammaire et une syntaxe, qui sont utilisés pour transmettre du sens, pour produire quelque chose et pour communiquer (Bers, 2018; 2019).

À cet égard, sur le plan de la similarité structurelle, l'élément central partagé par la programmation et le français réside dans l'utilisation d'une logique syntaxique (Bers, 2019; 2023). Le langage de programmation possède en effet une syntaxe dans laquelle les symboles représentent des actions et le code source doit suivre une structure grammaticale précise pour être correctement interprété par l'ordinateur. De même, en français, les phrases et les paragraphes doivent respecter une syntaxe spécifique pour être compréhensibles. Par ailleurs, malgré cette similarité sur le plan syntaxique, il apparaît qu'en matière de structure, la programmation et les mathématiques sont possiblement plus rapprochées en raison de leur utilisation d'une logique formelle plus stricte; il semble probable qu'en raison de l'usage plus libre de la langue, le français offre une palette plus étendue de moyens d'expression, et conséquemment que la similarité sur le plan de la structure est peut-être moins importante.

D'ailleurs, toujours sur le plan de la similarité structurelle, il semble plus difficile d'établir des liens entre les compétences identifiées dans le modèle de Robins *et al.* (2003) et les compétences linguistiques mobilisées en français. Par exemple, bien que la compétence de débogage en programmation exige une analyse critique et une révision minutieuse, ce qui rejoint le français, ces processus sont intrinsèquement différents de ceux impliqués dans la compréhension et l'interprétation de textes littéraires ou la production écrite en français. De même, il semble également plus difficile de trouver des liens entre les composantes de la pensée informatique, favorisée par l'apprentissage de la programmation, et la discipline du français. Bien que des auteurs comme Barr et Stephenson (2011) aient identifié quelques liens entre la programmation et les langues, au regard des composantes de la pensée informatique, ces liens apparaissent moins nombreux que pour les mathématiques et les sciences. Parmi ces liens, on trouve notamment la rédaction d'instructions qui pourrait mobiliser la composante de la pensée algorithmique et l'utilisation de figures de style qui pourrait s'appuyer sur la composante de l'abstraction.

Qui plus est, le modèle de Popat et Starkey (2019) suggère lui aussi que le transfert des compétences en programmation vers des disciplines scolaires comme le français pourrait être limité. En effet, selon ce modèle, les compétences en raisonnement de bas niveau, qui incluent les compétences mobilisées pour des disciplines spécifiques, sont moins susceptibles de bénéficier directement de l'apprentissage de la programmation. Dans le cas du français, cela pourrait être dû au fait que les compétences linguistiques nécessitent un ensemble distinct de capacités cognitives, comme la compréhension nuancée des textes, l'expression créative et la maîtrise de la grammaire, qui ne sont pas directement cultivées par la programmation.

Cependant, sur le plan de la similarité de surface, il nous semble possible d'établir de manière intuitive un certain nombre de liens entre la programmation et le français. D'abord, sur le plan de la notation et de la ponctuation, les deux domaines utilisent des notations spécifiques et la ponctuation pour clarifier la signification. En programmation, les caractères spéciaux, les virgules, parenthèses et points-virgules sont utilisés pour structurer le code. De même, en français, la ponctuation, telle que les virgules, points et points-virgules, est cruciale pour délimiter les phrases et clarifier la syntaxe. Ensuite, l'utilisation de variables et de noms suggère également une certaine forme de similarité : en programmation, les variables sont utilisées pour stocker des valeurs, et le choix de noms significatifs pour ces variables améliore la lisibilité du code. De manière similaire, en français, le choix de termes spécifiques contribue à la clarté et à la compréhension d'un texte. Une autre similarité relève de la mise en page : la présentation visuelle du

code, notamment à travers l'indentation, est cruciale en programmation pour rendre le code lisible. En français, la mise en page, les paragraphes et l'indentation facilitent également la lecture et la compréhension d'un texte. Finalement, l'utilisation de structures conditionnelles peut être observée à la fois en programmation et en français. En programmation, les structures conditionnelles, telles que les boucles et les instructions de type *elseif* (si alors), peuvent être comparées aux expressions conditionnelles et aux phrases subordonnées en français, car elles introduisent des conditions pour guider le flux d'exécution ou la compréhension de lecture.

Par ailleurs, une incursion dans la recherche en neuroscience cognitive permet d'apporter une perspective supplémentaire à cette question de la similarité entre la programmation et le français. Une étude de Ivanova *et al.* (2020) s'est en effet intéressé aux corrélats cérébraux de la lecture du code informatique, comparativement à la lecture de phrases écrites. Les résultats de cette étude appuient l'idée que les mathématiques seraient plus rapprochées de la programmation que le français. En effet, il semble que les régions cérébrales mobilisées lors de la lecture du code informatique ne sont pas les mêmes que celles qui entrent en jeu lors de la lecture de phrases écrites. Cela suggère que les processus cognitifs mobilisés pour traiter le code informatique et le français écrit ne sont pas les mêmes, malgré le fait qu'il s'agisse de deux langages qui partagent des similarités syntaxiques. Étant donné que les compétences sont des manifestations externes qui résultent de processus cognitifs internes, ces résultats suggèrent que les compétences en programmation et en lecture du français ne sont pas simplement des variations d'une même capacité cognitive, mais plutôt des aptitudes distinctes faisant appel à des processus cognitifs spécifiques. Les résultats l'étude de Ivanova *et al.* (idem) montrent également que la lecture et la compréhension du code informatique mobilisent des régions cérébrales bilatérales qui sont habituellement impliquées dans la résolution de problèmes en mathématiques, en logique ainsi que pour des tâches de nature exécutives. Ces résultats laissent donc entendre que le français et la programmation seraient plus éloignés d'un point de vue cognitif et que les mathématiques et la programmation seraient plus proches en raison de la mobilisation de régions cérébrales communes, ce qui pourrait donc contribuer à expliquer pourquoi le transfert est possiblement plus difficile à réaliser entre la programmation et le français, mais plus facile entre la programmation et les mathématiques, ce vers quoi convergent les résultats de ce projet.

En somme, la discipline du français semble partager quelques similarités, tant sur le plan de la structure que de la surface, avec la programmation. Néanmoins, ces similarités apparaissent dans l'ensemble moins

fortes que pour la discipline des mathématiques, et même celle des sciences. De ce fait, l'absence de transfert positif en français apparaît logique, compte tenu du fait que cette absence de transfert a également été observée pour les sciences, qui paraissent pourtant un peu plus proches de la programmation. L'explication qui nous apparaît la plus probable est donc qu'il n'est pas possible de constater un transfert positif entre la programmation et le français en raison de l'éloignement entre ces domaines qui serait possiblement trop important, et diminuerait donc les chances qu'un transfert puisse se produire.

La section qui suit poursuit cette réflexion en abordant de manière plus large la rareté d'observation du transfert éloigné pour différents domaines.

5.2.2 Transfert éloigné : un objectif convoité, mais pourtant peu observé

En éducation, le type de transfert le plus recherché est souvent celui qui s'opère de manière éloignée, l'objectif primordial de tout apprentissage étant qu'il puisse se manifester de manière pertinente dans une diversité de situations (Hattie, 2016 ; Richard et Bissonnette, 2012). Or, la littérature scientifique sur le transfert suggère qu'il y aurait une difficulté réelle à observer du transfert, surtout lorsqu'il est éloigné (Barnett et Ceci, 2002; Day et Goldstone, 2012; Lobato, 2006), étant donné que des situations éloignées partagent généralement moins de caractéristiques communes. De nombreux chercheurs soulignent même que la conclusion la plus fréquente des études portant sur le transfert des apprentissages entre deux situations est l'absence de transfert, signifiant qu'il n'a pas été possible de mobiliser un apprentissage réalisé dans une première situation pour l'appliquer dans une seconde (p. ex., Anolli *et al*., 2001; Sala et Gobet, 2017a). À cet égard, Sala et Gobet (2017a) ont voulu déterminer s'il est réellement possible d'observer du transfert éloigné (« *Does Far Transfer Exist ?* »). Pour ce faire, ils ont conduit deux méta-analyses évaluant respectivement l'effet de l'enseignement des échecs (Sala et Gobet, 2016) et de la musique (Sala et Gobet, 2017b) sur les compétences cognitives et académiques des enfants. Une troisième méta-analyse (Sala et Gobet, 2017c) a aussi évalué les effets de l'entraînement de la mémoire de travail, une compétence cognitive (fonction exécutive) corrélée à l'expertise en musique et aux échecs, sur les mêmes variables. Ces situations avaient été choisies, car la littérature suggère qu'elles font intervenir des compétences cognitives générales qui peuvent être entraînées par la pratique et qui, par hypothèse, pourraient alors se transférer à d'autres domaines. En effet, des études ont démontré que l'entraînement de la mémoire de travail conduit à des améliorations dans les capacités liées à l'intelligence fluide (Jaeggi *et al.*, 2008). De même, l'apprentissage de la musique a été associé à des gains d'intelligence générale,

pouvant potentiellement influencer un large éventail de compétences cognitives et académiques. En ce qui concerne les échecs, ils sont reconnus pour développer la mémoire de travail, l'intelligence fluide et la concentration, suggérant ainsi que la pratique des échecs pourrait entraîner une amélioration générale de ces capacités et favoriser leur transfert à d'autres contextes d'apprentissage (Bart, 2014). Ces éléments ne sont pas sans rappeler le postulat sous-jacent au transfert résultant de l'apprentissage de la programmation, à savoir que cet apprentissage permettrait de développer une pensée informatique englobant des compétences cognitives de haut niveau pouvant être réinvesties dans d'autres disciplines scolaires (cette idée a été illustrée plus tôt à la Figure 2.2).

Or, au terme de leurs méta-analyses, Sala et Gobet (2017a; 2020a; 2020b) se montrent très critiques quant aux possibilités réelles qu'un transfert éloigné se produise, malgré le fait que les contextes étudiés y semblaient particulièrement propices. Bien que les résultats de leurs trois méta-analyses indiquent de prime abord des effets de transfert globalement positifs, associés à des tailles d'effet allant de petites à modérées, les chercheurs ont constaté que l'ampleur des effets observés était inversement liée à la qualité du devis expérimental des études incluses dans l'analyse. Plus précisément, lorsque les analyses prenaient uniquement en compte les études pour lesquelles les groupes expérimentaux étaient comparés à des groupes de contrôle actifs, les tailles d'effet observées étaient minimes, voire même nulles.

Ces résultats convergent avec ceux d'études portant sur d'autres types d'apprentissages. Par exemple, considérant que les habiletés spatiales peuvent être améliorées par la pratique (Uttal *et al.*, 2013) et qu'il est reconnu que ces habiletés sont prédictrices de la réussite en mathématiques (Wai *et al.*, 2009), certaines études ont cherché à entraîner les habiletés spatiales afin de vérifier si celles-ci pouvaient être transférées à différentes tâches en mathématiques (Xu et LeFevre, 2016). Les tentatives visant à établir un tel transfert éloigné se sont cependant, jusqu'à présent, avérées infructueuses.

De façon similaire, Oei et Patterson (2015) ont remis en question l'idée selon laquelle la pratique de jeux vidéo d'action permettrait de développer des habiletés visuoattentionnelles et cognitives pouvant être transférées à un large éventail de tâches. Leurs résultats suggèrent plutôt que le transfert observé se limiterait aux tâches qui sont quasi identiques à celles réalisées dans le jeu vidéo. Dans une revue systématique de la littérature, Simons *et al.* (2016) soulignent également qu'il n'existerait aucune preuve que des programmes d'entraînement cérébral (« *brain training programs* ») pourraient permettre de développer ou d'améliorer des capacités cognitives transférables vers d'autres apprentissages plus

éloignés. En effet, ces programmes avancent souvent que les entraînements proposés sont susceptibles d'avoir des effets sur la performance et la réussite dans le monde réel, pour différentes tâches de la vie quotidienne. Or, les données existantes indiquent que ces programmes d'entraînement mènent à des gains de performance uniquement pour les tâches qu'ils entraînent directement ou au mieux, pour des tâches quasi identiques. Qui plus est, à l'instar de Sala et Gobet (2017), Simons et ses collaborateurs (2016) soulignent également que toutes les études présentant un devis expérimental rigoureux incluant un groupe de contrôle actif ne montrent pas d'effet de transfert éloigné découlant des programmes d'entraînement cérébral.

Ce corpus de recherches met donc en évidence que le transfert éloigné est, de façon générale, un phénomène qui semble peu observé dans la littérature scientifique. Ce constat suggère que les résultats de la présente étude ne font donc pas exception, mais s'inscrivent plutôt dans la lignée de nombreux résultats de recherche qui n'observent pas de transfert des apprentissages entre des situations globalement considérées comme étant éloignées. De plus, le corpus de recherche présenté souligne un autre facteur qui semble fortement influencer la détection d'un effet de transfert, à savoir la présence d'un groupe de contrôle actif au sein du protocole de recherche. Or, c'est précisément le cas de ce projet dont le devis inclut deux groupes de contrôle actifs. La section qui suit discute donc plus en détail de l'incidence possible de ce choix méthodologique sur les résultats obtenus, compte tenu de la nature des groupes de contrôle employés.

5.2.3 Autre piste d'interprétation des résultats concernant la présence de groupes de contrôle actifs

La section précédente a mis en évidence une première interprétation centrale des résultats concernant la nature du transfert entre l'apprentissage de la programmation et les disciplines scolaires étudiées. Cette interprétation suggère principalement que seule la discipline des mathématiques semble être suffisamment proche de la programmation pour permettre l'observation d'un transfert positif. Par ailleurs, la présente étude se distingue sur le plan méthodologique de la plupart des recherches antérieures portant sur les effets de transfert liés à la programmation, puisqu'elle comporte deux groupes de contrôle actifs, à savoir les concentrations Arts et Sports. Or, il est possible que la présence de ces groupes de contrôle actifs puisse aussi contribuer à expliquer, au moins en partie, les résultats observés.

L'utilisation d'un groupe de contrôle actif est préconisée lorsqu'on souhaite isoler les effets spécifiques associés à une condition donnée (Fortin et Gagnon, 2016). Dans le cadre de ce projet, si la concentration

Programmation avait été uniquement comparée à un groupe de contrôle passif (par exemple, un groupe d'élèves ne pratiquant pas la programmation), il aurait été difficile de distinguer les effets spécifiques de la programmation elle-même. Dans cette optique, les effets observés pourraient simplement résulter du fait que les élèves de la concentration Programmation étaient engagés dans une activité supplémentaire. Dans le cas où ces élèves auraient obtenu de meilleurs résultats, par exemple en sciences, il aurait été compliqué de déterminer si ces améliorations étaient attribuables à l'apprentissage de la programmation ou simplement au fait qu'ils avaient acquis des connaissances supplémentaires par rapport au groupe de contrôle (qu'elles soient ou non en programmation). La présence d'un groupe de contrôle actif permet donc de réellement isoler la valeur ajoutée de la variable d'intérêt (Fortin et Gagnon, 2016).

À cet égard, dans le cadre de leur méta-analyse, Scherer et ses collègues (2019) ont examiné plusieurs variables susceptibles d'influencer les effets de transfert associés à l'apprentissage de la programmation. Pour ce faire, ils ont attribué un code à différentes variables d'ordre méthodologique (par exemple, la présence d'un prétest et d'un posttest ou seulement d'un posttest; l'assignation aléatoire ou non des élèves au sein des groupes de contrôle et expérimental; l'utilisation d'un outil d'évaluation standardisé ou d'un outil maison; la publication au sein d'une revue incluant un processus de révision par les pairs ou non, etc.) et les ont intégrées en tant que variables modératrices dans leurs analyses, afin de déterminer si certaines d'entre elles avaient un effet significatif sur les tailles d'effet mesurées. Parmi la douzaine de variables modératrices incluses, seulement deux se sont avérées statistiquement significatives, et celle ayant le plus grand impact sur les tailles d'effet était la présence d'un groupe de contrôle actif, comparativement à un groupe de contrôle passif, au sein du devis méthodologique employé. Plus précisément, au regard du transfert éloigné, les tailles d'effet étaient significativement plus basses dans les études incluant un groupe de contrôle actif (en moyenne, g = 0,16) comparativement à celles incluant un groupe de contrôle passif (en moyenne, g = 0,65). Ces résultats rejoignent ainsi ceux de Sala et Gobet (2017a) discutés à la section précédente.

Ce constat ouvre donc la voie à deux hypothèses explicatives quant à l'absence de transfert positif observé en sciences et en français : d'une part, cette absence de transfert positif pourrait simplement être le reflet d'une réelle absence complète d'effet de transfert associé à l'apprentissage de la programmation pour ces disciplines; d'autre part, il est aussi possible que les effets associés aux groupes de contrôle actifs (Arts et Sports) soient statistiquement équivalents à ceux associés à l'apprentissage de la programmation et donc

que l'effet distinct de la programmation ne ressorte pas pour les disciplines des sciences et du français, lorsque comparé aux groupes de contrôle.

La deuxième hypothèse explicative signifierait que les apprentissages réalisés et les compétences développées par ces autres concentrations seraient également susceptibles d'être réinvestis en sciences et en français. L'idée qu'il puisse y avoir des liens entre les concentrations Sports et Arts et les disciplines des sciences et du français n'apparaît pas impossible. Néanmoins, tel que cela a été discuté dans la section précédente, il semble que de tels effets de transfert éloignés sont rarement observés. Pour ce qui est de la concentration Sports, quelques études mettent tout de même de l'avant que le fait de pratiquer une activité physique de façon régulière et soutenue pourrait être associé à de meilleures fonctions exécutives (de Greeff *et al.*, 2018; Li *et al.*, 2017; Salas-Gomez *et al.*, 2020; Tomporwoski *et al.*, 2008), et même dans certains cas à de meilleures performances scolaires (p.ex., Li *et al.*, 2017; Tomporowski *et al.*, 2008). Les fonctions exécutives représentent un ensemble de processus cognitifs dits « de haut niveau » à la base de la régulation de l'attention, de la pensée et des actions qui permettent de raisonner, de se concentrer et de résister à des interférences afin d'atteindre un objectif (Diamond, 2013, 2020). Il est reconnu que ces fonctions jouent un rôle important pour de nombreux apprentissages que doit réaliser l'élève (Houdé et Borst, 2014; Purpura *et al.*, 2017), incluant des apprentissages en sciences, en français et en mathématiques. Certaines études ont ainsi montré que l'exercice physique pourrait être associé à de meilleures fonctions exécutives, chez des élèves de différents âges. Dans le cadre de la présente étude, il est donc possible que le fait de pratiquer une activité physique régulière, au sein de la concentration Sports, favorise le développement des compétences liées aux fonctions exécutives, et que cela puisse contribuer à influencer positivement les apprentissages scolaires, notamment ceux en sciences et en français. Néanmoins, les résultats ne sont pas tous convergents et certaines études montrent même, à l'inverse, que les effets sur les résultats scolaires seraient peu courants (Kvalø *et al.*, 2017). Bien qu'il puisse s'agir d'une explication possible, il est donc difficile d'évaluer à quel point cela est probable.

De manière similaire, pour la concentration Arts, certaines études indiquent que le fait de prendre part à des programmes artistiques et culturels pourrait aussi avoir une influence sur les fonctions exécutives (Andersen *et al.*, 2019; Moss *et al.*, 2018; Park *et al.*, 2015). Comme pour la concentration Sports, il existe donc une possibilité que l'engagement dans des activités artistiques au sein de la concentration Arts puisse favoriser le développement des compétences liées aux fonctions exécutives, et que cela puisse

potentiellement favoriser les apprentissages scolaires des élèves. Cependant, les résultats à ce sujet sont également limités et présentent des divergences, ce qui laisse planer un doute sur cette éventualité.

Qui plus est, si cela s'avère être le cas et que les concentrations Sports et Arts sont associées à un transfert positif en raison de leur influence sur les fonctions exécutives, il apparaît peu probable que cela soit le cas uniquement pour les disciplines des sciences et du français. Il est en effet raisonnable de penser qu'elles pourraient également avoir un impact sur la discipline des mathématiques. Pourtant, à la différence des sciences et du français, les résultats suggèrent un effet de transfert positif pour les mathématiques, ce qui souligne que cette discipline semble dans tous les cas associée à un effet de transfert plus important. Bien que cette réflexion relative aux effets possibles des groupes de contrôle actifs sur les résultats scolaires soit intéressante d'un point de vue théorique et méthodologique, les éléments discutés nous ramènent tout de même à la piste d'interprétation centrale suggérant que les mathématiques entretiendraient un lien privilégié avec la programmation, ce qui expliquerait un possible transfert positif entre ces domaines.

5.3 Limites de l'étude

Lors de l'élaboration du devis de cette recherche, les choix ont été faits en portant une attention particulière à la rigueur méthodologique, tout en tenant compte de diverses contraintes logistiques. Le terrain de recherche a d'abord été sélectionné au regard de critères précis permettant de répondre à la question de recherche. Le choix de ce terrain a d'ailleurs permis de mettre en place un devis incluant deux groupes de contrôle actifs, comparables au groupe d'intérêt au regard de plusieurs critères. Le devis mis en place a également permis de surmonter les limites les plus importantes identifiées au terme de l'état des connaissances : l'étude du transfert a ainsi été réalisée de manière longitudinale et l'intervention en programmation était d'une durée importante. Or, en dépit des efforts consciencieux déployés pour mettre en place ces choix méthodologiques, cette étude présente tout de même certaines limites qui incitent à faire preuve de prudence dans l'interprétation des résultats.

Une première limite concerne d'abord les caractéristiques de l'échantillon et l'équivalence des groupes. Étant donné que cette étude porte sur les effets de transfert associés à l'apprentissage de la programmation, il apparaît important de reconnaître que les préférences et aptitudes préalables des élèves auraient pu influencer leurs performances scolaires. De ce fait, puisque cette étude a utilisé des données secondaires qui proviennent d'un contexte scolaire authentique, la répartition des élèves au sein des concentrations n'a pas été contrôlée. Bien que plusieurs caractéristiques apparaissent équivalentes

entre les concentrations, notamment sur le plan de leur structure et de leur organisation temporelle (voir Tableau 3.1 pour une synthèse), certaines différences pourraient néanmoins provenir du fait que les élèves choisissaient eux-mêmes de s'inscrire à la concentration de leur choix au début de leur parcours au secondaire. Or, cette attribution non aléatoire des élèves aux concentrations peut introduire un certain biais de sélection, susceptible d'influencer les performances scolaires observées. En effet, il se pourrait que les participants qui choisissent une concentration spécifique soient différents de ceux qui choisissent d'autres concentrations en termes de caractéristiques personnelles, telles que le niveau initial de compétences, le niveau de motivation ou les intérêts. Cela peut donc faire en sorte que les différences observées dans les résultats scolaires ne soient pas uniquement attribuables à la concentration elle-même, mais aussi à d'autres facteurs. Il est d'ailleurs probable que les élèves aient choisi leur concentration en fonction de leurs intérêts personnels ou encore de compétences particulières. Par exemple, il se peut que les élèves ayant un penchant ou de la facilité pour les mathématiques puissent être davantage attirés par la concentration Programmation. Cette préférence initiale pour les mathématiques pourrait alors fausser l'appréciation de l'impact réel de la concentration Programmation sur les performances scolaires de cette discipline. Il apparaît cependant important de mentionner que, selon les données recueillies au temps 1 de cette étude, les participants inscrits à la concentration Programmation ne présentaient pas de meilleurs résultats scolaires en mathématiques, comparativement aux élèves des autres concentrations. Il est donc peu probable que les résultats obtenus puissent s'expliquer par une prédisposition plus importante des élèves de la concentration Programmation pour les mathématiques.

Il convient tout de même de reconnaître que les participants qui choisissent une concentration basée sur leurs intérêts peuvent être intrinsèquement plus motivés à réussir dans cette discipline, ce qui peut influencer leurs performances scolaires. Par conséquent, les différences dans les résultats scolaires entre les concentrations peuvent être dues en partie à des différences intrinsèques de motivation plutôt qu'à des différences relatives aux concentrations elles-mêmes.

De la même façon, il est possible que les élèves d'une concentration spécifique manifestent un niveau de motivation ou d'engagement général plus élevé vis-à-vis de leur apprentissage, ce qui pourrait influencer positivement leurs performances dans d'autres disciplines. À cet égard, les résultats montrent d'ailleurs que les étudiants de la concentration Arts avaient initialement des performances supérieures à ceux des concentrations Sports et Programmation pour l'ensemble des disciplines. Cette disparité initiale des performances pourrait refléter un niveau de motivation ou d'engagement plus élevé chez les étudiants de

la concentration Arts, pouvant potentiellement influencer leurs résultats scolaires. Il est aussi possible que les élèves intéressés par les arts possèdent des compétences ou des intérêts qui facilitent leur réussite scolaire, comme une créativité accrue ou encore une meilleure capacité à visualiser et à interpréter des concepts complexes. Il se peut également que les élèves de la concentration Arts proviennent de milieux familiaux qui valorisent et encouragent les activités artistiques. Ces familles pourraient fournir non seulement un soutien spécifique à ces activités (comme des leçons supplémentaires de musique ou de danse), mais aussi un environnement général qui favorise l'apprentissage et la réussite scolaire. Par ailleurs, il est aussi possible que le choix d'une concentration puisse être influencé par des facteurs externes, tels que les attentes des parents, ce qui, à l'inverse, pourrait conduire à un manque d'engagement ou de motivation de l'élève vis-à-vis de la concentration choisie.

Néanmoins, en se concentrant sur la variation de la trajectoire des performances des élèves au fil du temps, plutôt que sur les performances absolues à un moment donné, les analyses parviennent toutefois à neutraliser en partie les différences de performance initiales entre les concentrations. Le type d'analyse réalisée permet en effet d'évaluer comment les performances évoluent dans le temps et si l'appartenance à une concentration spécifique influence cette évolution. Par exemple, si les élèves d'une concentration particulière montrent une amélioration ou une détérioration plus marquée de leurs performances au fil du temps par rapport aux autres concentrations, cela suggère un effet potentiel de la concentration, indépendamment des niveaux de performance initiaux. Ainsi, même si certains élèves étaient initialement plus performants, l'analyse se concentre sur la manière dont leur performance change selon leur appartenance à une concentration donnée, offrant ainsi une perspective plus dynamique et nuancée de l'effet de chacune des concentrations.

Il n'en demeure pas moins que sans un contrôle direct des variables pouvant influencer le choix de la concentration, telles que les résultats scolaires antérieurs, les intérêts personnels ou le contexte socio-économique, il n'est pas possible d'évaluer si celles-ci se distinguent au regard d'autres variables que leur contenu d'apprentissage respectif. Il est donc important de prendre en compte cette limite dans l'interprétation des résultats et d'envisager ce type de contrôle pour des recherches futures afin d'obtenir une compréhension plus nuancée de l'impact des concentrations sur les apprentissages des élèves.

Finalement, toujours au regard de l'équivalence des concentrations, il a été relevé que la proportion de garçons et de filles n'était pas équivalente au sein de chacune des concentrations (voir Tableau 3.2), la

différence étant particulièrement marquée pour les concentrations Programmation (72 % de garçons) et Arts (81 % de filles), la concentration Sports étant plus balancée (52 % de filles). Cela représente une limite dans la mesure où un biais de genre pourrait influencer les résultats : les différences observées dans les performances scolaires entre les concentrations pourraient ainsi être influencées par des différences liées au genre plutôt que par le contenu d'apprentissage propre à chaque concentration. Plus particulièrement, comme la concentration Programmation compte principalement des garçons et que la concentration Arts compte principalement des filles, les différences observées dans les résultats entre ces concentrations pourraient en partie être attribuables à des différences de genre. Ces différences de genre pourraient ainsi introduire des effets confondants, où d'autres facteurs liés au genre, tels que des différences relativement à la perception de compétence découlant de stéréotypes de genre (p. ex., Witherspoon et Schunn, 2021), pourraient influencer les résultats de l'étude et ajouter du bruit à l'interprétation des effets de transfert associés à l'apprentissage de la programmation.

Par ailleurs, un autre aspect qu'il apparaît important de reconnaître concerne le fait que l'ensemble des données utilisées provenaient exclusivement d'une seule école, qui plus est une école privée. Bien qu'il s'agissait d'un terrain de recherche privilégié pour faire l'étude de notre question de recherche, cela est susceptible d'entraîner un certain nombre de limites. D'abord, il est probable que les politiques, les modalités de sélection, les pratiques pédagogiques, les ressources disponibles, les enseignants et l'environnement de cette école privée diffèrent considérablement des écoles publiques ou même d'autres institutions privées. Par conséquent, les résultats de l'étude pourraient être influencés par des facteurs institutionnels spécifiques à cette école particulière, ce qui pourrait limiter la validité des conclusions. De la même façon, il est possible que les caractéristiques des élèves, particulièrement leur statut socio-économique, diffèrent considérablement de celles d'autres écoles privées ou publiques. Les résultats de l'étude ne peuvent donc pas être considérés comme étant représentatifs de la population des élèves du secondaire dans son ensemble et peuvent difficilement être généralisés à d'autres contextes.

Un autre élément à considérer relativement aux données de recherche, qui a déjà été abordé dans la section discutant des résultats obtenus en sciences, concerne le fait que seuls les résultats de fin d'année scolaire (secondaire 1 à 5) ont été utilisés dans le cadre de ce projet. Bien que ces mesures permettent de rendre compte de la trajectoire de la performance de chaque discipline à travers l'ensemble du parcours au secondaire, celles-ci présentent néanmoins une résolution temporelle limitée. En effet, se baser sur les résultats annuels offre une image relativement statique et ne permet pas de saisir en détail toutes les

fluctuations ou les progrès qui pourraient survenir à des moments spécifiques de l'année. Une analyse plus fréquente, par exemple pour chacune des étapes de chaque année scolaire, aurait pu fournir des informations plus détaillées sur la dynamique du processus d'apprentissage pour chaque concentration et aurait peut-être même permis d'identifier des périodes où le transfert est le plus probable. Qui plus est, des mesures plus granulaires relativement à chaque discipline auraient sans doute permis de réaliser une analyse plus fine des effets de transfert, y compris la capacité de discerner les effets sur des compétences ou des contenus plus ciblés pour chaque discipline.

Ensuite, il apparaît important de mentionner que l'attrition expérimentale qui a eu lieu à chacune des années scolaires a fait diminuer la taille de l'échantillon de près de 17 % au T5 (voir Tableau 3.4 pour les données complètes liées à l'attrition expérimentale). Cette attrition a pu contribuer à diminuer la puissance statistique de l'étude et à rendre plus difficile la détection d'effets réels associés à l'apprentissage de programmation sur les autres disciplines, en particulier si l'effet attendu est subtil ou modéré. Par ailleurs, le nombre de participants au sein de l'échantillon final demeurait suffisant pour mener les analyses envisagées (Fan *et al.*, 2003; Shi *et al.*, 2021; Soper *et al.*, 2023). De plus, il est important de rappeler que l'attrition était distribuée de manière aléatoire au sein des concentrations, ce qui fait en sorte qu'elle ne pose pas de limite majeure. En effet, si cette attrition n'avait pas été aléatoire et avait été observée de manière plus importante pour une concentration, cela aurait entraîné un biais dans les résultats : les participants restants n'auraient alors pas été représentatifs de la population initiale. Or, considérant la distribution aléatoire de l'attrition, il semble que celle-ci ne soit pas attribuable aux concentrations elles-mêmes, ce qui renforce la fiabilité des comparaisons entre les concentrations effectuées. Nous émettons l'hypothèse qu'il est possible que l'attrition observée soit due, au moins en partie, au départ de certains élèves, non pas de la concentration, mais de l'école elle-même. Celle-ci exige en effet que les élèves maintiennent une bonne moyenne académique pour demeurer admis.

Finalement, bien qu'il ne s'agisse pas d'une limite à proprement parler, nous estimons qu'il aurait été intéressant d'obtenir une mesure de la pensée informatique en plus des mesures de performance scolaire. En effet, puisque la pensée informatique est conceptualisée comme une variable médiatrice entre l'apprentissage de la programmation et les performances des autres disciplines scolaires, une telle mesure aurait permis d'examiner le processus médiat par lequel l'apprentissage de la programmation est susceptible d'influencer les performances scolaires. Cela aurait apporté des informations supplémentaires sur la manière dont les compétences acquises en programmation peuvent se transposer concrètement

dans les compétences mobilisées par d'autres domaines d'étude. Cela aurait également fourni des données empiriques pour étayer ou remettre en question l'hypothèse selon laquelle la pensée informatique joue un rôle de médiateur entre l'apprentissage de la programmation et les performances dans d'autres disciplines. Qui plus est, cette mesure aurait pu renforcer l'inférence du transfert de manière plus robuste, étant donné que la pensée informatique constitue le construit théorique sur lequel repose le transfert. De plus, en évaluant spécifiquement les différentes composantes de la pensée informatique, il aurait été possible de mieux comprendre, d'une part, comment l'apprentissage de la programmation se traduit en termes de résolution de problème, d'abstraction, de créativité, etc., et, d'autre part, comment ces composantes influencent les apprentissages des autres disciplines. Cependant, la mesure de la pensée informatique n'était pas réalisable dans le cadre des analyses a posteriori de cette thèse utilisant des données secondaires.

5.4 Principales forces de la présente recherche

Bien que certains choix méthodologiques effectués soient associés à des limites qui viennent d'être discutées, plusieurs de ces choix ont été faits, car ils apportent en contrepartie certaines forces au projet de recherche, lesquelles méritent aussi d'être brièvement présentées.

D'abord, en explorant les effets de l'apprentissage de la programmation sur trois disciplines scolaires au sein de la même étude, l'analyse permet d'aborder de manière plus complète les potentialités et les limites du transfert associé à l'apprentissage de la programmation vers d'autres disciplines. De plus, regrouper l'étude des effets de transfert associés à l'apprentissage de la programmation pour plusieurs disciplines au sein d'une même étude présente un avantage considérable en matière d'uniformité de l'intervention. Cette uniformité renforce la validité des résultats en réduisant les sources de variabilité et en permettant une comparaison plus précise des effets de la programmation entre les différentes disciplines. Cela facilite ainsi l'identification de tendances générales et de différences spécifiques entre les disciplines, offrant une perspective plus complète sur les effets possibles de la programmation.

Ensuite, l'approche longitudinale de cette recherche, s'étendant sur l'ensemble du parcours au secondaire, est d'autant plus intéressante en raison de la durée prolongée de l'intervention en programmation qui prend place sur trois années consécutives. Une telle intervention prolongée permet de favoriser le développement de compétences plus solides et approfondies, qui vont au-delà d'une familiarisation superficielle pouvant découler d'une intervention plus limitée. Cela offre aussi plus d'opportunités de

pratique et de consolidation des apprentissages, ce qui est de nature à favoriser le transfert. Qui plus est, un tel devis longitudinal permet non seulement de suivre les effets immédiats de l'apprentissage de la programmation, mais aussi d'observer comment cet apprentissage évolue et s'intègre dans le développement scolaire plus global des élèves sur une période étendue. D'ailleurs, dans le cas de cette étude, il a été possible de constater que l'effet de transfert positif associé à l'apprentissage de la programmation pour la compétence 1 en mathématiques semblait perdurer jusqu'à deux après l'exposition à la concentration. Cela renforce donc l'idée que la durée prolongée de l'intervention a mené à des apprentissages consolidés. Ce type de devis est également avantageux considérant que les effets sur les performances scolaires peuvent prendre du temps à se manifester et à être pleinement réalisés. Un devis longitudinal sur cinq ans offre donc une perspective approfondie et plus nuancée des effets de transfert de l'apprentissage de la programmation.

Une autre force de l'étude réside dans le fait qu'elle utilise des données collectées en contexte scolaire authentique. Cela signifie que les données reflètent les performances réelles des élèves dans leur environnement d'apprentissage habituel, ce qui permet d'observer comment les effets de transfert associés à l'apprentissage de la programmation se traduisent dans des situations concrètes et réelles. L'utilisation de données issues d'un contexte authentique améliore également la validité écologique des résultats : les conclusions tirées à partir de ces données sont plus susceptibles d'être représentatives de contextes éducatifs réels. De plus, il devient possible de mieux contextualiser les résultats. Sachant que ceux-ci suggèrent un effet de transfert entre la programmation et les mathématiques, il pourrait ainsi devenir pertinent d'analyser plus en profondeur le contenu de la concentration Programmation de l'école participante afin d'identifier quels sont les éléments les plus susceptibles de contribuer à ce transfert ou encore de s'intéresser aux pratiques pédagogiques mises en place au sein de cette concentration afin d'orienter l'action pédagogique pour favoriser le transfert.

Finalement, cette étude présente aussi comme force d'avoir inclus deux groupes de contrôle actifs plutôt qu'un seul au sein de son devis méthodologique, ce qui offre plusieurs avantages. Cela permet de comparer les effets de la concentration Programmation à deux conditions pédagogiques différentes, ce qui renforce la robustesse des conclusions. L'utilisation de deux groupes de contrôle actifs permet aussi de mieux contrôler les facteurs de confusion potentiels qui pourraient influencer les résultats et de mieux isoler les effets spécifiques associés à la programmation. Cela permet aussi de mieux évaluer ses effets relatifs, ce qui peut aider à mieux comprendre si la concentration Programmation est plus efficace, moins

efficace ou équivalente à d'autres types de concentrations scolaires au regard du transfert des apprentissages vers les disciplines scolaires traditionnelles. De plus, si les mêmes résultats sont observés pour les deux groupes de contrôle, cela renforce la confiance dans les conclusions de l'étude. Dans la présente étude, il a d'ailleurs été possible d'observer que les résultats scolaires en mathématiques des élèves de la concentration Programmation se distinguent à la fois de ceux des concentrations Arts et Sports.

5.5 Considérations éducatives relatives à l'apprentissage de la programmation

Les résultats obtenus apportent une perspective nuancée sur l'intégration de l'apprentissage de la programmation au sein des programmes d'études. En effet, bien que les résultats suggèrent un effet transfert positif associé à la programmation vers les compétences en résolution de problèmes en mathématiques, cela ne semble pas être le cas pour les disciplines des sciences et du français. Ce constat invite donc à une discussion plus approfondie sur les implications éducatives qui pourraient en découler.

À la lumière des résultats obtenus, il semble d'abord pertinent de chercher à concevoir des activités pédagogiques qui exploitent les similarités qu'entretiennent la programmation et les mathématiques et qui pourraient être bénéfiques sur le plan du transfert entre ces domaines. Cette approche revêt d'ailleurs un intérêt particulier pour un système éducatif comme celui du Québec, où la programmation est intégrée de manière transversale dans le programme éducatif plutôt que d'être considérée comme une discipline distincte. Par ailleurs, afin de maximiser les bénéfices de l'apprentissage de la programmation, une formation adéquate des enseignants est également cruciale. Cette formation devrait porter non seulement sur les aspects techniques de la programmation, mais aussi sur les stratégies pédagogiques permettant de mettre en évidence les liens entre la programmation et les concepts mathématiques.

Par ailleurs, cette proposition doit également être considérée de manière nuancée, en prenant en compte la littérature scientifique qui examine l'usage de la programmation comme un outil pédagogique dans l'enseignement des mathématiques. À cet égard, une étude de Laurent *et al.* (2022) a exploré les effets de l'apprentissage de la programmation avec Scratch sur l'apprentissage des mathématiques chez des élèves de quatrième et cinquième année du primaire (n = 2 472). Le devis de cette recherche prévoyait la comparaison d'un groupe expérimental qui utilisait Scratch comme outil pédagogique pour faire des mathématiques à un groupe de contrôle qui prenait part à un des activités pédagogiques en mathématiques plus traditionnelles. Or, les résultats obtenus par Laurent et ses collaborateurs (2022) ont révélé une observation inattendue : alors qu'il était présumé que l'utilisation de Scratch favoriserait les

apprentissages en mathématiques, les résultats ont mis en évidence des progrès en mathématiques moins importants pour le groupe expérimental, par rapport au groupe de contrôle. Ces résultats suggèrent que l'intégration de la programmation dans l'enseignement des mathématiques, du moins dans le contexte de cette étude, n'a pas produit les améliorations escomptées et aurait même pu être contre-productive. Les auteurs avancent plusieurs hypothèses pour expliquer ces résultats inattendus, notamment la possibilité que l'attention portée à l'apprentissage de la programmation ait détourné les élèves de la compréhension des concepts mathématiques essentiels, ou que la complexité supplémentaire introduite par l'utilisation de Scratch ait imposé une charge cognitive supplémentaire, entravant ainsi les apprentissages en mathématiques.

Cette étude met donc en lumière un point essentiel qui est complémentaire aux résultats de cette thèse : pour améliorer les compétences mathématiques des élèves, il n'est pas nécessairement plus efficace de mobiliser la programmation. Nos résultats indiquent en effet que les apprentissages réalisés en programmation pourraient être bénéfiques pour les mathématiques, mais ils n'indiquent pas qu'il s'agit de la meilleure façon de favoriser les apprentissages en mathématiques. Les résultats de Laurent et ses collègues (2022) suggèrent plutôt que consacrer davantage de temps à des activités mathématiques ciblées demeure probablement plus efficace pour l'apprentissage que d'intégrer la programmation dans l'enseignement des mathématiques.

Il est donc crucial de reconnaître que l'intégration de la programmation dans l'enseignement des mathématiques doit être soigneusement réfléchie. Les défis potentiels tels que la distraction des concepts mathématiques essentiels ou la surcharge cognitive introduite par l'utilisation d'outils comme Scratch doivent être pris en compte et atténués par des stratégies pédagogiques appropriées. À cet égard, il est possible que l'intégration de la programmation en mathématiques soit moins optimale lorsqu'il s'agit d'aborder de nouvelles notions ou des notions complexes qui nécessitent déjà une charge cognitive importante (Ayres, 2006; Dhlamini, 2016). Dans de tels cas, l'ajout de la programmation pourrait distraire les élèves des concepts mathématiques essentiels et entraver leur compréhension. L'intégration de la programmation pourrait être plus appropriée dans une optique de consolidation, où les élèves ont déjà une certaine familiarité avec les concepts mathématiques abordés. Dans ce contexte, la programmation pourrait servir de moyen supplémentaire pour renforcer et appliquer les compétences mathématiques déjà acquises, offrant ainsi une approche plus pratique et concrète pour la résolution de problèmes mathématiques. En somme, nous croyons que l'intégration de la programmation dans l'enseignement des

mathématiques ne constitue pas une solution universelle, mais plutôt une approche qui doit servir des objectifs éducatifs précis et qui nécessite une mise en œuvre réfléchie et adaptative pour être efficace.

De plus, l'absence de résultats pour les disciplines des sciences et du français souligne que le transfert n'est peut-être pas aussi étendu ou systématique que cela aurait pu être envisagé. Cela remet en perspective la décision de vouloir intégrer la programmation au cursus des élèves sur l'unique base d'un transfert probable vers d'autres disciplines. En effet, il semble que les résultats de la présente étude amènent à nuancer cette idée très répandue que l'apprentissage de la programmation contribuerait au développement de la pensée informatique, ce qui faciliterait d'emblée un apprentissage transversal dans d'autres disciplines (Shute *et al.*, 2017; Wing, 2006). Par conséquent, l'intégration de la programmation doit également reposer sur d'autres arguments, tels que ceux liés aux besoins du marché du travail et à l'équité sociale, qui ont été discutés dans la problématique de cette thèse.

D'un point de vue éducatif, il pourrait donc aussi être pertinent de réfléchir l'intégration de la programmation dans la lignée de la mission de qualification de l'École québécoise. En effet, l'apprentissage de la programmation représente une compétence essentielle pour les futurs citoyens qui envisagent d'intégrer un marché du travail en constante évolution, où les compétences numériques sont de plus en plus valorisées. De plus, en offrant une exposition à la programmation à tous les élèves dans le cadre de leur cursus scolaire, l'école peut également jouer un rôle crucial en tant que moteur d'équité sociale. Cela pourrait d'ailleurs être particulièrement bénéfique pour les élèves qui sont peu exposés au numérique à l'extérieur de l'école (Lewis, 2019), contribuant ainsi à réduire le fossé numérique qui persiste dans la société (diSessa, 2000). En garantissant un accès équitable à la programmation dès le plus jeune âge, l'école peut favoriser l'inclusion et permettre à tous les élèves de développer les compétences nécessaires pour réussir dans un monde de plus en plus numérique.

Finalement, tel que cela a été brièvement évoqué, l'absence de transfert en sciences et en français ne signifie pas forcément que l'apprentissage de la programmation ne peut pas contribuer à l'amélioration de certaines compétences spécifiques mobilisées par ces disciplines. Il serait donc judicieux d'explorer comment la programmation pourrait être utilisée pour soutenir des aspects plus ciblés de l'apprentissage du français ou des sciences, pour lesquels les similarités sont plus évidentes.

De plus, il serait également intéressant de réfléchir à la chronologie (au « timing ») et à la progression de l'intégration de la programmation dans le programme d'enseignement pour ces disciplines. Il est en effet

possible que certains effets de la programmation ne deviennent visibles qu'à un stade plus avancé du parcours scolaire, ou, à l'inverse, que le transfert soit plus probable à un jeune âge. Cependant, ces aspects nécessitent des recherches supplémentaires avant de pouvoir éclairer la pratique. La section qui suit poursuit donc cette réflexion en examinant plusieurs perspectives de recherches futures.

5.6 Perspectives de recherches futures

Cette recherche visait à vérifier s'il existe un effet de transfert associé à l'apprentissage de la programmation vers les disciplines scolaires traditionnelles. Tel que cela vient d'être abordé, les résultats obtenus proposent déjà quelques pistes de réflexion intéressantes, notamment au regard du transfert vers la discipline des mathématiques. Or, pour aspirer à mieux comprendre de quelle façon l'apprentissage de la programmation peut être profitable aux autres apprentissages scolaires des élèves, ainsi que les implications pratiques relativement à son intégration transversale au sein du programme éducatif, d'autres recherches apparaissent nécessaires. Les sections qui suivent visent à discuter de pistes de recherches futures qui apparaissent centrales.

5.6.1 Identifier les contenus disciplinaires susceptibles de profiter le plus de l'apprentissage de la programmation et évaluer les effets de transfert pour ces contenus

Comme souligné précédemment, il est raisonnable de supposer que le transfert associé à l'apprentissage de la programmation ne s'opère pas de manière uniforme sur l'ensemble des contenus d'une discipline, mais possiblement de manière plus significative pour certains éléments de contenu spécifiques qui sont plus proches de la programmation. Cette hypothèse soulève donc un point crucial pour les recherches futures : l'importance d'identifier les contenus disciplinaires qui apparaissent les plus susceptibles de bénéficier de l'apprentissage de la programmation. En effet, en ciblant spécifiquement les points d'ancrage entre la programmation et d'autres disciplines, il pourrait être possible d'observer davantage de transfert et de mieux identifier les retombées possibles de l'apprentissage de la programmation sur les contenus faisant partie des programmes d'études.

Dans la littérature scientifique existante, des efforts ont déjà été entrepris pour amorcer l'identification des contenus disciplinaires susceptibles de profiter davantage de l'apprentissage de la programmation, particulièrement pour les disciplines des sciences et des langues. En sciences, la littérature identifie déjà certains points d'ancrage entre la programmation et différents concepts scientifiques. Les modèles de Weintrop (2016) et de Barr et Stephenson (2011) illustrent d'ailleurs comment des éléments de la pensée

informatique issus de l'apprentissage de la programmation pourraient être réinvestis. De même, une revue de la littérature de Wang *et al.* (2022) a aussi cherché à comprendre comment intégrer efficacement la pensée informatique dans l'enseignement des sciences et a dégagé différentes façons d'opérationnaliser cette intégration, notamment via la programmation. Pour ce qui est des langues, certains auteurs ont suggéré des liens potentiels entre l'apprentissage de la programmation et certaines compétences linguistiques, notamment la mécanique de la langue, la compréhension de lecture et la créativité verbale (Clements, 1999). De plus, Hassenfeld et Bers (2020) ont mis en évidence des similarités entre l'écriture et la programmation qui pourraient avoir en commun des processus tels que la planification, la création, le test et l'évaluation.

Une cartographie approfondie des contenus disciplinaires partageant des liens privilégiés avec la programmation constituerait donc une première avenue de recherche essentielle. Une fois cette cartographie établie, il serait pertinent d'évaluer les effets de transfert possibles pour ces contenus spécifiques. Cela ouvrirait la voie au développement d'interventions pédagogiques plus ciblées, prenant en compte les contenus pour lesquels le transfert s'avère favorable. Cette approche permettrait donc aux enseignants d'avoir de meilleurs points de repère quant aux relations entre la programmation et les autres contenus enseignés et possiblement de mieux intégrer la programmation dans leur enseignement. Cela fournirait aussi des arguments plus solides aux décideurs politiques qui réfléchissent à l'intégration de la programmation au sein des programmes d'études, au Québec et ailleurs.

5.6.2 Mettre à l'essai des interventions éducatives établissant des liens multidisciplinaires explicites

Après avoir identifié les contenus disciplinaires pouvant bénéficier le plus de l'apprentissage de la programmation, une autre avenue prometteuse pour faciliter le transfert éloigné consisterait à élaborer et mettre à l'essai des interventions éducatives visant directement à favoriser le transfert des apprentissages en programmation (Robins *et al.*, 2019). Cette idée découle de l'observation selon laquelle réaliser des apprentissages et transférer des apprentissages sont deux processus cognitifs distincts (Barnett et Ceci, 2002; van Merrienboer *et al*., 2002). Par conséquent, les interventions éducatives devraient non seulement viser l'acquisition de compétences en programmation, mais également guider les apprenants dans l'établissement de liens explicites entre la programmation et d'autres disciplines (Robins *et al.*, 2019). Cette approche permettrait aux apprenants de comprendre comment les compétences et les connaissances acquises en programmation peuvent être mobilisées dans diverses disciplines, favorisant ainsi un transfert plus spontané des apprentissages.

Par exemple, en utilisant des stratégies telles que l'encodage analogique, où les élèves sont encouragés à créer des analogies entre les apprentissages en programmation et ceux d'autres disciplines (Ferguson, 1994), il pourrait être possible de renforcer leur capacité à transférer leurs apprentissages de manière plus flexible et adaptative. Une approche par sous-objectifs, qui vise à décomposer les problèmes complexes en composantes fonctionnelles plus petites et explicites (Margulieux *et al*., 2012), pourrait aussi être utilisée pour souligner les étapes de raisonnement communes entre la programmation et les autres disciplines. En identifiant plus clairement ces sous-objectifs dans les activités de programmation, il est possible que les élèves parviennent à appliquer ces mêmes stratégies de résolution de problèmes dans d'autres contextes, facilitant possiblement le transfert. Une autre avenue pourrait aussi être d'accentuer les similarités de surface de façon à ce qu'elles deviennent plus apparentes pour les apprenants. Une étude de Gick et Holyoak (1980) a permis de constater que des apprenants qui étaient initialement incapables de résoudre un problème dans un contexte nouveau devenaient capables de trouver la solution une fois que les similarités étaient mises de l'avant, ce qui suggère que le fait de rendre plus explicites ces similarités contribue à faciliter le transfert des apprentissages. Finalement, il est également reconnu que les apprenants novices en programmation vivent souvent une surcharge cognitive (Robins *et al*., 2019). Développer des interventions pédagogiques visant spécifiquement à réduire la charge cognitive des apprenants lors des premières phases de leur apprentissage de la programmation pourrait se révéler comme une avenue intéressante afin de maximiser les premiers apprentissages (Lapierre *et al.*, 2023).

Robins et ses collaborateurs (2019) soulignent d'ailleurs qu'aucune étude ne semble s'être intéressée à cette avenue de tester des interventions pédagogiques visant à favoriser le transfert associé à l'apprentissage de la programmation vers d'autres disciplines. Toutefois, à notre connaissance, les récents travaux de Bers (Bers, 2019; Bers *et al.*, 2023) reprennent notamment cette idée avec son approche Coding as Another Language (CAL) où l'enseignement de la programmation s'effectue comme s'il s'agissait d'un langage naturel, et donc, où des parallèles explicites sont établis entre la programmation et les langues. Les résultats de l'étude de Hassenfeld *et al.* (2020) qui a mis à l'essai l'approche CAL, suggèrent d'ailleurs que l'apprentissage de la programmation via l'enseignement des langues pourrait avoir un impact positif sur les compétences en langues des élèves. Cependant, bien que prometteuse, cette étude ne permet pas d'établir une distinction claire entre les effets de transfert et les effets d'entraînement direct; c'est d'ailleurs pour cette raison que les récents travaux de Bers et son équipe n'ont volontairement pas été intégrés à notre état des connaissances antérieur. Il apparaît donc essentiel de concevoir et tester des

interventions éducatives qui établissent de manière explicite des liens interdisciplinaires, tout en permettant de distinguer plus clairement ces deux types d'effets.

5.6.3 Reproduire le devis de recherche en incluant une mesure de la pensée informatique

Tel que cela a déjà été évoqué, une autre piste de recherche future consisterait à reproduire le devis de recherche actuel en y intégrant explicitement la mesure de la pensée informatique. Cela permettrait de saisir avec plus de précision les mécanismes par lesquels l'apprentissage de la programmation est susceptible d'influencer les performances scolaires dans diverses disciplines. L'inclusion d'une telle mesure offrirait la possibilité de vérifier si la pensée informatique agit réellement comme un médiateur dans le transfert des compétences de la programmation vers d'autres disciplines scolaires.

Il existe d'ailleurs des outils de mesure validés de la pensée informatique, tels que le Computational Thinking Test (CTt) (Román-González, 2015; Román-González *et al.*, 2017) destiné à des participants âgés entre 10 et 16 ans. Composé de 28 items, le CTt est spécifiquement conçu pour évaluer les composantes de la pensée informatique par le biais de concepts de base en programmation tels que les boucles, les conditionnels, les boucles conditionnelles et les fonctions simples. L'intégration d'un outil mesurant la pensée informatique, tel que le CTt dans un devis de recherche semblable permettrait non seulement de mesurer les effets de l'apprentissage de la programmation sur la pensée informatique, mais aussi d'explorer comment cette dernière peut contribuer au transfert, particulièrement pour la discipline des mathématiques. En précisant le rôle de la pensée informatique comme potentiel mécanisme de transfert, les chercheurs pourraient ainsi affiner les stratégies pédagogiques destinées à maximiser les bénéfices éducatifs de l'apprentissage de la programmation.

5.6.4 Reproduire le devis de recherche dans des contextes différents

La reproductibilité des résultats représente un élément essentiel de la recherche scientifique, assurant la validation et la généralisation des conclusions à différents contextes. Par conséquent, la réplication du devis de cette étude dans divers environnements éducatifs revêt une importance cruciale. Cette démarche vise non seulement à approfondir notre compréhension des effets de transfert de manière générale, mais également à évaluer la portée des résultats obtenus. Plus spécifiquement, la reproduction de cette étude dans une gamme variée de contextes éducatifs, tels que des écoles primaires, des établissements scolaires publics et des milieux socio-économiques défavorisés, pourrait enrichir notre compréhension de la

manière dont les effets de transfert associés à l'apprentissage de la programmation peuvent se manifester et influencer les performances scolaires des élèves.

Une réplication plus étendue du présent devis de recherche dans plusieurs milieux offrirait également la possibilité de conduire des analyses sur un échantillon de plus grande taille. Cela permettrait d'inclure dans les analyses des covariables supplémentaires, telles que le genre des participants, ce qui ne pouvait pas être fait dans le cadre de la présente étude. L'inclusion de ces variables supplémentaires permettrait ainsi de mieux contrôler les effets de variables potentiellement confondantes et d'isoler de manière plus précise les effets potentiels de l'apprentissage de la programmation sur la performance scolaire. Cette approche pourrait notamment contribuer à déterminer si certains effets de transfert observés sont uniformément distribués ou s'ils varient en fonction de caractéristiques spécifiques des élèves.

De façon complémentaire, il pourrait être intéressant de chercher à distinguer les effets de transfert associés à l'apprentissage de la programmation pour différents types d'élèves. Des résultats (Lindh et Holgersson, 2007) suggèrent en effet que l'apprentissage de la programmation via la robotique éducative pourrait influencer différemment le développement de la compétence en résolution de problèmes selon que les élèves soient très performants, moyennement performants ou peu performants. Comprendre les effets distincts du transfert associés à l'apprentissage de la programmation pour différentes catégories d'élèves pourrait donc favoriser une intégration plus différenciée de la programmation de manière à favoriser l'apprentissage d'un plus grand nombre d'élèves et à assurer une meilleure inclusion scolaire.

Enfin, explorer les effets de transfert associés à l'apprentissage de la programmation pour d'autres disciplines scolaires, comme l'histoire, la géographie, l'éthique et les arts, où la programmation a commencé à être intégrée de manière transversale (Aguirre-Muñoz et Pantoya, 2016; Maguth, 2012; Sullivan et Bers, 2017; *Sullivan et al*., 2017), permettrait de vérifier son potentiel de transfert pour un spectre plus large de connaissances et de compétences scolaires. Cette avenue revêt une importance particulière dans le contexte québécois où l'intégration transversale de la programmation dans le programme scolaire a été privilégiée.

En somme, la reproduction du présent devis de recherche dans des contextes différents apparaît essentielle pour approfondir et valider les conclusions issues de cette étude, qui a été menée dans le contexte spécifique d'une école secondaire privée à Montréal, pour un nombre limité de disciplines scolaires.

CONCLUSION

Tel que cela a été discuté dans le premier chapitre de cette thèse présentant la problématique, l'apprentissage de la programmation a connu dans les dernières années un renouveau marqué au sein des programmes éducatifs, tant au niveau primaire que secondaire, et ce, à l'échelle mondiale. Cette tendance reflète d'une part une prise de conscience croissante de l'importance des compétences en programmation dans une société de plus en plus orientée vers la technologie. D'autre part, le principal argument avancé pour justifier l'intégration de l'apprentissage de la programmation au sein du cursus scolaire des élèves repose sur la conviction que cela favoriserait le développement d'une « pensée informatique » englobant des compétences cruciales telles que la pensée logique, la résolution de problèmes et la créativité, qui transcendent le domaine de l'informatique et pourraient de ce fait être transférées et faciliter l'apprentissage d'autres disciplines scolaires, telles que les mathématiques, les sciences ou encore les langues. Au Québec, le Cadre de référence de la compétence numérique (2019) mentionne d'ailleurs précisément que l'élève doit « développer sa pensée informatique, notamment par le développement de sa compréhension et de ses habiletés à l'égard de la programmation informatique » (p. 14).

Cependant, bien que l'idée de développer une telle pensée informatique et des compétences transférables apparaisse convaincante sur le plan conceptuel, la recherche empirique sur les bénéfices concrets de l'apprentissage de la programmation en termes de transfert vers d'autres disciplines scolaires demeure à ce jour limitée. Malgré la place centrale de cet argument, il est paradoxal de constater que peu de données claires permettent une compréhension approfondie des possibilités de transfert associées à l'apprentissage de la programmation. Cette recherche visait donc spécifiquement à combler ce vide dans les connaissances et avait donc pour objectif général d'examiner les effets de transfert possibles découlant de l'apprentissage de la programmation sur trois disciplines centrales au programme de formation au secondaire : les mathématiques, les sciences et le français. À la lumière des écrits théoriques portant sur la didactique de la programmation et sur le transfert des apprentissages ainsi que des résultats de recherche existants, les hypothèses formulées suggéraient que l'apprentissage de la programmation serait associé à un effet de transfert positif vers les mathématiques, à un effet de transfert positif, mais de moindre ampleur vers les sciences, et qu'aucun effet de transfert ne serait observé vers le français.

Pour vérifier ces hypothèses, une approche longitudinale et comparative a été adoptée. Une base de données massive a été constituée en recueillant l'ensemble des résultats scolaires de fin d'année (français,

mathématiques et sciences) de près de 450 élèves tout au long de leur parcours au secondaire. Ces données provenaient d'une même école secondaire qui a intégré une concentration scolaire précisément axée sur l'apprentissage de la programmation depuis 2014. Deux autres concentrations (Arts et Sports) ont été utilisées comme groupes de contrôle actifs afin de mieux discerner la valeur ajoutée de l'apprentissage de la programmation. Une modélisation par trajectoires latentes a été réalisée pour chaque discipline (français, mathématiques et sciences) afin de pouvoir comparer les trajectoires des résultats scolaires associés à chacune des trois disciplines, pour chaque concentration, au fil du temps.

Les résultats obtenus suggèrent que seule la discipline des mathématiques semble bénéficier d'un effet de transfert positif associé à l'apprentissage de la programmation. Plus spécifiquement, la compétence mathématique « Résoudre une situation-problème » est celle qui serait associée au transfert le plus significatif. Aucun effet de transfert positif associé à l'apprentissage de la programmation ne semble présent pour les sciences ou le français.

À la lumière de ces résultats ainsi que des résultats d'études antérieures (Scherer *et al.*, 2019) et des écrits théoriques concernant la didactique de la programmation et le transfert des apprentissages (notamment Barnett et Ceci, 2002), nous postulons que ces résultats s'expliqueraient principalement par le degré de proximité/éloignement qu'entretient chaque discipline scolaire avec la programmation. En effet, il apparaît probable que la discipline des mathématiques soit celle présentant le plus de similarités avec la programmation, favorisant ainsi une plus grande proximité entre ces deux domaines et facilitant le transfert. En revanche, des similarités moins importantes et potentiellement moins directes entre les sciences et la programmation pourraient accentuer l'éloignement entre ces disciplines, expliquant en partie l'absence de transfert positif observé. De plus, considérant que les sciences englobent de nombreuses disciplines scientifiques, il est possible que cela puisse compliquer la détection des effets de transfert. En effet, les compétences et les connaissances acquises grâce à la programmation pourraient être plus directement liées à certaines disciplines scientifiques précises, par exemple la physique, et moins à d'autres, possiblement la biologie, ce qui pourrait atténuer les résultats observés. En ce qui concerne le français, bien qu'il puisse exister quelques similarités avec la programmation, celles-ci apparaissent moins marquées que pour les mathématiques et les sciences. Il est donc probable que l'écart entre la programmation et le français soit trop important pour permettre un transfert positif significatif entre ces domaines.

La principale conclusion de cette recherche est donc que l'apprentissage de la programmation pourrait contribuer aux autres apprentissages réalisés en mathématiques, surtout ceux impliquant de la résolution de problèmes, mais les effets de transfert pour les autres disciplines scolaires semblent toutefois plus incertains. Il apparaît donc important de ne pas exagérer le rôle que pourrait jouer l'apprentissage de la programmation en tant qu'outil pour stimuler le transfert et la réussite des élèves. Ces résultats mènent ainsi à envisager l'intégration de la programmation au cursus scolaire des élèves sur la base de différents arguments qui vont au-delà de l'idée de transfert. L'intégration de la programmation dans le cursus des jeunes apprenants est en effet susceptible de les préparer de manière plus efficace à s'adapter à un monde en constante évolution, de plus en plus tourné vers la technologie. De plus, en développant leurs compétences en programmation assez tôt dans leur parcours scolaire, ils seraient mieux armés pour répondre aux exigences croissantes du marché du travail dans ce domaine. Cette initiative pourrait également favoriser une plus grande équité en offrant à tous les élèves une exposition égale aux compétences technologiques, ce qui contribuerait à promouvoir une représentation plus diversifiée dans les professions liées à la technologie.

Ces premiers résultats offrent donc une perspective supplémentaire sur les implications de l'intégration d'une concentration en programmation au sein des programmes éducatifs et contribuent à mieux comprendre les retombées pédagogiques d'apprendre à programmer, en termes de transfert vers les autres disciplines scolaires. D'un point de vue pédagogique, il apparaît crucial de réfléchir avec soin à la meilleure façon de réaliser cette intégration. Il semble notamment important de s'intéresser aux manières d'établir des relations interdisciplinaires plus explicites. Des recherches futures apparaissent néanmoins essentielles afin d'établir un pont plus solide avec la pratique.

Au terme de ce projet, nous croyons d'ailleurs que l'avenue de recherche future qui se dégage comme étant la plus prioritaire serait d'affiner l'identification des contenus disciplinaires susceptible de bénéficier le plus de l'apprentissage de la programmation. Cela permettrait de concevoir et de mettre à l'essai des interventions pédagogiques mobilisant la programmation qui pourraient avoir le potentiel de contribuer davantage à l'apprentissage et à la réussite scolaire des élèves. Dans cette perspective, nous sommes d'avis que ce projet pourra représenter une base solide pour les recherches futures qui exploreront cette avenue.

RÉFÉRENCES

Aguirre-Muñoz, Z., & Pantoya, M. L. (2016). Engineering literacy and engagement in kindergarten classrooms. *Journal of Engineering Education, 105*(4), 630-654.

Akter, S., Michael, K., Uddin, M. R., McCarthy, G., & Rahman, M. (2022). Transforming business using digital innovations : The application of AI, blockchain, cloud and data analytics. *Annals of Operations Research*, 1-33.

Alfayez, A. A., & Lambert, J. (2019). Exploring Saudi computer science teachers' conceptual mastery level of computational thinking skills. *Computers in the Schools, 36*(3), 143-166

Allaire-Duquette, G., Chastenay, P., Bouffard, T., Bélanger, S. A., Hernandez, O., Mahhou, M. A., ... & Desjarlais, E. (2022). Gender differences in self-efficacy for programming narrowed after a 2-h science museum workshop. *Canadian Journal of Science, Mathematics and Technology Education*, *22*(1), 87-100.

Andersen, P. N., Klausen, M. E., & Skogli, E. W. (2019). Art of learning–An art-based intervention aimed at improving children's executive functions. *Frontiers in Psychology*, *10*, 1769.

Anolli, L., Antonietti, A., Crisafulli, L. et Cantoia, M. (2001). Accessing source information in analogical problem-solving. *The Quarterly Journal of Experimental Psychology Section A, 54*(1), 237–261.

Ausubel, D.P. (1963). *The Psychology of Meaningful Verbal Learning : An Introduction to School Learning*. New York. Grune & Stratton.

Avsec, S., & Jamsek, J. (2016). Technological literacy for students aged 6-18 : A New method for holistic measuring of knowledge, capabilities, critical thinking and decision-making. *International Journal of Technology and Design Education*, *26*(1), 43.

Ayres, P. (2006). Impact of reducing intrinsic cognitive load on learning in a mathematical domain. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, *20*(3), 287-298.

Barma, S. (2018). *Rapport final : Réaliser une étude de cas multiple qui vise à affiner les connaissances sur l'usage pédagogique ou didactique de la programmation dans les écoles du Québec* (Rapport nº 118982). Québec, Québec : Université Laval

Barnett, S. M., & Ceci, S. J. (2002). When and where do we apply what we learn? A taxonomy for far transfer. *Psychological Bulletin*, *128*(4), 612.

Baron, G. L., Drot-Delange, B., Grandbastien, M., & Tort, F. (2014). Computer science education in French secondary schools : Historical and didactical perspectives. *ACM Transactions on Computing Education (TOCE)*, *14*(2), 1-27.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12 : What is involved and what is the role of the computer science education community? *ACM Inroads*, *2*(1), 48-54.

Bartolotti, J. (2015). *Previous language experience shapes novel language learning*. (Thèse de doctorat, Northwestern University).

Bauer, Daniel J., Daniel M. McNeish, Scott A. Baldwin, and Patrick J. Curran. 2020. Analyzing Nested Data : Multilevel Modeling and Alternative Approaches. In *The Cambridge Handbook of Research Methods in Clinical Psychology* (eds.) Aidan G. C. Wright and Michael N. Hallquist. New York, NY : Cambridge University Press, 426–443.

Bell, T. (2015). Surprising computer science. In *Informatics in Schools. Curricula, Competences, and Competitions : 8th International Conference on Informatics in Schools : Situation, Evolution, and Perspectives, ISSEP 2015, Ljubljana, Slovenia, September 28-October 1, 2015, Proceedings 8* (pp. 1-11). Springer International Publishing.

Bennett, D. A. (2001). How can I deal with missing data in my study? *Australian and New Zealand Journal of Public Health, 25*(5) :464–469.

Bentler, P. (1990). Comparative fit indexes in structural models. *Psychological Bulletin, 107*(2), 238–246.

Bergson-Shilcock, A., Taylor, R. & Hodge, N. (2023). Closing the digital skill divide. *National Skills Coalition*, *390*, 8619.

Bernardo, M. A., & Morris, J. D. (1994). Transfer effects of a high school computer programming course on mathematical modeling, procedural comprehension, and verbal problem solution. *Journal of Rresearch on Computing in Education*, *26*(4), 523-536.

Bers, M. U. (2018). Coding as a literacy for the 21st century. *Education Week*.

Bers, M. U. (2019). Coding as another language: A pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, *6*(4), 499-528.

Bers, M. U., Blake-West, J., Kapoor, M. G., Levinson, T., Relkin, E., Unahalekhaka, A., & Yang, Z. (2023). Coding as another language: Research-based curriculum for early childhood computer science. *Early Childhood Research Quarterly*, *64*, 394-404.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering : Exploration of an early childhood robotics curriculum. *Computers & Education*, *72*, 145-157.

Blikstein, P., & Moghadam, S. H. (2019). Computing education. In S. A. Fincher & A. V. Robins (Eds.), *The Cambridge handbook of computing education research* (pp. 56–78). Cambridge University Press.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing Computational Thinking in Compulsory Education-Implications for policy and practice* (Report No. JRC104188). Joint Research Centre, European Union.

Bollen, K. A. (2002). Latent variables in psychology and the social sciences. *Annual Review of Psychology, 53*(1), 605–634.

Boylan, M., Demack, S., Wolstenholme, C., Reidy, J., & Reaney, S. (2018). *ScratchMaths : evaluation report and executive summary.* Project Report. Education Endownment Foundation.

Bracke, D. (1998). Vers un modèle théorique du transfert : les contraintes à respecter. *Revue des sciences de l'éducation, 24*(2), 235-266.

Bracke, O. (2004). Un modèle fonctionnel du transfert pour l'éducation. In A. Presseau et M. Frenay, (Éds.), *Le transfert des apprentissages : comprendre pour mieux intervenir* (pp. 77-106). Sainte-Foy, Les Presses de l'Université Laval.

Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn* (Vol. 11). Washington, DC : National academy press.

Braver, T. S., Cole, M. W. et Yarkoni, T. (2010). Vive les différences ! Individual variation in neural mechanisms of executive control. Current Opinion in Neurobiology, 20(2), 242–250.Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).

Brion, C., & Cordeiro, P. A. (2018). Learning transfer : The missing link to learning among school leaders in Burkina Faso and Ghana. In *Frontiers in Education* (Vol. 2, pp. 69) : Frontiers.

Brown, A. L. (1992). Design experiments : Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences, 2*(2), 141-178.

Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart : The resurgence of computer science in UK schools. *ACM Transactions on Computing Education (TOCE)*, *14*(2), 1-22.

Brown, Q., Mongan, W., Kusic, D., Garbarine, E., Fromm, E., & Fontecchio, A. (2008). *Computer Aided Instruction As A Vehicle For Problem Solving : Scratch Boards In The Middle Years Classroom.* Paper presented at the Annual Conference & Exposition, Pittsburgh, Pennsylvania.

Browne, M. W., & Cudeck, R. (1992). Alternative Ways of Assessing Model Fit. *Sociological Methods & Research, 21*(2), 230–258

Bugmann, J., Karsenti, T. et Parent, S. (2018, mai). Développer l'apprentissage de la programmation par l'usage d'un jeu vidéo : intégration de Scratch à Minecraft. Dans. Colloque international en éducation, Montréal, Canada.

Butterfield, E. C., & Nelson, G. D. (1989). Theory and practice of teaching for transfer. *Educational Technology Research and Development, 37*(3), 5-38.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing Mathematical Thinking with Scratch. In G. Conole, T. Klobučar, C. Rensing, J. Konert, & É. Lavoué (Eds.), *Design for Teaching and Learning in a Networked World : 10th European Conference on Technology Enhanced Learning, EC-TEL 2015, Toledo, Spain, September 15-18, 2015, Proceedings* (pp. 17-27). Springer International Publishing.

Canada Learning Code. (2020). *Apprendre pour un monde numérique : un cadre de référence pancanadien pour l'enseignement de l'informatique*. Récupéré à l'adresse suivante : https ://k12csframework.ca/wpcontent/uploads/apprendre_pour_un_monde_numerique_final.pdf

Caron, P. O. (2019). *La modélisation par équations structurelles avec Mplus*. PUQ.

Cartwright, K. B. (2012). Insights from cognitive neuroscience : The importance of executive function for early reading development and education. *Early Education and Development, 23*(1), 24-36.

Claro, M. & Castro-Grau, C. (2023). *The role of digital technologies in 21st century learning*. IIEP-UNESCO Regional Forum on Education Policy. Récupéré en ligne à l'adresse suivante : https://unesdoc.unesco.org/ark:/48223/pf0000386981_eng

Clements, D. H. (1986). Effects of Logo and CAI environments on cognition and creativity. *Journal of Educational Psychology, 78*(4), 309-318.

Clements, D. H. (1995). Teaching creativity with computers. *Educational Psychology Review*, *7*, 141-161.

Clements, D. H. (1999). The future of educational computing research: The case of computer programming. *Information Technology in Childhood Education Annual*, *1999*(1), 147-179.

Clements, D. H., & Sarama, J. (2009). Learning trajectories in early mathematics–sequences of acquisition and teaching. *Encyclopedia of Language and Literacy Development, 7*, 1-6.

Code.org. (2017). *Recommendations for states developing computer science teacher pathways*. Retrieved from https ://code.org/files/TeacherPathwayRecommendations. pdf

Code.org. (2023). *Supporters*. Code.org. Retrieved from https ://code.org/about/supporters

CodeMTL. (2023). *Accueil*. CodeMTL. Récupéré à l'adresse suivante : https ://codemtl.org/

Coffman, D. L., & Millsap, R. E. (2006). Evaluating latent growth curve models using individual fit statistics. *Structural Equation Modeling, 13*(1), 1-27.

Cohen, J. (2013). *Statistical power analysis for the behavioral sciences*. Academic press.

Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2013). *Applied multiple regression/correlation analysis for the behavioral sciences*. Routledge.

Cormier, S. M., & Hagman, J. D. (2014). *Transfer of learning : Contemporary research and applications*. Academic press.

Corradini, I., Lodi, M., & Nardelli, E. (2017). Conceptions and Misconceptions about Computational Thinking among Italian Primary School Teachers. *ICER – Proceedings of ACM Conference. International Computing Education Research* (pp. 136–144).

Cox, B. D. (1997). The rediscovery of the active learner in adaptive contexts : A developmental-historical analysis. *Educational Psychologist*, *32*(1), 41-55.

Crick, T. (2017). Computing Education : An Overview of Research in the Field. *The Royal Society, London*.

Cyr, A.-A. (2012). *Évaluation de l'impact d'interventions en orthographe sur le transfert des apprentissages, dans différents contextes d'écriture, auprès d'élèves en difficulté du 2e cycle du primaire* (Mémoire de maîtrise, Université du Québec à Montréal).

Day, S. B., & Goldstone, R. L. (2012). The import of knowledge export : Connecting findings and theories of transfer of learning. *Educational Psychologist, 47*(3), 153–176.

De Bruyckere, P., Kirschner, P. A., & Hulshof, C. (2020). If You Learn A, Will You Be Better Able to Learn B? Understanding Transfer of Learning. *American Educator*, *44*(1), 30.

De Greeff, J. W., Bosker, R. J., Oosterlaan, J., Visscher, C., & Hartman, E. (2018). Effects of physical activity on executive functions, attention and academic performance in preadolescent children: a meta-analysis. *Journal of Science and Medicine in Sport*, *21*(5), 501-507.

Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, *60*(6), 33-39.

Departement for Education (2013). Computing programmes of study key stages 1 and 2 of the National Curriculum in England. Récupéré en ligne à l'adresse suivante : https ://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/239033/PRIMARY_national_curriculum_-_Computing.pdf

Desolneux, A., Moisan, L., & Morel, J. M. (2008). Gestalt theory. *From Gestalt Theory to Image Analysis : A Probabilistic Approach*, 11-30.

Dhlamini, J. J. (2016). Enhancing Learners' Problem Solving Performance in Mathematics: A Cognitive Load Perspective. *European Journal of STEM Education*, *1*(1), 27-36.

Diamond, A. (2013). Executive functions. *Annual Review of Psychology*, *64*, 135-168.

Diamond, A. (2020). Executive functions. In A. Gallagher, C. Bulteau, D. Cohen, and J. L. Michaud. (eds) *Handbook of Clinical Neurology* (pp. 225–240), Elsevier.

Dionne, E., & Fleuret, C. (2016). L'analyse de données secondaires dans le cadre d'évaluation de programme : regard théorique et expérientiel. *Canadian Journal of Program Evaluation*, *31*(2), 253-261.

DiSessa, A. A. (2000). *Changing minds : Computers, learning, and literacy*. MIT Press.

Dôrner, D. et Wearing, A. J. (1995). Complex problem solving : Toward a computer-simulated theory. In P. A. Frensch et J. Funke (dir.), *Complex problem solving : The European perspective* (p. 65-99). Hillsdale, NJ : Erlbaum.

Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, *2*(1), 57-73.

Duncan, T. E., & Duncan, S. C. (2009). The ABC's of LGM : An introductory guide to latent variable growth curve modeling. *Social and Personality Psychology Compass, 3*(6), 979-991.

Ellis, H. C. (1965). *The transfer of learning*. New York : Macmillan.

Endy, D. (2005). Foundations for engineering biology. *Nature, 438*(7067), 449-453.

Ericson, B., Adrion, W. R., Fall, R., & Guzdial, M. (2016). State-based progress towards computer science for all. *ACM Inroads*, *7*(4), 57-60.

Esser, M. (2004). La programmation neuro-linguistique en débat. *La programmation neurolinguistique en débat*, 1-289.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, *32*(6), 576-593.

Fan, X. (2003). Power of latent growth modeling for detecting group differences in linear growth trajectory parameters. *Structural Equation Modeling, 10*, 380-400.

Ferguson, R. W. (2019, May). MAGI: Analogy-based encoding using regularity and symmetry. In *Proceedings of the sixteenth annual conference of the Cognitive Science Society* (pp. 283-288). Routledge.

Ferrada Ferrada, C., Carrillo-Rosúa, J., Díaz-Levicoy, D., & Silva Díaz, F. (2020). Robotics from stem areas in primary school : A systematic review. *Education in the Knowledge Society,* 22.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment : A case study. *Computers & Education*, *63*, 87-97.

Field, A. (2013). *Discovering statistics using IBM SPSS statistics*. sage.

Fields, D., Vasudevan, V., & Kafai, Y. B. (2015). The programmers' collective : fostering participatory culture by making music videos in a high school Scratch coding workshop. *Interactive Learning Environments*, *23*(5), 613-633.

Fincher, S. A., Tenenberg, J., Dorn, B., Hundhausen, C., McCartney, R., & Murphy, L. (2019). Computing education research today. *The Cambridge handbook of computing education research*, *1*, 40-55.

Flannery, L. P., Silverman, B., Kazakoff, E. R., Bers, M. U., Bontá, P., & Resnick, M. (2013, June). Designing ScratchJr : Support for early childhood learning through computer programming. In *Proceedings of the 12th international conference on interaction design and children* (pp. 1-10).

Fortin, M. F., & Gagnon, J. (2016). Fondements et étapes du processus de recherche, 3e édition. Édition Chenelière Éducation.

Franklin, D., Conrad, P., Boe, B., Nilsen, K., Hill, C., Len, M., ... & Waite, R. (2013, March). Assessment of computer science learning in a scratch-based outreach program. In *Proceeding of the 44th ACM technical symposium on Computer science education* (pp. 371-376).

Franks, J. J., Bilbrey, C. W., Lien, K. G. et McNamara, T. P. (2000). Transfer appropriate processing (TAP). *Memory and Cognition, 28*(7), 1140–1151.

Fritz, W. B. (1963). Selected definitions*. Communications of the ACM, 6*(4), 152-158*.*

Gaffield, C. (2020). COVID-19 is the first global pandemic in the digital age. *Royal Society of Canada*. Retrieved from: https ://rsc-src.ca/sites/default/files/Publication%20-%20COVID-19%20in%20the%20Digital%20Age_Apr17_0.pdf

Gagne, R. M. (1962). The acquisition of knowledge. *Psychological Review*, *69*(4), 355.

Gagne, R. M. (1965). The analysis of instructional objectives for the design of instruction. *Teaching machines and programmed learning II : Data and directions*, 21-65.

Gal-Ezer, J., & Stephenson, C. (2014). A tale of two countries : Successes and challenges in K-12 computer science education in Israel and the United States. *ACM Transactions on Computing Education (TOCE)*, *14*(2), 1-18.

Gallup and Northeastern University. (2019). Facing the future : U.S., U.K. and Canadian citizens call for a unified skills strategy for the AI age.

Garcia-Penalvo, F. J. (2016, September). A brief introduction to TACCLE 3—coding European project. In *2016 International Symposium on Computers in Education (SIIE)* (pp. 1-4). IEEE.

Gaudreau, L. (2011). *Guide pratique pour créer et évaluer une recherche scientifique en éducation*. Guérin,.

Gentner, D. (2002). Analogy in scientific discovery : The case of Johannes Kepler. In *Model-based reasoning : Science, technology, values* (pp. 21-39). Boston, MA : Springer US.

Gentner, D., Loewenstein, J. et Tompson, L. (2003). Learning and transfer: a general role for analogical encoding. *Journal of Educational Psychology, 95*(2), 393-408.

Gentner, D., Rattermann, M. J. et Forbus, K. D. (1993). The roles of similarity in transfer: Separating retrievability from inferential soundness. *Cognitive Psychology, 25*(4), 524-575.

Gentner, D., & Stevens, A. L. (2014). *Mental models*. Psychology Press.

Gick, M. L., & Holyoak, K. J. (1980). Analogical problem solving. *Cognitive Psychology, 12*(3), 306-355.

Gilster, P., & Glister, P. (1997). *Digital literacy* (p. 1). New York : Wiley Computer Pub.

Girard, S., & Béland, S. (2017). Analyser l'interaction de variables latentes: une exemplification méthodologique de la méthode d'équations structurelles avec interaction latente. *Revue des sciences de l'éducation, 43*(3), 28-60.

Giroux, P., Girard, J.-F. et M. Gagnon (2018). La robotique pour motiver ses élèves. Revue hybride de l'Éducation, 2(2), 68-77.

Google & Gallup (2016). Trends in the state of computer science in US K-12 schools. Retrieved from: https ://services.google.com/fh/files/misc/trends-in-the-state-ofcomputer-science-report.pdf

Google & Gallup. (2015). Images of computer science : Perceptions among students, parents and educators in the US. Retrieved from: https ://services.google.com/fh/files/misc/images-of-computer-science-report.pdf

Gouvernement de l'Ontario. (2020). *Nouveau programme-cadre de mathématiques de la 1re à la 8e année*. Récupéré à l'adresse suivante : https ://www.ontario.ca/fr/page/nouveau- programme-cadre-de-mathematiques-de-la-1re-la-8e-annee

Gouvernement de la Colombie-Britannique. (2016). *Introduction au programme Conception, compétences pratiques et technologies*. Récupéré en ligne à l'adresse suivante : https ://curriculum.gov.bc.ca/fr/curriculum/adst/introduction

Gouvernement du Québec (2007). Programme de formation de l'école québécoise : Science et technologie, 2e cycle. Récupéré en ligne à l'adresse suivante : https://www.education.gouv.qc.ca/fileadmin/site_web/documents/education/jeunes/pfeq/PFEQ_science-technologie-deuxieme-cycle-secondaire.pdf

Gouvernement du Québec (2011). *Cadre d'évaluation des apprentissages : Mathématiques, Enseignement secondaire 1er et 2e cycles*. Récupéré en ligne à l'adresse suivante : https://www.education.gouv.qc.ca/fileadmin/site_web/documents/education/jeunes/pfeq/CE_PFEQ_mathematique-secondaire_2011.pdf

Gouvernement du Québec (2011). Progression des apprentissages au secondaire : Science et technologie 1er et 2e cycles. Récupéré en ligne à l'adresse suivante : https://www.education.gouv.qc.ca/fileadmin/site_web/documents/education/jeunes/pfeq/PDA_PFEQ_science-technologie-secondaire_2011.pdf

Gouvernement du Québec. (2018). *Plan d'action numérique en éducation et en enseignement supérieur*. Récupéré à : https://cdn-contenu.quebec.ca/cdn-contenu/adm/min/education/publications-adm/enseignement-superieur/Plan-action-numerique/PAN_Plan_action_VF.pdf

Gouvernement du Québec (2019). *Cadre de référence de la compétence numérique*. Récupéré à : https://www.education.gouv.qc.ca/fileadmin/site_web/documents/ministere/Cadre-reference-competence-num.pdf

Greiff, S., Wüstenberg, S., Csapó, B., Demetriou, A., Hautamäki, J., Graesser, A. C., & Martin, R. (2014). Domain-general problem solving skills and education in the 21st century. *Educational Research Review*, (13), 74-83.

Grover, S. (2017). Assessing algorithmic and computational thinking in K-12 : Lessons from a middle school classroom. *Emerging research, practice, and policy on computational thinking*, 269-288.

Grover, S., & Basu, S. (2017, March). Measuring student learning in introductory block-based programming : Examining misconceptions of loops, variables, and boolean logic. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education* (pp. 267-272).

Grover, S., & Pea, R. (2013). Computational thinking in K–12 : A review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, *25*(2), 199–237.

Gülbahar, Y., & Kalelioğlu, F. (2017, November). Competencies of high school teachers and training needs for computer science education. In *Proceedings of the 6th Computer Science Education Research Conference* (pp. 26-31).

Guzdial, M. (2015). *Learner-centered design of computing education : Research on computing for everyone*. Morgan & Claypool Publishers.

Guzdial, M. (2019). Computing for other disciplines. In S. A. Fincher & A. V. Robins (Eds.), *The Cambridge handbook of computing education research* (pp. 584–615). Cambridge University Press.

Guzdial, M., & Du Boulay, B. (2019). The history of computing education research. In S. A. Fincher & A. V. Robins (Eds.), *The Cambridge handbook of computing education research* (pp. 11–39). Cambridge University Press.

Hagman, J. D., & Cormier, S. M. (Eds.). (1987). *Transfer of learning : Contemporary research and applications*. Academic Press.

Haskell, R. E. (2001). *Transfer of learning : Cognition, instruction and reasoning*. San Diego, Academic press.

Hassenfeld, Z. R., & Bers, M. U. (2020). Debugging the writing process : Lessons from a comparison of students' coding and writing practices. *The Reading Teacher*, *73*(6), 735-746.

Hattie, J. (2008). *Visible learning : A synthesis of over 800 meta-analyses relating to achievement*. Routledge.

Hattie, J. A., & Donoghue, G. M. (2016). Learning strategies : A synthesis and conceptual model. *npj Science of Learning*, *1*(1), 1-13.

Hayes, J., & Stewart, I. (2016). Comparing the effects of derived relational training and computer coding on intellectual potential in school-age children. *British Journal of Educational Psychology*, *86*(3), 397-411.

Heintz, F., & Mannila, L. (2018). Computational thinking for all: an experience report on scaling up teaching computational thinking to all students in a major city in Sweden. *ACM Inroads, 9*(2), 65-71.

Henri, F. (2014). *Les bases de la programmation*. JFD éditions.

Hermans, F., & Aivaloglou, E. (2017, November). To scratch or not to scratch? A controlled experiment comparing plugged first and unplugged first programming lessons. In *Proceedings of the 12th workshop on primary and secondary computing education* (pp. 49-56).

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education*, *4*, 48-69.

Hoc, J.M., Green, T.R.G., Samurcay, R., & Gillmore, D.J. (1990). *Psychology of programming*. London : Academic Press.

Horn, M., & Bers, M. (2019). Tangible computing. *The Cambridge handbook of computing education research*, *1*, 663-678.

Houdé, O., & Borst, G. (2014). Measuring inhibitory control in children and adults: brain imaging and mental chronometry. *Frontiers in Psychology, 5*, 616.

Hoyle, R. H. (1995). *Structural equation modeling*. Thousand Oaks, CA : Sage

Hunt, V., Yee, L., Prince, S., & Dixon-Fyle, S. (2018). *Delivering through Diversity. McKinsey & Company Report.* Retrieved from : www.mckinsey.com/businessfunctions/organization/our-insights/delivering-through-diversity

International Society of Technology in Education (ISTE). (2015). CT Leadership toolkit. Retrieved from: http ://www.iste.org/docs/ctdocuments/ct-leadershipttoolkit.pdf?sfvrsn=4

Ivanova, A. A., Srikant, S., Sueoka, Y., Kean, H. H., Dhamala, R., O'reilly, U. M., ... & Fedorenko, E. (2020). Comprehension of computer code relies primarily on domain-general executive brain regions. *elife*, *9*, e58906.

Jenkins, C. (2015). Poem Generator : A Comparative Quantitative Evaluation of a Microworlds-Based Learning Approach for Teaching English. *International Journal of Education and Development using Information and Communication Technology, 11*(2), 153-167.

Johnson-Laird, P. N. (1983). *Mental models : Towards a cognitive science of language, inference, and consciousness* (N°. 6). Harvard University Press.

Juškevičienė, A., & DagienĖ, V. (2018). Computational thinking relationship with digital competence. *Informatics in Education*, *17*(2), 265-284.

K-12 Computer Science Framework Steering Committee. (2016). K-12 computer science framework. ACM. Retrieved from: https ://k12cs.org/wp-content/ uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students : Code. org. *Computers in Human Behavior*, *52*, 200-210.

Kalelioglu, F., & Gülbahar, Y. (2014). The Effects of Teaching Programming via Scratch on Problem Solving Skills : A Discussion from Learners' Perspective. *Informatics in education, 13*(1), 33-50.

Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing, 4*(3), 583.

Karsenti, T. et Savoie-Zajc, L. (2018). *La recherche en éducation : Étapes et approches. 4e édition revue et mise à jour*. Les Presses de l'Université de Montréal.

Kay, K., & Greenhill, V. (2010). Twenty-first century students need 21st century skills. In *Bringing schools into the 21st century* (pp. 41-65). Dordrecht : Springer Netherlands.

Kazakoff, E. R., Sullivan, A., & Bers, M. U. (2013). The effect of a classroom-based intensive robotics and programming workshop on sequencing ability in early childhood. *Early Childhood Education Journal, 41*(4), 245-255.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games : A case study on mathematics learning during design and computing. *Computers & education*, *73*, 26-39.

Khine, M. S. (2018). Strategies for developing computational thinking. *Computational Thinking in the STEM Disciplines : Foundations and Research Highlights*, 3-9.

Kline, R. B. (2005). *Principles and practice of structural equation modeling* (2nd ed.). New York, NY : Guilford.

Knobelsdorf, M., Magenheim, J., Brinda, T., Engbring, D., Humbert, L., Pasternak, A., ... & Vahrenhold, J. (2015). Computer science education in North-Rhine Westphalia, Germany—a case study. *ACM Transactions on Computing Education (TOCE)*, *15*(2), 1-22.

Knuth, D. E. (1992). Literate Programming. Number 27 in CSLI Lecture Notes. *Center for the Study of Language and Information*, 349-358.

Korkmaz, Ö. (2016). The Effect of Scratch-Based Game Activities on Students' Attitudes, Self-Efficacy and Academic Achievement. *Modern Computing, 8*(1), 16-23.

Krauss, J., & Prottsman, K. (2016). *Computational thinking and coding for every student : The teacher's getting-started guide*. Corwin Press.

Kvalø, S. E., Bru, E., Brønnick, K., & Dyrstad, S. M. (2017). Does increased physical activity in school affect children's executive function and aerobic fitness?. *Scandinavian Journal of Medicine & Science in Sports*, *27*(12), 1833-1841.

Ladner, R. E., & Israel, M. (2016). For all" in" computer science for all. *Communications of the ACM*, *59*(9), 26-28.

Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Acm Sigcse Bulletin*, *37*(3), 14-18.

Langkamp, D. L., Lehman, A., & Lemeshow, S. (2010). Techniques for handling missing data in secondary analyses of large surveys. *Academic Pediatrics*, *10*(3), 205-210.

Lapierre, H. G., Charland, P., & Léger, P. M. (2023). Looking "Under the hood" of learning computer programming : the emotional and cognitive differences between novices and beginners. *Computer Science Education*, 1-22.

Larousse. (2023). *Programmation*. Larousse. Récupéré en ligne à l'adresse suivante : https://www.larousse.fr/dictionnaires/francais/programmation/64205

Larousse. (2023). *Programmer*. Larousse. Récupéré en ligne à l'adresse suivante : https ://www.larousse.fr/dictionnaires/francais/programmer/64209

Laurent, M., Crisci, R., Bressoux, P., Chaachoua, H., Nurra, C., de Vries, E., & Tchounikine, P. (2022). Impact of programming on primary mathematics learning. *Learning and Instruction*, *82*, 101667.

Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2020). Computational thinking from a disciplinary perspective : Integrating computational thinking in K-12 science, technology, engineering, and mathematics education. *Journal of Science Education and Technology*, *29*, 1-8.

Legendre, R. (2005). Transfert d'apprentissage. Dans R. Legendre (dir.), *Dictionnaire actuel de l'éducation* (3e éd., p. 1402). Guérin.

Lehrer, R., & Randle, L. (1987). Problem Solving, Metacognition and Composition : The Effects of Interactive Software for First-Grade Children. *Journal of Educational Computing Research, 3*(4), 409-427.

Lewis, C. M., & Shah, N. (2012, February). Building upon and enriching grade four mathematics standards with programming curriculum. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 57-62).

Lewis, C. M., Shah, N., & Falkner, K. (2019). Equity and Diversity. In S. A. Fincher & A. V. Robins (Eds.), *The Cambridge handbook of computing education research* (pp. 478-506). Cambridge University Press.

Li, J. W., O'Connor, H., O'Dwyer, N., and Orr, R. (2017). The effect of acute and chronic exercise on cognitive function and academic performance in adolescents: a systematic review. *J. Sci. Med. Sport, 20*, 841–848.

Li, Y., Wang, K., Xiao, Y., Froyd, J. E., & Nite, S. B. (2020). Research and trends in STEM education : A systematic analysis of publicly funded projects. *International Journal of STEM Education, 7*, 1-17.

Liao, Y. K. (2000). A meta-analysis of computer programming on cognitive outcomes : An updated synthesis. In *EdMedia+ Innovate Learning* (pp. 598-604). Association for the Advancement of Computing in Education (AACE).

Liao, Y. K. C., & Bright, G. W. (1991). Effects of computer programming on cognitive outcomes : A meta-analysis. *Journal of EducationalCcomputing Research, 7*(3), 251-268.

Liu, J., Lin, C. H., Hasson, E. P., & Barnett, Z. D. (2011, March). Introducing computer science to K-12 through a summer computing workshop for teachers. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 389-394).

Liu, Y., & Pásztor, A. (2022). Effects of problem-based learning instructional intervention on critical thinking in higher education : A meta-analysis. *Thinking Skills and Creativity*, *45*, 101069.

Lightbown, P. M. (2008). Transfer appropriate processing as a model for classroom second language acquisition. Dans Z. Han (dir.), *Understanding second language process* (p. 27–44). Multilingual Matters.

Lindh, J., & Holgersson, T. (2007). Does lego training stimulate pupils' ability to solve logical problems?.*Computers & education*, *49*(4), 1097-1111.

Lobato, J. (2006). Alternative perspectives on the transfer of learning : History, issues, and challenges for future research. *The journal of the learning sciences*, *15*(4), 431-449.

Lodi, M. (2020). Informatical thinking. *Olympiads in Informatics : An International Journal*, *14*, 113-132.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming : What is next for K-12?. *Computers in Human Behavior*, *41*, 51-61.

Maguth, B. M. (2012). In defense of the social studies: Social studies programs in STEM education. *Social Studies Research and Practice, 7*(2), 65-90.

Malcuit, G., A. Pomerleau et P. Maurice. 1995. *Psychologie de l'apprentissage : Termes et concepts*. Montréal, EDISEM, 243 p.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 1-15.

Manches, A., & Plowman, L. (2017). Computing education in children's early years : A call for debate. *British Journal of Educational Technology*, *48*(1), 191-201.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014, June). Computational thinking in K-9 education. In *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conferenc*e (pp. 1-29).

Margulieux, L. E., Guzdial, M., & Catrambone, R. (2012, September). Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In *Proceedings of the ninth annual international conference on International computing education research* (pp. 71-78).

Martin, C., Hughes, J., & Richards, J. (2018). Designing engaging learning experiences in programming. In *Computers Supported Education : 9th International Conference, CSEDU 2017, Porto, Portugal, April 21-23, 2017, Revised Selected Papers 9* (pp. 221-245). Springer International Publishing.

Mayer, R. E. (1975). Information processing variables in learning to solve problems. *Review of Educational Research*, *45*(4), 525-541.

Mayer, R. E. (1989). Models for understanding. *Review of educational research*, *59*(1), 43-64.

McArdle, J. J., & Prindle, J. J. (2008). A latent change score analysis of a randomized clinical trial in reasoning training. *Psychology and aging*, *23*(4), 702.

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with scratch. *Computer Science Education*, *23*(3), 239-264.

Melby-Lervåg, M., Hagen, Å. M., & Lervåg, A. (2020). Disentangling the far transfer of language comprehension gains using latent mediation models. *Developmental Science*, *23*(4), e12929.

Melby-Lervåg, M., Redick, T. S., & Hulme, C. (2016). Working memory training does not improve performance on measures of intelligence or other measures of "far transfer" : Evidence from a meta-analytic review. *Perspectives on Psychological Science, 11*, 512–534.

Merriam-Webster. (n.d.). Programming. In Merriam-Webster.com dictionary. Récupéré en ligne à l'adresse suivante : https ://www.merriam-webster.com/dictionary/programming

Merta Dhewa, K., Rosidin, U., Abdurrahman, A., & Suyatna, A. (2017). The development of Higher Order Thinking Skill (Hots) instrument assessment in physics study. *IOSR Journal of Research & Method in Education*, *7*(1), 26-32.

Meyer, J. (2022). *Programming 101 : Learn to Code with the Processing Language Using a Visual Approach (2nd edition)*. Apress.

Miller, G. (1956). Human memory and the storage of information. *IRE Transactions on Information Theory*, *2*(3), 129-137.

Miller, R. B., Kelly, G. N., & Kelly, J. T. (1988). Effects of Logo computer programming experience on problem solving and spatial relations ability. *Contemporary Educational Psychology, 13*(4), 348-357.

Mills, K., Coenraad, M., Ruiz, P., Burke, Q., & Weisgrau, J. (2021). *Computational thinking for an inclusive world : a resource for educators to learn and lead*. Digital Promise.

Mongrain, P., & Besançon, J. (1995). Étude du transfert des apprentissages pour les programmes de formation professionnelle. *Revue des sciences de l'éducation*, *21*(2), 263-288.

Moreno-León, J., Robles, G., & Román-González, M. (2016, April). Comparing computational thinking development assessment scores with software complexity metrics. In *2016 IEEE global engineering education conference (EDUCON)* (pp. 1040-1045). IEEE.

Morse, W. H., & Skinner, B. F. (1958). Some factors involved in the stimulus control of operant behavior. *Journal of the Experimental Analysis of Behavior*, *1*(1), 103.

Moss, T. E., Benus, M. J., & Tucker, E. A. (2018). Impacting urban students' academic achievement and executive function through school-based arts integration programs. *SAGE Open*, *8*(2), 2158244018773131.

Muthén, B. (1997). Latent variable modeling of longitudinal and multilevel data. In A. Raftery (Ed.), *Sociological methodology* (pp. 453-480). Boston : Blackwell.

Muthén, B. (2004). Latent variable analysis. *The Sage handbook of quantitative methodology for the social sciences, 345*(368), 106.

Muthén, B. & Muthén, L. (2009). *Statistical analysis with latent variables* (Vol. 123, No. 6). New York : Wiley.

Muthén, B. & Muthén, L. (2017). Mplus. In *Handbook of item response theory* (pp. 507-518). Chapman and Hall/CRC.

Nafzaoui, M. A., et Ferdoussi, S. (2021). La programmation budgétaire en période de Pandémie COVID-19. *Revue Internationale des Sciences de Gestion*, *4*(2).

National Research Council. (2012). *Education for life and work : Developing transferable knowledge and skills in the 21st century*. National Academies Press.

Nelson, J. (2009). Celebrating Scratch in libraries : creation software helps young people develop 21st-century literacy skills. *School Library Journal*, 20–21.

Noack, H., Lövdén, M., & Schmiedek, F. (2014). On the validity and generality of transfer effects in cognitive training research. *Psychological Research, 78*, 773-789.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings : Learning cultures and computers* (Vol. 17). Springer Science & Business Media.

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, *11*(1), 1-17.

Nugent, G., Barker, B., Grandgenett, N., & Adamchuk, V. I. (2010). Impact of Robotics and Geospatial Technology Interventions on Youth STEM Learning and Attitudes. *Journal of Research on Technology in Education, 42*(4), 391-408.

OECD (2023). *Is Education Losing the Race with Technology?: AI's Progress in Maths and Reading*, Educational Research and Innovation, OECD Publishing, Paris.

Oei, A. C., & Patterson, M. D. (2015). Enhancing perceptual and attentional skills requires common demands between the action video games and transfer tasks. *Frontiers in PsychoOQlogy*, *6*, 113.

Office québécoise de la langue française (2023). *Transfert des apprentissages*. Grand dictionnaire terminologique. Récupéré en ligne à l'adresse suivante : https ://www.oqlf.gouv.qc.ca/ressources/bibliotheque/dictionnaires/terminologie_relations_professionnelles/transfert_des_apprentissages.html# :~ :text=transfert%20des%20 connaissances% 20 n.%20 m.&text=D%C3 %A9 finition%20 %3 A,une%20autre%20classe%20de%20situations.

Ornelas Marques, F., Marques, M.T. (2012). No problem? No research, little learning... big problem!. *Systemics, Cybernetics and Informatics*, 10(3), 60–62.

Ortiz, A. M. (2015). Examining students' proportional reasoning strategy levels as evidence of the impact of an integrated LEGO robotics and mathematics learning experience. *Journal of Technology Education, 26*(2), 46-69. doi :10.21061/jte.v26i2.a.3

Owston, R., Wideman, H., Ronda, N. S., & Brown, C. (2009). Computer game development as a literacy activity. *Computers & Education, 53*(3), 977-989.

Page, L. C., Lenard, M. A., & Keele, L. (2020). The design of clustered observational studies in education. *AERA Open, 6*(3), 2332858420954401.

Palumbo, D. B., & Reed, W. M. (1991). The effect of BASIC programming language instruction on high school students' problem solving ability and computer anxiety. *Journal of Research on Computing in Education*, *23*(3), 343-372.

Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2017). Improving mathematics teaching in kindergarten with realistic mathematical education. *Early Childhood Education Journal*, *45*, 369-378.

Papert, S. (1980). *Mindstorms : Children, Computers, and Powerful Ideas*. New York : Basic Books.

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, *36*(2), 1–11.

Park, J. (2015). Effect of Robotics enhanced inquiry based learning in elementary Science education in South Korea. *Journal of Computers in Mathematics and Science Teaching*, *34*(1), 71-95.

Park, S., Lee, J. M., Baik, Y., Kim, K., Yun, H. J., Kwon, H., ... & Kim, B. N. (2015). A preliminary study of the effects of an arts education program on executive function, behavior, and brain structure in a sample of nonclinical school-aged children. *Journal of Child Neurology*, *30*(13), 1757-1766.

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, *2*(2), 137-168.

Péladeau, N., Forget, J., & Gagné, F. (2005). Le transfert des apprentissages et la réforme de l'éducation au Québec : quelques mises au point. *Revue des sciences de l'éducation*, *31*(1), 187-209.

Peng, C. Y. J., Harwell, M., Liou, S. M., & Ehman, L. H. (2006). Advances in missing data methods and implications for educational research. *Real Data Analysis*, *3178*, 102.

Perez, C. F., Such, F. P., & Karaletsos, T. (2018). Efficient transfer learning and online adaptation with latent variable models for continuous control. *arXiv 1812.03399*.

Perkins, D. N. et Salomon, G. (1992). Transfer of learning. Dans T. Husén et T. N. Postlethwaite (dir.), International encyclopedia of education (2e éd., vol. 11, p. 6452-6464). Pergamon press.

Perkins, D. N., & Martin, F. (1986, June). Fragile knowledge and neglected strategies in novice programmers. In *Papers presented at the first workshop on empirical studies of programmers on Empirical studies of programmers* (pp. 213-229).

Perkins, D. N., Schwartz, S., & Simmons, R. (2013). Instructional strategies for the problems of novice programmers. In *Teaching and learning computer programming* (pp. 153-178). Routledge.

Peters, S., Clarebout, G., Van Nuland, M., Aertgeerts, B., & Roex, A. (2018). A qualitative exploration of multiple perspectives on transfer of learning between classroom and clinical workplace. *Teaching and learning in medicine*, *30*(1), 22-32.

Popat, S., & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education*, *128*, 365-376.

Presseau, A. (2000). Analyse de l'efficacité d'interventions sur le transfert des apprentissages en mathématiques. *Revue des sciences de l'éducation, 26*(3), 515–544.

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, *45*(5), 583-602.

Purpura, D. J., Schmitt, S. A., & Ganley, C. M. (2017). Foundations of mathematics and literacy: The role of executive functioning components. *Journal of Experimental Child Psychology*, *153*, 15-34.

Qian, Y., & Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming : A literature review. *ACM Transactions on Computing Education (TOCE)*, *18*(1), 1-24.

Raman, R., Venkatasubramanian, S., Achuthan, K., & Nedungadi, P. (2015). Computer science (CS) education in Indian schools : Situation analysis using Darmstadt model. *ACM Transactions on Computing Education (TOCE)*, *15*(2), 1-36

Randell, B. (1994). The origins of computer programming. *IEEE Annals of the History of Computing, 16*(4), 6-14.

Rankin, Y., Gooch, A., & Gooch, B. (2008, February). The impact of game design on students' interest in CS. In *Proceedings of the 3rd international conference on Game development in computer science education* (pp. 31-35).

Rausch, L. M. (1998). High-tech industries drive global economic activity. *Issue Brief. Washington, DC, Division of Science Resources Studies, National Science Foundation*.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch : programming for all. *Communications of the ACM*, *52*(11), 60-67.

Rich, P. J., Bartholomew, S., Daniel, D., Dinsmoor, K., Nielsen, M., Reynolds, C., ... & Yauney, J. (2022). Trends in tools used to teach computational thinking through elementary coding. *Journal of Research on Technology in Education*, 1-22.

Richard, M. et Bissonnette, S. (2012). Les sciences cognitives et l'enseignement. Dans Gauthier, C. et Tardif, M. (dir.), *La pédagogie. Théories et pratiques de l'Antiquité à nos jours (3e édition)* (p. 237-255). Montréal : Gaëtan Morin éditeur.

Roberts, S., Glennon, M. O., Weissman, H., Fletcher, C., Dunton, S., Baskin, J., & Mak, J. (2022). 2022 State of Computer Science Education : Understanding Our National Imperative. Retrieved from: https ://advocacy.code.org/stateofcs

Robins, A. (2010). Learning edge momentum : A new account of outcomes in CS1. *Computer Science Education*, *20*(1), 37-71.

Robins, A. V., Margulieux, L. E., & Morrison, B. B. (2019). Cognitive sciences for computing education. In S. A. Fincher & A. V. Robins (Eds.), *The Cambridge handbook of computing education research* (pp. 231–275). Cambridge University Press.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming : A review and discussion. *Computer Science Education*, *13*(2), 137-172.

Rolandsson, L., & Skogh, I. B. Programming in School : Look Back to Move Forward. ACM Transactions on *Computer Science Education, 14*(2)(12), 2014.

Román-González, M. (2015). Computational thinking test: Design guidelines and content validation. In *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies* (EDULEARN 2015) (pp. 2436–2444).

Román-González, M., Moreno-León, J., & Robles, G. (2017). Complementary tools for computational thinking assessment. In S. C. Kong, J. Sheldon, & K. Y. Li (Eds.), *Proceedings of International Conference on Computational Thinking Education* (CTE 2017) (pp. 154–159).

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in HumanBbehavior*, *72*, 678-691.

Rose, S. P., Habgood, M. J., & Jay, T. (2019, May). *Using Pirate Plunder to develop children's abstraction skills in Scratch*. In Extended abstracts of the 2019 CHI conference on human factors in computing systems (pp. 1-6).

Rossignoli-Palomeque, T., Perez-Hernandez, E., & Gonzalez-Marques, J. (2018). Brain training in children and adolescents: Is it scientifically valid?. *Frontiers in Psychology*, 565.

Royal Society. (2017). *After the reboot : Computing education in UK schools*. Retrieved from : https ://royalsociety.org/~/media/policy/projects/computing-education/ computing-education-report.pdf

Royer, J. M. (1979). Theories of the transfer of learning. *Educational psychologist*, *14*(1), 53-69.

Rubin, D. B. (2007). The design versus the analysis of observational studies for causal effects : parallels with the design of randomized trials. *Statistics in Medicine*, *26*(1), 20-36.

Rushkoff, D. (2010). *Program or be programmed : Ten commands for a digital age*. Or Books.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school : A two year case study using “Scratch” in five schools. *Computers & Education*, *97*, 129-141.

Sahami, M. (2018). Paving a path to more inclusive computing. *ACM Inroads*, *9*(4), 85-88.

Sala G., Gobet F. (2016). Do the benefits of chess instruction transfer to academic and cognitive skills? A meta-analysis. *Educational Research Review*, 18, 46–57.

Sala, G., Gobet, F. (2017a). Working memory training in typically developing children : A meta-analysis of the available evidence. *Developmental Psychology, 53*(4), 671.

Sala G., Gobet F. (2017b). When the music’s over. Does music skill transfer to children’s and young adolescents’ cognitive and academic skills? A meta-analysis. *Educational Research Review*, 20, 55–67.

Sala G., Gobet F. (2017c). Working memory training in typically developing children: A meta-analysis of the available evidence. *Developmental Psychology*, 53, 671–685.

Sala, G., Gobet, F. (2020a). Cognitive and academic benefits of music training with children: A multilevel meta-analysis. *Memory & cognition*, *48*(8), 1429-1441.

Sala, G., Gobet, F. (2020b). Working memory training in typically developing children: A multilevel meta-analysis. *Psychonomic Bulletin & Review*, *27*(3), 423-434.

Salas-Gomez, D., Fernandez-Gorgojo, M., Pozueta, A., Diaz-Ceballos, I., Lamarain, M., Perez, C., ... & Sanchez-Juan, P. (2020). Physical activity is associated with better executive function in university students. *Frontiers in Human Neuroscience*, *14*, 11.

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming : A meta-analysis of transfer effects. *Journal of Educational Psychology*, *111*(5), 764.

Schlomer, G. L., Bauman, S., & Card, N. A. (2010). Best practices for missing data management in counseling psychology. *Journal of Counseling Psychology*, *57*(1), 1.

Schulte, C., Hornung, M., Sentance, S., Dagiene, V., Jevsikova, T., Thota, N., ... & Peters, A. K. (2012, November). Computer science at school/CS teacher education : Koli working-group report on CS at school. In *Proceedings of the 12th Koli Calling international conference on computing education research* (pp. 29-38).

Seidman, R. H. (1981). *The Effects of Learning a Computer Programming Language on the Logical Reasoning of School Children.* Paper presented at the Annual Meeting of the American Educational Research Association, Los Angeles, CA.

Selby, C.C. & J. Woollard. (2013). Computational thinking : The developing definition. *Paper presented at the 18th annual conference on innovation and technology in computer science education*, Canterbury.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation : A theoretical framework. *Education and Information Technologies*, *18*, 351-380.

Shi, D., DiStefano, C., Zheng, X., Liu, R., & Jiang, Z. (2021). Fitting latent growth models with small sample sizes and non-normal missing data. *International Journal of Behavioral Development*, *45*(2), 179-192.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142-158.

Simon, J. R. (1990). The effects of an irrelevant directional cue on human information processing. In *Advances in psychology* (Vol. 65, pp. 31-86). North-Holland.

Simons, D. J., Boot, W. R., Charness, N., Gathercole, S. E., Chabris, C. F., Hambrick, D. Z., & Stine-Morrow, E. A. (2016). Do "brain-training" programs work?. *Psychological Science in the Public Interest*, *17*(3), 103-186.

Skolverket. (2017). *Få syn på digitaliseringen på grundskolenivå – Ett kommentarmaterial till läro-planerna för förskoleklass, fritidshem och grundskoleutbildning*. Stockholm : Skolverket.

Soloway, E., & Spohrer, J.C. (1989). *Studying the novice programmer*. Hillsdale, NJ : Lawrence Erlbaum.

Soper, D.S. (2023). *A-priori Sample Size Calculator for Structural Equation Models* [Software]. Retrieved from: https ://www.danielsoper.com/statcalc

Spolaôr, N., & Benitti, F. B. V. (2017). Robotics applications grounded in learning theories on tertiary education : A systematic review. *Computers & Education*, *112*, 97-107.

Spradling, C., Linville, D., Rogers, M. P., & Clark, J. (2015). Are MOOCs an appropriate pedagogy for training K-12 teachers computer science concepts?. *Journal of Computing Sciences in Colleges*, *30*(5), 115-125.

Sullivan, A., Bers, M. U., & Mihm, C. (2017). Imagining, playing, & programming with KIBO: Using KIBO robotics to foster computational thinking in young children. In S. C. Kong, J. Sheldon, & K. Y. Li (Eds.). *Proceedings of the International Conference on Computational Thinking Education* (pp. 110-115). Wanchai, Hong Kong : The Education University of Hong Kong.

Sullivan, A., Strawhacker, A., & Bers, M. U. (2017). Dancing, drawing, and dramatic robots: Integrating robotics and the arts to teach foundational STEAM concepts to young children. In M. S. Khine (Ed.), *Robotics in STEM education: Redesigning the learning experience* (pp. 231-260). Springer International Publishing.

Sysło, M. M., & Kwiatkowska, A. B. (2015). Introducing a new computer science curriculum for all school levels in Poland. In *Informatics in Schools. Curricula, Competences, and Competitions : 8th International Conference on Informatics in Schools : Situation, Evolution, and Perspectives, ISSEP 2015, Ljubljana, Slovenia, September 28-October 1, 2015, Proceedings 8* (pp. 141-154). Springer International Publishing.

Tardif, J. (1992). *L'enseignement stratégique*. Montréal : Éditions Logiques.

Tardif, J. (1999). *Le transfert des apprentissages*. Montréal : Éditions Logiques.

Taylor, C., Zingaro, D., Porter, L., Webb, K. C., Lee, C. B., & Clancy, M. (2014). Computer science concept inventories : past and future. *Computer Science Education*, *24*(4), 253-276.

Tew, A. E. & Guzdial, M. (2010). Developing a validated assessment of fundamental CS1 concepts. In *Proceedings of the 40th ACM technical symposium on computer science education (SIGCSE)* (pp. 97–101), Milwaukee, WI.

Thianthai, C., & Sutamchai, K. (2022, May). Skills that matter : Qualitative study focusing on the transfer of training through the experience of Thai vocational students. In *Frontiers in Education* (Vol. 7, p. 897808). Frontiers Media SA.Thorndike, E. L. (1924). Mental discipline in high school studies. *Journal of Educational Psychology*, *15*(1), 1.

Thorndike, E. L., & Woodworth, R. S. (1901). The influence of improvement in one mental function upon the efficiency of other functions. II. The estimation of magnitudes. *Psychological Review*, *8*(4), 384.

Tomporowski, P. D., Davis, C. L., Miller, P. H., and Naglieri, J. A. (2008). Exercise and children's intelligence, cognition, and academic achievement. *Educational Psychology Review,* 20, 111–131.

Toupin, L. (1995). *De la formation au métier : savoir transférer ses connaissances dans l'action*. ESF éditeur.

Tricot, A. (2020). *Quelles fonctions pédagogiques bénéficient des apports du numérique ?* Paris : Cnesco.

Turgeon, J. B., & Bernatchez, J. (2009). Les données secondaires (5e éd). Dans B. Gauthier, *Recherche sociale : de la problématique à la collecte des données* (pp. 489 - 528).

Turski, W. M., & Wasserman, A. I. (1978). Computer programming methodology. *ACM SIGSOFT Software Engineering Notes*, *3*(2), 20-21.

United Nations Educational, Scientific and Cultural Organization. (2011). *UNESCO ICT competency framework for teachers*. Récupéré en ligne à l'adresse suivante : http ://unesdoc.unesco.org/images/0021/002134/213475e.pdf

Uttal D. H., Meadow N. G., Tipton E., Hand L. L., Alden A. R., Warren C., Newcombe N. S. (2013). The malleability of spatial skills: A meta-analysis of training studies. *Psychological Bulletin*, 139, 352–402.

Vahrenhold, J., Caspersen, M., Berry, G., Gal-Ezer, J., Kölling, M., McGettrick, A., Nardelli, E., Pereira, C., & Westermeier, M. (2017). Informatics Education in Europe : Are We All In The Same Boat? ACM and Informatics Europe.

van Merriënboer, J. J., Clark, R. E., & De Croock, M. B. (2002). Blueprints for complex learning: The 4C/ID-model. *Educational Technology Research and Development, 50*(2), 39–61.

Vegas, E., Hansen, M., & Fowler, B. (2021). Building skills for life. *How to expand and improve computer science education around the world*.

Vollstedt, A. M., Robinson, M., & Wang, E. (2007, April). Using robotics to enhance science, technology, engineering, and mathematics curricula. In *Proceedings of American Society for Engineering Education Pacific Southwest annual conference*, Honolulu : Hawaii.

Wai J., Lubinski D., Benbow C. P. (2009). Spatial ability for STEM domains: Aligning over 50 years of cumulative psychological knowledge solidifies its importance. *Journal of Educational Psychology*, 101, 817–835.

Waite, J., & Sentance, S. (2021). Teaching programming in schools : A review of approaches and strategies. *Raspberry Pi Foundation*.

Wang, C., Shen, J., & Chao, J. (2022). Integrating computational thinking in STEM education : A literature review. *International Journal of Science and Mathematics Education*, *20*(8), 1949-1972.

Watson, C., Li, F. W., & Godwin, J. L. (2014, March). No tests required : comparing traditional and dynamic predictors of programming success. In *Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 469-474).

Weinberg, G. M. (1971). *The psychology of computer programming* (Vol. 29). New York : Van Nostrand Reinhold.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*, 127-147.

Wexelblat, R. L. (Ed.). (2014). *History of programming languages*. Academic Press.

Wexler, B. E., Iseli, M., Leon, S., Zaggle, W., Rush, C., Goodman, A., ... & Bo, E. (2016). Cognitive priming and cognitive training : immediate and far transfer to academic skills in children. *Scientific Reports*, *6*(1), 32859.

Wing, J. (2011). Research notebook : Computational thinking—What and why. *The link magazine*, *6*, 20-23.

Wing, J. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Winslow, L. E. (1996). Programming pedagogy—a psychological overview. *ACM Sigcse Bulletin*, *28*(3), 17-22.

Witherspoon, E. B., & Schunn, C. D. (2022). Sources of gender differences in competency beliefs and retention in an introductory premedical science course. *Journal of Research in Science Teaching*, *59*(5), 695-719.

Xu C., LeFevre J. A. (2016). Training young children on sequential relations among numbers and spatial decomposition: Differential transfer to number line and mental transformation tasks. *Developmental Psychology*, 52, 854–866.

Zavala, L. A., Gallardo, S. C. H., & García-Ruíz, M. Á. (2013, June). Designing interactive activities within *Scratch* 2.0 for improving abilities to identify numerical sequences. In *Proceedings of the 12th International Conference on Interaction Design and Children* (pp. 423-426).

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education*, *141*, 103607.