Rapport d'intégration garantie

Juliette Gaborit, Julien Girard

11 décembre 2017

Introduction

Ce rapport présente le travail effectué dans le cadre du projet d'intégration garantie. Il comprend un bref état de l'art du domaine, une description du problème, notre réponse à la problématique (sous la forme d'un algorithme), une présentation de nos résultats, des critiques et pistes d'améliorations.

1 État de l'art

1.1 Problématiques

L'intégration garantie constitue une partie du calcul garanti, dont l'objectif est de borner les imprécisions dans les calculs numériques ainsi que l'ensemble des solutions possibles (atteignabilité). On peut citer comme exemple d'application des systèmes hybrides, qui mélangent des représentations d'états discrètes et continues. En effet, les systèmes hybrides sont soumis à des changements d'états discrets par leur partie logicielle (un ordinateur étant rythmé par la fréquence d'horloge de son processeur), et continus par leur partie physique (classiquement modélisée par des systèmes d'équations différentielles ordinaires ou algébro-différentielles). Pour représenter les phénomènes d'interaction avec l'environnement extérieur (par exemple les limites de précisions des capteurs), on représente classiquement une variable physique sous forme d'intervalle. On définit alors une *arithmétique d'intervalle* et *l'extension naturelle* qui consiste à appliquer une fonction de $\mathbb{R}^n$ à valeurs dans $\mathbb{R}^n$ à des intervalles. Voir par exemple la partie *Background* de [2] pour une description détaillée de l'extension naturelle.

Le calcul numérique est fondé sur un calcul en temps discret de problèmes à temps continu avec des possibles changements d'états discrets dans le cas de systèmes hybrides. Les problématiques rencontrées dans le calcul garanti sont les suivantes : [1].

- le saut d'état : on calcule les valeurs d'une dynamique à temps discret. Dans le cas d'une simulation numérique classique, si le passage d'une garde qui conduit à un changement instantané de dynamique (changement d'état, réinitialisation) arrive entre deux instants discrets, le changement d'état ne sera pas détecté.
- la dépendance aux pas de temps : le choix de la méthode d'intégration et donc des pas de temps à une forte influence sur la possibilité de détecter des changements d'états ou de dynamique. De plus une simulation à faibles pas de temps est coûteuse et n'assure pas de détecter les passages de garde.
- garantir l'intégration : pour éviter les problèmes précédents, on garantit l'intégration en encadrant les solutions par des intervalles par exemples. Un mauvais encadrement des solutions peut créer des effets d'enveloppement et pose aussi le problème de la détection des changements d'états. Se pose alors la question de la méthode d'encadrement (intervalles, zonotopes, ellipses, fonctions supports) qui ont des atouts et des inconvénients en fonction des opérations à effectuer sur ces encadrements. Par exemple, l'intersection de zonotopes ne donne pas un zonotope.
- l'effet d'enveloppement : lors de l'application successive de fonctions créées par extension naturelle, on introduit un certain pessimisme dans l'évaluation. Par exemple, pour $x = [0, 1]$, l'évaluation $x - x$ devrait produire le résultat 0, or l'arithmétique d'intervalle « classique »donne le résultat $[-1, 1]$. Dans un calcul garanti, les boîtes qui bornent le domaine de la solution s'agrandissent de plus en plus, ce qui introduit une imprécision qui gêne la déduction de propriétés intéressantes sur le calcul.
- la réinitialisation de l'état : une fois la garde détectée, il faut effectuer le changement de dynamique sur des encadrements de la solution. Cela nécessite un encadrement des solutions au passage de la garde suffisamment précis pour initialiser la nouvelle dynamique sans propager de sur-approximations. Le choix des structures d'encadrement et de la méthode d'encadrement dépend aussi de la dynamique et de sa rigidité.
- le coût de la simulation : l'objectif est d'avoir des résultats garantis avec une complexité de calcul et une complexité temporelle minimales.

1.2 Solutions

Plusieurs axes de solutions sont envisagées pour résoudre le passage de garde et obtenir une simulation garantie [1] :

1. le choix de la structure d'encadrement (intervalles, zonotopes, ellipses, fonctions supports)
2. le changement de structures de données en fonction du problème et de la proximité au passage de la garde : certaines structures comme les fonctions supports ont des propriétés intéressantes pour détecter le passage de la garde mais sont peu efficaces pour les zones d'intégration continues (coût, effet d'enveloppement). Détecter la proximité du passage de la garde par des seuils permet un passage à l'échelle.
3. le choix de la méthode de bissection et du contracteur liés au choix de la structure de données

L'article [3] implémente une méthode pour minimiser la sur-estimation de l'intersection entre le flux (simulation continue) et la garde en ayant un coût de calcul et un coût temporel plus faibles que les algorithmes qui existaient auparavant. Lors de la phase de simulation continue, l'algorithme proposé utilise une bissection puis une contraction avec la factorisation de Lohner. L'apport de [3] vient de la gestion de la transition où ils utilisent le contracteur HC4Revise de la bibliothèque Ibex. La méthode est de calculer une solution a priori, de vérifier qu'elle satisfait la condition de garde puis d'effectuer un branch-and-prune sur l'ensemble des variables pour éliminer les parties qui ne satisfont pas les conditions de la garde. Pour diminuer la complexité de l'algorithme, la bissection est effectuée sur l'axe temporel uniquement. L'idée ici est d'adapter la méthode de contraction à proximité de la garde (plutôt que de changer la structure de données).

Pour répondre à la problématique de l'enveloppement, l'article [2] propose quant à lui d'utiliser les « parallélotopes ». Un parallélotope est un triplet $(\mathbb{R}^{n*n}, \mathbb{IR}^n, \mathbb{R}^n)$ <A,u,x> = x + Au ($\mathbb{IR}$) désigne l'espace des intervalles). On peut voir un parallélotope comme l'image d'une boîte dans un repère affine bien choisi. Une propriété qui en découle est que l'image d'un parallélotope par une fonction affine peut être bornée par un parallélotope, ce qui permet d'obtenir de bonnes propriétés lors de compositions (par exemple, la recherche d'un point fixe).

De plus, [2] propose aussi de lier les états possibles entre les sauts en bornant le temps d'intersection par des parallélotopes, à la place des boîtes. L'idée développée est de calculer l'intervalle temporel de croisement de la garde le plus précis possible, pour ensuite simuler la trajectoire après la garde avec l'intervalle de temps nouvellement calculé comme condition initiale.

L'arithmétique affine (définie entre autres dans [5] et [2]) à la place de l'extension naturelle permet de limiter l'effet d'enveloppement en établissant des relations entre les différentes variables d'un problème. Cependant, les opérations non affines comme la multiplication sont mal supportées par cette arithmétique. Un autre problème soulevé dans [4] est la gestion des erreurs numériques liées aux nombres flottants qui impacte sur la simulation garantie et doivent être évaluées et minimisées.

1.3 Dynamique

On étudie la dynamique ci-dessous qui est donnée par le problème suivant : vérifier de manière garantie qu'un joueur de basket met le ballon dans le panier après 1, 2 ou 3 rebonds au sol. Ainsi, la dynamique peut être représentée par les deux états suivants et leur transition :

Etat	Transition
INITIALISATION $\frac{dY}{dt} = Vy_0$, $Y = Y_0$ $\frac{dX}{dt} = Vx_0$ $X = X_0$	INITIALISATION -> CHUTE : affectation : $Vy = Vy_0$; $Vx = Vx_0$
CHUTE : $\frac{dY}{dt} = Vy$, $\frac{dVy}{dt} = -9,81$ $\frac{dX}{dt} = Vx$ $\frac{dVx}{dt} = 0$	CHUTE -> CHUTE garde : $y = 0$ AND $v <= 0$ affectation : $Vy = -c * Vy$; $Vx = cVx$

FIGURE 1 – Dynamique

Paramètres On choisit les paramètres suivants pour l'étude du problème :
Ballon : diamètre 23.8cm, masse 567g, coefficient rebond c=0.8
Panier : distance 6.75m, diamètre 45cm, hauteur 3.05m
Hauteur du joueur : $Y_0 \in [1.6, 2.8]$, $X_0 = 0$ (traduit en intervalle : $[0., 0.]$)
Impulsion initiale :$Vx_0 \in [0, 10], Vy_0 \in [0, 10]$

On cherche à trouver des solutions garanties à ce problème. Nous nous concentrerons sur la gestion du changement de dynamique lors du contact avec le sol (rebond du ballon) et la gestion des contraintes. L'étape d'intégration garantie en dynamique de chute libre sera prise en charge par l'outil Dynibex (lien, [5]).

1.4 Contraintes

On souhaite vérifier que notre ballon rentre dans le panier. Les propriétés suivantes doivent être vérifiées :

1. *Condition globale* $X <= 6.75 + 0.212m$ (avant le dernier rebond, si le ballon dépasse le centre du panier avec la marge d'entrée calculée à partir de la largeur du ballon et du panier, on ne pourra jamais avoir le ballon dans le panier).
2. *Condition dernier rebond* $\frac{dY}{dt} <= 0$ et $Y >= 3.05$ (la balle doit arriver au dessus du panier en descendant)
3. *Condition dernier rebond* $\frac{dY}{dt} <= 0$ et $y = 3.05$ et $6.75 - 0.212 <= x <= 6.75 + 0.212$ (une fois à la hauteur du panier, la balle doit rester dans l'axe du panier tout en descendant)

2 Détails d'implémentation et justification

2.1 Algorithme pour trouver les conditions initiales

L'objectif est de trouver l'ensemble des valeurs initiales telles que le ballon passe dans le panier. A cet effet, nous proposons l'algorithme suivant.

Entrées : Nombre de rebonds souhaités **N** ; Conditions initiales X, Y, Vx, Vy, T
Note : dans un premier temps, seul Y est un intervalle, les autres variables sont des constantes représentées par des intervalles (e.g. X=[0,0])

Algorithme(N,X,Y,Vx,Vy,T)

For (n=0, n<N, n++) :

1. Phase de simulation garantie avec X,Y,Vx,Vy,T et la dynamique
2. Détection du passage de zéro
3. Contraction de l'intervalle de passage de zéro
4. Calcul de X_{new}, Y_{new}, Vx_{new}, Vy_{new}, T_{new} selon le modèle de la dynamique (rebond)
5. Test de la condition globale sur X (*Condition 1*)

Sur le dernier rebond, on effectue une phase en plus pour tester les conditions d'arrivée dans le panier.
Si les conditions sont vérifiées,on enregistre l'intervalle initial dans la liste des intervalles valides.
Sinon le panier n'est pas atteint. Dans ce cas, soit le diamètre de l'intervalle initial (Y) est plus grand qu'ϵ. On réitère alors l'algorithme précédent avec des intervalles initiaux de diamètre divisé par deux : faire **Algorithme(N,X,[Ymin, (Ymin+Ymax)/2], Vx, Vy, T)** et **Algorithme(N,X,[(Ymin+Ymax)/2, Ymax], Vx, Vy, T)**. Soit le diamètre de l'intervalle initial est déjà trop petit, on le sauve dans la liste des intervalles "limites" pour vérifier notre solution.

Sorties :

- la liste des intervalles initiaux valides (leur union devrait être continue),
- la liste des intervalles dits "limites" (ils devraient être l'extrêmité de l'union des intervalles initiaux valides)

2.2 Phase de simulation garantie jusqu'à détection de la garde

Étapes de notre simulation :

1. Initialisation avec conditions initiales
2. Calcul du temps d'intersection exact t_e entre la boule et le sol avec la dynamique exacte
3. Simulation garantie jusqu'au temps d'intersection
4. Contraction des intervalles obtenus à l'issue de la simulation
5. Redéfinition du problème en prenant comme conditions initiales les intervalles contractés, et en prenant la nouvelle dynamique
6. Lancement de la simulation
7. Vérification du respect des contraintes

La méthode d'intégration garantie utilisée est une Runge-Kutta d'ordre 4 garantie, fournie par l'outil DynIbex.

L'étape 2. est utile pour arrêter la simulation garantie au bon moment, pour ne pas prendre trop de ressources. Une fois trouver le temps T exact de l'intersection entre le flux et la garde, on ajoute une marge de ΔT pour s'assurer de ne pas rater l'intervalle du passage de la garde. Pour l'instant, ΔT est choisi arbitrairement égal à 1s.

Lors de l'élaboration de cette solution, nous avons eu à choisir entre faire un calcul exact (la solution du système d'ODE étant connue), ou de faire une première simulation non-garantie pour obtenir une estimation du temps de changement de dynamique. L'objectif étant dans les deux cas de récupérer une estimation du temps de franchissement de la garde pour ensuite lancer la simulation garantie pour ce temps donné (modifié de façon à englober tous les croisements possibles, ce qui n'est pas fait dans notre algorithme). C'est finalement la première façon de faire qui a été retenue, pour des raisons de simplicité et de coût.

2.3 Détection de la garde - gestion du rebond

Condition y=0 Il y a passage de la garde dans deux situations. Soit on trouve un intervalle où la borne maximale sur y est positive et la borne minimale sur y est négative. Alors y=0 appartient à cet intervalle. L'autre possibilité est que la garde soit comprise entre deux intervalles : l'intervalle avant la garde a ses deux bornes positives, celui après la garde a ses deux bornes négatives. Dans ce cas, on effectue l'union de ces deux intervalles (sur les 5 dimensions), ce qui nous fait perdre en précision.

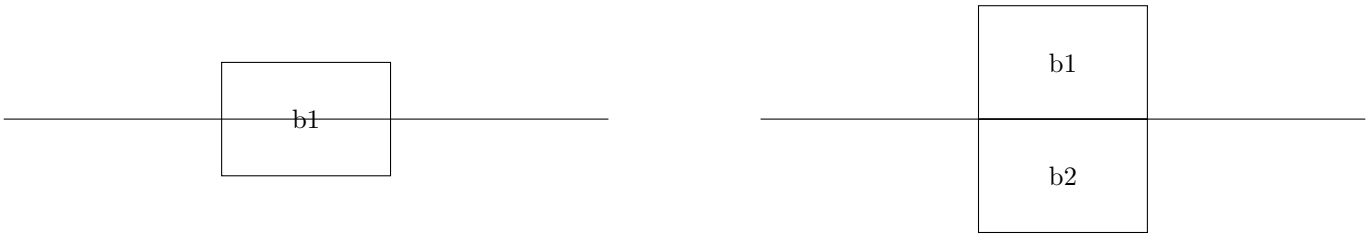

FIGURE 2 – Différentes possibilités de franchissement de la garde

Condition v<=0 On ajoute un test que la vitesse est négative pour éviter de sélectionner les intervalles qui pourraient vérifier les conditions précédentes en phase de départ après le rebond de la balle. En effet, la simulation peut donner des résultats arrondis à des valeurs négatives proches de 0 au début du rebond alors que la valeur est positive en théorie.

Contraction de la garde Une fois que l'on a isolé l'intervalle où il y a passage de la garde, on essaye de trouver la garde en appliquant des contractions Forward-Backward puis FixPoint de la bibliothèque Ibex.

Les conditions utilisées pour contracter l'intervalle sont les suivantes :

- Sur Y : $0 \leq Y \leq \epsilon$ fixé à 0.01
- Sur T : de la dynamique et de la condition précédente, on déduit $0 \leq \frac{-g*T^2}{2} + Vy_0T + Y_0 \leq \epsilon$. *Note : Vy_0 et Y_0 étant des intervalles, il faut choisir leur borne. Y_0 étant positif, pour avoir une inégalité toujours vérifiée, il faut prendre Y_0 maximal pour la comparaison à zéro et minimal pour la comparaison avec ϵ. Pour toutes les comparaisons suivantes, on choisit les bornes des intervalles de cette manière.* Ainsi,

  - on a $0 \leq \frac{-gT^2}{2} + maxEvaluate(Vy_0)T + \overline{Y_0}$ avec maxEvaluate qui renvoit $\underline{Vy_0}$ si $Vy_0 \leq 0$ et $\overline{Vy_0}$ sinon.
  - et $\frac{-g*T^2}{2} + minEvaluate(Vy_0T + \overline{Y_0} \leq \epsilon$ avec minEvaluate qui renvoit $\underline{Vy_0}$ si $Vy_0 \geq 0$ et $\overline{Vy_0}$

- Sur X : on a X qui vérifie toujours les conditions données par sa dynamique. $min(X(t)) <= X <= max(X(t))$ avec $X(t) = Vx_0T + X_0$. En lieu et place de Vx_0 et X_0 qui sont des intervalles, on prend la borne minimale pour la comparaison à gauche et la borne maximale pour la comparaison à droite étant donné Vx_0 et X_0 positifs.
- Sur Vx : de la même manière que pour X, Vx vérifie sa dynamique avec les bornes adéquates des intervalles : $min(Vx(t)) <= Vx <= max(Vx(t))$ avec $Vx(t) = Vx_0$.
- Sur Vy : Vy vérifie sa dynamique avec les bornes adéquates des intervalles : $min(Vy(t)) <= Vy <= max(Vy(t))$ avec $Vy(t) = -gT + Vy_0$. On prend les bornes suivantes :
  - on a $0 \leq -gT + maxEvaluate(Vy_0)$
  - et $-gT + minEvaluate(Vy_0) \leq \epsilon$

2.4 Contraintes d'arrivée dans le panier

Il existe deux types de contraintes comme explicité en 1.4. La première contrainte globale est appliquée à la fin de chaque rebond est consiste à vérifier que le ballon est resté entre le tireur et le panier. Les autres sont des conditions sur la position de la balle par rapport au panier, évaluée uniquement lors du dernier rebond. On procède en itérant sur tous les intervalles résultants de la simulation et en en cherchant un qui vérifie les trois conditions (vitesse négative, position verticale à hauteur du panier et position horizontale entre les deux bords du panier).

3 Résultats

Suivent les résultats de notre algorithme pour zéro rebond, suite à des problèmes constatés avec au moins un rebond (la dimension X, une fois passée par les contracteurs, est vide après le deuxième rebond).

Nous ne bissectons que la hauteur initiale dans notre algorithme, les vitesses sont fixées manuellement.

Sauf indication contraire, les graphiques auront le temps en abscisse et la hauteur en ordonnée. Le panier sera représenté par un rectangle vert. On constate que pour cet ensemble de conditions initiales, il existe de potentiels tubes de solutions. En faisant tourner notre algorithme, nous obtenons les résultats de la figure 3

Remarques Malgré la bissection effectuée de manière récursive sur l'intervalle initial si le panier n'est pas atteint, nous n'obtenons pas de résultat. Cela est sûrement dû aux imprécisions qui sont propagées pendant la phase de simulation. L'utilisation d'une arithmétique affine avec une structure de données plus complexe que les intervalles aurait sûrement donné de meilleurs résultats.

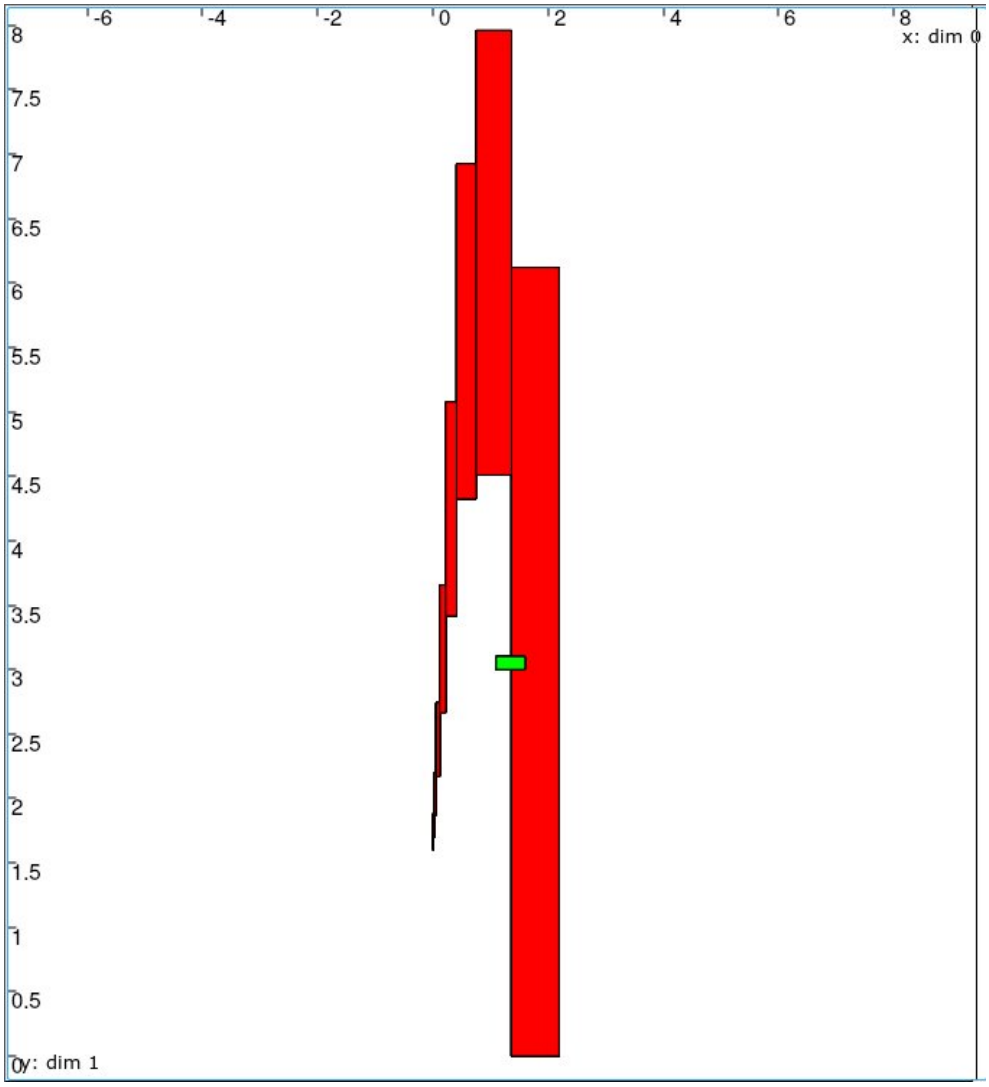

FIGURE 3 – Vx0=5,Vy0=10

4 Critiques et améliorations

Algorithme Nous avons implémenté un algorithme qui permet de donner les intervalles initiaux qui valident une condition finale de manière garantie et automatisée. Cet algorithme détecte bien le passage de la garde (y=0) puisque la valeur exacte est comprise dans l'intervalle évalué. Néanmoins, plusieurs problèmes ont été mis en évidence, outre le temps de calcul important. Plusieurs axes d'amélioration et limites sont explicités ci-dessous.

Choix de la structure de données Comme remarqué dans la section 3, le choix d'une arithmétique affine avec des parallélotopes pour les différentes variables (excepté le temps) aurait donné de meilleures approximations pendant la simulation garantie "continue" et aurait diminué l'effet d'enveloppement qui donne des intervalles très larges au moment du passage de la garde.

Contraction lors du rebond Lors du deuxième rebond, on obtient que l'intervalle contracté selon l'axe X est vide, malgré une bonen contraction selon les autres axes. Ainsi, cela bloque la poursuite de la simulation. Deux facteurs peuvent être la cause de ce problème. Le premier est la mauvaise évaluation de l'intervalle initial après le premier rebond. En effet, les contraintes de contraction dépendent des conditions initiales, et si celles-ci ont été mal évaluées après le premier rebond, cela donnera des résultats faux sur la suite. Le second facteur pourrait être l'inter-dépendance des contraintes. En effet, les contraintes sur X, Vx, Vy dépendent du temps T évalué et contracté en même temps. Une solution possible serait de changer les contraintes de contraction (adapter la valeur d'ϵ qui majore y, ne pas contracter selon certains axes) mais cela se répercuterai sur la suite de la simulation garantie. L'autre solution serait d'avoir une meilleure précision lors de la

phase de simulation "continue" pour ne pas avoir à contracter selon certains axes.

Changement de signe de la vitesse On peut remarquer que l'intervalle lors de la phase chute (Vy négatif) est beaucoup plus grand que ceux lors de la phase de montée (Vy positif). Dans notre implémentation, nous n'avons pas modifié le calcul garanti lors des phases de simulation continu. Un traitement spécifique, un changement de contracteur, un ajout de contraintes pourrait être adéquat lors du changement de signe de Vy0 pour diminuer l'effet d'enveloppement observé.

Références

[1] Goran Frehse. Verification of hybrid systems and validation of controllers. 2015.

[2] Alexandre Goldsztejn and Daisuke Ishii. A parallelotope method for hybrid system simulation. *Reliable Computing*, 23 :163–185, 2016.

[3] Nacim Ramdani Moussa Maïga and Louise Travé-Massuyès. A fast method for solving guard set intersection in nonlinear hybrid reachability. 2013.

[4] Alexandre Chapoutot Olivier Bouissou, Samuel Mimram. Hyson : Set-based simulation of hybrid systems. 2012.

[5] Julien Alexandre Dit Sandretto and Alexandre Chapoutot. *Validated Solution of Initial Value Problem for Ordinary Differential Equations based on Explicit and Implicit Runge-Kutta Schemes.* PhD thesis, ENSTA ParisTech, 2015.