Approximation de flots géométriques : des méthodes de champ de phase aux réseaux de neurones

Garry Terii

▶ **To cite this version:**

Garry Terii. Approximation de flots géométriques : des méthodes de champ de phase aux réseaux de neurones. Intelligence artificielle [cs.AI]. Université Claude Bernard - Lyon I, 2022. Français. NNT : 2022LYO10041 . tel-04074100

HAL Id: tel-04074100
https://theses.hal.science/tel-04074100

Submitted on 19 Apr 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THESE de DOCTORAT DE
L'UNIVERSITE CLAUDE BERNARD LYON 1

Ecole Doctorale N° 512
InfoMaths

Discipline : Mathématiques

Soutenue publiquement le 23/09/2022, par :
Garry Terii

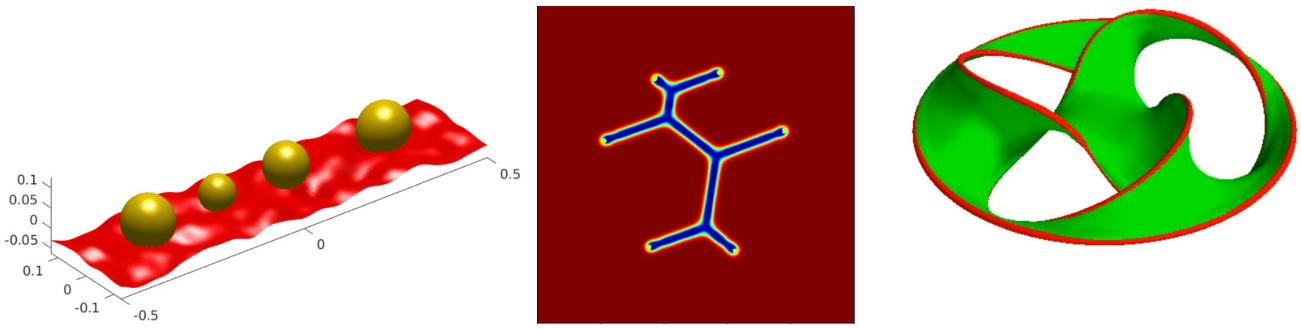
Approximation de flots géométriques : des méthodes de champ de phase aux réseaux de neurones

Devant le jury composé de :

Élie Bretin	Maître de conférences, INSA Lyon	Directeur de thèse
Simon Masnou	Professeur des Universités, Université Claude Bernard Lyon 1	Directeur de thèse
Valérie Perrier	Professeure des Universités, Université Grenoble Alpes	Présidente
Antonin Chambolle	Directeur de recherche, CNRS & Université Paris-Dauphine	Rapporteur
Emmanuel Maitre	Professeur des Universités, Grenoble INP — Université Grenoble-Alpes	Rapporteur
Laurence Cherfils	Maîtresse de conférences, Université de la Rochelle	Examinatrice
Frédéric Lagoutière	Professeur des Universités, Université Claude Bernard Lyon 1	Examinateur
Flore Nabet	Maîtresse de conférences, École Polytechnique	Examinatrice
Roland Denis	Ingénieur de recherche, CNRS	Invité

Institut Camille Jordan
Laboratoire de recherche en mathématiques Lyon/Saint-Étienne

Approximation de flots géométriques : des méthodes de champ de phase aux réseaux de neurones



Garry Terii

Thèse de doctorat

Université Claude Bernard Lyon 1

Table des matières

Problématiques et contextes

Chapitre 1

Introduction

Plusieurs phénomènes physiques ou biologiques peuvent être modélisés comme des problèmes de minimisation d'énergies d'interfaces sous des contraintes variées. On les retrouve aussi dans des contextes numériques, par exemple en traitement d'image, en informatique graphique et plus généralement en science des données. Ces énergies de nature géométrique dépendent classiquement de l'aire des interfaces ou de descripteurs géométriques d'ordre plus élevé comme la courbure. On appelle périmètre d'un ensemble régulier $\Omega \subset \mathbb{R}^N$ l'aire de son bord $\partial\Omega$:

$$P(\Omega) = \text{Aire}(\partial\Omega). \tag{1.0.1}$$

Cette notion s'étend à la classe plus générale des ensembles dits *de périmètre fini* [13].

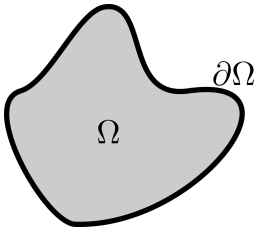


FIGURE 1.1 – Un domaine $\Omega \subset \mathbb{R}^2$ et son bord $\partial\Omega$.

La notion de périmètre explique par exemple la forme sphérique d'une bulle de savon (parmi toutes les surfaces enfermant un volume d'air donné, c'est la bulle sphérique qui a le plus petit périmètre) et plus généralement la forme de films de savon s'appuyant sur des contours donnés. L'observation de ces formes naturelles et la volonté de les expliquer a joué un rôle fondamental dans le développement du calcul des variations autour des notions de problème isopérimétrique et de surfaces minimales.

FIGURE 1.2 – Une bulle de savon (source : Wikipedia)

Lorsque plusieurs bulles de savon sont accolées, comme par exemple dans une mousse de savon, certaines propriétés simples sont vérifiées par la structure à l'équilibre : le système de bulles se décompose en plusieurs éléments de surface lisses dont la courbure est constante, et lorsque trois éléments de surface se rejoignent, ils se raccordent selon une courbe de jonction et leurs plans tangents font deux à deux un angle de $120°$. D'un point de vue énergétique, cette configuration minimise localement la somme des périmètres de chaque élément de surface intervenant dans le système. C'est donc une généralisation du cas d'une seule bulle : la forme à l'équilibre est un minimiseur local d'un périmètre multiphase sous la contrainte que la somme des volumes d'air enfermés par chaque film de savon soit fixée.

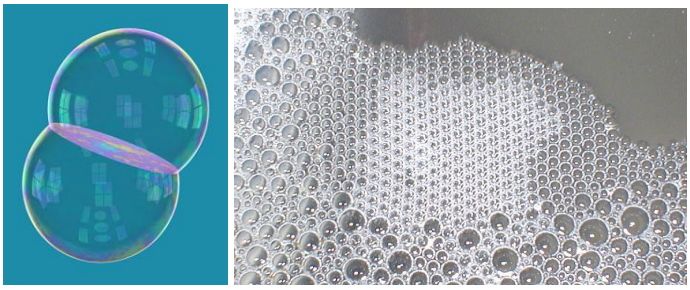


FIGURE 1.3 – A gauche, une double bulle de savon ; à droite, une mousse de savons.

La forme générale d'un périmètre multiphase est la suivante :

$$P(\mathbf{\Omega}) = \sum_{i,j=1}^{N} \sigma_{ij} \text{Aire}(\Gamma_{ij}), \tag{1.0.2}$$

où $\mathbf{\Omega} = (\Omega_1, \cdots, \Omega_N)$ et $\Gamma_{ij} = \Omega_i \cap \Omega_j$ représente l'interface entre les phases Ω_i et Ω_j. Les σ_{ij} sont les coefficients de tension de surface associés aux interfaces Γ_{ij}, ils dépendent de la nature du système sous-jacent.

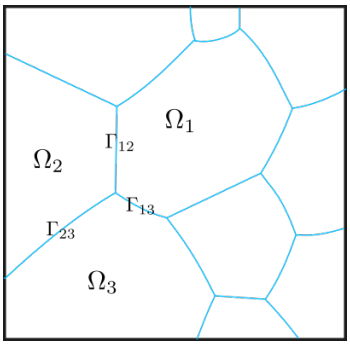


FIGURE 1.4 – Représentation d'un système à plusieurs phases en dimension 2.

Dans le cas de plusieurs bulles de savon en contact, les coefficients de tension de surface sont identiques pour toutes les interfaces et nous avons mentionné l'angle de $120°$ qui caractérise la répartition équilibrée des phases autour des lignes de contact triple. Des répartitions non équilibrées et des angles différents apparaissent lorsque les tensions de surface sont différentes. C'est le cas par exemple lorsqu'on dépose une goutte de liquide sur une surface solide non absorbante (voir la figure 1.5). Les angles qui se forment au niveau de la ligne triple dépendent des tensions de surface Γ_{LS}, Γ_{SV} et Γ_{SV} aux interfaces liquide-solide, solide-vapeur et vapeur-liquide, respectivement.

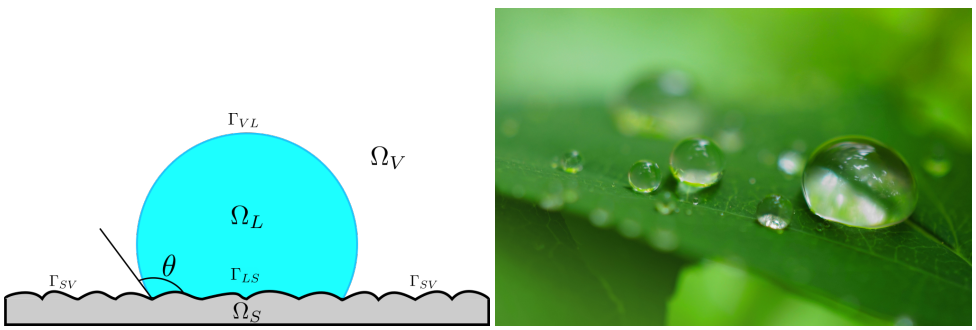


FIGURE 1.5 – Schéma du mouillage d'un liquide sur une surface solide non absorbante et exemple réel du mouillage de plusieurs gouttelettes d'eau déposées sur une feuille (source : wikipedia).

Selon le contexte, le périmètre qui intervient dans l'énergie (1.0.2) peut être de nature isotrope ou anisotrope. Il est isotrope pour expliquer la forme hexagonale des alvéoles dans une ruche d'abeilles, celle de de

bulles de savons (Figure 1) ou celle du mouillage de gouttes d'eau sur une feuille (Figure 1.5).

Dans d'autres situations, le périmètre est de nature anisotrope au sens où les interfaces s'orientent selon des directions privilégiées, c'est le cas notamment pour les matériaux polycristallins, voir la figurefig :cristal.

Nous avons mentionné quelques exemples naturels associés naturellement au périmètre multiphase. On le retrouve également comme énergie de régularisation ou de pénalisation en traitement d'images pour différents problèmes de débruitage, de restauration, de segmentation, de détection d'objets, etc. [16, 137, 244] et plus généralement en science des données.

FIGURE 1.6 – Un matériau polycristallin et une représentation schématique des grains composant ce type de matériau.

Dans les exemples que nous avons mentionnés jusqu'ici, les interfaces sont des bords de domaine et nous avons défini le périmètre comme l'aire de ces bords. De nombreuses situations peuvent cependant être modélisées comme des problèmes de minimisation sous contrainte de l'aire d'interfaces qui ne sont pas nécessairement des domaines. C'est le cas pour deux problèmes emblématiques qui jouent un rôle important dans cette thèse : le problème de Plateau et le problème de Steiner.

Le problème de Plateau (Figure 1.7) est celui des surfaces minimales : étant donné une courbe fermée ou plusieurs courbes fermées dans $\mathbb{R}^n$, on cherche la surface ou les surfaces $(n-1)-$dimensionnelles d'aire totale minimale dont le bord contient ces courbes.

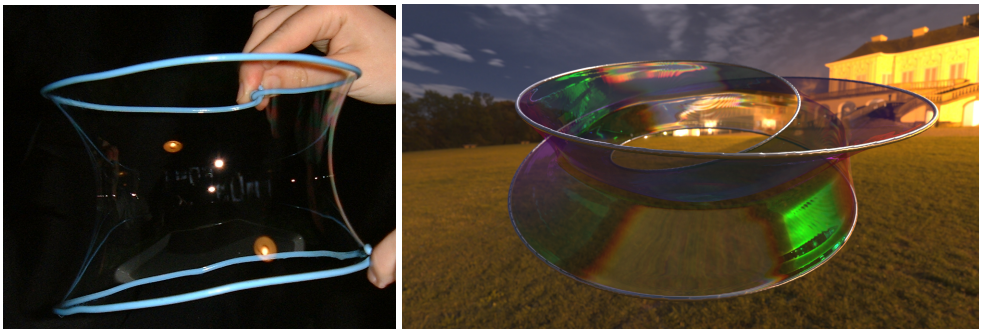


FIGURE 1.7 – Exemples de formes optimales au problème de Plateau. A gauche : surface minimale obtenu à l'aide de films de savon. Les bords sont donnés par les deux anneaux bleus (Source : Wikipedia). A droite : solution numérique où le bord est un ruban de Möbius. Cette solution a été obtenue à partir d'une méthode numérique basée sur les réseaux de neurones et que nous développerons dans cette thèse.

Le problème de Steiner (voir Figure 1.8) est celui de trouver dans $\mathbb{R}^n$ l'ensemble compact et connexe de longueur minimale qui contient un ensemble donné de points. Cet ensemble minimal est appelé arbre de Steiner. Le problème de Steiner a des applications en conception de circuits électroniques et en télécommunication. Des algorithmes utilisés en théorie des graphes permettent d'approcher très efficacement des arbres de Steiner 2D même si le nombre de points prescrits est très grand, ces algorithmes se généralisent toutefois très mal aux dimensions supérieures.

Les problèmes de Plateau et de Steiner sont très intéressants tant d'un point de vue théorique que numérique car ils peuvent faire intervenir des interfaces non orientables ou de co-dimension supérieure à 1.

Les énergies d'interfaces non nécessairement orientables apparaissent dans de nombreuses situations physiques, biologiques mais aussi en ingénierie ou dans des contextes numériques, par exemple en traitement d'images. On les retrouve notamment dans des problèmes de segmentation d'images, par exemple ceux qui font intervenir la minimisation d'une fonctionnelle de type Mumford-Shah impliquant le périmètre d'interfaces non nécessairement orientables (voir par exemple [225, 18]).

D'autres phénomènes physiques ou biologiques et d'autres problèmes numériques ou d'ingénierie nécessitent que l'on utilise des énergies d'ordre plus élevé faisant par exemple intervenir les courbures de la

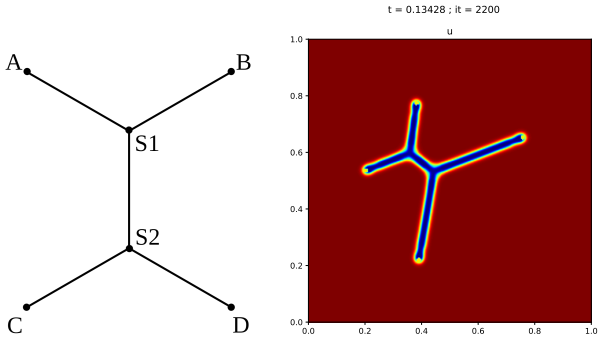


FIGURE 1.8 – Exemples de formes optimales au problème de Steiner. A gauche : l'arbre de Steiner associé aux 4 sommets d'un rectangle (source Wikipedia). On observe la formation de deux points S_1 et S_2 lieux de rencontres de branches formant des angles de 120 degrés. A droite : solution numérique construite à partir de quatre points disposés aléatoirement. Cette solution a été obtenue à partir d'une méthode numérique basée sur les réseaux de neurones que nous présentons dans cette thèse.

FIGURE 1.9 – Segmentation d'une image en utilisant un algorithme basé sur la fonctionnelle de Mumford-Shah (image extraite de [225]).

surface comme l'énergie de Willmore.

$$W(\Omega) = \int_{\partial\Omega} H^2 \, d\mathcal{H}^2, \tag{1.0.3}$$

où H est la courbure moyenne associée à la surface $\partial\Omega$ et $\mathcal{H}^2$ désigne la mesure de Hausdoff 2-dimensionnelle [156, 243].

Cette énergie est d'abord apparue dans sa version unidimensionnelle dans des travaux de Bernoulli et d'Euler relatifs à l'étude de la dynamique des tiges élastiques [296]. Depuis, elle est utilisée pour la modélisation dans de nombreuses applications en science des matériaux, en biologie (pour expliquer la forme des globules rouges) mais aussi en traitement d'images. Dans ce dernier contexte, sa dépendance à la courbure moyenne la rend intéressante comme énergie de régularisation, par exemple pour lisser des formes 3D pixellisées [70], pour segmenter, pour extrapoler, ou pour reconstruire une forme 3D à partir de coupes 2D, etc., voir différents exemples dans [15, 66, 85, 151, 328, 68].

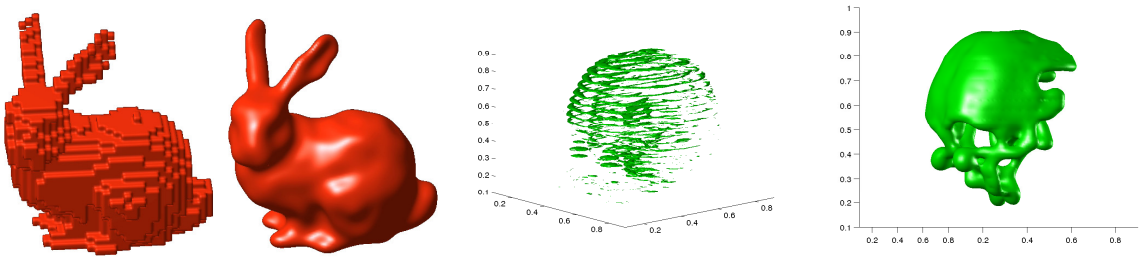


FIGURE 1.10 – Deux applications de l'énergie de Willmore : régularisation (à gauche) et reconstruction à partir de coupes (à droite)

Comme souvent en optimisation, la recherche de minimiseurs se fait principalement de deux manières. La première consiste à trouver de manière explicite les minimiseurs par analyse-synthèse en utilisant les outils du calcul variationnel avec l'étude de l'équation d'Euler-Lagrange si le problème le permet. La deuxième manière consiste plutôt à approcher les minimiseurs en partant d'une configuration initiale et en suivant une dynamique qui tend à minimiser la fonctionnelle en question. Dans ce deuxième cas, on parle de flot

de gradient lorsque la dynamique suit la direction de plus grande pente et pour une énergie géométrique on retrouve un cas particulier de la notion de flot géométrique. C'est cette dernière approche qui nous intéresse car elle permet, d'une part, de trouver de bonnes approximations de surfaces minimales ou d'arbres de Steiner, et d'autre part, de décrire certains phénomènes physiques comme les phénomènes de mouillage ou la croissance de matériaux polycristallins.

Cette thèse est consacrée à l'approximation de flots géométriques et la simulation des ces approximations par des méthodes numériques robustes et efficaces. Dans ce qui suit nous présentons les différents flots géométriques qui sont au cœur de cette thèse.

Chapitre 2

Flots géométriques d'interface

2.1 Mouvement par courbure moyenne

Le flot par courbure moyenne est sans doute, parmi les flots géométriques, l'un des plus connus et des mieux compris. Il doit son succès, d'une part, à ses très nombreuses applications en physique et en ingénierie notamment et, d'autre part, à la richesse de ses aspects théoriques. Le lecteur trouvera des exemples d'applications de ce mouvement dans le livre de Sethian [304] et pour une étude plus approfondie de ce flot, on pourra se référer aux ouvrages [246, 12, 33] et aux références qui y sont données.

2.1.1 Cas isotrope

Dans un cadre régulier, on dit qu'un domaine $\Omega(t) \subset \mathbb{R}^d$ dépendant du temps évolue par courbure moyenne lorsque la vitesse normale intérieure $V_n(t)$ à l'interface $\partial\Omega(t)$ coïncide avec la courbure moyenne scalaire sur $\partial\Omega(t)$, i.e.

$$V_n(t) = H(t), \tag{2.1.1}$$

avec la convention que $H(t) \geq 0$ si $\Omega(t)$ est convexe. Nous prendrons cette même convention sur l'orientation de la vitesse normale pour tous les autres flots présentés dans cette thèse.

Cette dynamique s'exprime aussi comme le flot de gradient pour la métrique L^2 (voir par exemple [7]) du périmètre de $\Omega(t)$,

$$P(\Omega(t)) = \int_{\partial\Omega(t)} d\mathcal{H}^{d-1}, \tag{2.1.2}$$

où $\mathcal{H}^{d-1}$ est la mesure de Hausdorff $(d-1)-$dimensionnelle [156, 243]. C'est donc un mouvement naturellement associé aux surfaces minimales [103].

L'analyse mathématique du mouvement par courbure moyenne est délicate car les solutions de (2.1.1) peuvent développer des singularités en temps fini. Les premiers résultats théoriques sur ce flot sont dûs à Huisken [199], Gage et Hamilton [163], et Grayson [176] qui ont étudié l'existence et le comportement asymptotique du flot pour une configuration initiale régulière. En particulier, ces auteurs ont montré l'existence de solutions régulières en temps fini et prouvé qu'un ensemble convexe reste convexe sous l'effet de (2.1.1) et se réduit en un point en temps fini, tout en devant asymptotiquement sphérique. En dimension deux, tout domaine à bord régulier devient convexe en temps fini puis évolue pour se concentrer en un point. En dimension strictement supérieure à deux, le résultat précédent n'est plus valide lorsque la condition initiale n'est pas convexe et l'évolution peut même présenter des singularités et des changements de topologie en temps fini [177]. Un exemple célèbre où ce type de phénomène apparaît est celui de la forme en haltère en dimension trois. Notons enfin qu'une propriété fondamentale du flot de courbure moyenne est son principe d'inclusion qui permet entre autres choses d'assurer l'unicité du flot régulier en temps fini.

Nous montrons en figure 2.1 un exemple de mouvement par courbure moyenne obtenu à partir d'une fleur comme forme initiale.

Les questions de définition, puis d'existence et d'unicité du flot deviennent délicates dès l'apparition de singularités ou de changements de topologie. Ces questions concernent en particulier le problème de la poursuite de la solution d'une manière raisonnable après une singularité. Pour y donner des éléments de réponse, on utilise le concept de solution faible du mouvement par courbure moyenne qui dépend du choix de la description mathématique du flot. On voit en effet qu'une description classique en utilisant par exemple une formulation paramétrique n'est plus adaptée lorsqu'un changement de topologie se produit.

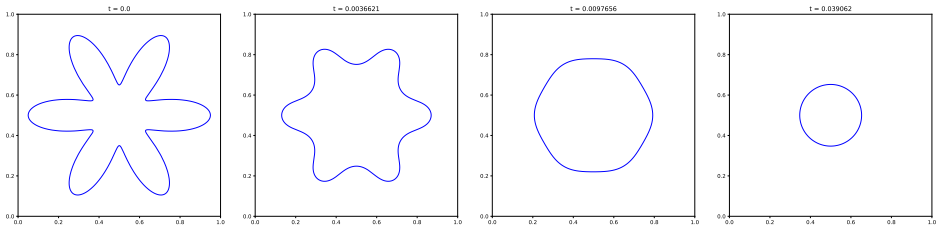


FIGURE 2.1 – Mouvement par courbure moyenne d'une fleur. Chaque image représente l'interface $\partial\Omega(t)$ à différent instants t.

Les premières contributions sur le concept de solution faible du mouvement par courbure moyenne sont dues à Brakke [64, 326] dans le cadre de la théorie géométrique de la mesure. Il a introduit une formulation (qui porte son nom aujourd'hui) qui décrit le mouvement par courbure moyenne grâce aux varifolds [314], une notion de surfaces faibles possédant de bonnes propriétés de compacité et munies (sous certaines conditions) d'une notion de courbure généralisée. Le flot de Brakke possède de bonnes propriétés de compacité mais il est bien connu qu'a priori cette formulation n'a ni unicité ni continuité dans le temps : une solution (ou une partie de celle-ci) peut instantanément disparaitre même dans le cas régulier. Un autre concept intéressant de solution faible est la formulation level-set [271, 153] que nous décrivons plus loin dans ce manuscrit en section 3.2 et qui décrit l'évolution de l'interface à travers la ligne de niveau d'une fonction auxiliaire. C'est un concept qui permet d'assurer des propriétés d'existence et d'unicité du flot [97] même après l'apparition de singularités ou de changements de topologie. Plusieurs autres notions de solution faible ont été proposées dans la littérature, selon des points de vue différents. A ce sujet, nous renvoyons le lecteur à l'ouvrage de Bellettini [33] et ses références, et aux récents travaux de Laux et al [216, 192].

2.1.2 Cas multiphase

Lorsque plusieurs phases sont en interaction, la notion de mouvement par courbure moyenne est localisée sur chaque interface commune à deux phases et des conditions d'angles sont imposées aux lieux de jonctions entre plusieurs phases. Plus précisément, pour une partition $(\Omega_i)_{1\leq i\leq L}$ d'un domaine $\Omega \subset \mathbb{R}^d$ d'ensembles relativement fermés dans Ω, la vitesse V_{ij} à l'interface $\Gamma_{ij} = \partial\Omega_i \cap \partial\Omega_j \cap \Omega$ (pour $i \neq j$) qui sépare les phases Ω_i et Ω_j est donnée par

$$V_{ij} = m_{ij}\sigma_{ij}H_{ij}, \tag{2.1.3}$$

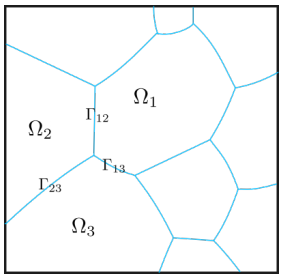


FIGURE 2.2 – Configuration du mouvement par courbure moyenne en contexte multiphase.

avec la convention $\Gamma_{ii} = \emptyset$ pour tout $i \in \{1, \ldots, L\}$. Dans cette expression, H_{ij} désigne la courbure moyenne à l'interface Γ_{ij}, σ_{ij} est la tension de surface et m_{ij} le coefficient de mobilité. On impose que

$$\sigma_{ij} = \sigma_{ji} > 0, \forall i \neq j \quad \text{et} \quad \sigma_{ii} = 0$$

Dans ce système, des singularités apparaissent souvent au cours de l'évolution même lorsque la condition initiale est régulière. Parmi ces singularités, les seules structures persistantes sont les lieux de jonctions où se rencontrent trois phases. En ces lieux, la condition de Herring est vérifiée :

$$\sigma_{ij}n_{ij} + \sigma_{ik}n_{ik} + \sigma_{kj}n_{kj} = 0, \tag{2.1.4}$$

où n_{ij} représente la normale à l'interface Γ_{ij} au niveau de la singularité. En dimension 2, la condition de Herring permet d'en déduire les angles qui se forment entre les interfaces aux points de jonction triple en fonction de leur tension de surface. Dans la configuration représentée en figure 10.17 la condition d'angle est donnée par

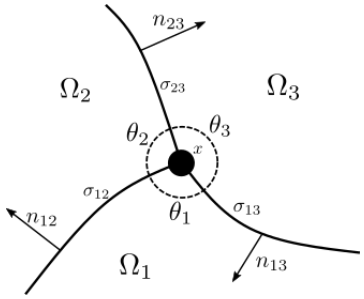


FIGURE 2.3 – Configuration en un point de jonction entre trois phases en 2D.

$$\frac{sin(\theta_1)}{\sigma_{23}} = \frac{sin(\theta_2)}{\sigma_{13}} = \frac{sin(\theta_3)}{\sigma_{12}}.$$

En particulier, lorsque $\sigma_{ij} = 1$ pour tout $i \neq j$, les angles aux points triples sont tous égaux à 120 degrés.

Les tensions de surface $\sigma_{i,j}$ et les mobilités m_{ij} jouent un rôle important. D'un point de vue physique, on peut les interpréter de la façon suivante : les tensions de surfaces σ_{ij} traduisent une contribution énergétique et sont donc associées à la forme d'équilibre du système alors que les mobilités m_{ij} caractérisent la cinétique de l'interface.

Avec ce point de vue, la dynamique (2.1.3) s'exprime comme flot de gradient du périmètre multiphase

$$P\left(\Omega_1, \cdots, \Omega_N\right) = \frac{1}{2} \sum_{1 \leq i < j \leq N} \sigma_{ij} \mathcal{H}^{d-1}(\Gamma_{ij}), \tag{2.1.5}$$

pour la métrique

$$\langle \mu, \nu \rangle = \sum_{1 \leq i < j \leq N} \int_{\Gamma_{ij}} \frac{1}{m_{ij}} \nu_i \mu_i \, d\mathcal{H}^{d-1},$$

et la condition (2.1.4) est équivalente à la loi de Young [259, 86, 243]

$$\sigma_{ik} \leq \sigma_{ij} + \sigma_{jk}, \text{ pour tout } i, j, k \in \{1, \cdots, N\}. \tag{2.1.6}$$

2.1.3 Cas anisotrope

Nous terminons cette brève description du mouvement par courbure moyenne avec la notion d'anisotropie qui intervient lorsque le mouvement privilégie certaines directions et que l'on retrouve dans un grand nombre de phénomènes physiques et biologiques, par exemple en croissance cristalline [324].

De nombreux problèmes physiques anisotropes peuvent être modélisés à l'aide d'une tension de surface qui dépend de la normale. Dans ce cas, le périmètre est remplacé par un périmètre anisotrope [42],

$$P_\gamma(\Omega) = \int_{\partial\Omega} \gamma(n(x)) \, d\mathcal{H}^{d-1}(x), \tag{2.1.7}$$

où $n(x)$ est la normale unitaire extérieure à $\partial\Omega$ en x et $\gamma : \mathbb{R}^d \to \mathbb{R}_+$ est une fonction d'anisotropie donnée. Lorsque celle-ci est donnée par la norme euclidienne, $\gamma(n(x)) = |n(x)| = 1$, on retrouve le cas isotrope avec la notion classique du périmètre.

Dans le cas général, l'anisotropie γ est supposée convexe pour assurer la semi-continuité inférieure de l'énergie (2.1.7) pour la convergence forte des fonctions caractéristiques des domaines de $\mathbb{R}^d$. On parle alors d'anisotropie convexe et si de plus $\gamma \in \mathcal{C}^2\left(\mathbb{R}^d \setminus \{0\}\right)$ on dit que l'anisotropie est régulière.

Tout comme dans le cas isotrope, le périmètre anisotrope (2.1.7) vérifie une inégalité isopérimétrique [152, 162, 322, 321, 320] et les formes optimales sont (à translation près) les formes de Wulff qui, pour une anisotropie γ donnée, sont définies par

$$B_{\gamma,R,x_0} = \left\{x \in \mathbb{R}^d, \ \gamma(x - x_0) \leq R\right\}, \tag{2.1.8}$$

où $R > 0$ et $x_0 \in \mathbb{R}^d$ sont respectivement le rayon et le centre de la forme de Wulff.

Ces formes permettent en particulier de visualiser l'anisotropie, voir par exemple sur la figure 2.4 des exemples de (bord de) formes de Wulff pour différentes anisotropies.

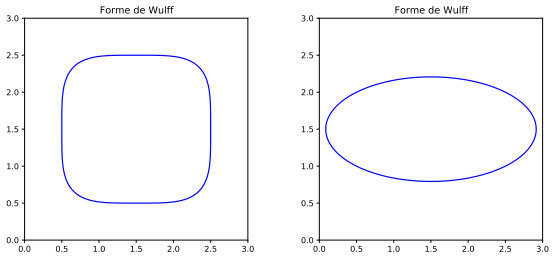


FIGURE 2.4 – Bord d'une forme de Wulff (ici $R = 1$ et $x_0 = (0, 0)$) pour les anisotropies $\gamma_2(x) = |x|_{l^4}$ (à gauche) et $\gamma_2(x) = \sqrt{2x_1^2 + x_2^2/2}$ (à droite).

Le mouvement par courbure moyenne anisotrope correspond alors au flot de gradient L^2 de l'énergie (2.1.7). En dimension 2 (d'espace ambiant), la vitesse normale à l'interface vérifie la relation explicite

$$V_n = \psi\,(\psi'' + \psi)\,\kappa,$$

où κ est la courbure de l'interface unidimensionnelle et ψ intervient dans la décomposition polaire de $\gamma(x) = |x|\psi(\arg(x))$ avec $\arg(x)$, l'argument de x.

Nous présentons en figure 2.5 deux exemples de mouvement par courbure anisotrope d'un cercle pour les anisotropies $\gamma_1(x) = |x|_{l^4}$ et $\gamma_2(x) = \sqrt{\frac{1}{2}x_1^2 + 2x_2^2}$.

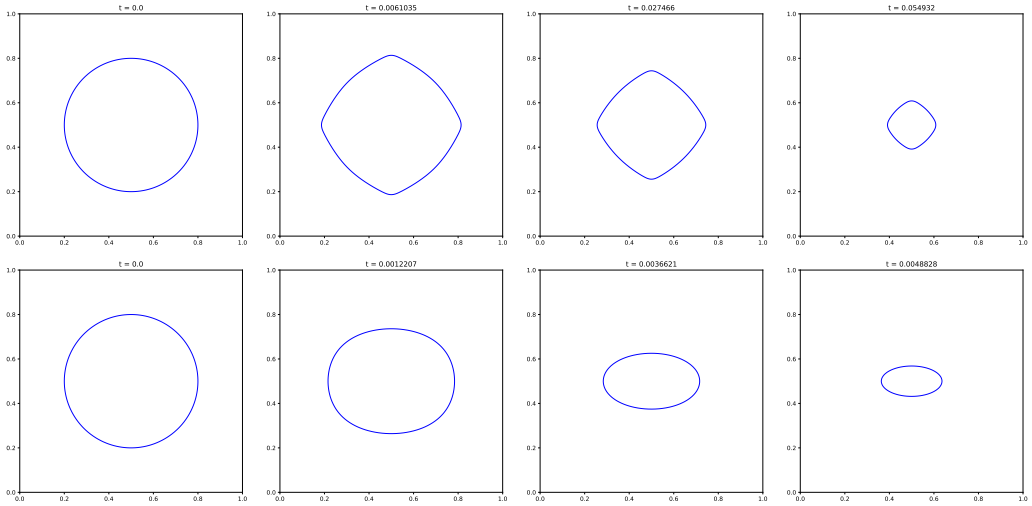


FIGURE 2.5 – Évolution d'un cercle pour les anisotropies $\gamma_1(\xi) = |\xi|_{l^4}$ (première ligne) et $\gamma_2(\xi) = \sqrt{\frac{1}{2}\xi_1^2 + 2\xi_2^2}$ (seconde ligne). Chaque ligne montre l'interface $\partial\Omega(t)$ à différents instants t.

2.2 Flot de diffusion de surface

Le flot de diffusion de surface peut être intéressant si l'on s'intéresse à la minimisation du périmètre tout en assurant une conservation locale du volume. Il correspond au mouvement géométrique dont la vitesse à l'interface $\partial\Omega(t)$ s'identifie à l'image de la courbure moyenne par l'opérateur de Laplace-Beltrami :

$$V_n(t) = \Delta_{\partial\Omega(t)} H(t) \tag{2.2.1}$$

où Δ_Γ désigne l'opérateur de Laplace-Beltrami défini sur Γ. La figure 2.6 montre l'évolution d'une fleur par diffusion de surface.

Ce mouvement a été introduit par Mullins [262, 114] pour modéliser la dynamique d'un cristal lorsque tout le transport de masse se fait par diffusion le long de la surface du cristal sous l'effet de la courbure.

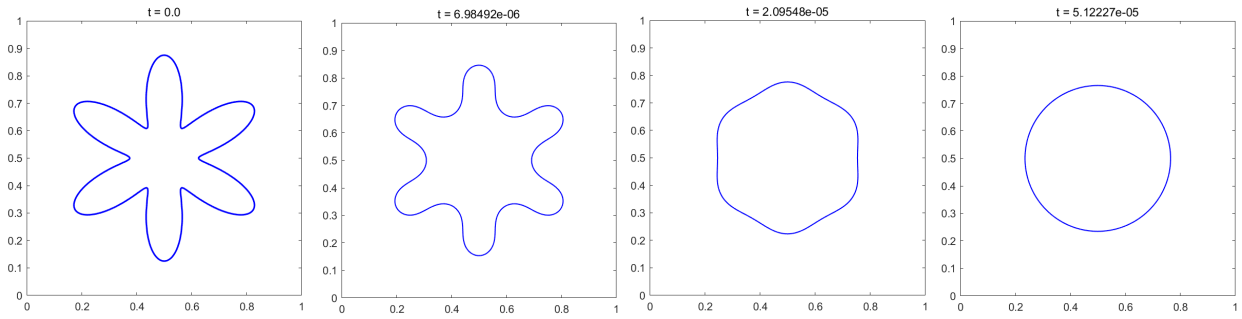


FIGURE 2.6 – Flot de diffusion de surface à partir d'une forme initiale de fleur. Chaque image représente l'interface $\partial\Omega(t)$ à différents instants t.

Depuis, un grand nombre d'études ont été menées tant d'un point de vu physique que mathématique [114, 81]. Dans [323] Taylor et Cahn identifient la dynamique (2.2.1) comme le flot de gradient pour la métrique H^{-1} du périmètre (2.1.2). Cependant, contrairement au mouvement par courbure moyenne, c'est un flot d'ordre 4 qui ne vérifie pas de principe d'inclusion et les propriétés d'unicité sont donc plus difficiles à prouver. L'existence, la régularité et les propriétés qualitatives ont été étudiés par plusieurs auteurs [249, 144, 145, 140, 306, 218]. En particulier, le flot de diffusion de surface permet une conservation locale du volume au cours de l'évolution. En effet :

$$\frac{d}{dt}\mathrm{Vol}(\Omega(t)) = \int_{\partial\Omega(t)} V(t)\, d\mathcal{H}^{d-1} = \int_{\partial\Omega(t)} \Delta_{\partial\Omega(t)} H(t)\, d\mathcal{H}^{d-1} = 0.$$

Dans notre contexte isotrope, les formes d'équilibre sont les sphères de $\mathbb{R}^d$.

Le flot de diffusion de surface s'étend tout naturellement au cadre multiphase en considérant cette fois-ci le flot de gradient H^{-1} du périmètre multiphase (2.1.5). La vitesse à l'interface Γ_{ij} est alors donnée par

$$V_{ij} = \sigma_{ij}\nu_{ij}\Delta_{\Gamma_{ij}(t)} H_{ij}(t).$$

Lorsqu'une anisotropie γ est imposée, le flot de diffusion de surface anisotrope (biphasique) est obtenu comme le flot de gradient H^{-1} du périmètre anisotrope (2.1.7) et, en dimension deux, la vitesse à l'interface est explicite [323] en fonction de la courbure κ :

$$V_n = \psi\partial_s\left(\psi\partial_s\left((\psi'' + \psi)\kappa\right)\right) \tag{2.2.2}$$

où ψ intervient dans la décomposition $\gamma(x) = |x|\psi(\arg(x))$ et ∂_s correspond à la dérivation suivant la longueur d'arc s.

2.3 Flot de Willmore

Le dernier flot qui nous intéresse dans ce manuscrit est le flot de Willmore défini comme le flot de gradient L^2 de l'énergie de Willmore

$$\mathcal{W}(\Omega) = \frac{1}{2}\int_{\partial\Omega} H^2\, d\mathcal{H}^{d-1}. \tag{2.3.1}$$

Il correspond au mouvement géométrique dont la vitesse normale à l'interface $\partial\Omega(t)$ (supposée régulière) est donnée par

$$V_n(t) = \Delta_{\partial\Omega(t)} H(t) + \|A(t)\|^2 H(t) - \frac{1}{2}H(t)^3, \tag{2.3.2}$$

où $\|A(t)\|^2 = \sum_{i=1}^{d-1}\kappa_i(t)^2$ désigne le carré de la norme de Frobenius de la seconde forme fondamentale $A(t)$ de $\partial\Omega(t)$ et les $\kappa_i(t)$ correspondent aux courbures principales de $\partial\Omega(t)$. Pour la dérivation de (2.3.2) à partir de (2.3.1) on pourra se référer à [215].

En dimension 2, l'énergie de Willmore d'une courbe coïncide avec l'énergie de Bernoulli-Euler et son flot (2.3.2) correspond à l'évolution selon la vitesse normale

$$V_n(t) = \partial_s\kappa(t) + \frac{1}{2}\kappa(t)^3,$$

où $\kappa(t)$ est la courbure de $\partial\Omega(t)$ et ∂_s est la dérivation suivant la longueur d'arc s.

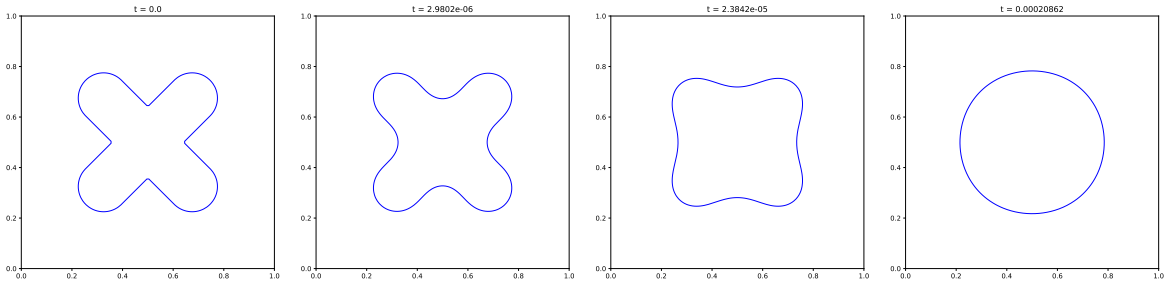


FIGURE 2.7 – Exemple de flot de Willmore en dimension 2

Plusieurs problèmes d'optimisation de forme en biologie, en imagerie et en géométrie discrète font intervenir cette énergie pour les propriétés intéressantes de ses points critiques. Cette dernière n'étant pas semi-continue inférieurement pour la topologie L^1 (dans le cas non régulier), sa minimisation sous contrainte reste particulièrement délicate à traiter et il est assez difficile de caractériser précisément la structure des minimiseurs. L'étude du flot (2.3.2) est elle aussi assez délicate à traiter. L'existence et la régularité des solutions de (2.3.2) ont été étudiées dans [133, 214, 213]. En particulier, l'existence en temps long des courbes simples en dimension deux et la convergence vers une sphère de formes initiales dont l'énergie de Willmore est petite ont pu être démontrées. Au contraire, pour des formes initiales dont l'énergie est élevée, des singularités peuvent apparaître [133].

Chapitre 3

Méthodes numériques pour la simulation de flots géométriques

L'élaboration de méthodes numériques pour la simulation de flots géométriques dépend fortement de la façon dont est représentée l'interface. Ainsi, la plupart des méthodes numériques actuelles sont basées sur des modèles mathématiques qui représentent l'interface soit de manière explicite et directe avec une paramétrisation par exemple ou de manière implicite et indirecte au moyen de variables ou de fonctions auxiliaires. En suivant ces deux approches, un grand nombre de modèles mathématiques ont été introduits pour traiter les problèmes d'évolutions d'interfaces géométriques. Ces modèles sont principalement basés sur les quatre approches suivantes :

— L'approche paramétrique qui décrit l'interface ponctuellement en utilisant une paramétrisation de celle-ci.
— La formulation *level-set* où l'interface est décrite comme la ligne de niveau d'une fonction auxiliaire.
— Une approche basée sur la diffusion de l'interface avec les algorithmes de type convolution-seuillage ou convolution-redistanciation.
— La formulation champ de phase où l'évolution de l'interface est approchée par celle d'une interface diffuse décrite comme la zone de transition d'une fonction régulière.

L'approche paramétrique est une méthode de type *front tracking* qui réalise un suivi explicite de l'interface au moyen d'une paramétrisation alors que les autres formulations citées ci-dessus sont de type *front capturing* qui capturent l'interface en la caractérisant implicitement par l'intermédiaire de fonctions auxiliaires. Chacune de ces approches a ses avantages et ses inconvénients, et le choix de la formulation utilisée dépend d'un choix de priorité parmi de nombreux critères tels que la simplicité, le coût et la robustesse numérique, le degré de précision souhaité, la fidélité de la physique décrite,etc.

Il existe d'autres approches basées sur la notion de varifold [326], ou les notions de barrière ou de mouvement minimisant introduites par De Giorgi. Ces approches offrent un cadre théorique très intéressant pour l'étude des flots, notamment lorsque des singularités apparaissent, là où le flot n'est plus défini de manière classique. Il est en effet important de noter que les flots géométriques développent assez souvent des singularités en temps fini, même lorsque la condition initiale est régulière. Lorsque des irrégularités apparaissent, une formulation classique du flot n'est donc plus possible et sa définition après ces irrégularités dépend du modèle mathématique utilisé. Sur ce point, la formulation level-set rejoint les dernières approches citées puisque l'évolution de l'interface est décrite à travers les solutions de viscosité d'une équation de type Hamilton-Jacobi qui permettent d'assurer l'existence de solutions après l'apparition d'irrégularités.

Pour les varifolds, les barrières ou les mouvements minimisants, les méthodes numériques de ces formulations ne sont malheureusement pas aussi développées que pour les approches précédentes. Pour cette raison nous ne présenterons que les approches paramétrique, level-set, convolution-seuillage et champ de phase dans ce qui suit.

Nous rappelons que l'objectif est de décrire numériquement l'évolution d'interfaces $\partial\Omega(t)$ dont les déplacements au cours du temps t sont prescrits par la vitesse normale $V_n(t)$. Pour les flots qui nous intéressent, la vitesse normale V_n dépend de la position mais aussi de la normale, de la courbure et de ses dérivées successives. Pour être en accord avec la définition de la vitesse normale donnée pour les flots géométriques, nous notons $V_n(t)$ la vitesse normale scalaire et $V(t) = V_n(t)n(t)$ le vecteur vitesse normal.

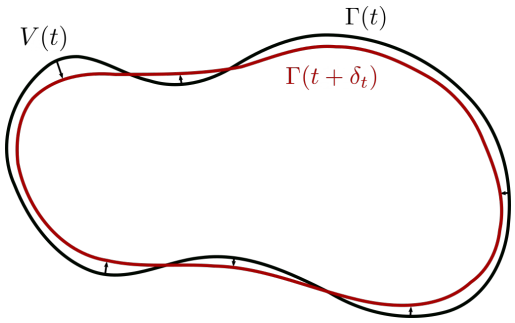


FIGURE 3.1 – Évolution d'une interface $\partial\Omega(t)$ suivant une vitesse $V_n(t)$ après un pas de temps δ_t.

3.1 Approche paramétrique

L'approche paramétrique consiste à modéliser l'évolution de l'interface $\partial\Omega$ à l'aide d'une paramétrisation locale de celle-ci $X(s,t) : U \times [0,T] \to \partial\Omega(t)$ où U est un référentiel fixé. En pratique, on peut supposer que ce référentiel est $\mathbb{S}^{d-1}$ et travailler avec des conditions aux bords périodiques. Grâce à la paramétrisation, les quantités géométriques comme la normale $n(s,t)$ et la courbure moyenne $H(s,t)$ peuvent s'exprimer en fonctions des dérivées de la position $X(s,t)$. En particulier, la loi d'évolution de $\partial\Omega(t)$ se traduit par l'équation du flot

$$\partial_t X(s,t) = V(s,t) = V_n(s,t)n(s,t) \tag{3.1.1}$$

où $n(s,t)$ est la normale unitaire à $\partial\Omega(t) = X(U,t)$ en position $X(s,t)$.

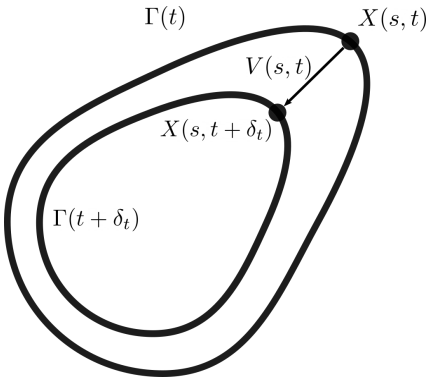


FIGURE 3.2 – Représentation paramétrique de l'interface et son évolution sur un pas de temps δ_t.

Avec cette description, il n'est pas nécessaire que l'interface $\partial\Omega(t)$ soit le bord d'un domaine. En particulier, la notion d'orientation ne pose pas réellement de problème d'un point de vue géométrique et ne dépend que de la paramétrisation utilisée. Il est donc aussi possible d'avoir différentes évolutions suivant la paramétrisation lors de l'apparition d'une ou plusieurs intersections. Ce qui rend en particulier cette approche sensible aux changements de topologies car il est nécessaire de redéfinir une nouvelle paramétrisation après l'apparition d'un tel changement. D'un point de vue numérique, cela pose un certain nombre de difficultés liées, d'une part au calcul des quantités géométriques qui pourrait perdre en consistance au cours de l'évolution et d'autre part à la redistribution des points du maillage qui est nécessaire après un changement de topologie. Il est donc indispensable de disposer d'une discrétisation fine de (3.1.1) qui puisse répartir convenablement les points du maillage au risque de rencontrer des instabilités même lorsque l'évolution est régulière.

Les premières méthodes numériques suffisamment robustes pour l'approche paramétrique sont dues aux travaux de Dziuk [130] dans le cadre du mouvement par courbure moyenne où il introduit une méthode numérique basée sur le fait que ce flot peut être caractérisé (dans le cas régulier) par une équation de la chaleur non linéaire,

$$\partial_t X(s,t) = \Delta_{\partial\Omega(t)} X(s,t) \tag{3.1.2}$$

où $\Delta_{\partial\Omega(t)}$ est l'opérateur de Laplace Beltrami sur $\partial\Omega(t)$.

Son approche consiste alors à déterminer l'évolution de l'interface en s'appuyant sur une formulation faible de (3.1.2) et en utilisant une base d'éléments finis sur le maillage de l'interface. Les points du maillage déterminent alors l'évolution de l'interface. Depuis, d'autres techniques basées sur une approche similaire à celle de Dziuk ont été développées pour la simulation du flot de courbure moyenne comme les méthodes qui introduisent une vitesse tangentielle artificielle [118] en utilisant une reparamétrisation de l'équation (3.1.2) sous forme sans divergence, les méthodes [24, 26, 28], initiées par Barrett, Garcke et Nürnberg, basées sur différentes approches variationnelles et différents choix de fonctions tests ; ou encore les méthodes qui utilisent l'astuce de reparamétrage de DeTurck proposées par Elliott et Fritz [242].

La difficulté de l'approche paramétrique est la gestion des points du maillage qui demande une attention particulière à chaque itération et surtout lors de l'apparition d'irrégularités comme des changements de topologie. Les méthodes précédentes permettent de gérer certaines irrégularités mais leurs généralisations en dimension supérieure à deux ou leurs adaptations à d'autres flots géométriques reste encore difficiles. En particulier, la gestion de termes de forçage ou de contraintes demande des adaptations qui ne sont pas toujours très simples à gérer [282]. Par ailleurs, la gestion des points triples est encore une question d'actualité, voir [282] dans le cas de la dimension deux. Concernant les flots d'ordre plus élevé comme la diffusion de surface et le flot de Willmore un certain nombre d'études ont été menées principalement par John W.Barrette et d'autres auteurs dans [26, 24, 27, 29, 22]. Encore une fois, la simulation numérique de ces flots peut mener à des effets indésirables du maillage qui sont dus au côté fortement non linéaire des équations sous-jacentes. Pour remédier à ces problèmes, les auteurs introduisent des techniques spécifiques pour maintenir les bonnes propriétés du maillage après chaque itération.

3.2 Méthode level-set

Cette approche est très populaire pour ces aspects théoriques mais aussi numériques. Elle a été introduite par Osher et Sethian [271, 269, 270] pour les mouvements de domaines dont la vitesse normale à l'interface dépend de la courbure. L'idée principale de la formulation level-set est de représenter de manière implicite un domaine $\Omega(t)$ à travers les lignes de niveaux d'une fonction auxiliaire $\varphi : \mathbb{R}^d \times [0,T] \to \mathbb{R}$ appelée fonction level-set,

$$\begin{cases} \varphi(x,t) < 0, \ x \in \Omega(t) \\ \varphi(x,t) = 0, \ x \in \partial\Omega(t) \\ \varphi(x,t) > 0, \ x \in \mathbb{R}^d \setminus \Omega(t) \end{cases}$$

L'interface est alors déterminée à chaque instant t comme la ligne de niveau 0 de φ,

$$\partial\Omega(t) = \left\{ x \in \mathbb{R}^d, \varphi(x,t) = 0 \right\}$$

et son évolution est déterminée par l'évolution en temps de φ. Par opposition à l'approche paramétrique, l'évolution de l'interface $\partial\Omega(t)$ est ici déterminée de manière globale à travers les lignes de niveau de φ. Ce caractère global montre aussi que la formulation level-set est adaptée aux changements de topologie (décrits de manière intrinsèque grâce à φ) et se généralise facilement en dimension supérieure. Notons enfin que cette approche nous donne un moyen direct de savoir si un point est à l'intérieur ou à l'extérieur de l'interface, grâce au signe de φ en ce point.

Bien sûr, le choix de la fonction level-set n'est pas unique et l'exemple classique souvent utilisé est la fonction distance signée $\varphi(x,t) = d(x, \partial\Omega(t))$. Pour un domaine Ω de $\mathbb{R}^d$, cette fonction d est définie comme suit

$$d(x,\Omega) = \begin{cases} \mathrm{dist}(x,\partial\Omega) \text{ si } x \in \Omega, \\ -\mathrm{dist}(x,\partial\Omega) \text{ sinon} \end{cases}$$

Comme nous le verrons tout au long de ce manuscrit, la fonction distance signée [33, 12] joue un rôle important dans l'étude des flots géométriques et de leurs approximations. Ceci vient en particulier du fait que cette fonction permet de capturer localement la géométrie de l'interface et que ses variations dans un voisinage de l'interface ne sont pas très abruptes. D'un point de vue numérique, cette dernière propriété est très importante en level-set car les méthodes numériques de cette approche sont impactées en grande partie par la précision du calcul de $|\nabla\varphi|$.

Finalement, l'objectif est de traduire l'évolution de $\Gamma(t)$ par une équation en φ. Formellement, il s'agit de différentier l'équation $\varphi(x,t) = 0$ par rapport à t en supposant que $x = x(t)$ est une fonction implicite de t. On obtient alors

$$\partial_t \varphi + \nabla\varphi \cdot V = 0 \tag{3.2.1}$$

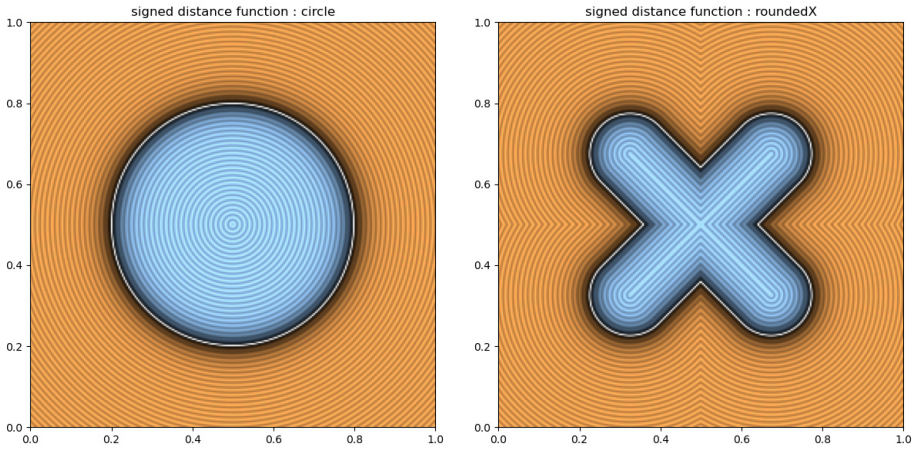


FIGURE 3.3 – Représentation de la fonction distance signée d'un cercle (à gauche) et d'une forme en X arrondie (à droite). La ligne de niveau 0 est représentée en blanc. L'intérieur et l'extérieur sont représentés respectivement par les couleurs bleu et marron.

où $\frac{d}{dt}x = V$ est le vecteur vitesse à l'interface. Pour les flots qui nous intéressent, on considère que les déplacements se font que dans la direction normale à l'interface et on écrit $V(t) = V_n(t)n(t)$ où $n(t)$ est la normale à $\partial\Omega(t)$. On obtient ainsi une équation de transport dont la condition initiale φ_0 est telle que $\Gamma(0) = \{x \in \mathbb{R}^d, \varphi_0(x) = 0\}$.

Il est important de noter qu'ici l'équation (3.2.1) n'est définie que sur $\Gamma(t)$ mais on peut montrer que celle-ci est en faite valable sur toutes les lignes de niveaux et donc sur $\mathbb{R}^d$. Lorsque l'interface $\Gamma(t)$ est suffisamment régulière, φ est deux fois différentiable au voisinage de $\Gamma(t)$ et on vérifie que

$$n = \frac{\nabla\varphi}{|\nabla\varphi|} \quad \text{et} \quad H = -\operatorname{div}(n),$$

sur $\Gamma(t)$. En particulier, lorsque la vitesse normale $V_n = V_n(x, n, H)$ dépend de la normale et de la courbure, l'équation (3.2.1) vérifiée par φ est en fait une équation de type Hamilton-Jacobi qui s'écrit

$$\begin{cases} \partial_t\varphi = \mathcal{H}(x, \nabla\varphi, \nabla^2\varphi)|\nabla\varphi| \\ \varphi(t=0) = \varphi_0 \end{cases} \tag{3.2.2}$$

où l'on a posé $\mathcal{H}(x, \nabla\varphi, \nabla^2\varphi) = -V_n(x, n, H)$ que l'on appelle Hamiltonien.

Les équations de type Hamilton-Jacobi sont des équations paraboliques fortement dégénérées dont les solutions peuvent développer des singularités même lorsque la donnée initiale est régulière. Il n'y a d'ailleurs pas unicité des solutions faibles de façon générale. C'est pour donner un sens particulier aux solutions de ces équations que Crandall et Lions ont introduit en 1983 la notion de solutions de viscosité [108] (voir aussi [11] pour une adaptation aux mouvements géométriques). Une caractéristique importante de cette notion est qu'elle permet, sous certaines conditions, l'extension des propriétés de type principe du maximum (qui sont classiques pour des solutions régulières d'équations paraboliques ou elliptiques) à des solutions peu régulières. Sous certaines hypothèses sur le hamiltonien et la condition initiale, on peut ainsi montrer l'existence et l'unicité des solutions de viscosité en se basant sur ces propriétés. Pour revenir aux équations de type (3.2.2), une étude a été effectuée par Chen, Giga et Goto [97] et par Evans et Spruck [157, 155, 154, 153]. Finalement, la notion de solutions de viscosité donne un sens aux solutions des équations (3.2.2) et permet en particulier de définir le mouvement d'interfaces dans un cadre assez général, même lorsque des irrégularités apparaissent.

Quelques aspects numériques de la méthode level-set

Grâce à la formulation level-set, le traitement des singularités ou des changements de topologie se fait tout naturellement en résolvant une équation d'Hamilton-Jacobi. Le prix à payer d'un point de vue numérique est de travailler avec une dimension supplémentaire : on ne travaille plus sur une zone localisée à l'interface comme pour la méthode paramétrique mais sur tout l'espace. Autrement dit, on stocke toutes les lignes de niveau de la fonction level-set φ alors qu'en pratique on s'intéresse uniquement à ce qui se passe au voisinage de l'interface $\{\varphi = 0\}$. Une approche plus économe est la méthode dite "narrow band"

introduite par Chopp [100] qui consiste à manipuler la fonction ϕ uniquement sur un voisinage tubulaire de l'interface, suffisamment large pour estimer précisément les quantités dans l'équation d'Hamilton-Jacobi.

Le choix de la fonction initiale.
Nous avons vu que théoriquement il n'y avait pas de choix de priorité sur la fonction level-set initiale φ_0 définissant l'interface initiale $\partial\Omega_0$ comme ligne de niveau 0. On se rend compte que ce n'est pas le cas lors de la résolution numérique de l'équation d'Hamilton-Jacobi (3.2.2). En effet, des variations trop fortes ou trop faibles de φ_0 autour de l'interface peuvent causer des instabilités numériques et la précision de suivi de l'interface se retrouve fortement dégradée. Cette dernière remarque est en fait vraie tout au long de la résolution numérique de (3.2.2). Un bon choix pour φ_0 est de prendre la fonction distance signée d_0 à $\partial\Omega_0$ dont on sait que les variations sont contrôlées autour de l'interface puisqu'elle vérifie une équation eikonale $|\nabla d_0| = 1$ au voisinage de l'interface. Pour calculer (ou approcher) une telle fonction on utilise la méthode de *Fast Marching* [303] ou de *Fast Sweeping* [345] ou encore plus récemment à l'aide de réseaux de neurones (voir par exemple [316]).

Redistanciation.
Nous constatons qu'il est important que la solution φ de (3.2.2) reste « proche » d'une fonction distance signée. Malheureusement, bien que la condition initiale soit une distance signée, la condition $|\nabla\varphi| = 1$ n'est pas toujours préservée au cours du temps. Il est donc d'usage de redéfinir périodiquement $\varphi(t,\cdot)$ comme une distance signée à $\Omega(t)$. Dans ce cas, le calcul est facilité par le fait que nous connaissons déjà la fonction fonction level-set contrairement à la phase d'initialisation. Pour cela plusieurs approches sont possibles. L'une des plus connues consiste à redresser φ à un instant t^n en une distance signée en résolvant l'équation d'Hamilton-Jacobi de redistanciation :

$$\partial_t\psi + \mathrm{sign}(\varphi(t^n))\,(|\nabla\psi| - 1) = 0, \tag{3.2.3}$$

avec la condition initiale $\psi(t=0,x) = \varphi(t^n,x)$. On cherche alors un état stationnaire $\psi(t^*)$ de cette équation, de sorte que $|\nabla\psi| \simeq 1$ au voisinage de $\{\psi = 0\}$. En pratique cela se fait en quelques itérations grâce à l'hyperbolicité de l'équation (3.2.3). Enfin, on remplace $\varphi(t^n)$ par $\psi(t^*)$ puis on résout (3.2.2) avec cette nouvelle condition initiale.

L'inconvénient de cette approche est qu'elle rajoute une deuxième équation à résoudre et la résolution de celle-ci peut introduire des déplacements de l'interface [143] et le volume de l'intérieur $\{\varphi = 0\}$ n'est pas toujours préservé (ce qui n'est pas lié aux méthodes numériques mais plutôt au côté non conservatif de l'équation (3.2.3)).

En pratique, on utilise couramment des schémas Essentiellement Non Oscillants (ENO) et Weighted ENO (WENO) pour résoudre l'équation d'Hamilton-Jacobi (3.2.2) et l'équation de redistanciation (3.2.3) où le terme discontinue $\mathrm{sign}(\psi_0)$ est souvent régularisé. Ce sont des schémas d'ordre élevé qui permettent de résoudre avec une grande précision les équations d'Hamilton-Jacobi.

Bilan de la formulation level-set

L'approche level-set fournit un formalisme pratique pour représenter les flots géométriques de domaines en toute dimension. À la différence de l'approche paramétrique, elle peut gérer les changements topologiques et peut être définie rigoureusement au-delà des singularités en utilisant la théorie des solutions de viscosité pour les équations de Hamilton-Jacobi. Il existe cependant plusieurs difficultés concernant le traitement numérique de cette approche. L'équation de Hamilton-Jacobi est non linéaire et fortement dégénérée, donc difficile à intégrer numériquement et des méthodes délicates sont nécessaires pour préserver certaines propriétés nécessaires de la fonction level-set φ. De plus, c'est une approche conçue pour représenter l'évolution d'interfaces orientées et il n'existe, à notre connaissance, aucune approche level-set qui puisse traiter l'évolution d'interfaces non orientables à ce jour. Notons enfin que c'est une approche qui se généralise difficilement au cadre multiphase. Ceci vient sans doute du fait que c'est un formalise qui s'interprète difficilement comme un problème variationnel en général.

3.3 Les algorithmes de type convolution-seuillage.

On s'intéresse maintenant aux algorithmes de type convolution-seuillage qui permettent de générer des flots géométriques discrets [46, 200, 150]. Ces algorithmes, très populaires pour leur simplicité, sont essentiellement des schémas alternant une opération de convolution et une opération de seuillage. La genèse de

ce type d'approche remonte à l'algorithme de Bence, Merriman et Osher [46] pour la simulation du mouvement par courbure moyenne dont nous rappelons le principe ici.

L'algorithme de Bence, Merriman et Osher.

Étant donné un domaine Ω_0 et un pas de temps $\delta_t > 0$, on génère une suite d'interfaces $(\Omega)_{n\geq 0}$ décrivant le mouvement par courbure moyenne partant de Ω_0 en suivant les étapes suivantes. En partant d'un domaine Ω_n, on définit Ω_{n+1} comme suit

$$\Omega_{n+1} = \left\{ G_{\delta_t} * \mathbb{1}_{\Omega_n} \geq \frac{1}{2} \right\}$$

où $G_{\delta_t}(x) = \dfrac{1}{(4\pi\delta_t)^{d/2}} e^{-\frac{|x|^2}{4\delta_t}}$.

La figure 3.4 illustre les étapes précédentes en partant d'une forme en X arrondie.

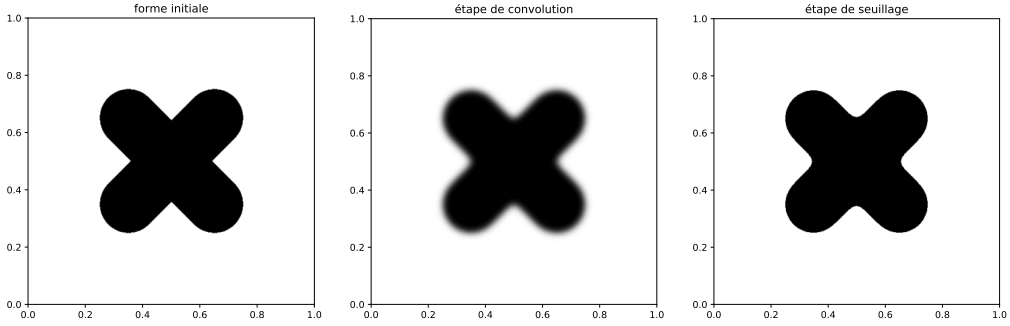


FIGURE 3.4 – Illustration de la méthode de convolution-seuillage appliquée à une forme en X arrondie.

Le lien avec le mouvement par courbure moyenne vient du fait qu'à l'interface, diffuser revient à déplacer l'interface suivant la vitesse normale $V = H$.

Plus précisément, il est possible de montrer, voir [179, 150], que la ligne de niveau $1/2$ de $\mathbb{1}_{\Omega_n}$ après diffusion avec le noyau de convolution G_{δ_t} s'est déplacée dans la direction normale à l'interface d'une distance y vérifiant

$$y = \kappa\delta_t - \frac{1}{2}\mathcal{W}\delta_t^2 + \mathcal{O}(\delta_t^3) \tag{3.3.1}$$

où κ est la courbure à l'interface $\Gamma = \partial\Omega$ et $\mathcal{W} = \Delta_{\partial\Omega(t)}\kappa + \frac{1}{2}\kappa^3$ est le terme de Willmore.

La méthode de Bence Merriman et Osher (BMO) a l'avantage d'être très simple à mettre en œuvre et le coût algorithmique est très faible puisque celui-ci ne dépend essentiellement que de la convolution qui peut être traitée efficacement avec la transformée de Fourier rapide. Cependant, elle nécessite que la discrétisation en espace soit suffisamment fine pour éviter que certaines zones de l'interface restent figées durant l'évolution. En effet, si la discrétisation n'est pas assez fine, il peut arriver que l'interface se retrouve « figée » sur certaines mailles lorsque le déplacement à l'interface est trop « faible » par rapport à la taille du maillage. Dans ce cas l'étape de seuillage renvoie la même indicatrice qu'à l'étape précédente et il n'y a plus de déplacement [251]. Pour y remédier, on devrait avoir un maillage adaptatif mais on perdrait en efficacité numérique.

Une approche intéressante proposée par Esedoglu, Ruuth et Tsai [149] est de diffuser la fonction distance signée qui a l'avantage de capturer la géométrie locale à l'interface [33, 12]. En notant d la fonction distance signée de Ω, on obtient alors le développement suivant en dimension deux :

$$G_{\delta_t} * d = d + \kappa\delta_t + \frac{1}{2}\left(\Delta_\Gamma\kappa + \kappa^3\right)\delta_t^2 + \mathcal{O}(\delta_t^3). \tag{3.3.2}$$

En remplaçant l'étape de seuillage par une redistanciation, on retrouve le mouvement par courbure moyenne. Pour cette étape de redistanciation, on utilise un algorithme de fast marching ou la résolution d'une équation d'Hamilton-Jacobi pour recalculer la fonction distance signée à chaque itération.

Depuis la publication de la méthode BMO, plusieurs améliorations et généralisations ont été développées pour un grand nombre de flots géométriques [150, 252, 147]. Une série de travaux a été menée par Esedoglu et ses co-auteurs dans le cas du mouvement par courbure [146, 141, 297, 147]. En particulier, dans

[147], les auteurs reformulent l'algorithme de BMO comme un problème variationnel. Cette nouvelle approche variationnelle leur a permis de généraliser l'algorithme de BMO au cas du mouvement par courbure moyenne anisotrope [146, 141] mais aussi au cadre multiphase [146, 297] avec un choix arbitraire de tensions de surface, tout en préservant le côté variationnel. Leur approche est intéressante car d'une part les algorithmes qu'ils utilisent restent très simples à implémenter et, d'autre part, l'anisotropie est totalement caractérisée par le choix du noyau de convolution utilisé.

Les flots comme la diffusion de surface ou le flot de Willmore ont déjà été étudiés dans [150, 179] et très récemment pour le flot Willmore dans [252] où les auteurs utilisent la fonction distance signée à la place d'une fonction caractéristique. Leurs approches s'appuient sur le fait que les termes impliquant la courbure comme le Laplace Beltrami de la courbure ou le terme de Willmore sont présents dans le développement en temps à l'étape de convolution (3.3.1). En utilisant des techniques d'extrapolation, on peut ainsi générer des flots d'ordre supérieur comme le flot de diffusion de surface ou le flot de Willmore. Les méthodes numériques pour ces flots d'ordre supérieur restent relativement simples à mettre en œuvre mais ne sont plus variationnelles et elles impliquent aussi certaines restrictions sur le pas de temps.

Bilan de la méthode de convolution-seuillage

Les méthodes de type convolution-seuillage sont très simples à comprendre et à implémenter. Elles permettent d'approcher un grand nombre de flots géométriques tels que le mouvement par courbure moyenne ou le flot de diffusion de surface. C'est une approche qui est maintenant très bien comprise dans le cas du flot par courbure moyenne où le côté variationnel de la méthode permet la généralisation au cas multiphase mais aussi au cas anisotrope où l'anisotropie est entièrement caractérisée par le noyau de convolution. Cependant, elle demande une attention particulière sur le pas d'espace et le pas de temps : le pas d'espace doit être suffisamment petit pour pouvoir traiter des flots dont les déplacements à l'interface sont faibles, le pas de temps doit aussi être choisi minutieusement pour générer des flots d'ordre élevé. La gestion des contraintes comme celle du volume doit aussi être gérée avec précaution car, très souvent, les méthodes associées impliquent d'avoir une bonne estimation d'objets géométriques comme la courbure [149, 252]. Des méthodes robustes doivent donc être utilisées pour estimer correctement la courbure.

Nous finissons cette section par la présentation de l'approche champ de phase.

3.4 La méthode champ de phase

L'approche champ de phase comme méthode d'approximation d'évolution d'interfaces remonte aux travaux de Van Der Waals [293]. Elle trouve ses origines en thermodynamique et en dynamique des transitions de phases, puis elle a été étendue pour un grand nombre d'applications.

Tout comme la méthode level-set, les méthodes champ de phase sont basées sur une description implicite de l'interface mais la grande différence vient de leur structure variationnelle où les énergies géométriques d'interface sont approchées par des énergies de champ de phase dont les flots de gradients permettent à leur tour d'approcher les flots géométriques considérés. L'avantage de cette structure variationnelle est qu'elle conduit aussi à des EDPs relativement simples à traiter numériquement en comparaison avec les équations d'Hamilton Jacobi des méthodes level set.

Dans le cas par exemple du mouvement par courbure moyenne, le périmètre,

$$P(\Omega) = \int_{\partial\Omega} d\mathcal{H}^{d-1},$$

est approché, au sens de la Γ-convergence [258, 94], par l'énergie de Cahn-Hilliard,

$$P_\varepsilon(u) = \int_{\mathbb{R}^d} \left(\frac{\varepsilon}{2}|\nabla u|^2 + \frac{1}{\varepsilon} W(u) \right) dx, \tag{3.4.1}$$

où ε est un paramètre caractérisant la précision de l'approximation et W désigne un potentiel. Dans cette thèse, on considérera toujours le potentiel double puits $W(s) = \frac{1}{2}s^2(1-s)^2$.

Cette énergie, initialement introduite dans les travaux de Van Der Waals [293], a été par la suite reprise par Cahn et Hilliard [82] pour modéliser l'interface entre deux liquides non miscibles. Formellement, le terme en $\frac{1}{\varepsilon}W(u)$ impose à u d'être une approximation d'une fonction caractéristique d'un domaine Ω dont la zone de transition est régularisée par le terme en $\frac{\varepsilon}{2}|\nabla u|$. Intuitivement, l'énergie (3.4.1) mesure la zone de

transition entre les phases 0 et 1 et donne une approximation du périmètre de Ω.

Le flot de courbure moyenne peut être alors approché en considérant le flot de gradient L^2 de l'énergie de Cahn-Hilliard (3.4.1) qui conduit, après un changement d'échelle, à la célèbre équation d'Allen-Cahn [11]

$$\partial_t u = \Delta u - \frac{1}{\varepsilon^2} W'(u). \tag{3.4.2}$$

La résolution de cette équation, associée à la condition initiale

$$u(x,0) = q\left(\frac{d(x,\Omega(0))}{\varepsilon}\right),$$

permet alors d'approcher le flot de courbure moyenne $t \mapsto \Omega(t)$ en considérant l'ensemble évoluant $t \mapsto \Omega_\varepsilon(t)$ défini par

$$\Omega_\varepsilon(t) = \{u_\varepsilon(\cdot,t) \geq 1/2\}.$$

Ici $d(\cdot,\Omega(0))$ est la fonction distance signée à $\Omega(0)$ et $q : \mathbb{R} \to [0,1]$ est le profil optimal et correspond au profile espéré de la zone de transition entre les phases 0 et 1. Pour le potentiel double puits, ce profil est explicit $q(s) = \frac{1}{2}\left(1 - \tanh\left(\frac{s}{2}\right)\right)$.

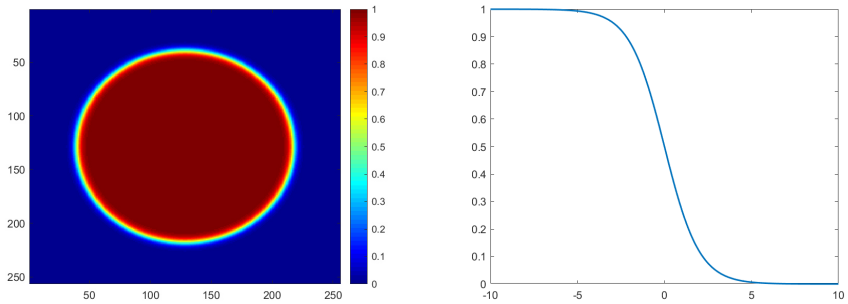


FIGURE 3.5 – Représentation champ de phase $q\left(\frac{d(x,D)}{\varepsilon}\right)$ d'un disque D (à gauche) en utilisant le potentiel q (graphe au centre) associé au potentiel double puits W (graphe à droite).

Plus précisément, un développement asymptotique de u_ε au voisinage de l'interface $\partial\Omega_\varepsilon(t) = \partial\{u_\varepsilon(\cdot,t) \leq 1/2\}$ [33] montre un développement de la forme

$$u_\varepsilon(x,t) = q\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2)$$

et la vitesse normale V_ε à l'interface $\partial\Omega_\varepsilon(t)$ vérifie

$$V_\varepsilon(t) = H_\varepsilon(t) + \mathcal{O}(\varepsilon^2).$$

Ici, le terme $H_\varepsilon(t)$ désigne la courbure moyenne à l'interface $\partial\Omega_\varepsilon(t)$. Au final, ce résultat suggère qu'à partir d'une résolution de l'équation d'Allen-Cahn, il est possible d'obtenir une approximation du mouvement par courbure moyenne avec une précision de l'ordre de ε^2 sur la position de l'interface.

L'efficacité de la méthode champ de phase dépend de l'erreur d'approximation liée au paramètre ε mais aussi de l'erreur de discrétisation. D'un point de vue numérique, l'équation d'Allen-Cahn ne pose pas de problème car c'est une équation de type réaction-diffusion dont les opérateurs peuvent être traités efficacement et indépendamment à l'aide d'une méthode de splitting. En pratique, on ne s'intéresse qu'à l'évolution d'interfaces dont le périmètre reste borné au cours du temps. Dans toute cette thèse, nous travaillerons donc dans un domaine de calcul de la forme $Q = [0,1]^d$ avec des conditions périodiques au bord. En particulier, dans ce contexte, l'opérateur de diffusion peut être traité efficacement avec la transformée de Fourier. Plus précisément, on utilise un schéma semi-implicite pour résoudre (3.4.2),

$$\frac{u^{n+1} - u^n}{\delta_t} = \Delta u^n - \frac{1}{\varepsilon^2} W'(u^n), \tag{3.4.3}$$

qui s'écrit aussi sous la forme

$$u^{n+1} = (Id - \delta_t \Delta)^{-1}\left(u^n - \frac{\delta_t}{\varepsilon^2} W'(u^n)\right),$$

où l'action $(Id - \delta_t \Delta)^{-1}$ est traitée dans l'espace de Fourier simplement par multiplication par le symbole $\sigma(\xi) = \dfrac{1}{1 + 4\pi^2 \delta_t |\xi|^2}$. Ce schéma correspond à une approximation d'ordre 1 en temps de (3.4.2) et le traitement explicite de l'opérateur de réaction nécessite la condition CFL $\delta_t \leq C\varepsilon^2$ où $C = \sup_{s\in[0,1]} W''(s)$.

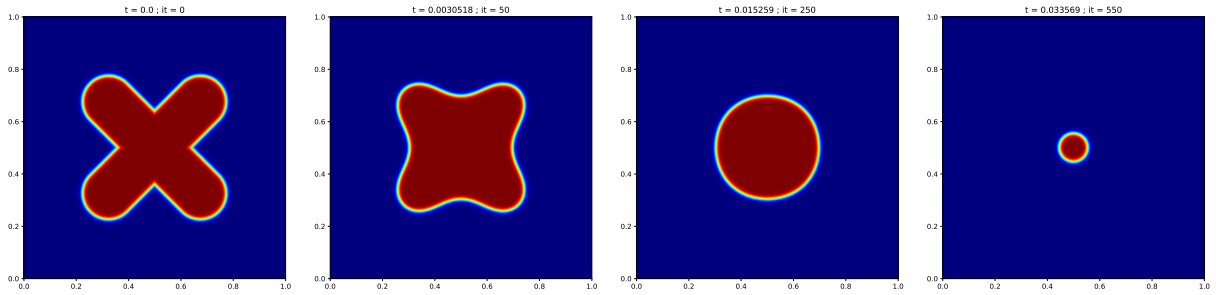


FIGURE 3.6 – Approximation champ de phase du mouvement par courbure moyenne d'une forme en X arrondie. Les images représentent la solution u^n de (3.4.3) à différentes itérations.

Pour s'affranchir de la condition CFL précédente, il est possible d'utiliser des techniques de relaxation sur l'énergie (3.4.1) [342, 158].

Pour conclure, nous venons d'illustrer l'efficacité des méthodes de champ de phase pour approcher des mouvements par courbure moyenne mais comme nous le verrons dans la suite de cette thèse, ce sera aussi le cas pour de nombreux flots géométriques d'interfaces, dans un contexte biphasique ou multiphasique, et en couplant parfois avec des contraintes de conservation de volume et des zones d'inclusion-exclusion.

3.5 Synthèse

L'approche paramétrique, la formulation level-set et les méthodes de type convolution seuillage reposent sur une description exacte, explicite ou implicite de l'interface en mouvement. Au contraire, la méthode champ de phase utilise une représentation diffuse de l'interface. L'approche paramétrique a l'avantage d'être intuitive, directe et n'impose pas que l'interface soit orientable. Cependant, la mise en œuvre numérique des modèles de cette approche est assez délicate et demande d'être minutieux sur la discrétisation, surtout pour des flots comme le flot de Willmore ou des flots avec contraintes (de volume ou d'inclusion). La formulation level-set donne un cadre pratique pour étudier le mouvement d'interfaces géométriques orientées grâce aux équations de Hamilton-Jacobi. Le caractère global de cette approche lui permet de gérer les changements de topologies et de définir les mouvements d'interfaces après l'apparition de singularités mais la résolution numérique des équations d'Hamilton-Jacobi est assez délicate et des méthodes fines sont nécessaires pour préserver certaines propriétés de la fonction level-set. Les méthodes de type convolution-seuillage fournissent un formalisme pratique et simple pour générer des approximations de flots géométriques de tout ordre. Les atouts d'une telle approche sont sa simplicité conceptuelle et son efficacité en termes de coût de calcul. Cependant, il est nécessaire en pratique d'avoir un maillage suffisamment raffiné au voisinage de l'interface pour éviter que le mouvement se retrouve « figé ». Tout comme la méthode level-set, la méthode champ de phase offre un cadre pratique pour étudier les flots géométriques d'interfaces, même lorsque des irrégularités apparaissent. C'est une formulation qui est suffisamment versatile pour traiter un grand nombre de flots géométriques : flots (biphasiques ou multiphasiques), avec ou sans contraintes/termes de forçage, flots anisotropes, flots d'ordre 4 et plus, etc. D'un point de vue numérique, de nombreuses techniques ont été développées ces dernières années pour traiter efficacement les modèles champ de phase. En pratique, on aboutit assez souvent à des méthodes simples et robustes.

Chapitre 4

Apprentissage machine et réseaux de neurones

L'apprentissage machine (*apprentissage automatique*) est une science qui vise à doter un algorithme de la capacité d'apprendre et de s'adapter à partir de données de natures très variées : sons, images, données statistiques, textes, etc. En particulier, l'objectif de l'apprentissage automatique est d'extraire des informations et des caractéristiques d'un ensemble de données en utilisant des algorithmes. Ce processus consiste en pratique à construire un *modèle de prédiction* et comporte deux grandes phases en général : la première phase consiste à estimer / ajuster un modèle à partir des données (c'est la *phase d'apprentissage* ou d'*entraînement*). Cette étape permet faire correspondre/imiter/reproduire un phénomène en présence d'informations de base (étiquettes, récompense). La deuxième phase correspond à la *phase de test* ou de production où le modèle estimé est appliqué à de nouvelles données jamais utilisées durant la première phase. Cette dernière étape sert à évaluer la consistance du modèle sur de nouvelles données et à estimer sa capacité à se *généraliser* à de nouvelles situations. Un algorithme d'apprentissage consiste donc à trouver un modèle qui enregistre les meilleurs scores de performance/prédiction (sur une base de données) parmi une classe d'applications (*classe des modèles*) que l'on choisit *a priori*. Cette classe peut être la classe des classificateurs dans le cas des problèmes de classification, la classe des polynômes dans un problème de régression polynomiale. En ce qui nous concerne, la classe qui nous intéresse est celle des réseaux de neurones.

Selon les informations disponibles durant le temps d'apprentissage et de test, on peut sommairement distinguer deux types d'apprentissage.

Apprentissage supervisé (*Supervised Learning*)

Lorsque les données étudiées sont étiquetées on parle d'*apprentissage supervisé* : les données d'entraînement (*observables*) $\{(x_i, y_i)\}_{1 \leq i \leq N}$ sont des paires de « question-réponse » où la donnée d'entrée x_i a été annotée par la sortie y_i par des experts. L'objectif de l'apprentissage supervisé consiste alors à trouver un modèle (une fonction ou une relation) qui puisse faire correspondre au mieux les entrées x des sorties y, sur la *base d'apprentissage* $\{(x_i, y_i)\}_{1 \leq i \leq N}$ mais aussi sur de nouvelles données. Ce domaine regroupe deux grandes familles de problèmes : la *classification* où l'ensemble des sorties est de nature discrète, et la *régression* où l'ensemble des sorties est de nature continue.

C'est cette approche qui nous intéresse dans ce manuscrit et que nous développerons plus tard dans le contexte des réseaux de neurones.

Apprentissage non supervisé (*Unsupervised Learning*)

L'*apprentissage non supervisé* s'intéresse à la structure sous-jacente de données non étiquetées. Son objectif est d'apprendre, sans superviseur, à extraire des classes ou groupes possédant des caractéristiques communes. Il se distingue donc de l'apprentissage supervisé par le fait qu'il n'a aucun *a priori* sur les classes (étiquettes) des entrées. En ce sens, les algorithmes d'apprentissage non supervisé permettent d'exécuter des tâches plus complexes que l'apprentissage supervisé mais peuvent aussi être très imprévisibles. Puisque les données ne sont pas labellisées, l'algorithme non supervisé ne peut garantir avec grande certitude un score de réussite.

Cette discipline regroupe les techniques de regroupement (*clustering*), de réduction de la dimension (*dimensionality reduction*), d'analyse en composantes principales (*Principal Component Analysis*, PCA), de décomposition en valeurs singulières (*Singular Value Decomposition*, SVD), générative (principalement avec les réseaux adversaires génératifs *Generative Adversarial Networks* , GAN [173]), etc. (voir par exemple [172, 76]).

4.1 Réseaux de neurones (*Neural Networks*) et modèles

Dans cette partie nous introduisons la notion de réseau de neurones artificiel (que nous désignerons indifféremment par réseau ou réseau de neurones). Ces objets ont récemment reçu beaucoup d'attention et sont devenus un concept central de l'apprentissage automatique moderne. La première structure de réseau a été imaginée par McCulloch et Pitts [250] en se basant sur un modèle simplifié des réseaux neuronaux biologiques du cerveau. Depuis, d'autres types d'architectures ont été imaginées pour résoudre différents problèmes de l'apprentissage automatique. De nos jours, l'utilisation de ces structures tire l'essentiel de sa motivation du fait que les réseaux de neurones profonds sont très pratiques pour traiter différents types de données : images, sons, vidéos, textes, données statistiques, séquence d'ADN etc. Actuellement, ce sont en effet ces modèles qui ont permis des progrès significatifs dans plusieurs disciplines : en traitement du signal et de l'image (en reconnaissance vocale et faciale, vision par ordinateur), en traitement de texte (traduction de texte ou reconnaissance de spams), etc.

Dans ce manuscrit nous nous intéressons à leurs utilisations la résolution de problèmes d'évolutions d'interfaces géométriques. Pour cela, nous commençons par rappeler les différents outils qui structurent les réseaux de neurones et le processus d'apprentissage. Nous renvoyons le lecteur aux ouvrages [305, 172, 191, 337, 76] pour une description plus complète de l'apprentissage automatique avec l'utilisation des réseaux de neurones.

Très simplement, un réseau de neurones désigne une *application paramétrique* f_θ dont la structure est basée sur une succession d'opérations linéaires et de non linéarités. En pratique, le *vecteur de paramètres* (ou *paramètre d'apprentissage*) θ est ajusté de sorte à optimiser un *score de performance empirique* sur des tâches de classification ou de régression. Cette phase de réglage des paramètres correspond à l'étape d'*entraînement* (ou la *procédure d'apprentissage*) du réseau de neurones, qui est décisive en apprentissage automatique. Nous verrons des exemples plus loin.

4.1.1 Neurone artificiel

Le neurone (artificiel) est la brique de base des réseaux de neurones. Son architecture a été pensée par McCulloch et Pitts [250] dans le but de proposer un modèle mathématique du cerveau humain. D'un point de vue mathématique, un neurone est une fonction f dont les entrées $x = (x_1, \cdots, x_d) \in \mathbb{R}^d$ sont pondérées par un vecteur de poids de connexion $w = (w_1, \cdots, w_d)$ complétée par un biais $b \in \mathbb{R}$, et associée à une fonction d'activation $\rho : \mathbb{R} \to \mathbb{R}$, à savoir

$$y = f(x; (w, b)) = \rho\left(x \cdot w + b\right) = \rho\left(\sum_{i=1}^{d} x_i w_i + b\right)$$

Parfois un neurone est aussi appelé perceptron mono-couche [292]. Afin de visualiser plus simplement un neurone, sa structure peut être représentée par un graphe acyclique. En figure 4.1, nous représentons l'action d'un neurone sur une entrée x.

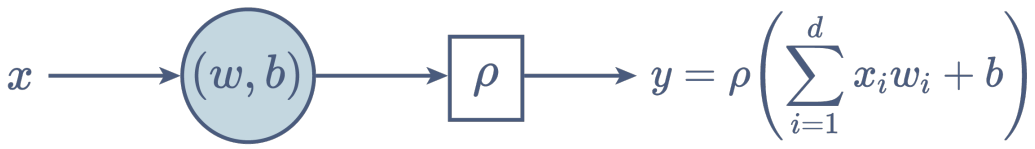


FIGURE 4.1 – Représentation d'un neurone artificiel par un graphe acyclique. L'entrée x passe par le neurone (en bleu) où une première opération linéaire puis une non linéarité ρ lui sont appliquées afin de produire la sortie y.

Un neurone n'est capable de résoudre que des problèmes très simples tels que de la classification linéairement séparable ou de la régression linéaire. Pour résoudre des problèmes plus complexes et en grande dimension, un neurone peut être utilisé comme structure de base pour des modèles plus complexes en formant des connexion entre plusieurs neurones. Ce sont ces modèles que l'on appelle réseaux de neurones.

La manière dont les neurones sont connectés détermine la façon dont les calculs se déroulent et constitue une importante décision de conception en amont pour le concepteur de réseaux de neurones.

4.1.2 Perceptron multicouche (*Multi Layer perceptron*)

Un réseau de neurones est construit en reliant plusieurs neurones entre eux, dans le sens où la sortie d'un neurone constitue une entrée pour un autre. La structure générale d'un réseau de neurones peut donc être vue comme la succession de plusieurs *couches de neurones* (ensembles de neurones) où généralement chacune des couches prend ses entrées sur les sorties de la couche précédente. Notons que la sortie d'un neurone peut également être l'entrée d'un neurone de la même couche ou d'un neurone des couches précédentes. Dans ce cas on parle de structure récurrente que nous n'utiliserons pas dans ce manuscrit. On ne s'intéresse qu'aux structures à propagation avant (*feedworward*) où un neurone ne prend ses entrées que sur les couches précédentes.

Un réseau simple avec une telle structure est le perceptron multicouche (MLP pour *Multi Layer perceptron*) introduit par Rosenblatt [292]. Pour ce modèle, chaque neurone d'une couche prend ses entrées sur les sorties de tous les neurones de la couche précédente. On parle dans ce cas de connexion dense entre chaque couche. Mathématiquement, un MLP est une function

$$f : \mathbb{R}^d \to \mathbb{R}^{N_L},\ x \mapsto x_L.$$

où
— d est la dimension d'entrée du réseau.
— L est le nombre de couches de neurones et N_L est la dimension de sortie du réseau.
— la sortie x_L résulte des opérations suivantes :

Entrée (*input layer*) : $x_0 = x$.

Couches cachées (*hidden layers*) : $x_k = \rho\left(W_k x_{k-1} + b_k\right),\ k = 1, \cdots, L-1$

Couche de sortie (*output layer*) $k = L$: $x_L = \psi\left(W_L x_{L-1} + b_L\right)$.

A chaque étape, W_k est une matrice dont le nombre de lignes correspond au nombre N_k de neurones de la couche k et le nombre de colonnes est le nombre N_{k-1} de neurones de la couche $k-1$. La fonction $\rho : \mathbb{R} \to \mathbb{R}$ est la fonction d'activation des neurones des couches cachées et $\psi : \mathbb{R} \to \mathbb{R}$ celle de la couche de sortie. En pratique, ψ est l'identité mais on retrouve parfois d'autres fonctions. Ici, les fonctions d'activation sont appliquées coordonnée par coordonnée i.e si $x = (x_1, \cdots, x_n) \in \mathbb{R}^n$, alors $\rho(x) := (\rho(x_1), \cdots, \rho(x_n))$.

L'*architecture* d'un MLP est donc totalement encodée par le tableau des dimensions $[d, N_1, \cdots, N_L]$ et des fonctions d'activation ρ et ψ.

Tout comme un neurone, il peut être intéressant de représenter un réseau de neurones par un graphe acyclique afin visualiser l'action de chaque neurone du réseau.

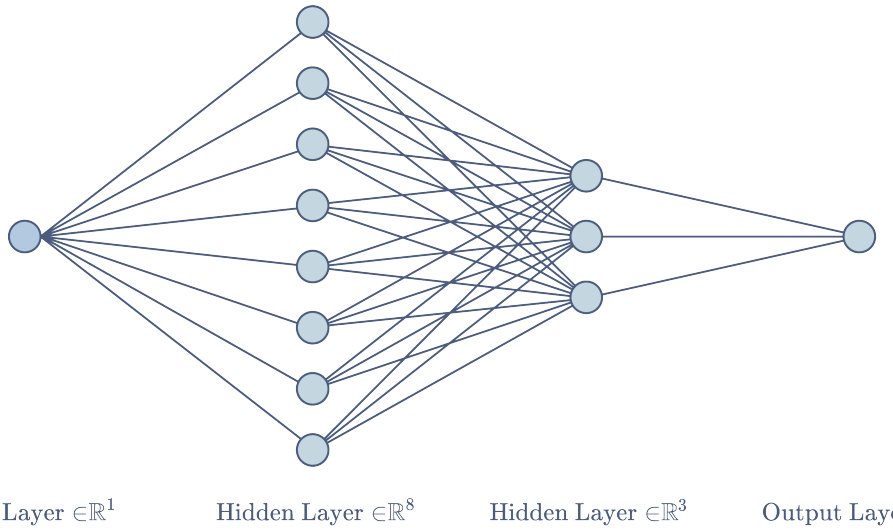


FIGURE 4.2 – Représentation par un graphe acyclique d'un réseau de neurones à deux couches cachées avec respectivement 8 neurones et 3 neurones - et une couche de sortie avec 1 neurone. Chaque neurone est représenté par un disque bleu. L'entrée est représentée par un disque gris.

Paramètres et fonctions d'activation

Les paramètres d'apprentissages d'un réseau de neurones sont les couples $\theta = (W_k, b_k)_{1 \leq k \leq L}$. Ce sont les paramètres du réseau qui sont ajustés au cours de la phase d'apprentissage. Ces paramètres sont aussi appelés *poids* (*weights*) du réseau de neurones. Ce sont les paramètres qui peuvent être modifiés durant l'entraînement d'un neurone.

La fonction d'activation n'est pas un paramètre d'apprentissage mais plutôt un hyperparamètre dans le sens où elle est choisie en amont pendant la définition du neurone, avant la phase d'apprentissage. Le choix de la fonction d'activation dépend du problème étudié. En pratique, on demande à ce qu'elle soit non linéaire et/ou différentiable. En effet, pour des structures plus complexes qu'un neurone, la non linéarité permet d'obtenir des approximations de fonctions très compliquées. La différentiabilité est elle aussi une caractéristique importante qui permet l'utilisation des algorithmes d'optimisation basés sur les gradients afin d'ajuster les paramètres d'apprentissage. Il y a donc une multitude de possibilités pour une fonction d'activation mais en pratique on n'utilise qu'un certain nombre de fonctions. En voici quelques-unes :

— la fonction seuillage ou Heaviside $\rho = \mathbb{1}_{\mathbb{R}^+}$. Elle a été introduite par McCulloch et Pitts [250] pour montrer que les fonctions simples de logique comme NOT, AND, OR peuvent se réaliser comme un réseau de neurones. Cependant, cette fonction d'activation est totalement inadaptée à l'apprentissage moderne puisque sa dérivée est nulle presque partout. Cela pose en effet un certain nombre de difficultés numériques dans le processus d'apprentissage où les paramètres d'apprentissage sont ajustés dans un grand nombre de cas par des algorithmes d'optimisation basées sur le calcul du gradient. Ce qui signifie que durant le processus d'apprentissage, aucun paramètre du modèle n'est ajusté avec ce type d'algorithmes.

— la fonction sigmoïde ou logistique $\rho(x) = \dfrac{1}{1+\exp(-x)}$. On la retrouve pour les problèmes de régression logistique (où les sorties sont dans $[0,1]$) ou en classification binaire en interprétant chaque sortie d'un réseau comme une probabilité d'appartenir à une classe. Une autre variante souvent utilisée aussi est la fonction tangente hyperbolique.

— la fonction *Rectified Linear Unit* (ReLU) $\rho(x) = \mathrm{ReLU}(x) = \max(x, 0)$. C'est l'une des fonctions les plus utilisées de nos jours. Elle n'est pas différentiable en 0 mais en pratique ce n'est pas réellement un problème puisque les valeurs d'entrées sont souvent éloignées de 0. C'est aussi une fonction qui est connue pour ces effets de parcimonie. Le gradient de ReLU est néanmoins nul pour les valeurs négatives, donc aucune information n'est récupérée pour des entrées négatives. Il est conseillé dans ce cas d'utiliser des variantes à ReLU comme $\rho(x) = \max(x, 0) + \alpha \min(x, 0)$ pour inclure les valeurs négatives (leurs contributions dépendent alors du paramètre α). D'autres variantes plus régulières ont aussi été proposées comme la fonction ELU (*Exponential Linear Unit*) [102].

— Les fonctions d'activation périodiques comme le sinus ou le cosinus sont aussi de plus en plus utilisées [275, 315, 233, 273, 291, 255].

— On retrouve aussi la fonction Gaussienne $x \mapsto e^{-x^2}$ pour son côté radial et multi-échelle [234, 186, 202, 255].

4.1.3 Théorème d'approximation universelle.

En tant que modèles de prédiction/d'approximation, il est naturel de s'interroger sur la puissance expressive [185, 283, 284] des réseaux de neurones c'est-à-dire la capacité d'approximation de ce type de structures. Cette question très générale concerne en particulier les classes de fonctions que l'on peut approcher par les réseaux et le type d'architecture de réseaux à utiliser.

Les premiers résultats à ce sujet sont dus à Cybenko [109] qui a montré que toute fonction continue sur un compact pouvait être arbitrairement bien approchée (au sens de la norme uniforme) par un MLP à une couche cachée et avec une fonction d'activation de type sigmoïdale. Hornik et ses collaborateurs [196, 194] ont montré que ce n'était pas le choix spécifique de la fonction d'activation mais plutôt l'architecture en multi-couche qui faisait des réseaux de potentiel approximateurs. Les auteurs de [226, 281] montrent d'ailleurs qu'il est indispensable que la fonction d'activation ne soit pas un polynôme. Ainsi, à un nombre de couches fixé, il est possible d'approcher toute fonction continue sur un compact en considérant suffisamment de neurones sur chaque couche. D'un point de vue pratique, on se rend compte que ce sont les *réseaux (très) profonds* qui donnent de meilleures performances. Leurs performances ont motivé l'étude théorique de ce type de structures où on privilégie la profondeur (plutôt que la largeur) d'un réseau [241, 190, 205, 276, 318].

Des études plus précises sur la qualité d'approximation des réseaux avec notamment des bornes sur l'erreur d'approximation ont été menées par plusieurs auteurs [101, 256, 254, 255, 339, 138] . Ces résultats sont importants d'un point de vue théorique mais aussi pratique car ils permettent de comprendre les limites de certaines architectures des réseaux de neurones. Nous renvoyons le lecteur intéressé à l'ouvrage [305] sur une présentation plus approfondie à ce sujet.

Plusieurs extensions au théorème de Hornik existent, comme celle à des fonctions d'activation discontinues [226], à des domaines non compacts [194, 205], et à des architectures de réseaux et des topologies alternatives [195, 194, 120, 256, 346]. En particulier, motivé par les récentes applications des réseaux de neurones aux équations aux dérivées partielles, les résultats d'approximations des réseaux ont été étendus aux espaces de Sobolev [184, 1, 183, 182, 197]. On trouve aussi des extensions du théorème d'approximation universelle aux opérateurs [93, 238]. Ces résultats ont notamment motivé la construction de nouveaux types de réseaux pour approcher des opérateurs [237].

En figure 4.3, nous illustrons un exemple d'approximation d'une fonction par un réseau de neurones. Le réseau utilisé est celui illustré en figure 4.2 avec la gaussienne comme fonction d'activation.

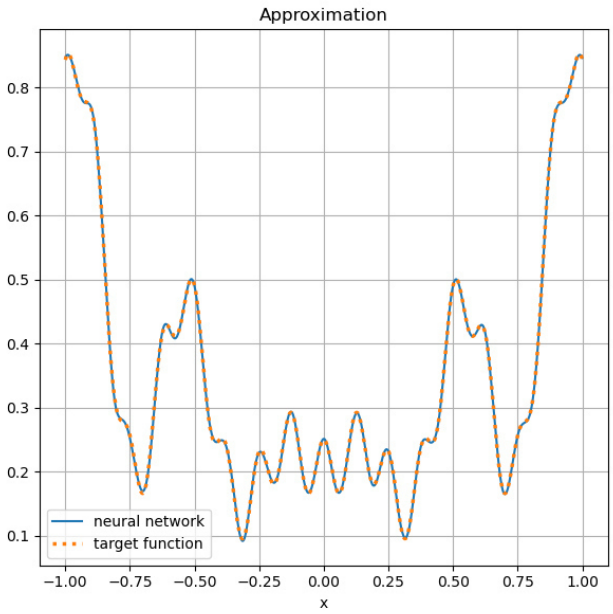


FIGURE 4.3 – Exemple d'approximation d'une fonction par un réseau de neurones : en pointillés orange, le graphe de la fonction à approcher ; en trait bleu plein, le graphe reconstruit par le réseau de la figure 4.2.

Vers l'apprentissage profond (*Deep Learning*).

D'un point de vue théorique, les théorèmes d'approximation par réseaux de neurones sont très intéressants car ils impliquent en particulier que ces structures simples peuvent représenter une grande variété de fonctions lorsqu'on leur attribue des poids (paramètres du réseau) appropriés. Cependant, ils ne fournissent généralement pas de construction pratique des paramètres d'apprentissage, mais indiquent simplement qu'une telle construction est possible. Ce qui pose un certain nombre de difficultés sur le choix de l'architecture à utiliser en pratique. Encore aujourd'hui, ce choix se fait par empirisme.

Ce n'est que récemment que la popularité des réseaux de neurones a fait un bond, depuis l'utilisation des *réseaux de neurones profonds*. Ces structures peuvent comporter des millions de neurones répartis sur plusieurs couches et sont utilisées en *apprentissage profond* pour concevoir des mécanismes d'apprentissage automatique. Ces modèles, taillés pour le traitement de bases massives de données, offrent en effet un bon compromis entre expressivité et simplicité : les premières couches permettent d'extraire des caractéristiques simples (les contours d'une image par exemple) des données d'entrée, qui sont ensuite combinées par les couches suivantes afin de former des représentations de plus en plus abstraites et complexes des données d'entrée. Les MLP sont un exemple de tels modèles mais il existe d'autres types de réseaux comme les réseaux convolutifs profonds utilisés pour les données spatiales et séquentielles comme les images, les sons et les vidéos. Nous y reviendrons plus tard.

Grâce aux capacités de stockage et de calcul machine, l'utilisation des réseaux de neurones (très) profonds est devenue plus plausible. L'augmentation de la puissance de calcul des machines (en particulier du calcul en parallèle et de la distribution des CPU et des GPU) et de la vitesse d'accès aux données massives (*Big Data*), et aux solutions software de l'apprentissage automatique (en particulier grâce aux outils libres *tensorflow* et *Pytorch*), les algorithmes de l'apprentissage profond sont devenus de plus en plus accessibles au grand public. Ces ressources informatiques ont énormément aidées les réseaux de neurones à atteindre des performances de pointe dans une variété impressionnante de problèmes d'apprentissage automatique.

4.2 Entraînement des réseaux de neurones

Dans cette partie, nous décrivons plus précisément comment sont ajustés les paramètres d'apprentissage d'un réseau de neurones. Ce sont en effet ces paramètres qui sont au cœur du processus d'apprentissage et qui permettent d'aboutir à des modèles de prédiction consistants et pertinents.

Base de données et critères d'efficacité

Le processus d'apprentissage consiste à estimer le vecteur de paramètres θ d'un réseau de neurones f_θ sur un jeu de données $(X_i, Y_i)_{1\leq i\leq N}$ (*données d'apprentissage*) à partir d'un algorithme d'optimisation (aussi appelé *algorithme d'apprentissage* dans ce contexte).

La question est maintenant de savoir en quel sens on estime que f_θ est un bon modèle de prédiction et comment mesure t-on l'erreur entre la prédiction $f_\theta(X_i)$ et le label Y_i ? Une manière de répondre à cette question consiste à introduire une *fonction d'erreur* (*coût* ou *perte*) $\ell(f_\theta(X_i, Y_i))$ qui mesure l'écart entre la *prédiction* $f_\theta(X_i)$ de l'entrée X_i par rapport au label Y_i. Le choix de la fonction de perte dépend des données et du problème étudié.

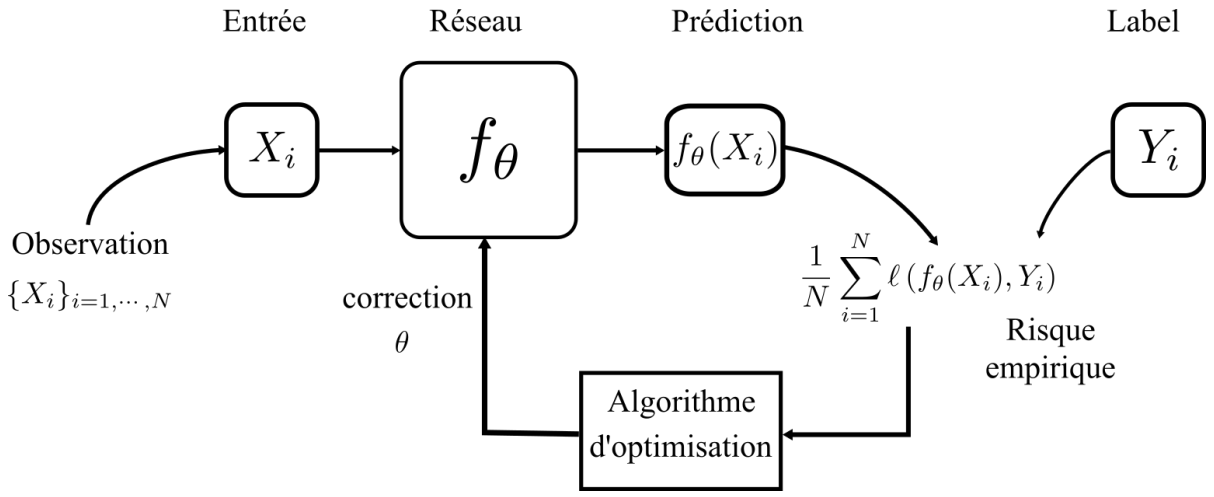


FIGURE 4.4 – Processus d'apprentissage d'un réseau f_θ. Le réseau est appliqué à chaque entrée X_i puis chaque estimation $f_\theta(X_i)$ est comparée au label Y_i à partir de la fonction d'erreur ℓ. Le vecteur de paramètres θ est ensuite corrigé par un algorithme d'optimisation de sorte que l'erreur empirique $L_N(\theta)$ soit minimale. On répète le processus suffisamment de fois afin que $f_\theta(X_i)$ estime correctement la sortie Y_i. Lorsque le processus boucle sur l'ensemble de toutes les données on parle d'une *époque*.

Du point de vue de l'apprentissage statistique [191], les données $(X_i, Y_i)_{1\leq i\leq N}$ étudiées peuvent être vues comme la réalisation d'une distribution de probabilité jointe μ. Les couples (X_i, Y_i) représentent alors les *observables*, les éléments que nous avons directement accès et pour lequel nous pouvons faire des mesures. La mesure μ correspond à toutes les possibilités que nous ne connaissons que partiellement à travers les couples (X_i, Y_i). Finalement, notre problème d'apprentissage supervisé peut être décrit comme un problème d'optimisation sur le paramètre θ :

$$\underset{\theta\in\Theta}{\operatorname{argmin}}\, L(\theta) := \mathbb{E}_{(X,Y)\sim\mu}\left[\ell(f_\theta(X), Y)\right], \tag{4.2.1}$$

où Θ est un ensemble des vecteurs de paramètres fixé au préalable. Il correspond à l'*a priori* que l'on se fixe sur la structure de réseau utilisée pendant le processus d'apprentissage.

La fonction L est souvent appelée *erreur* (ou le *risque*) de *généralisation* car elle mesure l'écart de toutes les possibilités (x, y) sous la mesure μ.

En pratique, puisque nous n'avons pas directement accès à la mesure jointe μ, le paramètre θ est optimisé à partir de l'*erreur* (ou *risque*) *empirique*

$$L_N(\theta) := \frac{1}{N}\sum_{i=1}^{N}\ell(f_\theta(X_i), Y_i). \tag{4.2.2}$$

On espère alors que la base d'entraînement $(X_i, Y_i)_{1\leq i\leq N}$ recouvre suffisamment l'ensemble de toutes les configurations possibles (donné par μ) pour que le réseau obtenu après minimisation soit capable d'extrapoler en dehors des données d'apprentissage. On parle dans ce cas de capacité de *généralisation* du réseau.

Remarque 4.2.1. *Suivant le problème, l'erreur empirique peut être pénalisée par un terme de régularisation sur les paramètres θ. Dans ce cas, celle-ci est plutôt de la forme,*

$$L_N(\theta) := \frac{1}{N}\sum_{i=1}^{N} \ell(f_\theta(X_i), Y_i) + \lambda \mathcal{R}(\theta). \tag{4.2.3}$$

Le premier terme correspond alors à une pénalité d'attache aux données $\{(X_i, Y_i)\}_{1\leq i\leq N}$, alors que le terme $\mathcal{R}(\theta)$ est une pénalité de régularisation sur le paramètre d'apprentissage θ. Ce dernier terme permet de retrouver certaines propriétés souhaitables sur le modèle de prédiction f_θ avec une importance relative donnée par λ afin de fournir des capacités de généralisation permettant de faire face au fait que nous n'avons accès qu'à un ensemble limité de données d'entraînement en pratique.

Les régularisations souvent utilisées sont la norme L^1 (pour de la parcimonie sur les paramètres d'apprentissage) et L^2 (souvent utilisée pour rendre strictement convexe le problème et accélérer l'apprentissage). Voir [55, 60] pour plus de détails sur les effets de la pénalisation dans un problème d'optimisation.

4.2.1 Algorithmes d'optimisation

Maintenant que le processus d'apprentissage a été présenté, nous expliquons comment est optimisé en pratique le risque empirique dans des conditions de grande cardinalité. Ce cas de figure arrive en effet très souvent pour des problèmes d'apprentissage où le nombre de données d'apprentissage et la dimension des entrées sont très grandes. Pour simplifier les notations, nous pouvons reformuler notre problème sous la forme

$$\min_{\theta\in\Theta} L_N(\theta) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(\theta), \tag{4.2.4}$$

où on a posé $\ell_i(\theta) = \ell(f_\theta(X_i), Y_i)$.

Lorsque la base de données est arbitrairement grande (N est très grand), il n'est plus possible d'utiliser un algorithme d'optimisation dont la complexité à chaque itération dépend de N. Pour pallier cette difficulté, on a souvent recours à des méthodes d'optimisation stochastique. L'idée général de ce type de méthodes n'est pas d'être nécessairement plus rapide que les algorithmes traditionnels déterministes mais plutôt d'alléger le coût de calcul lorsque N est très grand et que le calcul direct de $\nabla L_N(\theta)$ n'est plus possible en pratique.

L'algorithme le plus simple est la descente gradient stochastique (SGD pour *Stochastic Gradient Descent*) : on remplace le calcul directe de ∇L_N par celui d'un seul $\nabla \ell_i$ où i est tiré uniformément dans $\{1, \cdots, N\}$ à chaque itération. En partant d'une initialisation θ^0, le schéma complet est,

$$\theta^{k+1} = \theta^k - \tau^k \nabla \ell_{i(k)}(\theta^k),$$

où à chaque itération k, l'indice $i(k)$ est tiré uniformément et sans répétition dans $\{1, \cdots, N\}$ et avec cycle sur $\{1, \cdots, N\}$ afin de parcourir l'ensemble de toutes les données d'apprentissage. L'intérêt de cette approche est qu'elle fournit une estimation sans biais de ∇L_N dans le sens où $\mathbb{E}_{i(k)}\left[\nabla \ell_{i(k)}\right] = \nabla L_N$.

Si le calcul de chaque $\nabla \ell_i$ est de complexité linéaire p, alors à chaque itération, la SGD a une complexité $\mathcal{O}(p)$ alors que la descente de gradient classique a une complexité $\mathcal{O}(Np)$. La SGD est alors un algorithme plus avantageux lorsque N est très grand.

Le choix du pas de temps τ^k (aussi appelé *taux d'apprentissage* dans ce contexte) est primordial pour que l'algorithme puisse converger vers un minimum. En pratique, il est calibré périodiquement de sorte a être de plus en plus petit pour réduire le bruit introduit par l'échantillonnage sur le gradient et limiter les phénomènes d'oscillation proche d'un minimum, souvent observés avec cette méthode. On opte aussi pour une descente par mini-paquets (*mini-batch*) pour réduire la variance de l'estimation du gradient et tempérer les effets d'oscillation :

$$\theta^{k+1} = \theta^k - \tau^k \frac{1}{B}\sum_{i\in I_k} \nabla \ell_i(\theta^k),$$

où $I_k \subset \{1, \cdots, N\}$ dont les éléments sont tirés uniformément dans $\{1, \cdots, N\}$ et le cardinal $|I_k| = B \ll N$.

Notons que les itérations θ^{k+1} sont des vecteurs aléatoires et que l'analyse théorique de ce type de schéma consiste à étudier la convergence (en probabilité ou en moyenne quadratique en général) de la

suite de vecteurs aléatoires $(\theta^k)_k$ vers un vecteur déterministe minimisant L_N. Nous renvoyons le lecteur à l'ouvrage [305] pour plus de détails à ce sujet.

En pratique, du fait de la décroissance rapide de τ^k vers 0, la descente de gradient est souvent très lente à converger. Pour améliorer la vitesse de convergence et diminuer les effets d'oscillation de cette méthode, une possibilité est de coupler l'approche stochastique avec une méthode dite de *momentum*. C'est essentiellement une petite modification de la SGD, de sorte que le mouvement dans l'espace des paramètres est moyenné sur plusieurs pas de temps. On introduit pour cela un vecteur vélocité v^{k+1} à chaque itération,

$$\begin{cases} v^{k+1} = \mu v^k - \tau^k \nabla \ell_{i(k)}(\theta^k), \\ \theta^{k+1} = \theta^k + v^{k+1}, \end{cases}$$

où μ est le paramètre d'amortissement qui contrôle la vitesse à laquelle la vélocité v^k peut changer et dans quelle mesure le gradient influence le mouvement à long terme. Il est aussi possible de changer légèrement le schéma précédent afin de prendre en compte les prochains mouvements :

$$\begin{cases} v^{k+1} = \mu v^k - \tau^k \nabla \ell_{i(k)}(\theta^k + \mu v^{k+1}) \\ \theta^{k+1} = \theta^k + v^{k+1} \end{cases}$$

Cette méthode a été proposée pour la première fois par Nesterov [266] en 1933 afin d'accélérer la vitesse de convergence de la méthode précédente.

La méthode *Adam* [210] est sans doute la méthode la plus utilisée actuellement et la plus efficace dans plusieurs problèmes d'apprentissage automatique. Elle est basée sur un taux d'apprentissage adaptatif qui permet une bonne estimation des moments d'ordre 1 et 2 de ∇L_N à chaque itération. Elle est similaire à la méthode de Nesterov tout en assurant un meilleur contrôle du bruit introduit par la SGD grâce à une bonne estimation des moments de ∇L_N à chaque itération.

De nombreuses autres variantes offrent une meilleure stabilité et de meilleures propriétés de convergence que la SGD. Nous renvoyons le lecteur intéressé à la référence [294] pour un catalogue des différentes méthodes les plus utilisées actuellement en pratique. Voir aussi [48, 77, 55] pour les différentes techniques utilisées pour traiter le cas non régulier.

Un autre algorithme très important pour les problèmes d'apprentissage de grandes cardinalités est la différentiation automatique [30]. Cet algorithme, basé sur la règle de dérivation en chaîne, est très utilisé pour le calcul exact des dérivées de fonctions très complexes avec un grand nombre de paramètres d'entrée. Dans le cas des réseaux, cet algorithme est utilisé pour calculer les dérivées par rapport aux paramètres d'apprentissage. Son utilisation pratique en apprentissage est hautement facilitée grâce aux librairies telles que *Tensorflow* ou *Pytorch* qui offrent une machinerie simplifiée de la différentiation automatique.

4.3 Exemples classiques d'application des réseaux de neurones

4.3.1 Un problème d'approximation

En figure 4.5, on illustre un exemple d'utilisation des réseaux de neurones au problème de régression. On s'intéresse à une collection de paires entrée-sortie $(x_i, y_i)_{1 \leq i \leq N}$ dans $\mathbb{R}^2$. L'objectif est d'ajuster un réseau f_θ sur les données (x_i, y_i) à travers le choix du vecteur de paramètres θ afin d'approcher au mieux les données $f_\theta(x_i) \simeq y_i$. Une manière de procéder est de minimiser l'erreur quadratique moyen empirique $\frac{1}{N}\sum_{i=1}^N |f_\theta(x_i) - y_i|^2$ par un algorithme de descente de gradient sur θ. Le premier modèle (en vert) correspond à un neurone simple. Le second modèle (en rouge) correspond à un réseau à une couche cachée. La même fonction d'activation est utilisée pour les deux modèles : ReLU. On voit qu'en augmentant la complexité du réseau, on gagne en précision.

4.3.2 Un problème de classification

Un autre cas de figure fréquent est le problème de classification multi-classe où les entrées X sont distinguées en K classes (blanc-noire/chien-chat pour $K = 2$ par exemple). Dans ce cas, on utilise souvent le coût $\ell(x, y) = -\sum_{i=1}^K y_i \log(x_i)$ où l'on a identifié les labels aux sommets d'un simplexe de probabilité. Dans ce cas, le réseaux f_θ est construit de sorte à ce que les sorties $f_\theta(x)$ soient dans $[0, 1]^K$. Le problème de classification consiste alors à minimiser l'erreur d'entropie croisée,

$$-\mathbb{E}_{(X,Y)\sim\mu}\left[\sum_{i=1}^K Y_i \log((f_\theta(X))_i)\right].$$

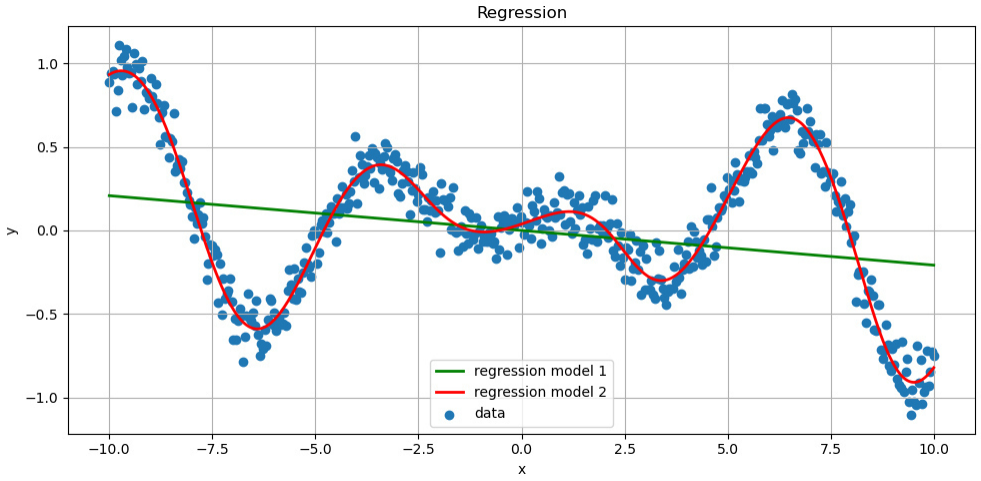


FIGURE 4.5 – Exemple d'application des réseaux à un problème de régression. On représente une collection de paires entrée-sortie $(x_i, y_i)_{1\leq i\leq N}$ dans $\mathbb{R}^2$ (point bleu). Le second modèle (en rouge) correspond à un réseau à une couche cachée. La même fonction d'activation est utilisée pour les deux modèles : ReLU.

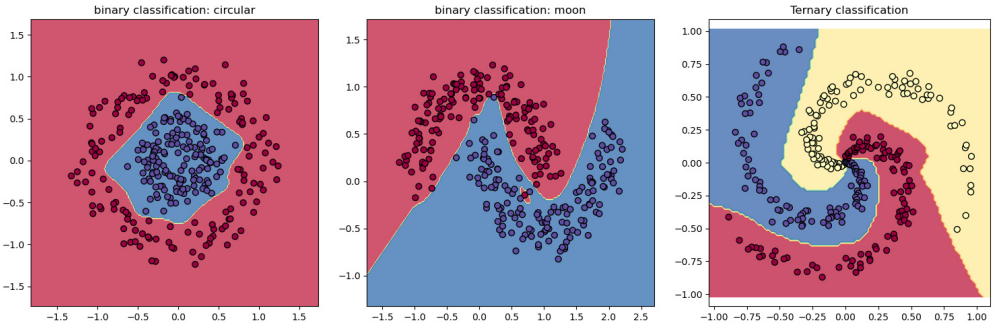


FIGURE 4.6 – Exemples d'application des réseaux à un problème de classification. On s'intéresse à une collection de paires $(x_i, y_i)_{1\leq i\leq N} \in \mathbb{R}^2 \times \{1, \cdots, K\}$ (K = nb de classes) représentées par des points dont la couleur indique les différentes classes (rouge, bleu) ou (rouge, bleu, jaune). Les sorties du réseau sont représentées par les lignes de niveau (bleu-rouge ou bleu-rouge-jaune).

4.4 Structure de réseaux et robustesse

Une architecture de réseau général (en particulier pour les réseaux profonds) mène souvent à un problème d'optimisation fortement non convexe. Il est donc difficile d'assurer la convergence des algorithmes d'optimisation vers des optimums globaux. On constate cependant que les techniques de type SGD fonctionnent étonnamment bien dans la pratique, et il semble que le succès des réseaux profonds (en particulier la capacité de ces modèles à bien se généraliser) est en grande partie dû à la dynamique du SGD (qui induit un effet de régularisation implicite) et d'autres techniques de régularisation.

4.4.1 Sous/sur-apprentissage (*under/over -fitting*).

Il peut arriver que l'erreur d'apprentissage empirique ne décroisse pas suffisamment au cours de l'apprentissage pour que le réseau puisse correctement représenter les données d'entraînement. Ce phénomène de sous-apprentissage peut être dû à plusieurs facteurs. L'une des principales raisons vient du calibrage en amont des hyperparamètres du problème (réseau pas assez complexe ou pas adapté au problème, nombre de données insuffisant pour représenter le problème, choix des paramètres de l'algorithme d'optimisation, etc). Au contraire, il peut aussi arriver que le réseau de neurones se spécialise trop aux données d'entrainement au cours du processus d'apprentissage et qu'il ne puisse pas se généraliser à de nouvelles données. On parle dans ce cas de sur-apprentissage. Ces deux phénomènes sont très fréquents en apprentissage...

Pour éviter le sous/sur-apprentissage, on a souvent recours à des méthodes de régularisation. Nous avons déjà abordé les techniques de régularisation sur les paramètres d'apprentissage par pénalisation du risque empirique. Voici d'autres type de régularisations que l'on rencontre souvent en pratique :

Augmentation de données : Lorsque la base d'apprentissage n'est pas suffisamment grande, on a souvent recours à une augmentation artificielle des données. C'est le cas par exemple pour les tâches

liées au traitement d'images : lorsqu'il n'y pas suffisamment de données, on utilise des transformations qui modifient l'aspect de l'image sans pour autant affecter les caractéristiques (*patterns*) de celle-ci [311]. On enrichit ainsi la base d'apprentissage en utilisant des transformations partielles des données.

Dropout : Le dropout est une technique qui consiste à désactiver certains neurones à chaque étape de la descente de gradient afin d'empêcher le sur-apprentissage sur les données d'entraînement.

4.4.2 Le fléau de la dimension (*The curse of dimensionality*)

L'objectif de l'apprentissage est d'inférer un modèle de prédiction à partir des données d'apprentissage. Cette stratégie donne satisfaction lorsque le réseau est capable d'extrapoler sur de nouvelles données. Dans le pire des scénarios, i.e lorsque les données d'entrainement ne sont pas suffisamment représentatives des configurations possibles (typiquement lorsque la taille N de la base de données est trop petite par rapport à la dimension d), des hypothèses (des *a priori*) sont requises afin de permettre au réseau de se généraliser correctement (*no free lunch theorem*). C'est le cas typique des données telles que les images, les signaux et les vidéos mais aussi les données textuelles. Pour ce type de données la dimension d d'une entrée peut être très grande alors que les informations structurantes qui les constituent sont locales. Il est important alors d'injecter suffisamment de connaissances au système d'apprentissage (principalement sur la structure du réseau ou par réduction de la dimension des entrées) afin de diminuer la *dimensionnalité* c'est-à-dire supprimer les informations qui ne sont pas importantes ou redondantes par rapport à la tâche à accomplir.

4.4.3 Les réseaux convolutionnels (*Convolutional neural networks*)

Afin d'être en mesure de traiter des données de grande taille et d'améliorer les performances d'apprentissage, il est important de tirer parti de certaines connaissances préalables sur la structure des données typiques à traiter. Pour les images, les signaux ou les vidéos il est par exemple important de prendre en compte l'information locale et certaines propriétés propres à ce type de données : invariance par translation, par localisation et aspect multi-échelle.

Les MLP ne sont malheureusement pas très adaptés à l'extraction de caractéristiques (*patterns*) de données avec une structure spatiale ou séquentielle : l'application d'un MLP à une image (par exemple) nécessite de transformer les images en vecteurs par une opération d'empilement. Cependant, cette opération intermédiaire ne permet pas de garder certaines informations spatiales contenues dans une image. Pour prendre en compte ces informations locales, il est plus judicieux d'effectuer des moyennes pondérées entre pixels voisins, des opérations de *filtration linéaire*. On limite ainsi le champ de réception d'une entrée tout en conservant une forte corrélation spatiale locale afin de produire des représentations parcimonieuses [245] à toute échelle. On parle alors de *connectivité locale* et *partage de poids*.

En itérant ces opérations en cascade, on arrive alors à récupérer des motifs/caractéristiques à toutes échelles. Ce principe est à la base des réseaux convolutifs [219] qui sont maintenant les briques de base de presque tous les modèles les plus utilisés en apprentissage automatique.

Architecture d'un réseau de convolutif

Une architecture de réseau de neurones convolutif (ou simplement CNN) est structurée par un empilement de trois opérations :

1. Une première couche de filtrage linéaire qui effectue une opération de convolution par une collection de noyaux (*filtres*) afin d'extraire différentes caractéristiques d'une entrée donnée. Le nombre de sorties est donné par le nombre de noyaux utilisés : si $x \in \mathbb{R}^d$ est une entrée et $(K_i)_{1 \leq i \leq q}$ sont les noyaux de la couche de convolution, alors celle-ci produit les sorties $\{x \star K_i + b_i\}_{1 \leq i \leq q}$ où $\star$ correspond au produit de convolution dans $\mathbb{R}^d$ et les b_i sont des biais.
2. Une deuxième couche de *Pooling* (de *sous-échantillonnage*) qui permet de compresser l'information en fusionnant des caractéristiques similaires en une seule. En particulier, les types de pooling les plus populaires sont le max et l'average pooling, où les valeurs maximales et moyennes sont prises, respectivement.
3. Une opération de seuillage/de correction : tout comme un MLP, on applique une fonction d'activation à la sortie de la deuxième couche afin de mettre plus en évidence certaines caractéristiques d'une entrée.

Paramètre d'apprentissage d'un réseau de convolution.

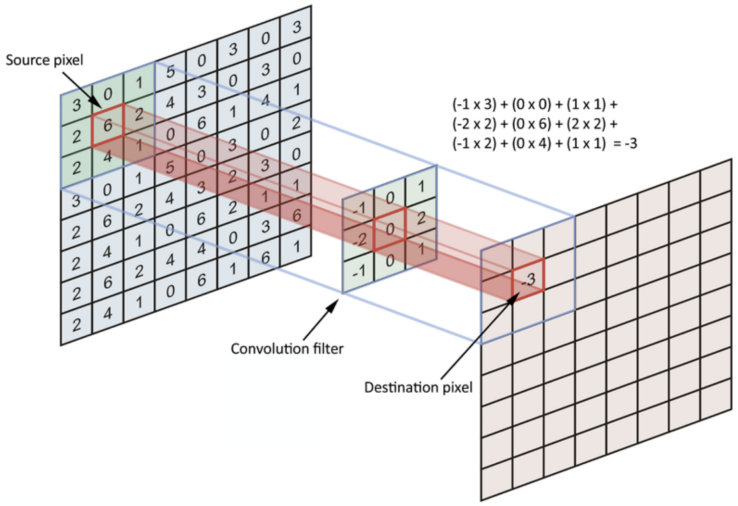


FIGURE 4.7 – Convolution layer.

C'est dans la couche de convolution que l'on trouve la force du CNN. Le réseau est paramétré par une collection de noyaux à toutes échelles (et occasionnellement avec des biais) $\theta = \{(K_i, b_i)\}_{1 \leq i \leq d}$ qui sont optimisés par descente de gradient sur des tâches d'apprentissage (classification, segmentation, etc...).

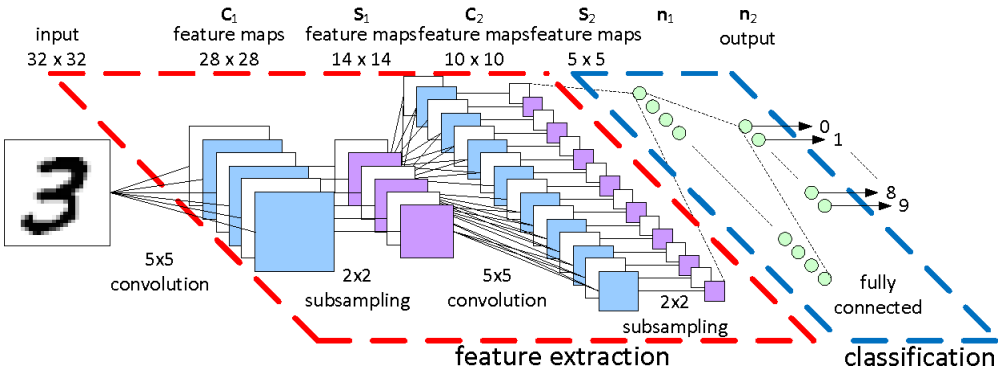


FIGURE 4.8 – Architecture d'un réseau convolutif pour l'extraction de caractéristiques en classification d'images. Source [279]. L'intuition qui sous-tend ce modèle est que plus on descend dans les couches, plus les neurones sont réceptifs à de plus grandes zones de l'image. L'utilisation de nombreuses couches de convolutions permet de définir différentes classes de "détecteurs" : une première couche qui détecte des motifs simples tels que les bords et les coins (par exemple), puis des couches intermédiaires qui capturent progressivement des formes plus élaborées en combinant les extractions des couches précédentes.

Chapitre 5

Motivation et résultats

Ce manuscrit de thèse est divisé en deux grandes parties. Nous exposons ici les différentes problématiques qui nous intéressent et que nous complétons au fur et à mesure par des éléments de réponses partielles qui seront exposées plus en détail dans les parties concernées.

5.1 Simulation de phénomènes de mouillage par diffusion de surface

La première partie de ce manuscrit est consacrée aux modèles de champ de phase pour l'approximation du flot de diffusion de surface et la simulation de phénomènes de mouillage. L'idée principale est de formuler le phénomène de mouillage comme un système multiphase Solide-Liquide-Vapeur pour lequel la condition d'angle de contact entre la phase solide et liquide serait traitée de manière implicite, sans avoir à rajouter de conditions aux bords non linéaires. Notre approche suit une démarche similaire aux travaux [71, 67] où les auteurs utilisent un modèle de Allen-Cahn multiphase pour la simulation du mouillage. Nous proposons dans notre cas un modèle basé sur la diffusion de surface en multiphase, modèle avec une conservation locale du volume qui décrit plus précisément la physique du mouillage.

5.1.1 Motivation

Le phénomène de mouillage [116, 84] décrit comment une goutte liquide s'étale sur une surface solide et il est bien connu que le liquide atteint une forme optimale où un angle de contact typique se forme entre le liquide et le solide. La physique qui régit ce problème est assez simple : en négligeant la gravité, la forme d'équilibre du liquide est obtenue par minimisation de l'énergie interfaciale totale entre les phases liquide-solide Γ_{LS}, solide-vapeur Γ_{SV} et vapeur-liquide Γ_{VL},

$$\mathcal{E} = \sigma_{SV}\mathcal{H}^{d-1}(\Gamma_{SV}) + \sigma_{LS}\mathcal{H}^{d-1}(\Gamma_{LS}) + \sigma_{VL}\mathcal{H}^{d-1}(\Gamma_{VL}), \tag{5.1.1}$$

sous la contrainte que le volume du liquide soit préservé au cours de l'évolution.

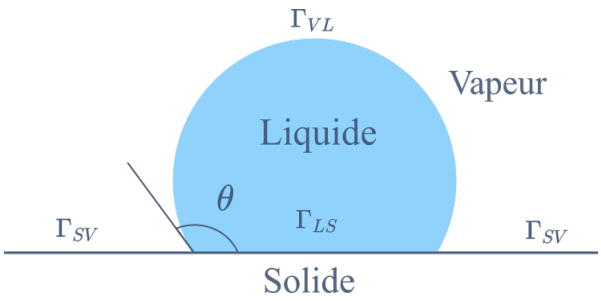


FIGURE 5.1 – Forme d'équilibre d'une goutte liquide sur un support solide plat.

Pour assurer le caractère bien posé du problème, les tensions de surface vérifient l'inégalité triangulaire [86, 243, 259],

$$\sigma_{ij} \leq \sigma_{ik} + \sigma_{kj},\ i,j,k \in \{S,L,V\}.$$

Lorsque le support solide est homogène comme en figure 5.1 l'angle de contact θ est explicit et vérifie l'équation de Young [340]

$$\cos(\theta) = \frac{\sigma_{SV} - \sigma_{LS}}{\sigma_{VL}}.$$

La modélisation du mouillage dans un cadre assez général est difficile car cela nécessite de suivre avec précision l'évolution de la phase liquide tout en assurant des conditions aux bords non linéaires. Dans la littérature, on trouve généralement deux approches distinctes : une première basée uniquement sur l'évolution de la phase liquide, tout en imposant des conditions aux bords non linéaires permettant de retrouver l'angle de contact à l'équilibre. Cette approche est donc limitée aux supports solides où la condition d'angle entre la phase liquide et le support solide est explicite. Une seconde approche consiste plutôt à décrire l'évolution d'un système multiphase avec les phases liquide solide Ω_S, liquide Ω_L et vapeur Ω_V en suivant une dynamique qui tend à minimiser l'énergie (5.1.1) tout en assurant la conservation du volume de chaque phase. Ici, aucune condition de bord n'est donc imposée et l'angle de contact est traité implicitement.

Ces deux approches ont conduit à plusieurs modèles d'approximations du phénomène de mouillage [335, 45, 135, 69, 312]. En formulation champ de phase, elles conduisent à des modèles basés sur l'approximation d'un périmètre.

Approche biphasique avec une énergie de bord

La première approche consiste à considérer le problème du mouillage comme un système n'impliquant que l'évolution de la phase liquide Ω_L dans un domaine fixé Ω comme représenté en figure 5.2,

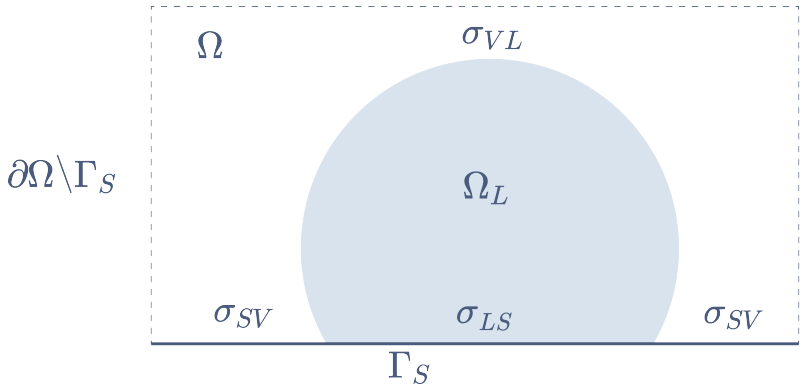


FIGURE 5.2 – Illustration schématique du domaine de simulation. Ω représente le domaine de calcul, $\Omega_L \subset \Omega$ correspond à la phase liquide et la phase vapeur est donnée par $\Omega_V = \Omega \setminus \Omega_L$. Le bord du domaine est décomposé $\partial\Omega = \Gamma_S \cup (\partial\Omega\backslash)$ où Γ_S représente le support solide. L'interface vapeur-liquide est donc $\partial\Omega_L \cap \partial\Omega_V$, l'interface Liquide-Solide est $\Gamma_S \cap \partial\Omega_L$ et l'interface solide-vapeur est $\Gamma_S \cap \partial\Omega_V$.

Dans ce cas, le problème revient à minimiser une énergie de la forme,

$$\sigma_{LV}\mathcal{H}^{d-1}(\partial\Omega_L \cap \partial\Omega_V) + \sigma_{SL}\mathcal{H}^{d-1}(\Gamma_S \cap \partial\Omega_L) + \sigma_{SV}\mathcal{H}^{d-1}(\Gamma_S \cap \partial\Omega_V),$$

sous la contrainte que le volume de Ω_L soit le même durant toute l'évolution.

La formulation champ de ce problème consiste alors à approcher l'énergie précédente par une énergie de la forme,

$$J(u) = \sigma_{VL}\int_\Omega \varepsilon\frac{|\nabla u|^2}{2} + \frac{W(u)}{\varepsilon}\,dx + \int_{\Gamma_S} F_{\sigma_{SV},\sigma_{LS}}(u)\,d\mathcal{H}^{d-1},$$

où $\partial\Omega = \Gamma_S \cup (\partial\Omega \setminus \Gamma_S)$ avec Γ_S qui représente le solide, $u : \Omega \to [0,1]$ représente la phase liquide et le terme d'énergie de surface $F_{\sigma_{SV},\sigma_{LS}}(u)$ est choisi de sorte à prendre comme valeur σ_{SV} lorsque $u = 0$ et σ_{LS} pour $u = 1$. Il n'existe pas vraiment de choix optimal et en pratique, il est par exemple possible d'utiliser l'expression suivante

$$F(u) = \frac{\sigma_{SV} + \sigma_{LS}}{2} - (2u^2(3-2u) - 1)(\sigma_{SV} - \sigma_{LS}).$$

A noter que ce terme d'énergie de surface est équivalent, au moins lorsque la phase solide est suffisamment régulière, à imposer une condition de bord de type Neuman non linéaire

$$\frac{\nabla u}{|\nabla u|} \cdot n = \cos(\theta),$$

permettant ainsi de forcer la condition d'angle de contact θ dérivant de la loi de Young. L'utilisation de ces conditions de Neuman non linéaires peut alors poser quelques difficultés numériques, notamment lorsque la phase solide n'est pas régulière comme dans le cas de support rugueux.

Approche multiphase avec mobilité nulle sur la phase solide

Une autre approche consiste à modéliser le phénomène de mouillage comme un système à trois phases (Solide-Liquide-Vapeur) où la phase solide est supposée « figée », comme représenté en figure 5.3.

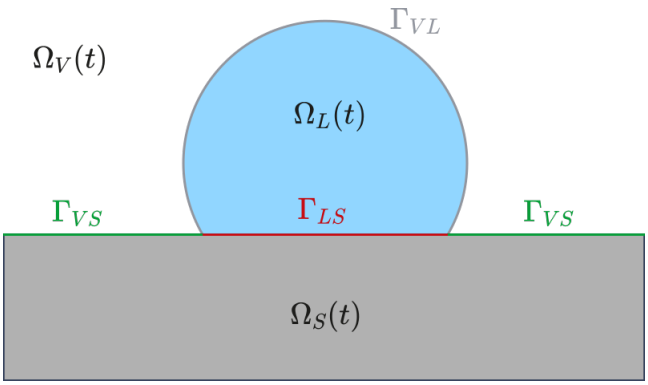


FIGURE 5.3 – Formulation multiphase du phénomène de mouillage. Dans cette configuration, Ω_S, Ω_V et Ω_L représentent respectivement les phases solide, vapeur et liquide.

Notre problème revient à minimiser l'énergie surfacique totale

$$\mathcal{E} = \sigma_{SV}\mathcal{H}^{d-1}(\Gamma_{SV}) + \sigma_{LS}\mathcal{H}^{d-1}(\Gamma_{LS}) + \sigma_{VL}\mathcal{H}^{d-1}(\Gamma_{VL}) \tag{5.1.2}$$

$$= \sigma_L\mathcal{H}^{d-1}(\partial\Omega_L) + \sigma_S\mathcal{H}^{d-1}(\partial\Omega_S) + \sigma_V\mathcal{H}^{d-1}(\partial\Omega_V) \tag{5.1.3}$$

où l'on a introduit les coefficients positifs (grâce à l'inégalité triangulaire),

$$\sigma_i = \frac{\sigma_{ij} + \sigma_{ik} - \sigma_{jk}}{2},\ i,j,k \in \{S,V,L\}.$$

On dit dans ce cas que les tensions de surfaces sont additives puisqu'elles vérifient,

$$\sigma_{VS} = \sigma_V + \sigma_S, \quad \sigma_{SL} = \sigma_S + \sigma_L \quad \text{et} \quad \sigma_{LV} = \sigma_L + \sigma_V.$$

On se retrouve donc avec la minimisation d'une énergie de bords de domaines $\partial\Omega_L, \partial\Omega_S$ et $\partial\Omega_L$. Cette formulation est particulièrement intéressante car l'énergie précédente peut être approchée (au sens de la Γ-convergence [71]) par une énergie de Cahn-Hilliard multiphasique,

$$P(\mathbf{u}) = \sum_{i\in\{V,S,L\}} \sigma_i \int_\Omega \left(\frac{\varepsilon}{2}|\nabla u_i| + \frac{1}{\varepsilon}W(u_i)\right) dx, \tag{5.1.4}$$

où $\mathbf{u} = (u_V, u_L, u_S)$ est soumise à la contrainte de partition $u_L + u_S + u_V = 1$. L'idée consiste alors à considérer un flot de gradient multiphase de cette énergie mais en rajoutant une contrainte permettant de bloquer l'évolution de la phase solide.

Dynamique à l'interface : flot de diffusion de surface

Les deux approches impliquent des dynamiques d'interfaces qui minimisent leur périmètre en imposant une conservation du volume de la phase associée. Une première approximation de la dynamique du mouillage serait alors d'utiliser un mouvement par courbure moyenne avec conservation du volume [36, 71]. Ce flot peut en effet être vu comme le flot de gradient L^2 du périmètre sous la contrainte de volume fixé. Cependant, la dynamique qui en découle entraîne dans certains cas un comportement qui ne correspond plus à la physique du mouillage avec des transferts de matière entre différentes phases connexes. Ce phénomène arrive typiquement lorsque plusieurs gouttes liquides interviennent dans le système (voir [335] pour un exemple avec deux gouttes). Ces effets viennent du fait que la conservation du volume est globale sous

cette dynamique. Une alternative serait alors d'utiliser le flot de diffusion de surface qui permet à la fois une conservation locale du volume et la minimisation du périmètre comme flot de gradient H^{-1} de celui-ci.

Notre objectif est donc de proposer un modèle champ de phase pour la diffusion de surface en multiphase, basé sur l'énergie de Cahn-Hilliard et suffisamment consistant pour modéliser l'évolution de structures fines sur un support rugueux avec une perte de volume négligeable. Nous commençons pour cela notre étude dans le cas le plus simple où uniquement deux phases sont en interaction avant d'étendre au cas général. Notre première étude vise donc à proposer un modèle champ de phase pour la diffusion de surface biphasique dont l'ordre d'approximation est suffisamment élevé pour traiter le cas d'évolution de structures finies.

Modèle de champ de phase pour la diffusion de surface : l'équation de Cahn Hilliard

Lorsque seulement deux phases qui interviennent, nous savons que le périmètre est approché par l'énergie de Cahn-Hilliard [258]. Comme la diffusion de surface est le flot de gradient H^{-1} du périmètre [323], il est donc tout naturel d'approcher le flot de diffusion de surface par un modèle obtenu comme flot de gradient H^{-1} de l'énergie de Cahn-Hilliard. On obtient ainsi dans le cas le plus simple l'équation classique de Cahn-Hilliard (**C-CH**),

$$\begin{cases} \varepsilon^2 \partial_t u = \Delta \mu, \\ \quad \mu = W'(u) - \varepsilon^2 \Delta u. \end{cases} \tag{5.1.5}$$

Ce modèle a été initialement proposé pour modéliser la séparation de phase [82, 79] et permet une conservation locale de la masse.

Cependant, il a été montré que le modèle limite lorsque $\varepsilon \to 0$ du modèle **C-CH** n'est pas la diffusion de surface et aboutissait au modèle de Mullins-Sekerka [280, 6] où la vitesse normale V_n à l'interface $\partial\Omega$ est donnée par $V_n = [\nabla_n v]_\pm$ où v est une extension harmonique de la courbure moyenne,

$$\begin{cases} \Delta v = 0 \text{ sur } \Omega_\pm, \\ \quad v = H \text{ sur } \Gamma \end{cases}$$

avec Ω_+ et Ω_- correspondant respectivement à l'intérieur et l'extérieur de Ω. Le terme $[\nabla_n v]_\pm = \nabla_n v^+ - \nabla_n v^-$ représente alors le saut avec $v^\pm$ la restriction de v sur $\Omega_\pm$.

Ce modèle aurait pu être utilisé pour approcher les phénomènes de mouillage mais l'aspect non local du flot limite rend l'étude asymptotique du modèle **M-CH** assez délicate dans un contexte multiphasique. A notre connaissance il n'existe d'ailleurs aucun résultat à ce sujet.

Equation de Cahn-Hilliard avec mobilité dégénérée

Une autre approche consiste à considérer un modèle plus localisé en introduisant une mobilité dégénérée. On aboutit alors à l'équation de Cahn-Hilliard avec mobilité dégénérée (**M-CH**),

$$\begin{cases} \partial_t u = \operatorname{div}\left(M(u)\nabla\mu\right), \\ \quad \mu = \Delta u - \dfrac{1}{\varepsilon^2} W'(u), \end{cases} \tag{5.1.6}$$

où $M(u)$ est le terme de mobilité dégénérée qui permet de localiser les variations de μ uniquement autour de l'interface. Ce modèle garantit une conservation locale de la masse et peut s'obtenir comme flot de gradient H^{-1} de l'énergie de Cahn-Hilliard en considérant la métrique

$$\langle \mu, \nu \rangle = \int M(u)\nabla\mu \cdot \nabla\nu \, dx.$$

Pour ce type de modèle, le choix du potentiel et de la mobilité est essentiel à la fois pour des questions d'existence de solutions [112, 140] mais aussi pour retrouver le bon flot limite [221, 81]. Cahn [81] montre par exemple qu'avec la mobilité $M(u) = u(u-1)$ et le potentiel logarithmique $W(s) = \frac{1}{2}\theta\left[s\ln(s) + (1-s)\ln(1-s)\right] + \frac{1}{2}s(1-s)$, le flot limite était bien la diffusion de surface. Néanmoins, d'un point de vue numérique le potentiel logarithmique est difficile à traiter. D'autres choix de potentiels sont possibles, nous utilisons dans notre cas le potentiel double puits $W(s) = \frac{1}{2}s^2(1-s)^2$ qui a de bonnes propriétés numériques.

L'analyse asymptotique de (5.1.6) montre que le choix de la mobilité est crucial. En effet, il a été observé dans [181] que certains choix conduisaient à une inconsistance dans les développements asymptotiques :

selon la façon dont les termes sont identifiés dans l'analyse asymptotique, on peut montrer la convergence soit vers le flot de diffusion de surface, soit vers un flot stationnaire avec une vitesse nulle. Les auteurs de [221, 220] ont suggéré qu'une telle inconsistance provenait de la présence d'un terme tangentiel dans le flot limite,

$$V_n = c_1 \Delta_{\partial\Omega(t)} H_\varepsilon + c_2 H \nabla_n H,$$

où c_1 et c_2 sont deux constantes qui dépendent de W et M.

Ce fait a aussi été observé numériquement dans [111, 110] où des effets indésirables de grossissement apparaissaient dans les zones de fortes courbures. Dans [221, 220], les auteurs montrent que la présence du terme tangentiel est liée à la dérivée de M. Pour retrouver le bon flot, il est en fait nécessaire que la dérivée de M s'annule aux puits 0 et 1. Un exemple d'une telle mobilité est donnée par exemple par $M(s) = s^2(1-s^2)$. Avec ce choix, on montre en effet que proche de l'interface, la solution u_ε de (5.1.6) s'écrit

$$u_\varepsilon(x,t) = q\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon),$$

où $\Omega_\varepsilon(t) = \left\{u_\varepsilon(\cdot,t) \leq \frac{1}{2}\right\}$ et la vitesse normale V_ε à l'interface $\partial\Omega_\varepsilon(t)$ satisfait alors

$$V_\varepsilon(t) = c\Delta_{\partial\Omega_\varepsilon(t)} H_\varepsilon(t) + \mathcal{O}(\varepsilon).$$

où c est une constante qui dépend de W et M.

On montre de plus que le volume du domaine $\Omega_\varepsilon(t)$ est préservé au cours du temps avec un ordre ε,

$$\mathrm{Vol}\left(\Omega_\varepsilon(t)\right) = \mathrm{Vol}\left(\Omega_\varepsilon(0)\right) + \mathcal{O}(\varepsilon).$$

Comme nous le savons, la qualité d'approximation d'un modèle champ de phase dépend fortement de son développement asymptotique lorsque $\varepsilon \to 0$. D'un point de vue numérique, l'évolution peut en effet être très impactée. L'exemple de la figure 5.4 illustre l'évolution d'un tube mince en utilisant le modèle **M-CH**. On observe que la structure fine disparaît à partir d'un certain temps. Le fait que cette dernière ne soit pas maintenue au cours de l'évolution vient bien de la précision champ de phase qui n'est pas suffisamment élevée pour ce type de structure, et non du numérique où un algorithme robuste a été utilisé pour cette expérience.

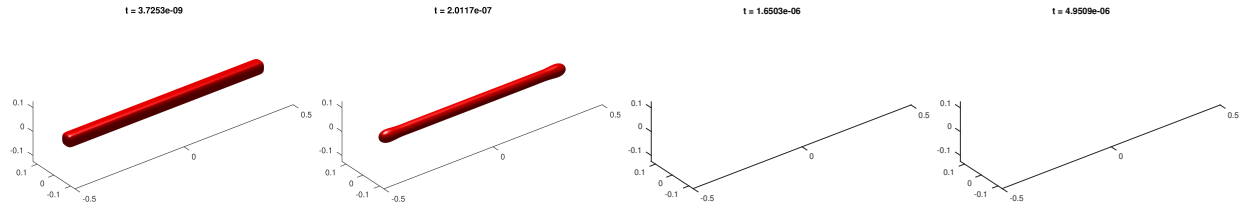


FIGURE 5.4 – Exemple de simulation numérique d'approximation de diffusion de surface avec le modèle **M-CH**.

5.1.2 Contribution de la thèse

Pour avoir un modèle avec une meilleure précision champ de phase, nous avons cherché à comprendre comment modifier le modèle **M-CH** afin d'obtenir une précision d'ordre 2 d'approximation de champ de phase et d'éviter ainsi les pertes de volume observées avec les modèles plus classiques. Nous souhaitions garder une approche variationnelle afin de pouvoir adapter notre démarche dans un cadre multiphase. Cela nous a conduits à introduire et étudier les propriétés du modèle **NMN-CH** qui fait intervenir une seconde mobilité dégénérée $N(u)$:

Un modèle d'ordre 2 avec double mobilités dégénérées

Plus précisément, le modèle est donné par

$$\begin{cases} \partial_t u = N(u)\,\mathrm{div}\left(M(u)\nabla\left(N(u)\mu\right)\right), \\ \mu = \Delta u - \dfrac{1}{\varepsilon^2} W'(u). \end{cases} \tag{5.1.7}$$

Le modèle **NMN-CH** est aussi variationnel et s'obtient comme flot de gradient H^{-1} du périmètre pour la métrique

$$\langle \mu, \nu \rangle = \int M(u) \nabla \left(N(u)\mu \right) \cdot \nabla \left(N(u)\nu \right) \, dx.$$

Nous avons alors cherché à trouver un choix de mobilités (M, N) pour lequel le modèle **NMN-CH** aurait une meilleure précision champ de phase que les modèles classiques. L'analyse asymptotique du modèle **NMNCH** a montré qu'avec le choix particulier des mobilités $M(s) = 2W(s)$ et $N(s) = \frac{1}{\sqrt{2W(s)}}$, on arrivait à gagner un ordre de précision par rapport au modèle **M-CH**,

$$u_\varepsilon(x,t) = q\left(\frac{d(x, \Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2),$$

et la vitesse normale V_ε à l'interface $\partial\Omega_\varepsilon(t)$ satisfait alors

$$V_\varepsilon(t) = c\Delta_{\partial\Omega_\varepsilon(t)} H_\varepsilon(t) + \mathcal{O}(\varepsilon),$$

où c est une constante qui dépend de W, M et N.

De plus, la conservation du volume au cours du temps est assurée avec une meilleure précision que le modèle **MCH**,

$$\mathrm{Vol}\left(\Omega_\varepsilon(t)\right) = \mathrm{Vol}\left(\Omega_\varepsilon(0)\right) + \mathcal{O}(\varepsilon^2).$$

Avec ce modèle, on gagne donc un ordre d'approximation sur le profil mais aussi sur le volume.

La figure 5.5 reproduit la même expérience utilisée avec le modèle **M-CH** sur l'évolution d'un tube mince en 3D mais cette fois-ci avec le modèle **NMN-CH**. On observe que le tube mince converge vers une forme stationnaire avec une union de cinq sphères et le volume semble être conservé au cours du temps. Que ce soit le modèle **M-CH** ou **NMN-CH**, des méthodes numériques robustes ont été utilisées pour cette expérience. Les résultats obtenus témoignent de l'important de la précision champ de phase d'un modèle dans les expériences numériques et que même avec des algorithmes performants il est difficile de maintenir les caractéristiques du flot limite si la précision champ de phase du modèle n'est pas assez élevée.

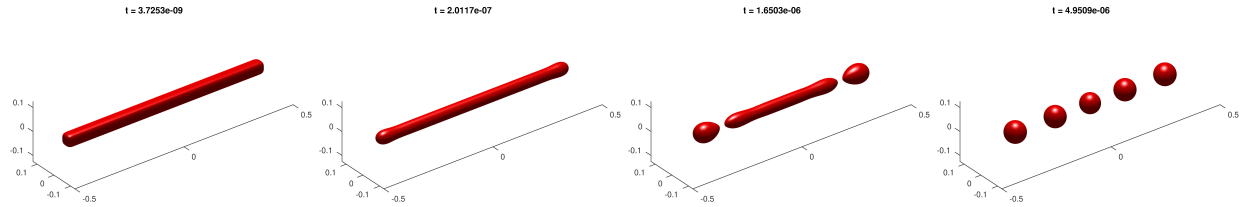


FIGURE 5.5 – Exemple de simulation numérique d'approximation de diffusion de surface avec le modèle **NMN-CH**.

Application au problème de diffusion de surface sur un bord solide

Nous rappelons que nous proposons d'approcher la dynamique du mouillage par un flot de diffusion de surface multiphasique qui assure la minimisation de l'énergie totale (5.1.2) et pour lequel la conservation du volume est locale contrairement au flot par courbure moyenne avec conservation du volume. La vitesse normale V_{ij} $(i, j \in \{V, S, L\})$ à l'interface Γ_{ij} peut s'écrire sous la forme,

$$V_{ij} = \nu_{ij}\sigma_{ij}\Delta_{\Gamma_{ij}} H_{ij},$$

où H_{ij} est la courbure moyenne à l'interface Γ_{ij}, $\Delta_{\Gamma_{ij}}$ est l'opérateur de Laplace-Beltrami sur Γ_{ij}, et $\nu_{ij} \geq 0$ est le coefficient de mobilité à l'interface Γ_{ij}. Les mobilités ν_{ij} décrivent la vitesse à laquelle les particules se déplacent d'une phase à une autre au fur et à mesure que l'interface se déplace. Ces paramètres sont associés à la cinétique de l'interface et interviennent à travers la métrique sous-jacente du système. En particulier, une phase peut donc être « figée » en imposant une mobilité nulle aux interfaces appropriées. Dans le cas du mouillage, seule l'interface Γ_{LV} est mobile et un choix de mobilités peut être

$$(\nu_{VS}, \nu_{SL}, \nu_{LV}) = (0, 0, 1).$$

afin de « fixer » la phase solide. Ce jeu de mobilités peut en fait être écrit sous la forme,

$$\frac{1}{\nu_{VS}} = \frac{1}{\nu_V} + \frac{1}{\nu_S}, \quad \frac{1}{\nu_{SL}} = \frac{1}{\nu_S} + \frac{1}{\nu_L} \quad \text{et} \quad \frac{1}{\nu_{LV}} = \frac{1}{\nu_L} + \frac{1}{\nu_V},$$

où $\nu_S = 0^+$ et $\nu_S = \nu_V = 2$ sont des coefficients positifs avec la convention $\frac{1}{0}^+ = +\infty$. On dit dans ce cas que les mobilités sont harmoniquement additives.

Pour pouvoir simuler les phénomènes de mouillage, nous étendons le modèle **NMN-CH** au cadre multiphase. Ce modèle s'obtient comme flot de gradient H^{-1} de l'énergie de Cahn-Hilliard à plusieurs phases (5.1.4) et la décomposition précédente des mobilités nous permet d'écrire,

$$\begin{cases} \partial_t u_k = \nu_k N(u_k) \operatorname{div}\left(M(u_k)\nabla\left(\sigma_k N(u_k)\mu_k + \lambda\right)\right), \\ \mu_k = \Delta u_k - \dfrac{1}{\varepsilon^2} W'(u_k), \end{cases} \tag{5.1.8}$$

où $k \in \{V, S, L\}$ et λ correspond au multiplicateur de Lagrange associé à la contrainte de partition $u_V + u_S + u_L = 1$.

Les résultats asymptotiques du modèle précédent sont similaires au cadre biphasique. En particulier, le choix de $M(s) = 2W(s)$ et $N(s) = 1/\sqrt{M(s)}$ assure que le modèle est d'ordre deux en ε. La vitesse normale V_{ij}^ε à l'interface Γ_{ij}^ε est

$$V_{ij}^\varepsilon = \nu_{ij}\sigma_{ij} c \Delta_\Gamma H_{ij} + \mathcal{O}(\varepsilon).$$

où la constante c dépend de W, M et N.

Différentes expériences numériques ont été réalisées à partir du modèle **NMN-CH** et des simulations sont présentées dans la partie numérique du chapitre II. En figure 5.6, nous présentons l'exemple du démouillage d'un tube mince sur un support solide rugueux non homogène. Dans cette expérience, la gestion de l'angle de contact se fait de manière totalement implicite grâce à l'approche multiphase.

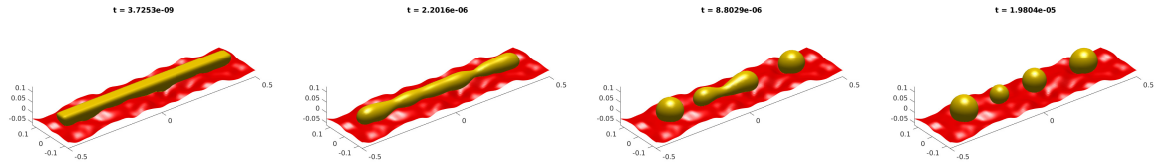


FIGURE 5.6 – Exemple de simulation d'un phénomène de démouillage en 3D d'un tube liquide mince sur un supports solides rugueux.

Enfin, notons que l'approche précédente se généralise naturellement au cas de plus de trois phases en faisant les mêmes hypothèses sur les tensions de surfaces et les mobilités c'est-à-dire que les tensions de surfaces soient additives et que les mobilités soient harmoniquement additives. On retrouve en particulier les mêmes résultats asymptotiques.

Méthode numérique et difficulté du traitement des mobilités dégénérées

D'un point de vue numérique, les modèles champ de phase étudiés dans cette partie impliquent des mobilités dégénérées qui nécessitent une adaptation des techniques numériques classiquement utilisées pour le modèle **C-CH**. Afin d'avoir des schémas numériques simples et robustes, nous avons commencé par étudier les techniques utilisées pour le modèle **C-CH** avant de les adapter aux modèles **M-CH** et **NMN-CH**.

Le cas biphasique

Schémas numériques pour le modèle C-CH. Nous commençons par présenter quelques schémas numériques simples, robustes et efficaces du modèle **C-CH** :

$$\partial_t u = \Delta\left(\frac{1}{\varepsilon^2} W'(u) - \Delta u\right).$$

Si δ_t est le paramètre, on construit une suite $(u_n)_{n\geq 0}$ d'approximation u^n de u au temps $n\delta_t$.

Une première approche. Une première approche consiste à utiliser un schéma semi-implicite en temps de sorte à pouvoir traiter de manière explicite le terme non linéaire et en implicite le terme de diffusion. On obtient alors le schéma suivant,

$$\frac{u^{n+1}-u^n}{\delta_t} = \Delta\left(-\Delta u^{n+1} + \frac{1}{\varepsilon^2}W'(u^n)\right),$$

qui s'écrit aussi sous la forme,

$$u^{n+1} = L\left(u^n + \frac{\delta_t}{\varepsilon^2}\Delta W'(u^n)\right),$$

où l'opérateur $L = \left(I_d + \delta_t\Delta^2\right)^{-1}$ peut être traité efficacement dans l'espace de Fourier simplement par multiplication du symbole $\sigma(\xi) = \dfrac{1}{1+16\pi^4\delta_t|\xi|^4}$. Numériquement, cet opérateur de diffusion peut donc être traité efficacement à l'aide de la transformée de Fourier rapide.

Le traitement explicite du terme de réaction nécessite tout de même la condition CFL $\delta_t \leq C\delta_x^4$ pour assurer la stabilité du schéma. Ici, δ_x correspond au pas d'espace et C est une constante qui dépend du potentiel W.

Méthode de relaxation. Une façon de s'affranchir de cette condition CFL est d'introduire un paramètre de relaxation α afin de stabiliser le terme de réaction. Cette idée a été introduite par Eyre [158] afin d'obtenir des schémas simples et robustes pour le modèle **C-CH**. Depuis, cette technique est énormément utilisée pour approximer divers problèmes d'évolution avec une structure de flot de gradient. L'idée consiste à décomposer l'énergie de Cahn-Hilliard en somme d'une énergie convexe P_c et d'une énergie concave P_e, puis à utiliser un traitement implicite du terme convexe et un traitement explicit du terme concave.

Notons que cette décomposition convexe-concave est toujours possible. Par exemple, on peut utiliser

$$P_c(u) = \int_\Omega \left(\frac{\varepsilon}{2}|\nabla u|^2 - \frac{\alpha}{2\varepsilon}u^2\right)dx \quad \text{et} \quad P_e(u) = \frac{1}{\varepsilon}\int_\Omega \left(W(u) - \frac{\alpha}{2}u^2\right)dx,$$

en choisissant le paramètre α suffisamment grand pour assurer la décomposition convexe-concave. Avec ce choix, on aboutit au schéma suivant,

$$\frac{u^{n+1}-u^n}{\delta_t} = \Delta\left[\left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2}u^{n+1}\right) + \frac{1}{\varepsilon^2}\left(W'(u^n) - \alpha u^n\right)\right].$$

De façon similaire au premier schéma, le terme u^{n+1} s'obtient efficacement par une approche spectrale.

On peut de plus montrer que le schéma obtenu est inconditionnellement stable, dans le sens où celui-ci fait décroître l'énergie de Cahn-Hilliard sans condition sur le pas de temps et dès que le paramètre de stabilisation α satisfait $\alpha > \sup_{s\in[0,1]}|W''(s)| = 1$. On obtient ainsi un schéma simple, stable et avec un faible coût algorithmique en utilisant une approche spectrale.

Un premier schéma pour le modèle M-CH. Une façon d'étendre le schéma précédent au modèle **M-CH** est d'utiliser un traitement explicite de la mobilité M. Le schéma qui en découle s'écrit alors,

$$\begin{cases} (u^{n+1}-u^n)/\delta_t &= \operatorname{div}\left(M(u^n)\nabla\mu^{n+1}\right), \\ \mu^{n+1} &= \left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2}u^{n+1}\right) + \frac{1}{\varepsilon^2}\left(W'(u^n) - \alpha u^n\right). \end{cases}$$

Cette approche soulève néanmoins quelques difficultés numériques car l'opérateur de diffusion qui intervient dans le terme implicite n'est plus homogène en espace. Une approche spectrale est donc difficile et la résolution de ce schéma nécessite alors la résolution d'un système linéaire. On perd donc l'efficacité de l'approche spectrale.

Relaxation sur la métrique. Une façon de contourner cette difficulté et de garder une approche spectrale est d'introduire un terme de régularisation $m > 0$ sur la métrique afin de pouvoir traiter μ en implicite lorsque la métrique est constante et en explicit dans le cas contraire. Plus précisément, on s'intéresse au schéma suivant,

$$\begin{cases} (u^{n+1}-u^n)/\delta_t &= m\Delta\mu^{n+1} + \operatorname{div}\left(\left(M(u^n) - m\right)\nabla\mu^n\right), \\ \mu^{n+1} &= \left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2}u^{n+1}\right) + \frac{1}{\varepsilon^2}\left(W'(u^n) - \alpha u^n\right). \end{cases}$$

On tire alors avantage de l'approche de splitting utilisée pour le modèle **C-CH** en faisant apparaître un opérateur homogène en espace dans la partie implicite. On obtient ainsi un schéma simple, inconditionnellement stable en pratique et rapide sans avoir recours à la résolution d'un système linéaire grâce à l'approche spectrale.

Il est intéressant de remarquer que cette approche peut être vue comme une adaptation de la méthode de splitting convexe-concave. En effet, en exploitant le côté variationnel de la métrique, le modèle **M-CH** peut se réécrire sous la forme,

$$\begin{cases} \partial_t u = -\nabla_\mu J_u(\mu), \\ \quad \mu = -\Delta u + \dfrac{1}{\varepsilon^2} W'(u). \end{cases} \tag{5.1.9}$$

où $J_u(\mu) = \frac{1}{2} \int M(u) |\nabla \mu|^2$ correspond à l'énergie associée à la métrique.

Dans ce contexte, le schéma précédent correspond alors à traiter le terme $\nabla_\mu J_u(\mu)$ de manière semi-implicite en utilisant la décomposition convexe-concave suivante de J_u :

$$J_u(\mu) = \frac{1}{2} \int m |\nabla \mu|^2 dx + \frac{1}{2} \int (M(u) - m) |\nabla \mu|^2 dx,$$

où m est choisi de telle sorte que $m = \max_{s \in [0,1]} \{M(s)\}$ afin d'assurer une décomposition convexe-concave de la métrique.

Cette approche est très intéressante car elle exploite la structure variationnelle du modèle **M-CH**. Elle est de plus compatible avec une approche spectrale en faisant apparaitre des opérateurs de diffusion homogènes en espace dans le terme implicite. Nous verrons dans la partie I comment exploiter cette idée pour traiter le modèle **NMN-CH** en remarquant que celui-ci peut aussi s'écrire sous la forme (5.1.9) où cette fois-ci l'énergie associée à la métrique est donnée par

$$J_u(\mu) = \frac{1}{2} \int M(u) |\nabla \left(N(u) \mu \right)|^2.$$

Extension au cadre multiphase

Nous proposons d'étendre les schémas numériques développés précédemment au cadre multiphase. Cette extension demande tout de même une attention particulière pour le multiplicateur de Lagrange : une approche classique où celui-ci serait traité implicitement implique la résolution d'une équation de Poisson qui nécessiterait la résolution d'un système linéaire à chaque itération. Une possibilité pour contourner cette difficulté est d'exploiter encore une fois le fait que le traitement du terme implicite est facilité par une approche spectrale, dès que les opérateurs de diffusion qui interviennent sont homogènes en espace. Une façon de faire est alors d'utiliser un traitement semi-implicite du multiplicateur de Lagrange. De cette manière, les nouveaux schémas sont traités efficacement grâce à une approche spectrale. En particulier, cela nous amène à résoudre L équations de Cahn-Hilliard à chaque pas de temps, évitant ainsi d'avoir à résoudre un système linéaire sur le multiplicateur de Lagrange dont la taille dépendrait du nombre de phases L. On réduit ainsi considérablement le coût en mémoire et en calcul en comparaison avec une approche classique. Pour plus de détails sur ces schémas nous renvoyons le lecteur à la partie I du manuscrit.

5.2 Approximation numérique de flots géométriques à l'aide de réseaux de neurones

Bien que les méthodes de champ de phase soient un outil extrêmement efficace pour approcher des évolutions géométriques d'interfaces, il reste des situations qui posent encore un certain nombre de difficultés. Nous décrivons maintenant brièvement celles auxquelles nous nous sommes intéressés dans cette thèse.

Une première difficulté concerne l'approximation champ de phase de flots d'ordre 4 comme celui de Willmore : les modèles champ de phase de cette catégorie de flots conduisent souvent à la résolution d'une équation aux dérivées partielles d'ordre 4 qui pose un certain nombre de difficultés dans leur discrétisation. Il existe par ailleurs d'autres approches de type Bence-Merriman-Osher (BMO), alternant des opérateurs de convolution et de seuillage, et qui conduisent à des schémas très simples et très efficaces malgré leur forte dépendance à la discrétisation spatiale utilisée. Une idée de départ de ce travail était donc d'essayer de trouver des algorithmes équivalents dans un contexte champ de phase, où les opérateurs de seuillage seraient remplacés par des opérateurs de réaction, l'utilisation de réseaux de neurones permettant ainsi l'identification de tels opérateurs dans la phase d'entraînement.

Une autre difficulté apparaît lorsque l'on s'intéresse au mouvement par courbure moyenne anisotrope. En effet, les modèles champ de phase introduisent dans ce cas des opérateurs de diffusion non linéaires qui posent des difficultés numériques dans leur traitement. Des approches consistant à linéariser ces opérateurs permettent de simplifier leur résolution mais les modèles champ de phase associés sont moins précis. En particulier, l'identification de bons noyaux anisotropes discrets est toujours un problème ouvert. L'intérêt d'utiliser des réseaux de neurones est donc d'essayer une nouvelle méthode pour identifier de tels noyaux. Une autre question intéressante est celle de retrouver les anisotropies à partir de ces noyaux afin d'être capable d'identifier des anisotropies à partir de données réelles.

Un dernier exemple sur lequel nous nous sommes beaucoup focalisés dans cette thèse concerne l'approximation de mouvements par courbure moyenne dans le cas non orienté. Les applications visées sont par exemple les problèmes de Steiner et de Plateau. Il existe en effet certains cas particuliers où une énergie d'interfaces non orientable est approchée par une énergie champ de phase comme celle de Ambrosio-Tortorelli [18] mais dans ce cas, l'énergie en question doit être couplée avec d'autres termes antagonistes. Une question naturelle qui se pose alors est la suivante : est-il possible de contourner cette difficulté afin d'obtenir des modèles numériques capables de simuler un simple flot par courbure moyenne d'interfaces non orientables en représentation champ de phase ? Nous verrons comment les réseaux de neurones permettent d'identifier ces modèles numériques.

Avant de préciser notre contribution numérique sur l'utilisation de réseaux de neurones pour approcher des évolutions d'interfaces géométriques, nous synthétisons dans un premier temps l'utilisation de réseaux de neurones pour la résolution d'équations au dérivées partielles.

L'apprentissage par réseaux de neurones a révolutionné une liste impressionnante de disciplines comme le traitement du signal, de l'image ou encore de texte avec des exemples très célèbres en reconnaissance d'images, de texte ou de la parole. Ces succès s'expliquent principalement par le fait que les réseaux de neurones sont capables de représenter efficacement des fonctions fortement non linéaires sur une base d'apprentissage puis de les extrapoler à de nouvelles données. Certaines architectures de réseaux sont d'ailleurs capables d'approcher des fonctions hautement complexes dont la dimension d'entrée est très grande et pour lesquelles il est en pratique très difficile d'inférer un modèle à partir d'un nombre limité de données. En partant de ce constat, il est naturel de s'interroger sur la capacité de ce type de structures à approcher des solutions d'équations aux dérivées partielles (EDP). En utilisant la technologique de l'apprentissage machine, la résolution d'EDP à partir de réseaux de neurones peut être une alternative intéressante aux méthodes traditionnelles.

5.2.1 Utilisation des réseaux de neurones pour la résolution d'équations aux dérivées partielles

Nous donnons ici un point de vue synthétique sur l'utilisation des réseaux de neurones pour les EDP. Notre objectif n'est pas de couvrir de manière exhaustive l'ensemble des techniques d'apprentissage par réseaux de neurones appliquées dans le contexte des EDP mais plutôt de donner un aperçu utile des différentes approches qui ont suscité beaucoup d'intérêt ces dernières années. Plus précisément, nous nous concentrerons sur la méthodologie générale de l'application des réseaux de neurones aux équations d'évolution en vu de les adapter à nos problèmes.

On s'intéresse aux équations de la forme

$$\begin{cases} \partial_t u(t,x) + \mathcal{L}u(t,x) = 0, \quad (t,x) \in [0,T] \times \Omega \\ u(t=0,x) = u_0(x), \quad x \in \Omega \qquad \text{(condition initiale)} \\ u(t,x) = g(t,x), \quad (t,x) \in [0,T] \times \partial\Omega \qquad \text{(condition aux bords)} \end{cases} \tag{5.2.1}$$

où $T > 0$ et $\Omega \subset \mathbb{R}^d$. Ici, $\mathcal{L}$ est un opérateur différentiel qui peut être linéaire (exemple : $\mathcal{L} = -\Delta$ pour l'équation de la chaleur) ou non linéaire (exemple : $\mathcal{L} = -\Delta + \frac{1}{\varepsilon^2} W'(\cdot)$ pour l'équation d'Allen-Cahn).

Un grand nombre de méthodes s'appuient sur le fait que les réseaux de neurones offrent des capacités d'approximations intéressantes et pratiques pour des fonctions compliquées et de régularités très variées [184, 1, 183, 182, 197]. L'idée générale de ces méthodes est d'approcher une solution de l'équation (5.2.1) par un réseau de neurones u_θ. Pour cela, le problème initial (5.2.1) est réinterprété comme un problème d'optimisation afin d'utiliser les méthodes d'apprentissage statistique et d'optimisation à grande échelle pour calibrer les poids de u_θ. Les performances du réseau sont donc mesurées à partir d'une fonction perte dont la forme dépend de l'interprétation du problème.

Première approche : un réseau, une solution particulière

Une première idée consiste à entraîner un réseau $\mathcal{U}_\theta$ de telle sorte que

$$u(x,t) \simeq \mathcal{U}_\theta(x,t), \text{ pour tout } (x,t) \in \Omega \times [0,T].$$

Dans un cadre d'apprentissage supervisée, l'entraînement pourra alors s'effectuer à partir de quelques estimations de u, comme par exemple
— une collection de N_1 données (t_i, x_i) aléatoirement dans $[0,T] \times \Omega$.
— une collection de N_2 données z_i aléatoirement dans Ω.
— une collection de N_3 données (τ_i, s_i) aléatoirement dans $[0,T] \times \partial\Omega$.
Il s'agira alors d'optimiser les paramètres du réseau θ en minimisant une fonctionnelle de la forme

$$\mathcal{L}(\theta) = \frac{1}{N_1}\sum_{i=1}^{N_1} |\mathcal{U}_\theta(t_i,x_i) - u(t_i,x_i)|^2 + \frac{1}{N_2}\sum_{i=1}^{N_2} |\mathcal{U}_\theta(0,z_i) - u_0(z_i)|^2 + \frac{1}{N_3}\sum_{i=1}^{N_3} |\mathcal{U}_\theta(\tau_i,s_i) - u(\tau_i,s_i)|^2.$$

Le premier terme correspond à l'erreur commise par rapport à la vraie solution u aux points d'échantillonnage, les second et troisième termes mesurent l'erreur commise par rapport à la condition initiale et aux conditions de bord.

Cette approche est intéressante car elle ne nécessite pas la discrétisation du domaine Ω. On parle de méthode axée sur les données (*data driven method*). Cela peut être très pratique pour deux raisons : d'une part lorsque l'on travaille dans un domaine avec une géométrie complexe pour lequel il est difficile d'avoir une discrétisation adaptée ; d'autre part, lorsque la dimension d est trop grande pour espérer résoudre l'équation (5.2.1) sur un maillage suffisamment fin avec un coût de calcul raisonnable [44]. Cependant, cette approche reste limitée à une instance spécifique de l'EDP : pour toute nouvelle condition initiale, il faut entraîner un nouveau réseau de neurones. De plus, elle requiert la connaissance de la solution sur un ensemble de points du domaines. Ce qui limite son champ d'application aux problèmes pour lesquels la solution est soit explicite soit simple à calculer sur un nombre suffisant de points du domaine de calcul. On remarque par ailleurs qu'en pratique il est très difficile avec une telle méthode de prendre en compte les contraintes physiques (loi de conservation, transport de matière, etc.) du problème. Pour que le réseau de neurones puisse apprendre ces contraintes, il faudrait un volume très important de données d'apprentissage.

Pour pallier cet inconvénient, une possibilité est d'intégrer certaines contraintes physiques liées au problème dans le processus d'apprentissage afin que le réseau soit en quelque sorte « informé » de la structure sous-jacente qui régit l'EDP. On parle alors d'approche informée par la physique (*Physics-informed approach*) et les réseaux de neurones entraînés à partir de cette approche sont appelés (*Physics-informed neural networks*, PINN).

La méthode la plus utilisée actuellement consiste à intégrer le résidu associé à l'équation (5.2.1) dans la fonction perte. La méthodologie reste identique à l'approche précédente à la différence que le réseau de neurones $\mathcal{U}_\theta$ est cette fois-ci optimisé par rapport à la fonction perte suivante,

$$\mathcal{L}(\theta) = \frac{1}{N_1}\sum_{i=1}^{N_1} |\partial_t \mathcal{U}_\theta + \mathcal{L}\mathcal{U}_\theta|^2 + \frac{1}{N_2}\sum_{i=1}^{N_2} |\mathcal{U}_\theta(0,z_i) - u_0(z_i)|^2 + \frac{1}{N_3}\sum_{i=1}^{N_3} |\mathcal{U}_\theta(\tau_i,s_i) - u(\tau_i,s_i)|^2.$$

où $\partial_t \mathcal{U}_\theta + \mathcal{L}\mathcal{U}_\theta$ est le résidu associé à (5.2.1).

En forçant la sortie du réseau à satisfaire le modèle mathématique correspondant, cette approche remplace l'exigence de gros volume de données par l'introduction d'informations provenant de la physique sous-jacente du problème.

Cette méthode est totalement non supervisée car elle ne requiert aucune information sur les solutions de l'EDP. Elle est donc moins restrictive que la première approche. Cependant elle demande un effort de calcul non négligeable durant le processus d'apprentissage avec le calcul du résidu $\partial_t \mathcal{U}_\theta + \mathcal{L}\mathcal{U}_\theta$. En pratique, on tire partie de la différentiation automatique pour le calcul des dérivées (exactes) présentes dans le résidu. Ainsi, sans avoir recours à la discrétisation du domaine de calcul, on garde une méthode totalement basée sur les données.

Depuis son introduction dans [286, 287], l'approche *informée par la physique* a suscité un grand intérêt dans la communauté scientifique. Elle a en effet l'avantage de pouvoir s'adapter à plusieurs situations très variées en prenant des formes différentes [285, 239, 188, 203, 236, 350, 317]. Dans [203] par exemple, les auteurs proposent une version variationnelle des PINN (basée sur la méthode de Petrov–Galerkin) afin de réduire la complexité de calcul des dérivées d'ordres élevés dans le processus d'apprentissage. Récemment, les chercheurs ont développé de nombreuses variantes des PINN avec des résultats prometteurs sur les problèmes inverses et les tâches partiellement observées [236, 350, 317].

L'approche informée par la physique est très attrayante pour le prototypage rapide lorsque l'efficacité et la haute précision ne sont pas les principales préoccupations. Tout comme la première approche, elle est limitée à une instance de l'EDP. Elle reste donc moins efficace que les méthodes traditionnelles dans un grand nombre de cas. En particulier parce que la résolution d'une équation algébrique est généralement plus simple que la résolution des problèmes d'optimisation à grande échelle fortement non convexes associés à l'entraînement des réseaux de neurones. En outre, de nombreuses méthodes spécialisées ont été développées au fil des années pour des problèmes spécifiques, intégrant souvent des contraintes ou des hypothèses physiques directement dans les approximations.

Deuxième approche : un réseau, un pas de temps

Une approche alternative récente consiste plutôt à apprendre l'opérateur associé à l'EDP [237, 50, 265, 19, 231, 230]. Dans cette catégorie de méthodes, la notion de réseaux de neurones est généralisée au cadre des opérateurs entre deux espaces de dimensions infinies, typiquement des espaces de fonctions dans le cas des EDP. On parle alors d'opérateur neuronal (*Neural operator*). Dans le contexte des EDP d'évolution, les opérateurs neuronaux $\mathcal{S}_\theta$ font la correspondance entre une condition initiale en entrée et la solution après une itération sur un pas de temps δ_t en sortie :

$$u(x, t_{n+1}) \simeq \mathcal{S}_\theta[u(\cdot, t_n)](x), \text{ où } u(x, t_{n+1}) = \mathcal{G}[u(\cdot, t_n)](x).$$

L'entraînement d'un opérateur de neurones ne requiert pas la discrétisation du domaine de calcul. La grande différence avec les méthodes présentées ci-dessus vient, d'une part, du fait que l'opérateur de neurones n'est entraîné qu'une seule fois : pour chaque nouvelle condition initiale, la solution (après un pas de temps δ_t) est le résultat de l'évaluation de la condition initiale par l'opérateur de neurones, ce qui réduit considérablement les problèmes de calcul majeurs rencontrés avec les méthodes précédentes. D'autre part, cette approche est entièrement axée sur les données et ne nécessite aucunement la connaissance de l'EDP sous-jacente. Ce qui peut être très avantageux lorsque nous n'avons accès qu'à des informations limitées de la structure de l'EDP, comme souvent en physique.

La méthodologie générale de cette approche s'inscrit principalement dans le cadre de l'apprentissage supervisé : l'objectif est d'approcher un opérateur noté $\mathcal{G}$ par une application paramétrique $\mathcal{S}_\theta$. Pour cela, on estime $\mathcal{S}_\theta$ sur une collection de données $(u_i, \mathcal{G}[u_i])_{1 \leq i \leq N}$ où les u_i sont des fonctions représentant différentes conditions initiales de l'EDP.

En pratique, chaque fonction u_i est évaluée sur une collection de points $\{x_j^i\}_{1 \leq j \leq M}$ et l'entraînement de $\mathcal{S}_\theta$ consiste alors à minimiser un risque empirique de la forme,

$$\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} \ell\left(S_\theta[u_i](x_j^i), \mathcal{G}[u_i](x_j^i)\right),$$

où ℓ est une métrique donnée. Son choix dépend du problème mais très souvent il s'agit du coût quadratique $\ell(x, y) = |x - y|^2$.

Approximer un opérateur $\mathcal{G}$ est une tâche différente et typiquement beaucoup plus difficile que de trouver la solution spécifique d'une EDP. Néanmoins, une fois que le réseau $\mathcal{S}_\theta$ est entraîné, les solutions de

l'EDP sont obtenues simplement en appliquant le réseau $\mathcal{S}_\theta$. Ce qui est donc bien moins coûteux et plus rapide en comparaison avec les méthodes précédentes. Un autre avantage de cette approche est que toute la dynamique et la physique de l'EDP sont encodées à travers l'opérateur $\mathcal{G}$. Ainsi, apprendre $\mathcal{G}$ nous permettrait d'apprendre la physique sous-jacente du problème. On tire ainsi parti des mêmes avantages des deux approches précédentes sans être contraint à une instance spécifique de l'EDP.

Pour que cette approche soit efficace et puisse être compétive par rapport aux méthodes traditionnelles, il y a deux aspects principaux à prendre en compte : les données et le modèle.

La génération de données pour ce type de problèmes peut être une tâche très ardue. Il faut que les données soient suffisamment riches et variées pour que le modèle puisse se généraliser correctement à de nouvelles données. Pour cela, il faut avoir accès à un dispositif numérique ou expérimental qui puisse générer des données suffisamment fiables. Ce qui n'est pas toujours facile en pratique dans les problèmes faisant intervenir des EDP.

Comme toujours en apprentissage, le choix de l'architecture utilisée est crucial pour obtenir des modèles de prédiction consistants. Ici, il s'agit de concevoir des modèles pour l'apprentissage d'opérateurs de solutions d'EDP très complexes.

Très récemment, plusieurs architectures de réseaux ont été proposées pour approcher des opérateurs avec des applications aux équations différentielles [237, 50, 265, 19, 231, 230]. Dans [237] par exemple, les auteurs introduisent *DeepOnet*, un opérateur de neurones dont la construction est inspirée par le théorème d'approximation universelle des opérateurs [93, 238]. DeepOnet est composé de deux réseaux (voir 5.7), l'un codant la fonction d'entrée u à des points fixes (*branch net*), l'autre codant la localisation de la fonction de sortie y (*trunk net*).

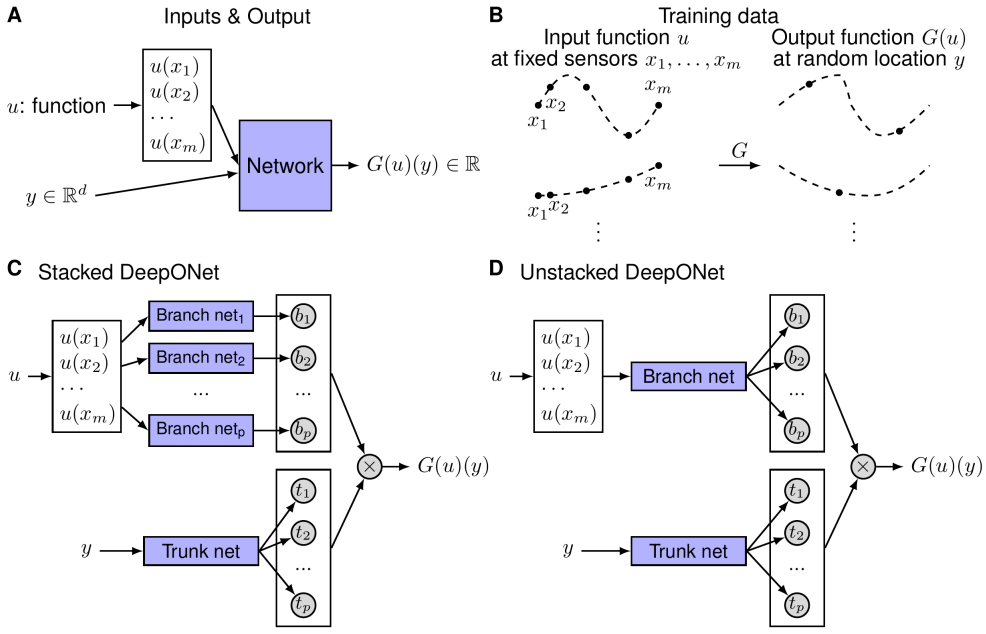


FIGURE 5.7 – Architecture de DeepOnet tirée de [237]. (A) Pour approcher $\mathcal{G}[u](y)$, le réseau prend deux entrées : un vecteur $(u(x_1), \cdots, u(x_m))$ sur la fonction d'entrée u et un point y pour évaluer la fonction de sortie $G[u](y)$. (B) Illustration des données d'apprentissage. On représente en deuxième ligne une version empilée (C) et non empilée (D) de DeepOnet.

DeepOnet à l'avantage d'avoir une architecture très flexible : ses deux composantes *branch net* et *trunk net* peuvent être n'importe quel réseau qui peut s'appliquer à un vecteur. Cela peut être de simples MLP ou encore des réseaux convolutionnels. DeepOnet est aussi versatile et trouve son application dans diverses problèmes [237, 121, 174, 343, 329].

L'opérateur neuronal de Fourier (*Fourier neural opérateur*, FNO) [230], est un autre opérateur de neurones de plus en plus utilisé de nos jours. Basé sur la transformée de Fourier (Voir 5.8), FNO permet d'avoir des modèles de prédictions efficaces et rapides [230, 180, 333, 344] grâce à l'utilisation de la transformée de Fourier rapide.

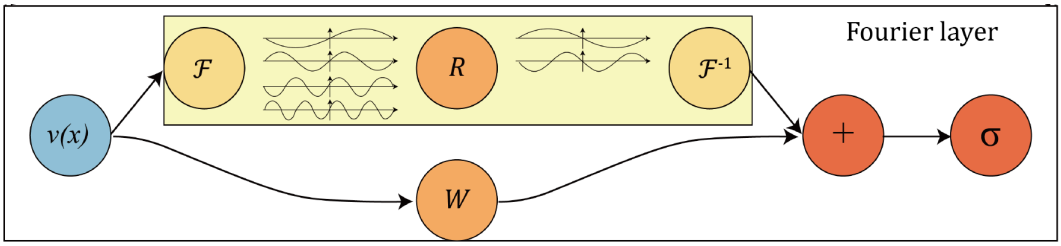


FIGURE 5.8 – Bloc de base d'un opérateur de neurones de Fourier tiré de [230]. Le réseau est paramétré à la fois dans l'espace de Fourier (à travers le réseau R) et le domaine de référence (par le poids W et un biais b). Pour une entrée x, la sortie est donnée par $y = \sigma\left(\mathcal{F}^{-1}\left(R[\mathcal{F}(x)]\right) + Wx + b\right)$.

D'autres opérateurs de neurones ont été proposés dans la littérature comme les réseaux basés sur les modèles de réduction (PCA ou auto-encoder) [50] ou encore des réseaux sur des graphes [19, 231]. Mais d'un point de vue pratique, les opérateurs de neurones FNO et DeepOnet sont plus simples à appliquer dans diverses situations.

5.2.2 Contributions de la thèse

Comme nous venons de l'expliquer, il existe un certain nombre d'approches dans la littérature pour résoudre numériquement des EDPs à l'aide de réseaux de neurones. Notre objectif initial est d'utiliser ces réseaux de neurones pour simuler des modèles de champ de phase et leur discrétisation de manière plus précise et plus rapide.

Par exemple, dans le cas le plus simple du mouvement par courbure moyenne, on pourrait envisager d'approcher numériquement les solutions de l'équation d'Allen-Cahn

$$u_t = \Delta u - \frac{1}{\varepsilon^2} W'(u),$$

afin d'obtenir des approximations de mouvement par courbure moyenne d'ordre 2 suivant le développement asymptotique suivant

$$u(x,t) = q\left(\frac{d(x,\Omega(t))}{\varepsilon}\right) + O(\varepsilon^2).$$

L'idée serait alors de suivre la deuxième approche consistant à introduire un réseau de neurones $\mathcal{S}_\theta$ pour approcher le semi-groupe associé à l'équation d'Allen-Cahn

$$u^{n+1} \simeq \mathcal{S}_\theta[u^n].$$

Ce réseau agirait comme un schéma numérique à un pas de temps et pourrait remplacer les discrétisations numériques classiques des modèles de champs de phase. Cependant, le problème d'une telle approche est qu'elle nécessiterait des réseaux de neurones extrêmement simples pour pouvoir être compétitive avec les schémas classiques de discrétisation de nos EDPs. Elle ne permettrait pas a priori de réduire les erreurs d'approximation de champ de phase d'ordre 2 en ε^2. Et enfin, cette approche serait limitée aux flots d'interface pour lesquels il existe une approximation champ de phase, ce qui n'est par exemple pas le cas pour les mouvements par courbure moyenne non orienté.

Afin d'obtenir une méthodologie plus flexible, nous avons développé une stratégie qui n'utilise pas l'équation d'Allen-Cahn mais seulement la représentation champ de phase de l'ensemble évoluant selon le flot de courbure moyenne

$$v(x,t) = q\left(\frac{d(x,\Omega(t))}{\varepsilon}\right).$$

Ainsi, l'idée a été d'entraîner les réseaux de neurones $\mathcal{S}_\theta$ non pas à partir de solutions ou de résidus de l'équation d'Allen-Cahn, mais plutôt sur des solutions exactes du mouvement par courbure moyenne en représentation champ de phase v.

Concernant la structure des réseaux de neurones $\mathcal{S}_\theta$ que nous avons utilisés, elle est fortement inspirée des schémas de résolution de l'équation d'Allen-Cahn alternant des neurones de diffusion (un noyau de convolution) avec des neurones de réaction (un perceptron multicouche). L'avantage d'une telle approche est que les réseaux de neurones, une fois entrainés, ont le même ordre de coût algorithmique qu'un schéma

classique de discrétisation des modèles de champs de phase, ce qui n'aurait pas été le cas avec des réseaux de type CNN ou Unet. Les réseaux pourront aussi apprendre à corriger à la fois les erreur de modélisation champ de phase et celles de discrétisations numériques, conduisant ainsi à des résolutions plus précises que les meilleurs schémas de discrétisation de l'équation d'Allen-Cahn. Enfin, comme l'apprentissage n'exploite pas le modèle de champ de phase porté par l'équation d'Allen-Cahn, il peut être généralisé à de nombreux flots géométriques, comme par exemple le flot de courbure moyenne non orienté.

Une structure de réseau de neurones adaptée au modèle de champ de phase

Notre première contribution a donc consisté à développer des structures de réseaux $\mathcal{S}_\theta$ très simples (avec peu de paramètres) et très efficaces en nous inspirant des schémas numériques de discrétisation de l'équation d'Allen-Cahn. Dans le cas de l'approche semi-implicite suivante

$$u^{n+1} = (Id - \delta_t \Delta)^{-1}(u^n - \frac{\delta_t}{\varepsilon^2} W'(u^n)),$$

le schéma alterne un opérateur de réaction

$$u^{n+1/2}(x) = \rho(u^n(x)), \text{ où } \rho(s) = s - \frac{\delta_t}{\varepsilon^2} W'(s),$$

avec un opérateur de convolution

$$u^{n+1}(x) = K * u^{n+1/2}(x), \text{ où le noyau } K = \mathcal{F}^{-1}\left[\frac{1}{1 + 4\pi^2|\xi|^2\delta_t}\right](x).$$

Nous avons donc commencé par introduire deux types de neurones : un neurone de diffusion D consistant simplement à appliquer un noyau de convolution K (de taille 17^2) et un neurone de réaction R constitué d'un perceptron à deux couches cachées avec respectivement huit et trois neurones gaussiens.

Un premier réseau s'obtient alors naturellement en alternant un neurone de diffusion avec un neurone de réaction. Ce premier réseau, noté $\mathcal{S}^{NN}_{\theta,1}$, contient 336 paramètres à optimiser, ce qui est très faible par rapport aux réseaux Unet classiques avec plusieurs millions de degrés de liberté.

Par la suite, nous avons complexifié la structure de ce premier réseau en nous inspirant des schémas de résolution de l'équation d'Allen-Cahn d'ordre supérieur. Un deuxième réseau, noté $\mathcal{S}^{NN}_{\theta,2}$, est par exemple illustré à la figure 5.9. Il contient 724 paramètres optimisables.

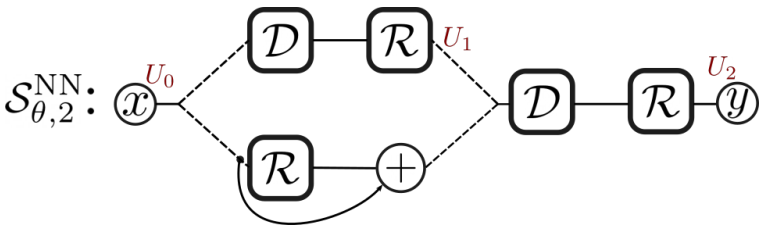


FIGURE 5.9 – Représentation du réseau $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. On adopte le même formalisme que pour les MLP en assimilant les réseaux $\mathcal{D}$ et $\mathcal{R}$ à des neurones. Ici tous les blocs $\mathcal{D}$ et $\mathcal{R}$ sont des neurones indépendants, i.e. leurs paramètres sont indépendants. Les traits en pointillés signifient que la sortie du neurone est multipliée par un poids avant d'être appliquée au neurone suivant. La ligne courbée signifie que le réseau $\mathcal{R}$ est équipé d'une structure résiduelle.

Approximation du mouvement par courbure moyenne dans le cas orienté et applications

Dans une deuxième contribution, nous avons entraîné nos réseaux à reproduire un mouvement par courbure moyenne dans le cas orienté à partir d'une base de données constituée essentiellement de l'évolution par courbure moyenne de cercles. Bien que cette base d'apprentissage soit relativement réduite, non seulement les réseaux ont réussi à apprendre de manière qualitative les flots pour reproduire différents mouvements par courbure moyenne, mais des estimations d'erreurs quantitatives ont aussi montré que cette approche numérique permettant d'obtenir des approximation de mouvement par courbure moyenne était plus précise que l'utilisation des schémas de discrétisations de l'équation d'Allen-Cahn.

Enfin, ces réseaux entraînés ont pu être ré-exploités dans des contextes multiphasiques et couplés avec des contraintes supplémentaires, ce qui illustre bien leur robustesse.

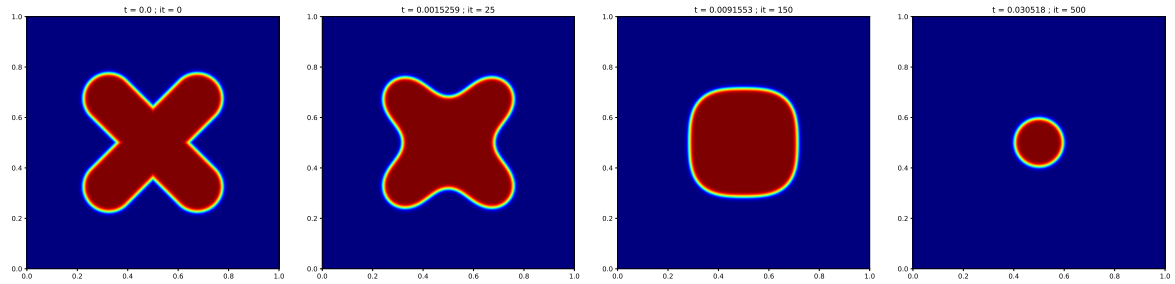


FIGURE 5.10 – Approximation du mouvement par courbure moyenne d'un ensemble non convexe en utilisant le réseau $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$. On affiche la solution u^n à différents instants t_n.

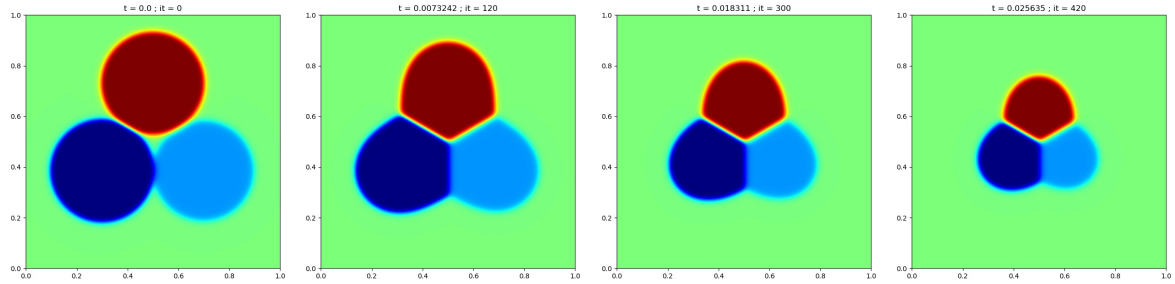


FIGURE 5.11 – Flot par courbure moyenne multiphasique de trois cercles contigus.

Approximation du mouvement par courbure moyenne dans le cas non orienté et applications

Notre principale contribution a ensuite été d'appliquer notre approche dans le cas de mouvements par courbure moyenne non orientés pour lesquels il n'existe pas, à notre connaissance, de modèle de champ de phase. Il est important de noter que la méthodologie reste identique au cas orienté. La seule différence vient de la représentation champ de phase utilisée qui ne doit pas dépendre de l'orientation de l'ensemble Ω. Nous avons alors proposé de remplacer simplement le profil q par sa dérivée q' conduisant à des représentations champ de phase de la forme

$$v(x,t) = q'\left(\frac{d(x,\Omega(t))}{\varepsilon}\right).$$

Contrairement au cas orienté, seul le deuxième réseau $\mathcal{S}^{NN}_{\theta,2}$ nous a permis n'obtenir des approximations stables de flots par courbure moyenne.

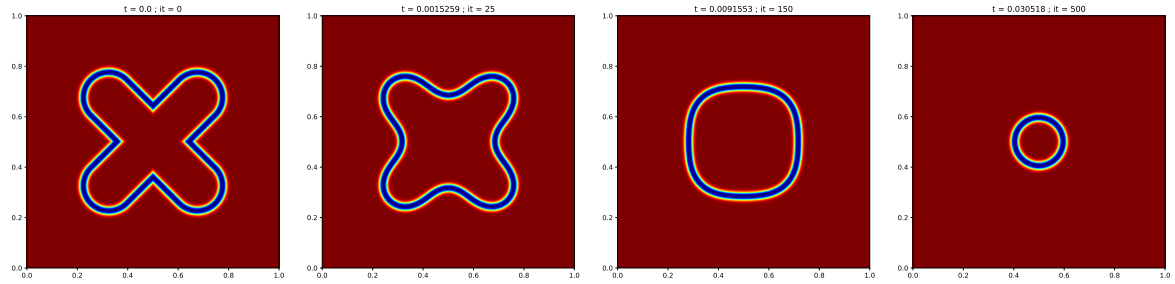


FIGURE 5.12 – Approximation du mouvement par courbure moyenne d'un ensemble non convexe en utilisant le réseau $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ entraîné sur les cercles en représentation champ de phase $q'(dist(x,\Gamma)/\varepsilon)$. On affiche la solution u^n à différents instants t_n.

Les résultats sont d'autant plus étonnants que le réseau parvient, même en présence de points triples, à reproduire la bonne évolution d'interfaces, ce qui témoigne de sa faculté à généraliser correctement le mouvement par courbure moyenne de cercles. Enfin, l'utilisation de nos réseaux pré-entrainés en dimension 2 et 3, couplés à des contraintes supplémentaires d'inclusion-exclusion, nous a aussi permis d'obtenir des bonnes approximations numériques des solutions du problème de Steiner en dimension 2 et de Plateau en dimension 3.

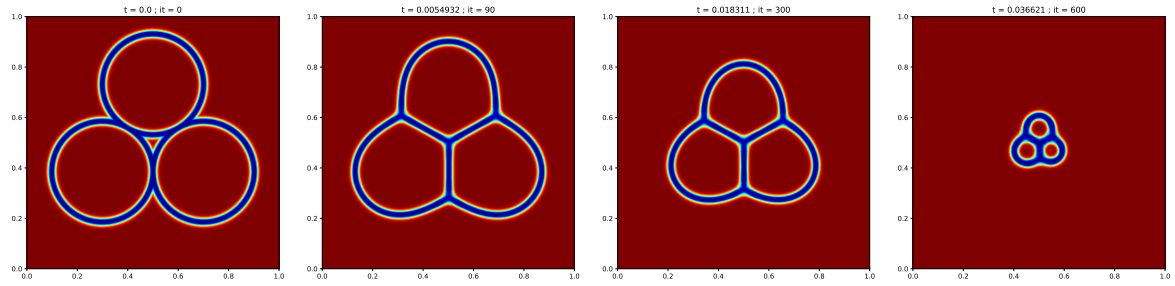


FIGURE 5.13 – Approximation du mouvement par courbure moyenne d'un ensemble non orienté. On affiche la solution u^n à différents instants t_n.

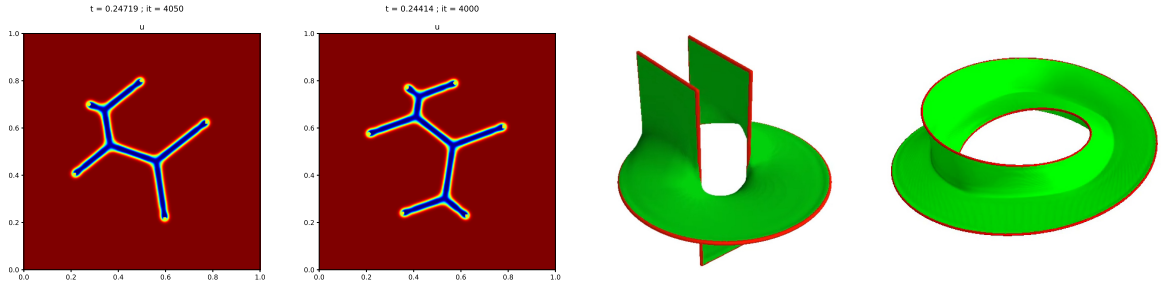


FIGURE 5.14 – Exemples numériques d'approximations d'arbres de Steiner en $2D$ et de surfaces minimales en 3D. Ces approximations sont obtenues en couplant le flot de courbure moyenne non orienté avec une contrainte d'inclusion-exclusion.

Approximation d'autres mouvements géométriques : flot de Willmore et flot de courbure moyenne anisotrope

Dans des travaux toujours en cours, nous avons essayé d'adapter les approches précédentes au cas du flot de Willmore. La difficulté supplémentaire est qu'il semble difficile d'apprendre le flot de Willmore uniquement à partir de l'évolution d'un cercle, l'information sur le laplacien surfacique de la courbure n'y apparaissant pas. La deuxième difficulté vient du comportement du flot en présence de singularités, qui n'est pas le même selon le modèle de champ de phase considéré. Cela suggère en effet différents types d'évolution possibles. Nous avons ainsi cherché à savoir comment influencer l'apprentissage pour s'autoriser ou pas la présence de singularités.

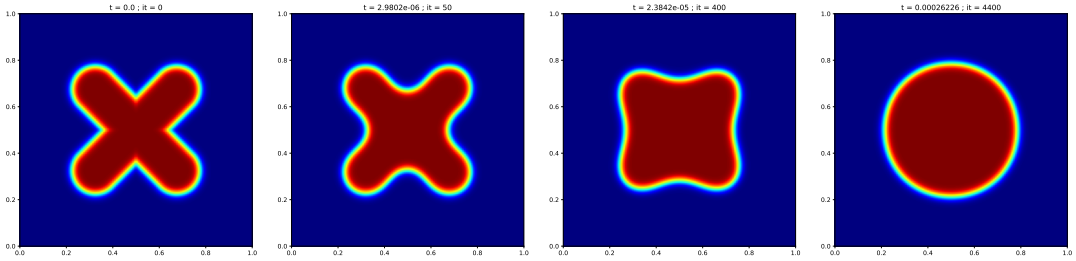


FIGURE 5.15 – Exemple d'approximation d'un flot de Willmore obtenue à l'aide d'un réseau de neurones

Un autre travail en cours concerne l'approximation de mouvements par courbure moyenne anisotrope. Ici aussi, l'idée était d'entraîner nos réseaux à partir de l'évolution des formes de Wulff qui est l'analogue dans le cas anisotrope de l'évolution des cercles dans le cas isotrope. Les premiers résultats montrent que tout se passe relativement bien pour des anisotropies régulières avec des réseaux qui arrivent à reproduire de bonnes approximations de mouvements par courbure moyenne anisotrope. Une application très intéressante est l'identification de l'anisotropie à partir des noyaux de convolution K qui ont été appris par le réseau.

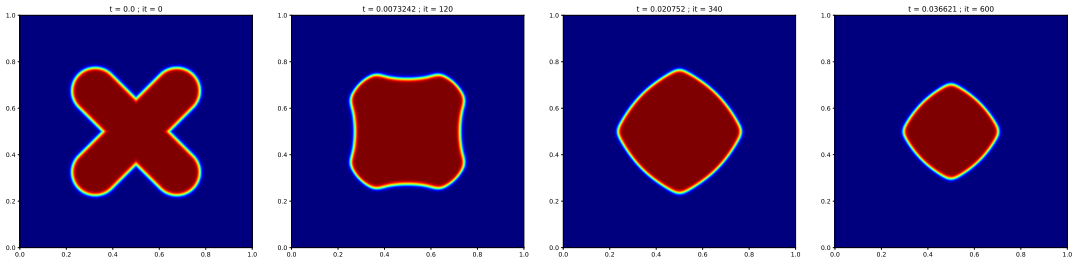


FIGURE 5.16 – Exemple d'approximation d'un flot de mouvement par courbure moyenne anisotrope obtenue à l'aide d'un réseau de neurones

Première partie

Approximations champ de phase du flot de diffusion de surface et simulation de phénomènes de mouillage

Chapitre 6

Introduction

Dans cette première partie, nous présentons les résultats de nos recherches concernant la modélisation de champ de phase de flot de diffusion de surface. Une question qui nous avait été posée dans le cadre de l'ANR Beep concernait l'analyse des conditions de mouillage d'une structure solide composée d'une forêt de nanofils. Ce projet d'ANR pluridisciplinaire porté par José Penuelas (Institut de nanotechnologie de Lyon) s'intéresse plus globalement à la production et au stockage d'hydrogène.

Les premiers modèles de champ de phase utilisés dans ce cadre ont malheureusement montré un certain nombre de limites, rendant leur utilisation inadaptée pour les applications visées. L'objectif de ce travail était donc de comprendre l'origine des pertes de précisions observées dans les expériences et de proposer de nouveaux modèles de champ capables d'approcher de manière précise des flots de diffusions de surface sur des structures complexes.

Ce travail a été rédigé dans deux articles différents présentés in extenso dans ce manuscrit.

Le premier article, intitulé, "Approximation of surface diffusion flow : a second order variational Cahn-Hilliard model with degenerate mobilities", a été écrit en collaboration avec mes directeurs de thèse et Arnaud Sengers qui était en post-doctorat dans l'ANR Beep. Il concerne l'approximation de diffusion de surface à une seule phase. Nous expliquons en particulier pourquoi les modèles classiques, c'est à dire les modèles de Cahn-Hilliard avec ou sans mobilités dégénérées, présentent des ordres d'approximation d'ordre 1 seulement en ε, la conséquence étant l'observation de pertes de volume conséquentes dans les différentes expériences numériques. Nous proposons alors un nouveau modèle à deux mobilités dégénérées, qui conserve une structure variationnelle et dont l'analyse asymptotique montre cette fois-ci un ordre de précision d'ordre 2 en ε. Ce gain réduit alors de manière drastique les pertes de volumes précédentes et permet d'approcher efficacement l'évolution par diffusion de surface d'une structure fine. Ce résultat est illustré par de nombreuses expériences numériques.

Dans une deuxième contribution correspondant à l'article "A multiphase Cahn-Hilliard system with mobility for the simulation of wetting", toujours en collaboration avec mes directeurs de thèse, Arnaud Sengers et Roland Denis (CNRS, Institut Camille Jordan), nous nous intéressons à un modèle de champ de phase qui prend en compte le support solide plus ou moins rugueux. L'idée était alors d'étendre les résultats précédents dans un cadre multiphase (solide, liquide, vapeur) où nous avons en particulier introduit des coefficients de mobilité associés à chacune des phases afin de pouvoir modéliser l'aspect figé de la phase solide. L'étude asymptotique de notre modèle de Cahn-Hilliard multiphase n'a pas posé de réelle difficulté supplémentaire. En revanche, l'implémentation numérique a nécessité une légère adaptation, notamment dans le traitement du multiplicateur de Lagrange associé à la contrainte de partition. Encore une fois, de nombreuses expériences numériques ont conclu ce travail en montrant l'avantage de notre approche par rapport aux modèles pré-existants.

Chapitre 7

Approximation of surface diffusion flow : a second order variational Cahn-Hilliard model with degenerate mobilities

7.1 Introduction

The surface diffusion flow in $\mathbb{R}^d$ (we will work here in dimension $d = 2$ or 3) is the motion of a time-dependent immersed, oriented $(d-1)$-surface without boundary whose normal velocity is given by

$$V(t) = \Delta_{\Gamma(t)} H(t),$$

where $H(t)$ is the scalar mean curvature on $\Gamma(t)$ and $\Delta_{\Gamma(t)}$ is the Laplace-Beltrami operator defined on the surface [262, 114, 323, 83, 140, 144, 145, 249]. We focus on surface diffusion flows where the surface remains embedded over a time interval $[0, T]$ ($T > 0$), we choose the orientation convention which makes the scalar mean curvature on a sphere positive and, accordingly, V is a positive outer normal velocity when the domain enclosed by $\Gamma(t)$ grows. We address in this chapter the approximation of such surface diffusion flows with a phase field method. The starting point of our approach is the classical Cahn–Hilliard equation

$$\varepsilon^2 \partial_t u = \Delta \left(W'(u) - \varepsilon^2 \Delta u \right),$$

where $u : (x, t) \mapsto u(x, t)$ is a smooth function whose level surface $\{u(\cdot, t) = \frac{1}{2}\}$ approximates $\Gamma(t)$, $\varepsilon > 0$ is a small parameter, and W is a reaction potential, typically $W(s) = \frac{1}{2} s^2 (1 - s)^2$.

The Cahn–Hilliard equation has been introduced as a mathematical model for phase separation and phase coarsening in binary alloys [79, 82], but it has also been used for applications as diverse as the modeling of two evolving components of intergalactic material or the description of a bacterial film, see the references in [268], or the modeling of multiphase fluid flows [61, 63]. More recently it was proposed as an inpainting model in image processing, see [47, 78, 99]. We refer to [268] for an inspiring general introduction to the Cahn–Hilliard equation, see also the recent book [257] where state-of-art results and many applications of the Cahn–Hilliard equation are presented.

Sharp interface limit and mobilities

Pego determined with formal arguments in [280], and Alikakos et al proved rigorously in [6], that the sharp limit flow of the Cahn–Hilliard equation (for suitable time regimes as $\varepsilon \to 0$) is the Mullins-Sekerka interface motion.

Observe now that the Cahn–Hilliard equation can be equivalently written as

$$\varepsilon^2 \partial_t u = \operatorname{div} \left(M(u) \nabla (W'(u) - \varepsilon^2 \Delta u) \right) \tag{7.1.1}$$

with the particular choice $M(u) \equiv 1$. If M is now chosen to be non constant, it plays the role of a concentration-dependent mobility. Cahn et al. showed formally in [81] that if one uses a degenerate mobility $M(u) = u(1 - u)$ (degenerate in the sense that there is no motion where $u = 0$ or 1) and a logarithmic potential

$$W(s) = \frac{1}{2} \theta \left[s \ln(s) + (1 - s) \ln(1 - s) \right] + \frac{1}{2} s(1 - s),$$

then the sharp limit motion is the surface diffusion flow. However, the singularity of such a logarithmic potential makes the model not well suited for numerical simulations. We shall see in this paper that a different model can be proposed which leads to the surface diffusion flow as well, but uses instead the smooth

quartic potential

$$W(s) = \frac{s^2(1-s)^2}{2}.$$

The choice of appropriate degenerate mobility and potential is important. It was observed in the review paper [181] that some choices lead to inconsistencies, in the sense that depending on how terms are identified in the matched asymptotic analysis, one can either show the convergence to surface diffusion flow, or to a stationary flow with null velocity. The authors of [220, 221] suggested that such inconsistencies come from the presence of an additional bulk diffusion term in the limit motion, i.e. the limit velocity is :

$$V = \frac{2}{3}\Delta_\Gamma H + \alpha H \partial_n H.$$

This has been corroborated numerically in [110, 111] where undesired coarsening effects are observed. The additional term in the velocity depends on the derivative of the mobility $M'(u_0)$, where u_0 is the outer solution in the matched asymptotics which equals either 0 or 1. To obtain a pure motion by surface diffusion, one needs to take a higher order mobility, for example $M(s) = s^2(1-s)^2$ [221]. With such a choice, the bulk diffusion appears in higher order terms and the correct limit velocity is recovered (with a different multiplicative constant), i.e.,

$$V = \alpha \Delta_\Gamma H.$$

These conclusions have been extended to the anisotropic case in [136].

Positivity property and approximation order

We now turn to the following question : starting from an initial $u(0,\cdot)$ with values in $[0,1]$ and using one of the above mobilities, does the solution u to (7.1.1) remain valued in $[0,1]$? This is often referred to as the *positivity condition* as it implies that all phase functions remain positive in a multiphase context. This condition is important because it means the function remains within the pure state phase boundaries.

The theoretical results of [221, 301] and the numerical evidences of [110, 111] establish that it is in general not the case with the Cahn–Hilliard model (7.1.1) with smooth quartic potential (typically $W(s) = s^2(1-s)^2$) and different choices of mobility M. More precisely, the profile of the solution always shows some oscillations when reaching the pure states (i.e. 0 and 1 for the aforementioned potential). This comes from the influence in the asymptotic expansion of the solution of the first order error term which does not vanish for this class of phase field models.

Using instead the logarithmic potential $W(s) = \frac{1}{2}\theta\left[s\ln(s) + (1-s)\ln(1-s)\right] + \frac{1}{2}s(1-s)$ and the mobility $M(u) = u(1-u)$ extended by 0 on $\mathbb{R}$, it is shown in [139] that there exists a weak solution that satisfies the positivity property whenever the initial condition does. However, as previously mentioned, the use of this logarithmic potential has numerical limitations due to its non smoothness.

If using a smooth quartic potential is better from a numerical perspective but prevents the positivity property to be fulfilled, is there at least a way to reduce the approximation error so to be closer to positivity ? This is the case with the non variational model introduced in [288] :

$$\begin{cases} \varepsilon^2 \partial_t u = \operatorname{div}\left(M(u)\nabla\mu\right) \\ g(u)\mu = W'(u) - \varepsilon^2 \Delta u \end{cases} \tag{7.1.2}$$

with $g(s) = \gamma |s|^p |1-s|^p$, $p \geq 0$.

The additional degenerate term g acts as a diffusion preventing term and forces the approximation error to be smaller and to converge to zero far from the interface. This model is known to achieve better numerical accuracy than the classical model (7.1.1), and it has been successfully used in various applications, see for example [4, 263, 299, 298].

Several choices have been made for p, the most popular ones being $p = 1, 2$, actually more for their numerical performances rather than on the basis of theoretical arguments. However it can be observed with calculations that the choice $p = 1$ yields the same second order approximation error as with the model we introduce in this paper. In other words, the positivity property is not satisfied but there is an improvement toward it.

While it has excellent numerical properties, the above model (7.1.2) does not derive from an energy. Its theoretical analysis is therefore delicate and it can hardly be extended to multiphase applications. This motivated the variational adaptation proposed in [301] :

$$\begin{cases} \varepsilon^2 \partial_t u = \operatorname{div}\left(M(u)\nabla\mu\right) \\ \mu = g(u)W'(u) - \varepsilon^2 \operatorname{div}\left(g(u)\nabla u\right) + g'(u)\left(W'(u) + \dfrac{\varepsilon^2}{2}|\nabla u|^2\right) \end{cases}$$

The idea is to inject the second degeneracy g in the energy. This model conserves the same advantages as the non variational version, in particular that the choice $p = 1$ is more interesting for it improves the

approximation order. However the new form of the energy is not well suited for the extension to complex multiphase applications and to anisotropy. Also, it seems more appropriate to incorporate the mobility in the metric rather than in the geometry of the evolution problem. This is what we propose in this paper.

A new variational Cahn–Hilliard model of second order
The idea to modify a phase field model to eliminate the first order term in the approximation error was successfully introduced for the Allen-Cahn equation in [95, 168]. Like in [301], we want to approximate the surface diffusion flow using a second order variational phase field model, but we want it closer to the original Cahn–Hilliard model.

The new Cahn–Hilliard model we propose reads as :

$$\begin{cases} \varepsilon^2 \partial_t u &= N(u)\,\mathrm{div}\,(M(u)\nabla(N(u)\mu)) \\ \mu &= W'(u) - \varepsilon^2 \Delta u, \end{cases}$$

and at least in the case where

$$W(s) = \frac{1}{2}s^2(1-s)^2, \quad M(s) = s^2(1-s)^2, \quad \text{and} \quad N(s) = \frac{1}{s(1-s)},$$

we will show that this model is of second order and approximates (at least formally) the surface diffusion flow. It does not satisfy the positivity property, but the second order approximation yields good numerical profiles, see the numerical section 7.5 for more details. A key property of our model is that it derives from the correct physical energy, only the metric used for the gradient flow is modified.

Volume conservation is a key feature of the Cahn–Hilliard equation (and of the surface diffusion flow in the embedded case), and we will discuss how different models manage to preserve this property, at least approximately. We will show that the higher consistency of the solution profile obtained with our model allows a very good approximate volume conversation, as in [65, 75].

Outline of this chapter. This chapter is organized as follows : first, we review in Section 7.2 the properties of the Cahn–Hilliard model with mobility and the drawbacks that need to be addressed. In Section 7.3 we present our new variational Cahn–Hilliard model and describe its properties. We prove these properties in Section 7.4 using the formal method of matched asymptotic expansion. The necessary tools are presented at the beginning of the proof. In the numerical section 7.5, we first explain how to derive a simple and efficient scheme using a convex splitting of the Cahn–Hilliard energy and exploiting the variational mobility structure. Lastly, we provide some numerical experiments that compare the various Cahn–Hilliard models and highlight the advantages of our new model.

7.2 Review of the properties of the Cahn–Hilliard equation with mobility

In this section we summarize the properties of various models and motivate the introduction of a new one. We observe first that surface diffusion can be obtained as the sharp limit of the Cahn–Hilliard equation with mobility whenever the mobility is of sufficiently high order.

7.2.1 The Cahn–Hilliard model with mobility to approximate surface diffusion flow

We recall that the outer normal velocity associated with the surface diffusion flow of a time-dependent smooth compact surface Γ without boundary is

$$V = \Delta_\Gamma H.$$

We will assume for the rest of the paper that Γ is contained at all times of its evolution in an open, bounded, and smooth domain Q such that $\mathrm{dist}(\Gamma, \partial Q) > 1$. If Ω denotes the open domain enclosed by Γ, a phase field associated with Γ is a smooth function approximating the characteristic function $\mathbb{1}_\Omega$ of Ω. Given a double well potential W vanishing at 0 and 1, a natural phase field is $u_\varepsilon = q(\frac{d(\cdot,\Omega)}{\varepsilon})$ where $q : \mathbb{R} \to [0,1]$ is the so-called *optimal profile* associated with W, ε represents the thickness of the smooth transition of u_ε from 0 to 1, and $d(\cdot,\Omega)$ denotes the signed distance function to Ω with the convention that $d(\cdot,\Omega) < 0$ in Ω. The optimal profile q associated with the particular choice $W(s) = \frac{1}{2}s^2(1-s)^2$ satisfies the following properties :

$$\begin{cases} q(z) = \dfrac{1 - \tanh(\frac{z}{2})}{2} \\ q'(z) = -\sqrt{2W(q)} \\ q''(z) = W'(q) \end{cases} \tag{7.2.1}$$

We will see that the velocity of the sharp limit flow depends on the following constants :

$$\begin{cases} c_W = \displaystyle\int_{\mathbb{R}} (q'(z))^2 dz = \frac{1}{6} \\ c_M = \displaystyle\int_{\mathbb{R}} M(q(z)) dz. \end{cases} \tag{7.2.2}$$

The choice $M(s) = s(1-s)$ gives $c_M = 1$, and the choice $M(s) = s^2(1-s)^2$ leads to $c_M = c_W = \frac{1}{6}$.

We start off with the classical Cahn–Hilliard model with potential $W(s) = \frac{s^2(1-s)^2}{2}$ and non negative mobility M depending on u, that we refer to as **M-CH** from now on :

$$\begin{cases} \varepsilon^2 \partial_t u = \operatorname{div}(M(u)\nabla\mu) \\ \mu = W'(u) - \varepsilon^2 \Delta u \end{cases} \qquad \textbf{(M-CH)} \tag{7.2.3}$$

It is an extension of the model

$$\begin{cases} \varepsilon^2 \partial_t u = \operatorname{div}(M\nabla\mu) \\ \mu = W'(u) - \varepsilon^2 \Delta u \end{cases} \tag{7.2.4}$$

where the mobility M is a scalar positive weight independent of u. We recall that this latter equation is the H^{-1} gradient flow of the Cahn–Hilliard energy

$$E(u) = \int_Q (\frac{\varepsilon}{2}|\nabla u|^2 + \frac{1}{\varepsilon} W(u)) dx \tag{7.2.5}$$

when considering the following scalar product in $H_0^1(Q)$ weighted by the mobility M :

$$\langle f, g \rangle_{H_0^1} = \int_Q M \nabla f \cdot \nabla g \ dx. \tag{7.2.6}$$

It is important to note that the mobility is incorporated in the metric used for computing the gradient flow, and not as a geometric parameter in the energy.

The **M-CH** model has been extensively studied and it is well understood that the mobility needs to be a polynomial with valuation at least 2 (i.e. the lowest degree of its non-zero monomials). Indeed, a polynomial mobility with valuation 1 would give a quicker motion (the phase field being valued essentially in $[0,1]$) but, as already mentioned, if was shown in [221] that it also yields an undesired bulk diffusion term in the limit velocity.

7.2.2 Properties of the classical Cahn–Hilliard model with mobility

The properties of **M-CH** are summarized below, see [221] :

Proposition 7.2.1. *If $M(s) = s^2(1-s)^2$, the solution u_ε to* (7.2.3) *expands formally near the interface $\Gamma_\varepsilon(t) = \left\{u_\varepsilon(t,\cdot) = \frac{1}{2}\right\}$ as*

$$u_\varepsilon(t) = q\left(\frac{d(x, \Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon) \tag{7.2.7}$$

with $\Omega_\varepsilon(t) = \left\{u_\varepsilon(t,\cdot) \leq \frac{1}{2}\right\}$. The associated normal velocity of Γ_ε approximates the velocity of the sharp limit interface Γ in the following sense :

$$V_\varepsilon = c_W c_M \Delta_\Gamma H + \mathcal{O}(\varepsilon) \tag{7.2.8}$$

with $c_W = c_M = \frac{1}{6}$. Moreover, the volume is preserved only up to the first order :

$$|\Omega_\varepsilon(t)| = |\Omega_\varepsilon(0)| + \mathcal{O}(\varepsilon). \tag{7.2.9}$$

For other choices of M, the sharp limit velocity may contain an additional undesired bulk diffusion term. For instance, if $M(s) = s(1-s)$ (in particular $M'(0),\ M'(1) \neq 0$), then

$$V = c_W c_M \Delta_\Gamma H \pm c_W^2 c_M H \nabla_n H \tag{7.2.10}$$

with $c_W = \frac{1}{6}$ and $c_M = 1$.

From now on, we choose the mobility

$$M(s) = s^2(1-s)^2.$$

With such a mobility, **M-CH** has a well identified drawback : the leading error term in (7.2.7) is of order ε and has a dependence in the curvature, see [221]. In particular, it becomes significant in high curvature regions. This is especially problematic when reaching the pure states because an overshoot due to oscillations occurs, and the solution does not stay within its physical range $[0, 1]$. This problem proves to be even more problematic in the multiphase case because the solutions may not be positive anymore and phantom phases may appear.

The volume conservation is a standard property of the Cahn–Hilliard model on a smooth domain Q with periodic or Neumann homogeneous boundary conditions on ∂Q because if u is a solution to (7.2.3) then

$$\frac{d}{dt}\int_Q u\,dx = \int_Q \partial_t u\,dx = \frac{1}{\varepsilon^2}\int_Q \operatorname{div}\left(M(u)\nabla\mu\right)dx = 0.$$

However, numerically, the conservation is approximate with an approximation accuracy which depends on how close u is to the optimal profile. As we will see, a more accurate solution u provides a better approximate volume conservation.

7.3 A new variational model with two mobilities

In this section, we propose a new variational Cahn–Hilliard model with two mobilities. In contrast with [301] where the energy is modified, we incorporate the additional degeneracy in the metric used for defining the gradient flow. First, we derive our model and explain the right choice for its parameters. Then we review its theoretical properties, and we show in particular that its approximation properties are similar to those of model (7.1.2) in the particular case $p = 1$. We compare the numerical behavior of each method in the next section devoted to numerics.

7.3.1 Derivation of the model

Our model derives from the classical Cahn–Hilliard energy :

$$E(u) = \int_Q \left(\frac{\varepsilon}{2}|\nabla u|^2 + \frac{1}{\varepsilon}W(u)\right)dx.$$

Let us consider a scalar product in $H_0^1(Q)$ with two scalar positive weights M and N :

$$\langle f, g\rangle_{H_0^1} = \int_Q M\nabla(Nf)\cdot\nabla(Ng)dx.$$

The H^{-1} gradient flow of the energy E with respect to this scalar product, rescaled by a factor ε, is

$$\partial_t u = \frac{1}{\varepsilon}\nabla_{H^{-1}}E$$

which leads to the following equation :

$$\begin{cases}\varepsilon^2\partial_t u = N\operatorname{div}(M\nabla(N\mu))\\ \mu = -\varepsilon^2\Delta u + W'(u)\end{cases}$$

Considering now a dependence on u of M and N gives the following equation, that we refer to as the **NMN-CH** model :

$$\begin{cases}\varepsilon^2\partial_t u = N(u)\operatorname{div}(M(u)\nabla(N(u)\mu))\\ \mu = -\varepsilon^2\Delta u + W'(u)\end{cases}\qquad\textbf{(NMN-CH)}\tag{7.3.1}$$

This model has the advantage of deriving from the classical Cahn–Hilliard energy and being variational in the following sense :

$$\begin{aligned}\frac{d}{dt}E(u) &= \int_Q\left(-\varepsilon\Delta u + \frac{1}{\varepsilon}W'(u)\right)\partial_t u dx\\ &= \int_Q \frac{1}{\varepsilon^3}N(u)\mu\operatorname{div}\left(M(u)\nabla(N(u)\mu)\right)dx\\ &= -\int_Q \frac{1}{\varepsilon^3}M(u)|\nabla(N(u)\mu)|^2dx + \int_{\partial Q}\frac{1}{\varepsilon^3}M(u)N(u)\mu\nabla(N(u)\mu)\cdot n d\sigma\\ &= -\int_Q \frac{1}{\varepsilon^3}M(u)|\nabla(N(u)\mu)|^2dx \le 0\end{aligned}$$

in the case of periodic or Neumann homogeneous boundary conditions on ∂Q.

7.3.2 Choosing N

Recalling that we set $M(s) = s^2(1-s)^2$, we want to define N so that it has an antagonist effect to M and forces the leading error term U_1 to be zero, see details in Section 7.4.3. The correct choice for N is actually

$$N(s) = \frac{1}{\sqrt{M(s)}} = \frac{1}{s(1-s)}. \tag{7.3.2}$$

Indeed, the following equation is obtained for U_1 (see Section 7.4.3) :

$$\partial_{zz} U_1 - W''(U_0)U_1 = -H\partial_z U_0 - \mu_1$$

and with the above choice for N we have

$$-H\partial_z U_0 = \mu_1,$$

thus $U_1 = 0$ while other choices for N only impose $U_1 \to 0$ far from the interface.

Since the **NMN-CH** model involves $N(u)$, what happens when the solution u to (7.3.1) takes values in $\{0,1\}$, i.e. reaches the singularities of N? The ideal solution of the system is $q\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right)$ (see Proposition 7.3.1) which never takes values 0 or 1, but no positivity condition is guaranteed with the **NMN-CH** model, so the values 0 or 1 may actually be reached by u. For the sake of simplicity, we will assume for all calculations in the asymptotic expansions of Section 7.4.3 that $u \neq 0, 1$. Remark however that similar expansions and the same conclusions as in Proposition 7.3.1 hold for a slightly different system which allows u to reach 0 or 1. Consider indeed the modified **NMN-CH** model (that we use in practice for the numerical approximation)

$$\begin{cases} \partial_t u &= \tilde{N}(u)\,\mathrm{div}(\tilde{M}(u)\nabla(\tilde{N}(u)\mu)) \\ \mu &= \frac{1}{\varepsilon^2}W'(u) - \Delta u \end{cases}$$

where $\tilde{M}(u) = 2W(u) + \gamma\varepsilon^2$, with $\gamma > 0$, and $\tilde{N}(u) = \frac{1}{\sqrt{\tilde{M}(u)}}$. Obviously, the term $\gamma\varepsilon^2$ prevents the cancellation of $\tilde{M}(u)$. Remarkably, it does not change the conclusions of Proposition 7.3.1. In particular, the normal velocity remains the same. Indeed, since $\tilde{M}(s) = M(s) + \gamma\varepsilon^2$ and $\tilde{N} = \frac{1}{\sqrt{\tilde{M}}}$ then

$$\tilde{N}(u) = \frac{1}{\sqrt{M(u)+\gamma\varepsilon^2}} = \frac{1}{\sqrt{M(u)}}\left[1 - \frac{1}{2}\frac{\gamma}{M(u)}\varepsilon^2 + \mathcal{O}(\varepsilon^4)\right] = N(u) - \frac{1}{2}N(u)^3\gamma\varepsilon^2 + \cdots,$$

It follows that only the third and fourth orders of the expansion in Section 7.4.3) will have additional terms. At third order, the additional terms are associated with $\mu_0 = 0$ and thus vanish. At fourth order, some are associated with μ_0 and vanish, while the others are associated with μ_1 but do not alter the velocity because their integral are zero due to the matching conditions. With this conclusion in mind, we keep for simplicity the choice $M(u) = 2W(u)$ and $N(u) = \frac{1}{\sqrt{M(u)}}$, and we assume for the expansion that $u \neq 0, 1$.

7.3.3 Properties of the NMN-CH model

They are summarized in the following result, to be compared with Proposition 7.2.1. The following new quantity associated with N is used :

$$c_N = \int_{-\infty}^{+\infty} \frac{q'(z)}{N(q(z))}dz = -c_W.$$

Proposition 7.3.1. *Let $M(s) = s^2(1-s)^2$ and $N(s) = \frac{1}{\sqrt{M(s)}} = \frac{1}{s(1-s)}$. The solution u_ε to* (7.3.1) *expands formally near the interface $\Gamma_\varepsilon(t) = \left\{u_\varepsilon(t,\cdot) = \frac{1}{2}\right\}$ as :*

$$u_\varepsilon = q\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2) \tag{7.3.3}$$

with $\Omega_\varepsilon(t) = \left\{u_\varepsilon(\cdot,t) \leq \frac{1}{2}\right\}$. The associated normal velocity satisfies :

$$V_\varepsilon = \frac{c_W c_M}{c_N^2}\Delta_\Gamma H + \mathcal{O}(\varepsilon). \tag{7.3.4}$$

Moreover, when $\partial\Omega_\varepsilon(t)$ is of class $\mathcal{C}^2$ at all times, the volume is preserved up to second order :

$$|\Omega_\varepsilon(t)| = |\Omega_\varepsilon(0)| + \mathcal{O}(\varepsilon^2). \tag{7.3.5}$$

7.4 Proof of Proposition 7.3.1

We start with the volume conservation (7.3.5), assuming the other properties as in [65]. Then, we introduce the tools and notations to derive the formal asymptotics and demonstrate (7.3.3) and (7.3.4).

7.4.1 Proof of the approximate volume conservation

In this part, we demonstrate (7.3.5) assuming that the solution satisfies (7.3.3), which will be proved after. We recall the following relations satisfied by W, M, and N :

$$W(s) = \frac{1}{2}s^2(1-s)^2, \quad M(s) = 2W(s), \quad \text{and} \ \ N(s) = \frac{1}{\sqrt{2W(s)}}.$$

Consider the function G defined by $G(s) = \int_0^s \sqrt{2W(t)}dt$. Assuming that $\partial\Omega_\varepsilon(t) = \Gamma_\varepsilon(t)$ is smooth at all times, we will prove that

$$|\Omega_\varepsilon(t)| = \int_Q 6(G\circ q)\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right)dx + \mathcal{O}(\varepsilon^2) \tag{7.4.1}$$

where $d(\cdot,\Omega_\varepsilon(t))$ is the signed distance function to $\Omega_\varepsilon(t)$ with the convention that $d(\cdot,\Omega_\varepsilon(t)) < 0$ in $\Omega_\varepsilon(t)$. Assuming that u_ε satisfies (7.3.3), we have :

$$\int_Q G(u_\varepsilon(x,t)) = \int_Q (G\circ q)\left(\frac{d(x,\Omega_\varepsilon(t)}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2)$$

so if (7.4.1) holds, it follows that :

$$\forall t \geq 0, \quad |\Omega_\varepsilon(t)| = 6\int_Q G\left(u_\varepsilon(x,t)\right)dx + \mathcal{O}(\varepsilon^2)$$

Considering periodic or Neumann boundary condition on Q leads to a conservation of the integral of G along time :

$$\begin{aligned}\frac{d}{dt}\int_Q G(u_\varepsilon)dx &= \int_Q G'(u_\varepsilon)\partial_t u_\varepsilon\,dx = \frac{1}{\varepsilon^2}\int_Q \sqrt{2W(u_\varepsilon)}N(u_\varepsilon)\,\mathrm{div}\left(M(u_\varepsilon)\nabla(N(u_\varepsilon)\mu)\right)dx \\ &= \frac{1}{\varepsilon^2}\int_Q \mathrm{div}\left(M(u_\varepsilon)\nabla(N(u_\varepsilon)\mu)\right)dx = \frac{1}{\varepsilon^2}\int_{\partial Q} M(u_\varepsilon)\nabla\left(N(u_\varepsilon)\mu\right)\cdot n\,dx = 0\end{aligned}$$

where n denotes the outer unit normal on ∂Q. If follows that the volume is (approximately) conserved over time and (7.3.5) is valid if (7.4.1) is satisfied.

We now turn to the proof of (7.4.1). By the coarea formula (see [11]) and because $|\nabla d(\cdot,\Omega_\varepsilon(t))| = 1$, we have

$$\int_Q 6(G\circ q)\left(\frac{d(x,\Omega_\varepsilon(t)}{\varepsilon}\right)dx = 6\int_{\mathbb{R}} h(s)G\left(q\left(\frac{s}{\varepsilon}\right)\right)ds$$

where $h(s) = \mathcal{H}^{d-1}\left(\{x\in Q, d(x,\Omega_\varepsilon(t)) = \varepsilon s\}\right)$ is well defined for a.e. s since $\partial\Omega_\varepsilon(t)$ is assumed to be smooth at all times [14, 243, 11]. Using the fact that $6G(q(-s)) = 6G(1-q(s)) = 1 - 6G(q(s))$, we calculate that :

$$\begin{aligned}\int_Q 6(G\circ q)\left(\frac{d(x,\Omega_\varepsilon(t)}{\varepsilon}\right)dx &= \int_{-\infty}^0 h(s)ds + \int_{-\infty}^0 h(s)\left(6G\left(q\left(\frac{s}{\varepsilon}\right)\right)-1\right)ds + 6\int_0^{+\infty} h(s)G\left(q\left(\frac{s}{\varepsilon}\right)\right)ds \\ &= |\Omega_\varepsilon(t)| - 6\int_{-\infty}^0 h(s)G\left(q\left(\frac{-s}{\varepsilon}\right)\right)ds + 6\int_0^{+\infty} h(s)G\left(q\left(\frac{s}{\varepsilon}\right)\right)ds \\ &= |\Omega_\varepsilon(t)| + 6\varepsilon\int_0^{+\infty}\left[h(\varepsilon s) - h(-\varepsilon s)\right]G(q(s))ds\end{aligned}$$

Equation (7.4.1) holds if we can prove that the second term of the right hand side is a $\mathcal{O}(\varepsilon^2)$. Using the regularity of $\Omega_\varepsilon(t)$, we have [243, 11] :

$$\forall s\in\,]0,|\log(\varepsilon)|[\,, \quad h(\varepsilon s) - h(-\varepsilon s) = 2\varepsilon s h'(0) + \mathcal{O}(s^2\varepsilon^2)$$

As $q(s) = \frac{1-\tanh(s/2)}{2}$ and G is a polynomial function which vanishes at 0, the moments $\int_0^{+\infty} s^n G(q(s))ds$ are finite. Then,

$$\left|\int_0^{|\log(\varepsilon)|}\left(h(\varepsilon s) - h(-\varepsilon s)\right)G(q(s))ds\right| \leq \left|\int_0^{|\log(\varepsilon)|}\left(2s\varepsilon h'(0) + Cs^2\varepsilon^2\right)G(q(s))ds\right| = \mathcal{O}(\varepsilon)$$

On the other hand, we know [14, 11] that $h(s) \sim_{s\to+\infty} s^{d-1}$, thus

$$\int_{|\log(\varepsilon)|}^{+\infty} h(\varepsilon s)G(q(s))ds \leq C\varepsilon^{d-1}\int_{|\log(\varepsilon)|}^{+\infty} s^{d-1}G(q(s))ds = \mathcal{O}(\varepsilon^{d-1}),$$

and that h is bounded in $\mathbb{R}^*_-$, therefore :

$$\int_{|\log(\varepsilon)|}^{+\infty} h(-\varepsilon s)G(q(s))ds \leq C\int_{|\log(\varepsilon)|}^{+\infty} G(q(s))ds = \mathcal{O}(\varepsilon)$$

We conclude that

$$6\varepsilon\int_0^{+\infty} \left[h(\varepsilon s) - h(-\varepsilon s)\right] G(q(s))ds = \mathcal{O}(\varepsilon^2)$$

so that (7.4.1) is true and the property (7.3.5) is established under the condition that (7.3.3) is satisfied. Proving it is the purpose of the next subsection.

7.4.2 Formal asymptotics toolbox

Before the actual computations, we first recall the tools necessary to perform our formal asymptotic derivation, following the notations of [5, 96, 72] and the results in differential geometry of [11]. The principle is to study separately the behavior of the solution near the interface and far from it. We will do the derivations in dimension 2 for the sake of simplicity of the notations and readability, but the principle is identical in higher dimension.

To derive the method we require that the interface $\Gamma(t) = \partial\Omega_\varepsilon(t)$ remains smooth enough and that there exists $\delta > 0$ such that

$$d : (x,t) \mapsto d(x,t) := \begin{cases} -\operatorname{dist}(x,\Gamma(t)) & \text{for } x \in \Omega_\varepsilon(t) \\ \operatorname{dist}(x,\Gamma(t)) & \text{for } x \in Q \setminus \overline{\Omega_\varepsilon(t)} \end{cases}$$

is well-defined at all times in $\mathcal{N}_{\delta,t} = \Gamma(t) \oplus B_{3\delta}(0) = \{x,\ |d(x,t)| < 3\delta\}$. $\mathcal{N}_{\delta,t}$ is called the *inner region* and its complementary the *outer region*.

Outer variables. Far from the interface, we consider the *outer functions* (u,μ) depending on the standard *outer variable* x. The system remains the same :

$$\begin{cases} \varepsilon^2\partial_t u = N(u)\operatorname{div}(M(u)\nabla(N(u)\mu)) \\ \mu = -\varepsilon^2\Delta u + W'(u) \end{cases} \tag{7.4.2}$$

Inner variables. Inside $\mathcal{N}_{\delta,t}$, we consider the *inner variables* (z,s) associated with the original variables (x,t) in the following way : $z = \frac{d(x,t)}{\varepsilon}$ is a variable along the normal direction to the interface $\Gamma(t)$ and $s = S(x,t)$ is associated with a parameterization $X_0(s,t)$ of $\Gamma(t)$. We define the *inner functions* $U, \mathfrak{\mu}$ depending on (z,s) as follows :

$$\begin{cases} U(z,s,t) := U\left(\dfrac{d(x,t)}{\varepsilon}, S(x,t), t\right) = u(x,t) \\ \mathfrak{\mu}(z,s,t) := \mathfrak{\mu}\left(\dfrac{d(x,t)}{\varepsilon}, S(x,t), t\right) = \mu(x,t) \end{cases}$$

In order to express the derivatives of U, we first need to calculate the gradient and the Laplacian of d and S. The properties of d are common knowledge in differential geometry, see for instance [11] :

$$\begin{cases} \nabla d(x,t) = n(x,t) \\ \Delta d(x,t) = \displaystyle\sum_{k=1}^{d-1} \frac{\kappa_k(\pi(x))}{1+\kappa_k(\pi(x))d(x,t)} \\ \qquad\quad = \dfrac{H}{1+\varepsilon z H} \text{ in dimension 2} \end{cases}$$

where π is the projection onto $\Gamma(t)$ and the κ_k's are the principal curvatures.

Let $X_0(s,t)$ be a point on $\Gamma(t)$ and $X(z,s,t)$ a point inside $\mathcal{N}_{\delta,t}$ at signed distance εz from the interface oriented by n and whose orthogonal projection onto the interface is X_0, i.e.,

$$X(z,s,t) = X_0(s,t) + \varepsilon z n(s,t)$$

Deriving with respect to z the equation $s = S\left(X_0(s,t) + \varepsilon z n(s,t), t\right)$ gives

$$0 = \varepsilon n \cdot \nabla S = \varepsilon \nabla d \cdot \nabla S,$$

which implies there are no cross derivative terms. Deriving the same equation with respect to s yields :

$$1 = (\partial_s X_0 + \varepsilon z \partial_s n) \cdot \nabla S = (1 + \varepsilon z H)\tau \cdot \nabla S$$

Since ∇S is orthogonal to n, therefore collinear with the tangent τ, we have that :

$$\nabla S = \frac{1}{1+\varepsilon z H}\tau$$

Taking the divergence, we find ΔS :

$$\begin{aligned}\Delta S = \operatorname{div}\left(\frac{1}{1+\varepsilon z H}\tau\right) &= \nabla\left(\frac{1}{1+\varepsilon z H}\right)\cdot\tau + \frac{1}{1+\varepsilon z H}\operatorname{div}(\tau) \\ &= \frac{1}{1+\varepsilon z H}\partial_s\left(\frac{1}{1+\varepsilon z H}\right) + \frac{1}{1+\varepsilon z H}\tau\cdot\partial_s\tau = -\frac{\varepsilon z \partial_s H}{(1+\varepsilon z H)^3}\end{aligned}$$

To express the connection between the derivatives of U, μ and u, μ, we come back to the definition of the inner functions :

$$u(x,t) = U\left(\frac{d(x,t)}{\varepsilon}, S(x,t), t\right) \tag{7.4.3}$$

Successive derivations with respect to x give the following equations

$$\begin{cases} \nabla u = \nabla d \dfrac{1}{\varepsilon}\partial_z U + \nabla S \partial_s U \\ \Delta u = \Delta d \dfrac{1}{\varepsilon}\partial_z U + \dfrac{1}{\varepsilon^2}\partial_{zz} U + \Delta S \partial_s U + |\nabla S|^2 \partial_{ss} U \\ \operatorname{div}\left(M(u)\nabla(N(u)\mu)\right) = \dfrac{1}{\varepsilon^2}\partial_z\left(M\partial_z(N\mu)\right) + \dfrac{M}{\varepsilon}\Delta d \partial_z(N\mu) \\ \qquad\qquad + |\nabla S|^2 \partial_s\left(M\partial_s(N\mu)\right) + \Delta S M \partial_s(N\mu) \end{cases} \tag{7.4.4}$$

The *inner system* of the **NMN-CH** model finally reads as :

$$\begin{cases} \varepsilon^2\partial_t U + \varepsilon^2 \partial_t S \partial_s U - \varepsilon V \partial_z U = \dfrac{N}{\varepsilon^2}\partial_z\left(M\partial_z(N\mu)\right) + \dfrac{NM}{\varepsilon}\Delta d \partial_z(N\mu) + T_1(s) \\ \mu = W'(U) - \partial_{zz} U - \varepsilon \Delta d \partial_z U - \varepsilon^2 T_2(s) \\ T_1(s) = -\dfrac{\varepsilon z \partial_s H N M}{(1+\varepsilon z H)^3}\partial_s(N\mu) + \dfrac{N}{(1+\varepsilon z H)^2}\partial_s(M\partial_s(N\mu)) \\ T_2(s) = -\dfrac{\varepsilon z \partial_s H}{(1+\varepsilon z H)^3}\partial_s U + \dfrac{1}{(1+\varepsilon z H)^2}\partial_{ss} U \\ \Delta d = \dfrac{H}{1+\varepsilon z H} \end{cases} \tag{7.4.5}$$

Independence in z of the normal velocity V The normal velocity of the interface $V(s,t)$ is defined by :

$$V(s,t) = \partial_t X_0(s,t) \cdot n(s,t)$$

In the neighbourhood $\mathcal{N}_{\delta,t}$, the following property is a direct consequence of the definition of the signed distance function :

$$d(X_0(s,t) + \varepsilon z n(s,t), t) = \varepsilon z$$

Deriving this with respect to t yields :

$$V(s,t) = \partial_t X_0(s,t) \cdot \nabla d(X_0(s,t) + \varepsilon z n(s,t), t) = -\partial_t d\left(X(z,s,t), t\right)$$

Thus, $\partial_t d(x,t)$ is independent of z and we can extend the function everywhere in the neighbourhood by choosing :

$$V(X_0(s,t) + \varepsilon z n, t) := -\partial_t d(X_0(s,t) + \varepsilon z n, t) = V(s,t)$$

This independence property is crucial to be able to extract the velocity from integrals in z in the following derivations.

Taylor expansions We assume the following Taylor expansions for our functions :

$$\begin{aligned}
u(x,t) &= u_0(x,t) + \varepsilon u_1(x,t) + \varepsilon^2 u_2(x,t) + \cdots \\
U(z,s,t) &= U_0(z,s,t) + \varepsilon U_1(z,s,t) + \varepsilon^2 U_2(z,s,t) + \cdots \\
\mu(x,t) &= \mu_0(x,t) + \varepsilon \mu_1(x,t) + \varepsilon^2 \mu_2(x,t) + \cdots \\
\boldsymbol{\mu}(z,s,t) &= \boldsymbol{\mu}_0(z,s,t) + \varepsilon \boldsymbol{\mu}_1(z,s,t) + \varepsilon^2 \boldsymbol{\mu}_2(z,s,t) + \cdots
\end{aligned}$$

We compose these expansions with a regular function F :

$$\begin{aligned}
F(U) = F(U_0) &+ \\
&+ \varepsilon F'(U_0)U_1 + \\
&+ \varepsilon^2 \left[F'(U_0)U_2 + \frac{F''(U_0)}{2} U_1^2 \right] + \\
&+ \varepsilon^3 \left[F'(U_0)U_3 + F''(U_0)U_1U_2 + \frac{F'''(U_0)}{6} U_1^3 \right] + \cdots
\end{aligned}$$

We can now investigate order by order the behavior of the system. We have to study four orders as the velocity appears in the fourth order of the first equation of the Cahn–Hilliard system.

To simplify the notation within the asymptotics, we adopt the following notations for $M(u)$:

$$M(u) = m_0 + \varepsilon m_1 + \varepsilon^2 m_2 + \cdots ,$$

where each term corresponds to :

$$\begin{cases}
m_0 = M(u_0) \\
m_1 = M'(u_0)u_1 \\
m_2 = M'(u_0)u_2 + \dfrac{M''(u_0)}{2}(u_1)^2
\end{cases}$$

We adopt the same convention for any generic *outer function* $F(u)$ or *inner function* $F(U)$:

$$\begin{aligned}
F(u) &= f_0 + \varepsilon f_1 + \varepsilon^2 f_2 + \varepsilon^3 f_3 + \cdots \\
F(U) &= F_0 + \varepsilon F_1 + \varepsilon^2 F_2 + \varepsilon^3 F_3 + \cdots
\end{aligned}$$

We can now investigate order by order the behavior of system (7.3.1). We have to study up to the fourth order where the leading order of the velocity will appear in the first equation of (8.2.7).

Flux matching condition between inner and outer equations. Instead of using the matching conditions directly between the first equations of the inner and outer systems, it is more convenient to perform the matching on the flux $j = M(u)\nabla(N(u)\mu)$. It has the following Taylor expansion :

$$\begin{aligned}
j = &\left[m_0 \nabla (n\mu)_0 \right] \\
&+ \varepsilon \left[m_1 \nabla (n\mu)_0 + m_0 \nabla (n\mu)_1 \right] \\
&+ \varepsilon^2 \left[m_2 \nabla (n\mu)_0 + m_1 \nabla (n\mu)_1 + m_0 \nabla (n\mu)_2 \right] \\
&+ \mathcal{O}(\varepsilon^3)
\end{aligned} \tag{7.4.6}$$

In inner coordinates, we only need to express the normal part $J_n := J \cdot n = \frac{M(U)}{\varepsilon} \partial_z \left(N(U) \boldsymbol{\mu} \right)$ because the tangential part terms are of higher order. It expands as :

$$\begin{aligned}
J_n = &\frac{1}{\varepsilon} \left[M_0 \partial_z (N\boldsymbol{\mu})_0 \right] \\
&+ \left[M_1 \partial_z (N\boldsymbol{\mu})_0 + M_0 \partial_z (N\boldsymbol{\mu})_1 \right] \\
&+ \varepsilon \left[M_2 \partial_z (N\boldsymbol{\mu})_0 + M_1 \partial_z (N\boldsymbol{\mu})_1 + M_0 \partial_z (N\boldsymbol{\mu})_2 \right] \\
&+ \varepsilon^2 \left[M_3 \partial_z (N\boldsymbol{\mu})_0 + M_2 \partial_z (N\boldsymbol{\mu})_1 \right. \\
&\qquad \left. + M_1 \partial_z (N\boldsymbol{\mu})_2 + M_0 \partial_z (N\boldsymbol{\mu})_3 \right] \\
&+ \mathcal{O}(\varepsilon^3)
\end{aligned} \tag{7.4.7}$$

The flux matching conditions allow to match the limit as $z \to \pm\infty$ of the terms of (7.4.7) with the corresponding order terms of (7.4.6).

7.4.3 Formal matched asymptotic analysis for the new NMN-CH model

Now that all necessary tools are defined, we prove Proposition 7.3.1. At first order, we determine the profile of the solution. At second order, we link the curvature with the leading term of μ and prove that the leading error term is zero. The third order is used to establish certain relations between different terms, and lastly we recover the velocity in the fourth order.

First order At order $(\mathcal{O}(1), \mathcal{O}(1))$ the outer system (8.2.7) reads as :

$$\begin{cases} 0 = n_0 \operatorname{div}(m_0 \nabla(n_0 \mu_0)) \\ \mu_0 = W'(u_0) \end{cases} \tag{7.4.8}$$

At order $(\mathcal{O}(\varepsilon^{-2}), \mathcal{O}(1))$ the inner system (7.4.5) reads as :

$$\begin{cases} 0 = N_0 \partial_z \left(M_0 \partial_z (N_0 \boldsymbol{\mu}_0) \right) \\ \boldsymbol{\mu}_0 = W'(U_0) - \partial_{zz} U_0 \end{cases}$$

The first equation gives that $M_0 \partial_z (N_0 \boldsymbol{\mu}_0) = B_0$ is constant in z. The matching conditions on the outer flux (7.4.6) and the inner flux (7.4.7) at order ε^{-1} impose this constant to be zero. Then $N_0 \boldsymbol{\mu}_0$ is constant. The matching conditions with the outer system (7.4.8) give that :

$$\boldsymbol{\mu}_0 = 0$$

Then U_0 satisfies the following differential equation, complemented with a translation condition to ensure uniqueness :

$$\begin{cases} \partial_{zz} U_0 - W'(U_0) = 0 \\ \forall (s,t), U_0(0,s,t) = \dfrac{1}{2} \end{cases}$$

The solution to this equation is the profile q given by (7.2.1). Thus the terms at first order are :

$$\begin{cases} \boldsymbol{\mu}_0 = \mu_0 = 0 \\ U_0 = q(z) := \dfrac{1 - \tanh(\frac{z}{2})}{2} \\ u_0 = 0 \text{ or } 1 \end{cases}$$

Second order At order $(\mathcal{O}(\varepsilon), \mathcal{O}(\varepsilon))$, using the values of u_0 and μ_0 obtained above, the outer system (8.2.7) reads as :

$$\begin{cases} 0 = n_0 \operatorname{div}(m_0 \nabla(n_0 \mu_1)) \\ \mu_1 = W''(u_0) u_1 = u_1 \end{cases} \tag{7.4.9}$$

At order $(\mathcal{O}(\varepsilon^{-1}), \mathcal{O}(\varepsilon))$ the inner system (7.4.5) reads as :

$$\begin{cases} 0 = N_0 \partial_z \left(M_0 \partial_z (N_0 \boldsymbol{\mu}_1) \right) \\ \boldsymbol{\mu}_1 = W''(U_0) U_1 - \partial_{zz} U_1 - H \partial_z U_0 \end{cases} \tag{7.4.10}$$

The first equation of (7.4.10) shows that $M_0 \partial_z (N_0 \boldsymbol{\mu}_1)$ is a certain constant B_1. The matching conditions between the inner flux (7.4.7) and the outer flux (7.4.6) at order 0 require that (by removing all null terms) :

$$B_1 = \lim_{z \to +\infty} M_0 \partial_z (N_0 \boldsymbol{\mu}_1) = 0.$$

Then there exists a function A_1 constant in z such that $N_0 \boldsymbol{\mu}_1 = A_1$. The matching from inner to outer for μ yields :

$$\mu_1 = \lim_{z \to \pm\infty} \boldsymbol{\mu}_1 = \lim_{z \to \pm\infty} \frac{A_1}{N_0} = 0.$$

From the matching conditions with the second equation of (7.4.9) we have :

$$u_1 = \mu_1 = 0.$$

We now determine the value of A_1 using the second equation of (7.4.10). We multiply it by $\partial_z U_0$ and integrate it. We divide the equation in three terms. The left hand side term gives :

$$\int_{-\infty}^{+\infty} \boldsymbol{\mu}_1 \partial_z U_0 dz = \int_{-\infty}^{+\infty} N_0 \boldsymbol{\mu}_1 \frac{\partial_z U_0}{N_0} dz = A_1 \int_{-\infty}^{+\infty} \frac{\partial_z U_0(z)}{N(U_0(z))} dz = A_1 c_N$$

The first two terms in the right hand side vanish :

$$\begin{aligned}\int_{-\infty}^{+\infty}(W''(U_0)U_1\partial_z U_0 - \partial_{zz}U_1\partial_z U_0)dz &= \int_{-\infty}^{+\infty}(\partial_z\left(W'(U_0)\right)U_1 - \partial_{zz}U_1\partial_z U_0)dz \\ &= -\int_{-\infty}^{+\infty}\underbrace{\left(W'(U_0)-\partial_{zz}U_0\right)}_{=0}\partial_z U_1 dz \\ &\qquad + [W'(U_0)U_1 - \partial_z U_0\partial_z U_1]_{-\infty}^{+\infty} \\ &= 0\end{aligned}$$

The fact that the functions in the bracket term vanish at the limit $z \to \pm\infty$ comes from the matching conditions.
The last right hand side term results in the curvature :

$$\int_{-\infty}^{+\infty} - H(\partial_z U_0)^2 dz = -H\int_{-\infty}^{+\infty} q'(z)^2 dz = -c_W H$$

Then :

$$N_0\mu_1 = A_1 = -\frac{c_W}{c_N}H \tag{7.4.11}$$

In conclusion, we have the following properties :

$$\begin{cases}\mu_1 = -\dfrac{c_W}{c_N}\dfrac{H}{N(q)} \\ \partial_{zz}U_1 - W''(U_0)U_1 = -Hq'(z) - \mu_1 \\ \mu_1 = u_1 = 0\end{cases}$$

Reminding that $N(q(z)) = \frac{1}{\sqrt{2W(q(z))}} = -\frac{1}{q'(z)}$ so that $c_N = -c_W$, the equation satisfied by U_1 is :

$$\partial_{zz}U_1 - W''(U_0)U_1 = 0$$

To solve this equation, we use the following rather standard lemma[6, 5] :

Lemma 7.4.1. *Let A(z) be a bounded function on $-\infty < z < \infty$. Then the problem :*

$$\begin{cases}\partial_{zz}\psi - W''(q(z))\psi = A(z) \\ \psi(0) = 0, \quad \psi \in \mathrm{L}^\infty(\mathbb{R})\end{cases}$$

has a solution if and only if :

$$\int_{-\infty}^{+\infty} A(z)q'(z)dz = 0 \tag{7.4.12}$$

Moreover the solution, if it exists, is unique, satisfies

$$\forall z \in \mathbb{R},\ |\psi(z)| \le C\left\|A\right\|_{\mathrm{L}^\infty} \tag{7.4.13}$$

and is given by the formula :

$$\psi(z) = q'(z)\int_0^z\left(\frac{1}{(q'(s))^2}\int_{-\infty}^s A(\sigma)q'(\sigma)d\sigma\right)ds \tag{7.4.14}$$

Sketch of the proof : Multiplying the equation by q' and integrating by parts, we see that condition (7.4.12) is necessary. Conversely, if (7.4.12) holds, the method of variation of constants leads to the explicit form (7.4.14) of the solution, see [6, 5] for more details. □

Using Lemma 7.4.1 with $A = 0$ and assuming that $U_1(0, s, t) = 0$, we have that $U_1 = 0$. Therefore the leading error term in U is of magnitude ε^2 and (7.3.3) of Proposition 7.3.1 follows.

Third order At order $(\mathcal{O}(\varepsilon^2), \mathcal{O}(\varepsilon^2))$ the outer system (8.2.7) reads as :

$$\begin{cases} 0 = n_0 \operatorname{div}\left(m_0 \nabla(n_0 \mu_2)\right) \\ \mu_2 = W'''(u_0)\dfrac{(u_1)^2}{2} + W''(u_0)u_2 - \Delta u_0 = u_2 \end{cases} \tag{7.4.15}$$

where we have simplified the terms in u_0 because of the matching conditions with the profile of $U_0 = q(z)$.

At order $(\mathcal{O}(1), \mathcal{O}(\varepsilon^2))$ the inner system (7.4.5) reads as :

$$\begin{cases} 0 = N_0 \partial_z \left(M_0 \partial_z (N_0 \boldsymbol{\mu}_2 + N_1 \boldsymbol{\mu}_1)\right) \\ \boldsymbol{\mu}_2 = W'''(U_0)\dfrac{(U_1)^2}{2} + W''(U_0)U_2 - \partial_{zz}U_2 - H\partial_z U_1 + zH^2\partial_z U_0 \end{cases} \tag{7.4.16}$$

The terms depending on the $z-$derivative of $\boldsymbol{\mu}_0$ and $N_0\boldsymbol{\mu}_1$ have been omitted as they are null because of the results of the second and first orders.

Similarly to previous orders, there exists a constant B_2 in z so that :

$$M_0 \partial_z (N_0 \boldsymbol{\mu}_2 + N_1 \boldsymbol{\mu}_1) = B_2$$

The matching of the flux terms from (7.4.6) and (7.4.7) of order ε (removing all null terms) yields :

$$B_2 = \lim_{z \to \pm\infty} M_0 \partial_z (N_0 \boldsymbol{\mu}_2 + N_1 \boldsymbol{\mu}_1) = 0$$

Thus, there exists a constant A_2 such that :

$$N_0 \boldsymbol{\mu}_2 + N_1 \boldsymbol{\mu}_1 = A_2 \tag{7.4.17}$$

The derivative of this term with respect to z appears at the next order. It is zero so we can omit it in the next paragraph.

Fourth order At order $(\mathcal{O}(\varepsilon^3), \mathcal{O}(\varepsilon^3))$ the outer system (8.2.7) reads as :

$$\begin{cases} 0 = n_0 \operatorname{div}\left(m_0 \nabla(n_1 \mu_2 + n_0 \mu_3)\right) \\ \mu_3 = W''(u_0)u_3 + W'''(u_0)u_1 u_2 + \dfrac{W''''(u_0)}{6}(u_1)^3 - \Delta u_1 \end{cases} \tag{7.4.18}$$

At order $(\mathcal{O}(\varepsilon), \mathcal{O}(\varepsilon^3))$ the inner system (7.4.5) reads as :

$$\begin{cases} -V_0 \partial_z U_0 = N_0 \partial_z \left(M_0 \partial_z (N_2 \boldsymbol{\mu}_1 + N_1 \boldsymbol{\mu}_2 + N_0 \boldsymbol{\mu}_3)\right) + N_0 \partial_s \left(M_0 \partial_s (N_0 \boldsymbol{\mu}_1)\right) \\ \boldsymbol{\mu}_3 = W''(U_0)U_3 + W'''(U_0)U_1 U_2 + \dfrac{W''''(U_0)}{6}(U_1)^3 - \partial_{zz}U_3 \\ \qquad\qquad - H^3 z^2 \partial_z U_0 + zH^2 \partial_z U_1 - H\partial_z U_2 - \partial_{ss}U_1 \end{cases} \tag{7.4.19}$$

We determine the velocity V_0 by multiplying by $\frac{1}{N_0}$ and integrating the first equation of (7.4.19). We divide the equality in three terms. The left hand side term isolates the velocity :

$$-V_0 \int_{-\infty}^{+\infty} \frac{\partial_z U_0}{N(U_0)} dz = -c_N V_0$$

The first term of the right hand side is a pure derivative :

$$\int_{-\infty}^{+\infty} \partial_z \left(M_0 \partial_z (N_2 \boldsymbol{\mu}_1 + N_1 \boldsymbol{\mu}_2 + N_0 \boldsymbol{\mu}_3)\right) dz$$

Using the matching conditions between the fluxes (7.4.7) and (7.4.6) at order ε^2 (the equations (7.4.17) and (7.4.11) ensuring that the other inner terms are zero) we obtain :

$$\left[M_0 \partial_z (N_2 \boldsymbol{\mu}_1 + N_1 \boldsymbol{\mu}_2 + N_0 \boldsymbol{\mu}_3)\right]_{-\infty}^{+\infty} = 0$$

Finally, using (7.4.11), the second term of the right hand side gives the surface diffusion part :

$$\int_{-\infty}^{+\infty} M_0 \partial_{ss} (N_0 \boldsymbol{\mu}_1) dz = -\frac{c_M c_W}{c_N} \partial_{ss} H$$

In conclusion, we obtain the expected motion (7.3.4) in 2D (again, the computations are similar in higher dimension) :

$$V_0 = \frac{c_M c_W}{(c_N)^2} \partial_{ss} H$$

This concludes the proof of Proposition 7.3.1. □

7.5 Numerics : discretization and experiments

In this section, we introduce a generic numerical scheme to approximate the solutions to the three different Cahn–Hilliard models :

— The classical Cahn–Hilliard equation (**C-CH**)

$$\begin{cases} \partial_t u &= \Delta \mu \\ \mu &= \frac{1}{\varepsilon^2} W'(u) - \Delta u, \end{cases}$$

where $W(s) = \frac{1}{2} s^2 (1-s)^2$.

— The Cahn–Hilliard model with classical mobility (**M-CH**)

$$\begin{cases} \partial_t u &= \operatorname{div}(M(u) \nabla \mu) \\ \mu &= \frac{1}{\varepsilon^2} W'(u) - \Delta u. \end{cases}$$

We use the mobility $M(u) = \frac{1}{c_N^2} 2W(u)$, where the constant $c_N = -\frac{1}{6}$ is introduced to get the same limit law as with our new Cahn–Hilliard model.

— The new second order variational Cahn–Hilliard model (**NMN-CH**)

$$\begin{cases} \partial_t u &= N(u) \operatorname{div}(M(u) \nabla (N(u) \mu)) \\ \mu &= \frac{1}{\varepsilon^2} W'(u) - \Delta u \end{cases}$$

with the modified mobilities $M(u) = 2W(u) + \gamma \varepsilon^2$ (with $\gamma > 0$) and $N(u) = \frac{1}{\sqrt{M(u)}}$ (keeping for simplicity the original notations M, N). Recall from the argument of Section 7.3.2 that these modified mobilities prevent the cancellation of $M(u)$ and do not change the conclusions of Proposition 7.3.1. We cannot guarantee in the numerical simulations that u will not take values 0 or 1 (in which case $W(u) = 0$), so these modified mobilities have to be used. We set $\gamma = 1$ for all numerical experiments presented below.

Our numerical algorithm is constructed as a semi-implicit Fourier spectral method in the spirit of [92, 65, 71, 75, 67], see [126] for a recent review of numerical methods for the phase field approximation of various geometric flows.

The numerical scheme we consider is based on a convex splitting of the Cahn–Hilliard energy, which was first proposed by Eyre [158] and became popular as a simple, efficient, and stable scheme to approximate various evolution problems with a gradient flow structure [98, 334, 171, 142, 309, 310]. More recently, a first- and second-order splitting scheme was proposed in [21, 301, 300] to address the case of the Cahn–Hilliard equation with mobility. However, these approaches are based on the finite element method and are not compatible with a Fourier spectral discretization.

Here, we propose to extend the principle of convex splitting using an additional convex splitting of the variational metric associated to the mobility. The resulting generic scheme is very simple (we will provide below a **Matlab** implementation with less than 40 lines for the NMN-CH model) and efficient, even in the case of highly contrasted and degenerate mobilities.

We will now give details about the variants of this scheme for the three models **C-CH**, **N-CH**, and **NMN-CH**, and we will show numerical comparisons in space dimensions 2 and 3.

7.5.1 Spatial discretization : a Fourier spectral approach

All equations are solved on a square-box $Q = [0, L_1] \times \cdots \times [0, L_d]$ with periodic boundary conditions. We recall that the Fourier $\boldsymbol{K}$-approximation of a function u defined in a box $Q = [0, L_1] \times \cdots \times [0, L_d]$ is given by

$$u^{\boldsymbol{K}}(x) = \sum_{\boldsymbol{k} \in K_N} c_{\boldsymbol{k}} e^{2i\pi \boldsymbol{\xi}_k \cdot x},$$

where $K_N = [-\frac{N_1}{2}, \frac{N_1}{2} - 1] \times [-\frac{N_2}{2}, \frac{N_2}{2} - 1] \cdots \times [-\frac{N_d}{2}, \frac{N_d}{2} - 1]$ with $N_i, i \in \{1, \cdots, d\}$, the number of samples for the i-th dimension, $\boldsymbol{k} = (k_1, \ldots, k_d) \in K_N$, and $\boldsymbol{\xi}_k = (k_1/L_1, \ldots, k_d/L_d)$. In all our numerical experiments, we choose $N_1, \cdots, N_d$ to have the same spatial resolution δ_x along all axis, i.e., $\forall i \ \frac{L_i}{N_i} = \delta_x$. In the decomposition formula above, the $c_{\boldsymbol{k}}$'s denote the discrete Fourier coefficients of u. The inverse discrete Fourier transform leads to $u^K_{\boldsymbol{k}} = \text{IFFT}[c_{\boldsymbol{k}}]$ where $u^K_{\boldsymbol{k}}$ denotes the value of u at the points $x_{\boldsymbol{k}} = (k_1 h_1, \cdots, k_d h_d)$ with $h_i = L_i/N_i$ for $i \in \{1, \cdots, d\}$ ($h_i = \delta_x$ in all our numerical experiments). Conversely, $c_{\boldsymbol{k}}$ can be computed as the discrete Fourier transform of $u^K_{\boldsymbol{k}}$, *i.e.*, $c_{\boldsymbol{k}} = \text{FFT}[u^K_{\boldsymbol{k}}]$.

7.5.2 Time discretization

Given a time discretization parameter $\delta_t > 0$, we construct a sequence $(u^n)_{n \geq 0}$ of approximations of u at times $n\delta_t$.

An IMEX scheme for the C-CH model

We will use a simple scheme to approximate the solutions to the classical Cahn–Hilliard equation

$$\begin{cases} \partial_t u &= \Delta \mu \\ \mu &= \frac{1}{\varepsilon} \nabla_u E(u) = \frac{1}{\varepsilon^2} W'(u) - \Delta u, \end{cases}$$

where E denotes the Cahn–Hilliard energy

$$E(u) = \int_Q (\varepsilon \frac{|\nabla u|^2}{2} + \frac{1}{\varepsilon} W(u)) dx.$$

A semi-implicit scheme based on a convex-concave splitting of the energy Following [158], we decompose the energy E as the sum of convex and concave energies

$$E(u) = E_c(u) + E_e(u),$$

and we consider a scheme where the convex term is handled implicitly and the concave term explicitly :

$$\begin{cases} (u^{n+1} - u^n)/\delta_t &= \Delta \mu^{n+1} \\ \mu^{n+1} &= \frac{1}{\varepsilon} \nabla_u E_c(u^{n+1}) + \frac{1}{\varepsilon} \nabla_u E_e(u^n) \end{cases}$$

Considering the convex auxiliary energy

$$u \mapsto \overline{E}_{u^n}(u) = E_c(u) + E_e(u^n) + \langle \nabla_u E_e(u^n), u - u^n \rangle,$$

we observe that $\mu^{n+1} = \frac{1}{\varepsilon} \nabla_u \overline{E}_{u^n}(u^{n+1})$ and u^{n+1} is the minimizer of

$$u \mapsto F_{u^n}(u) = \frac{\varepsilon}{2\delta_t} \int_Q |\nabla \Delta^{-1}(u - u^n)|^2 dx + \overline{E}_{u^n}(u)$$

whose gradient with respect to u is

$$-\frac{\varepsilon}{\delta_t} \Delta^{-1}(u - u^n) + \nabla_u \overline{E}_{u^n}(u).$$

By minimality,

$$\overline{E}_{u^n}(u^{n+1}) \leq F_{u^n}(u^{n+1}) \leq F_{u^n}(u^n) = \overline{E}_{u^n}(u^n) = E(u^n).$$

The concavity assumption for E_e implies that $E(u) \leq \overline{E}_{u^n}(u)$ and we conclude that the proposed splitting scheme guarantees without any assumption on the time step δ_t the decreasing of the energy E, i.e.,

$$\forall n \geq 0, \qquad E(u^{n+1}) \leq E(u^n).$$

A particular splitting choice for the Cahn–Hilliard energy For the smooth potential $W(s) = \frac{1}{2} s^2 (1-s)^2$, a standard convex/concave splitting choice for the Cahn–Hilliard energy is

$$E_c(u) = \frac{1}{2} \int_Q (\varepsilon |\nabla u|^2 + \frac{\alpha}{\varepsilon} u^2) dx \quad \text{and} \quad E_e(u) = \int_Q \frac{1}{\varepsilon} (W(u) - \alpha \frac{u^2}{2}) dx.$$

Notice that E_e is clearly concave as soon as $\alpha \geq \max_{s \in [0,1]} |W''(s)|$. This splitting choice leads to the semi-implicit scheme

$$\begin{cases} (u^{n+1} - u^n)/\delta_t &= \Delta \mu^{n+1} \\ \mu^{n+1} &= \left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2} u^{n+1}\right) + \left(\frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n)\right), \end{cases}$$

which also reads as

$$\begin{pmatrix} I_d & -\delta_t \Delta \\ \Delta - \alpha/\varepsilon^2 & I_d \end{pmatrix} \begin{pmatrix} u^{n+1} \\ \mu^{n+1} \end{pmatrix} = \begin{pmatrix} u^n \\ \frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n) \end{pmatrix}.$$

We deduce that the couple (u^{n+1}, μ^{n+1}) can be expressed as

$$u^{n+1} = L \left[u^n + \frac{\delta_t}{\varepsilon^2} \Delta \left(W'(u^n) - \alpha u^n \right) \right] \quad \text{and} \quad \mu^{n+1} = L \left[\frac{1}{\varepsilon^2} (W'(u^n) - \Delta u^n) \right].$$

Here, the operator $L = \left(I_d + \delta_t \Delta (\Delta - \alpha/\varepsilon^2 I_d) \right)^{-1}$ can be easily computed in Fourier space where its symbol is

$$\hat{L}(\xi) = 1/(1 + \delta_t 4\pi^2 |\xi|^2 (4\pi^2 |\xi|^2 + \alpha/\varepsilon^2)).$$

A numerical scheme for the M-CH model

We now consider the case of the **M-CH** model, which reads as

$$\begin{cases} \partial_t u &= \operatorname{div}\left(M(u)\nabla\mu\right) \\ \mu &= \frac{1}{\varepsilon^2}W'(u) - \Delta u. \end{cases}$$

As previously, it is interesting to consider the following scheme :

$$\begin{cases} (u^{n+1}-u^n)/\delta_t &= \operatorname{div}\left(M(u^n)\nabla\mu^{n+1}\right) \\ \mu^{n+1} &= \frac{1}{\varepsilon}\nabla_u E_c(u^{n+1}) + \frac{1}{\varepsilon}\nabla_u E_e(u^n). \end{cases}$$

whose solution u^{n+1} is the minimizer of the auxiliary energy

$$u \longmapsto \frac{\varepsilon}{2\delta_t}\int_Q |\sqrt{M(u^n)}\nabla(\operatorname{div}(M(u^n)\nabla))^{-1}(u-u^n)|^2 dx + \overline{E}_{u^n}(u)$$

where, as before,

$$\overline{E}_{u^n}(u) = E_c(u) + E_e(u^n) + \langle \nabla_u E_e(u^n), u-u^n\rangle.$$

An argument as above shows that the proposed scheme is energy-decreasing, i.e., $E(u^{n+1}) \leq E(u^n)$. However, it raises some numerical issues for it requires the computation of the operator

$$L_{M,u^n} = \left(I_d + \delta_t \operatorname{div}(M(u^n)\nabla(\Delta - \alpha/\varepsilon^2 I_d)\right)^{-1},$$

which cannot be done easily in Fourier space. Instead, an approach using finite elements has been studied recently in [21, 301, 300].

An IMEX approach for the variational mobility term To be able to benefit from the efficiency of the Fourier transform, we propose another approach keeping in mind the variational property of mobility :

$$\begin{cases} \partial_t u &= -\nabla_\mu J_u(\mu) \\ \mu &= \frac{1}{\varepsilon}\nabla_u \overline{E}_{u^n}(u) \end{cases}$$

where

$$J_u(\mu) = \frac{1}{2}\int_Q M(u)|\nabla\mu|^2 dx.$$

As for the energy E, we decompose J as the sum of convex and concave terms $J_u = J_{u,c} + J_{u,e}$ and we consider a scheme with, respectively, an implicit and explicit treatment of the convex and concave parts :

$$\begin{cases} (u^{n+1}-u^n)/\delta_t &= -\nabla_\mu J_{u^n,c}(\mu^{n+1}) - \nabla_\mu J_{u^n,e}(\mu^n), \\ \mu^{n+1} &= \frac{1}{\varepsilon}\nabla_u E_c(u^{n+1}) + \frac{1}{\varepsilon}\nabla_u E_e(u^n). \end{cases}$$

This scheme can be interpreted as an Euler implicit discretization of

$$\begin{cases} \partial_t u &= -\nabla_\mu \overline{J}_{u^n,\mu^n}(\mu) \\ \mu &= \frac{1}{\varepsilon}\nabla_u \overline{E}_{u^n}(u), \end{cases}$$

where the new mobility energy $\overline{J}_{u^n,\mu^n}$ is given by

$$\overline{J}_{u^n,\mu^n}(\mu) = J_{u^n,c}(\mu) + J_{u^n,e}(\mu^n) + \langle \nabla_\mu J_{u^n,e}(\mu^n), \mu-\mu^n\rangle.$$

To ensure the decreasing of $t \mapsto \overline{E}_{u^n}(u(\cdot,t))$ along the flow, we require at least the semi-implicit metric $\overline{J}_{u^n,\mu^n}$ to be non negative. This corresponds to the concavity condition on $J_{u,e}$, meaning that we have

$$0 \leq J_{u^n}(\mu) \leq \overline{J}_{u^n,\mu^n}(\mu).$$

Moreover, from the identity

$$\frac{d}{dt}\overline{E}_{u^n}(u) = \langle \nabla_u \overline{E}_{u^n}(u), \partial_t u\rangle = -\frac{1}{\varepsilon}\langle \mu, \nabla_\mu \overline{J}_{u^n,\mu^n}(\mu)\rangle,$$

we conclude that it is sufficient to show that

$$\langle \mu, \nabla_\mu \overline{J}_{u^n,\mu^n}(\mu)\rangle \geq 0.$$

to ensure the decreasing of the energy.

Application to the M-CH model Motivated by the previous section, we consider the following convex/-concave splitting $J_{u^n} = J_{u^n,c} + J_{u^n,e}$ with :

$$J_{u^n,c}(\mu) = \frac{1}{2}\int m|\nabla\mu|^2 dx \quad \text{and} \quad J_{u^n,e}(\mu) = \frac{1}{2}\int (M(u^n)-m)|\nabla\mu|^2 dx$$

with $m > 0$. We take $m = \max_{s\in[0,1]}\{M(s)\}$ in order to obtain the concavity of $J_{u^n,e}(\mu)$, and the scheme reads as

$$\begin{cases} (u^{n+1}-u^n)/\delta_t & = m\Delta\mu^{n+1} + \operatorname{div}((M(u^n)-m)\nabla\mu^n) \\ \mu^{n+1} & = \left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2}u^{n+1}\right) + \left(\frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n)\right), \end{cases}$$

or, in matrix form,

$$\begin{pmatrix} I_d & -\delta_t m\Delta \\ \Delta - \alpha/\varepsilon^2 & I_d \end{pmatrix}\begin{pmatrix} u^{n+1} \\ \mu^{n+1}\end{pmatrix} = \begin{pmatrix} u^n + \delta_t \operatorname{div}((M(u^n)-m)\nabla\mu^n) \\ \frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n). \end{pmatrix} = \begin{pmatrix} B^1_{u^n,\mu^n} \\ B^2_{u^n,\mu^n}\end{pmatrix}$$

Finally, the couple (u^{n+1}, μ^{n+1}) can be expressed as

$$u^{n+1} = L_M\left[B^1_{u^n,\mu^n} + \delta_t m\Delta B^2_{u^n,\mu^n}\right]$$

and

$$\mu^{n+1} = L_M\left[(-\Delta B^1_{u^n,\mu^n} + \alpha/\varepsilon^2 B^1_{u^n,\mu^n}) + B^2_{u^n,\mu^n}\right],$$

where the operator L_M is now given by $L_M = \left(I_d + \delta_t m\Delta(\Delta - \alpha/\varepsilon^2 I_d)\right)^{-1}$, which can be computed efficiently in Fourier space.

Case of the NMN-CH model

We now turn to the **NMN-CH** model :

$$\begin{cases} \partial_t u & = N(u)\operatorname{div}(M(u)\nabla(N(u)\mu)) \\ \mu & = \frac{1}{\varepsilon^2}W'(u) - \Delta u, \end{cases}$$

where $N(u) = \frac{1}{\sqrt{M(u)}}$ and $M(u) = 2W(u) + \gamma\varepsilon^2$.

In a similar manner to the other models, we study the model rewritten in a variational form

$$\begin{cases} \partial_t u & = -\nabla_\mu J_u(\mu) \\ \mu & = \frac{1}{\varepsilon}\nabla_u \overline{E}_{u^n}(u) \end{cases}$$

with

$$J_u(\mu) = \frac{1}{2}\int_Q M(u)\left|\nabla(N(u)\mu)\right|^2 dx.$$

J_u can be split in three parts :

$$J_u(\mu) = \frac{1}{2}\int_Q |\nabla\mu|^2 dx + \int_Q G(u)\cdot\nabla\mu\,\mu dx + \frac{1}{2}\int_Q |G(u)|^2\mu^2 dx,$$

with

$$G(u) = -\frac{1}{2}\nabla(\log(M(u)))$$

as $N(u) = \frac{1}{\sqrt{M(u)}}$ and $\sqrt{M(u)}\nabla(N(u)) = -\frac{1}{2}\frac{\nabla M(u)}{M(u)} = -\frac{1}{2}\nabla(\log(M(u)))$.
This suggests that we could use the following splitting $J_u(\mu) = J_{u,c}(\mu) + J_{u,e}(\mu)$ with

$$J_{u,c}(\mu) = \frac{1}{2}\int_Q m|\nabla\mu|^2 dx + \frac{1}{2}\int_Q \beta\mu^2 dx$$

and

$$J_{u,e}(\mu) = 2\int_Q G(u)\cdot\nabla\mu\mu dx + \frac{1}{2}\int_Q (|G(u)|^2 - \beta)\mu^2 dx + \frac{1}{2}\int_Q (1-m)|\nabla\mu|^2 dx,$$

with $\beta > 0$ and $m > 0$. As soon as $G(u)$ is bounded in $H^1(Q)$, a sufficiently large choice for m and β should ensure the concavity of $J_{u,e}(\mu)$, as the first term can be bounded by a combination of the other two quadratic terms. In practice, we take $m = 1$ and $\beta = 1/\varepsilon^2$ for our numerical experiments and these values did not show any sign of instability whatever the choice of the time step δ_t.

The above splitting leads to the following system

$$\begin{cases} (u^{n+1}-u^n)/\delta_t &= m\Delta\mu^{n+1} - \beta\mu^{n+1} + H(u^n,\mu^n) \\ \mu^{n+1} &= \left(-\Delta u^{n+1} + \frac{\alpha}{\varepsilon^2}u^{n+1}\right) + \left(\frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n)\right), \end{cases}$$

where

$$H(u^n,\mu^n) = N(u^n)\,\mathrm{div}(M(u^n)\nabla(N(u^n)\mu^n)) - m\Delta\mu^n + \beta\mu^n$$

The couple (u^{n+1},μ^{n+1}) is then solution to the system

$$\begin{pmatrix} I_d & -\delta_t(m\Delta - \beta I_d) \\ \Delta - \alpha/\varepsilon^2 & I_d \end{pmatrix}\begin{pmatrix} u^{n+1} \\ \mu^{n+1}\end{pmatrix} = \begin{pmatrix} u^n + \delta_t H(u^n,\mu^n) \\ \frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n). \end{pmatrix} = \begin{pmatrix} B^1_{u^n,\mu^n} \\ B^2_{u^n,\mu^n}\end{pmatrix}$$

satisfying

$$u^{n+1} = L_{NMN}\left[B^1_{u^n,\mu^n} + \delta_t(m\Delta B^2_{u^n,\mu^n} - \beta B^2_{u^n,\mu^n})\right]$$

and

$$\mu^{n+1} = L_{NMN}\left[(-\Delta B^1_{u^n,\mu^n} + \alpha/\varepsilon^2 B^1_{u^n,\mu^n}) + B^2_{u^n,\mu^n}\right],$$

with $L_{NMN} = \left(I_d + \delta_t(m\Delta - \beta I_d)(\Delta - \alpha/\varepsilon^2 I_d)\right)^{-1}$ whose action is easy to compute in Fourier space.

7.5.3 Matlab code

We provide in Figure 7.1 an example of **Matlab** script with less than 40 lines which implements the scheme approximating the solutions to the **NMN-CH** model, see the previous section. In particular :

— We consider here a computation box $Q = [-1/2,1/2]^2$ discretized with $N_s = 2^9$ nodes in each direction. The initial condition of u is a uniform noise and the numerical parameters are $\varepsilon = 2/N_s$, $\delta_t = 4\varepsilon^2$, $\alpha = 2$, $\beta = 2/\varepsilon^2$, and $m = 1$.

— Line 14 corresponds to the definition of the Fourier symbol associated with operator L_{NMN}. The action of L_{NMN} can then be computed using a simple multiplication in Fourier space with the array M_{LNMN}.

— The computation of $N(u)\,\mathrm{div}(M(u)\nabla(N(u)\mu))$ is made on line 28 and is based on the following equality

$$\begin{aligned} N(u)\,\mathrm{div}(M(u)\nabla(N(u)\mu)) &= \sqrt{M(u)}\Delta(N(u)\mu) + N(u)\nabla(M(u))\cdot\nabla(N(u)\mu) \\ &= \sqrt{M(u)}\Delta(N(u)\mu) + 2\nabla\left[\sqrt{M(u)}\right]\cdot\nabla(N(u)\mu), \end{aligned}$$

because $N(u) = 1/\sqrt{M(u)}$.

— Each computation of gradient and divergence operators are made in Fourier space. For instance the gradient of $\sqrt{M(u)}$ is computed on line 23.

— Figure 8.1 shows the phase field function u^n computed at different times t^n by using this script.

We believe that this implementation emphasizes the simplicity, efficiency, and stability of our numerical scheme.

Computational cost : our schemes rely on the fast Fourier transform which requires $\mathcal{O}(N_s^d\log(N_s))$ operations, where $d = 2,3$ is the dimension. Here are the numbers of Fourier transforms per time iteration for each of our numerical schemes : **C-CH** requires 2 Fourier transforms, **M-CH** needs 6 and, for the last scheme **NMN-CH**, 12 Fourier transforms are performed per iteration. The overall cost is therefore fairly small for all three schemes.

Asymptotic expansion and flow : numerical comparison of the models

The first numerical example concerns the evolution of an initial connected set. For each Cahn–Hilliard model, we plot on Figure 7.3 the numerical phase field u^n computed at different times. Each experiment is performed using the same numerical parameters : the spatial resolution $\delta_x = \frac{1}{2^8}$, $\varepsilon = 2\delta_x$, $\delta_t = \varepsilon^4$, $\alpha = 2/\varepsilon^2$, $m = 1$, and $\beta = 2/\varepsilon^2$. The first, second and third lines of Figure 7.3 show the approximate solutions to the **C-CH**, **M-CH**, and **NMN-CH** models, respectively. For all three models, the stationary limit appears to correspond to a ball of the same mass as that of the initial set. As expected, the **C-CH** model, whose limit flow is the Hele-Shaw model [280, 6] gives a slightly different flow compared to the other two models. The difference is rather small in the symmetric and simple situation of Figure 7.3, but more differences will appear in the other experiments. On the other hand, the numerical flows obtained with the **M-CH** and **NMN-CH** models are very similar and, according to our asymptotic analysis, they are close to the surface diffusion flow. We recall that we multiplied the model **M-CH** by a factor $\frac{1}{c_N^2} = 36$ so that it evolves at the same velocity as model **NMN-CH**. Thus, in general, the **NMN-CH** flow is expected to evolve 36 times faster than the **M-CH** flow.

```
clear all;
%%%%%%%%%%%%%%%%%% Numerical parameters %%%%%%%%%%%%%%%%%%%%%%%%
Ns = 2^9; epsilon =1/Ns; dt =epsilon^4; T =1;
%%%%%%%%%%%%%%%%%% Double well potential, mobilities %%%%%%%%%%%
W = @(U) 1/2*(U.*(U-1)).^2;
W_prim = @(U) (U.*(U-1).*(2*U-1));
MobM = @(U) 1/2*((((U).*(1-U)).^2+epsilon^2) );
MobN = @(U) 1./sqrt(MobM(U) );

%%%%%%%%%%%%%%%% Fourier operators %%%%%%%%%
k = [0:Ns/2,-Ns/2+1:-1]; [K1,K2] = meshgrid(k,k);
Delta = -4*pi^2*((K1.^2 + (K2).^2));
alpha = 2; beta = 1/epsilon^2; m = 1;
M_LNMN = 1./(1 + dt*(m*Delta - beta) .*(Delta - alpha/epsilon^2));

%%%%%%%%%%%%%%%%%% Initial condition %%%%%%%%%%%
U = rand(Ns,Ns); U_fourier = fft2(U);
Mu = zeros(Ns,Ns); Mu_fourier = zeros(Ns,Ns);
%%%%%%%%%%%%%%%%%% Scheme loop %%%%%%%%%%%
for i=1:T/dt,
mobMU = MobM(U); mobNU = MobN(U);
sqrtM = sqrt(mobMU); sqrtM_fourier = fft2(sqrtM);
nabla1_sqrtM= real(ifft2(2*pi*1i*K1.*sqrtM_fourier )); nabla2_sqrtM= real(ifft2(2*pi*1i
    *K2.*sqrtM_fourier ));

muN_fourier = fft2(Mu.*mobNU); muN = real(ifft2(muN_fourier));
nabla1_muN = real(ifft2(2*pi*1i*K1.*muN_fourier )); nabla2_muN = real(ifft2(2*pi*1i*K2
    .*muN_fourier ));
laplacien_muN = real(ifft2(Delta.*muN_fourier ));
NdivMgradNMu = sqrtM.*laplacien_muN + 2*(nabla1_sqrtM.*nabla1_muN +nabla2_sqrtM.*
    nabla2_muN);

B1 = U_fourier + dt*(fft2(NdivMgradNMu) - (m*Delta-beta).*Mu_fourier);
B2 = fft2(W_prim(U)/epsilon^2 - alpha/epsilon^2*U);

U_fourier = M_LNMN.*(B1 + dt*(m*Delta-beta).*B2);
U = real(ifft2(U_fourier));
Mu_fourier = M_LNMN.*((alpha/epsilon^2 - Delta).*B1 + B2);
Mu = real(ifft2(Mu_fourier));

end
```

FIGURE 7.1 – A **Matlab** implementation in dimension 2 of the numerical scheme proposed in this paper to approximate the solutions to the **NMN-CH** model, see section 7.5.2. Extension to higher dimension is straightforward.

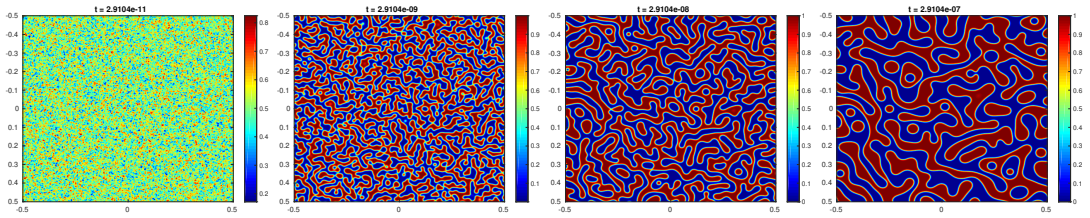


FIGURE 7.2 – Numerical approximation of solutions to the **NMN-CH** model starting from the left configuration and using the **Matlab** script of Figure 7.1.

Going further in the analysis of this numerical example to illustrate the asymptotic expansion of Section 7.4 and to highlight the advantages of the **NMN** model to approximate surface diffusion, we show on the first two plots of Figure 7.4 the slice $x_1 \mapsto u^T(x_1, 0)$ of the solution at the final time $T = 10^{-4}$ (the middle plot represents a zoom in the right transition zone of the slice). The profile associated to the **C-CH** model is plotted in red and clearly indicates that the solution u^T does not remain in $[0, 1]$ with an overshoot of order $O(\varepsilon)$. A similar behavior with an overshoot of order $O(\varepsilon)$ holds for the **M-CH** model (in blue). In contrast, the profile obtained using the **NMN** model (in green) seems to be very close to the optimal profile $q(d(\cdot)/\varepsilon)$ and almost remains in $[0, 1]$ up to an error of order $O(\varepsilon^2)$. The last plot of Figure 7.4 shows the evolution of the Cahn–Hilliard energy along the flow for each model. We can clearly observe a decrease of the energy in all cases.

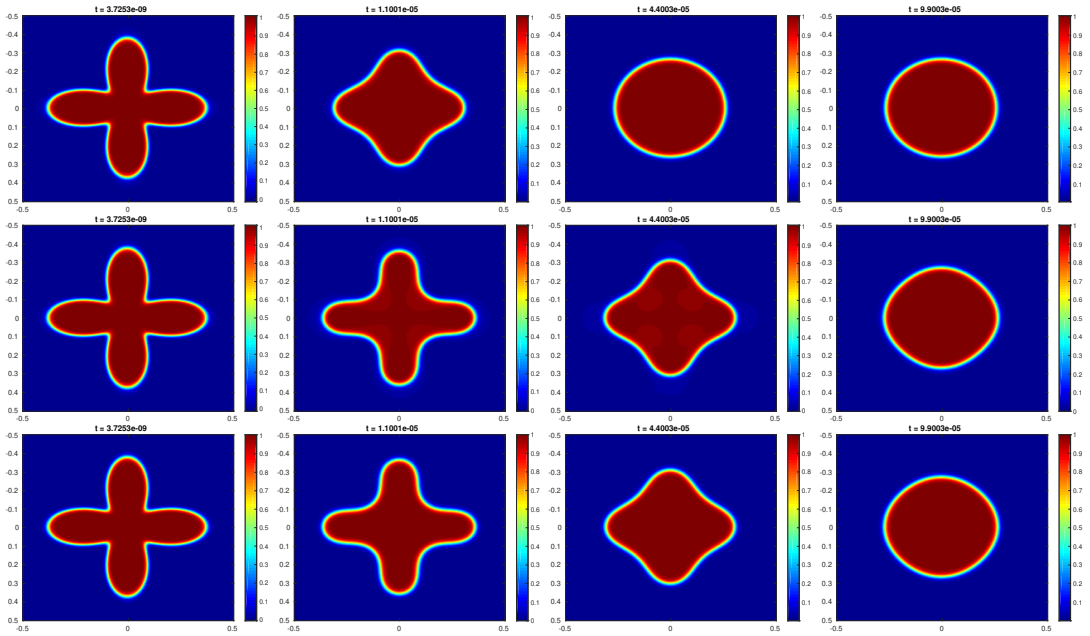


FIGURE 7.3 – Time evolution of the numerical phase field solutions to the three **CH** models. First line : **C-CH** ; Second line : **M-CH** ; Third line : **NMN-CH**.

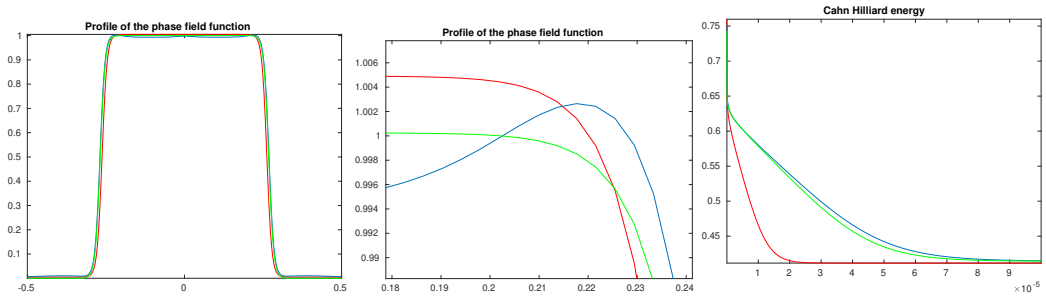


FIGURE 7.4 – First plot : representation of the slice $x_1 \mapsto u^T(x_1, 0)$ where u^T is the numerical phase field solution at the final time to **C-CH** (in red), **M-CH** (in blue), and **NMN-CH** (in green). Middle plot : Zoom in on the right transition zone of the slice. Right plot : time evolution of the Cahn–Hilliard energy for each model (same color code).

Influence of the mobility on the local conservation of mass

The second numerical experiment is intended to show the advantage of introducing mobility in the classical Cahn–Hilliard model to ensure local mass conservation. As previously, we use the same numerical parameters in each case : $N_s = 2^8$, $\delta_x = \frac{1}{2^8}$, $\varepsilon = 2/N_s$, $\delta_t = \varepsilon^4$, $\alpha = 2/\varepsilon^2$, $m = 1$, and $\beta = 2/\varepsilon^2$. Figure 8.2 shows the phase field function u^n obtained at different times using the different phase field models (first line : **C-CH**, second line : **M-CH**, third line **NMN-CH**). The initial set is a disjoint union of five small sets. As expected, the evolutions obtained using the **M-CH** and **NMN-CH** models show an independent evolution of each small disjoint set converging to a ball of equivalent volume. This is clearly not the case using the **C-CH** model for which the limit appears to be the union of three balls only. It suggests that the mass of the smaller set moves toward the larger set. This emphasizes the interest of plugging mobility in the Cahn–

Hilliard model to ensure a local conservation of mass, which is particularly relevant for various physical applications, for example the simulation of dewetting phenomena.

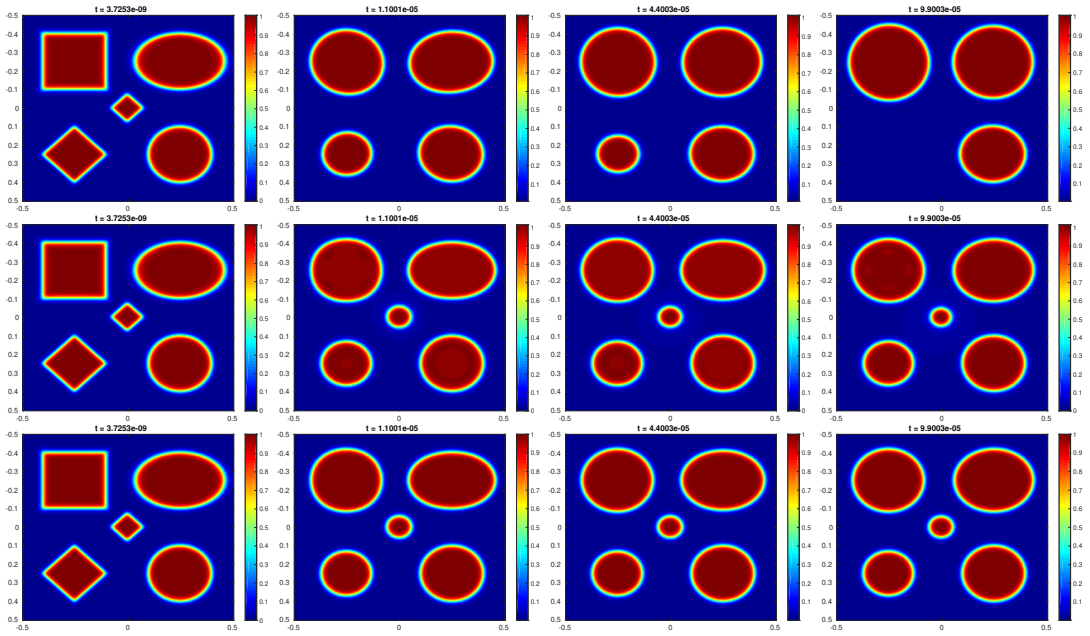


FIGURE 7.5 – Questioning the local conservation of mass. The evolution of the numerical phase fields associated with the three models is shown. First line : **C-CH** ; second line : **M-CH** ; third line : **NMN-CH**.

Evolution of a thin structure in dimension 3

We study now a numerical experiment in dimension 3 where the initial set is a very thin tube with volume of order $O(\varepsilon^2)$ in a non cubic domain $[-\frac{1}{2},\frac{1}{2}]\times[-\frac{1}{8},\frac{1}{8}]\times[-\frac{1}{8},\frac{1}{8}]$. Our motivation is to show the importance of having a second order accuracy of the model in the complicated case of such a thin structure evolution. Similarly to the previous computations, the numerical parameters are $N_s = 2^8$, $\delta_x = \frac{1}{2^8}$, $\varepsilon = 2/N_s$, $\delta_t = \varepsilon^4$, $\alpha = 2/\varepsilon^2$, $m = 1$, and $\beta = 2/\varepsilon^2$. We plot on each picture of Figure 8.6 the $1/2$-level set of the numerical approximation $\tilde{u}_\varepsilon$ of the solution u_ε for different times t. The first, second and third line correspond to the **C-CH**, **M-CH** and **NMN-CH** models, respectively. We observe that the thin structure disappears when the **C-CH** and **M-CH** models are used, whereas the **NMN-CH** model seems to have better phase and volume conservation properties, and the stationary set is the union of five small spheres.

The results are surprising at first glance as the mass $\int_Q u_\varepsilon dx$ of the exact solution u_ε is preserved using the **C-CH** and **M-CH** models. To convince ourselves that the problem arises from the models' first order accuracy and not the numerical approximation, we plot on Figure 7.7 for each model the evolution along the flow of the mass $t \mapsto \int_Q \tilde{u}_\varepsilon dx$ of the numerical approximation $\tilde{u}_\varepsilon$ of u_ε. We observe an excellent mass conservation with both **C-CH** and **M-CH** models despite the final disappearance of the structure. How can we explain this ? Recall that the structure plotted on Figure 8.6 is the numerical approximation of the $1/2$-level set of u_ε

$$\Omega_\varepsilon(t) = \{x \in Q,\ u_\varepsilon(x,t) \leq 1/2\}\,.$$

However, for both **C-CH** and **M-CH** models, the volume of $\Omega_\varepsilon(t)$ approximates the integral of u_ε with a $O(\varepsilon)$ error :

$$\mathrm{Vol}(\Omega_\varepsilon(t)) = \int_Q u_\varepsilon(x,t)dx + O(\varepsilon).$$

The volume of $\Omega_\varepsilon(t)$ is therefore conserved with an error of order $O(\varepsilon)$, this appears to be not sufficient here to preserve the volume in $O(\varepsilon^2)$ of the thin structure. In contrast, the **NMN-CH** model satisfies

$$\mathrm{Vol}(\Omega_\varepsilon(t)) = \int_Q G(u_\varepsilon(x,t))dx + O(\varepsilon^2)$$

and since in this example $\int_Q G(u_\varepsilon(x,t))dx = \int_Q u_\varepsilon(x,t)dx + O(\varepsilon^2)$, we obtain an explanation of the better conservation of the structure. To conclude, this 3D numerical experiment illustrates that the **NMN-CH** model is more suitable than **C-CH** and **M-CH** for the consistent approximation of the evolution of a very thin structure.

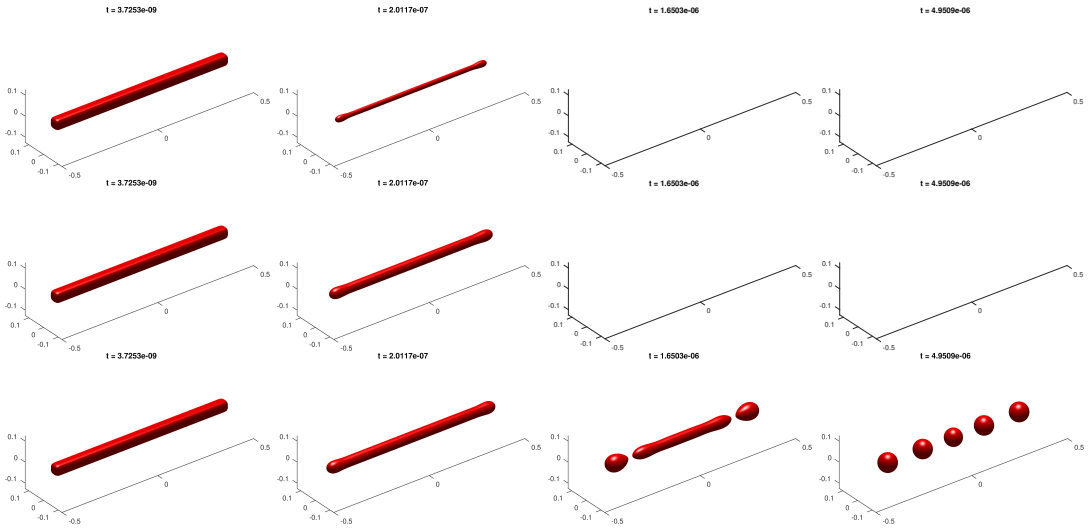


FIGURE 7.6 – Comparison of the different models applied to a thin structure in dimension 3. The first line shows the evolution of the level set $\{u_\varepsilon \leq \frac{1}{2}\}$ computed with the **C-CH** model, the second line with **M-CH**, and the third line with **NMN-CH**.

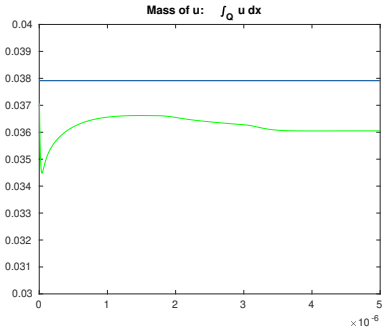


FIGURE 7.7 – Comparison of the different models applied to a thin structure in dimension 3. The graphs correspond to the time evolution of the mass $\int_Q u_\varepsilon dx$ of u_ε computed with the **C-CH** and **M-CH** models (same blue graph) and the **NMN-CH** model (in green).

Dewetting and surface diffusion of a thin plate

Figure 8.5 shows the last numerical example : the evolution of a thin plate using the **NMN-CH** model. As previously, the parameters are $N_s = 2^8$, $\delta_x = \frac{1}{2^8}$, $\varepsilon = 2/N_s$, $\delta_t = \varepsilon^4$, $\alpha = 2/\varepsilon^2$, $m = 1$, and $\beta = 2/\varepsilon^2$. We can observe an evolution very similar to the one observed in real dewetting experiments [21].

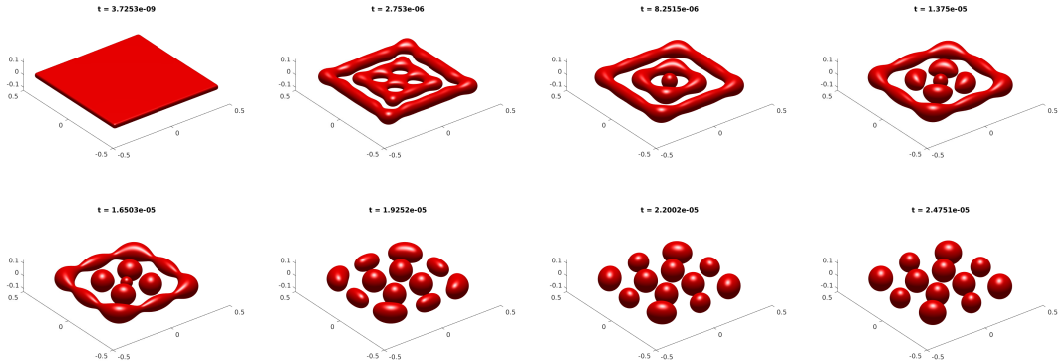


FIGURE 7.8 – Example of dewetting in dimension 3 using the **NMN-CH** model. Evolution of $\{u_\varepsilon \leq 1/2\}$ along the iterations.

Chapitre 8

A multiphase Cahn-Hilliard system with mobility for the simulation of wetting

8.1 Introduction

The wetting or dewetting phenomenon describes the behavior of a liquid phase put in contact with a fixed solid surface. This paper is devoted to the phase field approximation of this phenomenon as a multiphasic surface diffusion flow with mobilities. Young [340] identified in 1805 the optimal shape of the liquid phase and proposed the following law for the contact angle θ between the liquid and the solid :

$$\cos(\theta) = \frac{\sigma_{SV} - \sigma_{LS}}{\sigma_{VL}},$$

where $\sigma_{SV}, \sigma_{LS}, \sigma_{VL}$ represent the surfaces tensions of the interfaces solid-vapor Γ_{SV}, liquid-solid Γ_{LS} and vapor-liquid Γ_{VL}, respectively. Mathematically, Young's law can be derived by minimizing the total energy in the solid-liquid-vapor system. Ignoring gravity, this total energy reads as

$$\mathcal{E} = \sigma_{SV}\mathcal{H}^{d-1}(\Gamma_{SV}) + \sigma_{LS}\mathcal{H}^{d-1}(\Gamma_{LS}) + \sigma_{VL}\mathcal{H}^{d-1}(\Gamma_{VL}).$$

which is a particular instance of the generic L-phase perimeter

$$P(\Omega_1, \dots, \Omega_L) = \frac{1}{2}\sum_{i,j=1}^{L} \sigma_{i,j}\mathcal{H}^{d-1}(\Gamma_{i,j}), \tag{8.1.1}$$

where $\{\Omega_1, \dots, \Omega_L\}$ is an open partition of an open bounded domain $\Omega \subset \mathbb{R}^d$ and, for all $i, j \in \{1, \dots, L\}$, $\Gamma_{i,j} = \bar{\Omega}_i \cap \bar{\Omega}_j \cap \Omega$ is the interface between phases i, j, and $\sigma_{i,j}$ is the surface tension along this interface. To ensure the lower semicontinuity of the L-phase perimeter, see [259, 243, 86], we assume that the surface tensions are positive, i.e. $\sigma_{i,j} > 0$, and satisfy the triangle inequality

$$\sigma_{i,j} + \sigma_{j,k} \geq \sigma_{i,k} \quad \text{for any } i, j, k,$$

The evolution of the physical system can then be approximated by a multiphasic surface diffusion flow. This motion can be viewed as the H^{-1} gradient flow of the energy (8.1.1) which ensures its decay while maintaining locally the volume of each phase. In particular, the normal velocity V_{ij} at the interface Γ_{ij} reads as

$$\frac{1}{\nu_{ij}} V_{ij} = \sigma_{ij}\Delta_{\Gamma_{ij}(t)} H_{ij}(t),$$

where $H_{ij}(t)$ denotes the scalar mean curvature on $\Gamma_{ij}(t)$, $\Delta_{\Gamma_{ij}(t)}$ is the Laplace-Beltrami operator on the surface, and $\nu_{ij} > 0$ is the surface mobility coefficient. The above expression is the classical form of the velocity in this context, but it can be obviously rewritten to incorporate the degenerate no-motion case where $\nu_{ij} = 0$:

$$V_{ij} = \nu_{ij}\sigma_{ij}\Delta_{\Gamma_{ij}(t)} H_{ij}(t).$$

The simulation of wetting or dewetting requires $L = 3$ phases : the liquid phase Ω_L, the solid phase Ω_S and the vapor phase Ω_V. Moreover, as the surface tension coefficients $(\sigma_{LV}, \sigma_{SV}, \sigma_{SL})$ satisfy the triangle inequality, they form an additive set of coefficients, i.e. there exists three positive coefficients $\sigma_L, \sigma_S, \sigma_V$ such that

$$\sigma_{LV} = \sigma_L + \sigma_V, \quad \sigma_{SV} = \sigma_S + \sigma_V \text{ and } \quad \sigma_{SV} = \sigma_S + \sigma_V.$$

The surface mobilities can be set to

$$(\nu_{LV}, \nu_{SV}, \nu_{SL}) = (1, 0^+, 0^+),$$

in order to fix the solid phase. This set of coefficients is harmonically additive in the sense that, with the convention $\frac{1}{0^+} = +\infty$, there exist three non negative coefficients ν_S, ν_L and ν_V such that

$$\nu_{LV}^{-1} = \nu_L^{-1} + \nu_V^{-1}, \quad \nu_{SV}^{-1} = \nu_S^{-1} + \nu_V^{-1} \text{ and } \quad \nu_{SL}^{-1} = \nu_S^{-1} + \nu_L^{-1}.$$

Indeed, we can just consider $\nu_S = 0^+$ and $\nu_L = \nu_V = 2$.

Having in mind the application to wetting, we assume in the rest of the paper that :

— the surface tensions are additive, i.e. there exist coefficients $\sigma_i \geq 0$, $i \in \{1, \dots, L\}$, such that $\sigma_{ij} = \sigma_i + \sigma_j, \forall i, j \in \{1, \dots, L\}$;

— the mobility coefficients are harmonically additive, i.e. there exist nonnegative coefficients ν_i satisfying $\nu_{ij}^{-1} = \nu_i^{-1} + \nu_j^{-1}$ (with the convention that $\frac{1}{0^+} = +\infty$).

With such assumptions, it is easy to reformulate the expression of the L-phase perimeter in the more convenient following form

$$P(\Omega_1, \dots, \Omega_L) = \sum_{i=1}^{L} \sigma_i P(\Omega_i), \tag{8.1.2}$$

which can be approximated in the sense of Γ-convergence by a sum of scalar Cahn-Hilliard energies defined for every smooth $\mathbf{u} = (u_1, \dots, u_L) \in \mathbb{R}^L$ by

$$P_\varepsilon(\mathbf{u}) = \begin{cases} \displaystyle\sum_{k=1}^{L} \sigma_k \int_\Omega \left(\frac{\varepsilon}{2} |\nabla u_k|^2 + \frac{1}{\epsilon} W(u_k) \right) dx & \text{if } \sum_{k=1}^{L} u_k = 1, \\ +\infty & \text{otherwise,} \end{cases}$$

In this definition each u_i represents a smooth approximation of the characteristic function $\mathbb{1}_{\Omega_i}$, $W(s) = \frac{s^2(1-s)^2}{2}$ is a double-well potential, and the parameter ε characterizes the width of the diffuse interface, i.e. how concentrated is each $\nabla u_i dx$ around the Hausdorff measure supported on the reduced boundary of Ω_i.

Since the multiphase surface diffusion flow is the H^{-1}-gradient flow of (8.1.2), a natural idea to approximate it is to consider the H^{-1}-gradient flow of P_ε which yields the following Cahn-Hilliard system

$$\begin{cases} \varepsilon^2 \partial_t u_k = \nu_k \Delta \left(\sigma_k \mu_k + \lambda \right) \\ \mu_k = W'(u_k) - \varepsilon^2 \Delta u_k, \end{cases}$$

where λ is the Lagrange multiplier associated with the partition constraint $\sum u_k = 1$. Here, we follow the idea of [67] to handle the set $\{\nu_{i,j}\}$ of mobilities and we use explicitly its harmonic additive decomposition.

However, the asymptotic expansion of this phase field system is not clear and to the best of our knowledge, no analysis of its convergence has been made so far. The main obstacle to overcome is the non local nature of the system, which is particularly significant in the multiphase case.

Based on the results of the previous chapter, we propose instead to extend the **M-CH** and **NMN-CH** models to the case of L phases. From the modelling viewpoint, this amounts to integrating in the model the influence of surface tensions σ_{ij} and phase mobilities ν_{ij}. To this end, we adapt to the Cahn-Hilliard system the work of [67] done for the Allen-Cahn system. In particular, we propose to analyze the two following phase field models, where in both cases λ is the Lagrangian multiplier which encodes the partition constraint $\sum_{k=1}^{L} u_k = 1$:

— The **M-CH** multiphase field model defined for $k \in \{1, \dots, L\}$ by :

$$\begin{cases} \varepsilon^2 \partial_t u_k = \nu_k \operatorname{div} \left(M(u_k) \nabla (\sigma_k \mu_k + \lambda) \right), \\ \mu_k = W'(u_k) - \varepsilon^2 \Delta u_k, \end{cases} \tag{8.1.3}$$

with mobility $M(s) = 2W(s)$.

— The **NMN-CH** multiphase field model defined for $k \in \{1, \dots, L\}$ by :

$$\begin{cases} \varepsilon^2 \partial_t u_k = \nu_k N(u_k) \operatorname{div} \left(M(u_k) \nabla (\sigma_k N(u_k) \mu_k + \lambda) \right), \\ \mu_k = W'(u_k) - \varepsilon^2 \Delta u_k. \end{cases} \tag{8.1.4}$$

with mobilities $M(s) = 2W(s)$ and $N(s) = 1/\sqrt{M(s)}$. This model is well defined whenever $u \neq 0, 1$, which is the case near the interface $\{u = \frac{1}{2}\}$. To give sense to the model in the whole domain, it can be rewritten in two different ways :

— either by transferring N to the left-hand side to obtain the alternative model

$$\text{(NMN-CH reformulation I)} \qquad \begin{cases} \varepsilon^2 g(u_k)\partial_t u_k = \nu_k \operatorname{div}\left(M(u_k)\nabla(\sigma_k\mu_k + \lambda)\right), \\ g(u_k)\mu_k = W'(u_k) - \varepsilon^2\Delta u_k. \end{cases} \tag{8.1.5}$$

where $g(u_k) = \sqrt{M(u_k)}$ is always well-defined. Such a reformulation (strictly equivalent where $M(u_k)$ does not vanish) will be used for the asymptotic expansion.

— or by modifying the definitions of M, N to prevent them from vanishing while preserving the conclusions of the asymptotic expansion. This is the case with the following model :

$$\text{(NMN-CH reformulation II)} \qquad \begin{cases} \partial_t u_k &= \nu_k \tilde{N}(u_k) \operatorname{div}\left(\tilde{M}(u_k)\nabla\tilde{N}(u_k)(\sigma_k\mu_k + \lambda)\right) \\ \mu_k &= \frac{W'(u_k)}{\varepsilon^2} - \Delta u_k \end{cases}$$

where the mobilities $\tilde{M}$ and $\tilde{N}$ are defined by $\tilde{M}(s) = 2W(s) + \gamma\varepsilon^2$ and $\tilde{N}(s) = \frac{1}{\sqrt{\tilde{M}(s)}}$, with $\gamma > 0$. Obviously, $\tilde{M}$ never vanishes and $\tilde{N}$ is well-defined everywhere. We will explain in the first lines of Section 8.2.3 why this reformulation has the same asymptotic properties as the original model (8.1.4). The **NMN-CH** reformulation II model will be used for numerical approximation (because numerical errors require a choice for M that prevents cancellations).

8.1.1 Outline of this chapter

We first proceed to a formal asymptotic analysis of the **M-CH** and **NMN-CH** multiphase models. In particular, we show that the limit law of each model is indeed the multiphase surface diffusion flow with the advantage that **NMN-CH** guarantees an approximation error of order 2 in ε. In a second section, devoted to numerical approximation, we first introduce a numerical scheme suitable for both models. This scheme is based on a Fourier-spectral convex-concave semi implicit approach in the spirit of [158, 73]. We provide numerical experiments which illustrate the stability of our scheme and the asymptotic properties of both phase field models. In the last section, we consider the special case of the wetting / dewetting phenomenon for which we derive a simplified model using the liquid phase only. We illustrate this model with 3D numerical experiments using either smooth or rough surfaces, and choosing various set of parameters to get different Young angle conditions.

8.2 Formal matched asymptotic expansions

In this section, we give a formal proof of Propositions 8.2.1 and 8.2.2 below using the method of matched asymptotic expansions. As in the previous chapter, these results involve the so-called *optimal profile* q associated with the potential W and defined by the equation $q'(z) = -\sqrt{2W(q(z))}$ with a suitable constraint on $q(0)$. We recall here that in the case where $W(s) = \frac{1}{2}s^2(1-s)^2$ and $q(0) = \frac{1}{2}$, one gets

$$q(z) = \frac{1 - \tanh\left(\frac{z}{2}\right)}{2},$$

and the following constants are also used in both propositions :

$$c_W = \int_{\mathbb{R}} \left(q'(z)\right)^2 dz, \quad c_M = \int_{\mathbb{R}} M(q(z))dz \quad \text{and} \quad c_N = \int_{\mathbb{R}} \frac{q'(z)}{N(q(z))}dz.$$

Remark that with our particular choices for N and q, one has $c_N = -c_W$.

Proposition 8.2.1. *For $i, j \in \{1, \dots, L\}$ with $i \neq j$, let $\Omega_i^\varepsilon = \{x,\ u_i(x) \geq \frac{1}{2}\}$ and*

$$\Gamma_{ij}^\varepsilon = \partial\Omega_i^\varepsilon \cap \{x,\ u_j \geq u_k,\ k \in \{1, \dots, L\} \setminus \{i\}\}.$$

The solution $\mathbf{u}^\varepsilon$ to the **M-CH** *model defined for $k \in \{1, \dots, L\}$ by*

$$\begin{cases} \varepsilon^2\partial_t u_k = \nu_k \operatorname{div}\left(M(u_k)\nabla(\sigma_k\mu_k + \lambda)\right), \\ \mu_k = W'(u_k) - \varepsilon^2\Delta u_k, \end{cases}$$

satisfies (formally) near the interface Γ_{ij}^ε the following asymptotic expansions :

$$\begin{cases} u_i^\varepsilon = q\left(\dfrac{\mathrm{dist}(x,\Omega_i^\varepsilon)}{\varepsilon}\right) + \mathcal{O}(\varepsilon), \\ u_j^\varepsilon = 1 - q\left(\dfrac{\mathrm{dist}(x,\Omega_i^\varepsilon)}{\varepsilon}\right) + \mathcal{O}(\varepsilon), \\ u_k^\varepsilon = \mathcal{O}(\varepsilon). \end{cases}$$

where $\mathrm{dist}(\cdot,\Omega_i^\varepsilon)$ denotes the signed distance function to Ω_i^ε.
Moreover, the normal velocity V_{ij}^ε at the interface Γ_{ij}^ε satisfies (formally) :

$$\frac{1}{\nu_{ij}} V_{ij}^\varepsilon = \sigma_{ij} c_M c_W \Delta_\Gamma H_{ij} + \mathcal{O}(\varepsilon).$$

Proposition 8.2.2. *With the notations of Proposition 8.2.1, the solution $\mathbf{u}^\varepsilon$ to the* **NMN-CH** *model defined for $k \in \{1,\dots,L\}$ by*

$$\begin{cases} \varepsilon^2 \partial_t u_k = \nu_k N(u_k)\,\mathrm{div}\left(M(u_k)\nabla(\sigma_k N(u_k)\mu_k + \lambda)\right), \\ \mu_k = W'(u_k) - \varepsilon^2 \Delta u_k, \end{cases}$$

satisfies (formally) near the interface Γ_{ij}^ε the following asymptotic expansions :

$$\begin{cases} u_i^\varepsilon = q\left(\dfrac{\mathrm{dist}(x,\Omega_i^\varepsilon)}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2), \\ u_j^\varepsilon = 1 - q\left(\dfrac{\mathrm{dist}(x,\Omega_i^\varepsilon)}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2), \\ u_k^\varepsilon = \mathcal{O}(\varepsilon^2). \end{cases}$$

Moreover, the normal velocity V_{ij}^ε at the interface Γ_{ij}^ε satisfies (formally) :

$$\frac{1}{\nu_{ij}} V_{ij}^\varepsilon = \sigma_{ij} \frac{c_W c_M}{(c_N)^2} \Delta_\Gamma H_{ij} + \mathcal{O}(\varepsilon).$$

To prove these propositions, we first recall the tools necessary for our derivations following the notations of [5, 96, 72]. We follow closely the presentation of these tools of the previous chapter. We first proceed to the asymptotic expansion for the **M-CH** model. The proof is shown in dimension 2 only for the sake of simplicity of notations and readability, but it can be readily extended to higher dimensions. We end up with the **NMN-CH** model, which we have to rewrite to avoid indeterminate forms. For the most part, the calculations remain the same as for the biphasic case presented in [73].

8.2.1 Formal asymptotic analysis toolbox

In this multiphase context, we study the behavior of the system in two regions : near the interface $\Gamma := \Gamma_{ij}$ separating two given phases $i \neq j$, and far from it. We denote u_k the solution for an arbitrary phase k. Whether k can designate i or j in an equation will be clear from the context.

To derive the method we require that the interface $\Gamma = \Gamma_{ij}$ remains smooth enough so that there exist $\delta > 0$ and a neighborhood

$$\mathcal{N} = \mathcal{N}_{ij}^\delta(\Gamma) = \{x \in \Omega / |d(x,t)| < 3\delta\},$$

in which the signed distance function $d := d_{ij}$ to Γ is well-defined. $\mathcal{N}$ is called the *inner region* near the interface and its complement the *outer region*.

Outer variables :
Far from the interface, we consider the *outer functions* (u_k, μ_k) depending on the standard *outer variable* x. The systems remain the same, namely for **M-CH** :

$$\begin{cases} \varepsilon^2 \partial_t u_k = \nu_k\,\mathrm{div}(M(u_k)\nabla(\sigma_k\mu_k + \lambda)), \\ \mu_k = -\varepsilon^2 \Delta u_k + W'(u_k). \end{cases} \tag{8.2.1}$$

Inner variables :
Inside $\mathcal{N}$ we consider the *inner variables* (z,s) associated with the original variables (x,t) in the following way : $z = \frac{d(x,t)}{\varepsilon}$ is a variable along the normal direction to the interface Γ and $s = S(x,t)$ is associated with a parameterization $X_0(s,t)$ of Γ.We define the *inner functions* U, $\mathbf{\mu}$ depending on (z,s) as follows :

$$\begin{cases} U(z,s,t) := U\left(\dfrac{d(x,t)}{\varepsilon}, S(x,t), t\right) = u(x,t) \\ \mathbf{\mu}(z,s,t) := \mathbf{\mu}\left(\dfrac{d(x,t)}{\varepsilon}, S(x,t), t\right) = \mu(x,t) \end{cases}$$

For general geometric properties of the interface we refer the reader to the previous chapter. In particular, the properties of the d and S functions can be generalized to the multiphase case in the same way as in the two-phase case.

To express the connection between the derivatives of $U_k, \boldsymbol{\mu}_k$ and u_k, μ_k, we come back to the definition of the inner functions :

$$u_k(x,t) = U_k\left(\frac{d(x,t)}{\varepsilon}, S(x,t), t\right).$$

Successive derivations with respect to x give the following equations

$$\begin{cases} \nabla u_k = \nabla d \frac{1}{\varepsilon}\partial_z U_k + \nabla S \partial_s U_k, \\ \Delta u_k = \Delta d \frac{1}{\varepsilon}\partial_z U_k + \frac{1}{\varepsilon^2}\partial_{zz} U_k + \Delta S \partial_s U_k + |\nabla S|^2 \partial_{ss} U_k, \\ \operatorname{div}\left(M(u_k)\nabla(N(u_k)\mu_k)\right) = \frac{1}{\varepsilon^2}\left(\partial_z M_k \partial_z (N_k \boldsymbol{\mu}_k)\right) + \frac{M_k}{\varepsilon}\Delta d \partial_z (N_k \boldsymbol{\mu}_k), \\ \qquad\qquad + |\nabla S|^2 \partial_s\left(M_k \partial_s (N_k \boldsymbol{\mu}_k)\right) + \Delta S M_k \partial_s (N_k \boldsymbol{\mu}_k). \end{cases}$$

The *inner system* of the **M-CH** model near the interface Γ finally reads as :

$$\begin{cases} \frac{\varepsilon^2}{\nu_k}\left(\partial_t U_k + \partial_t S \partial_s U_k\right) - \frac{\varepsilon}{\nu_k} V_{ij} \partial_z U_k = \frac{1}{\varepsilon^2}\partial_z\left(M(U_k)\partial_z\left(\sigma_k \boldsymbol{\mu}_k + \Lambda\right)\right), \\ \qquad\qquad + \frac{M(U_k)}{\varepsilon}\partial_z\left(\sigma_k \boldsymbol{\mu}_k + \Lambda\right)\Delta d_{ij} + T_1(s), \\ \boldsymbol{\mu}_k = W'(U_k) - \partial_{zz} U_k + \varepsilon \Delta d_{ij} \partial_z U_k + \varepsilon^2 T_2(s), \\ \Delta d_{ij} = \frac{H_{ij}}{1+\varepsilon z H_{ij}} = H_{ij} - \varepsilon z H_{ij}^2 + \mathcal{O}(\varepsilon^2), \\ T_1(s) = \frac{\partial_s(M(U_k)\partial_s(\sigma_k \boldsymbol{\mu}_k + \Lambda))}{(1+\varepsilon z H_{ij})^2} - \frac{M(U_k)\varepsilon z \partial_s H_{ij}}{(1+\varepsilon z H_{ij})^3}\partial_s(\sigma_k \boldsymbol{\mu}_k + \Lambda), \\ T_2(s) = \frac{1}{(1+\varepsilon z H)^2}\partial_{ss} U_k - \frac{\varepsilon z \partial_s H_{ij}}{(1+\varepsilon z H_{ij})^3}\partial_s U_k. \end{cases}$$

Note that the terms in T_1 and T_2 are high-order tangential terms that play a role only at the fourth order in the asymptotic expansion.

Independence in z of the normal velocity V_{ij} :
The normal velocity $V_{ij}(s,t)$ of the interface is defined by :

$$V_{ij}(s,t) = \partial_t X_0(s,t) \cdot n(s,t).$$

Following the same reasoning as in the two-phase case, the normal velocity is independent of the variable z and one gets

$$V_{ij}(X_0(s,t) + \varepsilon z n, t) := -\partial_t d(X_0(s,t) + \varepsilon z n, t) = V_{ij}(s,t).$$

This property of independence is crucial to be able to extract the velocity from integrals in z in the following derivations.

Taylor expansions :
We assume the following Taylor expansions for our functions :

$$\begin{aligned} u_k(x,t) &= u_k^{(0)}(x,t) + \varepsilon u_k^{(1)}(x,t) + \varepsilon^2 u_k^{(2)}(x,t) + \cdots \\ U_k(z,s,t) &= U_k^{(0)}(z,s,t) + \varepsilon U_k^{(1)}(z,s,t) + \varepsilon^2 U_k^{(2)}(z,s,t) + \cdots \\ \mu_k(x,t) &= \mu_k^{(0)}(x,t) + \varepsilon \mu_k^{(1)}(x,t) + \varepsilon^2 \mu_k^{(2)}(x,t) + \cdots \\ \boldsymbol{\mu}_k(z,s,t) &= \boldsymbol{\mu}_k^{(0)}(z,s,t) + \varepsilon \boldsymbol{\mu}_k^{(1)}(z,s,t) + \varepsilon^2 \boldsymbol{\mu}_k^{(2)}(z,s,t) + \cdots \\ \lambda_k(x,t) &= \varepsilon \lambda_k^{(1)}(x,t) + \varepsilon^2 \lambda_k^{(2)}(x,t) + \cdots \\ \Lambda_k(z,s,t) &= \varepsilon \Lambda_k^{(1)}(z,s,t) + \varepsilon^2 \Lambda_k^{(2)}(z,s,t) + \cdots \end{aligned}$$

Since the numbering of the phase is present as a subscript, we indicate the order in the Taylor expansion as a superscript in brackets. We can then compose these expansions with a regular function F :

$$\begin{aligned} F(U_k) = F(U_k^{(0)}) + \varepsilon F'(U_k^{(0)}) U_k^{(1)} + \varepsilon^2 \left[F'(U_k^{(0)}) U_k^{(2)} + \frac{F''(U_k^{(0)})}{2}(U_k^{(1)})^2\right] \\ + \varepsilon^3 \left[F'(U_k^{(0)}) U_k^{(3)} + F''(U_k^{(0)}) U_k^{(1)} U_k^{(2)} + \frac{F'''(U_k^{(0)})}{6}(U_k^{(1)})^3\right] + \cdots \end{aligned}$$

To simplify the notations within the asymptotic expansion, we adopt the following notations for $M(u_k)$

$$M(u_k) = m_k^{(0)} + \varepsilon m_k^{(1)} + \varepsilon^2 m_k^{(2)} + \cdots,$$

where

$$\begin{cases} m_k^{(0)} = M(u_k^{(0)}), \\ m_k^{(1)} = M'(u_k^{(0)}) u_k^{(1)}, \\ m_k^{(2)} = M'(u_k^{(0)}) u_k^{(2)} + \dfrac{M''(u_k^{(0)})}{2} (u_k^{(1)})^2. \end{cases}$$

We adopt the same convention for any generic *outer function* $F(u_k)$ or *inner function* $F(U_k)$:

$$F(u_k) = f_k^{(0)} + \varepsilon f_k^{(1)} + \varepsilon^2 f_k^{(2)} + \varepsilon^3 f_k^{(3)} + \cdots$$
$$F(U_k) = F_k^{(0)} + \varepsilon F_k^{(1)} + \varepsilon^2 F_k^{(2)} + \varepsilon^3 F_k^{(3)} + \cdots$$

Flux matching condition between inner and outer equations :
Instead of using the matching conditions directly between the first equations of the inner and outer systems, it is more convenient to do the matching for the flux

$$j_k = M(u_k) \nabla(\sigma_k \mu_k + \lambda).$$

j_k has the following Taylor expansion for the **M-CH** model

$$\begin{aligned} j_k = & \left[m_k^{(0)} \nabla(\sigma_k \mu_k^{(0)} + \lambda^{(0)}) \right] + \varepsilon \left[m_k^{(1)} \nabla(\sigma_k \mu_k^{(0)} + \lambda^{(0)}) + m_k^{(0)} \nabla(\sigma_k \mu_k^{(1)}) \lambda^{(1)}) \right] \\ & + \varepsilon^2 \left[m_k^{(2)} \nabla(\sigma_k \mu_k^{(0)} + \lambda^{(0)}) + m_k^{(1)} \nabla(\sigma_k \mu_k^{(1)} + \lambda^{(1)}) + m_k^{(0)} \nabla(\sigma_k \mu_k^{(2)} + \lambda^{(2)}) \right] + \mathcal{O}(\varepsilon^3). \end{aligned} \tag{8.2.2}$$

In inner coordinates, we only need to express the normal part

$$J_{k,n} := J_k \cdot n = \frac{M(U_k)}{\varepsilon} \partial_z(\sigma_k \boldsymbol{\mu}_k + \Lambda),$$

as the tangential terms are of higher order. The normal part expands as

$$\begin{aligned} J_{k,n} = & \ \frac{1}{\varepsilon} \left[M_k^{(0)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(0)} + \Lambda^{(0)}) \right], \\ & + \ \left[M_k^{(1)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(0)} + \Lambda^{(0)}) + M_k^{(0)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}) \right], \\ & + \varepsilon \left[M_k^{(2)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(0)} + \Lambda^{(0)}) + M_k^{(1)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}) + M_k^{(0)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}) \right], \\ & + \varepsilon^2 \left[M_k^{(3)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(0)} + \Lambda^{(0)}) + M_k^{(2)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}), \right. \\ & \qquad \left. + M_k^{(1)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}) + M_k^{(0)} \partial_z(\sigma_k \boldsymbol{\mu}_k^{(3)} + \Lambda^{(3)}) \right] + \mathcal{O}(\varepsilon^3). \end{aligned} \tag{8.2.3}$$

The flux matching conditions allow to match the limit as $z \to \pm\infty$ of terms of (8.2.2) with the corresponding order terms of (8.2.3).

We can now investigate order by order the behavior of the **M-CH** model. We have to study up to the fourth order term where the leading order of the velocity will appear in the first equation of (8.2.1). After that, we adapt the argument to the **NMN-CH** model, where a reformulation of the problem will be necessary to avoid indeterminate forms in the asymptotic expansion.

8.2.2 Formal matched asymptotic expansion for the multiphasic M-CH model

We first establish Proposition 8.2.1 regarding the properties of the **M-CH** model. We recall that we study the behavior of the different terms of the system near the interface Γ_{ij} separating phases i and j. We assume the following matching conditions for the two phases :

$$\lim_{z \to +\infty} U_i^{(0)} = 0, \quad \lim_{z \to -\infty} U_i^{(0)} = 1,$$
$$\lim_{z \to +\infty} U_j^{(0)} = 1, \quad \lim_{z \to -\infty} U_j^{(0)} = 0.$$

For the other phases, we require the following matching conditions

$$\lim_{z \to \pm\infty} U_k^{(0)} = 0, \quad \lim_{z \to \pm\infty} U_k^{(1)} = 0.$$

First order :
At order $(\mathcal{O}(\varepsilon^{-2}), \mathcal{O}(1))$ the inner system reads as

$$\begin{cases} 0 = \partial_z \left(M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(0)}) \right), \\ \boldsymbol{\mu}_k^{(0)} = W'(U_k^{(0)}) - \partial_{zz} U_k^{(0)}. \end{cases}$$

From the first equation of the system, we deduce that $M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(0)})$ is constant. The matching conditions for the outer flux (8.2.2) and the inner flux (8.2.3) at order ε^{-1} impose this constant to be zero. Then there is a constant $A_k^{(0)}$ such that

$$\sigma_k \boldsymbol{\mu}_k^{(0)} = A_k^{(0)}.$$

Collecting all this information, we obtain that :

$$\partial_{zz} U_k^{(0)} - W'(U_k^{(0)}) = \frac{A_k^{(0)}}{\sigma_k} \quad \forall k \in \{1, \dots, L\}.$$

We complement with the initial conditions

$$\begin{cases} U_i^{(0)} = U_j^{(0)} = \dfrac{1}{2} \\ U_k^{(0)} = 0 \quad \forall k \in \{1, \dots, L\} \setminus \{i, j\}, \end{cases}$$

Finally, we conclude that

$$\begin{cases} U_i^{(0)} = q(z), \\ U_j^{(0)} = q(-z) = 1 - q(z), \\ U_k^{(0)} = 0 \quad \forall k \in \{1, \dots, L\} \setminus \{i, j\}, \\ \boldsymbol{\mu}_k^{(0)} = 0 \quad \forall k \in \{1, \dots, L\}, \end{cases}$$

where q is the optimal phase field profile.

Second order :
At order $(\mathcal{O}(\varepsilon^{-1}), \mathcal{O}(\varepsilon))$ the outer system reads as

$$\begin{cases} 0 = \partial_z \left(M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}) \right), \\ \boldsymbol{\mu}_k^{(1)} = W''(U_k^{(0)}) U_k^{(1)} - \partial_{zz} U_k^{(1)} - H_{ij} \partial_z U_k^{(0)}. \end{cases} \tag{8.2.4}$$

It follows that there exists a function $B_k^{(1)}$ constant in z such that

$$M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}) = B_k^{(1)}.$$

By the matching condition between the outer flux (8.2.2) and the inner flux (8.2.3) at order 1, it holds that

$$\lim_{z \to \pm\infty} M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)}) = 0.$$

We deduce that $B_k^{(1)} = 0$ and that there exists a function $A_k^{(1)}$ constant in z such that

$$\sigma_k \boldsymbol{\mu}_k^{(1)} + \Lambda^{(1)} = A_k^{(1)}.$$

Subtracting the case $k = i$ from the case $k = j$ gives

$$\sigma_j \boldsymbol{\mu}_j^{(1)} - \sigma_i \boldsymbol{\mu}_i^{(1)} = A_j^{(1)} - A_i^{(1)}.$$

The term $A_{ij}^{(1)} := A_j^{(1)} - A_i^{(1)}$ can be determined using the second equation of (8.2.4). Indeed, recall that

$$\begin{cases} \sigma_i \boldsymbol{\mu}_i^{(1)} = \sigma_i W''(U_i^{(0)}) U_i^{(1)} - \sigma_i \partial_{zz} U_i^{(1)} - \sigma_i H_{ij} q', \\ \sigma_j \boldsymbol{\mu}_j^{(1)} = \sigma_j W''(U_k^{(0)}) U_j^{(1)} - \sigma_j \partial_{zz} U_j^{(1)} + \sigma_j H_{ij} q'. \end{cases}$$

We multiply both equations by q' and integrate the difference. We can eliminate the terms in $U^{(1)}$ through integration by parts :

$$\begin{aligned}\int_{\mathbb{R}} \partial_z(W'(U_i^{(0)}))(U_i^{(1)} - \partial_{zz}U_i^{(1)}\partial_z U_i^{(0)})dz &= \left[W'(U_i^{(0)})U_i^{(1)} - \partial_z U_i^{(1)}\partial_z U_i^{(0)}\right]_{-\infty}^{+\infty}, \\ &\quad - \int_{\mathbb{R}} \partial_z U_i^{(1)} \left(\underbrace{W'(U_i^{(0)}) - \partial_{zz}U_i^{(0)}}_{=0} \right) dz, \\ &= 0.\end{aligned}$$

It follows that

$$A_{ij}^{(1)} = -\int_{\mathbb{R}} (\sigma_j \mu_j^{(1)} - \sigma_i \mu_i^{(1)}) q'\, dz = -(\sigma_j + \sigma_i) H_{ij} \int_{\mathbb{R}} (q')^2\, dz = -\sigma_{ij} c_W H_{ij}. \tag{8.2.5}$$

On the other hand, summing the second equation of system (8.2.4) for the phases i and j gives

$$\mu_i^{(1)} + \mu_j^{(1)} = W''(q)\left[U_i^{(1)} + U_j^{(1)}\right] - \partial_{zz}\left[U_i^{(1)} + U_j^{(1)}\right].$$

Multiplying by q' and integrating by parts gives :

$$\int_{\mathbb{R}} \left(\mu_i^{(1)} + \mu_j^{(1)}\right) q' dz = 0.$$

Thus there exists a profile ζ and a tangential function c such that

$$\mu_i^{(1)} + \mu_j^{(1)} = c(s)\zeta(z).$$

Combining this with equation (8.2.5), we obtain

$$\begin{cases} \mu_i^{(1)} = -c_W H_{ij} + \zeta(z)c(s)\dfrac{\sigma_i}{\sigma_{ij}}, \\ \mu_j^{(1)} = c_W H_{ij} + \zeta(z)c(s)\dfrac{\sigma_j}{\sigma_{ij}}. \end{cases}$$

and then,

$$\begin{cases} W''(U_i^{(0)})U_i^{(1)} - \partial_{zz}U_i^{(1)} = H_{ij}(c_W + q') + \zeta(z)c(s)\frac{\sigma_j}{\sigma_{ij}}, \\ W''(U_j^{(0)})U_j^{(1)} - \partial_{zz}U_j^{(1)} = -H_{ij}(c_W + q') + \zeta(z)c(s)\frac{\sigma_i}{\sigma_{ij}}. \end{cases}$$

which leads to

$$\begin{cases} U_i^{(1)}(z,s) = H_{ij}\eta(z) + c(s)\frac{\sigma_j}{\sigma_{ij}}\omega(z), \\ U_j^{(1)}(z,s) = -H_{ij}\eta(z) + c(s)\frac{\sigma_i}{\sigma_{ij}}\omega(z). \end{cases}$$

Here η and ω are two profiles defined as the solutions to $W'(q)y - y'' = q' + c_W$ and $W'(q)y - y'' = \xi$, respectively, with appropriate initial conditions.

In particular, if $H_{ij} \neq 0$, then $U_i^{(1)}$ and $U_j^{(1)}$ cannot vanish both together which explains why the leading error order term for the solution of the system is ε only. This yields the important conclusion that the **M-CH** model is *always of order* 1 when the mean curvature is non zero. It justifies the interest of the **NMN-CH** model which is of second order.

Third order :
At order $(\mathcal{O}(1), \mathcal{O}(\varepsilon^2))$ the inner system reads as

$$\begin{cases} 0 = \partial_z\left(M_k^{(0)}\partial_z(\sigma_k\mu_k^{(2)} + \Lambda^{(2)})\right), \\ \mu_k^{(2)} = \dfrac{W'''(U_k^{(0)})}{2}(U_k^{(1)})^2 + W''(U_k^{(0)})U_k^{(2)} - \partial_{zz}U_k^{(2)} + H_{ij}\partial_z U_k^{(1)} - zH_{ij}^{(2)}\partial_z U_k^{(0)}. \end{cases} \tag{8.2.6}$$

In the first equation, we used the results from the first two orders and left out the term that vanishes. From the first equality of (8.2.6), we find that :

$$M_k^{(0)}\partial_z(\sigma_k\mu_k^{(2)} + \Lambda^{(2)}) = B_k^{(2)}.$$

The matching conditions between the flux (8.2.2) and (8.2.3) at order ε yields (by removing all the null terms) :

$$B_k^{(2)} = \lim_{z\to\pm\infty} M_k^{(0)}\partial_z(\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}) = 0.$$

This means that the term $\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}$ is constant in z and will not intervene in the flux term of order ε^2.

Fourth order :
Collecting the previous results, the first equation of the inner system at order ε for the phase i and j simplifies to

$$\begin{cases} -\dfrac{1}{\nu_i} V_{ij} q' = \partial_z \left[M_i^{(0)} \partial_z (\sigma_i \boldsymbol{\mu}_i^{(3)} + \Lambda^{(3)}) \right] + \partial_s \left[M_i^{(0)} \partial_s (\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)}) \right], \\ \dfrac{1}{\nu_j} V_{ij} q' = \partial_z \left[M_j^{(0)} \partial_z (\sigma_j \boldsymbol{\mu}_j^{(3)} + \Lambda^{(3)}) \right] + \partial_s \left[M_j^{(0)} \partial_s (\sigma_j \boldsymbol{\mu}_j^{(1)} + \Lambda^{(1)}) \right], \end{cases}$$

We subtract the two equations, and integrate. We divide the computation in three :

— The left hand side gives :

$$\left(\frac{1}{\nu_i} + \frac{1}{\nu_j} \right) V_{ij} = \frac{1}{\nu_{ij}} V_{ij}.$$

— Collecting the result from the previous paragraphs, we find the following matching between the outer flux (8.2.2) and the inner flux (8.2.3) at order ε^2 :

$$\lim_{z\to\pm\infty} M_i^{(0)} \partial_z (\sigma_i \boldsymbol{\mu}_i^{(3)} + \Lambda^{(3)}) = m_i^{(1)} \nabla (\sigma_i \mu_i^{(1)} + \lambda^{(1)}).$$

Because $M'(0) = M'(1) = 0$, the limit term is zero and then

$$\int_{\mathbb{R}} \partial_z \left(M_i^{(0)} \partial_z (\sigma_i \boldsymbol{\mu}_i^{(3)} + \Lambda^{(3)}) \right) dz = 0.$$

The corresponding term for the j-th phase is treated similarly.

— Using (8.2.5), the second term of the right hand side is (noting that $M_i^{(0)} = M_j^{(0)}$) :

$$\int_{\mathbb{R}} M_i^{(0)} \partial_{ss} (\sigma_i \boldsymbol{\mu}_i^{(1)} - \sigma_j \boldsymbol{\mu}_j^{(1)}) dz = \sigma_{ij} \left(\int_{\mathbb{R}} M(q(z)) dz \right) c_W \partial_{ss} H_{ij}.$$

Finally, we obtain that

$$\frac{1}{\nu_{ij}} V_{ij} = \sigma_{ij} c_W c_M \partial_{ss} H_{ij}.$$

8.2.3 Formal matched asymptotic expansion for the multiphase NMN-CH model

We now give a proof of Proposition 8.2.2 concerning the properties of **NMN-CH**. The following matching conditions for the phase i and j are assumed :

$$\lim_{z\to+\infty} U_i^{(0)} = 0, \quad \lim_{z\to-\infty} U_i^{(0)} = 1,$$
$$\lim_{z\to+\infty} U_i^{(0)} = 1, \quad \lim_{z\to-\infty} U_i^{(0)} = 0,$$

and for the other phases

$$\lim_{z\to\pm\infty} U_k^{(0)} = 0, \quad \lim_{z\to\pm\infty} U_k^{(1)} = 0.$$

Reformulation of the model :
It is more convenient to rewrite the **NMN-CH** model by transferring $N(u_k)$ to the left hand side of the system, which yields the **NMN-CH** reformulation I model we already mentioned :

$$\begin{cases} \varepsilon^2 g(u_k) \partial_t u_k = \nu_k \operatorname{div} \left(M(u_k) \nabla (\sigma_k \mu_k + \lambda) \right), \\ g(u_k) \mu_k = W'(u_k) - \varepsilon^2 \Delta u_k. \end{cases} \tag{8.2.7}$$

where $g(u_k) = \sqrt{M(u_k)} = \frac{1}{N(u_k)}$ when $N(u_k)$ is well-defined, and, as before, λ is the Lagrangian multiplier which encodes the partition constraint $\sum_{u_k} = 0$. As already said, the advantage of such a formulation is that $g(u_k)$ is always well defined even if $u_k = 0$, which is not the case for $N(u_k)$. Note also that the definition of μ_k has been changed but we keep the same notation for simplicity.

Remark that similar calculations as those shown below can be done for the **NMN-CH** reformulation II model which is used for numerical approximation, and the same conclusions of Proposition 8.2.2 hold.

Actually, using the additional term $\gamma\varepsilon^2$ in the definition of M does not change the asymptotic results for at least the first four orders of interests. Indeed, this term appears to be associated with μ_0 (see below) which is zero, and μ_1 whose derivative in z vanishes.

The inner system for **NMN-CH** reformulation I now reads (for simplicity, we drop the expression "reformulation I" in the calculations below) :

$$\begin{cases} \dfrac{\varepsilon^2 G_k}{\nu_k}\partial_t U_k + \dfrac{\varepsilon^2 G_k}{\nu_k}\partial_t S\partial_s U_k - \dfrac{\varepsilon G_k}{\nu_k}V_{ij}\partial_z U_k = \dfrac{1}{\varepsilon^2}\partial_z\left(M_k\partial_z(\sigma_k\mu_k+\Lambda)\right) \\ \qquad\qquad + \dfrac{M_k}{\varepsilon}\Delta d_{ij}\partial_z(\sigma_k\mu_k+\Lambda) + T_1(s), \\ G_k\mu_k = W'(U_k) - \partial_{zz}U_k - \varepsilon\Delta d_{ij}\partial_z U_k + \varepsilon^2 T_2(s), \\ \Delta d_{ij} = \dfrac{H_{ij}}{1+\varepsilon z H_{ij}} = H_{ij} - \varepsilon z H_{ij}^2 + \mathcal{O}(\varepsilon^2), \\ T_1(s) = \dfrac{1}{(1+\varepsilon z H_{ij})^2}\partial_s(M_k\partial_s(\sigma_k\mu_k+\Lambda)) - \dfrac{\varepsilon z M_k\partial_s H_{ij}}{(1+\varepsilon z H_{ij})^3}\partial_s(\sigma_k\mu_k+\Lambda), \\ T_2(s) = \dfrac{1}{(1+\varepsilon z H_{ij})^2}\partial_{ss}U_k - \dfrac{\varepsilon z\partial_s H_{ij}}{(1+\varepsilon z H_{ij})^3}\partial_s U_k. \end{cases}$$

Because the $N(u_k)$ are now on the left hand side of the system in the form of G_k, the flux term $j = M(u_k)\nabla(\sigma_k\mu_k+\lambda_k)$ is the same as the one for **M-CH**. The flux matching condition is then also equal to the one given by (8.2.2) and (8.2.3).

First order :
At order $(\mathcal{O}(\varepsilon^{-2}),\mathcal{O}(1))$ the inner system reads :

$$\begin{cases} 0 = \partial_z\left(M_k^{(0)}\partial_z(\sigma_k\mu_k^{(0)})\right), \\ G_k^{(0)}\mu_k^{(0)} = W'(U_k^{(0)}) - \partial_{zz}U_k^{(0)}. \end{cases}$$

From the first equation of the system, we deduce that $M_k^{(0)}\partial_z(\sigma_k\mu_k^{(0)})$ is constant. The matching conditions on the outer (8.2.2) and inner fluxes (8.2.3) at order ε^{-1} impose this constant to be zero. Then there exists a constant $A_k^{(0)}$ in z such that

$$\sigma_k\mu_k^{(0)} = A_k^{(0)}.$$

Collecting all this information, we have

$$\partial_{zz}U_k^{(0)} - W'(U_k^{(0)}) = \frac{G\left(U_k^{(0)}\right)A_k^{(0)}}{\sigma_k}, \quad \forall k\in\{1,\dots,L\}.$$

Then, using the matching conditions and the initial conditions $U_i^{(0)} = U_j^{(0)} = \frac{1}{2}$ leads to

$$\begin{cases} U_i^{(0)} = q(z), \\ \mu_i^{(0)} = 0, \\ U_j^{(0)} = q(-z) = 1-q(z), \\ \mu_j^{(0)} = 0, \\ U_k^{(0)} = 0 \quad \forall k\in\{1,\dots,L\}\setminus\{i,j\}. \end{cases}$$

Notice that $\mu_k^{(0)}$ is a constant in z that can be nonzero.

Second order :
At order $(\mathcal{O}(\varepsilon^{-1}),\mathcal{O}(\varepsilon))$ the inner system reads

$$\begin{cases} 0 = \partial_z\left(M_k^{(0)}\partial_z(\sigma_k\mu_k^{(1)}+\Lambda^{(1)})\right), \\ G_k^{(0)}\mu_k^{(1)} + G_k^{(1)}\mu_k^{(0)} = W''(U_k^{(0)})U_k^{(1)} - \partial_{zz}U_k^{(1)} - H_{ij}\partial_z U_k^{(0)}. \end{cases} \tag{8.2.8}$$

For $k\neq i,j$, using the fact that $U_k^{(0)} = 0$, the second equation can be rewritten as

$$0 = \left(1-\mu_k^{(0)}\right)U_k^{(1)} - \partial_{zz}U_k^{(1)}.$$

As the matching conditions show that $\lim_{z\pm\infty} U_k^{(1)} = 0$, it follows that $U_k^{(1)} = 0$.

Now turning to the i-th phase (resp j-th), there exists a function $B_i^{(1)}$ constant in z such that

$$M_i^{(0)}\partial_z(\sigma_i \mu_i^{(1)} + \Lambda^{(1)}) = B_i^{(1)}.$$

From the matching condition between outer (8.2.2) and inner flux (8.2.3) at order 1, we deduce that

$$\lim_{z\to\pm\infty} M_i^{(0)}\partial_z(\sigma_i \mu_i^{(1)} + \Lambda^{(1)}) = 0,$$

and $B_i^{(1)} = 0$ (resp $B_j^{(1)} = 0$). Then there exist functions $A_i^{(1)}, A_j^{(1)}$ constant in z such that

$$\begin{cases} \sigma_i \mu_i^{(1)} + \Lambda^{(1)} = A_i^{(1)}, \\ \sigma_j \mu_j^{(1)} + \Lambda^{(1)} = A_j^{(1)}. \end{cases}$$

Subtracting the i-th term to the j-th term leads to

$$\sigma_j \mu_j^{(1)} - \sigma_i \mu_i^{(1)} = A_j^{(1)} - A_i^{(1)} := A_{ij}^{(1)}.$$

Moreover, recall that $U_k^{(1)} = 0$ for $k \neq i, j$, which implies that $U_i^{(1)} = -U_j^{(1)}$ as $\sum_{k=1}^N U_k^{(1)} = 0$. Using the symmetry properties $g(1-s) = g(s)$ and $W''(1-s) = W''(s)$, it follows that :

$$\begin{aligned} g(U_j^{(0)})\mu_j^{(1)} &= W''(U_j^{(0)})U_j^{(1)} - \partial_{zz}U_j^{(1)} - H_{ij}\partial_z U_j^{(0)}, \\ &= -W''(U_i^{(0)})U_i^{(1)} + \partial_{zz}U_i^{(1)} + H_{ij}\partial_z U_i^{(0)}, \\ &= -g(U_i^{(0)})\mu_i^{(1)}, \\ &= -g(U_j^{(0)})\mu_i^{(1)}. \end{aligned}$$

thus $g(U_j^{(0)})\left(\mu_i^{(1)} + \mu_j^{(1)}\right) = 0$. Finally, as $g(q) \neq 0$, $\mu_i^{(1)}$ and $\mu_j^{(1)}$ are necessarily constant in z and

$$\mu_j^{(1)} = -\mu_i^{(1)}.$$

It shows that we can express $A_{ij}^{(1)}$ as

$$A_{ij}^{(1)} = (\sigma_j + \sigma_i)\,\mu_j^{(1)} = -\sigma_{ij}\mu_i^{(1)}.$$

Now, multiplying the second equation of (8.2.8) for phase i

$$g(U_i^{(0)})\mu_i^{(1)} = W''(U_i^{(0)})U_i^{(1)} - \partial_{zz}U_i^{(1)} - H_{ij}q',$$

by the profile q' and integrating over $\mathbb{R}$ shows that

$$\mu_i^{(1)} = -\frac{c_W}{c_N}H_{ij} \quad \text{and} \quad A_{ij}^{(1)} = \frac{c_W}{c_N}\sigma_{ij}H_{ij}. \tag{8.2.9}$$

Indeed, on the one hand we have

$$\begin{aligned} \int_{\mathbb{R}} (\partial_z(W'(U_i^{(0)}))U_i^{(1)} - \partial_{zz}U_i^{(1)}\partial_z U_i^{(0)})dz &= \left[W'(U_i^{(0)})U_i^{(1)} - \partial_z U_i^{(1)}\partial_z U_i^{(0)}\right]_{-\infty}^{+\infty}, \\ &\quad - \int_{\mathbb{R}} \partial_z U_i^{(1)} \left(\underbrace{W'(U_i^{(0)}) - \partial_{zz}U_i^{(0)}}_{=0}\right) dz, \\ &= 0, \end{aligned}$$

and on the other hand

$$\int_{\mathbb{R}} g(q)\mu_i^{(1)}q'dz = \mu_i^{(1)}\int_{\mathbb{R}} \frac{q'}{N(q)}dz = c_N\mu_i^{(1)}, \quad \text{and} \int_{\mathbb{R}} H_{ij}(q')^2 dz = c_W H_{ij}.$$

Finally, it shows that

$$\partial_{zz}U_i^{(1)} - W''(q)U_i^{(1)} = -H_{i,j}(q' - \frac{c_W}{c_N}g(q)).$$

Now, recall that the choice $M(s) = \sqrt{2W(s)}$ and $N(s) = 1/\sqrt{M(s)} = 1/\sqrt{2W(s)}$ for the mobilities implies that

$$g(q) = 1/N(q) = \sqrt{2W(q)} = -q' \text{ and } c_N = -c_W.$$

This is the key point to understand why in this case the term $U_i^{(1)}$ is null as a solution of

$$\partial_{zz} U_i^{(1)} - W''(q) U_i^{(1)} = 0.$$

The same argument gives $U_j^{(1)} = 0$. In summary, we have $U_k^{(1)} = U_i^{(1)} = U_j^{(1)} = 0$. It means that the leading error order term in the solutions U_i and U_j is of magnitude ε^2 while the other phases are absent.
Third order :
At order $(\mathcal{O}(1), \mathcal{O}(\varepsilon^2))$, using both previous orders, the inner system simplifies to

$$\begin{cases} 0 = \partial_z \left(M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}) \right), \\ G_k^{(0)} \boldsymbol{\mu}_k^{(2)} = \dfrac{W'''(U_k^{(0)})}{2} (U_k^{(1)})^2 + W''(U_k^{(0)}) U_k^{(2)} - \partial_{zz} U_k^{(2)} + H_{ij} \partial_z U_k^{(1)} - z H_{ij}^2 \partial_z U_k^{(2)}. \end{cases}$$

From the first equality, we find that

$$M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^2 + \Lambda^2) = B_2.$$

The matching conditions at order ε for the fluxes given by (8.2.2) and (8.2.3) yield also (by removing all the null terms) :

$$B_2 = \lim_{z \to \pm\infty} M_k^{(0)} \partial_z (\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}) = 0.$$

Therefore, $\sigma_k \boldsymbol{\mu}_k^{(2)} + \Lambda^{(2)}$ is constant in z and will not intervene in the flux term of order ε^2.

Remark 8.2.3. *It is possible to show that $U_i^{(2)}$ and $U_j^{(2)}$ are of the form*

$$U_j^{(2)} = -U_i^{(2)} = H_{ij}^2 \zeta(z),$$

where ζ is the profile defined by

$$\begin{cases} y''(z) - W''(q) y(z) = z q' \\ y(0) = 0 \end{cases}$$

and which decreases to zero at infinity.

Fourth order :
Eliminating all the vanishing terms, the first equations for the i-th and j-th phase of the inner system read

$$\begin{aligned} -\frac{1}{\nu_i} V_{ij} g(U_i^{(0)}) \partial_z U_i^{(0)} &= \partial_z \left[M_i^{(0)} \partial_z \left(\sigma_i \boldsymbol{\mu}_i^{(3)} + \Lambda^{(3)} \right) \right] dz + \partial_s \left[M(U_i^{(0)}) \partial_s \left(\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)} \right) \right] dz, \\ -\frac{1}{\nu_j} V_{ij} g(U_j^{(0)}) \partial_z U_j^{(0)} &= \partial_z \left[M_j^{(0)} \partial_z \left(\sigma_j \boldsymbol{\mu}_j^{(3)} + \Lambda^{(3)} \right) \right] dz + \partial_s \left[M(U_j^{(0)}) \partial_s \left(\sigma_j \boldsymbol{\mu}_j^{(1)} + \Lambda^{(1)} \right) \right] dz. \end{aligned} \tag{8.2.10}$$

Integrating over $\mathbb{R}$ yields to

$$\begin{aligned} -\frac{c_N}{\nu_i} V_{ij} &= \int_{\mathbb{R}} \partial_z \left[M(U_i^{(0)}) \partial_z \left(\sigma_i \boldsymbol{\mu}_i^{(3)} + \Lambda^{(3)} \right) \right] dz + \int_{\mathbb{R}} \partial_s \left[M(U_i^{(0)}) q' \partial_s \left(\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)} \right) \right] dz, \\ +\frac{c_N}{\nu_j} V_{ij} &= \int_{\mathbb{R}} \partial_z \left[M(U_j^{(0)}) \partial_z \left(\sigma_j \boldsymbol{\mu}_j^{(3)} + \Lambda^{(3)} \right) \right] dz + \int_{\mathbb{R}} \partial_s \left[M(U_j^{(0)}) q' \partial_s \left(\sigma_j \boldsymbol{\mu}_j^{(1)} + \Lambda^{(1)} \right) \right] dz. \end{aligned}$$

The matching conditions for the fluxes at order $\mathcal{O}(\varepsilon^2)$ show that the first integral is zero. Note that most of the terms in the fluxes have been proven to be zero in the previous orders.

On the other hand, the second integral can be expressed with the terms from the second order calculations and the properties of the profile q give that

$$\begin{aligned} \int_{\mathbb{R}} \partial_s \left[M(U_i^{(0)}) \partial_s \left(\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)} \right) \right] dz &= \partial_{ss} \left(\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)} \right) \int_{\mathbb{R}} M(q) dz, \\ &= c_M \partial_{ss} \left(\sigma_i \boldsymbol{\mu}_i^{(1)} + \Lambda^{(1)} \right). \end{aligned}$$

The same result can be obtained for the integral in j. Subtracting the first equation of (8.2.10) to the second, we get

$$\frac{1}{\nu_{ij}} c_N V_{ij} = \left(\frac{1}{\nu_i} + \frac{1}{\nu_j} \right) c_N V_{ij} = c_M \partial_{ss} \left(-\sigma_i^{(1)} \boldsymbol{\mu}_i^{(1)} - \Lambda^{(1)} + \sigma_j \boldsymbol{\mu}_j^{(1)} + \Lambda^{(1)} \right) = c_M \partial_{ss} A_{ij}^{(1)}.$$

Using (8.2.9), it follows that

$$\frac{1}{\nu_{ij}} V_{ij} = \frac{c_W c_M}{(c_N)^2} \sigma_{ij} \partial_{ss} H_{ij},$$

which concludes the proof of Proposition 8.2.2.

8.3 Numerical approximation

In this section, we show how to compute efficiently numerical approximations to the solutions to phase field models **M-CH** and **NMN-CH** , and we provide various numerical illustrations of the performances and properties of both models in dimensions 2 and 3. The numerical approximation is performed with the original **M-CH** model and with the **NMN-CH** reformulation II model (see the introduction), whose definitions are recalled :

— **M-CH**

$$\begin{cases} \partial_t u_k &= \nu_k \operatorname{div}\left(M(u_k)\nabla(\sigma_k\mu_k + \lambda)\right) \\ \mu_k &= \frac{W'(u_k)}{\varepsilon^2} - \Delta u_k \\ 1 &= \sum_k u_k \end{cases}$$

where the mobility M is defined as $M(u) = \frac{1}{c_N^2} 2W(u)$. Here, the constant $|c_N| = \frac{1}{6}$ is added to get the same limit law as with our new Cahn–Hilliard model.

— **NMN-CH**-reformulation II

$$\begin{cases} \partial_t u_k &= \nu_k N(u_k) \operatorname{div}\left(M(u_k)\nabla N(u_k)(\sigma_k\mu_k + \lambda)\right) \\ \mu_k &= \frac{W'(u_k)}{\varepsilon^2} - \Delta u_k \\ 1 &= \sum_k u_k \end{cases}$$

where the mobilities M, N are defined by $M(s) = 2W(s) + \gamma\varepsilon^2$, with $\gamma > 0$, and $N(s) = \frac{1}{\sqrt{M(s)}}$. For simplicity we keep the original notations M, N although the definitions are different, and still for the sake of simplicity we drop the expression "reformulation II". We set $\gamma = 1$ for all numerical experiments presented below.

Various schemes have already been proposed in the literature, see [338, 228, 222, 223], to deal with multiphase Cahn-Hilliard type equations, especially when the number of phases is $L = 3$ [25, 62, 209, 206, 208, 106, 51] or $L = 4$ [207, 211, 347, 223].

Recall that the Cahn-Hilliard system is of fourth-order in space, which introduces severe restrictions on the time step for most classical methods due to numerical instability. To overcome these difficulties, a natural idea is to adapt the strategy of convex splitting of the Cahn–Hilliard energy which was first proposed by Eyre [158]. This technique has become very popular for it provides simple, efficient, and stable schemes to approximate various evolution problems with a gradient flow structure [98, 334, 171, 142, 309, 310, 126]. For instance, a first- and second-order splitting scheme was proposed in [21, 301, 300] to address the case of the Cahn–Hilliard equation with mobility. However, these approaches are based on finite elements and require the resolution of linear systems at each step, which can be ill-conditioned in the case of degenerate mobilities. As an alternative, we proposed recently in [73] a semi-implicit Fourier spectral method in the spirit of [92, 65, 71, 75, 67]. The idea is to exploit the variational structure of the mobility by using an additionally convex splitting of the associated metric. It gives a very simple, efficient, and stable scheme even in the case of degenerate mobilities.

An accurate non linear multigrid method was proposed in [223] to approximate the solution to the Cahn-Hilliard equation. However, this approach requires the resolution of a $2L \times 2L$ system of equations which can be problematic when L is large. Based on the first-order convex splitting method, Lee and al [222] developed a practically unconditionally gradient-stable conservative nonlinear numerical scheme for converting the L-phase Cahn–Hilliard system into a system of L Cahn–Hilliard equations. This reduces significantly the computational cost. More recently, Yang and Kim [338] proposed an unconditionally stable with second-order accuracy based on the Crank-Nicolson scheme and adopted the idea of stabilized method [348].

In this section, we propose to adapt to multiphase the approach proposed in the two-phase case. The novelty is to split the treatment of the Lagrange multiplier via the splitting of the metric so that **M-CH** and **NMN-CH** can be solved in a decoupled way. This means that we only need to solve L biphasic Cahn–Hilliard equations at each iteration, as in [222].

In the following, we extend to the multiphase case by using a semi-implicit treatment of the Lagrange multiplier which is explicitly given in Fourier space. For each model, a **Matlab** script is provided to give an example of implementation. Next, we provide a numerical comparison of phase field models in space dimension 2. In addition, some illustrations are provided to show the influence of mobilities and surface tensions using the **NMN-CH** model. These illustrations show also that our models can handle Cahn–Hilliard problems in complex domain without imposing any boundary condition or additional surface energy, but

rather by simply imposing a null mobility at the appropriate interfaces. Then we conclude the section with an application to the wetting problem where we derive a simplified model using the liquid phase only.

8.3.1 Numerical scheme for the M-CH model

We now introduce for the multiphase **M-CH** model

$$\begin{cases} \partial_t u_k &= \nu_k \operatorname{div}\left(M(u_k)\nabla(\sigma_k\mu_k + \lambda)\right), \\ \mu_k &= \frac{W'(u_k)}{\varepsilon^2} - \Delta u_k, \\ 1 &= \sum_k u_k. \end{cases}$$

a numerical scheme that is similar to the biphasic scheme proposed in the previous chapter, based on the same convex-concave splitting of the Cahn-Hilliard equation and its associated metric. This scheme reads as

$$\begin{cases} (u_k^{n+1} - u_k^n)/\delta_t &= \nu_k \left(m\Delta\left[\sigma_k\mu_k^{n+1} + \lambda^{n+1}\right] + \operatorname{div}\left[(M(u_k^n) - m)\nabla\left[\sigma_k\mu_k^n + \lambda^n\right]\right]\right), \\ \mu_k^{n+1} &= \left(-\Delta u_k^{n+1} + \frac{\alpha}{\varepsilon^2}u_k^{n+1}\right) + \left(\frac{1}{\varepsilon^2}(W'(u_k^n) - \alpha u_k^n)\right), \end{cases}$$

where the Lagrange multiplier λ^{n+1} is associated to the partition constraint $\sum_k u_k^{n+1} = 1$.

More precisely, the couple (u_k^{n+1}, μ_k^{n+1}) can be expressed as

$$\begin{cases} u_k^{n+1} &= u_k^{n+1/2} + \delta_t\nu_k m L_{M_k}\left[\Delta\lambda^{n+1}\right], \\ \mu_k^{n+1} &= \mu_k^{n+1/2} + \delta_t\nu_k m L_{M_k}\left[(-\Delta + \frac{\alpha}{\varepsilon^2})\Delta\lambda^{n+1}\right], \end{cases}$$

where

— the operator L_{M_k} is given by

$$L_{M_k} = \left(I_d + \delta_t m\sigma_k\nu_k\Delta(\Delta - \alpha/\varepsilon^2 I_d)\right)^{-1}.$$

— the couple $(u_k^{n+1/2}, \mu_k^{n+1/2})$ is defined as the solution to the decoupled system

$$\begin{cases} u_k^{n+1/2} &= L_{M_k}\left[B^1_{u_k^n,\mu_k^n} + \delta_t m\sigma_k\nu_k\Delta B^2_{u_k^n,\mu_k^n}\right], \\ \mu_k^{n+1/2} &= L_{M_k}\left[(-\Delta + \alpha/\varepsilon^2)B^1_{u_k^n,\mu_k^n} + B^2_{u_k^n,\mu_k^n}\right], \end{cases}$$

where

$$B^1_{u_k^n,\mu_k^n} = u_k^n + \delta_t\nu_k \operatorname{div}\left[(M(u_k^n) - m)\nabla\left[\sigma_k\mu_k^n + \lambda^k\right]\right],$$

and

$$B^2_{u_k^n,\mu_k^n} = \frac{1}{\varepsilon^2}(W'(u_k^n) - \alpha u_k^n).$$

In particular, λ^{n+1} satisfies the equation

$$\delta_t m\left(\sum_k \nu_k L_{M_k}\right)\Delta\lambda^{n+1} = 1 - \sum_k u_k^{n+1/2},$$

therefore,

$$\lambda^{n+1} = \frac{1}{\delta_t m}\left[\sum_k \nu_k L_{M_k}\Delta\right]^{-1}(1 - \sum_k u_k^{n+1/2}).$$

Here the operator $\left[\sum_k \nu_k L_{M_k}\Delta\right]^{-1}$ is still homogeneous and can be computed easily in Fourier space.

From the previous equations, we can implement the schemes within the **Matlab** framework almost immediately, see the 54-lines **Matlab** script of Table 8.1 which approximates the solution to the **M-CH** model. In particular :

— We consider here a discretized computation box $Q = [-1/2, 1/2]^2$ using $N = 2^8$ nodes in each direction. The initial condition of u is a uniform noise and the numerical parameters are set to $\varepsilon = 1/N$, $\delta_t = \varepsilon^4$, $\alpha = 2$, and $m = \max_{s\in[0,1]} M(s)$.

— First we define the terms $u_k^{n+1/2}$ and $\mu_k^{n+1/2}$ (lines 29-39) as in [73]. Then, we determine λ^{n+1} (lines 42-45) which allows us to correct and obtain u_k^{n+1} and μ_k^{n+1} (lines 48-52).

— Line 24 corresponds to the definition of the Fourier-symbol associated with the operator L_{M_k}. The application of L_{M_k} can then be performed by using a simple multiplication in Fourier space with the array M_L.

— Each computation of a gradient or a divergence is made in Fourier space. For instance the divergence $\operatorname{div}\left[(M(u_k^n)-m)\nabla\left[\sigma_k\mu_k^n+\lambda^k\right]\right]$ is computed on line 31.
— The computation of λ^{n+1} is illustrated on lines 42-45. λ^{n+1} is first computed in Fourier space using the Fourier-symbol of the operator $\left[\sum_k \nu_k L_{M_k}\Delta\right]^{-1}$. Then λ^{n+1} is obtained by applying the discrete inverse Fourier transform.

8.3.2 Numerical scheme for the NMN-CH model

The case of the **NMN-CH** model is slightly more complicated. We first recall the numerical scheme introduced in the case of only two phases, then we explain how to generalize it in the multiphase context. Recall that the **NMN-CH** model reads in the biphase case as

$$\begin{cases}\partial_t u &= N(u)\operatorname{div}\left(M(u)\nabla(N(u)\mu)\right)\\ \mu &= \frac{W'(u)}{\varepsilon^2}-\Delta u\end{cases}$$

and that the Fourier semi-implicit scheme we proposed in [73] to approximate its solutions is

$$\begin{cases}(u^{n+1}-u^n)/\delta_t &= [m\Delta-\beta]\mu^{n+1}+H(u^n,\mu^n),\\ \mu^{n+1} &= \left(-\Delta u^{n+1}+\frac{\alpha}{\varepsilon^2}u^{n+1}\right)+\left(\frac{1}{\varepsilon^2}(W'(u^n)-\alpha u^n)\right),\end{cases}$$

where

$$H(u^n,\mu^n)=N(u^n)\operatorname{div}((M(u^n)\nabla(N(u^n)\mu^n))-m\Delta\mu^n+\beta\mu^n.$$

Remark 8.3.1. *Recall that this approach is based on the convex-concave splitting of the associated metric*

$$\frac{1}{2}\int_Q M(u)\left|\nabla(N(u)\mu)\right|^2dx=J_{u,c}(\mu)+J_{u,e}(\mu),$$

with

$$J_{u,c}(\mu)=\frac{1}{2}\int_Q m|\nabla\mu|^2dx+\frac{1}{2}\int_Q\beta\mu^2dx$$

and

$$J_{u,e}(\mu)=\int_Q G(u)\cdot\nabla\mu\mu dx+\frac{1}{2}\int_Q(|G(u)|^2-\beta)\mu^2dx+\frac{1}{2}\int_Q(1-m)|\nabla\mu|^2dx.$$

Here, $G(u)=-\frac{1}{2}\nabla(\log(M(u)))$, and as it is bounded is $H^1(Q)$, a sufficiently large choice for m and β should ensure the concavity of $J_{u,e}(\mu)$ and the stability of the scheme. In practice, we take $m=1$ and $\beta=1/\varepsilon^2$ for our numerical experiments and these values did not show any sign of instability regardless of the choice of the time step δ_t.

The couple (u^{n+1},μ^{n+1}) associated with the biphasic system is therefore solution of

$$\begin{pmatrix}I_d & -\delta_t(m\Delta-\beta I_d)\\ \Delta-\alpha/\varepsilon^2 & I_d\end{pmatrix}\begin{pmatrix}u^{n+1}\\ \mu^{n+1}\end{pmatrix}=\begin{pmatrix}B^1_{u^n,\mu^n}\\ B^2_{u^n,\mu^n}\end{pmatrix},$$

with

$$\begin{pmatrix}B^1_{u^n,\mu^n}\\ B^2_{u^n,\mu^n}\end{pmatrix}=\begin{pmatrix}u^n+\delta_t H(u^n,\mu^n)\\ \frac{1}{\varepsilon^2}(W'(u^n)-\alpha u^n),\end{pmatrix},$$

hence

$$\begin{cases}u^{n+1} &= L_{NMN}\left[B^1_{u^n,\mu^n}+\delta_t(m\Delta B^2_{u^n,\mu^n}-\beta B^2_{u^n,\mu^n})\right]\\ \mu^{n+1} &= L_{NMN}\left[(-\Delta B^1_{u^n,\mu^n}+\alpha/\varepsilon^2 B^1_{u^n,\mu^n})+B^2_{u^n,\mu^n}\right].\end{cases}$$

where the operator $L_{NMN}=\left(I_d+\delta_t(m\Delta-\beta I_d)(\Delta-\alpha/\varepsilon^2 I_d)\right)^{-1}$ can be computed very efficiently in Fourier space.

We now propose to extend this approach to the multiphase case :

$$\begin{cases}\partial_t u_k &= \nu_k N(u_k)\operatorname{div}\left[M(u_k)\nabla[N(u_k)\sigma_k\mu_k+\lambda)]\right],\\ \mu_k &= \frac{W'(u_k)}{\varepsilon^2}-\Delta u_k,\\ 1 &= \sum_k u_k.\end{cases}$$

The scheme reads

$$\begin{cases}(u_k^{n+1}-u_k^n)/\delta_t &= \nu_k[m\Delta-\beta](\sigma_k\mu_k^{n+1}+\lambda^{n+1})+H_k(u_k^n,\mu_k^n,\lambda^n)\\ \mu_k^{n+1} &= \left(-\Delta u_k^{n+1}+\frac{\alpha}{\varepsilon^2}u_k^{n+1}\right)+\left(\frac{1}{\varepsilon^2}(W'(u_k^n)-\alpha u_k^n)\right),\end{cases}$$

```
clear all; colormap('jet');

 %%%%%%%%%%%%%%%%%% parameters %%%%%%%%%%%%%%%%%%%%%%%%
N = 2^8; epsilon =1/N; dt = epsilon^4; T = 10^(-4);
W_prim = @(U) (U.*(U-1).*(2*U-1));
MobMM = @(U) 2*36/2*((((U).*(1-U)).^2) );
alpha = 2;x = linspace(0,1,N); c=max(MobMM(x));
%%%%%%%%%%%%%%%% condition initiale %%%%%%%%%%%%%%%
U(:,:,1) = 2*rand(N,N)/3; U(:,:,2) = rand(N,N).*(1 -U(:,:,1) );
U(:,:,3) = 1-(U(:,:,1) + U(:,:,2) );
Mu = 0*U;lambda = 0; lambda_fourier = 0;Mu_fourier = zeros(N,N,3);
for k=1:3, U_fourier(:,:,k) = fft2(U(:,:,k)); end

%%%%%%%%%%%%%%%%%%% coefficient - mobilite %%%%%%%%%%
sigma12 =1; sigma13 =1; sigma23 =1;
sigma(1) = (sigma12 + sigma13 - sigma23)/2;
sigma(2) = (sigma12 + sigma23 - sigma13)/2;
sigma(3) = (sigma23 + sigma13 - sigma12)/2;
mob(1) = 1; mob(2)= 1; mob(3) = 1;
%%%%%%%%%%%%%%%%%% Diffusion Fourier %%%%%%%%%%%%%%%%
k = [0:N/2,-N/2+1:-1];
[K1,K2] = meshgrid(k,k);
Delta = -4*pi^2*((K1.^2 + (K2).^2));
for k=1:3, M_L(:,:,k) = 1./(1 + dt*sigma(k)*mob(k)*(c*Delta).*(Delta - alpha/epsilon^2)
    ); end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
k=1;
for i=1:T/dt,
%%%%%%%%%%%%% computation of u^{n+1/2}_k, \mu^{n+1/2}_k
for k=1:3,
mobUk = MobMM(U(:,:,k));
div_mob_laplacien_fourier = 2*1i*pi*K1.*fft2((mobUk-c).*ifft2(2*1i*pi*K1.*(sigma(k)*
    Mu_fourier(:,:,k) + lambda_fourier))...
   + 2*1i*pi*K2.*fft2((mobUk-c).*ifft2(2*1i*pi*K2.*(sigma(k)*Mu_fourier(:,:,k) +
       lambda_fourier)));
B1 = U_fourier(:,:,k) + dt*mob(k)*div_mob_laplacien_fourier;
B2 = fft2(W_prim(U(:,:,k))/epsilon^2 - alpha/epsilon^2*U(:,:,k));
U_fourier(:,:,k) = M_L(:,:,k).*(B1 + dt*mob(k)*sigma(k)*c*Delta.*B2);
U(:,:,k) = real(ifft2(U_fourier(:,:,k)));
Mu_fourier(:,:,k) = M_L(:,:,k).*((alpha/epsilon^2 - Delta).*B1 + B2);
Mu(:,:,k) = ifft2(Mu_fourier(:,:,k));
end

%%%%%%%%%%%%%%%%%%% computation of lambda and correction %%%%%%%%%%
Err_sum = fft2(1 - sum(U,3));
lambda_fourier = (1./((c*(M_L(:,:,1)*mob(1) +M_L(:,:,2)*mob(2) + M_L(:,:,3)*mob(3))).*
    Delta)).*Err_sum/dt;
lambda_fourier(1,1) = 0;
lambda = real(ifft2(lambda_fourier));

for k=1:3,
term = dt*(mob(k)*(c*Delta.*lambda_fourier));
U_fourier(:,:,k) = U_fourier(:,:,k) + M_L(:,:,k).*term;
Mu_fourier(:,:,k) = Mu_fourier(:,:,k) - (Delta - alpha/epsilon^2).*M_L(:,:,k).*term;
U(:,:,k) = real(ifft2(U_fourier(:,:,k)));
Mu(:,:,k) = ifft2(Mu_fourier(:,:,k));
end
end
```

TABLE 8.1 – **Matlab** implementation of our scheme to approximate in dimension 2 the solutions to the **M-CH** model.

where

$$H_k(u_k^n, \mu_k^n, \lambda^n) = \nu_k \left(N(u_k^n) \operatorname{div}((M(u_k^n)\nabla(N(u_k^n)(\sigma_k \mu_k^n + \lambda^n))) - [m\Delta - \beta](\sigma_k \mu_k^n + \lambda^n) \right).$$

and λ^{n+1} is associated to the partition constraint $\sum_k u_k^{n+1} = 1$.

Let us now introduce the couple $(u_k^{n+1/2}, \mu_k^{n+1/2})$ defined by

$$\begin{cases} u_k^{n+1/2} &= L_{NMN,k} \left[B^1_{u_k^n, \mu_k^n, \lambda^n} + \delta_t \nu_k \sigma_k ([m\Delta - \beta] B^2_{u_k^n, \mu_k^n, \lambda^n}) \right] \\ \mu_k^{n+1/2} &= L_{NMN,k} \left[(-\Delta B^1_{u_k^n, \mu_k^n, \lambda^n} + \alpha/\varepsilon^2 B^1_{u_k^n, \mu_k^n, \lambda^n}) + B^2_{u_k^n, \mu_k^n, \lambda_k^n} \right]. \end{cases}$$

where

$$L_{NMN,k} = \left(I_d + \delta_t \nu_k \sigma_k (m\Delta - \beta I_d)(\Delta - \alpha/\varepsilon^2 I_d) \right)^{-1}$$

and

$$B^1_{u^n, \mu^n, \lambda^n} = u^n + \delta_t H(u^n, \mu^n, \lambda^n) \text{ and } \quad B^2_{u^n, \mu^n, \lambda^n} = \frac{1}{\varepsilon^2}(W'(u^n) - \alpha u^n).$$

It is not difficult to see that

$$\begin{cases} u_k^{n+1} &= u_k^{n+1/2} + \delta_t \nu_k L_{NMN,k} \left[[m\Delta - \beta] \lambda^{n+1} \right] \\ \mu_k^{n+1} &= \mu_k^{n+1/2} + \delta_t \nu_k L_{NMN,k} \left[(-\Delta + \frac{\alpha}{\varepsilon^2})(m\Delta - \beta) \lambda^{n+1} \right] \end{cases}$$

which shows that λ^{n+1} satisfies

$$\lambda^{n+1} = \frac{1}{\delta_t} \left[\sum_k \nu_k L_{NMN,k} (m\Delta - \beta) \right]^{-1} (1 - \sum_k u_k^{n+1/2}).$$

where the operator $[\sum_k \nu_k L_{NMN,k}(m\Delta - \beta)]^{-1}$ is homogeneous and can be, again, computed easily in Fourier space.

Similarly to the **M-CH** model, the implementation of the previous scheme is simple and efficient. We present in Table 8.2 an example of a **Matlab** script with less than 60 lines which implements the scheme approximating the solutions to the **NMN-CH** model. In particular :

— We consider here a computation box $Q = [-1/2, 1/2]^2$ discretized with $N = 2^8$ nodes in each direction. The initial condition of u is a uniform noise and the numerical parameters are set to $\varepsilon = 1/N$, $\delta_t = \varepsilon^4$, $\alpha = 2$, $\beta = 2/\varepsilon^2$ and $m = 1$.

— The implementation is almost identical to the previous model. Only the treatment of the divergence term $H_k(u_k^n, \mu_k^n, \lambda^n)$ makes a difference. The computation is done in lines 33 to 36 and is based on the following equality :

$$\begin{aligned} N(u) \operatorname{div}(M(u)\nabla(N(u)\mu)) &= \sqrt{M(u)}\Delta\left(N(u)\mu\right) + N(u)\nabla(M(u)) \cdot \nabla(N(u)\mu) \\ &= \sqrt{M(u)}\Delta\left(N(u)\mu\right) + 2\nabla\left[\sqrt{M(u)}\right] \cdot \nabla(N(u)\mu), \end{aligned}$$

as $N(u) = 1/\sqrt{M(u)}$, see [73] for more details.

— Figure 8.1 shows the function $u_2^n + 2u_3^n$ computed at different times t^n using this script.

We believe that the proposed implementation shows the simplicity, efficiency, and stability of our numerical scheme.

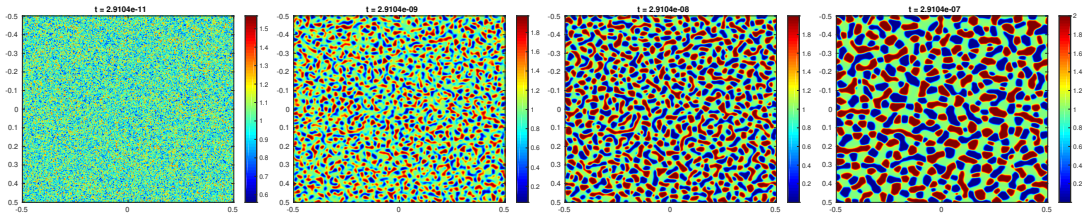


FIGURE 8.1 – First numerical experiment using the **NMN-CH** model ; the solutions u are computed with the **Matlab** script of Table 8.2. We plot the function $x \mapsto u_2(x) + 2u_3(x)$ on each picture which means that the first, second, and third phases appear in blue, green, and red, respectively.

```
clear all;
%%%%%%%%%%%%%%%%% parameters %%%%%%%%%%%%%%%%%%%%%%%%
N = 2^8; epsilon =1/N; dt = epsilon^4; T = 10^(-4);
alpha = 2;gamma=1; beta = 2/epsilon^2;
W_prim = @(U) (U.*(U-1).*(2*U-1));
MobM = @(U) 1/2*((((U).*(1-U)).^2+epsilon^2) );
MobN = @(U) 1./sqrt(MobM(U) );
%%%%%%%%%%%%%%%%% initial condition %%%%%%%%%%%%%%%
U(:,:,1) = 2*rand(N,N)/3; U(:,:,2) = rand(N,N).*(1 -U(:,:,1) );
U(:,:,3) = 1-(U(:,:,1) + U(:,:,2) );
Mu = 0*U;lambda = 0; lambda_fourier = 0;Mu_fourier = zeros(N,N,3);
for k=1:3, U_fourier(:,:,k) = fft2(U(:,:,k)); end
%%%%%%%%%%%%%%%%%% surface tension coefficients - mobilities %%%%%%%%%%
sigma12 =1; sigma13 =1; sigma23 =1;
sigma(1) = (sigma12 + sigma13 - sigma23)/2;
sigma(2) = (sigma12 + sigma23 - sigma13)/2;
sigma(3) = (sigma23 + sigma13 - sigma12)/2;
mob(1) = 1; mob(2)= 1; mob(3) = 1;
%%%%%%%%%%%%%%%%% Kernel %%%%%%%%%%%%%%%
k = [0:N/2,-N/2+1:-1]; [K1,K2] = meshgrid(k,k);
Delta = -4*pi^2*((K1.^2 + (K2).^2)); M_L = zeros(N,N,3);
for k=1:3,
M_L(:,:,k) = 1./(1 + dt*sigma(k)*mob(k)*(gamma*Delta - beta) .*(Delta - alpha/epsilon
    ^2));
end
%%%%%%%%%%loop %%%%%%%%%%%%%%%%%%
for i=1:T/dt,
%%%%%%%%%%%% computation of u^{n+1/2}_k, \mu^{n+1/2}_k
for k=1:3,
mobMUk = MobM(U(:,:,k)); mobNUk = MobN(U(:,:,k));
sqrtMk = sqrt(mobMUk); sqrtMk_fourier = fft2(sqrtMk);
nabla1_sqrtMk= real(ifft2(2*pi*1i*K1.*sqrtMk_fourier )); nabla2_sqrtMk= real(ifft2(2*pi
    *1i*K2.*sqrtMk_fourier ));
muN_fourier = fft2((sigma(k)*Mu(:,:,k) + lambda).*mobNUk);
nabla1_muN = real(ifft2(2*pi*1i*K1.*muN_fourier ));
nabla2_muN = real(ifft2(2*pi*1i*K2.*muN_fourier ));
laplacien_muN = real(ifft2(Delta.*muN_fourier ));
NdivMgradNMu = sqrtMk.*laplacien_muN + 2*(nabla1_sqrtMk.*nabla1_muN +nabla2_sqrtMk.*
    nabla2_muN);
B1 = U_fourier(:,:,k) + dt*(mob(k)*fft2(NdivMgradNMu) - mob(k)*(gamma*Delta-beta).*((
    sigma(k)*Mu_fourier(:,:,k)+lambda_fourier)));
B2 = fft2(W_prim(U(:,:,k))/epsilon^2 - alpha/epsilon^2*U(:,:,k));
U_fourier(:,:,k) = M_L(:,:,k).*(B1 + dt*mob(k)*sigma(k)*(gamma*Delta-beta).*B2);
U(:,:,k) = real(ifft2(U_fourier(:,:,k)));
Mu_fourier(:,:,k) = M_L(:,:,k).*((alpha/epsilon^2 - Delta).*B1 + B2);
Mu(:,:,k) = real(ifft2(Mu_fourier(:,:,k)));
end
%%%%%%%%%%%%%%%%%% computation of lambda and correction %%%%%%%%%%
Err_sum = fft2(1 - sum(U,3));
weight = (M_L(:,:,1)*mob(1) +M_L(:,:,2)*mob(2) + M_L(:,:,3)*mob(3)).*(gamma*Delta-1*
    beta);
lambda_fourier = (1./(weight)).*Err_sum/dt;
lambda = real(ifft2(lambda_fourier));
for k=1:3,
term = (mob(k)*M_L(:,:,k)./((M_L(:,:,1)*mob(1) +M_L(:,:,2)*mob(2) + M_L(:,:,3)*mob(3)))
    ).*Err_sum;
U_fourier(:,:,k) = U_fourier(:,:,k) + term;
Mu_fourier(:,:,k) = Mu_fourier(:,:,k) - (Delta - alpha/epsilon^2).*term;
U(:,:,k) = real(ifft2(U_fourier(:,:,k)));
Mu(:,:,k) = real(ifft2(Mu_fourier(:,:,k)));
end
end
```

TABLE 8.2 – **Matlab** implementation of our scheme to approximate in dimension 2 the solutions to the **NMN-CH** model.

8.3.3 Numerical validation

Asymptotic expansion and flow : a numerical comparison

The first numerical example concerns the evolution of an initial connected set. For each Cahn–Hilliard model, we plot on Figure 8.2 the phase field function $u_2^n+2u_3^n$ computed at different times t. Each experiment is performed using the same numerical parameters : $N = 2^8$, $\varepsilon = \delta_x$, $\delta_t = \varepsilon^4$, $\alpha = 2$, $m = 1$, and $\beta = 2/\varepsilon^2$.

The first and second lines of Figure 8.2 correspond to the solutions given by the **M-CH** and the **NMN-CH** models, respectively. Notice that the numerical experiments obtained with both models are very similar and should give a good approximation of the surface diffusion flow. In addition, for each model, the stationary flow limit appears to correspond to a ball of the same mass as that of the initial set.

To illustrate the asymptotic expansion performed in Section 8.2, we plot on Figure 8.3 (first two pictures) the slice $x_1 \mapsto u_1(x_1, 0)$ at the final time $T = 10^{-4}$. The profile associated to the **M-CH** model is plotted in red and clearly indicates that the solution u does not remain in the interval $[0, 1]$ with an overshoot of order $O(\varepsilon)$. In contrast, the profile obtained using the **NMN-CH** model (in green) seems to be very close to q and remains in $[0, 1]$ up to an error of order $O(\varepsilon^2)$. Finally, we plot the evolution of the Cahn–Hilliard energy along the flow for each model on the last picture of Figure 8.3. We can clearly observe a decrease of the energy in each case.

In conclusion, this first numerical experiment confirms the asymptotic expansion obtained in the previous section, and highlights the interest of our **NMN-CH** model to approximate surface diffusion flows.

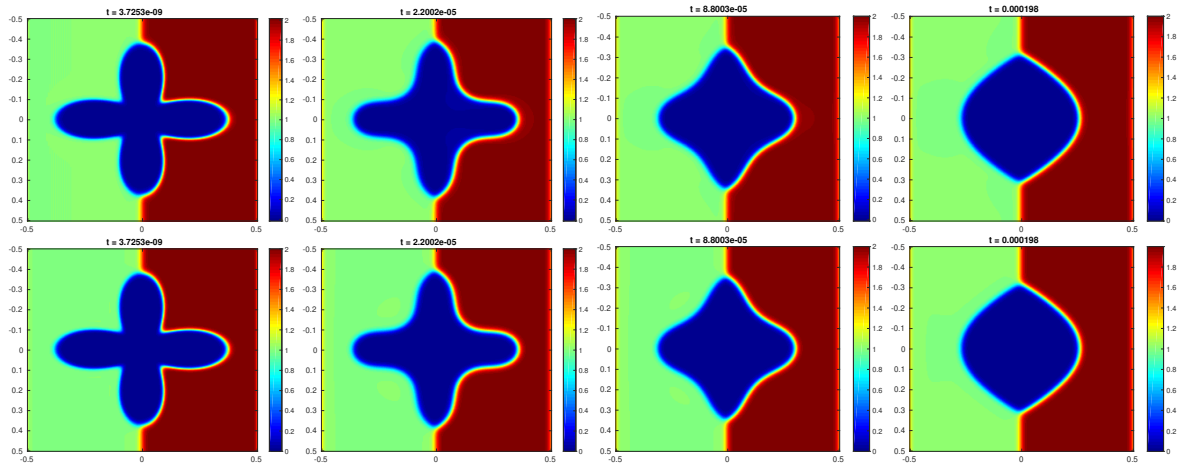


FIGURE 8.2 – First numerical comparison of the two **CH** models : evolution of **u** along the iterations. First line using **M-CH**, second line using **NMN-CH**.

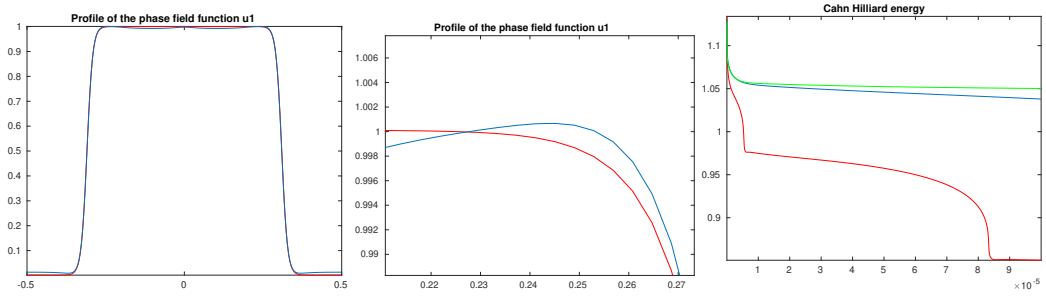


FIGURE 8.3 – Comparison of the two different models : profile and energy ; **M-CH** in blue, **NMN-CH** in red ; First figure : slice of u : $x_1 \mapsto u_1(x_1, 0)$; Second figure : zoom on the slice of u_1 ; last figure : evolution of the Cahn–Hilliard energy along the flow.

Influence of the mobility using the NMN-CH model

The second numerical experiment is intended to show the influence of surface mobilities (ν_{ij}) only on the velocity of each interface. To illustrate this, we show in Figure 8.4 the evolution of **u** in two different cases : a first case where $\nu_i = 1$ (see the first row on Figure 8.4); a second case where $\nu_2 = \nu_3 = 1$ and $\nu_1 = 0$ (see the second row). In both cases, the (σ_i) coefficients associated with surface tensions (σ_{ij}) are set

to $\sigma_i = 1$. As previously, we use the same numerical parameters in each case : we set $N = 2^8$, $\varepsilon = 2/N$, $\delta_t = \varepsilon^4$, $\alpha = 2$, $m = 1$, and $\beta = 2/\varepsilon^2$.

As expected, we observe in the first row of Figure 8.4 that all phases are active along the iterations since the mobility coefficients ν_i are all equal to 1. On the contrary, in the second row, the first phase (u_1 in blue) is fixed along the iterations, which is consistent with the fact that the coefficient mobility associated with the first phase u_1 is $\nu_1 = 0$. Indeed, it is important to notice that mobilities play a role only in the gradient flow and therefore imposing a zero mobility $\nu_k = 0$ forces the k-th phase u_k to be fixed. In particular, this allows us to deal easily and efficiently with the Cahn–Hilliard problem in irregular domains (see [308, 229, 325, 227, 338]) and the second row of Figure 8.4 is a perfect illustration of it. We insist that our model does not impose any boundary conditions on the complex domain, nor the insertion of a surface energy. Another important remark is that the width of the diffuse interface depends only on ε and does not depend neither on surface tensions nor on mobilities.

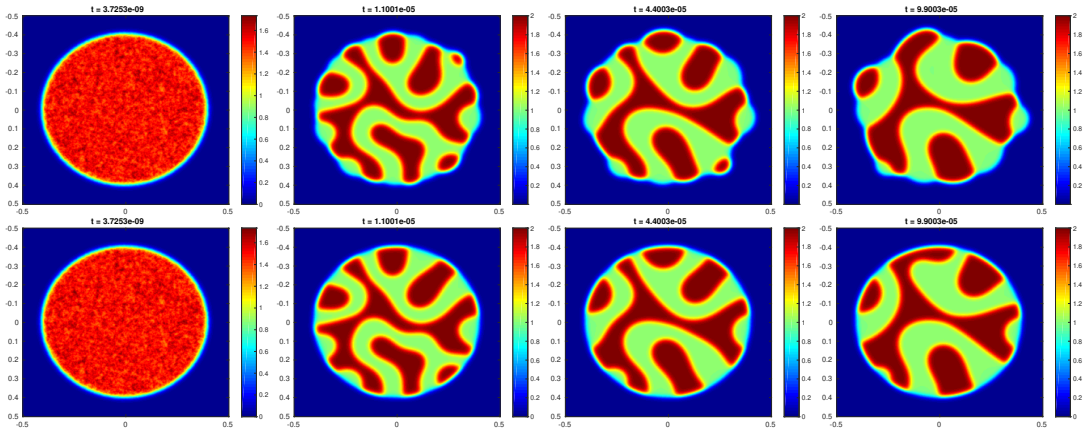


FIGURE 8.4 – Influence of the mobility using **NMN-CH** : evolution of **u** along the iterations ; First line using $\nu_1 = \nu_2 = \nu_3 = 1$; second line, using $\nu_1 = 0$ and $\nu_2 = \nu_3 = 1$.

Influence of the surface tension coefficients using the NMN-CH model

The **NMN-CH** model can also handle the case of the evolution of a liquid phase on a fixed solid surface by simply imposing a null mobility of the solid interface. Here we propose an application in space dimension 2. Figure 8.5 illustrates numerical results obtained with different sets of surface tension coefficients $\sigma = (\sigma_{12}, \sigma_{13}, \sigma_{23})$, with mobilities $\nu_1 = 0,\ \nu_2 = \nu_3 = 1$ and the same initial condition : $\sigma = (1, 1, 1)$, $\sigma = (1.9, 1, 1)$ and $\sigma = (1, 1.9, 1)$ for the first, the second and the third rows of Figure 8.5, respectively. The solid u_1, liquid u_2 and vapor u_3 phases are represented in blue, red, and green, respectively. Similarly to the previous computations, the numerical parameters are set to $N = 2^8$, $\varepsilon = 2/N$, $\delta_t = \varepsilon^4$, $\alpha = 2$, $m = 1$, and $\beta = 2/\varepsilon^2$. As in the previous numerical experiment, we notice the ability of our model to handle the case of null mobilities (here to fix the exterior solid phase u_1 in blue). In Figure 8.5, we can also see the strong influence of the contact angle on the evolution of the liquid phase. We emphasize that our model does not prescribe the contact angle. Rather, its value is an implicit consequence of the multiphase interface energy considered in each simulation.

8.4 Application to the simulation of wetting / dewetting

Let us recall that the behavior of liquids on solid surfaces has been of interest to the academic and engineering communities for many decades. Young [340] determined the optimal shape of a drop at equilibrium on a solid surface. More precisely, the shape of the drop minimizes the following energy :

$$P(\Omega_L) = \int_{\Gamma_{L,S}} \sigma_{L,S} \mathcal{H}^{d-1} + \int_{\Gamma_{L,V}} \sigma_{L,V} d\mathcal{H}^{d-1} + \int_{\Gamma_{L,V}} \sigma_{S,V} \mathcal{H}^{d-1},$$

under a constraint on the volume of the set Ω_L which represents the droplet. Here, $\sigma_{L,S}$, $\sigma_{L,V}$, and $\sigma_{L,V}$ are the surface tensions between the liquid (L), solid (S), and vapor (S) phases. In particular, the minimizers of this energy satisfy Young's law for the contact angle θ of the droplet with the solid.

$$\cos(\theta) = \frac{\sigma_{S,V} - \sigma_{S,L}}{\sigma_{L,V}}.$$

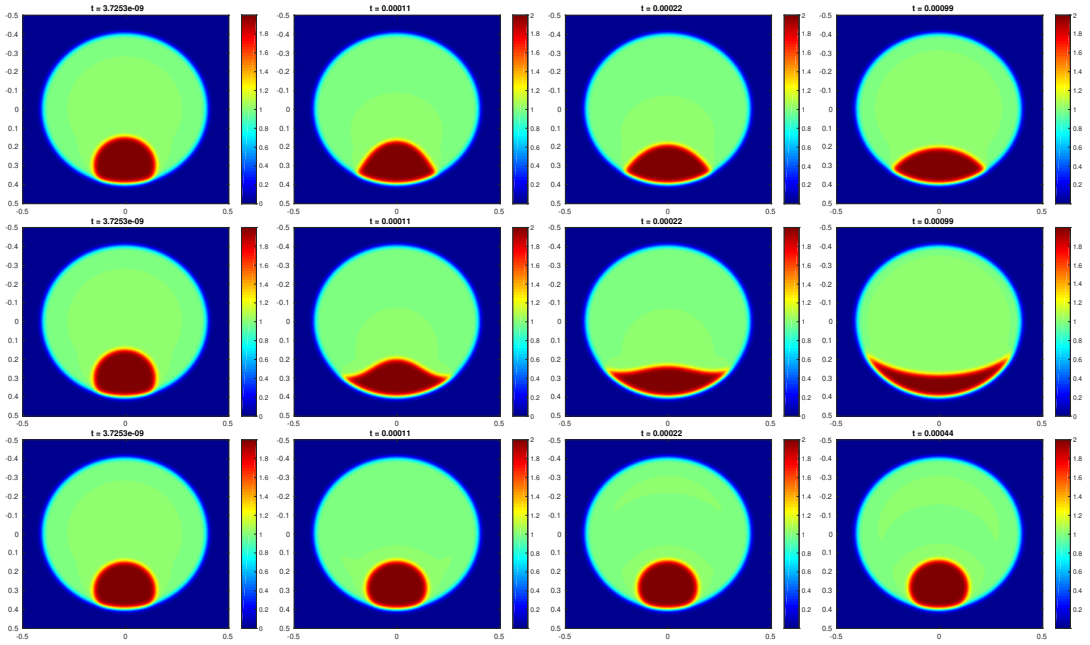


FIGURE 8.5 – Influence of the surface tension coefficients using **NMN-CH** : evolution of $\mathbf{u}$ along the iterations ; First line using $\sigma_{12} = \sigma_{13} = \sigma_{23} = 1$; second line using $\sigma_{12} = 1.9$ and $\sigma_{13} = \sigma_{23} = 1$; third line using $\sigma_{13} = 1.9$, and $\sigma_{12} = \sigma_{23} = 1$.

As illustrated in Figure 8.5, the approximation of such an evolution can be obtained using the previous phase field model with the following set of mobilities : $\nu_{LV} = 1$ and $\nu_{SV} = \nu_{SL} = 0$.

Other approaches have been developed, for example in [80] where Cahn proposed a phase-field approach to model the situation using an additional surface energy on the boundary of the solid phase. However, this method is only applicable for a contact angle $\theta < \frac{\pi}{2}$. An approach using the smoothed boundary conditions in order to force the correct contact angle condition is available in [327] within the Allen-Cahn equation context. Other methods relying on the Allen-Cahn equation using this idea are proposed in [45, 122]. Alternative methods using wall boundary conditions with a third order polynomial to impose the contact angle have been proposed in [312, 313, 20]. A convexity splitting scheme using this idea with a sinusoidal boundary condition can be found in [332]. In [123, 253, 87, 307, 63] the angle is imposed using wall boundary conditions again. The dynamic case can be treated via a coupled Cahn–Hilliard/Navier-Stokes system. In most cases, see for example [201, 232, 327, 2, 61], the contact angle is set to the static contact angle $\frac{\pi}{2}$.

In the convolution-thresholding framework, some recent approaches have been proposed to simulate the wetting phenomenon. Expanding the original scheme of Bence, Merriman, and Osher [46], Esedoglu and Otto have proposed a multiphase convolution-thresholding method in [147] for arbitrary surface tensions satisfying the triangle inequality. Wang et al. then applied this generalization to the wetting case in [331]. A different approach proposed in [336] does not impose the contact angle in the formulation but requires the use of sophisticated techniques while solving the heat equation.

In [71, 67], two of the authors proposed an Allen-Cahn equation where the solid phase was frozen in order to approximate droplet wetting. It was based on the use of zero surface mobilities for the solid-vapor and solid-liquid interfaces. In this paper, we extend this idea to the Cahn–Hilliard equation and, coupled with a reformulation of the problem, we introduce a new simple and efficient method for simulating the wetting phenomenon. It is important to emphasize that this method does not impose the contact angle, which is determined implicitly by the surface tension coefficients $(\sigma_{S,V}, \sigma_{S,L}, \sigma_{L,V})$.

8.4.1 Rewriting of our phase field approach using the liquid phase only

The motivation of this section is to present an equivalent phase field model using only one phase, the liquid phase, as Ω_S is fixed and Ω_V can be obtained from Ω_L and Ω_S. Indeed, numerical experiment in dimension 3 of a complete model (u_L, u_V, u_S) can be quite challenging numerically and it is preferable to reduce the system to the only unknown, the liquid phase.

Recall that in such a case, the Cahn-Hilliard energy reads as

$$P_\varepsilon(\mathbf{u}) = \sum_{k\in\{S,L,V\}} \frac{\sigma_k}{2}\int_Q \frac{\varepsilon}{2}|\nabla u_k|^2 + \frac{1}{\varepsilon}W(u_k),$$

where

$$\sigma_L = \frac{\sigma_{LS}+\sigma_{LV}-\sigma_{SV}}{2}, \sigma_S = \frac{\sigma_{LS}+\sigma_{SV}-\sigma_{LV}}{2} \text{ and } \sigma_V = \frac{\sigma_{LV}+\sigma_{SV}-\sigma_{LS}}{2}.$$

Here, u_S represents the phase field function associated with the solid set Ω_S and the previous asymptotic developments show that u_S should be of the form $u_S = q\left(\frac{\text{dist}(x,\Omega_S)}{\varepsilon}\right)$. On the other hand, the vapor phase field function u_V can be obtained using the partition constraint with $u_V = 1-(u_S+u_L)$. Then the Cahn–Hilliard energy can be rewritten using only the variable u_L as follows :

$$\begin{aligned}\tilde{P}_\varepsilon(u_L) &= \frac{\sigma_L}{2}\int_Q (\frac{\varepsilon}{2}|\nabla u_L|^2 + \frac{1}{\varepsilon}W(u_L))dx \\ &+ \frac{\sigma_V}{2}\int_Q (\frac{\varepsilon}{2}|\nabla(1-(u_S+u_L)|^2 + \frac{1}{\varepsilon}W(1-(u_L+u_S)))dx \\ &+ \frac{\sigma_S}{2}\int_Q (\frac{\varepsilon}{2}|\nabla u_S|^2 + \frac{1}{\varepsilon}W(u_S))dx.\end{aligned}$$

Notice that its L^2-gradient satisfies

$$\nabla_{L^2}\tilde{P}_\varepsilon(u_L) = \frac{\sigma_{SL}}{2}[-\varepsilon\Delta u_L + \frac{1}{\varepsilon}W'(u_L)] + \frac{\sigma_V}{2}\varepsilon R_{u_S}(u_L)$$

where the first term

$$\frac{\sigma_{SL}}{2}[-\varepsilon\Delta u_L + \frac{1}{\varepsilon}W'(u_L)],$$

is a classical Allen-Cahn term and the second term

$$R_{u_S}(u_L) = -\left[\Delta u_S + \frac{1}{\varepsilon^2}(W'(u_L) + W'(1-(u_L+u_S))\right],$$

appears as a smooth penalization term which is active only on the boundary of Ω_S.

Finally, incorporating mobilities leads us to consider the following Cahn–Hilliard models :

— **M-CH model**

$$\begin{cases}\partial_t u_L &= \text{div}\,(M(u_k)\nabla(\sigma_{LV}/2\mu_L + \sigma_V R_{u_S}(u_L))) \\ \mu_L &= \frac{W'(u_L)}{\varepsilon^2} - \Delta u_L\end{cases}$$

— **NMN-CH model**

$$\begin{cases}\partial_t u_L &= N(u_L)\,\text{div}\,(M(u_L)\nabla(N(u_L)\,(\sigma_{LV}/2\mu_L + \sigma_V R_{u_S}(u_L)))) \\ \mu_L &= \frac{W'(u_L)}{\varepsilon^2} - \Delta u_L\end{cases}$$

Note that in practice, we will only use here the **NMN-CH** model for our simulations because the wetting of a thin structure requires to have a model as precise as possible.

Regarding the numerical scheme, the idea is to apply the previous scheme with an explicit treatment of the penalization term $R_{u_S}(u_L)$.

Notice also that the penalization term $R_{u_S}(u_L)$ is active on the whole boundary of Ω_S. In particular, when $u_L = 0$ this term is still active and can be important as it corresponds to the Allen-Cahn term associated to u_S :

$$R_{u_S}(u_L) = -\left(\Delta u_S + \frac{1}{\varepsilon^2}W'(1-u_S)\right) = -\Delta u_S - \frac{1}{\varepsilon^2}W'(u_S).$$

In practice, we propose to localize it only at the liquid phase boundary u_L, which can be done by considering the following variant

$$\tilde{R}_{u_S}(u_L) = R_{u_S}(u_L)\frac{\sqrt{2W(u_L)}}{\sqrt{2W(u_L)+\varepsilon}}.$$

This variant is interesting for it contributes to stabilizing the numerical scheme without disturbing the evolution of the liquid phase.

8.4.2 Influence of the surface tension coefficients

We now propose a numerical experiment in dimension 3 where the initial set is a thin tube. The numerical parameters are given by $N = 2^8$, $\varepsilon = 1/N$, $\delta_t = \varepsilon^4$, $\alpha = 2$, $m = 1$, and $\beta = 2/\varepsilon^2$. We plot on each image of Figure 8.6 the solution $\mathbf{u}$ calculated at different times t where the solid and liquid phase boundaries are plotted in red and gold, respectively. As in the 2D case, surface tension coefficients have a considerable influence on the evolution of the liquid phase. They affect both the wetting rate and the final shape of the liquid phase.

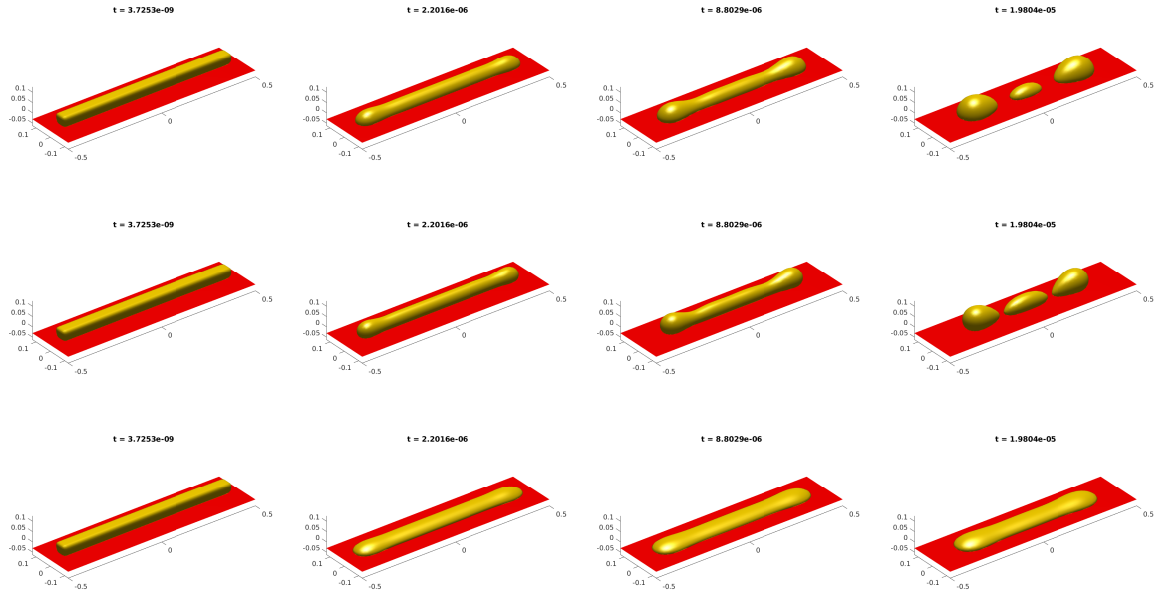


FIGURE 8.6 – Influence of the surface tension coefficients using the **NMN-CH** model : evolution of $\mathbf{u}$ along the iterations ; first line using $\sigma_{LV} = \sigma_{VS} = \sigma_{SL} = 1$; second line using $\sigma_{LS} = 1.7$ and $\sigma_{LV} = \sigma_{VS} = 1$; third line, using $\sigma_{VS} = 1.7$, and $\sigma_{LV} = \sigma_{LS} = 1$.

8.4.3 Influence of the roughness of the solid support

Our approach is also well suited for handling solid supports with roughness, i.e., notably difficult configurations for the simulation of wetting. In Figure 8.7, we test the case of a classical flat support, a support with a randomly generated roughness, and an oscillating support. We observe a direct influence of the substrate roughness on the wetting dynamics, each simulation being initialized in a similar way and using the same set of coefficients.

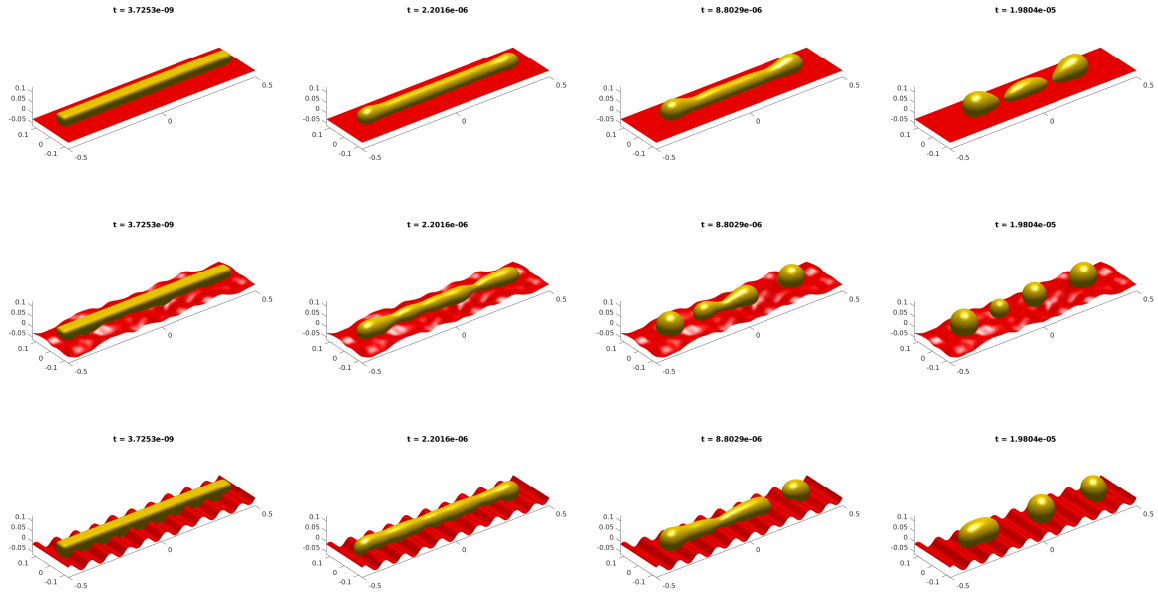


FIGURE 8.7 – Influence of the roughness of the solid support using the **NMN-CH** model : evolution of $\mathbf{u}$ along the iterations using $\sigma_{LS} = 1.7$ and $\sigma_{LV} = \sigma_{VS} = 1$.

Deuxième partie

Approximation de flots géométriques d'interfaces à l'aide de réseaux de neurones

Chapitre 9

Introduction

Dans cette deuxième partie de la thèse, nous allons montrer comment les réseaux de neurones permettent d'améliorer la méthodologie champ de phase pour l'approximation de flots d'interfaces géométriques.

Dans la première partie, nous nous sommes intéressés à l'approximation du flot de diffusion de surface et nous avons notamment montré l'importance de l'ordre de convergence des modèles champ de phase. Il a en effet été nécessaire de travailler sur la modélisation champ de phase afin de proposer des modèles d'ordre 2 adaptés au problème qui nous intéressait, à savoir la modélisation du phénomène de mouillage.

Dans cette seconde partie, l'idée est encore une fois d'améliorer les méthodes d'approximations numériques utilisées, mais en travaillant davantage sur la discrétisation numérique que sur la modélisation champ de phase. Plus précisément, il s'agit d'interpréter les schémas numériques classiques comme des structures de réseaux de champ de phase, qu'il est alors possible d'optimiser dans un processus d'apprentissage, afin d'améliorer la convergence de la méthode numérique vers le flot géométrique d'interfaces souhaité.

Notre stratégie de départ a été de nous intéresser à l'approximation du mouvement par courbure moyenne. Pour cela, notre démarche a d'abord été d'aborder la résolution de l'équation d'Allen-Cahn à partir de réseaux de neurones. Or, les schémas numériques efficaces les plus simples associés à cette équation sont construits simplement en alternant des opérations de convolution et de réaction. Nous nous sommes alors intéressés à des structures de réseaux assez simples, composées de neurones de diffusion (une simple couche de convolution paramétrée par un noyau K) et de neurones de réaction (un simple perceptron multicouche). En effectuant l'entraînement de tels réseaux grâce au package Pytorch, les résultats ont été très concluants et ont permis de retrouver la même efficacité que les schémas classiques de résolution de l'équation d'Allen-Cahn.

Dans une première contribution correspondant à l'article "Learning phase field mean curvature flows with neural networks", rédigé en collaboration avec mes directeurs de thèse et Roland Denis (CNRS, Institut Camille Jordan), nous nous sommes intéressés à l'adaptation de la méthode précédente dans le contexte de flots d'interfaces non orientées. La difficulté ici est qu'il n'existe pas de modèle de champ de phase adapté pour traiter ce cas précis. Il a fallu alors adapter notre stratégie en évitant d'exploiter l'équation d'Allen-Cahn. L'idée a simplement été de constituer une base de données composée d'évolutions de cercles sous une représentation champ de phase qui n'intègre pas l'orientation de celle-ci. Les résultats ont alors montré qu'il était possible, non seulement d'optimiser nos structures de réseaux pour en déduire un algorithme capable de reproduire l'évolution par courbure moyenne d'interfaces régulières, mais aussi de traiter correctement le comportement de points triples alors qu'aucune information sur ces derniers n'avait été transmise dans la base de données. Ces résultats montrent donc l'adaptabilité et la robustesse de nos structures pour approcher l'évolution du mouvement par courbure moyenne. Dans une dernière partie de l'article, nous démontrons également l'utilité des réseaux pré-entraînés en tant qu'alternatives dans diverses applications, telles que la résolution du problème de Steiner ou du problème du Plateau, où les sorties des réseaux sont couplées à des contraintes supplémentaires d'inclusion-exclusion pour approcher la solution. Ces applications variées témoignent de la possibilité d'utiliser nos réseaux entraînés dans des contextes différents de celui pour lequel ils ont été entraînés.

Nous nous sommes ensuite intéressés au cours de cette thèse à l'utilisation de nos réseaux pour l'approximation du flot de Willmore. La motivation ici est double. Tout d'abord, comme nous l'expliquerons plus tard, il existe différents modèles de champ de phase pour approcher ce flot, et le comportement au voisinage d'une singularité est différent d'un modèle à l'autre. Nous souhaitions donc entraîner des réseaux de neurones afin de comprendre comment la structure influence le comportement de l'évolution aux points

singuliers. Une deuxième motivation vient aussi du coût algorithmique des schémas de résolution de ces EDPs d'ordre 4. En s'inspirant des approches de type Bence-Merriman-Osher et de leur efficacité pour approcher de tels flots, nous voulions en déduire des structures de réseaux efficaces pour approcher le flot de Willmore dans un formalisme de champ de phase. Dans ce chapitre, nous présenterons ainsi quelques résultats qui montrent la faisabilité de la méthode.

Une dernière partie concerne l'approximation de mouvement par courbure moyenne anisotrope. Comme nous l'expliquerons, le traitement de l'anisotropie conduit à des équations d'Allen-Cahn anisotropes où le laplacien correspond cette fois-ci à un opérateur de diffusion non linéaire. La difficulté numérique réside alors dans le traitement numérique de cet opérateur. Des approches consistant à linéariser cet opérateur par rapport à la base de Fourier simplifient leur résolution mais perdent aussi en précision. L'idée était alors d'exploiter l'apprentissage de nos réseaux pour identifier de bons opérateurs de diffusion à travers les noyaux entraînés K afin d'améliorer la précision de ces modèles de champ de phase. Un deuxième enjeu, peut-être plus intéressant, concerne aussi l'identification de l'anisotropie qui n'est souvent pas explicite dans les modèles physiques. En effet, il existe des formules qui permettent de retrouver directement l'anisotropie à partir de la connaissance des noyaux de diffusion. Notre idée était donc d'identifier différentes anisotropies en entraînant nos réseaux sur des données réelles. Cette dernière partie ne sera abordée que superficiellement.

Chapitre 10

Learning phase field mean curvature flows with neural networks

10.1 Introduction

Many applications in physics, biology, mechanics, engineering, or image processing involve interfaces whose shape evolves to decrease a particular surface energy.

A very common example is the area energy that explains for instance the shapes of soap bubbles, bee honeycombs, the interface between two fluids, some crystalline materials, some communication networks, etc.

In image processing, the area energy is used to quantify regularity, for instance in the celebrated TV (total variation) model. It is well-known that the L^2-gradient flow of the area energy, i.e. the flow which decreases the energy in the direction of steepest descent with respect to the L^2 metric, is the mean curvature flow.

The mean curvature flow is classically defined for smooth, embedded $(N-1)$-surfaces in $\mathbb{R}^N$ without a boundary : each point of the surface moves with the velocity vector equal to the (vector) mean curvature, see [33]. Such a flow is well defined until the onset of singularities. Various definitions of mean curvature flows have been proposed to handle singularities as well, but also to handle non orientable sets (typically planar networks with triple points satisfying the Herring's condition [246]), higher codimensional sets, or sets with boundaries, see the many references in [33].

We propose in this paper to learn with neural networks a numerical approximation of a phase field representation of either the mean curvature flow of an oriented set, or the mean curvature flow of a possibly non orientable interface. Interestingly, as will be seen later, we train our neural networks on few examples of smooth sets flowing by mean curvature but, once trained, some networks can handle consistently sets with singularities as well.

There is a vast literature on the numerical approximation of mean curvature flows, and the methods roughly divide in four categories (some of them are exhaustively reviewed and compared in [119]) :

1. Parametric methods [119, 26] are based on explicit parameterizations of smooth surfaces. The numerical approximation of the parametric mean curvature flow is quite simple in 2D and the approach can be extended to non-orientable surfaces since there is no need for the surface to be the boundary of a domain nor to separate its interior from its exterior. The numerical approximation is however more difficult in dimensions higher than 2 for the method can hardly handle topological changes. The processing of singularities is difficult even in dimension 2, see for instance the recent work [282].
2. The level set method was introduced by Osher and Sethian [271] for interface geometric evolution problems, see also [269, 270, 153, 97]. The main idea is to represent implicitly the interface as the zero level-set of an auxiliary function φ (typically the signed distance function associated with the domain enclosed by the interface) and the evolution is described through a Hamilton-Jacobi equation satisfied by φ. The level set approach provides a convenient formalism to represent the mean curvature flow in any dimension. In a strong contrast with the parametric approach, it can handle topological changes and it can be defined rigorously beyond singularities using the theory of viscosity solutions for the Hamilton-Jacobi equation. There are however several difficult issues regarding the numerical approximation of the level set approach. First, the Hamilton-Jacobi equation is nonlinear and highly degenerate, thus difficult to approximate numerically and, secondly, delicate methods are needed to preserve some needed properties of the level set function φ. Moreover, the method is basically designed to represent the evolution of the boundary of an oriented domain and, to the best of our

knowledge, there is no level set method that can handle the mean curvature flow of non-orientable interfaces.

3. Convolution/thresholding type algorithms [46, 200, 295] involve a time-discrete scheme alternating the convolution with a suitable kernel of the characteristic function at time t_n of the domain enclosed by the interface, followed by a thresholding step to define the set at time $t_n + dt$. The asymptotic limit of the scheme coincides with the smooth mean curvature flow. It should also be noted that in recent work [147, 217], the authors have succeeded in reinterpreting these thresholding schemes as gradient flows, allowing them to obtain convergence and stability results even in the multiphase context and for anisotropic energies.
4. Phase field approaches [258, 94] constitute the fourth category of methods for the numerical approximation of the mean curvature flow. In these approaches, the sharp interface between is approximated by a smooth transition, the interfacial area is approximated by a smooth energy depending on the smooth transition, and the gradient flow of this energy appears to be a relatively simple reaction-diffusion system. Phase field approaches are widely used in physics since the seminal works of van der Waals' on liquid-vapor interfaces (1893), of Ginzburg & Landau on superconductivity (1950), and of Cahn & Hilliard on binary alloys (1958). However, most phase field methods are designed to approximate the mean curvature flow of the boundary of a domain but cannot handle the evolution of non-orientable interfaces.

Our work starts with the following question : can we design and train neural networks to approximate the mean curvature flow of either oriented or non-orientable interfaces ? Our strategy is to draw inspiration from phase field approaches and their numerical approximations.

The phase field approach to the mean curvature flow of domain boundaries A time-dependent smooth domain $\Omega(t) \subset \mathbb{R}^d$ evolves under the classical mean curvature flow if its inner normal velocity satisfies $V_n(t) = H(t)$, where $H(t)$ denotes the scalar mean curvature of the boundary $\partial\Omega(t)$ (with the convention that the scalar mean curvature on the boundary of a convex domain is positive). This evolution coincides with the L^2-gradient flow of the perimeter of $\Omega(t)$

$$P(\Omega(t)) = \int_{\partial\Omega(t)} d\mathcal{H}^{d-1},$$

where $\mathcal{H}^{d-1}$ denotes the $(d-1)$-dimensional Hausdorff measure. In the phase field approach, the perimeter functional is approximated (up to a multiplicative constant) in the sense of Γ-convergence [258, 94] by the Cahn-Hilliard energy P_ε defined for every smooth function u by

$$P_\varepsilon(u) = \int_{\mathbb{R}^d} (\varepsilon \frac{|\nabla u|^2}{2} + \frac{1}{\varepsilon} W(u)) dx,$$

where ε is a real positive parameter which quantifies the accuracy of the approximation, and W is a double-well potential, typically $W(s) = \frac{1}{2}s^2(1-s)^2$. It follows from the Γ-convergence as $\varepsilon \to 0$ of P_ε to $c_W P$ (c_W is a constant depending only on W) that when u is a smooth approximation of the characteristic function of $\Omega(t)$, $P_\varepsilon(u)$ is close to $c_W P(\Omega(t))$.

The L^2-gradient flow of the Cahn-Hilliard energy P_ε leads to the celebrated Allen-Cahn equation [11] which reads as, up to a rescaling :

$$\partial_t u = \Delta u - \frac{1}{\varepsilon^2} W'(u) \tag{10.1.1}$$

The existence and uniqueness of a solution to the Allen-Cahn equation and the fact that it satisfies a comparison principle are well-known properties, see for instance [11, theorem 32].

With the choice $W(s) = \frac{1}{2}s^2(1-s)^2$, a evolving set associated naturally with the Allen-Cahn equation is

$$\Omega_\varepsilon(t) = \left\{ u_\varepsilon(\cdot, t) \geq \frac{1}{2} \right\},$$

where u_ε denotes the solution to (10.1.1) with the well-prepared initial data

$$u_\varepsilon(x, 0) = q\left(\frac{d(x, \Omega(0))}{\varepsilon}\right). \tag{10.1.2}$$

Remark 10.1.1. *Notice that the gradient of the solution of the Allen Cahn equation u_ε can cancel on the boundary of the set Ω_ε and thus lead to a non-smooth interfaces. This is typically the case in dimension 3 where singularities can appear in finite time.*

Here, $d(\cdot,\Omega(0))$ denotes the signed distance function to $\Omega(0)$ with the convention that $d(\cdot,\Omega(0)) < 0$ in $\Omega(0)$ and $q : \mathbb{R} \to [0,1]$ is a so-called *optimal profile* which minimizes the parameter-free one-dimensional Allen-Cahn energy under some constraints :

$$q = \underset{p}{\operatorname{argmin}} \left\{ \int_{\mathbb{R}} (\frac{|p'(s)|^2}{2} + W(p(s)))ds,\ p \in C^{0,1}(\mathbb{R}),\ p(-\infty) = 1,\ p(0) = \frac{1}{2},\ p(+\infty) = 0 \right\}$$

Note that in the particular case $W(s) = \frac{1}{2}s^2(1-s)^2$, one has $q(t) = \frac{1}{2}(1 - \tanh(\frac{t}{2}))$. The initial condition $u_\varepsilon(x,0) = q\left(\frac{d(x,\Omega(0))}{\varepsilon}\right)$ is considered as well-prepared initial data because $q\left(\frac{d(x,\Omega(0))}{\varepsilon}\right)$ is almost energetically optimal : with a suitable gluing $\tilde{q}$ of q to 1 on the left and 0 on the right, one gets that

$$P_\varepsilon\left(\tilde{q}\left(\frac{d(x,\Omega(0))}{\varepsilon}\right)\right) \longrightarrow c_W P(\Omega(0)),$$

as $\varepsilon \to 0^+$.

A formal asymptotic expansion of the solution u_ε to (10.1.1)-(10.1.2) near the associated interface $\partial\Omega_\varepsilon(t) = \partial\left\{u_\varepsilon(\cdot,t) \geq \frac{1}{2}\right\}$ gives (see [33])

$$u_\varepsilon(x,t) = q\left(\frac{d(x,\Omega_\varepsilon(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2), \tag{10.1.3}$$

witch shows that u_ε remains energetically quasi-optimal with second-order accuracy. Furthermore, the velocity V_ε of the boundary $\partial\Omega_\varepsilon(t)$ satisfies

$$V_\varepsilon(t) = H_\varepsilon(t) + \mathcal{O}(\varepsilon^2),$$

where $H_\varepsilon(t)$ denotes the scalar mean curvature on $\partial\Omega_\varepsilon(t)$, which suggests that the Allen-Cahn equation approximates a mean curvature flow with an error of order ε^2 [33].

A rigorous proof of the convergence to the smooth mean curvature flow for short times (in particular before the onset of singularities) has been proved in [94, 117, 41] with a quasi-optimal error on the Hausdorff distance between $\Omega(t)$ and $\Omega_\varepsilon(t)$ given by

$$\operatorname{dist}_{\mathcal{H}}(\Omega(t),\Omega_\varepsilon(t)) \leq C\epsilon^2|\log(\varepsilon)|^2,$$

where the constant C depends on the regularity of $\Omega(t)$.

These convergence results, combined with the good suitability of the Allen-Cahn equation for numerical approximation, make the phase field approach a very effective method to approximate the mean curvature flow. This is true however only for the mean curvature motion of domain boundaries, i.e. codimension 1 orientable interfaces without boundary. There do exist some phase field energies to approximate the perimeter of non-orientable interfaces, as for instance in the Ambrosio-Torterelli functional [18], but these energies need to be coupled with additional terms and cannot be used to approximate directly the mean curvature motion of the interfaces.

Neural networks and phase field representation We introduce in this paper neural networks that learn, at least approximately, how to move a set by mean curvature. Our networks are trained on a collection of sets whose motion by mean curvature is, preferentially, known exactly. However, these networks are not designed to work with exact sets, but rather with an implicit, smooth representation of them. A key aspect of our approach is the choice of this representation. A first option (see Table 10.1, left column) is to choose the representation provided by the Allen-Cahn phase field approach, i.e., given a set $\Omega(0)$, we compute the solution $u_\varepsilon(\cdot,t)$ to the Allen-Cahn equation (10.1.1) with initial data $u_\varepsilon(\cdot,0) = q\left(\frac{d(\cdot,\Omega(0))}{\varepsilon}\right)$. Recall that with the particular choice $W(s) = \frac{1}{2}s^2(1-s)^2$, the $1/2$–isosurface of $u_\varepsilon(\cdot,0)$ is *exactly* $\partial\Omega(0)$. However, using the Allen-Cahn phase field approach introduces a bias : if $t \mapsto \Omega(t)$ denotes the motion by mean curvature starting from $\Omega(0)$ (Table 10.1, middle column), the $1/2$–isosurface of $u_\varepsilon(\cdot,t)$ is no more than a good approximation of $\partial\Omega(t)$, in general it is different (and the same holds for other isosurfaces of $u_\varepsilon(\cdot,t)$). Instead, we use another phase field representation which introduces no bias, see Table 10.1, right column : for all t (before the onset of singularities), the $1/2$–isosurface at time t of $q\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ is *exactly* $\partial\Omega(t)$ (still assuming that $W(s) = \frac{1}{2}s^2(1-s)^2$ but the argument can obviously be adapted for other choices of W). With such a choice, we ensure that our neural networks will be trained on exact implicit representations of the sets moving by mean curvature.

It is obviously more accurate to train our networks with the above choice of exact phase field representations, rather than with the Allen-Cahn phase field approximate solutions. Beyond accuracy, there is another

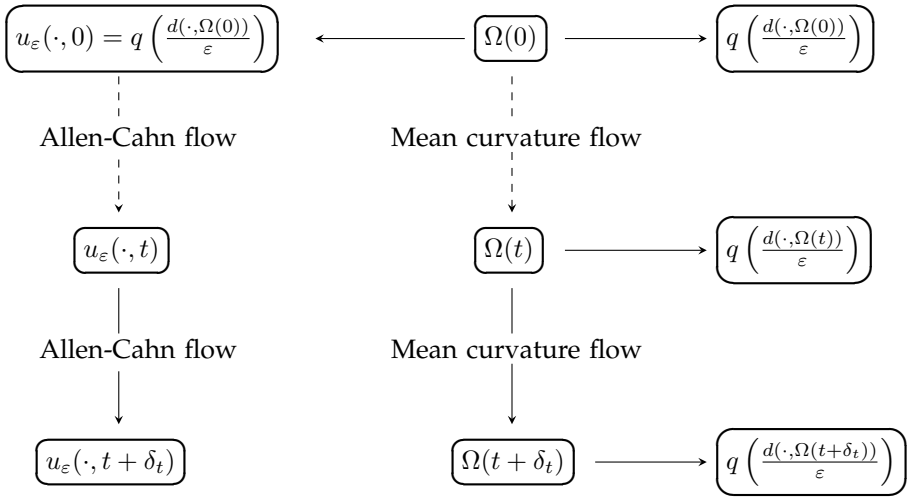


TABLE 10.1 – **Approximate vs exact phase field representations of mean curvature flows**. Middle column : a smooth motion by mean curvature $t \mapsto \Omega(t)$. Left column : the solution to the Allen-Cahn equation starting from the energetically quasi-optimal and exact phase field representation $q\left(\frac{d(\cdot,\Omega(0))}{\varepsilon}\right)$. Right column : exact phase field representations $q\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ of the $\Omega(t)$'s.

key advantage of this choice : the possibility to address situations which are beyond the capacity of the classical Allen-Cahn phase field approach. Indeed, instead of working with the representation $q\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ which is well suited for domain boundaries, other implicit representations can be used which do not require any interface orientation. A typical example is the phase field $q'\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ where the optimal profile q has been simply replaced by its derivative q'. In the case $W(s) = \frac{1}{2}s^2(1-s)^2$, one has $q(s) = \frac{1}{2}(1-\tanh(\frac{s}{2}))$ and its derivative $q'(s) = \frac{1}{4}(\tanh^2(\frac{s}{2}) - 1)$ is even so $q'\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ is symmetric on both sides of $\partial\Omega(t)$, therefore

$$q'\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right) = q'\left(\frac{\mathrm{dist}(\cdot,\partial\Omega(t))}{\varepsilon}\right)$$

where dist denotes the classical distance function. Note that $q'\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$ is another *exact* phase field representation : its $\frac{1}{4}$–isosurface coincides with $\partial\Omega(t)$. However, in contrast with $q\left(\frac{d(\cdot,\Omega(t))}{\varepsilon}\right)$, the phase field is identical on both sides of $\partial\Omega(t)$. To illustrate this idea of another phase field representation which does not carry any orientation information, Figure 10.1 shows the two phase field representations of a circle obtained using either $q(t) = \frac{1}{2}(1-\tanh(\frac{t}{2}))$ or q'. Obviously, the approach can be adapted for other choices of the profile q.

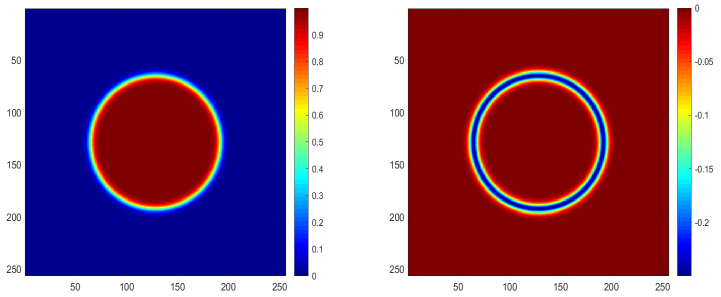


FIGURE 10.1 – Two phase field representations of the same circle C using the profile $q(t) = \frac{1}{2}(1-\tanh(\frac{t}{2}))$ and its derivative : on the left, $q\left(\frac{d(\cdot,D}{\varepsilon}\right)$ with D the disk enclosed by C. On the right, $q'\left(\frac{\mathrm{dist}(\cdot,C)}{\varepsilon}\right)$. The first phase field representation carries an orientation information, the second one does not.

Our contribution in this paper is a new class of neural networks aimed to learn, at least approximately, the flow by mean curvature of either oriented or non orientable sets. Since the classical motion by mean curvature of a smooth set can be defined with nonlinear partial differential equations, our contribution

falls in the category of learning methods for PDEs. Let us give a brief overview of the literature on neural network-based numerical methods for approximating the solutions to PDEs.

— A first category of approaches relies on the fact that very general functions can be approximated with neural networks [109, 205] so it is natural to seek the solution to a PDE as a neural network [187, 349, 3, 49, 204, 189]. Such a method is accurate and useful for specific problems, but not convenient : for each new initial condition, a new neural network must be trained. This approach was for instance used in [189] to numerically solve high-dimensional nonlinear partial differential equations with the particular example of the Allen-Cahn equation as part of their experiments. This approach is not well suited to our purposes as it requires at each time to train a new network to produce an approximation of the solution at time t_{n+1} from the solution at time t_n. However, their method needs to know the specific structure of the equation, which is limited when the underlying equation is only partially known. Another approach considers a neural network as an operator between Euclidean spaces of same dimension depending on the discretization of the PDE [285, 23, 316, 274, 341]. This approach depends on the discretization and requires to modify the architecture of the network when the discrete resolution or the discretization are changed ;
— Most neural network architectures can be interpreted as numerical schemes [240, 8]. Neural networks can therefore be seen as operators acting between infinite-dimensional spaces (typically spaces of functions) : for instance, for a time-dependent PDE, the forward propagation of an associated neural network can be viewed as the flow associated to the PDE when a time-step δ_t is fixed. Consequently, the neural network is trained only once : for each new initial condition, the solution is obtained by applying the neural network to the initial condition. This reduces significantly the computational cost in comparison with the first category of approaches mentioned above. This second category of approaches is mesh-free and fully data-driven, i.e., the learning procedure, as well as the neural network, do not require any knowledge of the underlying PDE, only the knowledge of particular solutions to the PDE. This can be very advantageous when very limited information is given about the PDE, which is often the case in physics. Recently, several works have followed this idea, see [237, 50, 265, 19, 278, 230]. In [237] the authors propose a network architecture based on a theorem of approximation by neural networks of infinite dimensional operators. Very recently, the authors of [230] developed a new neural network based on the Fourier transform. The latter has the advantage of being very simple to implement and computationally very cheap thanks to the Fast Fourier Transform.
— Lastly, there are also methods based on a stochastic approach which uses the links between PDEs and stochastic processes (see [52] and references therein for more details).

The approach proposed in this paper falls in the second category of approaches, i.e., our neural networks approximate the action of semi-group operators for which only few information is known, they are fully data-driven and enough training data is available to get an accurate approximation.

Outline of the paper The paper is organized as follows : we first present in Section 10.2 the strategy we adopt for the construction of our numerical schemes based on neural networks. In particular, we introduce the different semigroups involved in our study and detail the whole learning protocol starting with the choice of the training data and the metrics used in the learning procedure. The different neural network architectures are described in Section 10.2, the architectures being inspired by the discretization schemes of the Allen-Cahn equation in the same spirit as in [158, 342]. We focus on two particular networks denoted as $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, and we address in Section 10.3 their capacity to approximate the mean curvature flows of either oriented or non-orientable surfaces. In the oriented case, we provide some numerical experiments and we give quantitative error estimates to highlight the accuracy and the stability of the numerical schemes associated with our networks. In the non orientable case, both networks seem to learn the flow but a first validation shows that $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ fails to maintain the evolution of a circle while $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ succeeds perfectly. To test the reliability of $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ to approximate the mean curvature flow, evolutions starting from different initial sets are shown in Section 10.3. We observe in particular that $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ is able to handle correctly non-orientable surfaces, even with singularities, and the Herring's condition [67] seems to be respected at points with triple junction. This is somewhat surprising because only evolving smooth sets are used to train our networks (e.g., circles in $2D$, spheres in $3D$). Finally, we propose various applications in Section 10.4 (multiphase mean curvature flows, Steiner trees, minimal surfaces) to bring out the versatility of our approach and to show that the schemes derived from our trained networks are sufficiently stable to be coupled with additional constraints such as volume conservation or inclusion constraints, and sufficiently stable to be extended to the multiphase case.

10.2 Neural networks and phase field mean curvature flows

A first possible way, which is not the one we will opt for, to associate neural networks and phase field mean curvature flows is to compute approximations to the solutions of the Allen-Cahn equation (10.1.1) by training a neural network $\mathcal{S}_\theta^{\mathrm{NN}}$ (depending on a parameter vector θ) to reproduce the action of the Allen-Cahn semigroup $\mathcal{S}_{\delta_t,\varepsilon}^{\mathrm{AC}}$ defined by

$$\mathcal{S}_{\delta_t,\varepsilon}^{\mathrm{AC}}[u_\varepsilon(\cdot,t)] = u_\varepsilon(\cdot,t+\delta_t),$$

where u_ε is solution to the Allen-Cahn equation (10.1.1) on a set Q with periodic boundary conditions and where the parameters δ_t and ε are fixed.

From this neural network, it is possible to derive the simple numerical scheme

$$u^{n+1} = \mathcal{S}_\theta^{\mathrm{NN}}[u^n],$$

where the iterate u^n is expected to be a good approximation of u_ε at time $t_n = n\delta$. However, as explained previously, learning to compute solutions to the Allen-Cahn equation is not necessarily of great interest since extremely simple and robust numerical schemes for computing these solutions already exist.

The idea we propose in this paper is rather to train a network $\mathcal{S}_\theta^{\mathrm{NN}}$ to approximate the semigroup $\mathcal{S}_{\delta_t,\varepsilon}^{q}$ defined by

$$\mathcal{S}_{\delta_t,\varepsilon}^{q}[v_\varepsilon(\cdot,t)] = v_\varepsilon(\cdot,t+\delta_t),$$

where $v_\varepsilon = q\left(\frac{d(x,\Omega(t)}{\varepsilon}\right)$ is an exact phase field representation of an exact mean curvature flow $t \mapsto \Omega(t)$. This second approach is by nature more accurate than the above one since the Allen-Cahn equation is only an approximation to the mean curvature flow whereas v_ε encodes exactly the flow. However, the convergence results of the Allen-Cahn equation to the mean curvature flow show that the two semigroups $\mathcal{S}_{\delta_t,\varepsilon}^{\mathrm{AC}}$ and $\mathcal{S}_{\delta_t,\varepsilon}^{q}$ are very close. And since very efficient numerical schemes exist for the Allen-Cahn equation, it makes sense to take inspiration from their structures to design efficient networks.

In a second step, we will adapt our strategy to approximate the mean curvature flow of possibly non-orientable sets by simply replacing the phase field profile q. More precisely, we will introduce a network $\mathcal{S}_\theta^{\mathrm{NN}}$ to learn an approximation of the semigroup $\mathcal{S}_{\delta_t,\varepsilon}^{q'}$ defined by

$$\mathcal{S}_{\delta_t,\varepsilon}^{q'}[w_\varepsilon(\cdot,t)] = w_\varepsilon(\cdot,t+\delta_t),$$

where $w_\varepsilon = q'\left(\frac{\mathrm{dist}(x,\Gamma(t))}{\varepsilon}\right)$ with $t \mapsto \Gamma(t)$ the mean curvature flow of a possible non orientable set.

10.2.1 Training database and loss function

Before providing details about the structure of our neural networks, let us describe the data on which they will be trained and the training energy (the so-called *loss function* in the literature of neural networks). As before, we shall denote with $t \mapsto \Omega(t)$ the mean curvature flow associated with an initial smooth open set $\Omega(0)$, and with $t \mapsto \Gamma(t)$ the mean curvature flow associated with an initial, possibly non-orientable set $\Gamma(0)$.

Recall that our idea is to obtain an approximation of the operator $\mathcal{S}_{\delta_t,\varepsilon}^{\varphi}$, where $\varphi = q$ or $\varphi = q'$, by training a neural network S_θ^{NN} on a suitable dataset. We expect that the numerical scheme

$$u^{n+1} = \mathcal{S}_\theta^{\mathrm{NN}}[u^n]$$

coupled with the initial data

$$u^0(x) = \begin{cases} q\left(\frac{d(x,\Omega(0))}{\varepsilon}\right) & \text{if } \varphi = q, \\ q'\left(\frac{\mathrm{dist}(x,\Gamma(0))}{\varepsilon}\right) & \text{if } \varphi = q', \end{cases}$$

will be a good approximation of the mean curvature flows starting from either $\Omega(0)$ or $\Gamma(0)$.

The training of the network S_θ^{NN} consists in a gradient descent for the parameter vector θ with respect to a loss energy defined from the exact mean curvature flows of various circles in 2D, of various spheres in 3D, etc.

Let us describe more precisely how the dataset and the associated loss energy are constructed in dimension $d = 2$, the construction being similar in higher dimension. Recall that a circle of radius R_0 evolving under mean curvature flow remains a circle of radius $R(t) = \sqrt{R_0^2 - 2t}$ which decreases until the extinction time $T_{R_0} = \frac{R_0^2}{2}$. We select a finite family of radii R_i for $i \in \{1, \cdots, N_{\mathrm{train}}\}$ and we define the training dataset as follows :

$$\{(X_i, Y_i)\}_{i\in\{1\cdots N_{\text{train}}\}} = \left(\varphi_{R_i}, \varphi_{\sqrt{R_i^2-2\delta_t}}\right)_{i\in\{1\cdots N_{\text{train}}\}}$$

where

$$\varphi_R(x) = \begin{cases} q\left(\frac{\mathrm{d}(x,B_R)}{\varepsilon}\right) & \text{if } \varphi = q, \\ q'\left(\frac{\mathrm{dist}(x,\partial B_R)}{\varepsilon}\right) & \text{if } \varphi = q'. \end{cases}$$

with B_R the ball of radius R centred at 0.

We introduce a first loss function :

$$J_1(\theta) = \frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}} \|\mathcal{S}_\theta^{\text{NN}}[X_i] - Y_i\|^2 = \frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}} \int_Q \left(\mathcal{S}_\theta^{\text{NN}}[\varphi_{R_i}] - \varphi_{\sqrt{R_i^2-2\delta_t}}\right)^2 dx.$$

To stabilize the training of the network S_θ^{NN}, one can opt for a multipoint version which consists in introducing the enriched data set

$$\{(X_i, Y_{i,j})\}_{\{i\in\{1,\cdots,N_{\text{train}}\},\, j\in\{1,\cdots,k\}\}} = \left(\varphi_{R_i}, \varphi_{\sqrt{R_i^2-2j\delta_t}}\right)_{\{i\in\{1,\cdots,N_{\text{train}}\},\, j\in\{1,\cdots,k\}\}}$$

and in minimizing the loss functional J_k defined by

$$J_k(\theta) = \frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{j=1}^{k} \|(\mathcal{S}_\theta^{\text{NN}})^{(j)}[X_i] - Y_{i,j}\|^2 = \frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{j=1}^{k} \int_Q \left((\mathcal{S}_\theta^{\text{NN}})^{(j)}[\varphi_{R_i}] - \varphi_{\sqrt{R_i^2-2j\delta_t}}\right)^2 dx,$$

where $(\mathcal{S}_\theta^{\text{NN}})^{(j)}$ is the iterated j times composition of S_θ^{NN}.

In all our experiments, the computational domain is $[0,1]^d$ and 2D/3D training data consists of an average of $N_{\text{train}} = 100$ circles/spheres with a range of radii ranging from 0.05 to 0.45 to capture enough information about the curvatures.

10.2.2 From the numerical approximation of the Allen-Cahn semigroup to the structure of neural networks

We introduce in this section efficient neural network structures to approximate the semigroups $\mathcal{S}^q_{\delta_t,\varepsilon}$ and $\mathcal{S}^{q'}_{\delta_t,\varepsilon}$. The design of a network architecture is a very delicate question because there is no single generic choice of network that can accurately approximate an operator. The approach developed in this paper consists in deriving the architecture of our networks from splitting methods, which are often used to solve numerically evolution equations having a gradient flow structure. Doing so, we draw on the rich knowledge of numerical analysis to guide us in designing new and more efficient networks.

To derive the structure of our networks, let us start by recalling the principle of splitting schemes [224] to approximate the solutions of the Allen-Cahn equation defined in the domain $Q = [0,1]^d$ with periodic boundary conditions.(for a recent review of numerical methods for phase field approximation of various geometric flows see [126]). As the two semigroups $S^{\text{AC}}_{\delta_t,\varepsilon}$ and $\mathcal{S}^q_{\delta_t,\varepsilon}$ are closely related in the case of smooth mean curvature motion, our expectation is that imitating splitting schemes will lead us to very efficient networks to approximate first $\mathcal{S}^q_{\delta_t,\varepsilon}$, but also $\mathcal{S}^{q'}_{\delta_t,\varepsilon}$.

Given a time step δ_t, we construct an approximation sequence $(u^n)_{n\geq 0}$ of the solution u_ε of (10.1.1) at time $n\delta_t$ using various splitting approaches.

First-order neural network $\mathcal{S}^{\text{NN}}_{\theta,1}$

The first splitting method is the semi-implicit approach where the sequence (u^n) is defined recursively from

$$\frac{u^{n+1}-u^n}{\delta_t} = \Delta u^{n+1} - \frac{1}{\varepsilon^2}W'(u^n),$$

i.e.,

$$u^{n+1} = (I_d - \delta_t\Delta)^{-1}\left(u^n - \frac{\delta_t}{\varepsilon^2}W'(u^n)\right). \tag{10.2.1}$$

More precisely, this numerical scheme can be written as a combination of a convolution kernel K_1 and an activation function ρ_1 :

$$u^{n+1} = K_1 * \rho_1(u^n),$$

with $\rho_1(s) = s - \frac{\delta_t}{\varepsilon^2} W'(s)$ and

$$K_1(x) = \mathcal{F}\left[\xi \mapsto \frac{1}{1 + \delta_t 4\pi^2 |\xi|^2}\right](x),$$

where $\mathcal{F}$ denotes the Fourier transform. This scheme is stable as soon as $\delta_t < \sup_{s\in[0,1]} |W''(s)|\varepsilon^2$. Moreover, the operator $S^{\mathrm{AC}}_{\delta_t,\varepsilon,1} : u \mapsto K_1 * \rho_1(u)$ that encodes the semi-implicit scheme can be viewed as an approximation of order 1 of the Allen-Cahn semigroup $S^{\mathrm{AC}}_{\delta_t,\varepsilon}$. This method has also the advantage of decoupling the action of the diffusion operator and the reaction operator, therefore each operator can be handled independently.

The previous stability constraint can be avoided using a convex-concave splitting of the Cahn-Hilliard energy. Following the idea introduced by Eyre [158], the functional $P_\varepsilon = E_1 + E_2$ is decomposed as the sum of a convex energy E_1 and a concave energy E_2 defined by

$$E_1(u) = \int_Q \left(\varepsilon \frac{|\nabla|^2}{2} + \frac{\alpha}{\varepsilon}\frac{u^2}{2}\right) dx \quad \text{and} \quad E_2(u) = \frac{1}{\varepsilon}\int_Q \left(W(u) - \alpha\frac{u^2}{2}\right) dx,$$

with α a sufficiently large stabilization parameter. Treating the convex energy implicitly and the concave energy explicitly yields the scheme

$$u^{n+1} = u^n - \frac{\delta_t}{\varepsilon}\left(\nabla E_1(u^{n+1}) + \nabla E_2(u^n)\right),$$

i.e.,

$$u^{n+1} = \left(I_d - \delta_t\left(\Delta - \frac{\alpha}{\varepsilon^2} I_d\right)\right)^{-1}\left(u^n - \frac{\delta_t}{\varepsilon^2}(W'(u^n) - \alpha u^n)\right) = K_2 * \rho_2(u^n). \tag{10.2.2}$$

Again, this numerical scheme is of the form

$$u^{n+1} = K_2 * \rho_2(u^n)$$

where the convolution operator K_2 is now given by

$$K_2(x) = \mathcal{F}\left[\xi \mapsto \frac{1}{1 + \delta_t\left(4\pi^2|\xi|^2 + \frac{\alpha}{\varepsilon^2}\right)}\right](x)$$

and the activation function ρ_2 satisfies $\rho_2(s) = s - \frac{\delta_t}{\varepsilon^2}(W'(s) - \alpha s)$.

The advantage of this scheme is to be unconditionally stable – in the sense that it decreases the Cahn-Hilliard energy – as soon as the stabilization parameter α satisfies $\alpha > \sup_{s\in[0,1]}|W''(s)| = 1$ (to be complete, note that choosing a large value for α has also a bad influence on the dynamics of the flow). It is also an approximation of order 1 to the Allen-Cahn semigroup $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon}$.

This stabilization effect leads us to believe that other such schemes could be obtained by directly learning the diffusion kernel K and the activation function ρ from the training data. The extension to networks is then straightforward since the latter operations can be interpreted as networks. Indeed, the schemes (10.2.1) and (10.2.2) are reminiscent of the structure of convolutional neural networks (CNN) [219, 245] : a convolution operation coupled with a nonlinear activation function, where the nonlinearity is given by ρ_1 (or ρ_2 for (10.2.2)) and the learning parameters are the parameters of the kernel associated with the convolution. A first idea of neural network would be to use a CNN with ρ_1 (for instance) as nonlinearity but it is restrictive because of the choice of ρ_1 that depends on the parameters δ_t and ε. We propose instead a neural network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ constructed as the composition of a pure convolution neuron and a multilayer perceptron (MLP) [292] that will act as a nonlinearity. The network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ we propose can be represented by the following diagram :

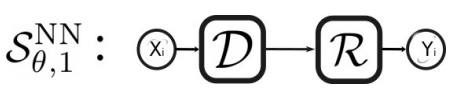


FIGURE 10.2 – Definition of $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ where $\mathcal{D}$ represents a convolution neuron (or diffusion neuron in reference to diffusion operators), and $\mathcal{R}$ represents a multilayer perceptron (called reaction network in reference to reaction operators). The parameter θ is the vector of learning parameters, it contains all weights of the $\mathcal{D}$ and $\mathcal{R}$ networks.

The choice of representing the reaction network by a MLP comes from the fact that the action of the reaction operator is totally encoded by a 1D function. According to the fundamental neural network approximation theorem [194, 195, 302] which states that any function can be approximated by a two-layer

MLP, it is quite natural to consider a MLP to represent the action of the reaction operator. However, we also choose to use a multilayer perceptron (MLP) as a non-linearity in order to keep some flexibility and not to depend on the context in which we work. The characteristics of the non-linearity are therefore entirely determined during the training stage.

Remark 10.2.1. *Note that in contrast to the splitting schemes* (10.2.2) *and* (10.2.1)*, we have chosen to define the network $\mathcal{S}^{NN}_{\theta,1}$ by starting with a diffusion neuron $\mathcal{D}$ followed by a non-linearity $\mathcal{R}$ to have a structure similar to the usual convolutional network. The other architecture where we start with $\mathcal{R}$ followed by $\mathcal{D}$ is also very legitimate and gives training results which are similar to those obtained with $\mathcal{S}^{NN}_{\theta,1}$. However, better numerical results are observed in the test phase when we use $\mathcal{S}^{NN}_{\theta,1}$ so we shall study only this network.*

Higher-order neural networks $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,3}$

The network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ provides us with the **basic block** to design more complex and deeper networks as is often the case in deep learning. One can mention, for instance, deep convolutional neural networks (where the basic block is the convolution operation) and deep residual neural networks (with the residual block). Many different networks can derive from $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$. In particular, in order to obtain more complex and efficient networks, we now try to adapt the structure of these networks from the higher order semi-implicit schemes developed in [342] in the case of a gradient flow structure. These schemes are structured as follows : starting from $U_0 = u^n$, we set $u^{n+1} = U_M$ where U_M is defined recursively from U_0 as

$$U_m = U_0 - \frac{\delta_t}{\varepsilon}\left(\sum_{i=0}^{m}\gamma_{i,m}\nabla E_1(U_i) + \sum_{i=0}^{m-1}\tilde{\gamma}_{i,m}\nabla E_2(U_i)\right) \text{ for } m = 1,\dots,M \tag{10.2.3}$$

where the parameters $\gamma_{i,m}$ and $\tilde{\gamma}_{i,m}$ are predefined to ensure the numerical scheme to be highly accurate, see [342] for more information about the choice of the coefficients $\gamma_{i,m}$ and $\tilde{\gamma}_{i,m}$. As can be observed in this scheme (10.2.3), the diffusion and reaction operators are applied in chain but also in parallel in order to keep the information of each U_i for $i = 1,\dots,M-1$.

The second-order network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ which is shown in Figure 10.3 is inspired by the scheme (10.2.3) with $M = 2$, which reads

$$U_2 = K_2 * [\rho_3(U_1) + [U_0 + \rho_2(U_1)],$$

with $U_1 = K_1 * [\rho_1(U_0)]$, and $\gamma_{0,1} = \gamma_{0,2} = \gamma_{0,1} = 0$. In Figure 10.3, we should be more precise about the meaning of the bottom part of the architecture, especially the curved line (with the symbol $\bigoplus$) which means that the network $\mathcal{R}$ is equipped with a residual structure. This indicates that each entry u to the network $\mathcal{R}$ is added to the output $\mathcal{R}(u)$. The full output of the bottom part of the architecture is then simply $u + \mathcal{R}(u)$. The curved line is also refereed as a skip connection in machine learning framework and correspond to skip one or several layers of the neural network as we did with all the layers of $\mathcal{R}$.

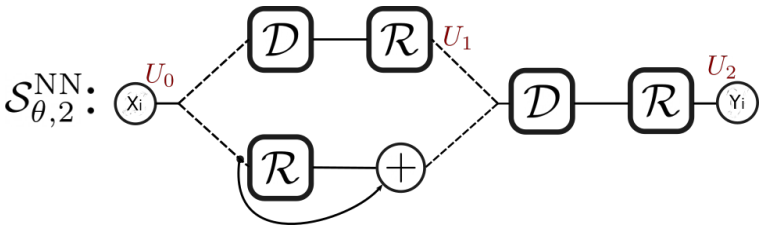


FIGURE 10.3 – Representation of the 2nd order neural network DR $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. We adopt the same formalism as for MLPs by assimilating the $\mathcal{D}$ and $\mathcal{R}$ networks to neurons. Here all the blocks $\mathcal{D}$ and $\mathcal{R}$ are different networks. The dotted lines mean that the output of the source neuron is multiplied by a weight. The curved line corresponds to a skip connection and means that the network $\mathcal{R}$ is equipped with a residual structure.

Remark 10.2.2. *The residual structure is also a consequence of different experiments we have done where networks with this type of structure had better learning scores and the learning process was faster. In the particular case of the network $\mathcal{S}^{NN}_{\theta,2}$ we can make the following conjecture : in the classical schemes* (10.2.1) *and* (10.2.2) *reaction operators are related to the double-well potential but also to the phase field profile. In the case of the network $\mathcal{S}^{NN}_{\theta,2}$, the residual structure allows to maintain this profile after each iteration. Indeed, several recent works [178] highlight the particularity of residual networks to stay close to the inputs by adding the identity.*

We can go further in the complexity and the depth of our networks by taking inspiration again from the structure of the scheme (10.2.3) with $M = 3$. For instance, following the principle of the previous network, we obtain the architecture shown in figure (10.4).

In this paper, we will limit ourselves to the first two networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ for the reason that they already give very satisfactory results and it appeared more relevant to us to focus on these first two structures.

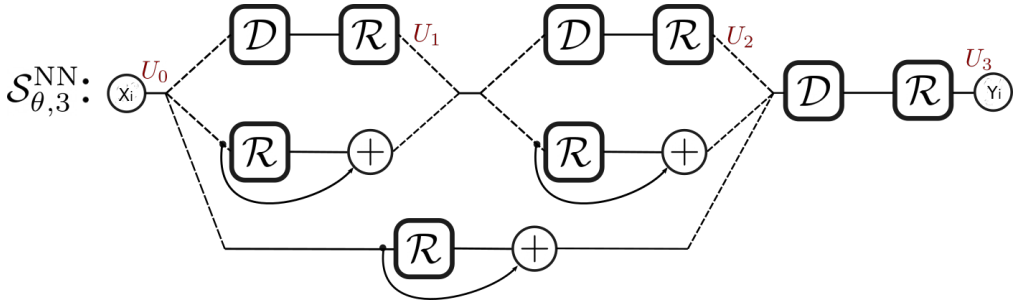


FIGURE 10.4 – Representation of the 3rd order neural network DR $\mathcal{S}^{\mathrm{NN}}_{\theta,3}$. The same idea used for the structure of the network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ is applied to design $\mathcal{S}^{\mathrm{NN}}_{\theta,3}$.

10.2.3 Discretization and specification of our neural networks

In practice, we will consider some approximations of phase field mean curvature flow defined on a square-box $Q = [0,1]^d$ using Cartesian grid with N nodes in each direction and using periodic boundary conditions. We consider the approximation parameter $\varepsilon = 2/N$ and the fixed timestep $\delta_t = \varepsilon^2$. Our neural networks are applied to discrete images $U^n_{i,j}$ which correspond to a sampling of the function $u^n(x_{i,j})$ at the points $x_{i,j} = ((i-1)/N, (j-1)/N)$. Our networks depend on the specific choice of the parameters $(\delta_x, \varepsilon, \delta_t)$ which have been fixed before the learning procedure. In particular, in all the numerical experiments presented below, we set $\delta_x = 1/N = 1/2^8$, $\varepsilon = 2\delta_x$ and $\delta_t = \varepsilon^2$.

Remark 10.2.3. *Our approach does not correspond exactly to a method based on neural operators since the definition of the convolutional kernel depends on the discretization in space. It will therefore not be possible to use the trained networks for other choices of discretization parameters. Nevertheless, it is possible to use the trained networks on different sizes of computational box as long as the value of the parameter δ_x remains unchanged. For instance, we can consider a square-box $Q = [0,2]^d$ discretized with $N = 2^9$ nodes in each direction. In this sense, our method remains resolution-invariant thanks to multi-scale techniques.*

Diffusional network based on a discrete kernel convolution.
As $\mathcal{D}$ corresponds to a convolution operation, one needs to specify its hyper-parameters, especially the kernel size N_K which is related to the domain discretization and the timestep δ_t. In practice, we consider a discrete square kernel $\mathbb{K}$ of size $N_K = 17$ where the kernel convolution reads as

$$\mathcal{D}[u^N]_{\mathbf{k}} = (\mathbb{K} * u^N)_{\boldsymbol{k}} = \sum_{\boldsymbol{\ell} \in L_N} \mathbb{K}_{\boldsymbol{\ell}}\, u^N_{\boldsymbol{k}-\boldsymbol{\ell}}$$

with $L_N = [-(N_K-1)/2, (N_K-1)/2]^d$. Here, we assume that the padding extension of u^N is periodic. Moreover, the convolution product is computed in practice using the Fast Fourier transform. Here, recall that the Fourier $\boldsymbol{K}$-approximation of a function u defined in a box $Q = [0,L]^d$ is given by

$$u^{\boldsymbol{N}}(x) = \sum_{\boldsymbol{k} \in K_N} c_{\boldsymbol{k}} e^{2i\pi \boldsymbol{\xi}_k \cdot x},$$

where $K_N = [-\frac{N}{2}, \frac{N}{2} - 1]^d$, $\boldsymbol{k} = (k_1, \ldots, k_d)$ and $\boldsymbol{\xi}_k = (k_1/L, \ldots, k_d/L)$. In this formula, the $c_{\boldsymbol{k}}$'s denote the K^d first discrete Fourier coefficients of u. The inverse discrete Fourier transform leads to $u^N_{\boldsymbol{k}} = \mathrm{IFFT}[c_{\boldsymbol{k}}]$ where $u^N_{\boldsymbol{k}}$ denotes the value of u at the points $x_{\boldsymbol{k}} = (k_1 h, \cdots, k_d h)$ and where $h = L/N$. Conversely, $c_{\boldsymbol{k}}$ can be computed as the discrete Fourier transform of $u^K_{\boldsymbol{k}}$, *i.e.*, $c_{\boldsymbol{k}} = \mathrm{FFT}[u^N_{\boldsymbol{k}}]$.

Remark 10.2.4. *In practice, it can be interesting to use kernels with sufficiently small size to facilitate their training but it is necessary to choose a size large enough to avoid anisotropy phenomena due to its square shape.*

Remark 10.2.5. *As we have seen previously, the characteristic size of $\mathbb{K}$ depends strongly on the choice of the time step δ_t. In our case, using the time step $\delta_t = \epsilon^2 = (2/N)^2$, as shown in Figure10.5, the support of the heat kernel K_{δ_t} is contained in the interval $[-8/N, 8/N]$ which suggests a good approximation by a discrete kernel of size $N_K = 17$. On the other hand, a time step four times larger $\delta_t = 4\epsilon^2$ would have required a kernel twice as large, i.e., $N_K = 33$.*

Reaction network based on a $1D$ multilayer perceptron
The reaction network is a multilayer perceptron seen as a real function. In practice, the number of hidden layers is set to 2 with 8 neurons on the first layer and 3 neurons on the second layer. More precisely, the idea is to apply the MLP independently to each component of u^N, i.e.

$$\mathcal{R}[U^N]_{\mathbf{k}} = \rho[2](W^{[3]}(\rho[2](W^{[2]}(\rho[1](W^{[1]}(u^N_{\mathbf{k}}) + b^{[1]})) + b^{[2]})) + b^{[3]}),$$

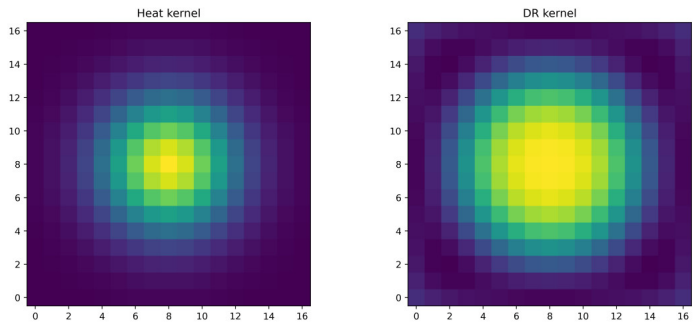


FIGURE 10.5 – Discrete kernel $\mathbb{K}$. Left : sampling of the heat kernel ; Right : example of learned kernel $\mathbb{K}$ for the network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ trained to approximate the semigroup $\mathcal{S}^{q}_{\varepsilon,\delta_t}$.

where $(W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}), W^{[3]}, b^{[3]})$ correspond to the parameters of the network to be optimized and $(\rho[1], \rho[2], \rho[3])$ represent the three activation functions. Thus, although the reaction neuron is represented as a 1D multilayer perceptron, when applied on an image, it is applied on each pixel of the image and allows to transform this image into an image of the same size.

We also select the appropriate activation function in the reaction network by testing different choices of activation functions (ReLU, ELU, Sigmoid, SiLU, Tanh, Gaussian) in our numerical experiments. In the end, the Gaussian model seems to have the best properties by presenting more efficient learning rates and speeds.

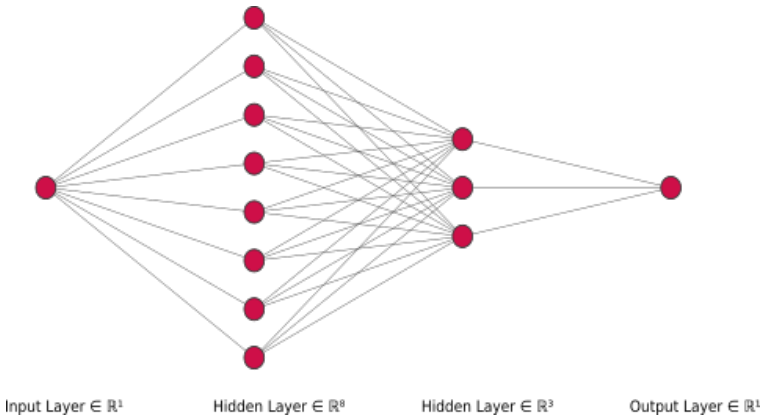


FIGURE 10.6 – Reaction network based on a $1D$ multilayer perceptron using two hidden layers with respectively 8 and 3 neurons

10.2.4 Neural network optimization, stopping criteria and Pytorch environment

We now turn to the question of how the learning parameters of our networks are estimated. The learning procedure consists in seeking an optimal vector parameter θ minimizing the energy J_1 (or $J_k,\ k > 1$ depending on the problem) using a mini-batch stochastic gradient descent algorithm with adaptive momentum [210, 59, 248] on a set of training data.

The reason we use a mini-batch approach is that our problem is a high-dimensional non-convex optimization problem. It is then often recommended to optimize on mini-batches rather than on the whole dataset to reduce the computational cost but also the memory cost. This strategy is also commonly used to escape saddle points of the non-convex energy J_1 (see [169] on this subject).

More precisely, the training strategy is as follows :

Step 1 Define the neural network $\mathcal{S}_\theta$ with the training vector parameter θ.

Step 2 Generate data using mini-batch : Randomly sample a mini-batch of size B of training labeled couples $(X_{i_k}, Y_{i_k}),\ k = 1, \cdots, B$ over the training dataset $\{(X_i, Y_i)\}_{i=1,\cdots,N_{\mathrm{train}}}$. This means that the training dataset is organized into N_{train}/B mini-batches and the parameter θ will be updated for each of these batches. The training **batch size** B is set to $B = 10$ over all experiments. One need to mention that the dataset is reshuffled after every epoch, so that the collection of mini-batches is different at every epoch.

Step 3 **Compute the loss** of the mini-batch $J_1(\theta) = \frac{1}{B}\sum_{k=1}^{B} \|\mathcal{S}_\theta[X_{i_k}] - Y_{i_k}\|^2$.

Step 4 **Compute the gradient** of the previous loss using **back-propagation**.

Step 5 **Update the learning parameter** θ using the stochastic gradient descent algorithm with adaptive momentum (**Adam** [210] in our case) and using with the gradient computed at the previous step.

Step 6 Repeat the steps 2 to 5 until all mini-batches have been used. Once all the mini batches have been used, an **epoch** is said to have passed. In practice, the epoch is set to $\text{epoch} = 20000$ for all experiments.

Step 7 Repeat steps 2 to 6 until a **stopping criterion** is satisfied.

Remark 10.2.6. *The number of learning parameters of each network is small enough with $N_\theta = 336$ for the first-order network $\mathcal{S}^{NN}_{\theta,1}$ and $N_\theta = 724$ for the second-order network $\mathcal{S}^{NN}_{\theta,2}$.*

Remark 10.2.7. *Very roughly speaking, stochastic gradient descent algorithms are parametrized by the learning rate, i.e. the step parameter just next to the gradient. The choice of the learning rate is a challenge. Indeed, a too small value may lead to a long training process that could get stuck, while a value that is too large may lead to a selection of sub-optimal learning parameters or an unstable learning process. In practice, schedulers are often used to gradually adjust the learning rate during training. In our case we use the Adam algorithm in step 5 which has the advantage of adjusting the learning rate according to the obtained value of the gradient in step 4.*

Remark 10.2.8. *About the loss function*

$$J_1(\theta) = \frac{1}{N_{train}}\sum_{i=1}^{N_{train}} \|\mathcal{S}^{\mathrm{NN}}_\theta[X_i] - Y_i\|^2 = \frac{1}{N_{train}}\sum_{i=1}^{N_{train}} \int_Q \left(\mathcal{S}^{\mathrm{NN}}_\theta[\varphi_{R_i}] - \varphi_{\sqrt{R_i^2-2\delta_t}}\right)^2 dx,$$

we consider its following discretized version

$$J_1^N(\theta) = \frac{1}{N_{train}}\delta_x^d \sum_{\mathbf{k}\in K_N} \left(\mathcal{S}^{\mathrm{NN}}_\theta[\varphi^N_{R_i}]_{\mathbf{k}} - [\varphi^N_{\sqrt{R_i^2-2\delta_t}}]_{\mathbf{k}}\right)^2,$$

the quadrature methods having all the same orders in periodic boundary conditions.

Stopping criterion

It may happen that after the training process the learning parameters are biased according to the training data. To prevent this and to avoid under- or over-fitting, it is recommended to add an intermediate step during the training procedure. This is called the validation step which consists in evaluating the model on new data (the so-called validation data) and then measuring the error made on these data using one or more metrics. In our case, the validation step is applied at regular intervals every 100 epochs. It is important to note that the parameters of the network are not updated during this procedure.

This step is used as a stopping criterion with the following validation metric based on the evolution of the volume of a sphere flowing under mean curvature. This second metric measures the error committed on the evolution of the volume of a given sphere. In some sense, this metric is also used to enforce the robustness of the neural network to predict the correct output over several iterations. In practice we select the learning parameter for which the validation metric gives the best score

To be more precise, the validation metric is given by

$$E_l(\theta) = \sum_{i=1}^{l}\sum_{n=0}^{n^i_{\max}} \left| \int_Q u_i^n - \int_Q \varphi_{\sqrt{R_i^2-2n\delta_t}} \right|,$$

where the sequence (u_i^n) is defined iteratively by

$$\begin{cases} u_i^{n+1} &= \mathcal{S}_\theta[u_i^n], \\ u_i^0 &= \varphi_{R_i}, \end{cases}$$

and $n^i_{\max} = \max\{n \in \mathbb{N},\ n\delta_t < T_{R_i}\}$ where $T_{R_i} = \frac{R_i^2}{2}$ is the extinction time of the mean curvature flow of the initial sphere with radius R_i for $i = 1, \dots, l$.

Note that this choice of metric is not meaningless and stems from the following observation : the evolution of the volume of a sphere with initial radius R_0 evolving under mean curvature flow is explicitly given by $V(t) = \pi R(t)^2 = \pi\left(R_0^2 - 2t\right)$.

Moreover, if $\varphi = q$, as $\varphi_R \approx 1_{B_R}$ one can show that

$$\int_Q \varphi_{R(t)} = \mathrm{Vol}(B_{R(t)}) + \mathcal{O}(\varepsilon^2) = V(t) + \mathcal{O}(\varepsilon^2).$$

In addition, it can also be shown with a similar argument that for $\varphi = -\frac{1}{\varepsilon}q'$,

$$\int_Q \varphi_{R(t)} = \mathcal{H}^{d-1}(\partial B_{R(t)}) + \mathcal{O}(\varepsilon^2).$$

In this way, one can see $t \mapsto \int_Q \varphi_{R(t)}$ as a good measure of the evolution of the volume (or the perimeter when $\varphi = -\frac{1}{\varepsilon}q'$) of a sphere with initial radius R_0 that evolves by mean curvature flow. In this sense, the energy E_l allows to preserve some stability over time (at least on circles or spheres).

All these networks, datasets, visualization tools, etc., have been developed in a Python module dedicated to machine learning for the approximation of interfaces that evolve according to a geometric law (as the mean curvature flow) and are represented by a phase field. This module uses the machine learning framework PyTorch [277] and the high-level interface PyTorch Lightning [159] as the machine learning back-end. Source code will be made available on line beside a forthcoming description.

Training computation has been made on a computational server hosted at the Camille Jordan Institute and using a general-purpose graphical processing unit Nvidia V100S. Typical training (20000 epochs) used in the following numerical experiments requires less than 30 minutes of computation without the validation steps.

10.3 Validation

In this first numerical section, we test the ability of two neural networks to learn either the mean curvature flow $t \mapsto \Omega(t)$ or oriented domains or the mean curvature flow $t \mapsto \Gamma(t)$ of a possible non-orientable sets. These two neural networks are :

— The 1st-order DR network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ which combines successively a diffusion neuron and a reaction network.

— The 2nd-order network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, which uses two diffusion neurons and three reaction networks, according to the architecture illustrated on Figure 10.3.

More specifically, in each case we will propose a methodology to learn the action of the semigroups $\mathcal{S}^{q}_{\delta_t,\varepsilon}$ and $S^{q'}_{\delta_t,\varepsilon}$ where the model parameters are set to $\varepsilon = 2\delta_x$ and $\delta_t = \varepsilon^2$.

First, the learning of the two networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ allows us to obtain sufficiently accurate and efficient approximations of the semigroup $\mathcal{S}_{\delta_t,\varepsilon,q}$ to get an approximation of the mean curvature flow of oriented domains whose quality is at least comparable with what would be obtained by solving the Allen-Cahn equation. Going further, a numerical error analysis in the particular case of the evolution of a circle shows that both networks make it possible to reduce the phase field approximation error related to the convergence of the Allen-Cahn equation to the motion by mean curvature in $O(\varepsilon^2)$.

In the case of the mean curvature flow of possibly non-orientable sets, the first order DR network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ fails to learn a sufficiently accurate approximation of the semigroup $\mathcal{S}^{q'}_{\delta_t,\varepsilon}$ and reveals that the basic block architecture DR is not well adapted to this specific setting. On the other hand, the structure of the second order DR network seems much more suitable and presents an interesting approximation of the semigroup $\mathcal{S}^{q'}_{\delta_t,\varepsilon}$ allowing to obtain at least qualitatively approximation of evolutions of interface under mean curvature flow even in the case of non-orientable Γ interface with multiple junctions.

10.3.1 Oriented mean curvature flow $t \mapsto \partial\Omega(t)$ and approximation of $S^{q}_{\delta_t,\varepsilon}$

We now present some preliminary results about the approximation of the semigroup $S^{q}_{\delta_t=\varepsilon^2,\varepsilon}$ based on the training of both neural networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ on circles evolving by mean curvature flow and using exact oriented phase field representation. We first plot on Figure 10.7 the evolution of the training loss energy throughout the optimization process, in blue for the network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and in orange for the second one $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. Clearly, we observe that we manage to learn an approximation of $S^{q}_{\delta_t=\varepsilon^2,\varepsilon}$ with both networks. However, the second order DR network $\mathrm{NN}_{\theta,2}$ reaches more quickly acceptable errors on the learning loss, which suggests that the structure of this network is better suited to our problem.

We now want to check that both trained networks achieve accurate approximations of the mean curvature flow. To do so, we test the two associated numerical schemes $u^{n+1} = \mathcal{S}^{\mathrm{NN}}_{\theta,\alpha}[u^n]$ with $\alpha \in \{1, 2\}$, coupled

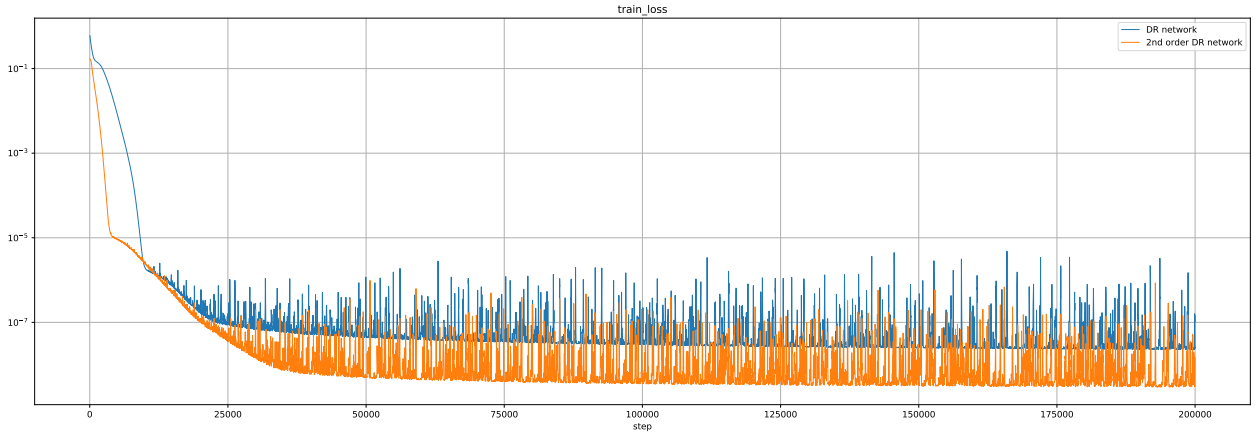


FIGURE 10.7 – Learning procedure via optimization of the error J_1 over training data. Training losses of $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ plotted in blue and orange, respectively.

with an initial condition u^0 given by $u^0 = q(d(\Omega_0, \cdot)/\varepsilon)$ where Ω_0 corresponds to a disk of radius $R_0 = 0.3$.

More precisely, we plot in each line of Figure 10.8 the numerical approximation u^n obtained at different times $t_n = n\delta_t$ using respectively the classical Lie splitting scheme $\mathcal{S}^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$, the first order network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and the second order one $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. All three approximations of the mean curvature flow are so similar that it is difficult to observe any difference between them. At least on this particular example, the networks provide qualitatively the expected evolution.

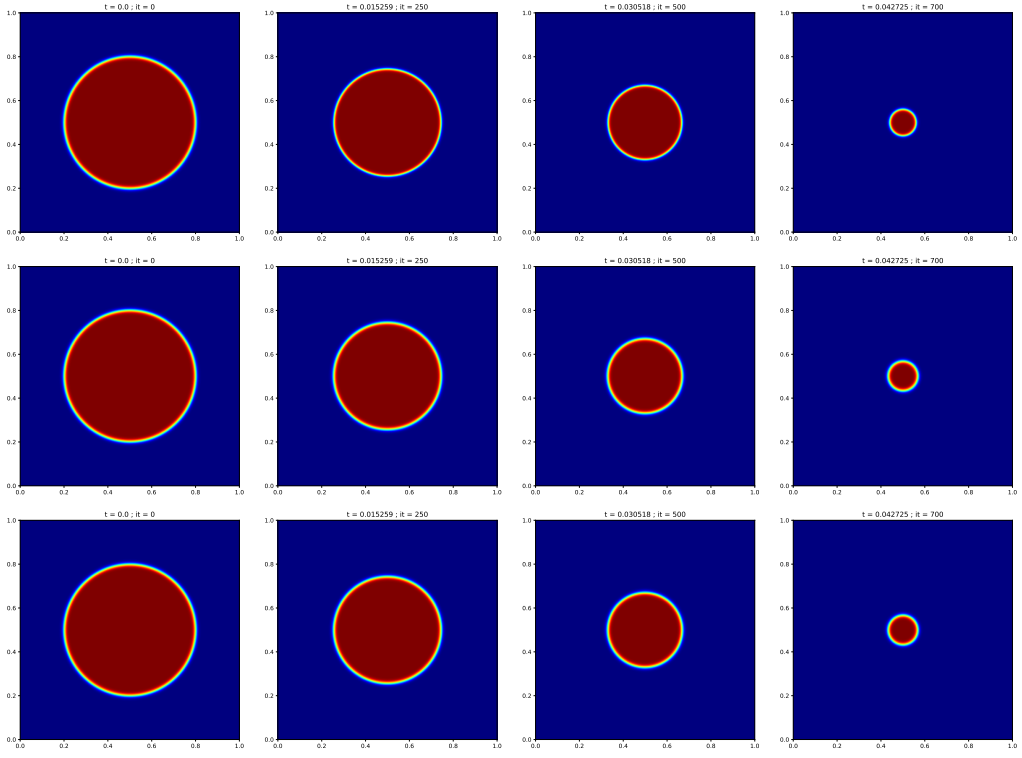


FIGURE 10.8 – Comparison of the numerical semigroups $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$, $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ for the approximation of the mean curvature flow of a circle; each row corresponds to the evolution obtained at different time t using respectively the numerical semigroup $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$ and the networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$.

In order to have more quantitative error estimates, we display on Figure 10.9 the error on the integral of the phase field function $\varphi_{R(t)}$ of a circle of radius $R(t)$ during the iterations knowing that the evolution by mean curvature of an initial disk of radius R_0 is a disk of radius $R(t) = \sqrt{R_0^2 - 2t}$, at least until the extinction time $t_0 = \frac{1}{2}R_0^2$. More precisely, we plot the error

$$n \mapsto \left| \int_Q u^n(x)dx - \int_Q \varphi_{\sqrt{R_0^2 - 2n\delta_t}}(x)dx \right|$$

using

— the trained network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ in blue
— the trained network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ in orange
— the Lie splitting scheme $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$ with $\delta_t = \epsilon^2$ in green
— the Lie splitting scheme $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$ with $\delta_t = \epsilon^2/10$ in red

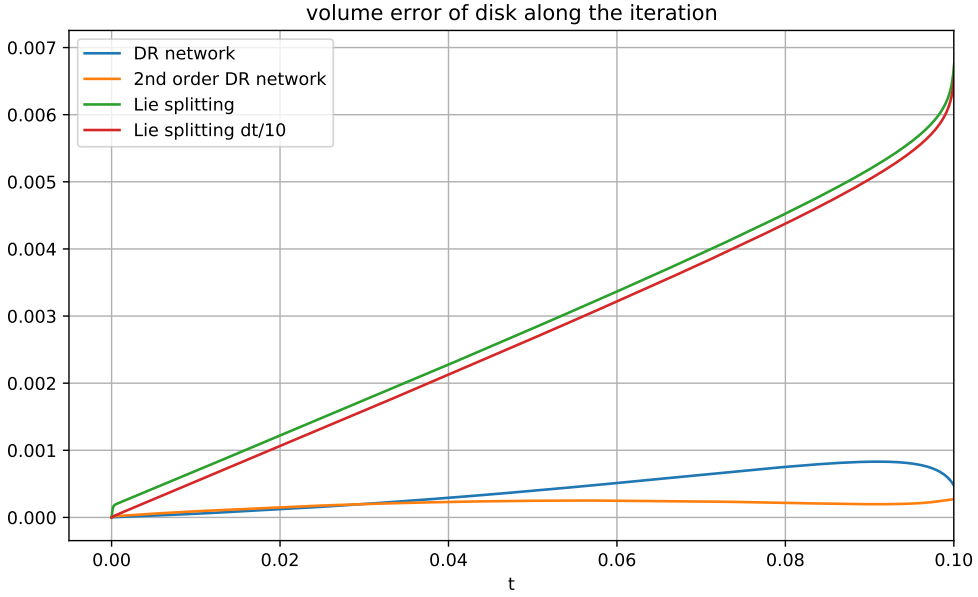


FIGURE 10.9 – Comparison of different schemes on the evolution of the radius of a circle evolving by mean curvature flow. Plots in blue, orange, green and red correspond to the error $n \mapsto \left(\int_Q u^n(x)dx - \int_Q \varphi_{\sqrt{R_0^2-2n\delta_t}}(x)dx\right)^2$ along iterations of respectively the trained networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, and the schemes $\mathcal{S}^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ and $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$ with $\delta_t = \epsilon^2/10$.

These results clearly show that both our neural networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ provide more accurate numerical schemes than the classical discretization of the Allen-Cahn equation, even using a smaller time step δ_t.

This first validation test shows also that the training of our networks manages to correct both the errors of discretization of the Allen-Cahn equation, and the modeling errors of the approximation by the Allen-Cahn equation of the mean curvature flow.

We also propose to repeat the experiment but considering a starting radius R_0 which does not belong to the interval $[0.05, 0.45]$ in order to evaluate the neural networks to generalize its learning towards unknown curves. For this, we now use a computational domain 2 times larger $Q = [0, 2]^2$ in order to consider an initial radius $R_0 = 0.75$ without reaching the boundary of Q. We specify here that we use the same networks as before and no new learning is necessary. As the parameters δ_x, ε and δ_t are fixed, the evolution obtained with the first network $\mathcal{S}^{NN}_{\theta,1}$ presented in the figure 10.10 presents a diffuse interface size equivalent to the previous one but which appears visually 2 times on each image. We also present on the figure 10.11 the error on the evolution of the radius $R(t)$ for each of the methods and as previously the second order network $S^{NN}_{2,\theta}$ appears as the most accurate method, and clearly shows that the network manages to find the evolution by mean curvature flow for curvatures even when the curves that come into play are not present in the learning base.

We also plot on Figure 10.12 a last numerical comparison in the case of a non-convex initial set $\Omega(0)$ and as before, we can clearly observe that each flow is qualitatively very similar.

In conclusion, the use of neural networks not only allows us to obtain good approximations of the mean curvature motion but these approximations are also of better quality than what would be obtained by a phase field approach and a solution of the Allen-Cahn equation.

These results are therefore very encouraging and show even more the interest of neural networks for phase field approximations in the even worser case where the models would not be as accurate as the Allen-Cahn equation or the numerical schemes would not be as efficient and accurate.

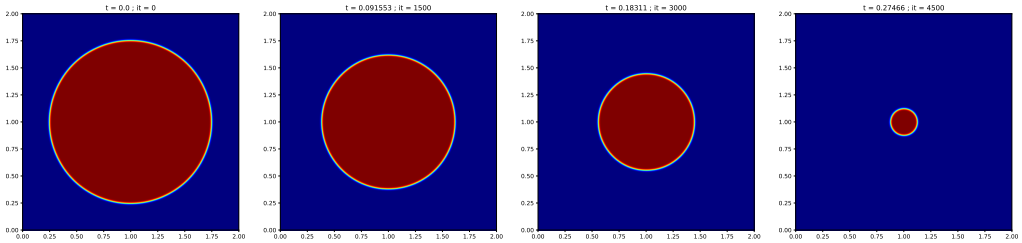


FIGURE 10.10 – Evolution obtained at different time t using the networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ using a domain Q twice as large and an initial circle of radius $R_0 = 0.75$.

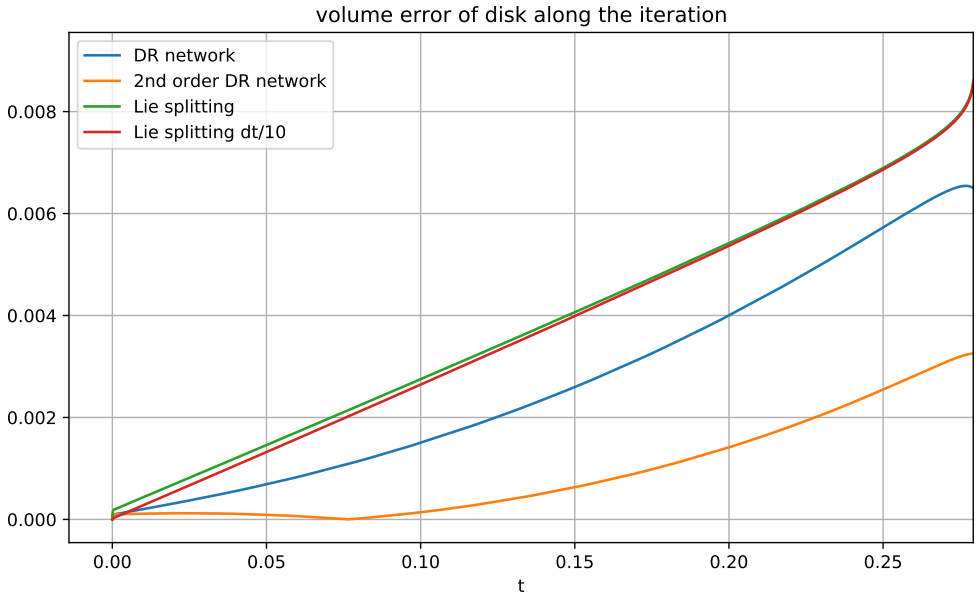


FIGURE 10.11 – Comparison of different schemes on the evolution of the radius of a circle evolving by mean curvature flow. Plots in blue, orange, green and red correspond to the error $n \mapsto \left(\int_Q u^n(x)dx - \int_Q \varphi_{\sqrt{R_0^2-2n\delta_t}}(x)dx\right)^2$ along iterations of respectively the trained networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, and the schemes $\mathcal{S}^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ and $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$ with $\delta_t = \epsilon^2/10$.

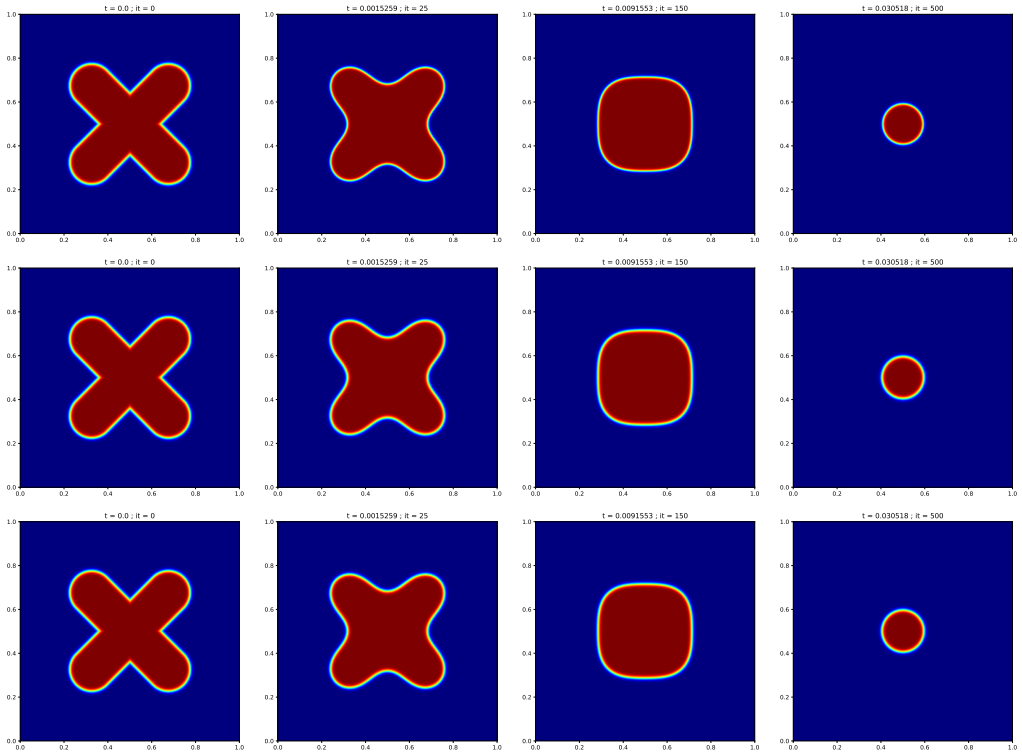


FIGURE 10.12 – Comparison of numerical semigroups $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$, $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ to approximate the mean curvature flow of a non convex initial set ; each line corresponds respectively to the evolution obtained at different time t by using respectively the numerical semigroups $\mathcal{S}^{\mathrm{AC}}_{\delta_t,\varepsilon,1}$, $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$.

10.3.2 Non orientable mean curvature flow $t \mapsto \Gamma(t)$ and approximation of $S^{q'}_{\delta_t,\varepsilon}$

Let us describe some numerical results on the approximation of the $S^{q'}_{\delta_t,\varepsilon}$ semigroup, still using the same architecture for the two neural networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ but with a learning on data built from evolutions of circles in the exact non-oriented phase field representation. Let us recall here that to the limit of our knowledge, there is no phase field model allowing to approximate such a flow by solving an Allen-Cahn-type PDE. As before, the idea is to train both our networks on a database still made of circles evolution on a δ_t time step, but using the q' profile instead of q.

We first plot on Figure 10.13 the evolution of the training loss energy during the learning process, respectively in blue and orange for the networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. Here, both networks seem to succeed in learning the flow, although the train loss value seems much better for the second network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$.

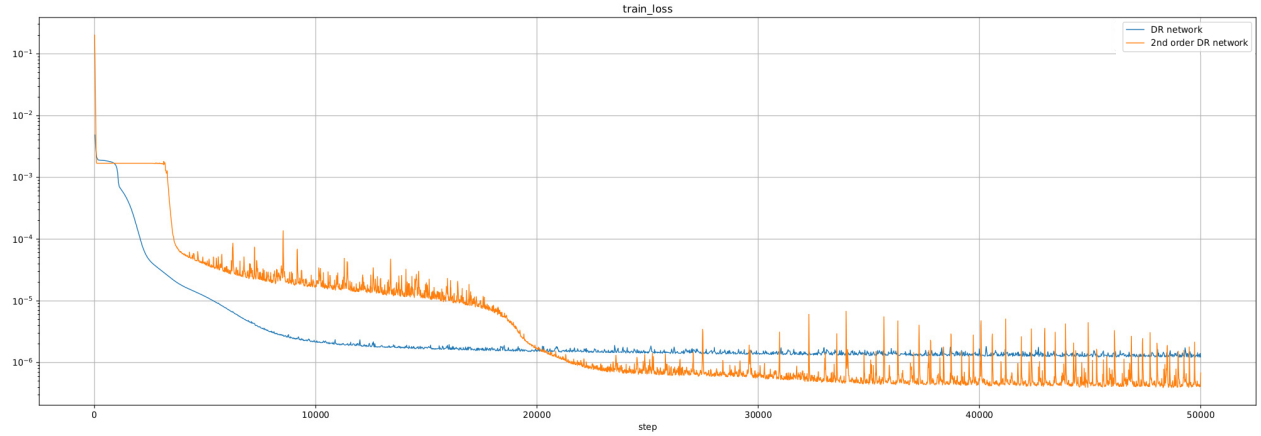


FIGURE 10.13 – Learning process by optimizing the training losses J_k with $k = 5$ with respect to the networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, respectively plotted in blue and orange.

In contrast, when we test both networks to approximate the motion by the mean curvature of a circle, we clearly observe on Figure 10.14 that the first order network $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ does not manage to keep the profile q' in a

stable way and to decrease the radius of the circle. The good news is that the second network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ leads to a numerical scheme that is stable enough to reproduce well the evolution of the circle while keeping the q' profile along the evolution. In order to have a more quantitative criterion on the evolution of the circle, we draw on Figure 10.15 the evolution $n \mapsto \pi(\frac{1}{2\pi\varepsilon}\int u^n(x)dx)^2$ which should correspond to the evolution of the area of a circle of radius $R(t)$ given by $t \mapsto \pi(R_0^2 - 2t)$. This is indeed clearly the case on Figure 10.15 which shows that the obtained evolution law corresponds well to the motion by mean curvature.

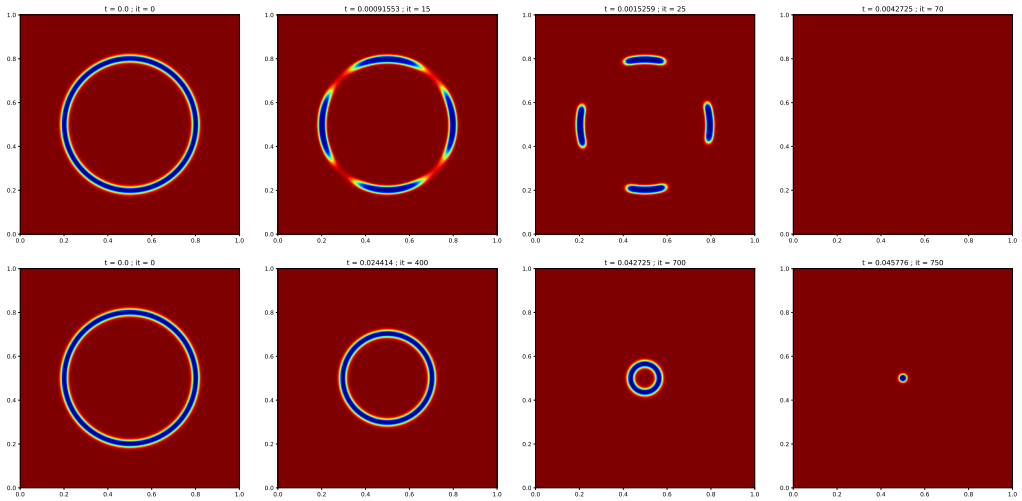


FIGURE 10.14 – Numerical comparison of networks $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ to approximate the mean curvature flow of a circle in the non oriented setting; each row corresponds to the evolution obtained at different times using $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, respectively.

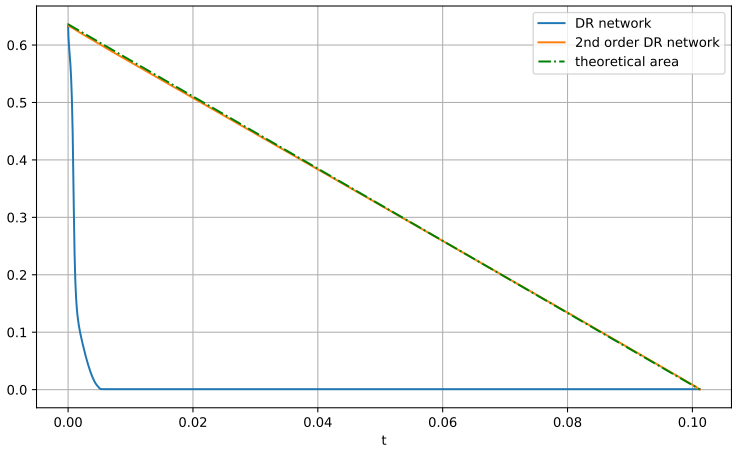


FIGURE 10.15 – Comparison of different schemes on the evolution of the radius of a circle evolving by mean curvature flow. Plots in blue, orange correspond to $n \mapsto \pi(\frac{1}{2\pi\varepsilon}\int u^n(x)dx)^2$ along iterations using respectively $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ and $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. In green, the theoretical evolution of the circle area $n \mapsto \pi(R_0^2 - 2n\delta_t)$.

In order to test the reliability of the network $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ to approximate motions by mean curvature in a general way, we also compare the evolution obtained with the scheme deriving from our network with the one deriving from the discretization of the Allen Cahn equation.

Here, although the starting set $\Omega(0)$ is the same in both numerical experiments, the initial phase field solution u^0 is different and depends on the profile used. We then plot on the figure 10.16 the solution u^n computed at different times t_n using the two different numerical schemes. In particular, we can clearly observe a similar flow that validates the methodology and the use of our neural networks to approximate the motion by mean curvature in the case of a non-orientable set.

We also plot on the figure 10.17 the numerical approximation of the mean curvature flow obtained in the case of non-orientable initial sets. In each of these experiments, we observe the evolution of points with triple junction which seem to satisfy the Herring condition [67]. This result is all the more surprising since our learning base contained only circle evolutions and no interface with the triple point. The presence of stable triple points is therefore very good news and shows the potential of this approach for the general case of mean curvature motion.

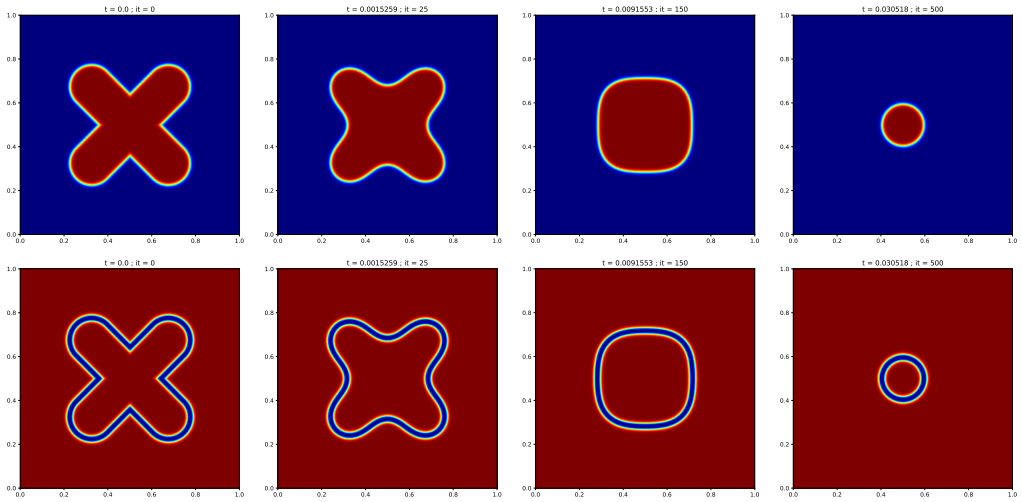


FIGURE 10.16 – Approximation of the mean curvature flow of a non convex initial set : numerical comparison of the classical discretisation of the Allen Cahn equation $\mathcal{S}^{\text{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ (first line) with the network $\mathcal{S}^{\text{NN}}_{\theta,2}$ trained on the non oriented phase field $q'(dist(x,\Gamma)/\varepsilon)$ (second line). In each line, we plot the solution u^n at different times t_n.

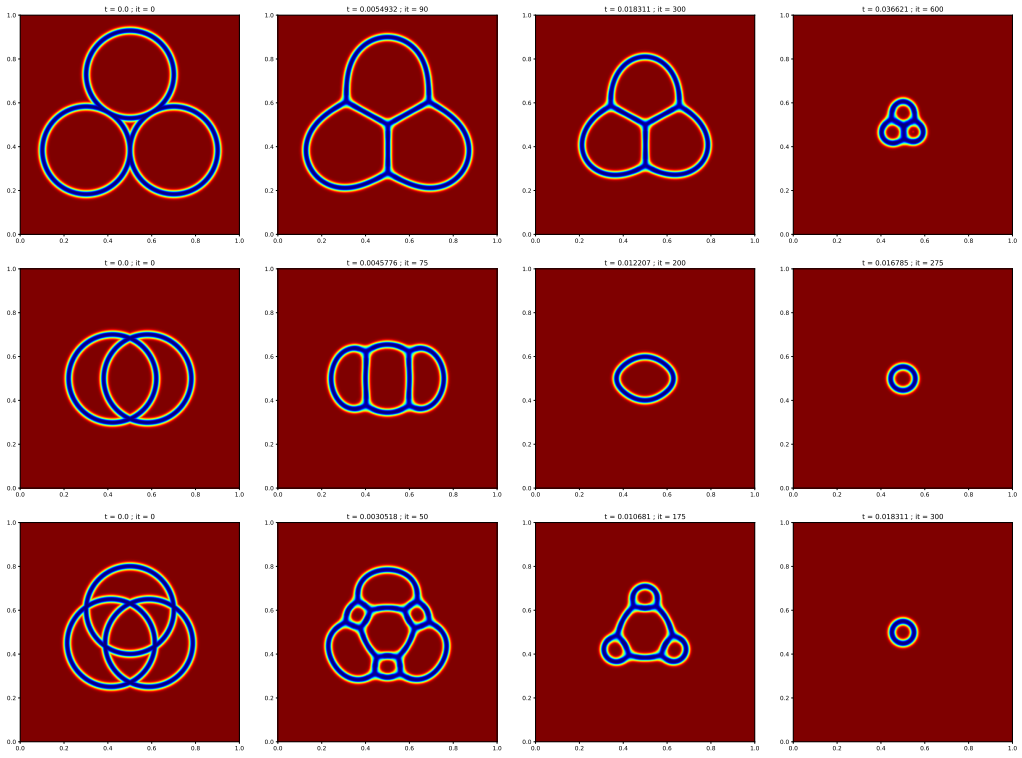


FIGURE 10.17 – Approximation of the mean curvature flow of a non orientable initial set. Each line corresponds to a different choice of the initial set $\Gamma(0)$. We display the solution u^n at different times t_n along the iterations.

We give a last example in 10.18 of an approximation of mean curvature flow in the case of an initial non-closed interface. Here the example is quite pathological since it is very difficult to make sense of the motion by mean curvature, and we expect the interface to disappear in infinitesimal time. Surprisingly, the evolution seems relatively stable with a speed of the end points of the order of $1/\varepsilon$.

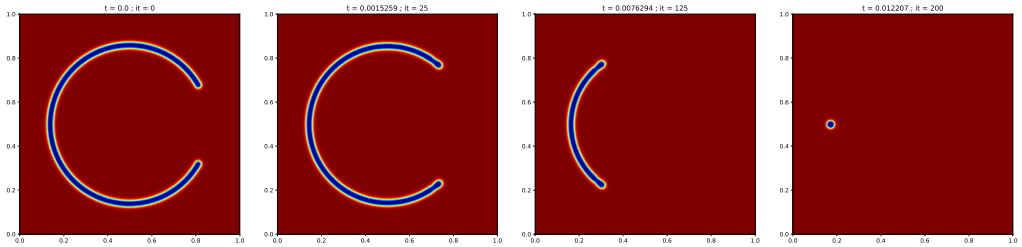


FIGURE 10.18 – Approximation of the motion by mean curvature of an initial open curve.

10.4 Applications : multiphase mean curvature flows, Steiner trees, and minimal surfaces

We highlight in this section how the previous schemes derived from our trained networks $S^{NN}_{\theta,\alpha}$, $\alpha \in \{1,2\}$ are sufficiently stable to be coupled with additional constraints such as volume conservation ($\int_Q u^n dx =$ const) or inclusion constraints (reformulated as an inequality $u^n \geq u_{in}(x)$, see [68]), and sufficiently stable to be extended to the multiphase case.

Instead of retraining the networks for these applications, we simply reuse the networks trained earlier on circles and spheres, and couple their predictions with an additional constraint in a post-processing step to approximate the solution. Thus, we demonstrate the adaptability of our neural network approach.

Partition problem in dimension 2**.** The first application concerns the approximation of multiphase mean curvature flow with or without volume conservation. We recall that the evolution of a partition with N phases $\mathbf{\Omega(t)} = (\Omega_1(t), \Omega_2(t), \cdots \Omega_N(t))$ can be obtained as the L^2-gradient flow of a multiphase perimeter

$$P(\mathbf{\Omega}) = \frac{1}{2}\sum_{i=1}^{N} P(\Omega_i) = \frac{1}{2}\sum_{i=1}^{N} \int_{\partial\Omega_i} d\mathcal{H}^1,$$

with the normal velocity $V_{i,j}$ of each interface $\Gamma_{i,j} = \partial\Omega_i \cap \partial\Omega_j$ satisfying

$$V_{i,j} = H_{i,j},$$

where $H_{i,j}$ represents the mean curvature at the interface $\Gamma_{i,j}$.

The Steiner problem in dimension 2**.** The second application is the approximation of Steiner trees in dimension 2, i.e. solutions of the Steiner problem. Recall that the Steiner problem consists, given a collection of points $a_1, \ldots, a_L \in \Omega$, in finding a compact connected set $K \subset \Omega$ containing all the a_i's and having minimal length. In other words, it is equivalent to finding the optimal solution to the following problem

$$\min\left\{\mathcal{H}^1(K),\ K \subset \Omega, K \text{ connected}, a_i \in K,\ \forall i = 1, \cdots, L\right\},$$

where $\mathcal{H}^1(K)$ corresponds to the one-dimensional Hausdorff measure of K. As the optimal set K needs not be orientable, our idea is to combine the network $S^{NN}_{\theta,2}$ trained on the non-oriented phase field database with the inclusion constraints associated to all points a_i's.

The Plateau problem in dimension 3**.** The last application focuses on the Plateau problem in dimension 3 and co-dimension 1. Recall that it consists in finding, for a given closed boundary curve Γ, a compact set E in Ω with minimal area and whose boundary coincides with Γ [115]. In other words, it amounts to solving the minimization problem

$$\min\{\mathcal{H}^2(E); E \subset \Omega, \text{connected and such as } \partial E = \Gamma\}, \tag{10.4.1}$$

, where $\mathcal{H}^2(E)$ stands for the two-dimensional Hausdorff measure of E. Analogously to the method proposed for the Steiner problem, the approximation method we propose consists in

1. training a $S^{NN}_{\theta,2}$ network on databases built from $3D$ spheres evolving under mean curvature flow with a non-oriented phase field representation,
2. coupling the trained network $S^{NN}_{\theta,2}$ with the inclusion constraint to force the boundary constraint $\partial E = \Gamma$.

10.4.1 Evolution of a partition in dimension 2

As explained previously, the motivation in this first application is to approximate multiphase mean curvature flows. Recall that the phase field approach consists in general [167, 166, 164, 272, 71] in introducing a multiphase field function $\mathbf{u} = (u_1, u_2, \cdots, u_N)$, solution of an Allen Cahn system obtained as the L^2-gradient flow of the multiphase Cahn-Hilliard energy P_ε defined by

$$P_\varepsilon(\mathbf{u}) = \begin{cases} \frac{1}{2}\sum_{k=1}^N \int_Q \frac{\varepsilon}{2}|\nabla u_k|^2 + \frac{1}{\varepsilon}W(u_k)dx & \text{if } \sum_{k=1}^N u_k = 1, \\ +\infty & \text{otherwise,} \end{cases}$$

More precisely, the Allen-Cahn system [75, 67] reads as

$$\partial_t u_k = \Delta u_k - \frac{1}{\varepsilon^2}W'(u_k) + \lambda\sqrt{2W(u_k)}, \quad k = 1, \cdots, N$$

where the Lagrange multiplier λ is associated with the partition constraint $\sum_{k=1}^L u_k = 1$. As explained in [75, 67], this PDE can be for instance computed in two steps :

1. Solve the decoupled Allen-Cahn system

$$u_k^{n+1/2} = \mathcal{S}^{\text{AC}}_{\delta_t,\varepsilon,1}[u_k^n], \quad k = 1, \cdots, N.$$

2. Project onto the partition constraint $\sum_k u_k^{n+1} = 1$:

$$u_k^{n+1} = u_k^{n+1/2} + \lambda^{n+1}\sqrt{2W(u_k^{n+1/2})}, \quad k = 1, \cdots, N.$$

Notice that the Lagrange multiplier λ^{n+1} is computed following the expression

$$\lambda^{n+1} = \frac{1 - \sum_{k=1}^N u_k^{n+1/2}}{\sum_{k=1}^N \sqrt{2W(u_k^{n+1/2})}},$$

in order to satisfy exactly the partition constraint.

This discretization scheme associated with the Allen-Cahn system suggests to simply replace the Lie splitting Allen-Cahn operator $\mathcal{S}^{\text{AC}}_{\delta_t,\varepsilon,1}$ by our network $\mathcal{S}^{NN}_{\theta,1}$ trained on oriented mean curvature flow in section (10.3.1). A numerical experiment obtained with such a strategy is presented on the first row of Figure 10.19. More precisely, we consider here an evolution of a multiphase system with four phases and we observe at least qualitatively that the flow seems to be correct with a triple junction forming angles of $2\pi/3$. The second row of Figure 10.19 gives a similar numerical experiment with additional constraint on the volume of each phase. Here, following the approach developed in [165, 267, 65, 75], the idea is to consider the Allen-Cahn system with volume conservation

$$\partial_t u_k = \Delta u_k - \frac{1}{\varepsilon^2}W'(u_k) + \lambda\sqrt{2W(u_k)} + \mu_k\sqrt{2W(u_k)},$$

where the new Lagrange multiplier μ_k is defined to satisfy $\text{Vol}_k = \int_Q u_k = \int_Q u_0$. The previous scheme can then be modified by considering now a projection on the partition and the volume constraints as

$$u_k^{n+1} = u_k^{n+1/2} + (\lambda^{n+1} + \mu_k^{n+1})\sqrt{2W(u_k^{n+1/2})},$$

where the Lagrange multipliers $(\lambda^{n+1}, \mu_1^{n+1}, \mu_2^{n+1}, \cdots \mu_N^{n+1})$ are for instance given by

$$\mu_k^{n+1} = \frac{\text{Vol}_k - \int_Q u_k^{n+1/2}}{\int_Q \sqrt{2W(u_k^{n+1/2})}}, \text{ and } \lambda^{n+1} = \frac{1 - \sum_{k=1}^N \left[u_k^{n+1/2} + \mu_k^{n+1}\sqrt{2W(u_k^{n+1})}\right]}{\sum \sqrt{2W(u_k^{n+1/2})}},$$

which allows each of the constraints to be satisfied. As previously, the idea consists simply to replace the Lie splitting Allen-Cahn operator $\mathcal{S}^{\text{AC}}_{\delta_t,\varepsilon_1}$ by our network $\mathcal{S}^{NN}_{\theta,1}$. In particular, the numerical experiment plotted on the second row of Figure 10.19 shows a stable multiphase evolution where the volume of each phase seems to be conserved. These two numerical examples clearly show that our networks previously trained on mean curvature motion flow can be re-exploited in more complex situations.

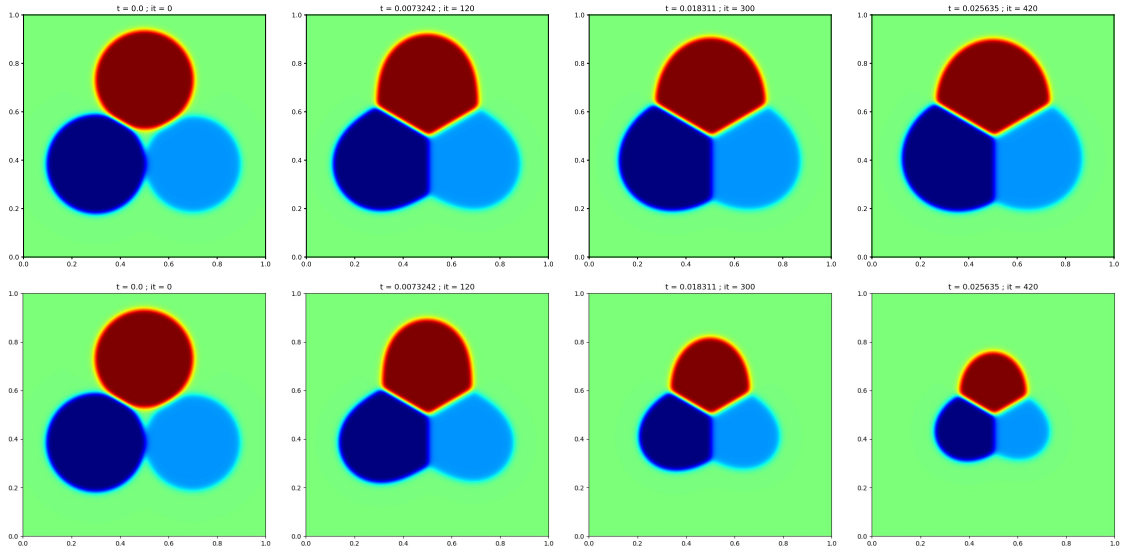


FIGURE 10.19 – Mean curvature flows of three interconnecting circles without (first row) and with (second row) additional volume constraint.

10.4.2 Approximation of Steiner trees in $2D$

Variational phase field models have been recently introduced [58, 57, 53, 54, 90] to tackle the Steiner problem. For example, the model introduced in [58] proposes the Ambrosio-Tortorelli functional

$$F_\varepsilon(u) = \int_\Omega \varepsilon |\nabla u|^2 + \frac{1}{\varepsilon}(1-u)^2 dx,$$

coupled with an additional penalization term $G_\varepsilon(u)$ which forces the connectedness of the set K and which is defined by

$$G_\varepsilon(u) = \frac{1}{\varepsilon}\sum_{i=1}^{N} \mathbf{D}(u^2; a_0, a_i), \text{ with } \mathbf{D}(w; a, b) := \inf_{\Gamma: a \rightsquigarrow b} \int_\Gamma w \, d\mathcal{H}^1 \in [0, +\infty],$$

where the notation $\Gamma : a \rightsquigarrow b$ means that Γ is a rectifiable curve in Ω connecting a and b.

Intuitively, the phase field function u_ε is expected to be of the form $u_\varepsilon = \varphi(dist(x, K)/\varepsilon)$, the Ambrosio-Torterelli term approximates the length of K, and the geodesic term G_ε forces the solution u_ε to vanish on K, and K to connect the different points a_i. An analysis of this model and, more precisely, a Γ-convergence result established in [58] suggest that the minimization of this phase field model should give an approximation of a Steiner solution, i.e. a Steiner tree. However, numerical experiments as proposed in [58] and in [57] show the ability of this coupling to approximate solutions of the Steiner problem in dimensions 2 and 3 but show also all the difficulties to minimize efficiently the geodesic term G_ε to preserve the connectedness of the set K. The conclusions are quite similar for the approaches developed in [53, 54, 90] where the idea is rather to use the measure-theoretic notion of current and the connectedness of the set K is ensured by adding a divergence constraint of the form $div(\tau) = \sum \alpha_i \delta_{a_i}$.

We propose a completely different approach by considering a non oriented mean curvature flow $t \mapsto \Gamma(t)$ of an initial connected set $\Gamma(0)$ containing all the points a_i coupled with the additional inclusion constraint $\{a_i,\ i = 1, \cdots, N\} \subset \Gamma(t)$. Indeed, we expect that the stationary state of such an evolution is at least a local minimum of the Steiner problem. From a phase field point of view, the strategy is to consider the non-oriented phase field representation $q'(dis(x, \Gamma(t)/\varepsilon))$, to use the non oriented trained network $\mathcal{S}^{NN}_{\theta,2}$ to let the interface evolve by mean curvature, and to incorporate the inclusion constraint by using an additional inequality constraint

$$u \leq u_{\text{in}}(x) = \sum_{i=1}^{N} q'(dist(a_i, x)/\varepsilon),$$

in the spirit of [74, 68]. Finally, the scheme reads as follows :

1. Approximation of a non-oriented mean curvature flow step

$$u^{n+1/2} = \mathcal{S}^{\text{NN}}_{\theta,2}[u^n].$$

2. Projection on the inclusion constraint $u \leq u_{\text{in}}(x)$:

$$u_k^{n+1} = \min(u_{\text{in}}, u^{n+1/2}).$$

We present in Figure 10.20 three numerical experiments using respectively 4, 5 and 6 points a_i arranged non-uniformly on a circle. The first picture of each line corresponds to the initial set $\Gamma(0)$ constructed in such a way to connect the point a_1 in a linear way to all the other points $\{a_i\}_{i\in\{2\cdots N\}}$. We then clearly observe an evolution of the set $\Gamma(t)$ which seems to converge to a Steiner tree connecting all the points a_i. In particular, the triple points seem to be handled accurately enough. In the end, the method seems extremely effective, simple to implement, and the computation of Steiner solutions is really fast compared to other methods proposed in the literature. These results illustrate that our network trained on non-oriented mean curvature flows has sufficient numerical stability properties to be coupled with inclusion constraints.

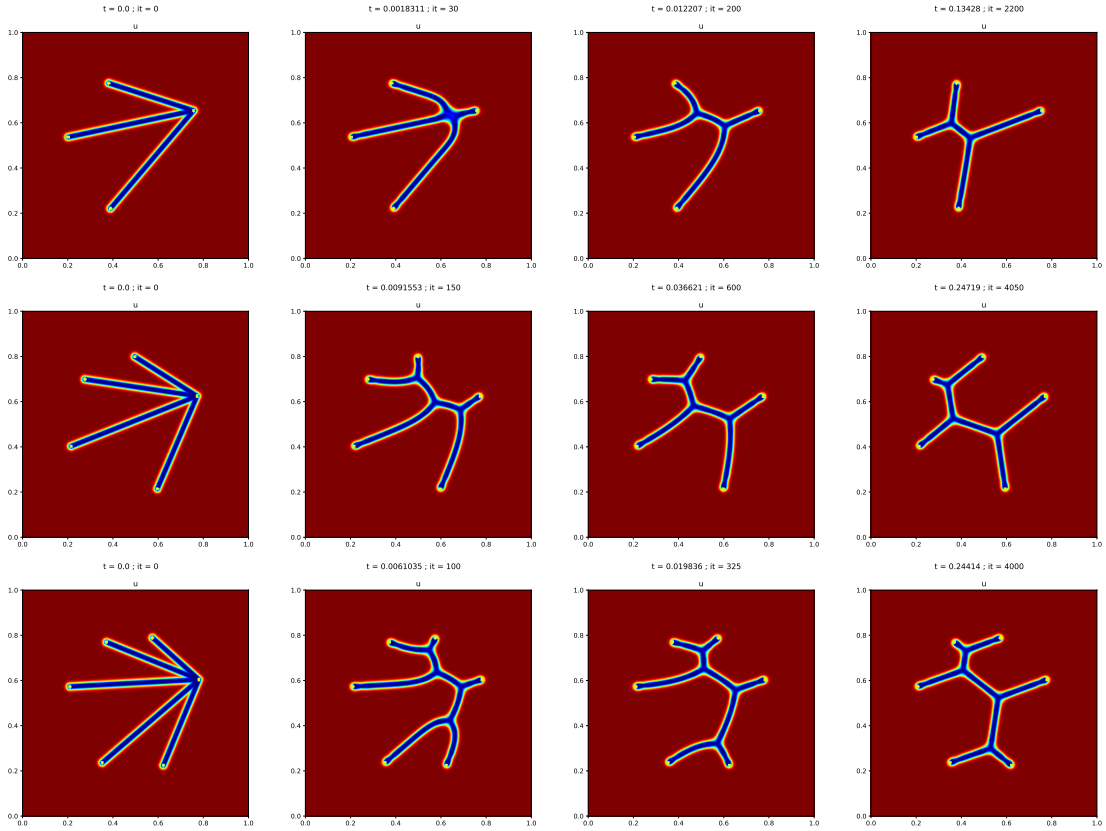


FIGURE 10.20 – Approximation of Steiner trees in $2D$ using a non oriented mean curvature flow coupled with inclusion constraints according to the scheme (10.4.2) ; Each line presents an evolution of the numerical solution u^n along the iterations with, respectively, 4, 5, and 6 points a_i.

10.4.3 Approximation of minimal surfaces in $3D$

The Plateau problem was formulated by Lagrange in 1760 and consists in showing the existence of a minimal surface in $\mathbb{R}^3$ with prescribed boundary Σ. Existence and regularity of solutions have been studied in different contexts, see for instance [125] for smooth and orientable solutions and [289] for existence and uniqueness of soap films including orientable and non orientable surfaces and possibly multiple junctions.

From the numerical point of view, there exist many methods to compute minimal surfaces, as for instance the first one [124] and others [330, 107] which use a parametric representation of the surface. In the same spirit, the papers [131, 132] exploit a finite element method to compute numerical approximations of minimal surfaces. Finally, note also some numerical approaches using an implicit representation of the minimal surface in [100, 88] where a level set method is used. As for phase field approaches, a current-based method was proposed and analyzed in [89], and provides numerical approximations in the case of oriented surfaces.

We propose in this section to extend to $3D$ the phase field approach used for the Steiner problem. The idea is very similar in the sense that

1. We train a new network $\mathcal{S}_{\theta,2}^{NN}$ to approximate the semigroup $S_{\delta_t,\varepsilon,q'}$ in dimension 3. Here we consider the case of the mean curvature flow of co-dimension 1. The training of the network is performed on a database consisting of spheres involving under motion by mean curvature which is also explicit in this case.
2. We apply the previous scheme where the function u_{in} is now defined by

$$u_{\text{in}}(x) = \sum_{i=1}^{N} q'(dist(\Sigma, x)/\varepsilon).$$

We present in Figures 10.21 and 10.22 four numerical experiments using respectively the boundary of a Mobius strip, a trefunknot, a union of rings, and a pit. The prescribed boundary Σ and the boundary of $\{x \in Q; u_n(x) \geq -0.2\}$ are respectively plotted in red and green along the iterations. We display in Figure 10.21 the evolution of an interface along the iterations towards the stationary numerical solution. We show in Figure 10.22 only the stationary solutions, i.e. the minimal surfaces associated with the respective prescribed boundaries.

These numerical results are very encouraging because they are examples of non-orientable solutions with more or less complex topologies and possibly with triple line singularities.

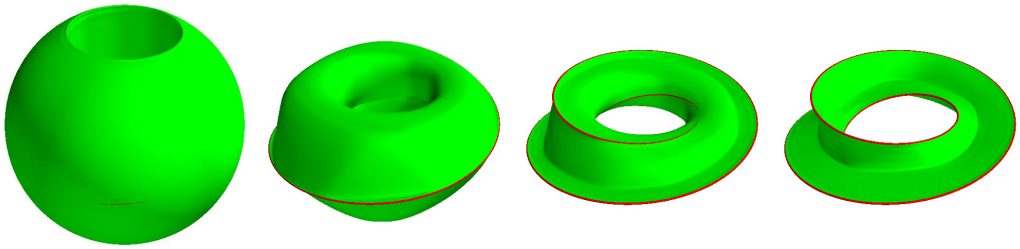


FIGURE 10.21 – Approximation of a minimal surface using a non-oriented mean curvature flow coupled with an inclusion constraint. The images represent the numerical solution u^n at four different times. The prescribed boundary Σ and the boundary of $\{x \in Q; u_n(x) \geq -0.2\}$ are plotted in red and green, respectively .

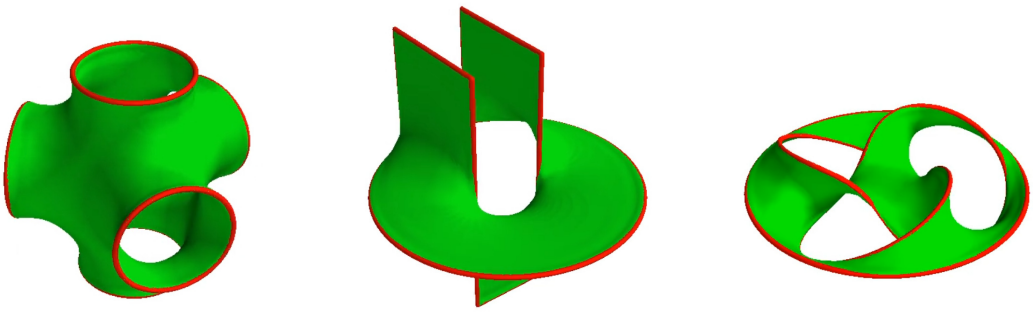


FIGURE 10.22 – Approximation of minimal surfaces using a non-oriented mean curvature flow coupled with an inclusion constraint ; Each picture shows the numerical stationary solution u^n in green and the prescribed boundary Σ in red.

10.5 Conclusion

We have shown in this work how neural networks allow to obtain simple and very efficient numerical schemes to approximate motions by mean curvature in the classical oriented case but also in the non-oriented case for which phase field approaches are not suitable. The use of network structure based on numerical discretization schemes of the Allen-Cahn equation (alternating linear diffusion operators with local nonlinear reaction operators) allowed to obtain very efficient networks despite their low number of degrees of freedom (less than 1000) compared to standard networks such as U-net which easily reach more than millions of parameters to optimize. The advantage is then to facilitate the learning of such network while gaining in robustness, the best illustration of which is their ability to generalize mean curvature flow despite a very limited database. These first results are very encouraging and it is quite reasonable to expect

to adapt our approach to many interface geometric flows such as isotropic mean curvature motion, surface diffusion or Willmore flow. But beyond the application to the phase field model, the approach seems quite generic and would make it possible to deal with any kind of diffusion reaction equation whose applications are very numerous.

Chapitre 11

Flot de Willmore et réseaux de neurones

Dans cette partie, on s'intéresse à la simulation numérique du flot de Willmore qui s'obtient en considérant le flot-gradient L^2 de l'énergie de Willmore [213],

$$\mathcal{W}(\Omega) = \int_{\partial\Omega} H^2 d\sigma.$$

Plus précisément, il correspond au mouvement géométrique dont la vitesse normale à l'interface $\partial\Omega$ est donnée par

$$V = \Delta_{\partial\Omega} H - \frac{1}{2}H^3 + H\|A\|^2,$$

où $\Delta_{\partial\Omega}$ et H désignent respectivement l'opérateur de Laplace Beltrami sur $\partial\Omega$ et la courbure moyenne de $\partial\Omega$. Le terme $\|A\|^2 = \sum_{i=1}^{d-1} \kappa_i^2$ correspond au carré de la norme de Frobenius de la seconde forme fondamentale A de $\partial\Omega$ et les κ_i sont les courbures principales de $\partial\Omega$.

L'étude de ce flot est assez délicate du fait de sa nature fortement non linéaire. L'existence et la régularité des solutions ont été étudiées par exemple dans [133, 214, 213] où les auteurs montrent en particulier l'existence en temps long des courbes simples en dimension 2 et la convergence vers une sphère pour des formes initiales dont l'énergie de Willmore est faible. Au contraire, pour des formes initiales dont l'énergie est élevée, des singularités peuvent apparaître au cours de l'évolution [133].

11.1 Méthodes de champ de phase

11.1.1 Approximation de l'énergie de Willmore

De nombreux modèles d'approximation champ de phase de l'énergie de Willmore ont été proposés dans la littérature. Parmi ceux-ci, on trouve :

— L'approximation de Bellettini [31], en dimension $d \geq 2$

$$\mathcal{W}_\varepsilon^{\text{Be}}(u) = \frac{1}{2}\int_{Q\setminus\{|\nabla u|=0\}} (|\operatorname{div}\frac{\nabla u}{|\nabla u|}|^2)(\frac{\varepsilon}{2}|\nabla u|^2 + \frac{W(u)}{\varepsilon})dx$$

— L'approximation de Esedoglu-Rätz-Röger [148] en dimension $d \geq 2$

$$\begin{aligned}\mathcal{W}_\varepsilon^{\text{EsRäRö}}(u) &= \frac{1}{2\varepsilon}\int_Q \left(\varepsilon\Delta u - \frac{W'(u)}{\varepsilon}\right)^2 dx \\ &+ \frac{1}{2\varepsilon^{1+\alpha}}\int_Q (\varepsilon\Delta u - \frac{W'(u)}{\varepsilon} - (\varepsilon|\nabla u|\sqrt{2W(u)})^{\frac{1}{2}}\operatorname{div}\frac{\nabla u}{|\nabla u|})^2 dx.\end{aligned}$$

et les approximations qui nous intéressent plus particulièrement dans ce manuscrit :

— L'approximation classique de De Giorgi-Bellettini-Paolini [40, 319, 38, 260, 290, 264] en dimension $d \geq 2$

$$\mathcal{W}_\varepsilon(u) = \frac{1}{2\varepsilon}\int_Q \left(\varepsilon\Delta u - \frac{W'(u)}{\varepsilon}\right)^2 dx$$

— L'approximation de Mugnai [39] en dimension $d = 2$

$$\mathcal{W}_\varepsilon^{\text{Mu}}(u) = \frac{1}{2\varepsilon}\int_\Omega \left|\varepsilon\nabla^2 u - \frac{W'(u)}{\varepsilon}\frac{\nabla u}{|\nabla u|}\otimes\frac{\nabla u}{|\nabla u|}\right|^2 dx$$

Ces deux dernières approximations nous serviront de références et de modèles de comparaison dans nos expériences numériques. Nous renvoyons le lecteur intéressé à l'article [54] et ses références pour une comparaison des différents modèles cités ci-dessus.

La Γ-convergence de $\mathcal{W}_\varepsilon$, lorsque $\varepsilon \to 0^+$, est plus compliquée à établir que dans le cas de l'énergie de Cahn-Hilliard $\mathcal{P}_\varepsilon$. Elle a été étudiée dans un premier temps par Bellettini et Paolini dans [40] où ils démontrent la propriété de $\Gamma - \limsup$ dans le cas régulier. La question de la $\Gamma - \liminf$ a ensuite été montrée en dimension 2 et 3 par Röger et Schätzle [290] et indépendamment par Nagasa et Tonegawa [264] en dimension 2. Ainsi, en dimensions 2 et 3, nous avons plus précisément la Γ-convergence de $\mathcal{P}_\varepsilon + \mathcal{W}_\varepsilon$ vers $\lambda(\mathcal{P} + \mathcal{W})$ pour les ensembles à bord de classe C^2, où $\mathcal{P}$ désigne le périmètre et λ une constante dépendant du potentiel W. En dimension strictement supérieure à 3, la Γ-convergence reste un problème ouvert.

Le résultat précédent répond partiellement à la conjecture énoncée par De Giorgi [170]. On rappelle en effet que cette conjecture est reliée à la Γ-convergence de $\mathcal{W}_\varepsilon$ et notamment à l'identification de sa limite dans le cas singulier. Cependant, dans ce contexte, l'énergie de Willmore n'est pas semi-continue pour la topologie L^1. Il est donc naturel de se demander si la Γ-convergence de $\mathcal{W}_\varepsilon$ peut s'étendre à la relaxée de Willmore notée $\overline{\mathcal{W}}$ i.e à son enveloppe semi-continue inférieurement pour la topologie L^1,

$$\overline{\mathcal{W}}(\Omega) = \inf\left\{\liminf \mathcal{W}(\Omega_h),\ \partial\Omega_h \in C^2,\ \Omega_h \xrightarrow{L^1} \Omega\right\}.$$

La relaxée est explicite en dimension 2 [35, 37], et en particulier, on peut montrer qu'elle n'est finie que pour les ensembles rectifiables. En dimension supérieure, elle est un peu moins explicite et nous renvoyons par exemple aux références [15, 247] pour étude de cette énergie dans ce cas.

Dans le cas singulier, l'existence de solutions singulières de l'équation d'Allen-Cahn stationnaire [113] montre que la Γ-limite de $\mathcal{W}_\varepsilon$ ne peut pas toujours être identifiée à la relaxée L^1 de $\mathcal{W}$. En effet, pour chacune des solutions singulières illustrées en figure 11.1, la ligne de niveau $1/2$ (que l'on note E) admet des croisements. Par conséquent, nous savons que la relaxée de E est infinie et il est possible de montrer de plus que $\Gamma - \lim \mathcal{W}_\varepsilon(\mathbb{1}_E) = 0 < +\infty$. Nous renvoyons le lecteur intéressé aux références [54, 35, 113] pour d'autres exemples d'ensembles pour lesquels la Γ-convergence vers la relaxée est invalidée.

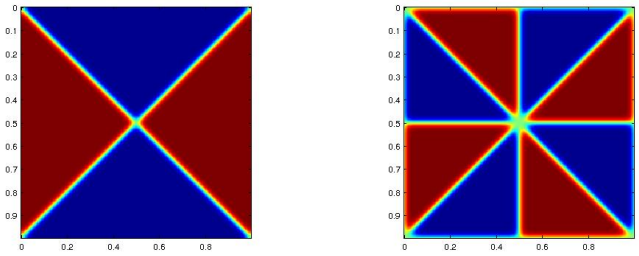


FIGURE 11.1 – Exemples de solutions singulières u_ε de l'équation d'Allen-Cahn stationnaire.

Contrairement à l'énergie classique, l'approximation de Mugnai Γ-converge en dimension 2 vers la relaxée L^1 de Willmore pour tout ensemble de périmètre fini [261]. On peut d'ailleurs montrer que ce résultat est aussi valide pour certaines des autres énergies présentées en début de section [54].

11.1.2 Modèles d'approximation champ de phase du flot de Willmore

Il est intéressant de comparer le flot de gradient L^2 de l'énergie classique

$$\begin{cases} \varepsilon^2 \partial_t u = \Delta\mu - \frac{1}{\varepsilon^2} W''(u)\mu, \\ \mu = W'(u) - \varepsilon^2 \Delta u, \end{cases} \tag{11.1.1}$$

à celui de l'approximation de Mugnai

$$\begin{cases} \varepsilon^2 \partial_t u = \Delta\mu - \frac{1}{\varepsilon^2} W''(u)\mu + W'(u)\mathcal{B}(u), \\ \mu = W'(u) - \varepsilon^2 \Delta u, \end{cases} \tag{11.1.2}$$

où la différence tient essentiellement dans la présence du terme $W'(u)\mathcal{B}(u)$ où

$$\mathcal{B}(u) = \operatorname{div}\left(\operatorname{div}\left(\frac{\nabla u}{|\nabla u|}\right)\frac{\nabla u}{|\nabla u|}\right) - \operatorname{div}\left(\nabla\left(\frac{\nabla u}{|\nabla u|}\right)\frac{\nabla u}{|\nabla u|}\right).$$

Le caractère bien posé du modèle classique (11.1.1) a été étudié dans [104] avec une contrainte supplémentaire de volume et dans [105] avec la contrainte de volume et d'aire. En se basant sur les idées développées dans [104, 105], Zwilling [351] a montré l'existence et l'unicité de solutions faibles de l'équation (11.1.1) lorsque des conditions périodiques aux bords sont imposées. Concernant le modèle de Mugnai (11.1.2), la question reste encore ouverte.

L'analyse asymptotique [54, 235, 160] de ces deux modèles montre que la vitesse normale à l'interface du flot limite est donnée par

$$V = \Delta_\Gamma H - \frac{1}{2}H^3 + H\|A\|^2,$$

pour le modèle classique et

$$V = \Delta_\Gamma H + \sum_{i=1}^{d-1} \kappa_i^3 - \frac{1}{2}H\|A\|^2,$$

pour celui de Mugnai.

Un calcul permet alors de montrer que ces deux flots coïncident en dimensions 2 et 3. Du point de vue de la modélisation, cela montre qu'ils ont des comportements similaires au moins dans un cadre régulier.

La figure 11.2 donne un exemple d'évolution obtenue à partir du modèle classique où la condition initiale est un parallélépipède avec 2 trous. Nous observons que l'interface converge vers une surface de Lawson-Kusner de genre 2, surface qui est conjecturée être un minimiseur de l'énergie de Willmore parmi les surfaces de genre 2 [212, 198].

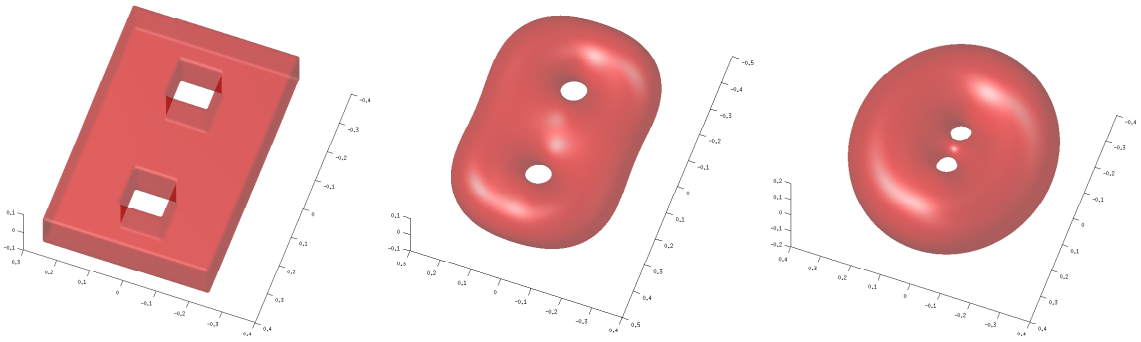


FIGURE 11.2 – Exemple de flot de Willmore simulé à partir du modèle classique en dimension 3.

En présence de singularités, les deux modèles agissent très différemment. Le terme $\mathcal{B}(u)$ agissant uniquement lorsque deux interfaces deviennent trop proches l'une de l'autre, le modèle de Mugnai a tendance à repousser les interfaces qui se rapprochent alors que le modèle classique autorise au contraire les croisements d'interfaces comme dans la figure 11.1. La figure 11.3 illustre cette différence : les deux cercles disjoints grandissent au cours du flot mais tandis qu'ils finissent par s'intersecter pour former un huit sous l'effet du modèle classique, ils vont au contraire se repousser sous l'effet du modèle de Mugnai.

En figure 11.4 nous donnons un autre exemple d'approximation du flot de Willmore obtenu à partir du modèle classique. Là encore, on observe que les intersections sont maintenues tout au long de l'évolution.

Comme dans le cas des équations de type Cahn-Hilliard, le traitement numérique des modèles classique (11.1.1) et de Mugnai (11.1.2) pose un certain nombre de difficultés numériques et nécessite des schémas très précis en espace afin d'obtenir des approximations cohérentes d'évolutions d'interfaces selon le flot de Willmore. On trouve plusieurs approches dans la littérature. Des schémas semi-implicites comprenant des différences finies, des éléments finis et des méthodes spectrales de Fourier ont par exemple été élaborés dans [127, 128, 148] avec une analyse de ces schémas dans [129]. Une approche basée sur une méthode de point fixe a aussi été proposée dans [54].

Il est aussi possible d'opter pour une approche de splitting convexe-concave afin d'obtenir des schémas inconditionnellement stables. Dans le cas du modèle classique par exemple, l'idée consiste à utiliser le

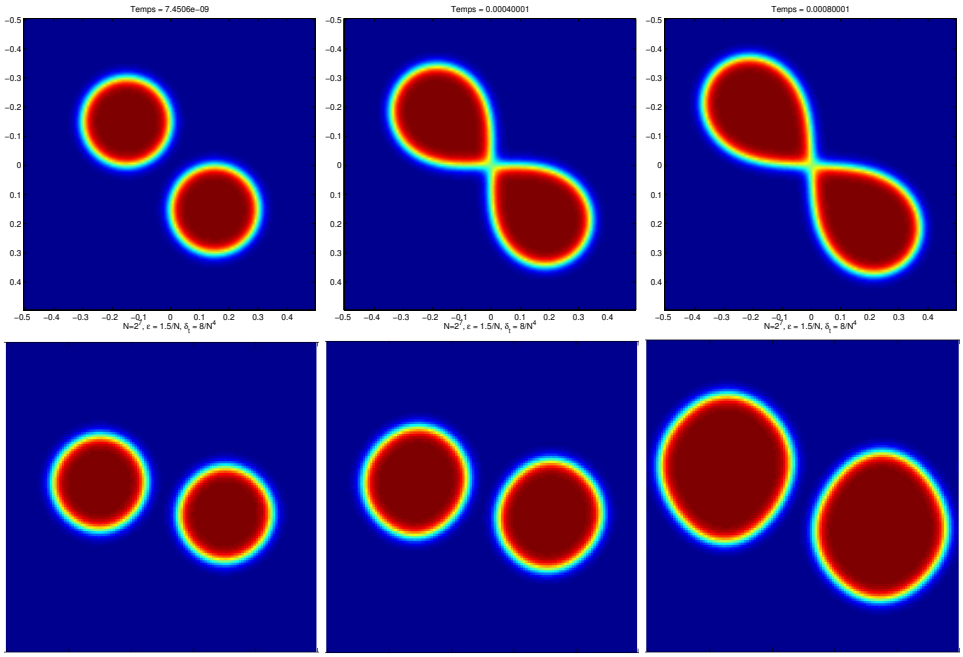


FIGURE 11.3 – Exemple de flot de Willmore simulé à partir du modèle classique (première ligne) et du modèle de Mugnai (deuxième ligne).

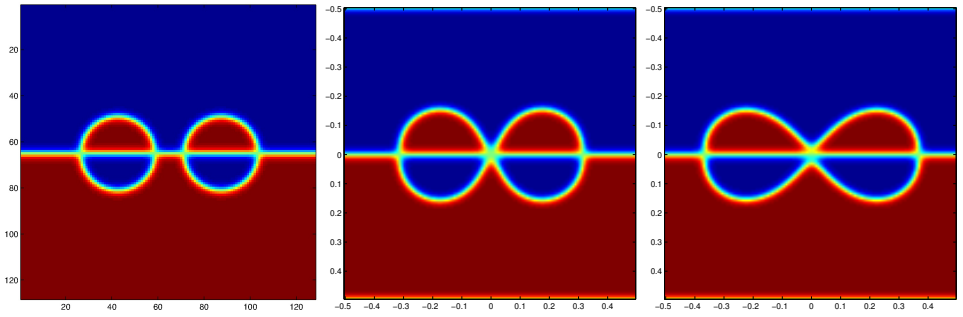


FIGURE 11.4 – Exemple de flot de Willmore simulé à partir du modèle classique. Là encore on observe des croisements d'interfaces qui persistent au cours de l'évolution.

splitting suivant

$$\begin{aligned}\mathcal{W}_\varepsilon(u) &= \frac{1}{2\varepsilon}\int_Q \left(\varepsilon\Delta u - \frac{1}{\varepsilon}W'(u)\right)^2 dx = \frac{1}{2}\int_Q \alpha(\Delta u)^2 + \beta|\nabla u|^2 + \gamma u^2 dx \\ &\quad + \frac{1}{2}\int_Q \frac{1}{\varepsilon}\left(\varepsilon\Delta u - \frac{1}{\varepsilon}W'(u)\right)^2 - (\alpha(\Delta u)^2 + \beta|\nabla u|^2 + \gamma u^2)dx.\end{aligned}$$

Sous des hypothèses de régularité sur u, il est possible de montrer que le dernier terme associé au traitement explicite est bien concave dès lors que les constantes α, β et γ sont choisies suffisamment grandes.

Une alternative à l'approche précédente est d'interpréter l'équation classique

$$\begin{cases}\varepsilon^2\partial_t u = \Delta\mu - \frac{1}{\varepsilon^2}W''(u)\mu \\ \mu = W'(u) - \varepsilon^2\Delta u,\end{cases}$$

comme le flot de l'énergie de Cahn-Hilliard par rapport à la métrique

$$J_u(\mu) = \int_Q \frac{1}{2}|\nabla\mu|^2 + \frac{1}{2\epsilon^2}W''(u)\mu^2 dx.$$

L'idée est alors de réutiliser les approches développées précédemment pour les équations de Cahn-Hilliard afin d'obtenir des schémas numériques très simples et efficaces.

Dans chaque cas, ces discrétisations numériques nous amènent à appliquer successivement plusieurs opérateurs de diffusion et de réaction. Comme dans le cas du flot de courbure moyenne, cela ouvre la voie à l'introduction de structures de réseaux adaptés pour l'approximation du flot de Willmore.

11.2 Algorithmes de type Bence–Merriman–Oher

Les approches précédentes peuvent être comparées aux méthodes de type convolution-seuillage qui semblent beaucoup plus simples à mettre en œuvre [150] et dont un exemple emblématique est la méthode de Bence-Merriman-Osher (BMO). On rappelle que l'algorithme de BMO permet de simuler le mouvement par courbure moyenne en alternant une étape de convolution et une étape de seuillage, ce qui conduit au schéma numérique suivant : en partant d'un ensemble Ω_n, l'itération suivante est donnée par

$$\Omega_{n+1} = \left\{G_{\delta_t} * \mathbb{1}_{\Omega_n} \geq \frac{1}{2}\right\},$$

où $G_{\delta_t}(x) = \dfrac{1}{(4\pi\delta_t)^{d/2}} e^{-\frac{|x|^2}{4\delta_t}}$.

Le lien avec le mouvement par courbure moyenne vient du fait qu'à l'interface, diffuser revient à déplacer l'interface suivant la vitesse normale $V = H$. Plus précisément, il est possible de montrer, voir [179, 150], que la ligne de niveau $1/2$ de $\mathbb{1}_{\Omega_n}$ après diffusion avec le noyau de convolution G_{δ_t} s'est déplacée dans la direction normale à l'interface d'une distance y vérifiant

$$y = H\delta_t - \frac{1}{2}\mathcal{W}'\delta_t^2 + \mathcal{O}(\delta_t^3), \tag{11.2.1}$$

où H est la courbure à l'interface $\partial\Omega$ et le terme d'ordre 2 correspond alors à la variation première de l'énergie de Willmore $\mathcal{W}' = \Delta_{\partial\Omega}H - \frac{1}{2}H^3 + H\|A\|^2$.

Plusieurs améliorations et généralisations de cet algorithme ont été proposées. On peut citer les travaux d'Esedoglu et Otto [147] qui ont donné une formulation variationnelle au schéma de BMO pouvant se généraliser dans un contexte multiphase avec des tensions de surface générales. Dans le cas du flot de Willmore, une approche possible est d'utiliser l'algorithme de BMO en combinant des convolutions à différentes échelles de temps afin de récupérer uniquement le terme d'ordre 2 dans le développement (11.2.1). Cette approche a été en particulier proposée par Grzhibovskis et Heintz [179] et conduit au schéma suivant

$$\Omega^{n+1} = \left\{\frac{1}{2\sqrt{\pi}\delta_t^{1/4}}\left(2G_{\sqrt{\delta_t}} * \mathbb{1}_{\Omega^n} - G_{2\sqrt{\delta_t}} * \mathbb{1}_{\Omega^n}\right) \geq \frac{1}{2}\right\}.$$

Cette méthode a un coût de calcul similaire à l'algorithme de BMO. En particulier, elle est stable sans condition sur le pas de temps, contrairement aux schémas classiques des modèles champ de phase de Willmore qui introduisent une condition CFL sur le pas de temps pour assurer leur stabilité. Cependant, son

efficacité dépend fortement de la discrétisation en espace. En effet, si la discrétisation en espace n'est pas assez fine, il peut arriver que l'interface se retrouve « figée » sur certaines mailles : si le déplacement est trop « faible » par rapport à la taille du maillage, alors l'étape de seuillage renvoie la même indicatrice qu'à l'étape précédente et il n'y a plus de déplacement. Pour y remédier, une possibilité serait d'avoir recours à un maillage adapté à chaque étape mais on perdrait en efficacité numérique.

Une alternative pour remédier à ce défaut souvent observé pour les méthodes de type convolution seuillage a été proposée par Esedoglu, Ruuth et Tsai [149]. Leur idée consiste à diffuser la fonction distance signée au domaine plutôt que son indicatrice. La fonction distance signée a en effet de bonnes propriétés géométriques au voisinage de l'interface [33, 12] et permet d'éviter une transition abrupte à l'interface comme avec la fonction indicatrice. Pour ce type d'approches, l'étape de seuillage est alors remplacée par une étape de re-distanciation qui peut cependant se révéler très coûteuse en dimension 3 en comparaison avec une étape de seuillage.

Nous proposons d'adapter le raisonnement précédent dans un formalisme champ de phase afin de garder l'efficacité numérique de ces algorithmes tout en conservant une approche d'interface diffuse qui permet d'éviter une forte dépendance à la discrétisation spatiale.

11.3 Une version champ de phase de l'approche Bence-Merriman-Osher

Les algorithmes précédents nous amènent en effet à penser qu'il est possible d'approcher le flot de Willmore en combinant le flot d'Allen-Cahn à différentes échelles de temps. Une première motivation vient de la remarque suivante : si $\mathcal{G}(t)$ est le flot global de l'équation d'Allen-Cahn, alors pour tout u suffisamment régulière, nous avons

$$\mathcal{G}(t)u = u + t\partial_t u + \frac{t^2}{2}\partial_{tt}^2 u + \mathcal{O}(t^3).$$

Comme $\partial_t u = \Delta u - \frac{1}{\epsilon^2}W'(u)$, on en déduit que $\partial_{tt}^2 u = \Delta\left(\Delta u - \frac{1}{\epsilon^2}W'(u)\right) - \frac{1}{\epsilon^2}W''(u)\left(\Delta u - \frac{1}{\epsilon^2}W'(u)\right)$ et on obtient alors

$$\mathcal{G}(t)u = u + t\left(\Delta u - \frac{1}{\epsilon^2}W'(u)\right) + \frac{t^2}{2}\left(\Delta\left(\Delta u - \frac{1}{\epsilon^2}W'(u)\right) - \frac{1}{\epsilon^2}W''(u)\left(\Delta u - \frac{1}{\epsilon^2}W'(u)\right)\right) + \mathcal{O}(t^3) \tag{11.3.1}$$

$$= u - t\mathcal{P}_\epsilon'(u) - \frac{t^2}{2}\mathcal{W}_\epsilon'(u) + \mathcal{O}(t^3) \tag{11.3.2}$$

où $\mathcal{P}_\epsilon'(u)$ et $\mathcal{W}_\epsilon'(u)$ sont respectivement les gradients L^2 de l'énergie de Cahn-Hilliard $\mathcal{P}_\epsilon$ et de l'énergie champ de phase classique $\mathcal{W}_\epsilon$. Le terme d'ordre 2 en temps du flot d'Allen-Cahn est donc le gradient L^2 de l'énergie classique $\mathcal{W}_\epsilon$.

Pour capturer ce terme avec une méthode numérique, il est nécessaire d'utiliser une discrétisation d'ordre 2 en temps, en utilisant par exemple une méthode de Strang. Plus précisément, si $\mathcal{X}(t) = e^{t\Delta}$ est le semi-groupe de la chaleur et $\mathcal{Y}(t)$ est le flot de réaction associé à l'équation d'Allen-Cahn, alors le splitting de Strang est défini par

$$\mathcal{S}(t) = \mathcal{Y}(t/2)\mathcal{X}(t)\mathcal{Y}(t/2). \tag{11.3.3}$$

Il est assez facile de vérifier que c'est un schéma d'ordre 2, c'est à dire $S(t) = \mathcal{G}(t) + \mathcal{O}(t^3)$. En effet, en remarquant que

$$\begin{aligned}\mathcal{X}(t)\mathcal{Y}(t/2)u = u + t\left(\Delta u - \frac{1}{2\epsilon^2}W'(u)\right)\\ + \frac{t^2}{2}\left(\Delta^2 u - \frac{1}{\epsilon^2}\Delta W'(u) + \frac{1}{4\epsilon^4}W''(u)W'(u)\right) + \mathcal{O}(t^3)\end{aligned}$$

on en déduit que

$$\mathcal{Y}(t/2)\mathcal{X}(t)\mathcal{Y}(t/2)u = u - t\mathcal{P}_\epsilon'(u) - \frac{t^2}{2}\mathcal{W}_\epsilon'(u) + \mathcal{O}(t^3).$$

Ce résultat montre en particulier qu'il est possible de récupérer le terme d'ordre 2 de $\mathcal{G}(t)$ à partir d'une combinaison du type

$$\left[2\mathcal{S}\left(\sqrt{t}\right) - \mathcal{S}\left(2\sqrt{t}\right)\right]u = u - t\mathcal{W}_\epsilon'(u) + \mathcal{O}(t^{3/2})$$

mais ce développement montre que le schéma ne sera consistant qu'à l'ordre $1/2$ en temps seulement.

Pour obtenir des schémas numériques plus précis, une possibilité est de considérer davantage de combinaisons. Par exemple, on peut retrouver un schéma d'ordre 1 en optimisant les couples $(\alpha_i, \beta_i)_{1\leq i\leq 4}$ afin de vérifier

$$\sum_{i=1}^{4} \beta_i S(\alpha_i t)u = u - t^2 \mathcal{W}_\epsilon'(u) + \mathcal{O}(t^4)$$

Il suffit pour cela de vérifier le système de Vandermond suivant :

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \alpha_4^2 \\ \alpha_1^3 & \alpha_2^3 & \alpha_3^3 & \alpha_4^3 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -2 \\ 0 \end{pmatrix}.$$

Cette équation matricielle admet une unique solution si, et seulement si, les α_i sont deux à deux distincts, et dans ce cas les β_i sont uniques et vérifient

$$\beta_i = \frac{\prod_{j\neq i} \alpha_j}{\prod_{j\neq i} (\alpha_j - \alpha_i)} - 2\frac{\sum_{j\neq i} \alpha_j}{\prod_{j\neq i} (\alpha_j - \alpha_i)},\ i = 1, \cdots, 4.$$

Ainsi, en choisissant opportunément les valeurs des α_i, on obtient un schéma d'ordre 1. L'avantage d'une telle approche est qu'il est bien plus simple de simuler des flots d'Allen-Cahn (tout comme des combinaisons de tels flots) que de simuler directement le flot de gradient de l'énergie classique $\mathcal{W}_\epsilon$. Cependant, les α_i (et donc des β_i) doivent être choisis judicieusement pour assurer la stabilité du schéma.

Ce résultat nous conduit à penser qu'une nouvelle structure de réseaux combinant en parallèle 2 ou 4 réseaux adaptés au flot d'Allen Cahn devrait aussi permettre d'approcher efficacement le flot souhaité. L'avantage avec une telle approche est que les paramètres α_i et β_i peuvent être optimisés durant le processus d'apprentissage.

11.4 Adaptation aux réseaux de neurones

Nous décrivons dans cette section la démarche que nous utilisons pour construire des schémas numériques d'approximation champ de phase du flot de Willmore à l'aide de réseaux de neurones. Encore une fois, l'idée est d'approcher le semi-groupe $\mathcal{W}^q_{dt,\varepsilon}$ défini par

$$\mathcal{W}^q_{dt,\varepsilon}\left[v_\varepsilon(\cdot, t)\right] = v_\varepsilon(\cdot, t + \delta_t)$$

où $v_\varepsilon(\cdot, t) = q\left(\dfrac{d(\cdot, \Omega(t))}{\varepsilon}\right)$. Ici, q représente le profil classique associé au potentiel double puits, la fonction d désigne la distance signée et $(\Omega(t))_{t\geq 0}$ correspond au flot de Willmore d'une forme initiale Ω_0.

L'idée consiste alors à entraîner un réseau de neurones $\mathcal{W}^{\mathrm{NN}}_\theta$ sur une base de données que nous présentons plus bas. Nous espérons alors que l'entraînement du réseau $\mathcal{W}^{\mathrm{NN}}_\theta$ soit suffisamment concluant pour que le schéma numérique,

$$u^{n+1} = \mathcal{W}^{\mathrm{NN}}_\theta\left[u^n\right],$$

associé à la donnée initiale

$$u^0 = q\left(\frac{d(\cdot, \Omega(0))}{\varepsilon}\right),$$

soit une bonne approximation du flot de Willmore à partir du domaine Ω_0.

11.4.1 Structures de réseaux

En nous inspirant de l'approche précédente, nous construisons de nouvelles structures de réseaux basées sur celles que nous avions déjà utilisées dans le chapitre précédent. En effet, afin d'avoir des schémas précis, nous remplaçons le schéma de Strang par un réseau de neurones $\mathcal{DR}$ d'ordre 2 (noté $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$) qui nous a permis de gagner en ordre de précision numérique par rapport aux schémas classiques dans le cas du mouvement par courbure moyenne. Nous rappelons son architecture en figure 11.5.

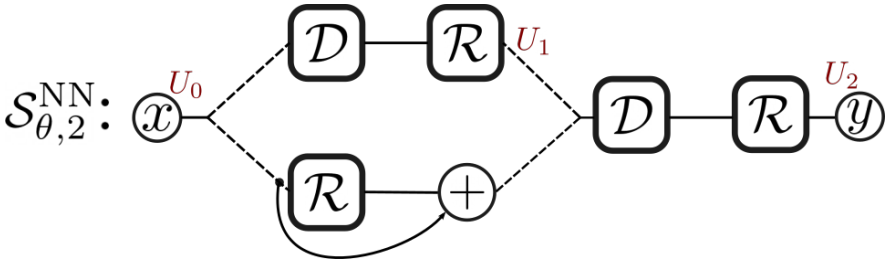


FIGURE 11.5 – Réseau de neurones $\mathcal{DR}$ d'ordre 2, $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$.

Les réseaux que nous proposons pour approcher le flot de Willmore sont alors de simples combinaisons linaires de réseaux du type $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$, les coefficients de la combinaison linéaire étant aussi des paramètres d'apprentissage. La figure 11.6 montre deux exemples de réseaux que nous notons respectivement $\mathcal{W}^{\mathrm{NN}}_{\theta,1}$ et $\mathcal{W}^{\mathrm{NN}}_{\theta,2}$, construits en mettant respectivement 2 et 4 réseaux $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ différents en parallèle.

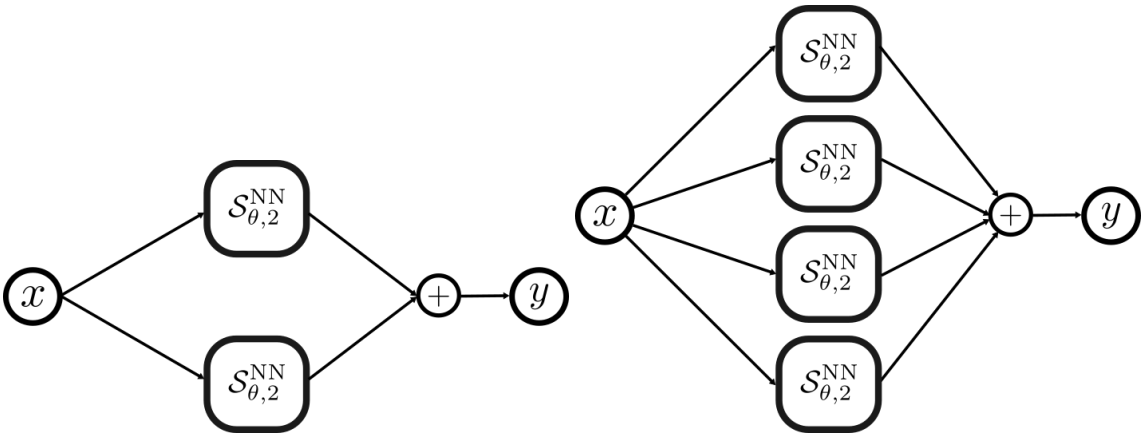


FIGURE 11.6 – Structures de réseau $\mathcal{W}^{\mathrm{NN}}_{\theta,1}$ (à gauche) et $\mathcal{W}^{\mathrm{NN}}_{\theta,2}$ (à droite) inspirées de l'approche BMO, adaptées à l'approche champ de phase et construites à partir du réseau $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$. Les blocs $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ en parallèle ont tous des paramètres indépendants.

Enfin, notons que dans toutes les expériences numériques que nous présenterons plus loin, la taille de tous les noyaux est fixée à 17×17 et l'architecture du neurone de réaction est identique à celle utilisée pour le mouvement par courbure moyenne.

11.4.2 Base de données

On pourrait utiliser le même raisonnement que dans le chapitre précédent en utilisant des évolutions explicites d'interfaces avec une représentation champ de phase exacte. Cependant, les données explicites que nous avons à notre disposition sur ce flot sont limitées aux cas des cercles pour lesquels l'évolution ne prend pas en compte la diffusion de surface. En effet, la seule évolution explicite connue en 2D est celle des cercles : étant donné un cercle de rayon R_0, le flot de Willmore préserve le cercle tout en augmentant le rayon selon la loi

$$R(t) = \left(R_0^4 + 2t\right)^{1/4}.$$

Nous pouvons ainsi construire une première base de données à partir d'une famille finie de rayons R_i avec $i \in \{1, \cdots, N_1\}$, en considérant les couples

$$(X_i, Y_i) = \left(\varphi_{R_i}, \varphi_{\left(R_0^4+2\delta_t\right)^{1/4}}\right), \quad \forall i \in \{1, \cdots, N_1\},$$

où on a posé $\varphi_R(x) = q\left(\frac{d(x,B_R)}{\varepsilon}\right)$ avec B_R le cercle de rayon R centré à l'origine du repère. Le pas de temps est fixé à $\delta_t = \varepsilon^4$.

Étant donné que la diffusion de surface n'est pas prise en compte durant l'évolution d'un cercle, il est a priori difficile de pouvoir correctement retrouver l'évolution limite,

$$V = \Delta_{\partial\Omega} H - \frac{1}{2} H^3 + H\|A\|^2,$$

uniquement à partir de ce type de données.

Afin d'enrichir les données d'entraînement, nous construisons une seconde base de données obtenue en considérant cette fois-ci des ellipses. Contrairement au cas du cercle, le terme $\Delta_{\partial\Omega} H$ est actif pour ces évolutions. On génère alors dans un premier temps N_2 ellipses E_i dont les axes sont choisis aléatoirement de sorte à prendre en compte différentes directions. On obtient ainsi une seconde base de données constituée de couples de la forme $(\tilde{X}_i, \tilde{Y}_i)$ où $\tilde{X}_i = q\left(\frac{d(x,E_i)}{\varepsilon}\right)$ et $\tilde{Y}_i$ correspond à l'itération après un pas de temps $\delta_t = \varepsilon^4$, générée à partir du modèle classique (11.1.1). Pour cela, on utilise un schéma de discrétisation de (11.1.1) dont le pas de temps est divisé par 100, $\delta_t/100$, afin que les erreurs liées à la méthode numérique soient négligeables.

Ainsi, en concaténant ces deux collections de données, on obtient une base de données d'entraînement de taille $N_{\text{train}} = N_1 + N_2$.

Ainsi, pour entraîner un réseau $\mathcal{W}_\theta$, on minimise l'énergie suivante,

$$J(\theta) = \frac{1}{N_{\text{train}}} \left(\sum_{i=1}^{N_1} \|\mathcal{W}_\theta(X_i) - Y_i\|^2 + \sum_{i=1}^{N_2} \left\|\mathcal{W}_\theta(\tilde{X}_i) - \tilde{Y}_i\right\|^2 \right),$$

où $\|f\| = \left(\int_Q f^2(x)\,dx\right)^{\frac{1}{2}}$.

De manière similaire au cas du mouvement par courbure moyenne, il est possible d'étendre la démarche précédente en comparant sur plusieurs itérations au lieu d'une seule. Dans ce cas, les Y_i sont remplacés par les $\{Y_{i,j}\}_{1\le j\le k}$ où k correspond au nombre d'itérations et $Y_{i,j} = \varphi_{\left(R_0^4 + 2j\delta_t\right)^{1/4}}$. On fait de même avec les $\tilde{Y}_i$ en considérant cette fois-ci k itérations $\{\tilde{Y}_{i,j}\}_{1\le j\le k}$ générées à partir du modèle classique (11.1.1). On se retrouve alors avec la minimisation de l'énergie suivante,

$$J_k(\theta) = \frac{1}{N_{\text{train}}} \left(\sum_{i=1}^{N_1} \sum_{j=1}^{k} \left\|\left(\mathcal{W}_\theta\right)^{(j)}(X_i) - Y_{i,j}\right\|^2 + \sum_{i=1}^{N_2} \sum_{j=1}^{k} \left\|\left(\mathcal{W}_\theta\right)^{(j)}(\tilde{X}_i) - \tilde{Y}_{i,j}\right\|^2 \right),$$

où $\left(\mathcal{W}_\theta\right)^{(j)}$ est la composition itérée j fois de $\mathcal{S}_\theta$.

Ce procédé permet de stabiliser l'apprentissage du réseau $\mathcal{S}_\theta$, notamment lorsque le pas de temps δ_t entre chaque itération est très petit et que les variations entre deux approximations ne sont donc pas suffisamment grandes pour que le réseau puisse efficacement reproduire la bonne évolution, même dans le cas le plus simple des cercles.

C'est cette dernière stratégie multi-pas que nous adoptons dans nos expériences. En particulier, nous générons $N_1 = 100$ cercles et $N_2 = 100$ ellipses, et nous comparons sur $k = 10$ itérations.

11.5 Expériences numériques et validation de la méthode

Dans cette section nous testons la capacité de nos réseaux à apprendre le flot de Willmore orienté. Pour approcher le semi-groupe $\mathcal{W}^q_{dt,\varepsilon}$, les réseaux $\mathcal{W}^{\text{NN}}_{\theta,1}$ et $\mathcal{W}^{\text{NN}}_{\theta,2}$ sont entraînés sur la base d'apprentissage présentée dans la section précédente où nous fixons ici les paramètres $\varepsilon = \delta_x$ et $\delta_t = \varepsilon^4$, où δ_x désigne le pas d'espace.

Nous présentons en figure 11.7 l'évolution de l'énergie J_k à travers le processus d'apprentissage avec en bleu la courbe d'erreur d'entraînement du réseau $\mathcal{W}^{\text{NN}}_{\theta,1}$ et en orange celle de $\mathcal{W}^{\text{NN}}_{\theta,2}$. On observe une décroissance des deux courbes d'entraînement au cours de l'apprentissage. Il semble donc que, dans les deux cas, les sorties des réseaux permettent d'avoir de bonnes approximations des sorties du semi-groupe $\mathcal{W}^q_{dt,\varepsilon}$ sur des cercles et ellipses, au moins en norme 2.

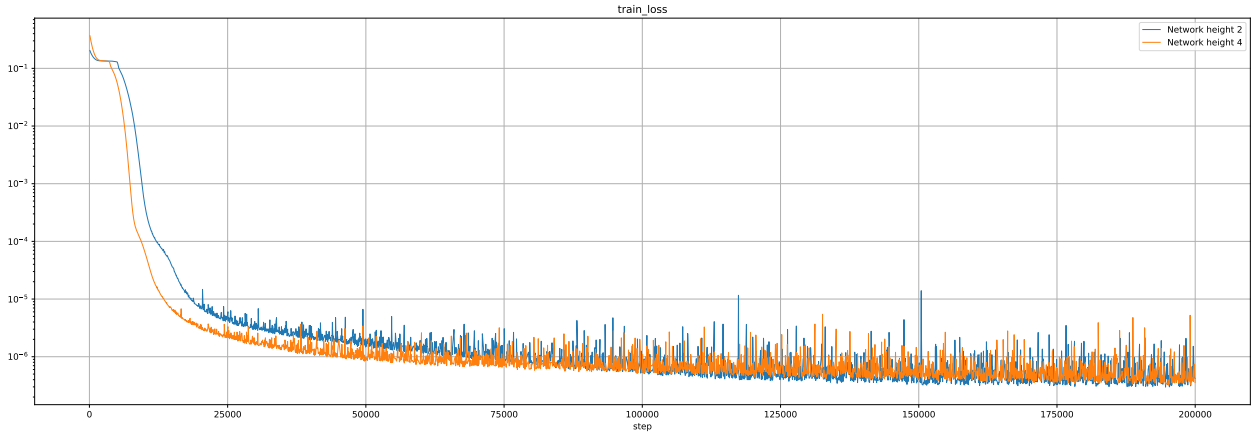


FIGURE 11.7 – Processus d'apprentissage par optimisation de $J_k(\theta)$. Courbes d'erreur d'entraînement des réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ (en bleu) et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ (en orange).

Validations qualitative et quantitative sur l'évolution d'un cercle

Afin d'avoir des résultats plus qualitatifs, nous testons les deux réseaux entraînés $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ sur l'évolution d'un cercle. Pour cela, nous considérons les schémas $u^{n+1} = \mathcal{W}_{\theta,i}^{\mathrm{NN}}\left[u^n\right],\ i = 1, 2$ avec la condition initiale $u^0 = q\left(\frac{d(\cdot,\Omega(0)}{\varepsilon}\right)$ où $\Omega(0)$ est le cercle de rayon $R = 0.05$. Nous comparons alors les évolutions obtenues à partir des schémas précédents à l'évolution exacte partant de $\Omega(0)$ en représentation champ de phase.

Plus précisément, nous affichons en figure 11.8 les approximations numériques u^n à différents instants $t_n = n\delta_t$ en utilisant respectivement le flot exact de $\Omega(0)$ en représentation champ de phase, le réseau $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$. Nous observons que dans les trois cas la vitesse d'évolution semble identique. Au moins dans ce cas-là, les deux réseaux donnent des résultats qualitatifs assez satisfaisants. Il reste néanmoins difficile de juger de la qualité de nos réseaux sur ce seul exemple.

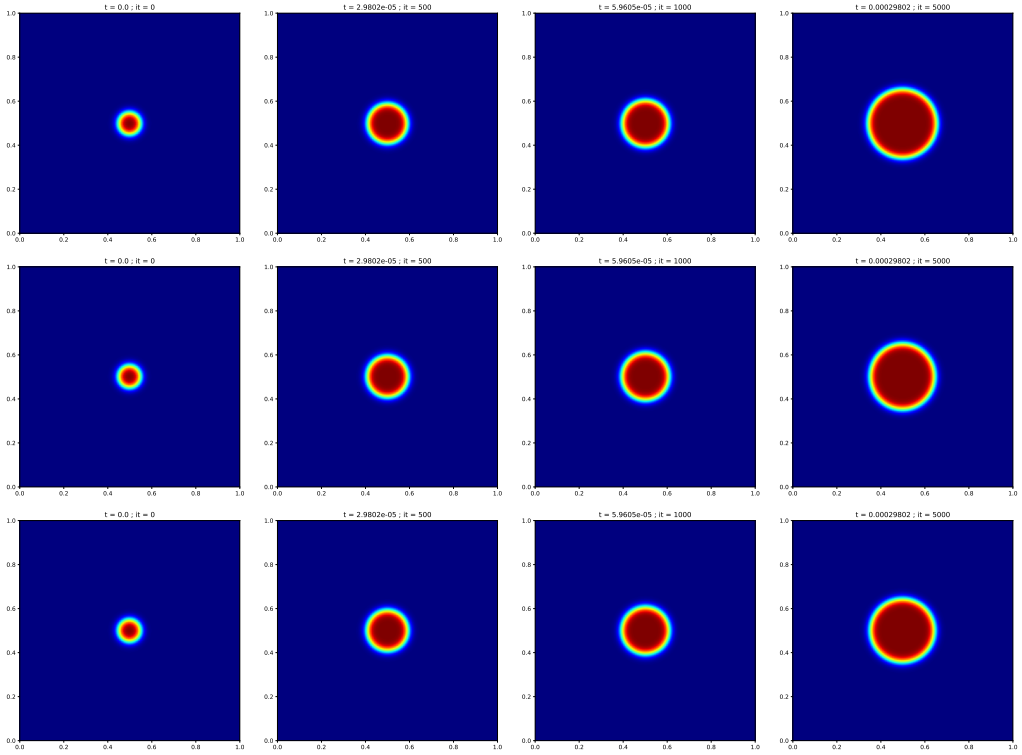


FIGURE 11.8 – Simulation du flot de Willmore d'un cercle en utilisant respectivement le flot exact, et les réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$

Afin d'avoir un critère plus quantitatif sur l'évolution du cercle, nous traçons en figure 11.9 l'évolution de $n \mapsto \left(\frac{1}{\pi}\int u^n\,dx\right)^2$ qui devrait correspondre à l'évolution du produit de 1

pi^2 par le carré de l'aire du cercle de rayon $R(t) = \left(R_0^4 + 2t\right)^{1/4}$, produit qui est donné par $t \mapsto R_0^4 + 2t$. Le rayon initial est fixé à $R_0 = 0.05$.

Nous observons que la loi d'évolution ne correspond pas à exactement à l'évolution attendue du rayon, et en particulier des erreurs plus importantes apparaissent lorsque le rayon du cercle devient de plus en plus grand. Du point de vue de l'apprentissage, cette observation pourrait se justifier par le fait que les déplacements aux interfaces des grands cercles sont très minimes en comparaison à celles des petits cercles. Les erreurs commises sur les grands cercles durant l'apprentissage sont donc négligeables par rapport aux erreurs sur les petits cercles. L'entraînement du réseau aura donc tendance à corriger plutôt les erreurs importantes sur les petits cercles au détriment des erreurs sur les grands cercles, comme semblent le montrer les évolutions observées en figure 11.9.

Ces résultent montrent l'importance de la base de données et de la stratégie d'apprentissage pour que nos réseaux soient capables d'apprendre à reproduire correctement le mouvement souhaité. Afin de mieux prendre en compte les grands cercles durant l'apprentissage, une stratégie possible serait d'ajouter plus de grands cercles dans base de données. Il serait aussi intéressant d'ajouter des poids associés aux différentes données présentes dans l'énergie J_k afin d'accorder plus d'importance aux grands cercles pendant l'entraînement.

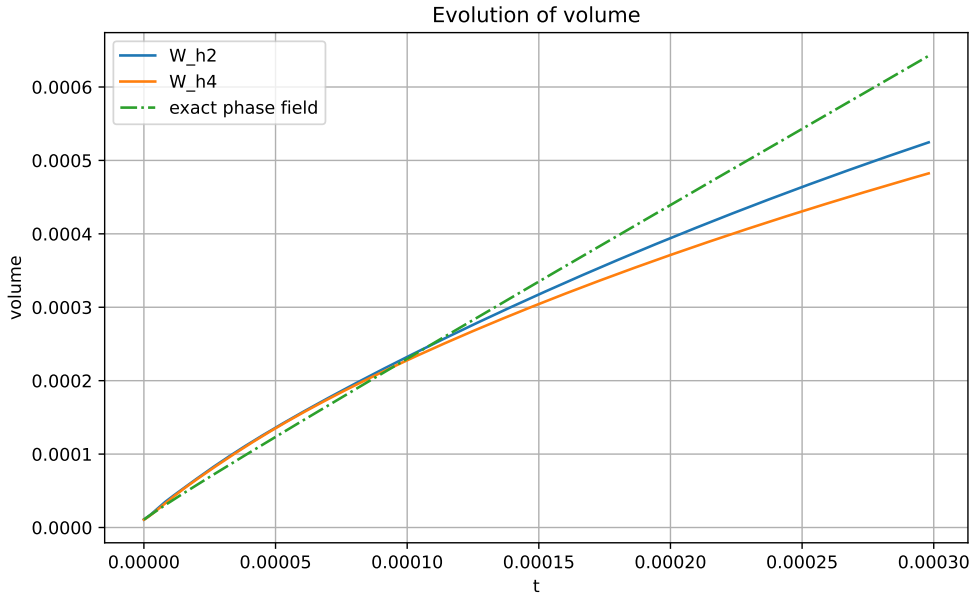


FIGURE 11.9 – Comparaison des réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ (en bleu) et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ (en orange) sur l'évolution $n \mapsto \left(\frac{1}{\pi}\int u^n\, dx\right)^2$ par rapport à celle du flot exact (en vert).

Un exemple d'évolution sur un rectangle

Afin d'analyser la capacité de nos réseaux à se généraliser à des formes jamais utilisées durant l'apprentissage, nous les testons également sur l'exemple d'un rectangle, que nous comparons à l'évolution obtenue à partir du modèle classique champ de phase de Willmore. Nous affichons en figure 11.10 les approximations numériques u^n en utilisant respectivement les schémas découlant des réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$, $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ et du modèle classique. Nous observons un comportement similaire sur les évolutions obtenues à partir des deux réseaux. Dans les deux cas, les coins du rectangle sont arrondis pour former une forme en ellipse à partir d'un certain temps t. Nous remarquons néanmoins une certaine différence en comparaison avec le modèle classique où nous observons une évolution légèrement plus lente par rapport aux évolutions découlant des deux réseaux. La vitesse d'évolution déduite de $\mathcal{W}_{\theta,i}^{\mathrm{NN}}, i = 1, 2$ semble plus rapide aux zones de fortes courbures, ce qui est en corrélation avec ce que nous avons déjà observé en figure 11.9 où l'évolution des petits cercles est plus rapide que la vraie évolution pour les deux réseaux.

Évolution de deux cercles disjoints et formation de singularités

Dans cette sous-section, nous nous interrogeons sur le comportement des approximations u^n obtenues à partir des schémas $u^{n+1} = \mathcal{W}_{\theta,i}^{\mathrm{NN}}\left[u^n\right],\ i = 1, 2$ lorsque des singularités apparaissent durant l'évolution.

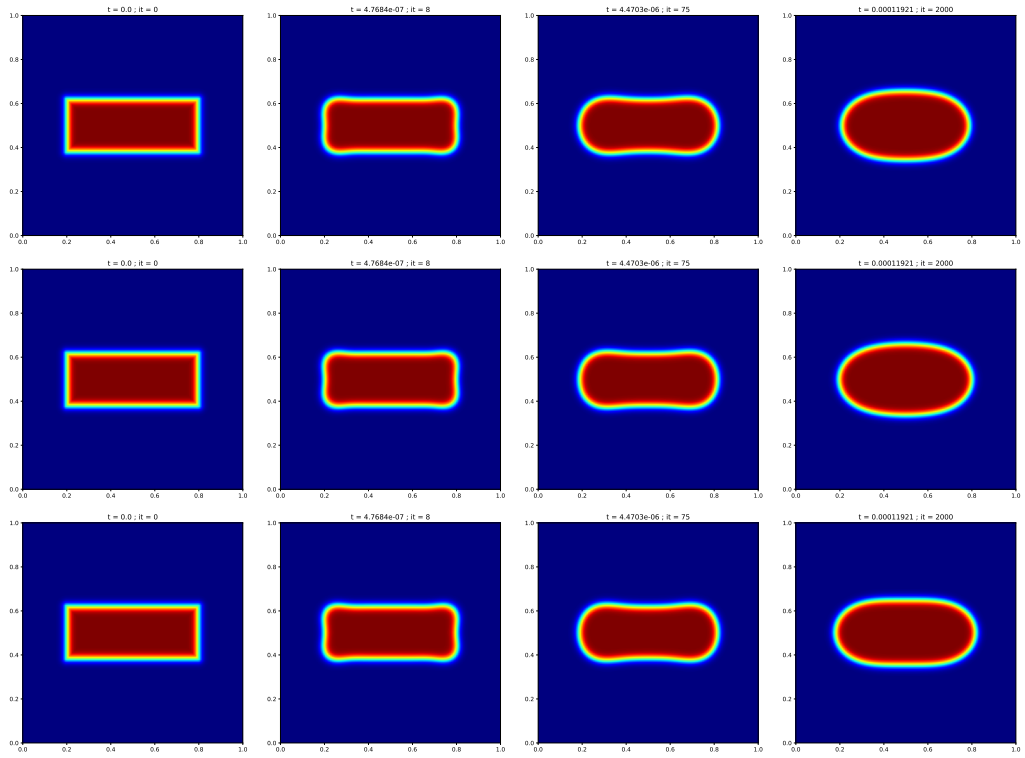


FIGURE 11.10 – Simulation du flot de Willmore d'un rectangle en utilisant les réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ (première ligne), $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ (deuxième ligne) et le modèle classique (troisième ligne).

Nous considérons pour cela l'exemple de deux cercles disjoints de même rayon $R = 0.15$. Nous savons dans ce cas que les deux cercles grossissent indépendamment, jusqu'à leur collision. D'un point de vue théorique, l'évolution après la collision des deux cercles n'est plus définie au sens classique et le comportement à partir des premières singularités dépend alors de la formulation mathématique utilisée. Nous avons déjà observé dans le cas de l'approximation champ de phase classique et de l'approximation de Mugnai que le comportement, lorsque deux interfaces se rapprochent l'une de l'autre, dépend du modèle champ de phase utilisé : le modèle classique autorise les croisements alors que deux interfaces qui se rapprochent ont tendance à se repousser dans le cas du modèle de Mugnai. Nous constatons de la même façon sur la figure 11.11 que le comportement à proximité de la collision des deux cercles est différent suivant le réseau utilisé. Plus précisément, nous affichons en figure 11.11 l'approximation u^n à différents instants $t_n = n\delta_t$ obtenue à partir du réseau $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ (première ligne) et du réseau $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ (deuxième ligne). Bien que les deux réseaux aient été entraînés sur la même base de données, leur action sur des données pouvant conduire à des singularités est très différente. On retrouve en fait la différence observée entre le modèle champ de phase classique et le modèle de Mugnai : le réseau $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ autorise les croisements tandis que le réseau $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$ les empêche.

Ce test montre que l'architecture et l'apprentissage influencent la capacité de généralisation du réseau et un nouvel apprentissage pourrait sans doute donner des résultats différents. En effet, nous n'avons pour l'instant aucun début d'explication qui justifierait la différence de comportement obtenu avec les deux réseaux. Il semble donc important d'enrichir la base de données afin d'aider le réseau à choisir entre les comportements du modèle de champ de phase classique de celui du modèle de Mugnai. Pour ce faire, on pourrait forcer le modèle classique en ajoutant des solutions stationnaires de l'équation Allen Cahn afin d'autoriser la présence de croisements. Pour le modèle de Mugnai, une autre idée serait de coupler le réseau avec un terme qui pénaliserait la présence de singularité comme dans [148].

Couplage avec une contrainte d'inclusion/exclusion

Nous finissons cette partie par une application à la minimisation de l'énergie de Willmore sous contrainte d'inclusion-exclusion,

$$\underset{\Omega_1 \subset \Omega \subset \Omega_2}{\operatorname{argmin}} \ \mathcal{W}(\Omega) \tag{11.5.1}$$

où Ω_1 et Ω_2 sont nos contraintes d'inclusion-exclusion.

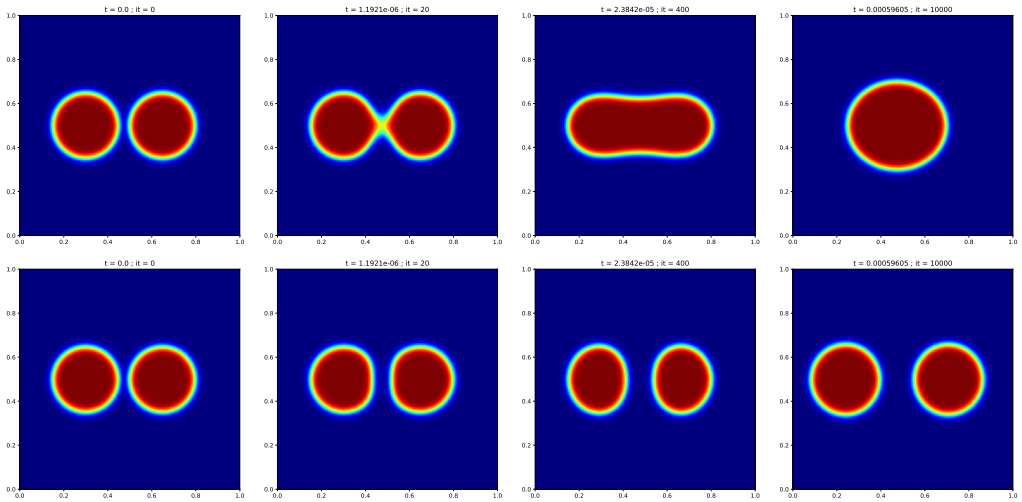


FIGURE 11.11 – Simulation du flot de Willmore de deux cercles. Chaque ligne représente l'évolution à différents instant t où on a utilisé respectivement les réseaux $\mathcal{W}_{\theta,1}^{\mathrm{NN}}$ et $\mathcal{W}_{\theta,2}^{\mathrm{NN}}$.

Nous allons donner deux exemples où dans un premier cas les contraintes seront régulières alors que dans le deuxième cas elles seront supposées singulières. L'objectif est de voir si nos réseaux sont suffisamment robustes pour être couplés à de telles contraintes, même lorsque celles-ci sont singulières.

Pour résoudre le problème (11.5.1), nous adoptons une méthodologie similaire à celle proposée dans les articles [74, 68]. Elle consiste à considérer le flot de Willmore $t \mapsto \Omega(t)$ d'une forme initiale Ω_0 vérifiant $\Omega_1 \subset \Omega_0 \subset \Omega_2$ en nous imposant la contrainte d'inclusion-exclusion $\Omega_1 \subset \Omega(t) \subset \Omega_2$ le long du flot. Ainsi, l'état stationnaire du flot est un point critique et un potentiel minimiseur (au moins local) de notre problème. Du point de vue champ de phase, la stratégie consiste à coupler nos schémas précédents à la contrainte d'inclusion-exclusion : en partant de la condition initiale,

$$u^0 = q\left(\frac{d(\cdot,\Omega_0)}{\varepsilon}\right),$$

la contrainte initiale se réécrit $u_1 \leq u^0 \leq u_2$ où $u_i = q\left(\dfrac{d(\cdot,\Omega_i)}{\varepsilon}\right)$, $i = 1, 2$. Le flot minimisant discret (u^n) est alors construit récursivement en suivant les étapes suivantes :

1. On applique le réseau $\mathcal{W}_{\ell,\theta}^{\mathrm{NN}}$:

$$u^{n+1/2} = \mathcal{W}_{\ell,\theta}^{\mathrm{NN}}\left[u^n\right].$$

2. On projette sur la contrainte :

$$u^{n+1} = \min\left(\max\left(u^{n+1/2}, u_1\right), u_2\right).$$

Ici, ℓ est fixé à 1 ou 2 selon que l'on utilise le réseau $\mathcal{W}_{1,\theta}^{\mathrm{NN}}$ ou $\mathcal{W}_{2,\theta}^{\mathrm{NN}}$ pour simuler le flot de Willmore. Cette même approche peut être utilisée avec le modèle champ de phase classique où nous remplaçons la première étape du schéma précédent par la résolution du modèle classique sur un pas de temps δ_t pour générer cette étape du flot de Willmore.

Un premier cas avec des contraintes lisses

Nous testons ce schéma sur un exemple d'inclusion-exclusion avec deux ellipses. Plus précisément, nous affichons en figure 11.12 la ligne de niveau $1/2$ des approximations u^n à différents instants $t_n = n\delta_t$ en utilisant, respectivement, le modèle classique et les réseaux $\mathcal{W}_{1,\theta}^{\mathrm{NN}}$ ou $\mathcal{W}_{2,\theta}^{\mathrm{NN}}$, où la condition initiale Ω_0 est un cercle (représenté en bleu sur la première image de chaque ligne) et les contraintes sont deux ellipses de tailles différentes (une petite en vert et une grande en noir). Sur cet exemple, nous observons un comportement similaire dans les trois cas : le cercle grossit avant de rencontrer la grande ellipse puis se déforme en devenant « quasiment » une ellipse.

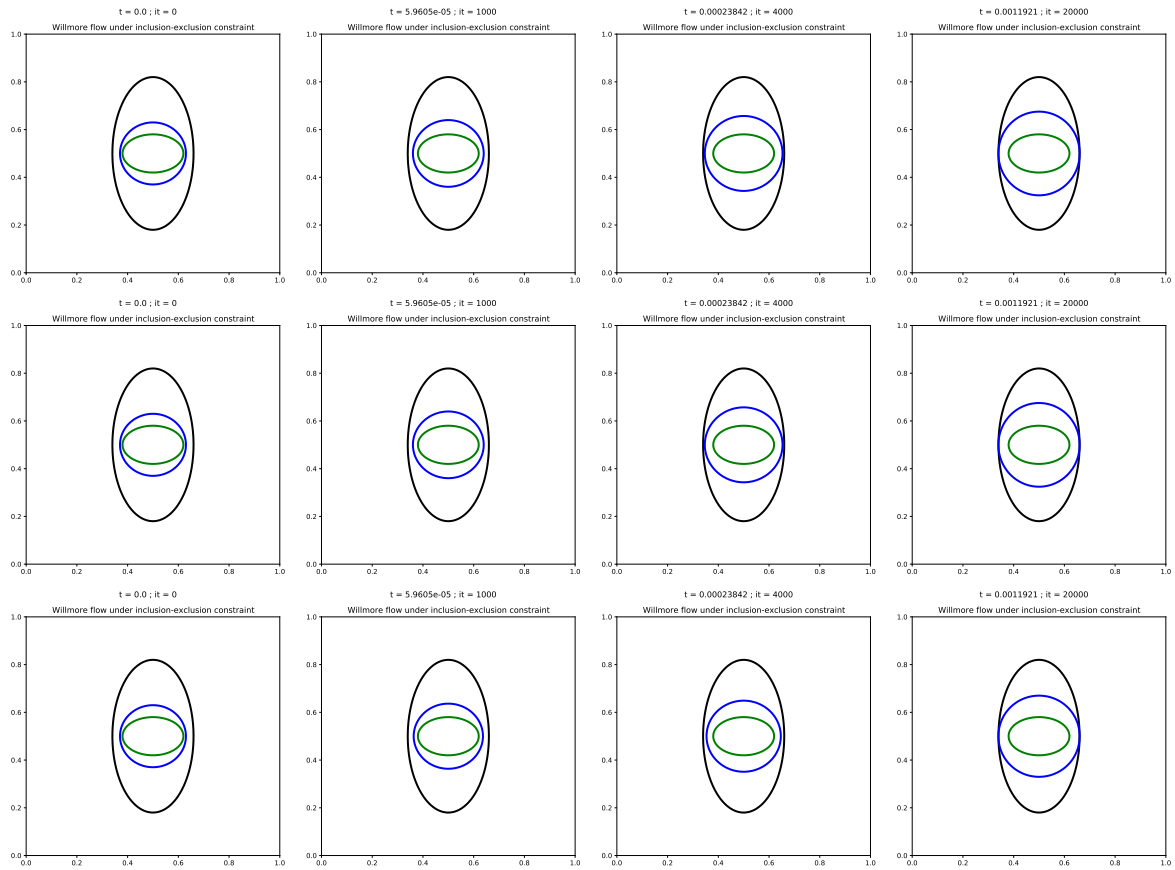


FIGURE 11.12 – Simulation du flot de Willmore sous contrainte d'inclusion-exclusion. Chaque ligne représente l'évolution de la ligne de niveau $\{u^n = 1/2\}$ (en bleu) à différents instant $t_n = n\delta_t$ où l'on a utilisé respectivement, le modèle classique et les réseaux $\mathcal{W}^{\mathrm{NN}}_{\theta,1}$ et $\mathcal{W}^{\mathrm{NN}}_{\theta,2}$. Les contraintes sont données par les ellipses (contrainte extérieure en noir et contrainte intérieure en vert).

Un second cas avec des contraintes singulières

Nous avons pu constater que nos réseaux étaient suffisamment robustes pour être couplés à des contraintes d'inclusion-exclusion d'ensembles réguliers. Nous voulons à présent voir comment se comportement nos réseaux en présence de contraintes singulières. Pour cela, nous proposons d'appliquer nos schémas précédents au problème de reconstruction à partir de coupes en dimension 2. Étant donnée une famille de coupes (segments disjoints) de $\mathbb{R}^2$, l'idée est de reconstruire la « meilleure » courbe qui englobe cette famille. Ce principe se généralise aussi en dimension supérieure et un exemple concret d'application en dimension 3 concerne la reconstruction d'une image en $3D$ d'un organe à partir de coupes obtenues à partir d'une imagerie par résonance magnétique. Le lecteur pourra trouver des exemples de telles reconstructions dans [68].

Pour résoudre ce type de problèmes, nous proposons une méthode qui consiste à trouver la courbe en question comme solution d'un problème d'optimisation sous contraintes. La méthodologie que nous adoptons ici s'inspire de l'article [68] et nous en rappelons le principe ici : on considère une famille finie de droites $(\Gamma_i)_{0\leq i\leq N}$ de $\mathbb{R}^2$. Sur chacune des droites Γ_i, on se donne deux coupes, i.e deux ensembles $\Omega_i^{int},\ \Omega_i^{ext} \subset \Gamma_i$. La coupe intérieure Ω_i^{int} représente la partie interne du domaine à reconstruire et la coupe extérieure Ω_i^{ext} permet d'éviter que notre domaine englobe trop grossièrement Ω_i^{int} en lui imposant d'éviter Ω_i^{ext}. La seule condition imposée est que $\Omega_i^{ext} \cap \Omega_i^{int} = \emptyset$ mais il n'est pas nécessaire que les coupes Ω_i^{int} et Ω_i^{ext} forment une partition de Γ_i, ce qui permet de laisser de la liberté à notre méthode de reconstruction. Pour résoudre notre problème nous appliquons nos schémas précédents en considérant cette fois-ci les contraintes :

$$u_1 = \mathbb{1}_{\Omega_1} \quad \text{et} \quad u_2 = 1 - \mathbb{1}_{\Omega_2}$$

où $\Omega_1 = \cup_i \Omega_i^{int}$ et $\Omega_2 = \cup_i \Omega_i^{ext}$.

Nous affichons en figure 11.13 les résultats obtenus sur deux types de contraintes. Les premières images de chaque ligne représentent les contraintes d'inclusion-exclusion. Dans le premier cas, nous avons $\Omega_i^{ext} \cup \Omega_i^{int} \neq \Gamma_i$ alors que dans le second cas on autorise les espacements entre les coupes intérieures et extérieures.

Chaque image de la deuxième colonne représente les contraintes et la ligne de niveau $1/2$ (en noir) de la condition initiale que nous avons utilisées dans chacun des cas. Nous avons sélectionné le réseau pour lequel la reconstruction nous semblait être la « meilleure », à savoir le réseau $\mathcal{W}^{\mathrm{NN}}_{\theta,2}$. Nous affichons sur la dernière colonne de la figure 11.13 les résultats obtenus à partir de ce réseau et pour chacune des contraintes, respectivement. Nous voyons sur ces deux exemples que notre réseau parvient globalement à trouver un bon candidat à la reconstruction dans chacun des cas. Nous constatons tout de même que, dans le premier cas, la contrainte d'inclusion n'est pas complètement vérifiée sur certaines zones entre les coupes internes et externes. C'est la conséquence de l'utilisation d'une contrainte raide, plus simple à construire mais moins efficace que des contraintes lisses calculées suivant un modèle champ de phase mais qui nécessitent de calculer la fonction distance à nos ensembles Ω_1 et Ω_2.

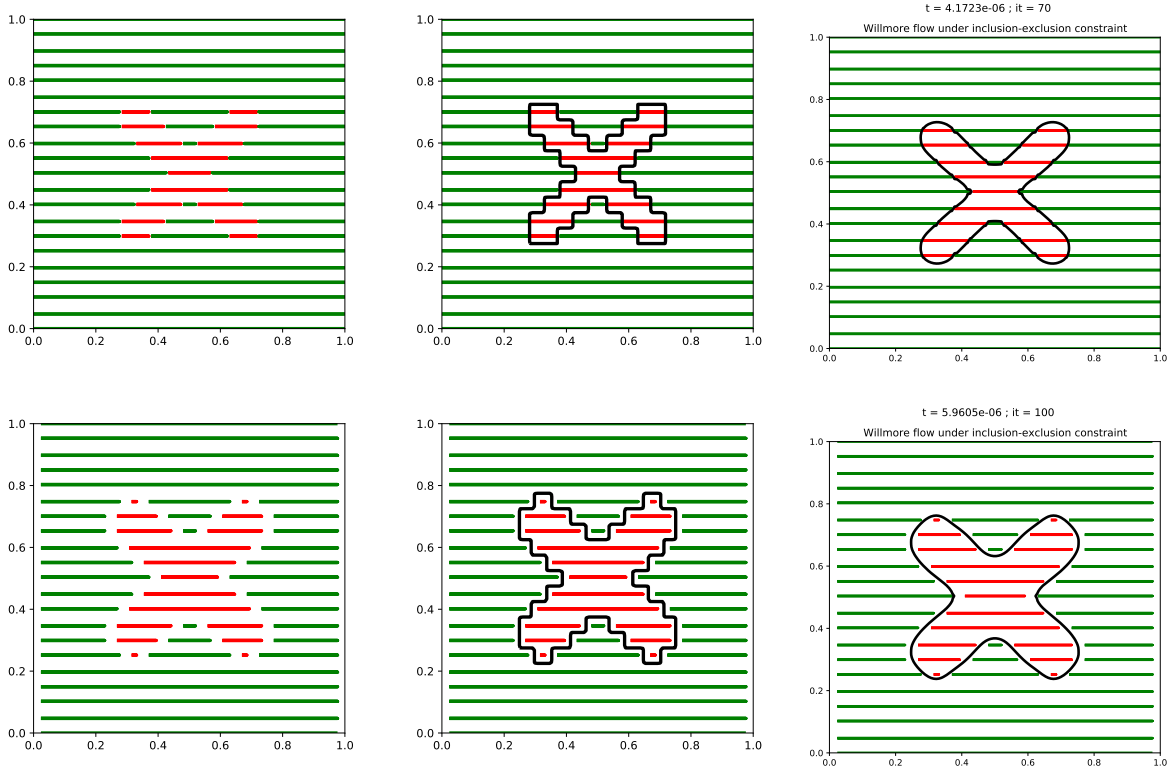


FIGURE 11.13 – Première colonne : deux exemples de familles de coupes dans $[0,1]^2$. Deuxième colonne : initialisation de la reconstruction. Troisième colonne : reconstructions obtenues à partir du réseau $\mathcal{W}^{\mathrm{NN}}_{\theta,2}$.

Conclusion

Les différentes expériences présentées dans la section précédente montrent le potentiel de notre approche pour approcher le flot de Willmore. Nos réseaux entraînés sur une petite base de données donnent des résultats satisfaisants et parviennent même à généraliser le flot sur des exemples différents de la base de données. Le cas d'apparition de singularités est aussi traité mais les résultats diffèrent selon la structure du réseau, ce que nous ne savons pas précisément expliquer pour l'instant. Ce travail n'est donc pas encore tout à fait abouti et nécessiterait de mieux comprendre comment améliorer le processus d'apprentissage afin d'obtenir des algorithmes plus performants. Il faudrait par exemple considérer une base de données plus conséquente et plus variée, mais aussi sans doute utiliser une fonction de perte multi-pas avec un nombre de pas k supérieur à 10 pour gagner en robustesse.

Chapitre 12

Mouvement par courbure moyenne anisotrope et noyau de diffusion

Comme nous l'avons vu dans l'introduction, de nombreux problèmes physiques font intervenir des énergies de surface anisotropes qui généralisent la notion de périmètre. Une manière d'introduire ces énergies est de se placer dans le cadre de la géométrie de Finsler [42] qui repose sur l'utilisation d'un couple d'anisotropies (ϕ, ϕ^o).

Dans ce contexte, on considère d'abord le cas d'une fonction $\phi : \mathbb{R}^d \to \mathbb{R}$ strictement positive en dehors de l'origine, convexe et positivement 1-homogène, c'est-à-dire

$$\phi(\lambda\xi) = |\lambda|\phi(\xi),\ \forall(\lambda, \xi) \in \mathbb{R} \times \mathbb{R}^d.$$

L'anisotropie ϕ^o correspond au dual de ϕ obtenue par sa transformée de Legendre :

$$\phi^o(\xi^*) = \sup_{\xi} \{\xi \cdot \xi^*,\ \phi(\xi) \leq 1\}.$$

Avec cette définition, ϕ^o vérifie les mêmes propriétés que ϕ à savoir, la positivité, l'homogénéité et la convexité.

Le périmètre anisotrope est défini à partir de ce couple d'anisotropies de la manière suivante :

$$P_\phi(\Omega) = \int_{\partial\Omega} \phi^o(n(x))d\mathcal{H}^{d-1}(x),$$

où $n(x)$ désigne la normale au point $x \in \partial\Omega$. En particulier, lorsque $\phi(\xi) = \|\xi\|$ alors $\phi^o = \phi$ et le périmètre anisotrope s'identifie au périmètre classique.

Plus généralement, pour un couple (ϕ, ϕ^o) d'anisotropie quelconque, le périmètre anisotrope préserve certaines propriétés du cas isotrope [42]. En particulier, la convexité de ϕ^o permet de montrer que P_ϕ est bien semi-continu inférieurement pour la topologie issue de la convergence L^1 des fonctions caractéristiques des domaines de $\mathbb{R}^d$.

Il est intéressant d'introduire la forme de Wulff (unitaire) B_ϕ définie par

$$B_\phi = \left\{\xi \in \mathbb{R}^d; \phi(\xi) \leq 1\right\}$$

et dont l'intérêt est qu'elle minimise à volume fixé le périmètre anisotrope [43]. Dans le cas isotrope où $\phi(\xi) = \phi^o(\xi) = \|\xi\|$, on retrouve la sphère comme forme de Wulff.

Le diagramme de Frank B_{ϕ^o}, quant à lui, permet de représenter l'anisotropie avec

$$B_{\phi^o} = \left\{\xi \in \mathbb{R}^d; \phi^o(\xi) \leq 1\right\},$$

qui correspond aussi à une sphère dans le cas isotrope.

Les exemples les plus simples d'anisotropies sont les anisotropies elliptiques de la forme $\phi^o(\xi) = \sqrt{A\xi \cdot \xi}$ où A est une matrice symétrique réelle définie positive de taille $d \times d$. La forme de Wulff et le diagramme de Frank correspondent dans ce cas à des ellipsoïdes. Un exemple est donné en Figure 12.1.

Dans cette thèse, nous nous intéresserons aussi aux exemples des normes ℓ^p dans $\mathbb{R}^d$ avec $\phi = \|\cdot\|_p$ et $\phi^o = \|\cdot\|_q$, où $1 \leq p \leq \infty$ et $\frac{1}{p} + \frac{1}{q} = 1$. La figure 12.2 donne la forme de Wulff et le diagramme de Frank pour un tel couple d'anisotropies.

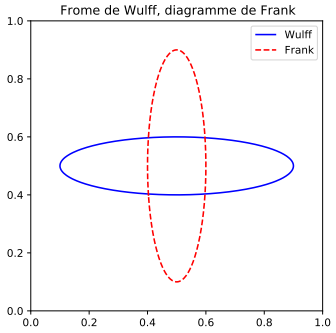


FIGURE 12.1 – Bord de la forme de Wulff et bord du diagramme de Frank pour l'anisotropie $\phi(\xi) = \left(\frac{\xi_1}{2}\right)^2 + (2\xi_2)^2$.

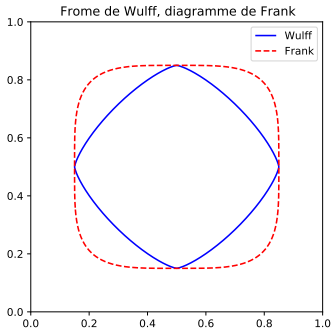


FIGURE 12.2 – Bord de la forme de Wulff et bord du diagramme de Frank pour l'anisotropie $\phi = \|\cdot\|_{\frac{4}{3}}$.

On se place à présent dans un cadre régulier où ϕ et ϕ^o sont supposées de classe C^2 sur $\mathbb{R}^d \setminus \{0\}$ afin de définir la notion de normale anisotrope n_ϕ et de courbure anisotrope H_ϕ. Pour cela, il est utile d'introduire la distance anisotrope signée d_ϕ à un domaine Ω, définie de la manière suivante,

$$d_\phi(x,\Omega) = \inf\{\mathrm{dist}_\phi(x,y),\ y \in \Omega\} - \inf\left\{\mathrm{dist}_\phi(x,y),\ y \in \mathbb{R}^d \setminus \Omega\right\}$$

où

$$\mathrm{dist}_\phi(x,y) = \inf_\gamma \left\{ \int_0^1 \phi(\gamma'(t))dt,\ \gamma \in W^{1,1}([0,1];\mathbb{R}^d),\ \gamma(0) = x,\ \gamma(1) = y \right\}.$$

La courbure anisotrope s'obtient alors en généralisant la propriété $H = \mathrm{div}(n)$ avec,

$$H_\phi = \mathrm{div}(n_\phi),$$

où n_ϕ est le vecteur de Cahn-Hoffman donné par $n_\phi = \phi^o_\xi(\nabla d_\phi)$ avec ϕ^o_ξ le gradient de ϕ^o. Nous renvoyons le lecteur intéressé à la référence [42] pour une analyse plus détaillée de ces différentes quantités géométriques.

On peut montrer que, comme dans le cas isotrope, cette définition de H_ϕ apparaît dans la dérivée de forme du périmètre anisotrope (voir par exemple [42, 33]) avec

$$\frac{d}{dt} P_\phi(\Omega_t)|_{t=0} = -\int_{\partial\Omega} \phi^o(n) H_\phi(x) h(x) d\mathcal{H}^{d-1}(x),$$

où $\Omega_t = \Phi_t(\Omega)$ et $\Phi_t(x) = x + th(x)n_\phi(x)$.

La définition du mouvement par courbure moyenne anisotrope s'obtient alors comme le flot de gradient L^2 du périmètre anisotrope et consiste à faire évoluer l'interface $\partial\Omega$ selon la vitesse normale V_n,

$$V_n = \phi^o(n) H_\phi, \tag{12.0.1}$$

où ϕ^o intervient comme une mobilité naturelle $\mu(n) = \phi^o(n)$ associée au flot.

Il est alors intéressant de remarquer que la courbure anisotrope est constante sur les formes de Wulff et la forme est conservée lorsque l'on considère un mouvement par courbure moyenne anisotrope. Plus précisément, en considérant la forme de Wulff de rayon caractéristique R,

$$B_{\phi,R} = \{\xi \in \mathbb{R}^d,\ \phi(\xi) \leq R\},$$

l'évolution par courbure moyenne anisotrope associée à la condition initiale B_{ϕ,R_0} est une forme de Wulff $B_{\phi,R(t)}$ de rayon $R(t)$ dont le rayon caractéristique vérifie

$$R(t) = \sqrt{R_0^2 - 2t}.$$

Ce flot canonique va nous permettre, comme dans le cas isotrope, de construire une base d'apprentissage explicite sur laquelle nos réseaux seront entraînés.

12.1 Approximation et méthode de champ de phase

L'extension de l'approximation de Cahn-Hilliard dans le cas anisotrope a par exemple été étudié dans l'article [42], où les auteurs montrent en particulier la Γ-convergence de l'énergie P_{ε,ϕ^o}, définie par

$$P_{\varepsilon,\phi^o}(u) = \int_{\mathbb{R}^d} \left(\frac{\varepsilon}{2} \phi^o \left(\nabla u\right)^2 + \frac{1}{\varepsilon} W(u) \right) dx, \tag{12.1.1}$$

vers le périmètre anisotrope P_ϕ (modulo une constante multiplicative).

Le mouvement par courbure moyenne anisotrope peut alors s'approcher en considérant le flot de gradient L^2 de P_{ε,ϕ^o}, qui conduit à l'équation d'Allen-Cahn anisotrope suivant,

$$\partial_t u = \Delta_\phi u - \frac{1}{\varepsilon^2} W'(u), \tag{12.1.2}$$

où le Laplacien anisotrope $\Delta_\phi u$ est défini par l'expression suivante :

$$\Delta_\phi u = \operatorname{div} \left(\nabla_\xi \phi^o(\nabla u) \phi^o(\nabla u) \right).$$

Cette approximation a été étudiée théoriquement et numériquement dans différents contextes : des anisotropies régulières [42, 91, 175], des anisotropies cristallines [34], et des simulations dans le cas d'anisotropies non convexes [161].

En particulier, des résultats de convergence de l'équation d'Allen-Cahn anisotrope vers le mouvement par courbure moyenne anisotrope ont été établis dans [32]. Par ailleurs, Bellettini et Paolini ont montré dans [42] qu'il s'agit d'un modèle d'ordre 2 dont le développement asymptotique de la solution u_ε proche de l'interface est de la forme

$$u_\varepsilon = q\left(\frac{d_\phi(x,\Omega)}{\varepsilon} \right) + \mathcal{O}(\varepsilon^2).$$

On retrouve ainsi un résultat similaire à l'équation d'Allen-Cahn classique où la distance signée isotrope a été remplacée par son homologue anisotrope.

Cette approche peut traiter des anisotropies très variées, mais la résolution numérique de l'équation (12.1.2) est généralement bien plus difficile que dans le cas isotrope. La difficulté ici vient du traitement numérique du laplacien anisotrope Δ_ϕ qui est généralement non linéaire et même non elliptique pour des anisotropies ϕ^o non convexes. En particulier, on aboutit très souvent à des schémas dont le coût algorithmique est bien plus élevé que celui des méthodes numériques du cas isotrope.

Il existe néanmoins une situation pour laquelle la résolution numérique ne pose pas de difficultés particulières : le cas d'anisotropies elliptiques. Il s'agit de modèles dont l'anisotropie est de la forme $\phi^o(\xi) = \sqrt{A\xi \cdot \xi}$ où A est une matrice symétrique définie positive de taille $d \times d$ et dont les formes de Wulff et diagrammes de Frank associés sont des ellipsoïdes. Dans ce cas particulier et uniquement dans ce cas, le laplacien anisotrope est bien linéaire et s'identifie à

$$\Delta_\phi u = \operatorname{div}(A \nabla u).$$

Il peut donc se traiter très simplement dans la base de Fourier comme dans le cas isotrope et conduit à des noyaux de la forme

$$K_{\phi^o,t}(x) = \mathcal{F}^{-1}\left[\xi \mapsto e^{-4\pi^2\langle A\xi,\xi\rangle t}\right](x)$$

où $\mathcal{F}$ désigne la transformée de Fourier.

12.2 Équation d'Allen-Cahn linéarisée et approche bidomaine

Afin de pallier la difficulté du traitement numérique du laplacien anisotrope Δ_ϕ, Bonnetier et al [56] ont proposé de le linéariser dans la base de Fourier. Ils ont introduit pour cela un nouvel opérateur $\tilde{\Delta}_\phi$ qui a l'avantage d'être linéaire et qui correspond simplement à la multiplication par le symbole

$$\sigma(\xi) = -4\pi^2\phi^o(\xi)^2,$$

dans le domaine de Fourier. Cela conduit alors à l'équation d'Allen-Cahn anisotrope linéarisée,

$$u_t = \tilde{\Delta}_\phi u - \frac{1}{\varepsilon^2}W'(u), \tag{12.2.1}$$

où l'opérateur Δ_ϕ a été remplacé par $\tilde{\Delta}_\phi$.

La consistance de ce modèle a été démontrée dans [56] sous une version correspondant à un schéma de type Bence- Merriman-Osher (BMO) où l'opération de convolution a été remplacée par l'opérateur $\tilde{\Delta}_\phi$. En particulier, comme nous le verrons plus loin en section 12.3 la positivité du noyau $K(x) = \mathcal{F}^{-1}\left[e^{-4\pi^2\phi(\xi)^2}\right](x)$ dans le domaine de Fourier a des implications importantes pour la formulation énergétique de l'algorithme de BMO.

Un autre avantage de leur approche est qu'elle permet de traiter des anisotropies très variées, en dimension 2 mais aussi en dimension plus grande.

L'équation (12.2.1) peut aussi être vue comme le flot-gradient L^2 de l'énergie suivante :

$$\tilde{P}_{\varepsilon,\phi}(u) = \int_{\mathbb{R}^d}\frac{\varepsilon}{2}4\pi^2\phi^o(\xi)^2|\mathcal{F}[u](\xi)|^2d\xi + \int_{\mathbb{R}^d}\frac{1}{\varepsilon}W(u(x))dx.$$

Cependant, la Γ-convergence de $\tilde{P}_{\epsilon,\phi}$ vers le périmètre anisotrope reste une question ouverte en général. Elle peut toutefois être prouvée pour des cas particuliers d'anisotropies en exploitant une approche bidomaine [17, 10].

Nous illustrons cette approche en considérant une anisotropie ϕ^o qui s'exprime à l'aide d'une moyenne harmonique faisant intervenir des anisotropies elliptiques :

$$\phi^o(\xi)^2 = \frac{1}{(A_1\xi\cdot\xi)^{-1} + (A_2\xi\cdot\xi)^{-1}} = \frac{(A_1\xi\cdot\xi)(A_2\xi\cdot\xi)}{(A_1\xi\cdot\xi) + (A_2\xi\cdot\xi)},$$

où A_1 et A_2 sont deux matrices symétriques définies positives.

L'intérêt d'une telle anisotropie est que l'énergie de Cahn Hilliard $\tilde{P}_{\varepsilon,\phi}$ peut se réécrire sous la forme

$$\tilde{P}_{\varepsilon,\phi}(u) = \min_{u_1+u_2=u}\left\{\frac{\varepsilon}{2}\int_{\mathbb{R}^d}A_1\nabla u_1\cdot\nabla u_1 + A_2\nabla u_2\cdot\nabla u_2\,dx\right\} + \int_{\mathbb{R}^d}\frac{1}{\varepsilon}W(u)\,dx,$$

En effet, il suffit de remarquer que l'équation d'Euler-Lagrange associée à ce problème de minimisation s'écrit

$$\begin{cases}\operatorname{div}(A_1\nabla u_1^*) + \lambda^* = 0\\ \operatorname{div}(A_2\nabla u_2^*) + \lambda^* = 0\end{cases}$$

où λ^* est le multiplicateur de Lagrange associé à la contrainte $u_1^* + u_2^* = u^*$. On en déduit en particulier que

$$\operatorname{div}(A_1\nabla u_1^*) = \operatorname{div}(A_2\nabla u_2^*).$$

Il vient alors, en notant $\hat{u} = \mathcal{F}[u]$ la transformée de Fourier de u, que

$$\begin{aligned}
&\underset{u_1+u_2=u}{\operatorname{argmin}} \left\{ \frac{\varepsilon}{2} \int_{\mathbb{R}^d} A_1 \nabla u_1 \cdot \nabla u_1 + A_2 \nabla u_2 \cdot \nabla u_2 \, dx \right\} \\
= \;& \frac{\varepsilon}{2} \int_{\mathbb{R}^d} A_1 \nabla u_1^* \cdot \nabla u_1^* + A_2 \nabla u_2^* \cdot \nabla u_2^* \, dx = \frac{\varepsilon}{2} \int_{\mathbb{R}^d} A_1 \nabla u_1^* \cdot \nabla u^* \, dx = \frac{\varepsilon}{2} \int_{\mathbb{R}^d} 4\pi^2 A_1 \xi \hat{u_1^*} \cdot \xi \overline{\hat{u^*}} \, d\xi \\
= \;& \frac{\varepsilon}{2} \int_{\mathbb{R}^d} 4\pi^2 A_1 \xi \cdot \xi \frac{A_2 \xi \cdot \xi}{A_1 \xi \cdot \xi + A_2 \xi \cdot \xi} \hat{u^*} \overline{\hat{u^*}} \, d\xi = \int_{\mathbb{R}^d} \frac{\varepsilon}{2} 4\pi^2 \phi^o(\xi)^2 |\mathcal{F}[u^*](\xi)|^2 d\xi
\end{aligned}$$

où l'avant-dernière égalité s'obtient en utilisant l'égalité $\operatorname{div}((A_1 + A_2)\nabla u_1^*) = \operatorname{div}(A_2 \nabla u^*)$ dans l'espace de Fourier.

On retrouve ainsi l'égalité désirée,

$$\tilde{P}_{\varepsilon,\phi}(u) = \min_{u_1+u_2=u} \left\{ \frac{\varepsilon}{2} \int_{\mathbb{R}^d} A_1 \nabla u_1 \cdot \nabla u_1 + A_2 \nabla u_2 \cdot \nabla u_2 \, dx \right\} + \int_{\mathbb{R}^d} \frac{1}{\varepsilon} W(u) \, dx.$$

En exploitant la forme de l'anisotropie, on a ainsi pu reformuler l'énergie $\tilde{P}_{\varepsilon,\phi}$ sous une forme dont la Γ-convergence vers le périmètre anisotrope associé à l'anisotropie

$$\phi^o(\xi) = \frac{1}{\sqrt{(A_1\xi \cdot \xi)^{-1} + (A_2\xi \cdot \xi)^{-1}}}$$

a été montrée dans [17].

Notons de plus que le flot de gradient L^2 associé à $\tilde{P}_{\varepsilon,\phi}$ s'écrit aussi sous la forme suivante,

$$\begin{cases} \partial_t u = \nabla \cdot A_1 \nabla u_1 - \frac{1}{\varepsilon^2} W'(u), \\ \partial_t u = \nabla \cdot A_2 \nabla u_2 - \frac{1}{\varepsilon^2} W'(u), \end{cases}$$

où $u = u_1 + u_2$. Ce système a en particulier été étudié par Bellettini et al dans [10] où ils montrent, en utilisant des développements asymptotiques formels, que ce système d'équations conduit à une approximation d'un mouvement par courbure anisotrope avec un ordre de précision en $\mathcal{O}(\varepsilon)$ seulement.

Plus généralement, une preuve de Γ-convergence [17] peut s'obtenir de la même façon pour des anisotropies ϕ^o qui s'écrivent à l'aide d'une moyenne harmonique faisant intervenir N anisotropies elliptiques :

$$\phi^o(\xi)^2 = \frac{1}{\sum_{i=1}^N (A_i\xi \cdot \xi)^{-1}}$$

Comme précédemment, il suffit de remarquer que

$$\tilde{P}_{\varepsilon,\phi}(u) = \min_{\sum_{i=1}^N u_i = u} \left\{ \frac{\varepsilon}{2} \sum_{i=1}^N \int_{\mathbb{R}^d} A_i \nabla u_i \cdot \nabla u_i \, dx \right\} + \int_{\mathbb{R}^d} \frac{1}{\varepsilon} W(u) \, dx. \tag{12.2.2}$$

En effet, l'équation d'Euler-Lagrange s'écrit

$$\operatorname{div}(A_i \nabla u_i^*) + \lambda^* = 0, \; \forall i \in \{1, \cdots, N\}.$$

Ce qui implique que

$$\operatorname{div}(A_i \nabla u_i^*) = \operatorname{div}\left(A_j \nabla u_j^*\right) \quad \text{et} \quad \hat{u_i^*}(\xi) = \frac{(A_i\xi \cdot \xi)^{-1}}{\sum_{j=1}^N (A_j\xi \cdot \xi)^{-1}} \hat{u^*}(\xi),$$

où $u^* = \sum_{i=1}^N u_i^*$. On en déduit que

$$\frac{\varepsilon}{2} \sum_{i=1}^N \int_{\mathbb{R}^d} A_i \nabla u_i^* \cdot \nabla u_i^* dx = \frac{\varepsilon}{2} \sum_{i=1}^N \int_{\mathbb{R}^d} A_1 \nabla u_1^* \cdot \nabla u_i^* dx = \frac{\varepsilon}{2} \int_{\mathbb{R}^d} A_1 \nabla u_1^* \cdot \nabla u^* dx,$$

et on obtient bien

$$\min_{\sum_{i=1}^N u_i = u} \left\{ \frac{\varepsilon}{2} \sum_{i=1}^N \int_{\mathbb{R}^d} A_i \nabla u_i \cdot \nabla u_i dx \right\} = \frac{\varepsilon}{2} \int_{\mathbb{R}^d} 4\pi^2 A_1 \xi \cdot \xi \hat{u^*}_1 \overline{\hat{u^*}} d\xi = \frac{\varepsilon}{2} \int_{\mathbb{R}^d} \frac{1}{\sum_{i=1}^N (A_i\xi \cdot \xi)^{-1}} |\hat{u^*}|^2 d\xi,$$

qui nous permet de retrouver l'égalité souhaitée (12.2.2).

Le flot de gradient L^2 se récrit alors sous la forme

$$\begin{cases} \partial_t u = \nabla \cdot A_1 \nabla u_1 - \frac{1}{\epsilon^2} W'(u), \\ \partial_t u = \nabla \cdot A_2 \nabla u_2 - \frac{1}{\epsilon^2} W'(u), \\ \dots \\ \partial_t u = \nabla \cdot A_N \nabla u_N - \frac{1}{\epsilon^2} W'(u), \end{cases}$$

où $u = \sum_{i=1}^N u_i$.

Ce type de modèles a par exemple été étudié dans les travaux [10, 9]. En particulier, des choix opportuns d'anisotropies A_i permettent aussi de traiter le cas d'anisotropies non convexes pour lesquelles les laplaciens de Finsler classiques perdent leur ellipticité.

Remarque 12.2.1. *Une question serait alors de savoir s'il est possible ou non d'approcher de manière uniforme n'importe quel type d'anisotropie ϕ^o. En d'autre terme, $\forall \sigma > 0$, existe t-il un entier N_σ suffisamment grand tel que*

$$|\phi^o(\xi) - \phi^o_{N_\sigma}(\xi)| \leq \sigma |\xi|,$$

où $\phi^o_{N_\sigma}$ s'obtiendrait comme la somme de N_σ anisotropies elliptiques. Une idée serait de remarquer que

$$\phi^o(\xi)^{-2} = \int_{\mathcal{S}^{d-1}} \frac{1}{A_\theta \xi \cdot \xi} d\theta,$$

si A_θ est par exemple une matrice dégénérée avec pour valeur $\phi^o(\theta)^2$ dans la direction e_θ et $+\infty$ dans les directions orthogonales à e_θ. Une discrétisation de cette formule couplée à une approximation A_θ permettrait ainsi d'approcher n'importe quel type d'anisotropies y compris des anisotropies non convexes !

L'intérêt d'un tel formalisme est aussi de faire le lien avec des schémas de résolution utilisant en parallèle plusieurs réseaux de neurones de type $S^{\mathrm{NN}}_{\theta,1}$ ou $S^{\mathrm{NN}}_{\theta,2}$.

12.3 Algorithme de Bence-Merriman-Osher et noyau anisotrope

Il est intéressant de rappeler dans ce chapitre quelques résultats récents sur l'adaptation de l'algorithme de Bence-Merriman-Osher (BMO) au mouvement par courbure moyenne anisotrope et notamment le lien entre les noyaux d'anisotropies utilisés pour traiter une anisotropie ϕ^o et une mobilité μ données.

Dans [147, 141, 146], Esedoglu, Elsey et Otto se sont intéressés à la généralisation de l'algorithme de BMO à différents contextes avec des jeux de tensions de surfaces qui pouvaient dépendre de l'orientation du domaine. En se basant sur les travaux de [200], Esedoglu et al ont étudié une variante de l'algorithme de BMO qui consiste à générer une suite d'ensembles $(\Omega^n)_{n\geq 0}$ en utilisant la récurrence suivante :

$$\Omega^{n+1} = \left\{ K_{\delta_t} * \mathbb{1}_{\Omega^n} \geq \frac{1}{2} \right\} \tag{12.3.1}$$

où $K_{\delta_t}(x) = \frac{1}{\delta_t^{d/2}} K(x/\sqrt{\delta_t})$. Ici, le noyau K n'est plus nécessairement le noyau de la chaleur mais il vérifie les hypothèses suivantes

$$K(x) = K(-x), \quad \int_{\mathbb{R}^d} K(x)dx = 1 \quad \text{et} \quad \int_{\mathbb{R}^d} |x|^2 K(x)dx < \infty,$$

afin d'assurer la consistance du schéma précédent.

Esedoglu, Elsey et Otto ont introduit un point de vue variationnel en montrant que l'ensemble Ω^{n+1} peut aussi s'obtenir comme le minimiseur de

$$\tilde{E}_{\delta_t}(\Omega) = \frac{1}{\delta_t} \left(E_{\delta_t}(\Omega) + \frac{1}{\delta_t} \int (1_\Omega - 1_{\Omega^k}) K_{\delta_t} * (1_\Omega - 1_{\Omega^k}) dx \right),$$

où

$$E_{\delta_t}(\Omega) = \frac{1}{\delta_t} \int (1 - 1_\Omega) K_{\delta_t} * 1_\Omega dx.$$

L'avantage est alors de pouvoir identifier facilement une mobilité μ_K et une anisotropie ϕ^o_K associées au mouvement généré par l'algorithme (12.3.1) en étudiant la limite de chacun des termes précédents lorsque

δ_t tend vers 0. Plus précisément, Esedoglu, Elsey et Otto ont montré tout d'abord que l'énergie non locale $E_{\delta_t}(\Omega)$ tend vers un périmètre anisotrope P_K associé à l'anisotropie ϕ^o_K définie par

$$\phi^o_K(n) = -\frac{1}{\pi}\text{P.F}\int_{\mathbb{R}} \frac{\hat{K}(n\xi)}{\xi^2}d\xi = -\frac{1}{2\pi}\int_{\mathbb{R}} \frac{\hat{K}(n\xi)-\hat{K}(0)}{\xi^2}d\xi.$$

où P.F désigne la partie finie.

De plus, ils ont vérifié que

$$\lim_{\delta_t\to 0}\lim_{t\to 0}\frac{\sqrt{\delta_t}}{t^2}\int (1_{\Omega(t)}-1_{\Omega(0)})K_{\delta_t} * (1_{\Omega(t)}-1_{\Omega(0)})dx = \int_{\partial\Omega(0)}\frac{1}{\mu_K(n(x))}\phi(x)^2 d\mathcal{H}^{d-1}(x),$$

où la mobilité μ_K est définie par

$$\mu_K(n) = \frac{2\pi}{\int_{\mathbb{R}}\hat{K}(n\xi)d\xi}.$$

Avec cette approche, on peut facilement identifier le flot limite anisotrope dont la vitesse normale est donnée par

$$V_n = \mu_K(n)H_\phi,$$

où l'anisotropie ϕ^o_K et la mobilité μ_K s'expriment simplement à partir du noyau K utilisé. Cette approche permet aussi d'obtenir des résultats de stabilité en remarquant que $\tilde{E}_{\delta_t}$ peut également s'obtenir comme une linéarisation de E_{δ_t} au voisinage de l'ensemble Ω. En particulier, la stabilité de l'algorithme de Bence-Merriman-Osher est vérifiée sous des hypothèses de concavité de E_{δ_t}, ce qui est vrai pour des noyaux K dont la transformée de Fourier $\hat{K}$ est positive.

Inversement, la difficulté est alors de trouver, à partir d'une anisotropie ϕ^o et d'une mobilité μ données, un noyau K qui vérifie les relations

$$\phi^o_K(n) = -\frac{1}{2\pi}\int_{\mathbb{R}}\frac{\hat{K}(n\xi)-\hat{K}(0)}{\xi^2}d\xi \quad \text{et} \quad \mu_K(n) = \frac{2\pi}{\int_{\mathbb{R}}\hat{K}(n\xi)d\xi}. \tag{12.3.2}$$

Par exemple, dans le cas par particulier de la mobilité naturelle $\mu_K(n) = \phi^o(n)$, il est alors possible d'utiliser le noyau $K(x) = \mathcal{F}^{-1}[\exp(-4\pi^2\phi^o(\xi)^2)](x)$ dérivant des travaux de Bonnetier et al [56]. L'avantage d'un tel noyau est qu'il est positif dans l'espace de Fourier. Il assure donc la décroissance de l'énergie E_{δ_t} au cours du temps. En revanche, ces noyaux ne sont généralement pas positifs dans le domaine spatial, ce qui implique une perte de monotonie des schémas numériques associés.

Nous pensons qu'une approche par réseaux de neurones peut permettre d'identifier d'autres noyaux, avec de meilleures propriétés numériques, et ainsi corriger leur erreur de modélisation de champ de phase en ε.

Un autre intérêt, certainement encore plus intéressant, serait d'essayer d'identifier l'anisotropie et la mobilité à partir de données expérimentales correspondant à des flots par courbure moyenne anisotrope. En effet, il suffit dans ce cas d'estimer le noyau K d'un réseau de neurones $S^{\text{NN}}_{1,\theta}$ puis d'appliquer les formules précédentes.

12.4 Structures de réseaux et base de données pour l'identification des lois d'anisotropie

On reproduit ici une démarche analogue à celle utilisée dans le cas isotrope. Notre objectif est donc d'approcher, à l'aide d'un réseau de neurones $\mathcal{S}^{\text{NN}}_\theta$, le semi-groupe $\mathcal{S}^q_{\phi^o,dt,\varepsilon}$ défini par

$$\mathcal{S}^q_{\phi,dt,\varepsilon}\left[v_\varepsilon(\cdot,t)\right] = v_\varepsilon(\cdot,t+\delta_t)$$

où nous avons noté $v_\varepsilon(\cdot,t) = q\left(\dfrac{d(\cdot,\Omega_\phi(t))}{\varepsilon}\right)$. Ici, q représente le profil classique associé au potentiel double puits, la fonction d désigne la distance signée ***isotrope*** et $(\Omega_\phi(t))_{t\geq 0}$ correspond à un mouvement par courbure moyenne anisotrope pour un couple d'anisotropies (ϕ,ϕ^o) donné.

Notons que nous n'utilisons pas la forme optimale observée dans le développement asymptotique des solutions d'Allen-Cahn anisotrope, i.e

$$u_\varepsilon(x,t) = q\left(\frac{d_\phi(x,\Omega_\phi(t))}{\varepsilon}\right) + \mathcal{O}(\varepsilon^2).$$

Nous avons choisi de remplacer la distance anisotrope signée par la distance signée classique afin que l'épaisseur de l'interface diffuse de dépende pas de l'orientation du domaine $\Omega(t)$ à chaque instant t de l'évolution. Cette propriété est en particulier très intéressante dans un contexte multiphase.

Nous espérons alors que l'entraînement du réseau S_θ^{NN} soit suffisamment concluant pour que le schéma numérique,

$$u^{n+1} = S_\theta^{\mathrm{NN}}\left[u^n\right],$$

à partir de la donnée initiale

$$u^0 = q\left(\frac{d(\cdot,\Omega_\phi(0))}{\varepsilon}\right),$$

permette d'obtenir une bonne approximation du mouvement par courbure moyenne anisotrope de $\Omega_\phi(0)$.

12.4.1 Structures de réseaux

Les réseaux que nous utilisons dans ce chapitre sont similaires aux réseaux étudiés dans le chapitre sur le mouvement par courbure moyenne isotrope, c'est-à-dire des réseaux obtenus en alternant des neurones de convolution et des neurones de réaction. En effet, les modèles champ de phase des flots par courbure moyenne anisotrope étant là-aussi des équations de type Allen-Cahn, il parait naturel d'utiliser ces réseaux qui ont déjà donné de très bons résultats dans le cas isotrope. Les deux réseaux qui nous intéressent ici sont donc $S_{\theta,1}^{\mathrm{NN}}$ et $S_{\theta,2}^{\mathrm{NN}}$ dont les architectures sont rappelées en figure 12.3.

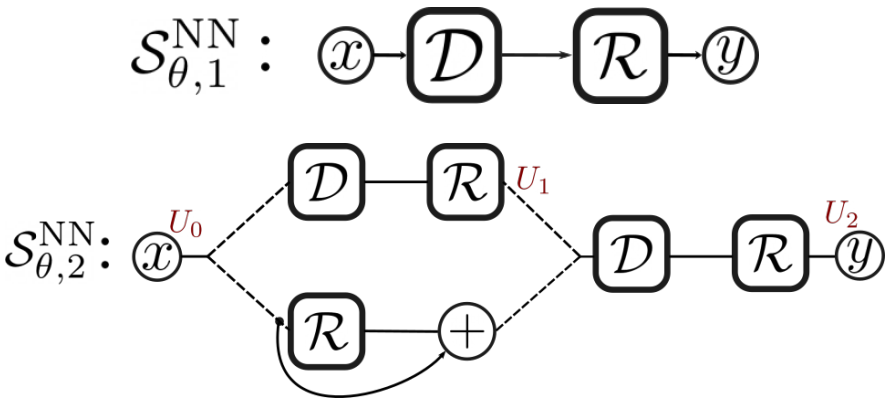


FIGURE 12.3 – Réseaux de neurones inspirés des schémas numériques pour l'équation d'Allen-Cahn

Les neurones de réaction restent identiques au cas isotrope. En revanche, concernant les neurones de diffusion, la taille des noyaux pourra être ajustée en fonction de l'anisotropie considérée.

12.4.2 Base de données et formes de Wulff

Dans cette sous-section, nous explicitons la base de données que nous avons utilisé dans nos expériences numériques. Encore une fois, nous exploitons la représentation champ de phase d'évolutions d'interface générées de manière exacte sans avoir recours à l'équation d'Allen-Cahn anisotrope. Nous utilisons pour cela l'évolution des formes de Wulff dont nous savons que le rayon caractéristique $R(t)$ vérifie

$$R(t) = \sqrt{R_0^2 - 2t}$$

où R_0 est le rayon caractéristique de la forme de Wulff initiale. Nous pouvons ainsi construire une première base de données à partir d'une famille finie de rayons R_i avec $i \in \{1,\cdots,N_1\}$, en considérant les données,

$$\{X_i, Y_{i,j}\}_{1\leq j\leq k} = \left\{\varphi_{R_i}, \varphi_{\sqrt{R_i^2-2j\delta_t}}\right\}_{1\leq j\leq k}, \quad \forall i \in \{1,\cdots,N_1\},$$

où on a posé $\varphi_R(x) = q\left(\frac{d(x,B_{\phi,R})}{\varepsilon}\right)$ avec $B_{\phi,R}$ la forme de Wullf de rayon R centrée à l'origine du repère. Comme dans le cas isotrope, le pas de temps est fixé à $\delta_t = \varepsilon^2$.

Afin d'approcher le semi-groupe $S^q_{\phi,dt,\varepsilon}$, nous utilisons la distance signée classique d au lieu de la distance signée anisotrope d_ϕ pour générer nos données.

Ainsi, l'entraînement d'un réseau $\mathcal{S}_\theta$ consiste à minimiser l'énergie suivante

$$J_k(\theta) = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \sum_{j=1}^{k} \left\| (\mathcal{S}_\theta)^{(j)}[X_i] - Y_{i,j} \right\|^2 = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \sum_{j=1}^{k} \int_Q \left((\mathcal{S}_\theta)^{(j)}[\phi_{R_i}] - \varphi_{\sqrt{R_i^2 - 2j\delta_t}} \right)^2 dx,$$

où $(\mathcal{S}_\theta)^{(j)}$ est la composition itérée j fois de $\mathcal{S}_\theta$.

La comparaison sur plusieurs itérations permet de stabiliser l'apprentissage du réseau. Dans toute la suite, on fixe $k = 5$, ce qui signifie que nous comparons les sorties du réseau sur $k = 5$ itérations.

Enfin, l'étape de validation est similaire au cas isotrope où la métrique de validation mesure l'erreur commise sur l'évolution du volume d'une forme de Wulff donnée. Nous rappelons que durant cette étape aucun paramètre du réseau n'est modifié, elle sert à choisir les hyper-paramètres. Dans les expériences numériques qui suivent, cette étape est appliquée de manière périodique toutes les 100 époques. En pratique, elle nous permet de choisir les tailles des noyaux pour lesquelles la métrique de validation donne le meilleur score.

12.5 Validation de la méthode sur une anisotropie régulière

Dans cette section nous testons la capacité de nos réseaux à apprendre le mouvement par courbure moyenne anisotrope orienté pour le couple d'anisotropies (ϕ, ϕ^o) où $\phi = \|\cdot\|_{4/3}$ est la norme de $\ell^{\frac{4}{7}3}$ dans $\mathbb{R}^2$ et $\phi^o = \|\cdot\|_4$ est sa norme duale.

L'objectif de l'entraînement est alors de permettre à nos réseaux d'approcher le semi-groupe $S^q_{\phi,dt,\varepsilon}$ où nous fixons les paramètres $\varepsilon = \delta_x$ et $\delta_t = \varepsilon^2$, avec δ_x désignant le pas d'espace. Dans cette expérience, les noyaux associés aux réseaux de diffusion ont tous une taille fixée à 33×33.

Nous présentons en figure 12.4 l'évolution de l'énergie J_k au cours du processus d'apprentissage avec en bleu la courbe d'erreur d'entraînement du réseau $S^{\text{NN}}_{\theta,1}$ et en orange celle de $S^{\text{NN}}_{\theta,2}$. Nous observons dans les deux cas une décroissance des courbes d'erreurs d'entraînement avec une convergence vers des erreurs plus petites pour le réseau $S^{\text{NN}}_{\theta,2}$.

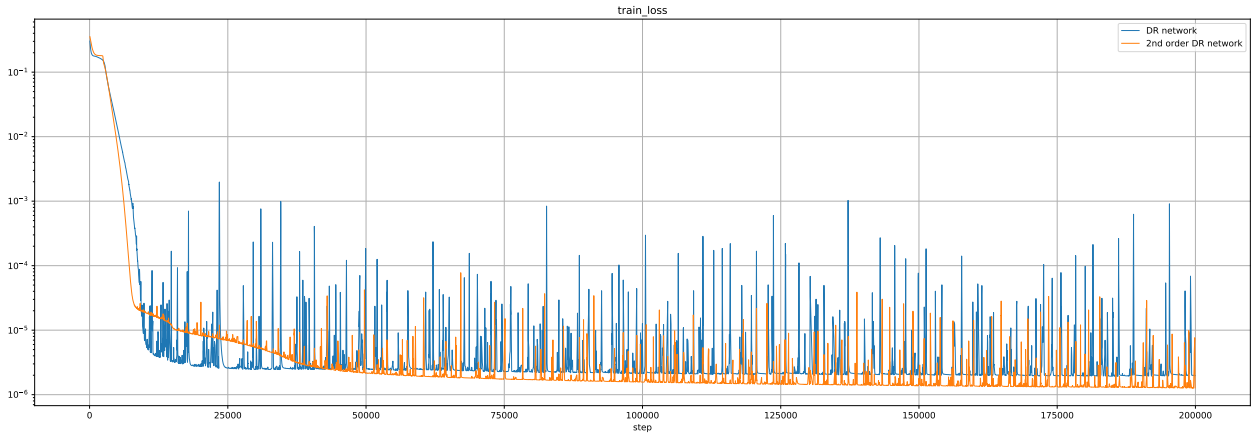


FIGURE 12.4 – Processus d'apprentissage par optimisation de $J_k(\theta)$ avec $k = 5$. Courbes d'erreur d'entraînement des réseaux $S^{\text{NN}}_{\theta,1}$ (en bleu) et $S^{\text{NN}}_{\theta,2}$ (en orange).

Validation quantitative sur l'évolution des formes de Wulff

Nous voulons maintenant vérifier que les deux réseaux entraînés donnent des approximations suffisamment précises du mouvement par courbure moyenne anisotrope associé à l'anisotropie $\phi = \|\cdot\|_{4/3}$. Pour cela, nous commençons par tester les schémas $u^{n+1} = \mathcal{S}^{\mathrm{NN}}_{\theta,i}[u^n]$, $i = 1, 2$ sur la condition initiale $u^0 = q\left(\frac{d(\cdot,\Omega_\phi(0))}{\varepsilon}\right)$ où $\Omega_\phi(0)$ est la forme de Wulff de rayon caractéristique $R = 0.3$ centrée en l'origine.

Plus précisément, nous représentons en figure 12.5 l'évolution exacte en représentation champ de phase (première ligne), et les approximations numériques u^n, à différents temps $t_n = n\delta_t$, en utilisant le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ pour l'équation d'Allen-Cahn linéarisée (deuxième ligne), les réseaux $\mathcal{S}^{\mathrm{NN}}_{\theta,1}$ (troisième ligne) et $\mathcal{S}^{\mathrm{NN}}_{\theta,2}$ (quatrième ligne).

Remarque 12.5.1. *Le splitting de Lie pour l'équation d'Allen-Cahn linéarisée consiste à alterner la résolution d'une équation de diffusion*

$$u_t = \tilde{\Delta}_\phi u,$$

(en appliquant simplement le filtre $\sigma(\xi) = e^{-4\pi^2\phi^o(\xi)^2\delta_t}$ dans l'espace de Fourier) avec la résolution de l'équation de réaction,

$$u_t = -\frac{1}{\varepsilon^2}W'(u),$$

avec les mêmes techniques que dans le cas isotrope.

Les évolutions sont très similaires et nous observons dans chacun des cas que la forme de Wulff est bien préservée au cours de temps. Nous pouvons confirmer sur cet exemple que les deux réseaux fournissent de manière qualitative l'évolution attendue.

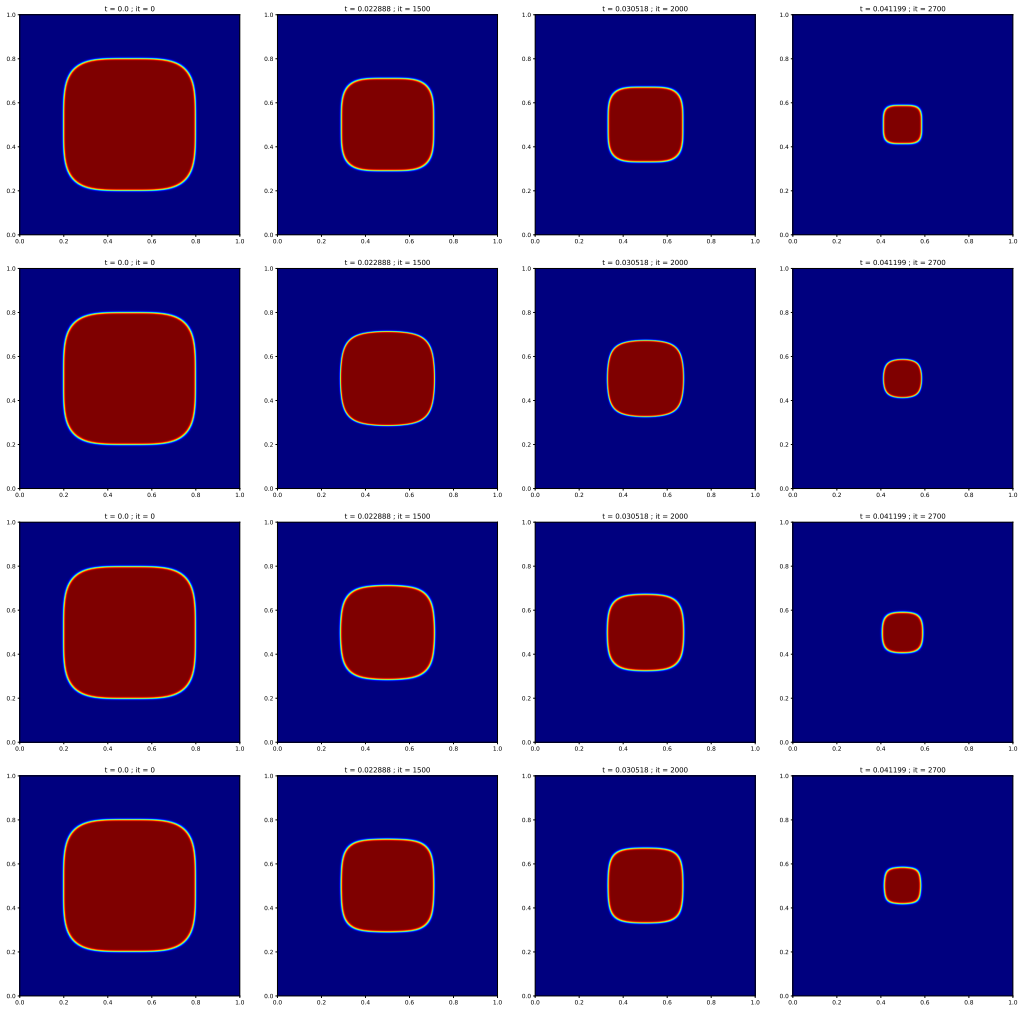


FIGURE 12.5 – Simulation du mouvement par courbure moyenne anisotrope d'une forme de Wulff, en utilisant respectivement le flot exact en représentation champ de phase, le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$, les réseaux entraînés $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$.

Afin d'avoir des résultats plus quantitatifs, nous étudions l'erreur commise par rapport à l'intégrale de la fonction champ de phase exacte $\int \varphi_R(x)\,dx$ sur l'évolution d'une forme de Wulff. Plus précisément, nous traçons en figure 12.6, l'erreur $n \mapsto \left|\int_Q u^n - \varphi_{R(n\delta_t)}\,dx\right|^2$ sur plusieurs itérations en utilisant

— le réseau $S^{\mathrm{NN}}_{\theta,1}$ en bleu
— le réseau $S^{\mathrm{NN}}_{\theta,2}$ en orange
— le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ en vert

et en partant de la condition initiale $u^0 = \varphi_{R_0} = q\left(\frac{d(\cdot,\Omega_\phi(0)}{\varepsilon}\right)$ où $\Omega_\phi(0)$ est la forme de Wulff de rayon caractéristique $R_0 = 0.3$ centrée en l'origine. Les résultats montrent clairement que nos deux réseaux $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$ fournissent des schémas numériques plus précis que le splitting de Lie. Nous voyons de plus sur cette comparaison que l'entraînement de nos réseaux a permis de corriger à la fois les erreurs de discrétisation numérique et d'approximation champ de phase de l'équation d'Allen-Cahn (au moins dans le cas linéarisé).

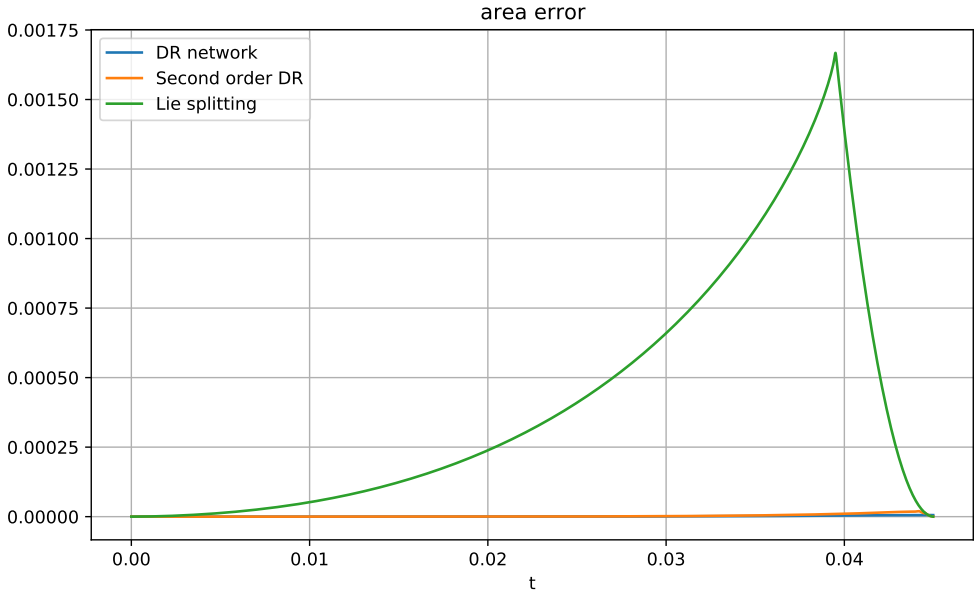


FIGURE 12.6 – Comparaison des différents schémas sur l'évolution $n \mapsto \left(\int_Q u^n - \varphi_{R(n\delta_t)}\,dx\right)^2$ en utilisant les réseaux entraînés $S^{\mathrm{NN}}_{\theta,1}$ (courbe bleue), $S^{\mathrm{NN}}_{\theta,2}$ (courbe orange) et le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ (en vert).

Validation qualitative à partir d'une forme initiale quelconque

Nous voulons à présent valider que nos réseaux sont capables de se généraliser à de nouvelles données jamais utilisées durant l'apprentissage. Nous comparons alors en figure 12.7 l'évolution d'une forme initiale en X arrondi en utilisant respectivement le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ et les réseaux $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$. Cette expérience montre clairement que nos réseaux permettent de reproduire l'évolution attendue même sur une forme initiale qui n'était pas dans la base d'entraînement.

Forme de Wulff et identification de la loi d'anisotropie à partir des noyaux K

Dans cette sous-section, nous voulons évaluer la capacités de nos réseaux à apprendre l'anisotropie $\phi = \|\cdot\|_{4/3}$. Une manière qualitative de juger de la pertinence de notre apprentissage est de coupler nos schémas à une contrainte de volume. Ainsi, en partant par exemple d'un cercle comme condition initiale, nous savons que celui-ci devrait converger vers la forme de Wulff de même aire. Cette expérience permet alors de voir si nous pouvons en effet retrouver l'anisotropie à travers les formes optimales obtenues grâce à ce procédé.

Nous adoptons une approche similaire au cas isotrope en considérant le schéma suivant,

1. On applique le réseau $S^{\mathrm{NN}}_{\theta,i}$:

$$u^{n+1/2} = S^{\mathrm{NN}}_{\theta,i}\left[u^n\right]$$

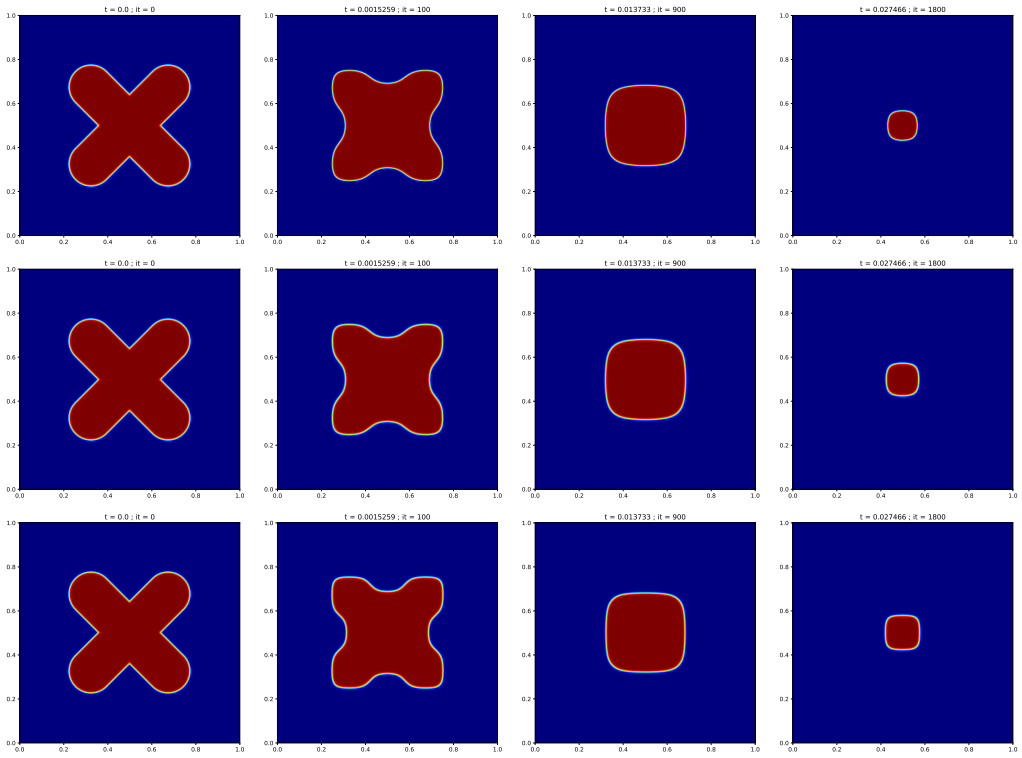


FIGURE 12.7 – Simulation du mouvement par courbure moyenne anisotrope d'une forme en X arrondi, en utilisant respectivement $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$, $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$.

2. On applique une projection sur la contrainte de volume :

$$u^{n+1} = u^{n+1/2} + \lambda^{n+1}\sqrt{W(u^{n+1/2})}$$

où nous rappelons que W est le potentiel double puits. Ici, $i = 1$ ou 2 selon que l'on souhaite évaluer le réseau $S^{\mathrm{NN}}_{\theta,1}$ ou $S^{\mathrm{NN}}_{\theta,2}$.

Nous comparons notre approche avec le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$. Pour cela, nous utilisons le même schéma que précédemment en appliquant cette fois-ci $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ à la place du réseau.

La figure 12.8 représente l'évolution du disque de rayon $R = 0.3$ en utilisant respectivement le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$, les réseaux $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$. En trait jaune, nous affichons le bord de la forme de Wulff de même volume que le disque de rayon $R = 0.3$. Chaque évolution montre bien que le disque converge effectivement vers une bonne approximation de la forme de Wulff dans les trois cas.

Nous allons plus loin dans l'analyse des anisotropies obtenues par nos réseaux et nous voulons identifier plus précisément l'anisotrope induite des noyaux entraînés, au moins pour le réseau $S^{\mathrm{NN}}_{\theta,1}$. Pour cela, nous reprenons les travaux de Esedoglu, Elsey et Otto [147, 141, 146] sur l'identification de l'anisotropie à partir des noyaux de convolution des schémas de type BMO. Rappelons comme nous l'avons écrit en début de section que ces auteurs se sont intéressés à une version de l'algorithme de BMO qui consiste à générer une suite d'ensembles $(\Omega^n)_{n\geq 0}$ en utilisant la récurrence

$$\Omega^{n+1} = \left\{ K_{\delta_t} * \mathbb{1}_{\Omega^n} \geq \frac{1}{2} \right\} \tag{12.5.1}$$

où le noyau K n'est plus nécessairement le noyau de la chaleur.

Comme déjà dit, le flot limite lorsque $\delta_t \to 0$ correspond au flot par courbure moyenne anisotrope dont la vitesse à l'interface est

$$V_n = \mu_K(n) H_\phi,$$

où l'anisotropie ϕ^o_K et la mobilité μ_K s'expriment à partir du noyau K utilisé. Plus précisément [147, 141, 146] :

$$\phi^o_K(n) = -\frac{1}{\pi}\mathrm{P.F}\int_{\mathbb{R}} \frac{\hat{K}(n\xi)}{\xi^2} d\xi = -\frac{1}{2\pi}\int_R \frac{\hat{K}(n\xi) - \hat{K}(0)}{\xi^2} d\xi.$$

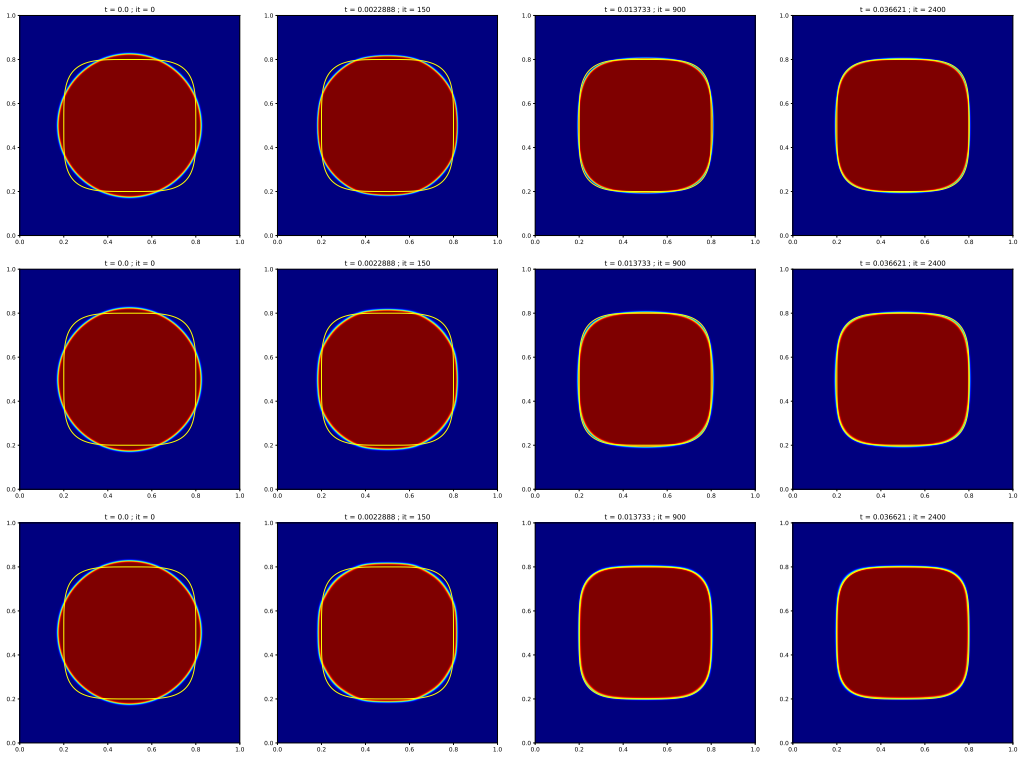


FIGURE 12.8 – Simulation du mouvement par courbure moyenne anisotrope avec conservation d'aire d'un cercle, en utilisant respectivement le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ et les réseaux entraînés $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$. En trait plein jaune, nous représentons le bord de la forme de Wulff de même aire que le cercle initial.

et

$$\mu_K(n) = \frac{2\pi}{\int_{\mathbb{R}} \hat{K}(n\xi)d\xi}.$$

Nous appliquons ces formules (ou plus précisément une approximation discrète dans le domaine de Fourier) afin d'identifier l'anisotropie apprise par $S^{\mathrm{NN}}_{\theta,1}$ (première ligne, figure 12.9), que nous comparons à celle calculée à partir du splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ (deuxième ligne figure 12.9). Sur chaque ligne de la figure 12.9, nous affichons respectivement le noyau dans le domaine spatial, dans le domaine de Fourier et enfin la forme de Wulff calculée à partir du noyau (en trait bleu) comparée à la forme de Wulff exacte (en trait orange). Nous observons que les noyaux ont des formes assez similaires. Notons cependant que le noyau du réseau $S^{\mathrm{NN}}_{\theta,1}$ autorise les valeurs négatives, ce qui n'est pas le cas du noyau de splitting de Lie.

La forme de Wulff calculée à partir du noyau de splitting $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ est très proche de la forme de Wulff attendue. Dans le cas du réseau $S^{\mathrm{NN}}_{\theta,1}$, bien que certaines zones ne soient pas totalement bien approchées, nous observons que le réseau est capable de retrouver globalement la forme de Wulff souhaitée. Un apprentissage plus poussé (en nombre d'itérations par exemple) permettrait peut-être de trouver une meilleure approximation de la forme de Wulff attendue.

Ces résultats sont très encourageants et montrent tout l'intérêt des réseaux de neurones pour l'apprentissage du flot de courbure moyenne anisotrope alors qu'on ne connaît que partiellement l'anisotropie via les formes contenues dans la base de données d'apprentissage. Ils illustrent également le fait que nos réseaux de neurones conduisent à des schémas simples, stables et souvent très précis. Les noyaux appris permettent non seulement d'approcher correctement le mouvement souhaité mais aussi de retrouver une bonne approximation de l'anisotropie étudiée.

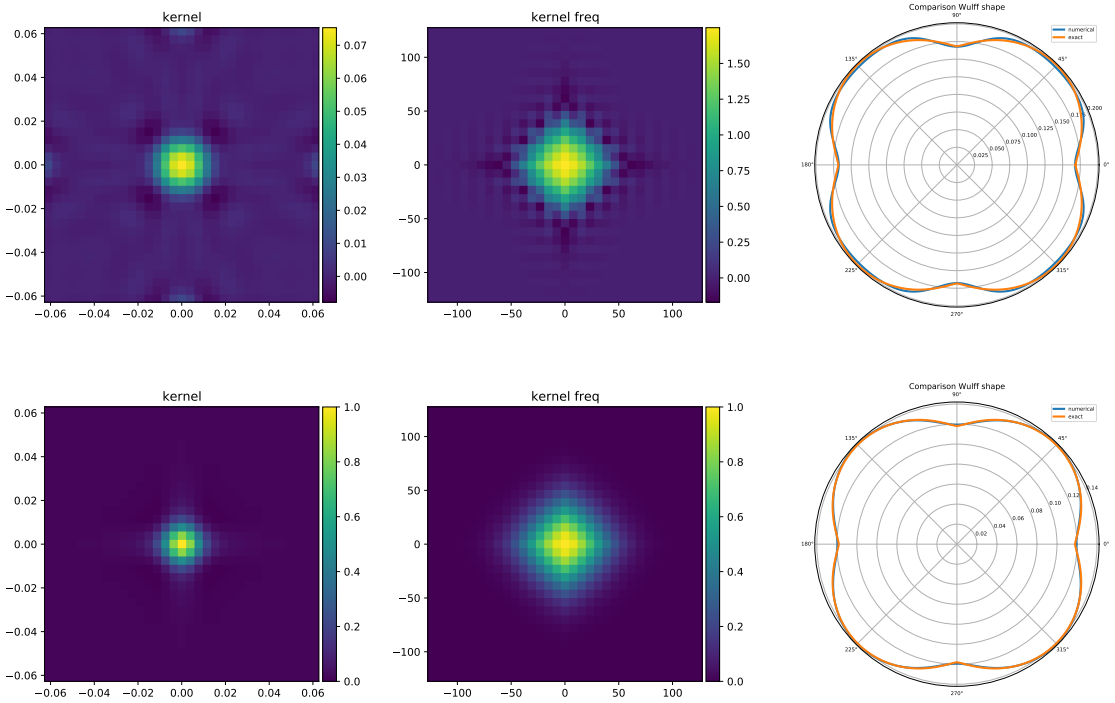


FIGURE 12.9 – Chaque ligne montre, de gauche à droite, le noyau dans le domaine spatial, dans le domaine fréquentiel et sur la même figure la forme de Wulff calculée à partir du noyau (en trait bleu) et la forme de Wulff exacte (en trait orange). La première ligne correspond au réseau $S^{\mathrm{NN}}_{\theta,1}$ et la seconde au splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$.

12.6 Cas d'une anisotropie cristalline

Dans cette section, nous tentons d'apprendre, à l'aide de nos réseaux, le mouvement par courbure moyenne anisotrope pour une anisotropie cristalline, i.e. une anisotropie associée à des formes de Wulff et des diagrammes de Frank dont les bords sont polygonaux. La faible régularité des formes de Wulff pose un certain nombre de difficultés numériques dans le traitement de l'anisotropie. En outre, ces formes de Wulff peuvent être associées à différentes anisotropies non convexes (dont les enveloppes convexes correspondent à ϕ^o) et il n'est pas clair que les réseaux parviennent à retrouver exactement ces anisotropies. Enfin, le dernier point intéressant est que les formes de Wulff ne font intervenir qu'un nombre limité de normales, ce qui peut poser un problème d'apprentissage avec une base de données ne faisant jamais intervenir certaines directions. On peut ainsi tester la capacité de nos réseaux à généraliser ou non le bon flot pour ces directions nouvelles.

L'un des exemples les plus classiques d'anisotropies cristallines est celui du couple d'anisotropies (ϕ, ϕ^o) où $\phi = \|\cdot\|_\infty$ est la norme infinie et $\phi^o = \|\cdot\|_1$ la norme 1 sur $\mathbb{R}^2$. Nous allons essayer d'apprendre le flot associé à ce couple. Ici, la taille des noyaux des réseaux est fixée à 17×17.

L'entraînement de nos réseaux s'effectue comme précédemment avec des courbes d'entraînement qui convergent vers des erreurs d'apprentissage acceptables.

Validation et évolution d'une forme de Wulff

La figure 12.10 illustre un premier exemple d'évolution sur une forme de Wulff où nous comparons encore une fois le flot exact (première ligne), le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ (deuxième ligne), et les réseaux $S^{\mathrm{NN}}_{\theta,1}$ (troisième ligne) et $S^{\mathrm{NN}}_{\theta,2}$ (quatrième ligne). Cet exemple montre clairement que l'évolution obtenue à partir du réseau d'ordre 2 $S^{\mathrm{NN}}_{\theta,2}$ est plus précise que celles obtenue avec le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ et le réseau $S^{\mathrm{NN}}_{\theta,1}$. On observe en effet que le réseau $S^{\mathrm{NN}}_{\theta,2}$ conserve des coins très marqués et la forme de Wulff semble très bien préservée au cours de l'évolution.

Anisotropie et approximation de la forme de Wulff

Nous avons pu voir que le réseau $S^{\mathrm{NN}}_{\theta,2}$ permettait d'avoir une très bonne approximation de l'évolution attendue sur les formes de Wulff. Nous voulons à présent évaluer sa capacité à apprendre la bonne anisotropie. Nous adoptons alors la même approche que précédemment en simulant le mouvement par courbure moyenne anisotrope avec contrainte d'aire. Nous comparons en figure 12.11 les formes optimales obtenues respectivement en utilisant le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ et le réseau $S^{\mathrm{NN}}_{\theta,2}$. Nous observons que les formes optimales sont légèrement différentes. Dans le premier cas, la forme d'équilibre est assez proche de la forme de Wulff attendue, sauf aux quatre coins qui sont plus arrondis pour la forme optimale. Dans le second cas, nous voyons que la forme de Wulff est correctement approchée, notamment au niveau des coins.

Comparaison de l'évolution à partir d'une forme initiale différente d'une forme de Wulff

Nous voulons à présent tester notre réseau $S^{\mathrm{NN}}_{\theta,2}$ sur une forme éloignée de la forme de Wulff et qui n'a donc pas été observée durant la phase d'entraînement. L'objectif de cette section est donc d'illustrer la faculté de notre réseau à réussir ou non à bien généraliser le flot par courbure moyenne anisotrope appris seulement à partir de l'évolution exacte de formes de Wulff de différentes tailles.

Nous proposons donc de comparer le schéma numérique dérivant de notre réseau avec celui obtenu par l'approche Fourier-splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$. La condition initiale utilisée ici correspond à la forme de Wulff tournée d'un angle de $\pi/4$. Les flots obtenus sont présentés en figure 12.12 où sont affichées, sur chacune des lignes, les solutions u^n à différents instants t en utilisant respectivement les schémas $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ et $S^{\mathrm{NN}}_{\theta,2}$. Comme nous pouvons facilement le constater, les flots sont très différents. En effet, dans le cas de la méthode de splitting, le carré est rongé au niveau des angles jusqu'à obtenir la forme de Wulff approchée alors que dans le cas du réseau $S^{\mathrm{NN}}_{\theta,2}$, les sommets sont fixés et ce sont au contraire les côtés qui grossissent, jusqu'à converger aussi vers une forme de Wulff mais de rayon caractéristique plus grand que dans le premier cas. Nous avons utilisé dans cette expérience une résolution deux fois plus fine pour augmenter la précision de ces deux simulations numériques. En pratique, on utilise toujours le même réseau mais on l'applique sur un domaine de calcul deux fois plus grand.

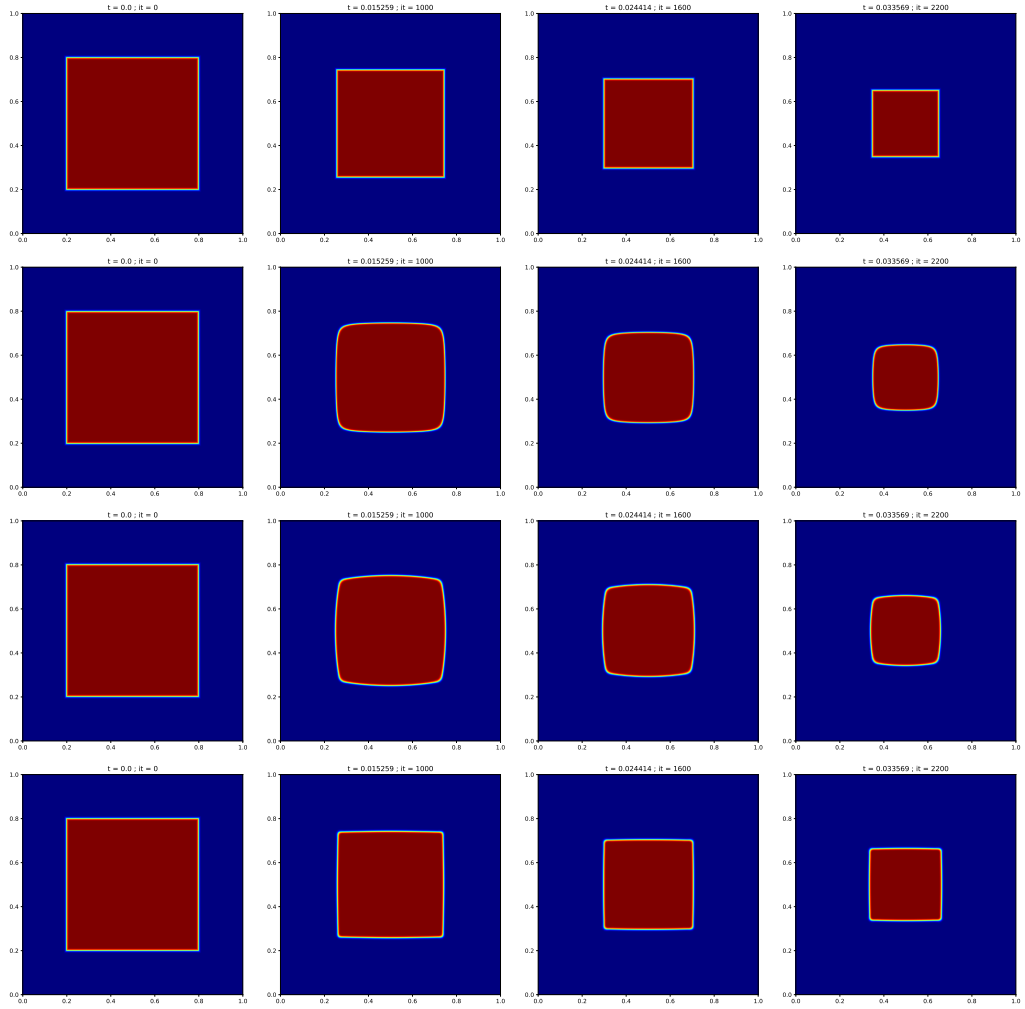


FIGURE 12.10 – Simulation du mouvement par courbure moyenne anisotrope d'une forme de Wulff, en utilisant le flot exact en représentation champ de phase (première ligne), le splitting de Lie $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$ (deuxième ligne) et les réseaux entraînés $S^{\mathrm{NN}}_{\theta,1}$ (troisième ligne) et $S^{\mathrm{NN}}_{\theta,2}$ (quatrième ligne).

Ces résultats illustrent l'importance de la base d'apprentissage pour aider le réseau à bien généraliser le flot par courbure moyenne anisotrope. En effet, dans le cas pratique qu'on a étudié, le réseau n'avait jamais rencontré de directions obliques dans son entraînement, et il a donc dû apprendre à généraliser son action en présence de ces directions. On n'obtient pas le comportement attendu puisque le réseau déplace les interfaces dans des directions non représentées dans les formes de Wulff alors qu'il ne devrait pas. Comme dans le cas du flot de Willmore, ce résultat pourrait indiquer qu'il est sans doute nécessaire, pour apprendre correctement des flots cristallins, d'introduire dans la base d'apprentissage des évolutions d'ensembles faisant intervenir le plus grand nombre de directions possibles.

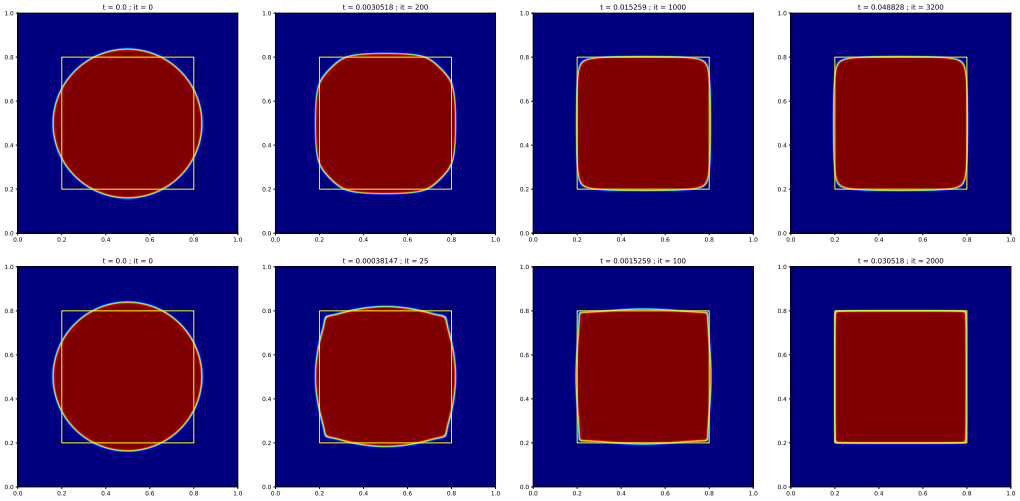


FIGURE 12.11 – Simulation du mouvement par courbure moyenne anisotrope avec conservation d'aire. Première ligne : $S^{\mathrm{AC}}_{\delta_t=\varepsilon^2,\varepsilon,1}$. Deuxième ligne : $S^{\mathrm{NN}}_{\theta,2}$. Le trait plein jaune représente le bord de la forme de Wulff de même aire que la forme initiale.

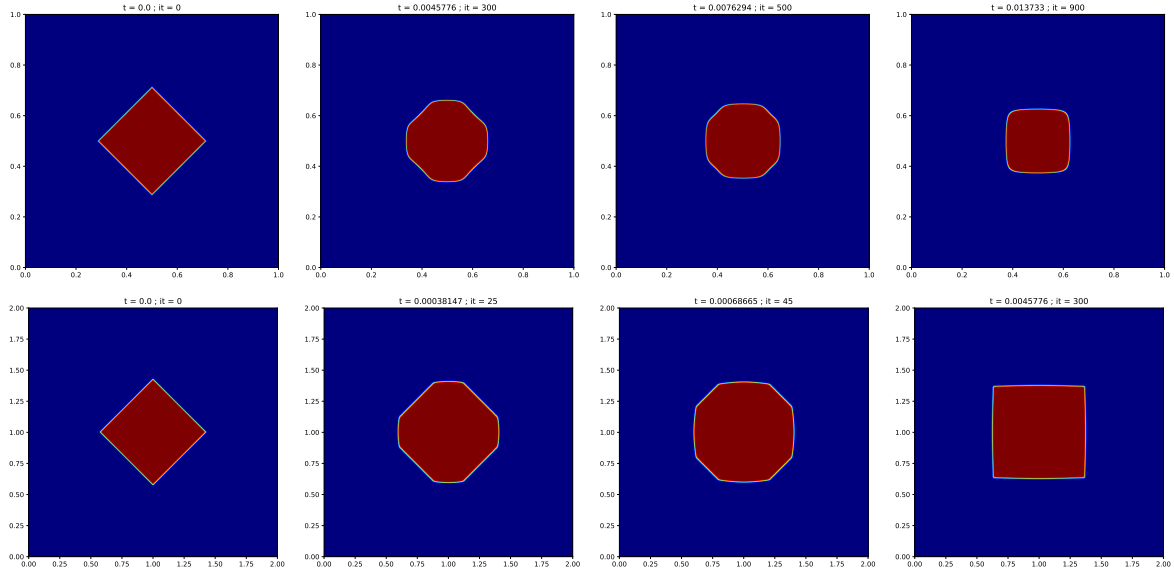


FIGURE 12.12 – Simulations de l'évolution d'une forme de Wulff tournée d'un angle de $\pi/4$, obtenues respectivement à partir du splitting de Lie et du réseau $S^{\mathrm{NN}}_{\theta,2}$.

12.7 Apprentissage d'une anisotropie non symétrique

Dans cette section, nous nous intéressons au cas d'une fonction d'anisotropie ϕ non symétrique c'est-à-dire ne vérifiant pas la propriété $\phi(-\xi) = \phi(\xi),\ \forall \xi \in \mathbb{R}^2$. Ce cas de figure sort du cadre de la théorie que nous avons présentée au début de ce chapitre car la fonction ϕ ne vérifie plus la propriété d'homogénéité d'ordre 1. Le mouvement par courbure moyenne anisotrope associé ne peut donc plus être considéré au sens que nous avons défini dans ce chapitre. Néanmoins, puisque nous connaissons ϕ, nous pouvons encore définir les formes de Wulff et considérer la même loi d'évolution sur ces dernières. Notre objectif est alors d'entraîner nos réseaux sur ces évolutions afin de voir ce qu'ils sont capables d'apprendre dans un tel contexte, en particulier pour les formes de Wulff. Quels flots va-t-on pouvoir générer ? Seront-ils géométriques ?

Cette expérience nous semble très intéressante car nous ne connaissons pas explicitement la fonction ϕ^o. C'est donc un cas de figure où l'apprentissage par réseaux pourrait donner des résultats là où il serait très difficile d'utiliser une approche basée sur l'équation linéarisée d'Allen-Cahn anisotrope.

Dans les expériences numériques qui suivent, nous considérons l'anisotropie $\phi(\xi) = \phi(r, \theta) = (1 + a\cos(k\theta))\, r$ où $r = |\xi|$ et $\theta = \arg(\xi)$ sont, respectivement le module et l'argument de ξ, avec $k = 3$ et $a = 0.124$. Un exemple de forme de Wulff associée à ϕ est donné en figure 12.13.

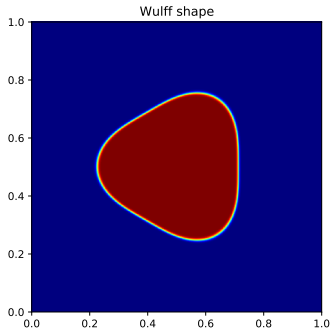


FIGURE 12.13 – Forme de Wulff en représentation champ de phase pour l'anisotropie $\phi(\xi) = \phi(r, \theta) = (1 + a\cos(k\theta))\, r$.

L'apprentissage de nos réseaux sur cette anisotropie se déroule comme précédemment où les courbes d'entraînement décroissent vers des erreurs très petites.

Validation sur l'évolution d'une forme de Wulff

Afin de valider notre approche au moins qualitativement, nous présentons dans la figure 12.14 les évolutions d'une forme de Wulff obtenues en utilisant, respectivement, la loi exacte et les flots générés par nos réseaux entraînés $S^{\mathrm{NN}}_{\theta,1}$ et $S^{\mathrm{NN}}_{\theta,2}$. Les évolutions semblent assez proches, celle engendrée par $S^{\mathrm{NN}}_{\theta,1}$ paraît néanmoins un peu plus lente. Nous constatons tout de même que les « côtés » sont arrondis pour le réseau $S^{\mathrm{NN}}_{\theta,2}$ alors qu'ils sont bien maintenus pour le réseau $S^{\mathrm{NN}}_{\theta,1}$.

Perte de la symétrie du flot

Nous comparons dans la figure 12.15 l'évolution exacte précédente avec celle obtenue à l'aide du réseau $S^{\mathrm{NN}}_{\theta,1}$ en partant d'une condition initiale dont on a modifié l'orientation en remplaçant u_0 par $1 - u_0$. L'objectif ici est de voir si le réseau maintient le bon flot indépendamment de l'orientation de l'interface. La figure 12.15 montre une différence très nette de vitesse d'évolution, le réseau induisant une vitesse nettement plus rapide. La forme de Wulff semble néanmoins être préservée au cours de l'évolution. Cet exemple montre que le mouvement obtenu à partir du réseau $S^{\mathrm{NN}}_{\theta,1}$ n'est pas géométrique dans le sens où l'évolution d'une interface dépend de son orientation.

En conclusion, nos réseaux sont capables de reproduire une assez bonne approximation du flot attendu sur les formes de Wulff. Cependant, nous observons une perte de symétrie du flot qui montre encore une fois la nécessité d'enrichir la base d'apprentissage avec d'autres données que les formes de Wulff.

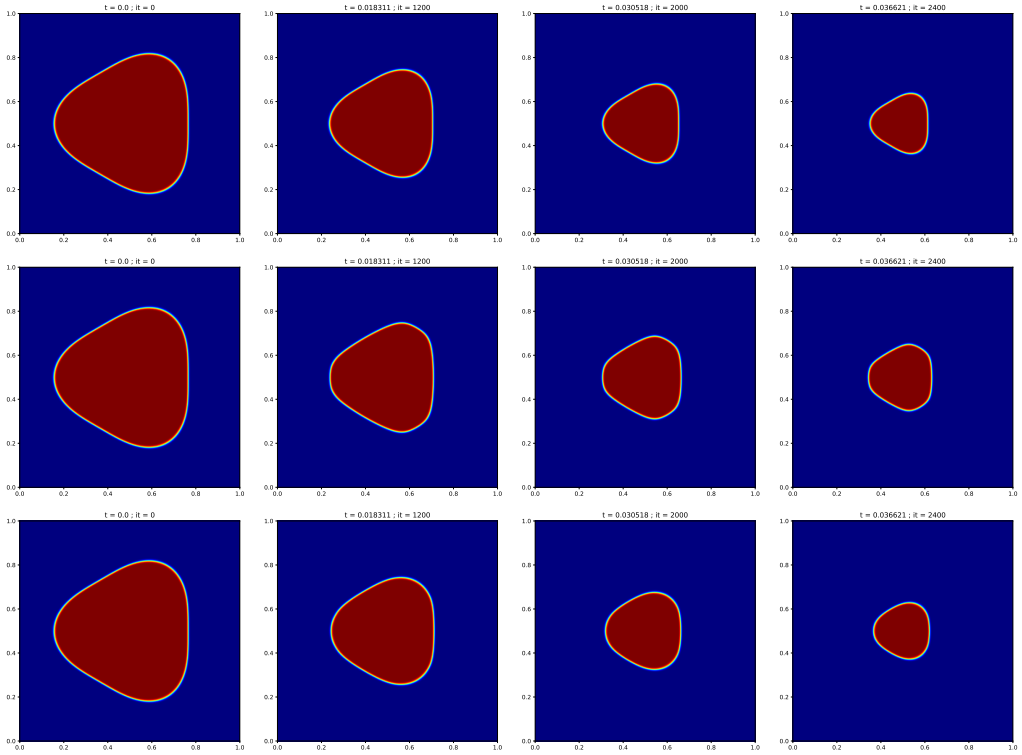


FIGURE 12.14 – Simulation de l'évolution d'une forme de Wulff obtenue, respectivement, en utilisant le flot exact (première ligne), le réseau $S^{\mathrm{NN}}_{\theta,1}$ (seconde ligne) et le réseau $S^{\mathrm{NN}}_{\theta,2}$ (dernière ligne).

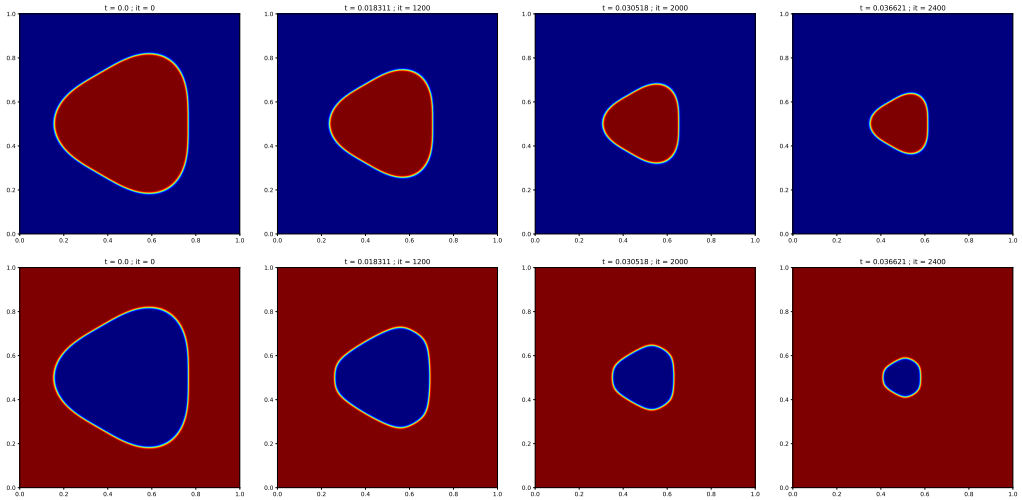


FIGURE 12.15 – Évolution d'une forme de Wulff et prise en compte de l'orientation. En haut, l'évolution exacte. En bas, l'évolution obtenue avec le réseau $S^{\mathrm{NN}}_{\theta,1}$ en partant d'une condition initiale dont on a changé l'orientation en remplaçant u_0 par $1 - u_0$.

12.8 Apprentissage d'une anisotropie régulière dans le cas non orienté

Nous nous intéressons dans cette section à l'approximation du mouvement par courbure moyenne anisotrope d'interfaces non orientables pour l'anisotropie $\phi = \|\cdot\|_{4/3}$. Comme dans le cas isotrope, le principe est d'entraîner nos réseaux sur des évolutions de formes de Wulff représentées implicitement à l'aide de fonctions champ de phase qui n'encodent pas d'information d'orientation sur le bord. Tout comme dans le cas isotrope, nous utilisons le profil q' (la dérivée de q) pour générer nos données. Ainsi, grâce à la symétrie axiale de ce profil, nous pouvons représenter nos données sans prendre en compte l'orientation de l'interface comme nous pouvons le voir en figure 12.16.

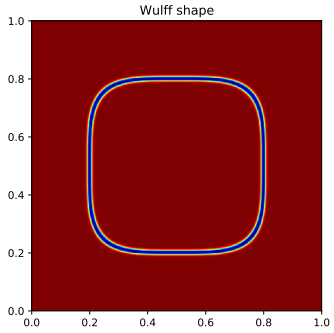


FIGURE 12.16 – Représentation champ de phase d'une forme de Wulff sans tenir compte de l'orientation du bord.

Nous rappelons, qu'à notre connaissance, il n'existe pas de modèle champ de phase pour approcher par une équation de type Allen-Cahn le flot de courbure moyenne anisotrope d'une interface qui n'est pas le bord orientable d'un domaine. L'intérêt de l'approche que nous proposons est donc de pouvoir traiter des situations, par exemple le cas des interfaces non orientables, pour lesquelles l'approche champ de phase n'est pas adaptée génériquement.

Nous avons déjà vu dans le cas isotrope que le réseau $S^{\mathrm{NN}}_{\theta,1}$ ne permettait pas d'obtenir une bonne approximation du flot par courbure moyenne d'interfaces non orientables. Les expériences numériques que nous avons menées avec ce réseau dans le cas anisotrope des résultats similaires ne seront pas détaillées. Nous présentons uniquement les résultats obtenus avec le réseau $S^{\mathrm{NN}}_{\theta,2}$.

En reprenant les notations introduites à la section 12.4, notre objectif est donc d'entraîner le réseau $S^{\mathrm{NN}}_{\theta,2}$ pour qu'il fournisse une bonne approximation du semi-groupe $\mathcal{S}^{q'}_{\phi^o,dt,\varepsilon}$. L'entraînement s'effectue sur des formes de Wulff représentées implicitement à l'aide de fonctions champ de phase basées sur le profil q'. Cet entraînement se déroule de manière analogue au cas isotrope : la fonction d'erreur décroît au cours du temps pour atteindre un taux d'erreur acceptable.

Validation qualitative sur l'évolution d'une forme de Wulff

Les simulations numériques présentées dans la figure 12.17 font penser que le réseau $S^{\mathrm{NN}}_{\theta,2}$ fournit une bonne approximation de l'évolution attendue pour une forme de Wulff, c'est une première validation positive de notre approche.

Validation sur une forme quelconque

Afin de tester la fiabilité de notre réseau $S^{\mathrm{NN}}_{\theta,2}$ pour approcher le bon flot dans un cadre non orienté et sur d'autres conditions initiales qu'une forme de Wulff, nous le testons sur une forme en X arrondi et nous comparons ses performances à celles obtenues quand on utilise une représentation champ de phase orientée. Comme on le constate, le réseau $S^{\mathrm{NN}}_{\theta,2}$ permet de bien reproduire le flot attendu, c'est une bonne indication de sa capacité à faire évoluer correctement des formes plus générales que des formes de Wulff.

Évolution d'interfaces non orientables

Toujours avec le même réseau $S^{\mathrm{NN}}_{\theta,2}$, nous testons son action sur un réseau de courbes formant un ensemble non orientable d'interfaces. Les résultats sont présentés en figure 12.19. Comme dans le cas isotrope, les points triples sont maintenus au cours de l'évolution ce qui est très surprenant car ces configurations singulières n'interviennent pas du tout dans le processus d'apprentissage. Il est cependant difficile d'évaluer

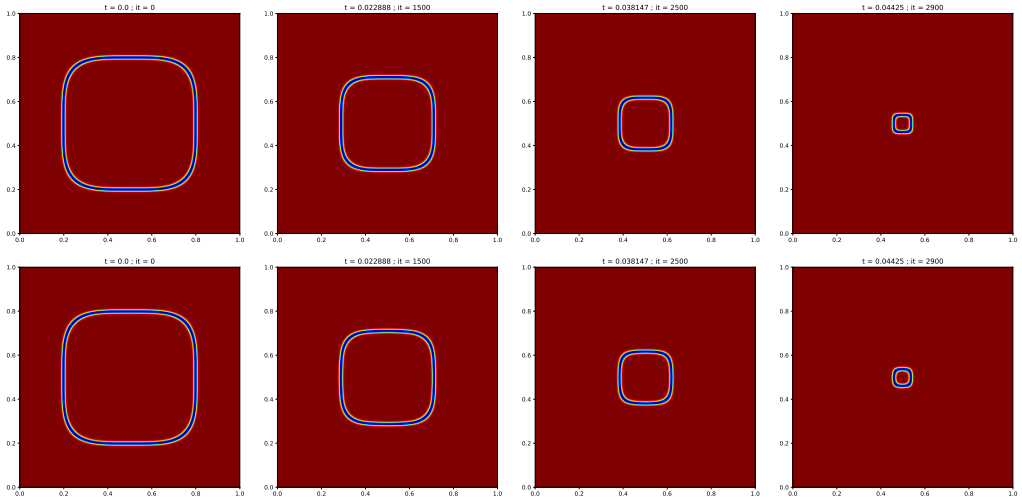


FIGURE 12.17 – Simulation du mouvement par courbure moyenne anisotrope d'une forme de Wulff en utilisant le flot exact en représentation champ de phase (première ligne) et le réseau $S^{\mathrm{NN}}_{\theta,2}$ (deuxième ligne).

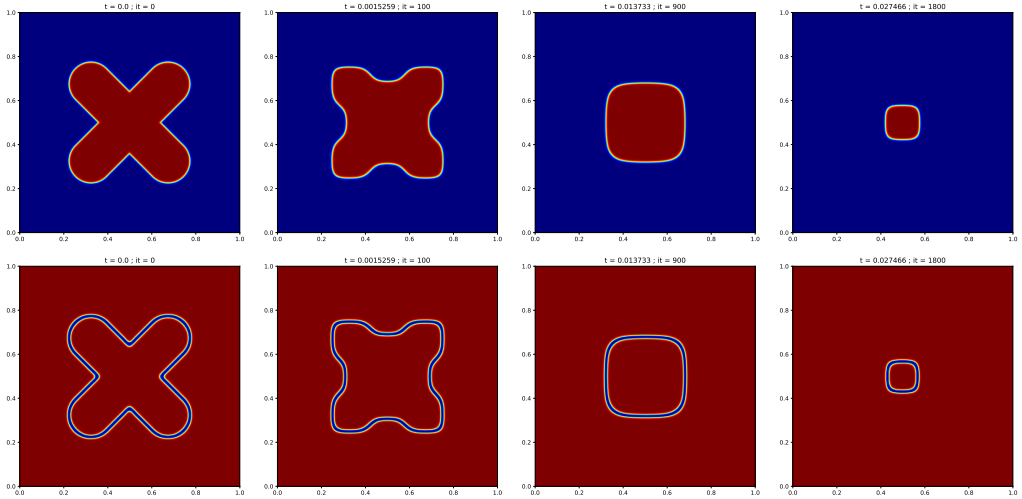


FIGURE 12.18 – Simulation à l'aide du réseau $S^{\mathrm{NN}}_{\theta,2}$ du mouvement par courbure moyenne anisotrope d'une forme en X arrondi en représentation champ de phase orientée sur la première ligne, non orientée sur la seconde ligne.

si la condition de Herring [193] est bien respectée en ces points, nous observons cependant que la condition d'angle semble dépendre de l'orientation. Cette expérience montre une nouvelle fois, comme dans le cas isotrope, la capacité de généralisation de notre réseau à des formes faisant intervenir des singularités.

Application au problème de Steiner anisotrope

La dernière expérience numérique que nous proposons vise à tester si notre réseau $S^{\mathrm{NN}}_{\theta,2}$, entraîné pour approcher le mouvement par courbure moyenne d'interfaces non orientables, est suffisamment robuste pour être couplé à des contraintes d'inclusion. Pour cela nous nous intéressons au problème de Steiner anisotrope en dimension 2 où nous remplaçons le périmètre isotrope par le périmètre P_ϕ pour l'anisotropie $\phi = \|\cdot\|_{4/3}$.

Étant donné un domaine Ω de $\mathbb{R}^2$ et un ensemble de points $a_1, \dots, a_L \in \Omega$, le problème consiste à trouver un ensemble $K \subset \Omega$ compact, connexe, contenant tous les a_i et tel que $P_\phi(K) := \int_K \phi^o(n(x))\, d\mathcal{H}^1(x)$ soit minimal.

Pour approcher les solutions de ce problème, notre idée consiste à considérer, comme dans le cas isotrope, le mouvement par courbure moyenne anisotrope non orienté $(\Gamma(t))_{t\geq 0}$ partant d'un ensemble connexe initial $\Gamma(0)$ contenant les points a_i, en imposant pour tout temps $t \geq 0$ la contrainte d'inclusion $\{a_1, \cdots, a_L\} \subset \Gamma(t)$. Nous espérons que l'état d'équilibre de ce flot soit un minimum local pour le problème de Steiner anisotrope.

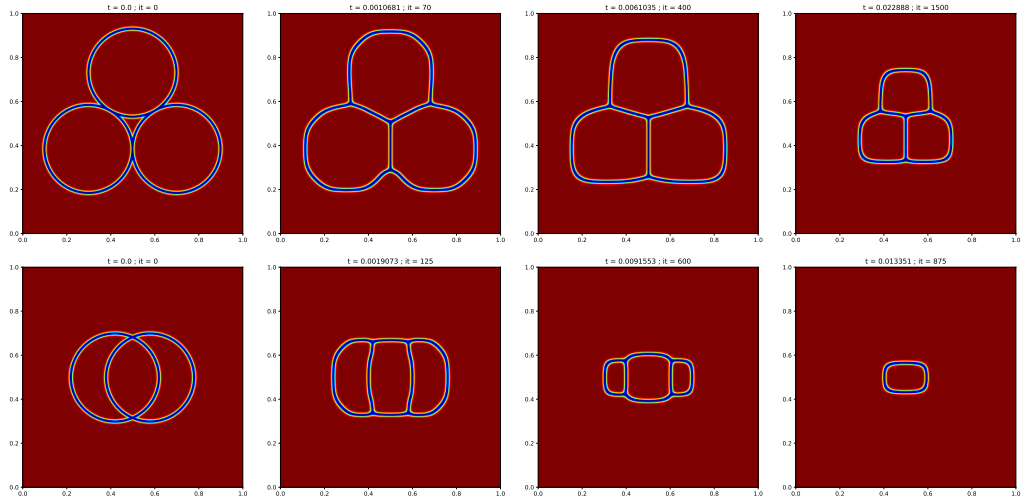


FIGURE 12.19 – Simulation à l'aide du réseau $S^{\mathrm{NN}}_{\theta,2}$ du mouvement par courbure moyenne anisotrope d'un ensemble non orientable de courbes.

Nous utilisons le réseau $S^{\mathrm{NN}}_{\theta,2}$ pour générer une suite u^n de champs de phase approchant le mouvement par courbure moyenne anisotrope non orienté couplé à une contrainte d'inclusion des points a_i qu'on impose, comme dans le cas isotrope, via l'inégalité

$$u^n \leq u_{\mathrm{in}} = \sum_{i=1}^{L} q'\left(\frac{\mathrm{dist}(\cdot, a_i)}{\varepsilon}\right),$$

Pour cela, nous considérons le schéma suivant en partant de la condition initiale $u^0 = q'\left(\frac{\mathrm{dist}(\cdot,\Gamma(0))}{\varepsilon}\right)$:

1. On applique le réseau $S^{\mathrm{NN}}_{\theta,2}$:

$$u^{n+1/2} = S^{\mathrm{NN}}_{\theta,2}[u^n].$$

2. On applique la contrainte d'inclusion :

$$u^{n+1} = \min\left(u^{n+1/2}, u_{\mathrm{in}}\right).$$

La figure 12.20 montre les approximations numériques obtenues en utilisant comme contraintes d'abord 5 puis 6 points a_i répartis non uniformément sur un cercle fixé. La première image de chaque ligne affiche l'ensemble connexe initial $\Gamma(0)$ contenant les a_i. La dernière image de chaque ligne donne la forme d'équilibre obtenue à partir du schéma précédent. Il est difficile de dire si les formes d'équilibre sont bien des solutions au problème de Steiner anisotrope car nous n'avons pas de modèle de référence pour les comparer. Ces premières expériences permettent au moins de visualiser de potentielles solutions à ce problème.

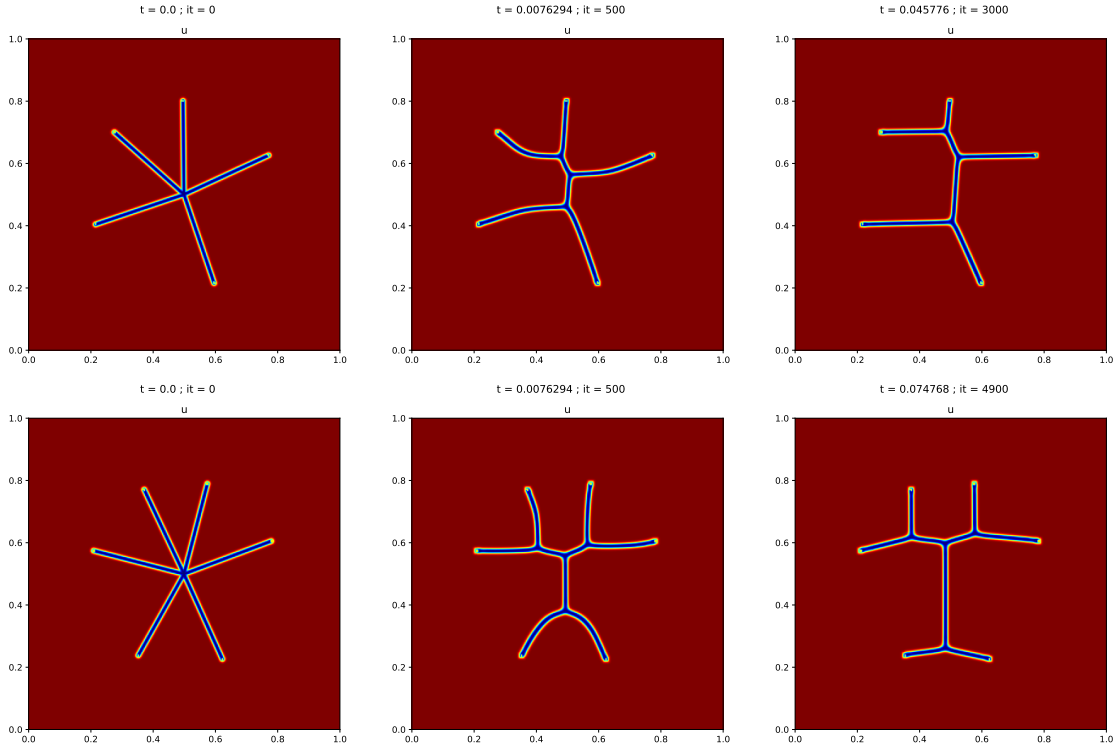


FIGURE 12.20 – Approximation d'arbres de Steiner anisotropes en 2D à l'aide du flot par courbure moyenne anisotrope non orienté couplé à des contraintes d'inclusion en suivant le schéma 12.8. Chaque ligne présente quelques étapes de l'évolution de la solution numérique u^n en imposant comme contraintes d'abord 5 puis 6 points.

Conclusion

Dans ce chapitre, nous nous sommes intéressés à l'approximation champ de phase de mouvements par courbure moyenne anisotrope par des méthodes numériques utilisant des réseaux de neurones. Une première expérience avec l'anisotropie $\phi = \|\cdot\|_{4/3}$ a montré que nos deux réseaux entraînés sont capables de reproduire le mouvement souhaité pour les formes de Wulff, et donnent aussi de très bons résultats sur des formes quelconques. Des résultats qualitatifs mais aussi quantitatifs montrent de plus que nos réseaux sont capables de retrouver de manière précise et à l'aveugle la bonne anisotropie à partir de la donnée de formes qui évoluent selon cette anisotropie. Sur ce test, la base d'apprentissage était suffisamment riche pour que l'entraînement de nos réseaux aboutisse à des schémas d'évolutions simples et précis.

Nous avons montré dans une deuxième expérience que notre approche permettait aussi de gérer des cas d'anisotropies cristallines telles que $\phi = \|\cdot\|_\infty$, au moins pour reproduire le bon flot sur les formes de Wulff. Nous constatons néanmoins que nos réseaux se généralisent mal sur des formes faisant intervenir des directions non observées durant l'apprentissage. Ceci montre la nécessité d'enrichir la base d'apprentissage par des données incluant toutes les directions afin d'aider nos réseaux à mieux apprendre et à mieux généraliser le flot à de nouvelles formes.

Dans une troisième expérience, nous avons aussi étudié le cas d'une anisotropie non symétrique, pour lequel le mouvement par courbure moyenne n'est pas forcément bien défini. Encore une fois, notre approche a permis d'approcher correctement le flot attendu sur les formes de Wulff. Cependant, un second test a aussi révélé une perte de symétrie du flot généré par nos réseaux. Les formes de Wulff comme seules données d'apprentissage ne facilitent pas l'entraînement et il est difficile pour nos réseaux de se généraliser à des cas jamais observés durant l'apprentissage. Des expériences plus poussées devront être faites pour pouvoir analyser plus en détail l'impact de l'apprentissage, notamment si nous autorisons des données dont l'intérieur et l'extérieur sont renversés dans la base d'apprentissage.

Une dernière expérience illustre enfin la capacité de notre approche à traiter le cas du mouvement par courbure moyenne anisotrope d'interfaces non orientables. L'étude sur l'anisotropie $\phi = \|\cdot\|_{4/3}$ a montré que le réseau $S^{\mathrm{NN}}_{\theta,2}$ était capable de générer une bonne approximation du flot dans ce contexte et sur des formes quelconques, même pour des interfaces non orientables. Une dernière application sur le problème de Steiner anisotrope a permis de montrer que notre réseau était suffisamment robuste pour être couplé à des contraintes d'inclusion afin d'approcher les solutions de notre problème.

Toutes ces expériences témoignent du potentiel de notre approche pour approcher le mouvement par courbure moyenne anisotrope. Les tests que nous avons effectués ont montré la nécessité d'avoir une base de données suffisamment « riche et complète » pour que nos réseaux puissent reproduire avec précision le bon flot en partant de n'importe quelle forme initiale. Des expériences plus poussées sont néanmoins nécessaires afin de mieux comprendre la capacité de généralisation de nos réseaux, notamment dans le cas d'anisotropies cristallines.

Une fois que les cas précédents seront mieux compris, il serait aussi intéressant d'étendre notre approche aux anisotropies non convexes. Il s'agira alors de comprendre l'influence de la non convexité sur l'apprentissage et notamment sur les mouvements découlant de nos réseaux entraînés sur les formes de Wulff. Nous pensons que, de manière similaire aux cas cristallin et non symétrique, il sera nécessaire d'avoir une base de données plus conséquente afin de faciliter l'entraînement et d'aider nos réseaux à mieux se généraliser à de nouvelles données.

Il est important de noter qu'aucune contrainte de positivité des noyaux de convolution n'a été imposée durant l'apprentissage de nos réseaux. Nous savons que, dans le cas des schémas de BMO, la positivité du noyau de convolution permet d'assurer un principe de comparaison (lorsque le noyau K_{δ_t} est positif en espace) ou d'assurer la décroissance de l'énergie

$$E_{\delta_t}(\Omega) = \frac{1}{\delta_t}\int (1 - 1_\Omega) K_{\delta_t} * 1_\Omega dx,$$

lorsque le noyau est positif dans l'espace de Fourier.

Il serait intéressant d'imposer de telles contraintes durant l'entraînement du réseau pour comprendre leur impact sur les noyaux générés afin d'assurer une approximation suffisamment correcte de la loi d'anisotropie étudiée.

Par ailleurs, nous n'avons malheureusement pas pu adapter l'approche didomaine aux réseaux de neurones. L'enjeu serait de comprendre comment exploiter ce point de vue pour produire des structures de réseaux plus efficaces et plus précises. Une première idée serait d'utiliser des réseaux de diffusion en parallèle qui seraient ensuite combinés à l'aide d'un perceptron multicouche. Nous espérons alors que la combinaison de tous ces noyaux permette d'aboutir à de meilleurs résultats d'approximation.

Conclusion et perspectives

Nous nous sommes intéressés dans cette thèse à l'approximation champ de phase de certains flots géométriques et à leurs simulations numériques.

Dans la partie I du manuscrit, nous avons abordé le problème de la simulation du phénomène de mouillage sur une surface solide plus ou moins rugueuse. Pour cela nous avons poursuivi les travaux entamés dans [71, 67] portant sur la simulation de la croissance de nanofils, en proposant un nouveau modèle champ de phase alternatif, plus adapté aux applications qui nous intéressaient et capables notamment d'approcher de manière précise des flots de diffusions de surface sur des structures complexes.

Dans le cas biphasique, nous avons introduit au chapitre 7 un nouveau modèle variationnel à deux mobilités dégénérées dont l'analyse asymptotique montre qu'il a une précision champ de phase d'ordre 2, c'est-à-dire un ordre de plus par rapport aux modèles champ de phase classiques pour la diffusion de surface. Ce gain nous a permis de réduire de manière drastique les pertes de volumes observées dans les approches traditionnelles. En particulier, cela nous a permis d'approcher efficacement l'évolution par diffusion de surface d'une structure fine, comme le montre les différentes expériences numériques présentées dans le chapitre 7.

D'un point de vue numérique, la présence des deux mobilités nous a demandé d'adapter les schémas classiquement utilisés pour résoudre numériquement les équations de type Cahn-Hilliard. Notre approche a alors consisté à tirer parti du côté variationnel de la métrique en appliquant un splitting convexe-concave sur l'énergie de Cahn-Hilliard mais aussi sur la métrique. Cette approche nous a conduit à des schémas simples, rapides, précis et inconditionnellement stables en pratique.

Afin d'appliquer notre approche au problème du mouillage, nous avons étendu au chapitre 8 les résultats précédents dans un cadre multiphase (solide, liquide, vapeur) où nous avons en particulier introduit des coefficients de mobilité associés à chacune des phases afin de pouvoir modéliser l'aspect figé de la phase solide. Nous avons montré en particulier que notre modèle de Cahn-Hilliard multiphase est aussi d'ordre 2.

L'implémentation numérique des modèles multiphase a nécessité une légère adaptation des schémas que nous avions utilisés dans le cas biphasique, notamment pour le traitement du multiplicateur de Lagrange associé à la contrainte de partition. Grâce à un traitement semi-implicite de ce dernier nous avons obtenu des schémas, là encore, très simples et dont l'avantage est de résoudre L équations de Cahn-Hilliard découplées (L étant le nombre de phases) ce qui ne pose pas vraiment de difficulté même pour un grand nombre de phases. Plusieurs expériences numériques ont permis de montrer l'avantage de notre approche par rapport aux modèles pré-existants.

Une première perspective de ce travail serait de prouver la stabilité inconditionnelle de nos schémas numériques. Pour cela, il serait intéressant d'aborder dans un premier temps le cas biphasique, la preuve devrait alors s'étendre plus simplement au cas multiphasique. En pratique, nous observons sur différentes expériences numériques que nos schémas sont inconditionnellement stables mais nous n'avons pas réussi à le prouver théoriquement. La difficulté vient principalement du splitting de la métrique J qui a tendance à décentrer la métrique. Une perspective serait alors de mieux comprendre les propriétés du linéarisé $\tilde{J}_{\mu^n}$ et son impact sur la dynamique du flot.

Une autre étude qui pourrait être intéressante est la preuve d'existence du modèle **NMN** à deux mobilités dégénérées. La difficulté ici vient de la présence de la seconde mobilité dégénérée N qui a des comportements critiques aux phases pures lorsque $u \sim 0$ ou 1. Il faudrait donc pouvoir adapter les preuves dans le cas d'une seule mobilité dégénérée à notre modèle en imposant les bonnes conditions de décroissance sur N. Comme nous avons pu le voir dans les développements asymptotiques, pour avoir un modèle d'ordre 2 le choix de N est conditionné par celui de M. Nous pensons que les hypothèses imposées à N seront aussi fortement conditionnées par rapport au choix de la mobilité M. Pour cela, on pourrait par exemple suivre les travaux de Dziwnik [134] qui a étudié l'existence des solutions faibles de l'équation de Cahn-Hilliard anisotrope à une mobilité dégénérée avec potentiel régulier de type double puits en utilisant l'aspect varia-

tionnel du modèle et une approche de type Galerkin. Nous pourrions peut-être adapter son raisonnement en utilisant le côté variationnel de notre modèle.

Une autre perspective intéressante serait enfin d'étendre notre étude au cas de la diffusion de surface anisotrope afin de simuler des phénomènes de mouillage en présence d'une anisotropie. Cette approche serait aussi intéressante dans l'étude de la croissance de nanofils où l'anisotropie intervient. Une manière de procéder serait d'adapter notre approche en remplaçant l'énergie de Cahn-Hilliard classique par l'énergie de Cahn-Hilliard anisotrope. On obtiendrait ainsi un nouveau modèle de Cahn-Hilliard anisotrope avec deux mobilités dégénérées. L'objectif serait alors de montrer que les résultats asymptotiques obtenus dans le cas isotrope restent vrais dans ce cadre c'est-à-dire que ce nouveau modèle approche effectivement un flot de diffusion de surface anisotrope avec une précision identique au cas isotrope. Une étude préliminaire serait déjà d'établir ce résultat pour les modèles classiques de Cahn-Hilliard à une mobilité dégénérée avant de l'étendre à notre modèle à deux mobilités dégénérées, l'analyse asymptotique étant sans doute plus simple. D'un point de vue numérique, le traitement du laplacien anisotrope pourrait impacter considérablement l'efficacité de nos schémas. Une alternative serait alors d'adapter les travaux de Bonnetier et al [56] à nos modèles en remplaçant le laplacien anisotropie par le laplacien linéarisé dans l'espace de Fourier. Nos schémas resteraient alors identiques et le traitement de l'opérateur de diffusion pourrait toujours se faire de manière efficace grâce à la transformée de Fourier rapide.

Dans la partie II de cette thèse, nous avons introduit de nouvelles méthodes numériques efficaces basées sur les réseaux de neurones pour l'approximation du flot par courbure moyenne de surfaces orientées ou non orientables. Nous avons montré à travers divers exemples numériques que les réseaux entraînés sur des jeux de données très simples sont capables de se généraliser à des surfaces orientables et non orientables plus complexes, même en présence de singularités. L'architecture des réseaux est inspirée de la structure des schémas numériques utilisés pour résoudre l'équation d'Allen-Cahn. Nous avons également démontré l'utilité des réseaux pré-entraînés en tant qu'alternatives dans diverses applications, telles que la résolution du problème de Steiner ou du problème du Plateau, où les sorties des réseaux sont couplées à des contraintes supplémentaires pour approcher la solution.

Dans le chapitre 11, nous avons utilisé la méthodologie précédente pour l'approximation du flot de Willmore. Nous avons alors adapté la structure de nos réseaux en nous inspirant de schémas de type Bence-Merriman-Osher utilisés pour approcher ce flot. La difficulté est venue de l'aspect fortement non linéaire de ce flot et aux données exactes qui sont limitées au cas des cercles, et donc insuffisantes pour apprendre correctement ce flot. Bien que le travail à ce sujet ne soit pas abouti, les premiers résultats sont enthousiasmants. Les réseaux que nous avons entraînés semblent apprendre suffisamment correctement sur les petits cercles, et on retrouve bien le comportement attendu sur d'autres formes que le cercle. Enfin, l'architecture semble influencer considérablement le comportement à proximité de zones de création de singularités.

Dans le chapitre 12, nous avons abordé la question de l'approximation du mouvement par courbure moyenne anisotrope par des réseaux de neurones et la question de l'identification des anisotropies. Nous avons entraîné nos réseaux sur une base de données constituée d'évolutions exactes de formes de Wulff. Les résultats numériques montrent encore une fois le potentiel de cette approche, nous retrouvons notamment le comportement attendu sur des cas simples de formes de Wulff mais aussi sur des exemples non utilisés durant l'apprentissage ce qui montre la capacité de généralisation de nos réseaux entraînés. L'utilisation de formules dues à Esedoglu, Elsey et Otto [147, 141, 146] nous a par ailleurs permis d'identifier l'anisotropie apprise par les réseaux à partir de leurs noyaux. Des premiers tests de comparaison avec l'anisotropie attendue ont montré le potentiel très intéressant de cette approche.

Le travail présentée dans cette seconde partie de la thèse ouvre beaucoup de perspectives pour l'approximation numérique de flot d'interfaces géométriques en général.

Une première perspective serait de poursuivre les travaux entamés sur le flot de Willmore. Il s'agirait de mieux comprendre l'influence de l'apprentissage mais aussi des structures de réseaux sur les résultats numériques, à la fois dans des cas simples d'évolutions régulières mais aussi pour des évolutions en présence de singularités. Pour cela, il faudrait sans doute générer une base de données plus conséquente que celle que nous avons utilisée au chapitre 10. Les déplacements aux interfaces sur un pas de temps étant souvent trop petits pour que le réseau puisse correctement apprendre le flot, il faudrait aussi envisager un entraînement sur un plus grand nombre de pas que ce que nous avons utilisé dans nos expériences. Une autre perspective serait d'étendre ensuite cette approche au cas de la dimension 3. L'apprentissage devrait sans doute être plus simple mais beaucoup plus lent qu'en dimension 2.

Une seconde perspective serait de finaliser l'analyse entamée au chapitre 12, avec notamment des entraînements sur d'autres exemples d'anisotropie régulières afin de valider notre approche. Il serait également intéressant d'étendre notre méthodologie pour apprendre des anisotropies moins régulières et non convexes. Là encore, l'idée serait alors de comprendre l'influence de l'entraînement mais aussi de la structure de réseau sur l'anisotropie apprise. On pourrait alors effectuer une comparaison plus précise de notre

approche avec les modèles champ de phase existants et notamment avec les travaux de Bonnetier et al [56] où est proposée une approche capable de traiter toutes sortes d'anisotropies.

Plus généralement, il serait intéressant de comprendre, tant d'un point de vue théorique que numérique, pour quels types d'équations nos structures de réseaux permettent d'en approcher les solutions. Plus globalement, il s'agirait de mieux comprendre comment adapter notre protocole pour approcher une loi d'évolution donnée. Un cas simple serait le réseau $\mathcal{DR}$ qui alterne un réseau de diffusion puis un réseau de réaction. Quels types de semi-groupes pouvons nous approcher avec ce type de réseaux ? Et pouvons-nous établir des résultats rigoureux d'approximation d'un semi-groupe donné par ce type de réseaux ?

Bibliographie

[1] Ahmed Abdeljawad and Philipp Grohs. Approximations with deep neural networks in Sobolev time-space. *arXiv preprint arXiv :2101.06115*, 2020. 35, 53

[2] Helmut Abels. On a diffuse interface model for two-phase flows of viscous, incompressible fluids with matched densities. *Arch. Ration. Mech. Anal.*, 194(2) :463–506, 2009. 109

[3] Jonas Adler and Ozan Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12) :124007, 2017. 121

[4] Marco Albani, Roberto Bergamaschini, and Francesco Montalenti. Dynamics of pit filling in heteroepitaxy via phase-field simulations. *Physical Review B*, 94(7) :075303, 2016. 66

[5] Matthieu Alfaro and Pierre Alifrangis. Convergence of a mass conserving Allen-Cahn equation whose Lagrange multiplier is nonlocal and local. *Interfaces Free Bound.*, 16 :243–268, 2014. 72, 76, 92

[6] Nicholas D Alikakos, Peter W Bates, and Xinfu Chen. Convergence of the Cahn-Hilliard equation to the Hele-Shaw model. *Archive for rational mechanics and analysis*, 128(2) :165–205, 1994. 46, 65, 76, 82

[7] Fred Almgren, Jean E Taylor, and Lihe Wang. Curvature-driven flows : a variational approach. *SIAM Journal on Control and Optimization*, 31(2) :387–438, 1993. 15

[8] Tobias Alt, Karl Schrader, Matthias Augustin, Pascal Peter, and Joachim Weickert. Connections between numerical algorithms for pdes and neural networks. *arXiv preprint arXiv :2107.14742*, 2021. 121

[9] S Amato. *Some results on anisotropic mean curvature and other phase transition models. SISSA*. PhD thesis, Ph. D. Thesis. 164

[10] Stefano Amato, Giovanni Bellettini, and Maurizio Paolini. The nonlinear multidomain model : a new formal asymptotic analysis. In *Geometric Partial Differential Equations proceedings*, pages 33–74. Springer, 2013. 162, 163, 164

[11] L. Ambrosio. Geometric evolution problems, distance function and viscosity solutions. In *Calculus of variations and partial differential equations (Pisa, 1996)*, pages 5–93. Springer, Berlin, 2000. 24, 28, 71, 72, 118

[12] L. Ambrosio and N. Dancer. *Calculus of variations and partial differential equations*. Springer-Verlag, Berlin, 2000. Topics on geometrical evolution problems and degree theory, Papers from the Summer School held in Pisa, September 1996, Edited by G. Buttazzo, A. Marino and M. K. V. Murthy. 15, 23, 26, 148

[13] L. Ambrosio, N. Fusco, and D. Pallara. *Functions of bounded variation and free discontinuity problems*. Oxford Mathematical Monographs, 2000. 9

[14] Luigi Ambrosio, Nicola Fusco, and Diego Pallara. *Functions of bounded variation and free discontinuity problems*. Oxford Mathematical Monographs, The Clarendon Press, Oxford University Press, New York, 2000. 71, 72

[15] Luigi Ambrosio and Simon Masnou. A direct variational approach to a problem arising in image reconstruction. *Interfaces and Free Boundaries*, 5(1) :63–81, 2003. 12, 144

[16] Luigi Ambrosio, Jean-Michel Morel, Simon Masnou, and Vicent Caselles. Connected components of sets of finite perimeter and applications to image processing. *Journal of the European Mathematical Society*, 3(1) :39–92, 2001. 11

[17] Luigi Ambrosio, Giuseppe Savaré, and Piero Colli Franzone. On the asymptotic behaviour of anisotropic energies arising in the cardiac bidomain model. *Interfaces and Free Boundaries*, 2(3) :213–266, 2000. 162, 163

[18] Luigi Ambrosio and Vincenzo Maria Tortorelli. Approximation of functionals depending on jumps by elliptic functionals via Γ-convergence. *Comm. Pure Appl. Math.*, 43(8) :999–1036, 1990. 11, 52, 119

[19] Anima Anandkumar, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Nikola Kovachki, Zongyi Li, Burigede Liu, and Andrew Stuart. Neural operator : Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020. 54, 55, 56, 121

[20] Benjamin Aymard, Urbain Vaes, Marc Pradas, and Serafim Kalliadasis. A linear, second-order, energy stable, fully adaptive finite element method for phase-field modelling of wetting phenomena. *J. Comput. Phys. X*, 2 :100010, 22, 2019. 109

[21] Rainer Backofen, Steven M. Wise, Marco Salvalaglio, and Axel Voigt. Convexity splitting in a phase field model for surface diffusion. *Int. J. Numer. Anal. Model.*, 16(2) :192–209, 2019. 78, 80, 87, 101

[22] Weizhu Bao and Quan Zhao. A structure-preserving parametric finite element method for surface diffusion. *SIAM Journal on Numerical Analysis*, 59(5) :2775–2799, 2021. 23

[23] L Bar and N Sochen. Unsupervised deep learning algorithm for pde-based forward and inverse problems, arxiv. *arXiv preprint arXiv :1904.05417*, 2019. 121

[24] John W Barrett, Harald Garcke, and Robert Nürnberg. On the variational approximation of combined second and fourth order geometric evolution equations. *SIAM Journal on Scientific Computing*, 29(3) :1006–1041, 2007. 23

[25] John W. Barrett, Harald Garcke, and Robert Nürnberg. A parametric finite element method for fourth order geometric evolution equations. *J. Comput. Phys.*, 222(1) :441–462, 2007. 101

[26] John W. Barrett, Harald Garcke, and Robert Nürnberg. On the parametric finite element approximation of evolving hypersurfaces in $\mathbb{R}^3$. *J. Comput. Phys.*, 227 :4281–4307, April 2008. 23, 117

[27] John W Barrett, Harald Garcke, and Robert Nürnberg. Parametric approximation of Willmore flow and related geometric evolution equations. *SIAM Journal on Scientific Computing*, 31(1) :225–253, 2008. 23

[28] John W Barrett, Harald Garcke, and Robert Nürnberg. Numerical approximation of gradient flows for closed curves in $\mathbb{R}^d$. *IMA journal of numerical analysis*, 30(1) :4–60, 2010. 23

[29] John W Barrett, Harald Garcke, and Robert Nürnberg. Parametric approximation of isotropic and anisotropic elastic flow for closed and open curves. *Numerische Mathematik*, 120(3) :489–542, 2012. 23

[30] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning : a survey. *Journal of Marchine Learning Research*, 18 :1–43, 2018. 38

[31] G Bellettini. Variational approximation of functionals with curvatures and related properties. *Journal of Convex Analysis*, 4 :91–108, 1997. 143

[32] G Bellettini, R Goglione, and M Novaga. Approximation to driven motion by crystalline curvature in two dimensions. *ADVANCES IN MATHEMATICAL SCIENCES AND APPLICATIONS*, 10(1) :467–493, 2000. 161

[33] Giovanni Bellettini. *Lecture notes on mean curvature flow, barriers and singular perturbations*, volume 12 of *Appunti. Scuola Normale Superiore di Pisa (Nuova Serie) [Lecture Notes. Scuola Normale Superiore di Pisa (New Series)]*. Edizioni della Normale, Pisa, 2013. 15, 16, 23, 26, 28, 117, 119, 148, 160

[34] Giovanni Bellettini, Vicent Caselles, Antonin Chambolle, and Matteo Novaga. The volume preserving crystalline mean curvature flow of convex sets in rn. *Journal de mathématiques pures et appliquées*, 92(5) :499–527, 2009. 161

[35] Giovanni Bellettini, Gianni Dal Maso, and Maurizio Paolini. Semicontinuity and relaxation properties of a curvature depending functional in 2d. *Annali della Scuola Normale Superiore di Pisa-Classe di Scienze*, 20(2) :247–297, 1993. 144

[36] Giovanni Bellettini and Sh Yu Kholmatov. Minimizing movements for mean curvature flow of droplets with prescribed contact angle. *Journal de Mathématiques Pures et Appliquées*, 117 :1–58, 2018. 45

[37] Giovanni Bellettini and Luca Mugnai. Characterization and representation of the lower semicontinuous envelope of the elastica functional. *Annales de l'Institut Henri Poincaré C*, 21(6) :839–880, 2004. 144

[38] Giovanni Bellettini and Luca Mugnai. On the approximation of the elastica functional in radial symmetry. *Calculus of Variations and Partial Differential Equations*, 24(1) :1–20, 2005. 143

[39] Giovanni Bellettini and Luca Mugnai. Approximation of helfrich's functional via diffuse interfaces. *SIAM journal on mathematical analysis*, 42(6) :2402–2433, 2010. 143

[40] Giovanni Bellettini and Maurizio Paolini. Approssimazione variazionale di funzionali con curvatura. *Seminario di Analisi Matematica, Dipartimento di Matematica dell'Universita di Bologna*, pages 87–97, 1993. 143, 144

[41] Giovanni Bellettini and Maurizio Paolini. Quasi-optimal error estimates for the mean curvature flow with a forcing term. *Differential and Integral Equations*, 8(4) :735–752, 1995. 119

[42] Giovanni Bellettini and Maurizio Paolini. Anisotropic motion by mean curvature in the context of Finsler geometry. *Hokkaido Mathematical Journal*, 25(3) :537–566, 1996. 17, 159, 160, 161

[43] Giovanni Bellettini, Maurizio Paolini, and Sergio Venturini. Some results on surface measures in calculus of variations. *Annali di Matematica Pura ed Applicata*, 170(1) :329–357, 1996. 159

[44] Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences of the United States of America*, 38(8) :716, 1952. 53

[45] Marouen Ben Said, Michael Selzer, Britta Nestler, Daniel Braun, Christian Greiner, and Harald Garcke. A phase-field approach for wetting phenomena of multiphase droplets on solid surfaces. *Langmuir*, 30(14) :4033–4039, 2014. 44, 109

[46] J. Bence, B. Merriman, and S. Osher. Diffusion generated motion by mean curvature. *Computational Crystal Growers Workshop,J. Taylor ed. Selected Lectures in Math., Amer. Math. Soc.*, pages 73–83, 1992. 25, 26, 109, 118

[47] Andrea Bertozzi, Selim Esedoglu, and Alan Gillette. Inpainting of binary images using the Cahn-Hilliard equation. *IEEE Transactions on Image Processing*, 16 :285 – 291, 02 2007. 65

[48] Dimitri P Bertsekas et al. Incremental gradient, subgradient, and proximal methods for convex optimization : A survey. *Optimization for Machine Learning*, 2010(1-38) :3, 2011. 38

[49] Saakaar Bhatnagar, Yaser Afshar, Shaowu Pan, Karthik Duraisamy, and Shailendra Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64(2) :525–545, 2019. 121

[50] Kaushik Bhattacharya, Bamdad Hosseini, Nikola B Kovachki, and Andrew M Stuart. Model reduction and neural networks for parametric pdes. *arXiv preprint arXiv :2005.03180*, 2020. 54, 55, 56, 121

[51] Saswata Bhattacharyya and T. A. Abinandanan. A study of phase separation in ternary alloys. *Bulletin of Materials Science*, 26(1) :193–197, 2003. 101

[52] Jan Blechschmidt and Oliver G Ernst. Three ways to solve partial differential equations with neural networks—a review. *GAMM-Mitteilungen*, page e202100006, 2021. 121

[53] Mauro Bonafini, Giandomenico Orlandi, and Édouard Oudet. Variational approximation of functionals defined on 1-dimensional connected sets : the planar case. *SIAM J. Math. Anal.*, 50(6) :6307–6332, 2018. 138

[54] Mauro Bonafini and Édouard Oudet. A convex approach to the Gilbert-Steiner problem. *Interfaces Free Bound.*, 22(2) :131–155, 2020. 138, 144, 145

[55] Joseph-Frédéric Bonnans, Jean Charles Gilbert, Claude Lemaréchal, and Claudia A Sagastizábal. *Numerical optimization : theoretical and practical aspects*. Springer Science & Business Media, 2006. 37, 38

[56] Eric Bonnetier, Elie Bretin, and Antonin Chambolle. Consistency result for a non monotone scheme for anisotropic mean curvature flow. *Interfaces and Free Boundaries*, 14(1) :1–35, 2012. 162, 165, 186, 187

[57] Matthieu Bonnivard, Elie Bretin, and Antoine Lemenant. Numerical approximation of the Steiner problem in dimension 2 and 3. *Math. Comp.*, 89(321) :1–43, 2020. 138

[58] Matthieu Bonnivard, Antoine Lemenant, and Filippo Santambrogio. Approximation of length minimization problems among compact connected sets. *SIAM J. Math. Anal.*, 47(2) :1489–1529, 2015. 138

[59] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010. 127

[60] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 37

[61] F. Boyer, C. Lapuerta, S. Minjeaud, B. Piar, and M. Quintard. Cahn–Hilliard/Navier–Stokes Model for the Simulation of Three-Phase Flows. *Transport in Porous Media*, 82(3) :463–483, 2010. 65, 109

[62] Franck Boyer and Céline Lapuerta. Study of a three component Cahn-Hilliard flow model. *M2AN Math. Model. Numer. Anal.*, 40(4) :653–687, 2006. 101

[63] Franck Boyer and Flore Nabet. A DDFV method for a Cahn-Hilliard/Stokes phase field model with dynamic boundary conditions. *ESAIM : Mathematical Modelling and Numerical Analysis*, 51, 11 2016. 65, 109

[64] Kenneth A Brakke. *The motion of a surface by its mean curvature.(MN-20)*. Princeton University Press, 2015. 16

[65] M. Brassel and E. Bretin. A modified phase field approximation for mean curvature flow with conservation of the volume. *Mathematical Methods in the Applied Sciences*, 34(10) :1157–1180, 2011. 67, 71, 78, 101, 137

[66] Kristian Bredies, Thomas Pock, and Benedikt Wirth. A convex, lower semicontinuous approximation of Euler's elastica energy. *SIAM Journal on Mathematical Analysis*, 47(1) :566–613, 2015. 12

[67] Elie Bretin, Alexandre Danescu, José Penuelas, and Simon Masnou. Multiphase mean curvature flows with high mobility contrasts : a phase-field approach, with applications to nanowires. *Journal of Computational Physics*, 365 :324–349, 2018. 43, 78, 90, 101, 109, 121, 134, 137, 185

[68] Elie Bretin, François Dayrens, and Simon Masnou. Volume reconstruction from slices. *SIAM J. Imaging Sci.*, 10(4) :2326–2358, 2017. 12, 136, 138, 155, 156

[69] Elie Bretin, Roland Denis, Simon Masnou, Arnaud Sengers, and Garry Terii. A cahn-hilliard multiphase system with mobilities for the simulation of wetting. *arXiv preprint arXiv :2105.09627*, 2021. 44

[70] Élie Bretin, J-O Lachaud, and Édouard Oudet. Regularization of discrete contour by Willmore energy. *Journal of Mathematical Imaging and Vision*, 40(2) :214–229, 2011. 12

[71] Elie Bretin and Simon Masnou. A new phase field model for inhomogeneous minimal partitions, and applications to droplets dynamics. *Interfaces and Free Boundaries*, 19(2) :141–182, 2017. 43, 45, 78, 101, 109, 137, 185

[72] Elie Bretin, Simon Masnou, and Édouard Oudet. Phase-field approximations of the Willmore functional and flow. *Numer. Math.*, 131(1) :115–171, 2015. 72, 92

[73] Elie Bretin, Simon Masnou, Arnaud Sengers, and Garry Terii. Approximation of surface diffusion flow : a second order variational Cahn-Hilliard model with degenerate mobilities. arXiv :2007.03793, 2020. 91, 92, 101, 102, 103, 105

[74] Elie Bretin and Valerie Perrier. Phase field method for mean curvature flow with boundary constraints. *ESAIM Math. Model. Numer. Anal.*, 46(6) :1509–1526, 2012. 138, 155

[75] Bretin, Élie, Denis, Roland, Lachaud, Jacques-Olivier, and Oudet, Édouard. Phase-field modelling and computing for a large number of phases. *ESAIM : M2AN*, 53(3) :805–832, 2019. 67, 78, 101, 137

[76] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering : Machine learning, dynamical systems, and control*. Cambridge University Press, 2022. 31, 32

[77] Sébastien Bubeck et al. Convex optimization : Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4) :231–357, 2015. 38

[78] Martin Burger, Lin He, and Carola-Bibiane Schönlieb. Cahn-Hilliard inpainting and a generalization for grayvalue images. *SIAM J. Imaging Sciences*, 2 :1129–1167, 01 2009. 65

[79] John W. Cahn. On spinodal decomposition. *Acta Metallurgica*, 9(9) :795–801, 1961. 46, 65

[80] John W. Cahn. Critical point wetting. *The Journal of Chemical Physics*, 66(8) :3667–3672, 1977. 109

[81] John W Cahn, Charles M Elliott, and Amy Novick-Cohen. The cahn–hilliard equation with a concentration dependent mobility : motion by minus the laplacian of the mean curvature. *European journal of applied mathematics*, 7(3) :287–301, 1996. 19, 46, 65

[82] John W. Cahn and John E. Hilliard. Free energy of a nonuniform system. I. Interfacial free energy. *The Journal of Chemical Physics*, 28(2) :258–267, 1958. 27, 46, 65

[83] J.W. Cahn and J.E. Taylor. Overview no. 113 surface motion by surface diffusion. *Acta Metallurgica et Materialia*, 42(4) :1045–1063, 1994. 65

[84] Mathilde Callies and David Quere. On water repellency. *Soft matter*, 1(1) :55–61, 2005. 43

[85] Frédéric Cao, Yann Gousseau, Simon Masnou, and Patrick Pérez. Geometrically guided exemplar-based inpainting. *SIAM Journal on Imaging Sciences*, 4(4) :1143–1179, 2011. 12

[86] David G. Caraballo. The triangle inequalities and lower semi-continuity of surface energy of partitions. *Proc. Roy. Soc. Edinburgh Sect. A*, 139(3) :449–457, 2009. 17, 43, 89

[87] Andreas Carlson, Minh Do-Quang, and Gustav Amberg. Dissipation in rapid dynamic wetting. *Journal of Fluid Mechanics*, 682 :213–240, 2011. 109

[88] Thomas Cecil. A numerical method for computing minimal surfaces in arbitrary dimension. *J. Comput. Phys.*, 206(2) :650–660, 2005. 139

[89] Antonin Chambolle, Luca A. D. Ferrari, and Benoit Merlet. Variational approximation of size-mass energies for k-dimensional currents. *ESAIM Control Optim. Calc. Var.*, 25 :Paper No. 43, 39, 2019. 139

[90] Antonin Chambolle, Luca Alberto Davide Ferrari, and Benoit Merlet. A phase-field approximation of the Steiner problem in dimension two. *Adv. Calc. Var.*, 12(2) :157–179, 2019. 138

[91] Antonin Chambolle and Matteo Novaga. Convergence of an algorithm for the anisotropic and crystalline mean curvature flow. *SIAM journal on mathematical analysis*, 37(6) :1978–1987, 2006. 161

[92] L.Q. Chen and Jie Shen. Applications of semi-implicit Fourier-spectral method to phase field equations. *Computer Physics Communications*, 108 :147–158, 1998. 78, 101

[93] Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4) :911–917, 1995. 35, 55

[94] Xinfu Chen. Generation and Propagation of Interfaces in Reaction-Diffusion Systems. *Transactions of the American Mathematical Society*, 334(2) :877–913, 1992. 27, 118, 119

[95] Xinfu Chen, Gunduz Caginalp, and Christof Eck. A rapidly converging phase field model. *Discrete & Continuous Dynamical Systems*, 15(4) :1017–1034, 2006. 67

[96] Xinfu Chen, Danielle Hilhorst, and Elisabeth Logak. Mass conserving Allen–Cahn equation and volume preserving mean curvature flow. *Interfaces and Free Boundaries*, 12(4) :527–549, 2011. 72, 92

[97] Y. G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *Proc. Japan Acad. Ser. A Math. Sci.*, 65(7) :207–210, 1989. 16, 24, 117

[98] Mowei Cheng and James A. Warren. An efficient algorithm for solving the phase field crystal model. *J. Comput. Phys.*, 227(12) :6241–6248, 2008. 78, 101

[99] Laurence Cherfils, Hussein Fakih, and Alain Miranville. A complex version of the Cahn-Hilliard equation for grayscale image inpainting. *Multiscale Modeling and Simulation*, 15 :575–605, 03 2017. 65

[100] David L. Chopp. Computing minimal surfaces via level set curvature flow. *J. Comput. Phys.*, 106(1) :77–91, 1993. 25, 139

[101] Charles K Chui, Xin Li, and Hrushikesh Narhar Mhaskar. Neural networks for localized approximation. *mathematics of computation*, 63(208) :607–623, 1994. 34

[102] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv :1511.07289*, 2015. 34

[103] Tobias H Colding and William P Minicozzi II. Minimal surfaces and mean curvature flow. *arXiv preprint arXiv :1102.1411*, 2011. 15

[104] Pierluigi Colli and Philippe Laurençot. A phase-field approximation of the Willmore flow with volume constraint. *Interfaces and Free Boundaries*, 13(3) :341–351, 2011. 145

[105] Pierluigi Colli and Philippe Laurencot. A phase-field approximation of the Willmore flow with volume and area constraints. *SIAM Journal on Mathematical Analysis*, 44(6) :3734–3754, 2012. 145

[106] M. I. M. Copetti. Numerical experiments of phase separation in ternary mixtures. *Math. Comput. Simulation*, 52(1) :41–51, 2000. 101

[107] C. Coppin and D. Greenspan. A contribution to the particle modeling of soap films. *Appl. Math. Comput.*, 26(4) :315–331, 1988. 139

[108] Michael G Crandall and Pierre-Louis Lions. Viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American mathematical society*, 277(1) :1–42, 1983. 24

[109] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4) :303–314, December 1989. 34, 121

[110] Shibin Dai and Qiang Du. Motion of interfaces governed by the Cahn–Hilliard equation with highly disparate diffusion mobility. *SIAM Journal on Applied Mathematics*, 72(6) :1818–1841, 2012. 47, 66

[111] Shibin Dai and Qiang Du. Coarsening mechanism for systems governed by the Cahn–Hilliard equation with degenerate diffusion mobility. *Multiscale Modeling & Simulation*, 12(4) :1870–1889, 2014. 47, 66

[112] Shibin Dai and Qiang Du. Weak solutions for the Cahn–Hilliard equation with degenerate mobility. *Archive for Rational Mechanics and Analysis*, 219(3) :1161–1184, 2016. 46

[113] Ha Dang, Paul C Fife, and LA Peletier. Saddle solutions of the bistable diffusion equation. *Zeitschrift für angewandte Mathematik und Physik ZAMP*, 43(6) :984–998, 1992. 144

[114] Fabrizio Davì and Morton E. Gurtin. On the motion of a phase interface by surface diffusion. *Zeitschrift für angewandte Mathematik und Physik ZAMP*, 41(6) :782–811, 1990. 18, 19, 65

[115] Guy David. Should We Solve Plateau's Problem Again? Lecture for the conference in Honor of E. Stein, 2011, Jul 2012. 136

[116] Pierre-Gilles De Gennes, Françoise Brochard-Wyart, David Quéré, et al. *Capillarity and wetting phenomena : drops, bubbles, pearls, waves*, volume 315. Springer, 2004. 43

[117] Piero De Mottoni and Michelle Schatzman. Geometrical Evolution of Developed Interfaces. *Transactions of the American Mathematical Society*, 347(5) :1533–1589, 1995. 119

[118] K Deckelnick and G Dziuk. On the approximation of the curve shortening flow. *Calculus of variations, applications and computations*, 326 :100, 1995. 23

[119] Klaus Deckelnick, Gerhard Dziuk, and Charles M. Elliott. Computation of geometric partial differential equations and mean curvature flow. *Acta Numer.*, 14 :139–232, 2005. 117

[120] Ronald DeVore, Boris Hanin, and Guergana Petrova. Neural network approximation. *Acta Numerica*, 30 :327–444, 2021. 35

[121] P Clark Di Leoni, Lu Lu, Charles Meneveau, George Karniadakis, and Tamer A Zaki. Deeponet prediction of linear instability waves in high-speed boundary layers. *arXiv preprint arXiv :2105.08697*, 2021. 55

[122] Felix Diewald, Charlotte Kuhn, Michaela Heier, Kai Langenbach, Martin Horsch, Hans Hasse, and Ralf Müller. Investigating the stability of the phase field solution of equilibrium droplet configurations by eigenvalues and eigenvectors. *Computational Materials Science*, 141 :185–192, 2018. 109

[123] S. Dong. On imposing dynamic contact-angle boundary conditions for wall-bounded liquid–gas flows. *Computer Methods in Applied Mechanics and Engineering*, 247-248 :179–200, 2012. 109

[124] Jesse Douglas. A method of numerical solution of the problem of Plateau. *Ann. of Math. (2)*, 29(1-4) :180–188, 1927/28. 139

[125] Jesse Douglas. Solution of the problem of Plateau. *Trans. Amer. Math. Soc.*, 33(1) :263–321, 1931. 139

[126] Qiang Du and Xiaobing Feng. Chapter 5 - The phase field method for geometric moving interfaces and their numerical approximations. In Andrea Bonito and Ricardo H. Nochetto, editors, *Handbook of Numerical Analysis*, volume 21 of *Geometric Partial Differential Equations - Part I*, pages 425–508. Elsevier, 2020-01-01. 78, 101, 123

[127] Qiang Du, Chun Liu, and Xiaoqiang Wang. A phase field approach in the numerical study of the elastic bending energy for vesicle membranes. *Journal of Computational Physics*, 198(2) :450–468, 2004. 145

[128] Qiang Du, Chun Liu, and Xiaoqiang Wang. Simulating the deformation of vesicle membranes under elastic bending energy in three dimensions. *Journal of computational physics*, 212(2) :757–777, 2006. 145

[129] Qiang Du and Xiaoqiang Wang. Convergence of numerical approximations to a phase field bending elasticity model of membrane deformations. 2006. 145

[130] G. Dziuk. An algorithm for evolutionary surfaces. *Numer. Math.*, 58(6) :603–611, 1991. 22

[131] Gerhard Dziuk and John E. Hutchinson. The discrete Plateau problem : algorithm and numerics. *Math. Comp.*, 68(225) :1–23, 1999. 139

[132] Gerhard Dziuk and John E. Hutchinson. The discrete Plateau problem : convergence results. *Math. Comp.*, 68(226) :519–546, 1999. 139

[133] Gerhard Dziuk, Ernst Kuwert, and Reiner Schatzle. Evolution of elastic curves in \rn : Existence and computation. *SIAM journal on mathematical analysis*, 33(5) :1228–1245, 2002. 20, 143

[134] Marion Dziwnik. Dewetting of thin solid films. 2016. 185

[135] Marion Dziwnik, Andreas Münch, and Barbara Wagner. A phase-field model for solid-state dewetting and its sharp-interface limit. 2014. 44

[136] Marion Dziwnik, Andreas Münch, and Barbara Wagner. An anisotropic phase-field model for solid-state dewetting and its sharp-interface limit. *Nonlinearity*, 30(4) :1465, 2017. 66

[137] Abdallah El Chakik, Abdderahim Elmoataz, and Xavier Desquesnes. Mean curvature flow on graphs for image and manifold restoration and enhancement. *Signal processing*, 105 :449–463, 2014. 11

[138] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on learning theory*, pages 907–940. PMLR, 2016. 34

[139] Charles M. Elliott and Harald Garcke. On the Cahn–Hilliard Equation with Degenerate Mobility. *SIAM Journal on Mathematical Analysis*, 27(2) :404–423, 1996. 66

[140] Charles M. Elliott and Harald Garcke. Existence results for diffusive surface motion laws. *Adv. Math. Sci. Appl*, 7 :465–488, 1997. 19, 46, 65

[141] Matt Elsey and Selim Esedoğlu. Threshold dynamics for anisotropic surface energies. *Mathematics of Computation*, 87(312) :1721–1756, 2018. 26, 27, 164, 170, 186

[142] Matt Elsey and Benedikt Wirth. A simple and efficient scheme for phase field crystal simulation. *ESAIM Math. Model. Numer. Anal.*, 47(5) :1413–1432, 2013. 78, 101

[143] Douglas Enright, Ronald Fedkiw, Joel Ferziger, and Ian Mitchell. A hybrid particle level set method for improved interface capturing. *Journal of Computational physics*, 183(1) :83–116, 2002. 25

[144] Joachim Escher, Uwe F. Mayer, and Gieri Simonett. On the surface diffusion flow. In *Navier-Stokes equations and related nonlinear problems (Palanga, 1997)*, pages 69–79. VSP, Utrecht, 1998. 19, 65

[145] Joachim Escher, Uwe F. Mayer, and Gieri Simonett. The surface diffusion flow for immersed hypersurfaces. *SIAM J. Math. Anal.*, 29(6) :1419–1433, 1998. 19, 65

[146] Selim Esedoğlu. Algorithms for motion of networks by weighted mean curvature. In *Proceedings of the International Congress of Mathematicians : Rio de Janeiro 2018*, pages 3947–3966. World Scientific, 2018. 26, 27, 164, 170, 186

[147] Selim Esedoğlu and Felix Otto. Threshold dynamics for networks with arbitrary surface tensions. *Communications on Pure and Applied Mathematics*, 68 :808–864, 2015. 26, 27, 109, 118, 147, 164, 170, 186

[148] Selim Esedoglu, Andreas Rätz, and Matthias Röger. Colliding interfaces in old and new diffuse-interface approximations of Willmore-flow. *arXiv preprint arXiv :1209.6531*, 2012. 143, 145, 154

[149] Selim Esedoğlu, Steven Ruuth, Richard Tsai, et al. Diffusion generated motion using signed distance functions. *Journal of Computational Physics*, 229(4) :1017–1042, 2010. 26, 27, 148

[150] Selim Esedoğlu, Steven J Ruuth, and Richard Tsai. Threshold dynamics for high order geometric motions. *Interfaces and Free Boundaries*, 10(3) :263–282, 2008. 25, 26, 27, 147

[151] Selim Esedoğlu and Jianhong Shen. Digital inpainting based on the Mumford–Shah–Euler image model. *European Journal of Applied Mathematics*, 13(4) :353–370, 2002. 12

[152] Luca Esposito, Nicola Fusco, and Cristina Trombetti. A quantitative version of the isoperimetric inequality : the anisotropic case. *Annali della Scuola Normale Superiore di Pisa-Classe di Scienze*, 4(4) :619–651, 2005. 17

[153] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. I. *Journal of Differential Geometry*, 33(3) :635–681, 1991. 16, 24, 117

[154] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. II. *Transactions of the American Mathematical Society*, 330(1) :321–332, 1992. 24

[155] L. C. Evans and J. Spruck. Motion of level sets by mean curvature III. *The Journal of Geometric Analysis*, 2(2) :121–150, 1992-03-01. 24

[156] Lawrence C Evans and Ronald F Garzepy. *Measure theory and fine properties of functions*. Routledge, 2018. 12, 15

[157] Lawrence C. Evans and Joel Spruck. Motion of level sets by mean curvature IV. *The Journal of Geometric Analysis*, 5(1) :77–114, 1995-03-01. 24

[158] David J. Eyre. Unconditionally Gradient Stable Time Marching the Cahn-Hilliard Equation. *MRS Online Proceedings Library Archive*, 529, 1998/ed. 29, 50, 78, 79, 91, 101, 121, 124

[159] William Falcon et al. Pytorch lightning. *GitHub. Note : https ://github. com/PyTorchLightning/pytorch-lightning*, 3 :6, 2019. 129

[160] MINGWEN FEI and YUNING LIU. Sharp interface limit of a phase field model for elastic bending energy. *arXiv preprint arXiv :1904.11139*, 2019. 145

[161] Francesca Fierro, Roberta Goglione, and Maurizio Paolini. Numerical simulations of mean curvature flow in the presence of a nonconvex anisotropy. *Mathematical Models and Methods in Applied Sciences*, 8(04) :573–601, 1998. 161

[162] Irene Fonseca and Stefan Müller. A uniqueness proof for the wulff theorem. *Proceedings of the Royal Society of Edinburgh Section A : Mathematics*, 119(1-2) :125–136, 1991. 17

[163] Michael Gage and Richard S Hamilton. The heat equation shrinking convex plane curves. *Journal of Differential Geometry*, 23(1) :69–96, 1986. 15

[164] H. Garcke and R. Haas. Modelling of non-isothermal multi-component, multi-phase systems with convection. Technical report, 2008. 137

[165] Harald Garcke, Britta Nestler, Björn Stinner, and Frank Wendler. Allen-Cahn systems with volume constraints. *Math. Models Methods Appl. Sci.*, 18(8) :1347–1381, 2008. 137

[166] Harald Garcke, Britta Nestler, and Barbara Stoth. On anisotropic order parameter models for multi-phase systems and their sharp interface limits. *Physica D : Nonlinear Phenomena*, 115(1-2) :87 – 108, 1998. 137

[167] Harald Garcke, Britta Nestler, and Barbara Stoth. A multi phase field concept : Numerical simulations of moving phase boundaries and multiple junctions. *SIAM J. Appl. Math*, 60 :295–315, 1999. 137

[168] Harald Garcke and Björn Stinner. Second order phase field asymptotics for multi-component systems. *Interfaces Free Bound*, 8 :131–157, 2006. 67

[169] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842. PMLR, 2015. 127

[170] Ennio De Giorgi. Some remarks on γ-convergence and least squares method. In *Composite Media and Homogenization Theory*, pages 135–142. Springer, 1991. 144

[171] Hector Gomez and Thomas J. R. Hughes. Provably unconditionally stable, second-order time-accurate, mixed variational methods for phase-field models. *J. Comput. Phys.*, 230(13) :5310–5327, 2011. 78, 101

[172] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 31, 32

[173] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 31

[174] Somdatta Goswami, Katiana Kontolati, Michael D Shields, and George Em Karniadakis. Deep transfer learning for partial differential equations under conditional shift with deeponet. *arXiv preprint arXiv :2204.09810*, 2022. 55

[175] Carsten Gräser, Ralf Kornhuber, and Uli Sack. Time discretizations of anisotropic Allen–Cahn equations. *IMA Journal of Numerical Analysis*, 33(4) :1226–1244, 2013. 161

[176] MA Grayson. The heat equation shrinking embedded plane curves to round points. *J. Differential Geometry 26 (1987), 285*, 314. 15

[177] Matthew A Grayson. A short note on the evolution of a surface by its mean curvature. *Duke Mathematical Journal*, 58(3) :555–558, 1989. 15

[178] Klaus Greff, Rupesh K Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation. *arXiv preprint arXiv :1612.07771*, 2016. 125

[179] Richards Grzhibovskis and Alexei Heintz. A convolution thresholding scheme for the Willmore flow. *Interfaces and Free Boundaries*, 10(2) :139–153, 2008. 26, 27, 147

[180] Steven Guan, Ko-Tsung Hsu, and Parag V Chitnis. Fourier neural operator networks : A fast and general solver for the photoacoustic wave equation. *arXiv preprint arXiv :2108.09374*, 2021. 55

[181] Clemens Gugenberger, Robert Spatschek, and Klaus Kassner. Comparison of phase-field models for surface diffusion. *Physical Review E*, 78(1) :016703, 2008. 46, 66

[182] Ingo Gühring, Gitta Kutyniok, and Philipp Petersen. Complexity bounds for approximations with deep relu neural networks in Sobolev norms. 2019. 35, 53

[183] Ingo Gühring, Gitta Kutyniok, and Philipp Petersen. Error bounds for approximations with deep relu neural networks in w s, p norms. *Analysis and Applications*, 18(05) :803–859, 2020. 35, 53

[184] Ingo Gühring and Mones Raslan. Approximation rates for neural networks with encodable weights in smoothness spaces. *Neural Networks*, 134 :107–130, 2021. 35, 53

[185] Ingo Gühring, Mones Raslan, and Gitta Kutyniok. Expressivity of deep neural networks. *arXiv preprint arXiv :2007.04759*, 2020. 34

[186] Ozge Gundogdu, Erol Egrioglu, Cagdas Hakan Aladag, and Ufuk Yolcu. Multiplicative neuron model artificial neural network based on gaussian activation function. *Neural Computing and Applications*, 27(4) :927–935, 2016. 34

[187] Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 481–490, 2016. 121

[188] Yanan Guo, Xiaoqun Cao, Bainian Liu, and Mei Gao. Solving partial differential equations using deep learning and physical constraints. *Applied Sciences*, 10(17) :5917, 2020. 54

[189] Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. USA*, 115(34) :8505–8510, 2018. 121

[190] Boris Hanin and Mark Sellke. Approximating continuous functions by relu nets of minimal width, 2018. *arXiv preprint arXiv :1710.11278*. 34

[191] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning : data mining, inference, and prediction*, volume 2. Springer, 2009. 32, 36

[192] Sebastian Hensel and Tim Laux. A new varifold solution concept for mean curvature flow : Convergence of the Allen-Cahn equation and weak-strong uniqueness. *arXiv preprint arXiv :2109.04233*, 2021. 16

[193] C Herring. Surface tension as a motivation for sintering. in w. e. kingston, editor. *The physics of powder metallurgy*. 179

[194] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2) :251–257, 1991. 34, 35, 124

[195] Kurt Hornik. Some new results on neural network approximation. *Neural networks*, 6(8) :1069–1072, 1993. 35, 124

[196] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5) :359–366, 1989. 34

[197] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural networks*, 3(5) :551–560, 1990. 35, 53

[198] Lucas Hsu, Rob Kusner, and John Sullivan. Minimizing the squared mean curvature integral for surfaces in space forms. *Experimental Mathematics*, 1(3) :191–207, 1992. 145

[199] Gerhard Huisken. Flow by mean curvature of convex surfaces into spheres. *Journal of Differential Geometry*, 20(1) :237–266, 1984. 15

[200] H Ishii, G. E Pires, and P. E. Souganidis. Threshold dynamics type approximation schemes for propagating fronts. *J. Math. Soc. Japan*, 51(2) :267–308, 1999. 25, 118, 164

[201] David Jacqmin. Calculation of two-phase Navier-Stokes flows using phase-field modeling. *J. Comput. Phys.*, 155(1) :96–127, 1999. 109

[202] Qinghua Jiang, Lailai Zhu, Chang Shu, and Vinothkumar Sekar. Multilayer perceptron neural network activated by adaptive gaussian radial basis function and its application to predict lid-driven cavity flow. *Acta Mechanica Sinica*, pages 1–16, 2022. 34

[203] Ehsan Kharazmi, Zhongqiang Zhang, and GE Karniadakis. Vpinns : Variational physics-informed neural networks for solving partial differential equations. *arXiv preprint arXiv :1912.00873*, 2019. 54

[204] Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving parametric pde problems with artificial neural networks. *European Journal of Applied Mathematics*, 32(3) :421–435, 2021. 121

[205] Patrick Kidger and Terry Lyons. Universal Approximation with Deep Narrow Networks. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 2306–2327. PMLR, 09–12 Jul 2020. 34, 35, 121

[206] Junseok Kim. Phase field computations for ternary fluid flows. *Comput. Methods Appl. Mech. Engrg.*, 196(45-48) :4779–4788, 2007. 101

[207] Junseok Kim. A generalized continuous surface tension force formulation for phase-field models for multi-component immiscible fluid flows. *Comput. Methods Appl. Mech. Engrg.*, 198(37-40) :3105–3112, 2009. 101

[208] Junseok Kim and Kyungkeun Kang. A numerical method for the ternary Cahn-Hilliard system with a degenerate mobility. *Appl. Numer. Math.*, 59(5) :1029–1042, 2009. 101

[209] Junseok Kim, Kyungkeun Kang, and John Lowengrub. Conservative multigrid methods for ternary Cahn-Hilliard systems. *Commun. Math. Sci.*, 2(1) :53–77, 2004. 101

[210] Diederik P Kingma and Jimmy Ba. Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*, 2014. 38, 127, 128

[211] Tomonori Kitashima, Jincheng Wang, and Hiroshi Harada. Phase-field simulation with the CALPHAD method for the microstructure evolution of multi-component Ni-base superalloys. *Intermetallics*, 16(2) :239–245, 2008. 101

[212] Rob Kusner. Comparison surfaces for the Willmore problem. *Pacific Journal of Mathematics*, 138(2) :317–345, 1989. 145

[213] Ernst Kuwert and Reiner Schätzle. Gradient flow for the Willmore functional. *Communications in Analysis and Geometry*, 10(2) :307–339, 2002. 20, 143

[214] Ernst Kuwert and Reiner Schätzle. Removability of point singularities of Willmore surfaces. *Annals of Mathematics*, pages 315–357, 2004. 20, 143

[215] Ernst Kuwert and Reiner Schätzle. The Willmore functional. In *Topics in modern regularity theory*, pages 1–115. Springer, 2012. 19

[216] Tim Laux. Distributional solutions to mean curvature flow. *arXiv preprint arXiv :2108.08347*, 2021. 16

[217] Tim Laux and Felix Otto. Convergence of the thresholding scheme for multi-phase mean-curvature flow. *Calc. Var. Partial Differential Equations*, 55(5) :Art. 129, 74, 2016. 118

[218] Jeremy LeCrone and Gieri Simonett. On well-posedness, stability, and bifurcation for the axisymmetric surface diffusion flow. *SIAM Journal on Mathematical Analysis*, 45(5) :2834–2869, 2013. 19

[219] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 2, 1989. 40, 124

[220] Alpha A Lee, Andreas Münch, and Endre Süli. Degenerate mobilities in phase field models are insufficient to capture surface diffusion. *Applied Physics Letters*, 107(8) :081603, 2015. 47, 66

[221] Alpha Albert Lee, Andreas Munch, and Endre Suli. Sharp-Interface limits of the Cahn-Hilliard equation with degenerate mobility. *SIAM Journal on Applied Mathematics*, 76(2) :433–456, 2016. 46, 47, 66, 68, 69

[222] Hyun Geun Lee, Jeong-Whan Choi, and Junseok Kim. A practically unconditionally gradient stable scheme for the N-component Cahn–Hilliard system. *Physica A : Statistical Mechanics and its Applications*, 391(4) :1009–1019, 2012. 101

[223] Hyun Geun Lee and Junseok Kim. A second-order accurate non-linear difference scheme for the N-component Cahn–Hilliard system. *Physica A : Statistical Mechanics and its Applications*, 387(19) :4787–4799, 2008. 101

[224] Hyun Geun Lee and June-Yub Lee. A semi-analytical Fourier spectral method for the Allen–Cahn equation. *Computers & Mathematics with Applications*, 68(3) :174–184, August 2014. 123

[225] Antoine Lemenant. A selective review on Mumford–Shah minimizers. *Bollettino dell'Unione Matematica Italiana*, 9(1) :69–113, 2016. 11, 12

[226] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6) :861–867, 1993. 34, 35

[227] X. Li, J. Lowengrub, A. Rätz, and A. Voigt. Solving PDEs in complex geometries : a diffuse domain approach. *Commun. Math. Sci.*, 7(1) :81–107, 2009. 108

[228] Yibao Li, Jung-Il Choi, and Junseok Kim. Multi-component Cahn-Hilliard system with different boundary conditions in complex domains. *J. Comput. Phys.*, 323 :1–16, 2016. 101

[229] Yibao Li, Darae Jeong, Jaemin Shin, and Junseok Kim. A conservative numerical method for the Cahn-Hilliard equation with Dirichlet boundary conditions in complex domains. *Comput. Math. Appl.*, 65(1) :102–115, 2013. 108

[230] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv :2010.08895*, 2020. 54, 55, 56, 121

[231] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. *Advances in Neural Information Processing Systems*, 33 :6755–6766, 2020. 54, 55, 56

[232] Chun Liu and Jie Shen. A phase field model for the mixture of two incompressible fluids and its approximation by a Fourier-spectral method. *Phys. D*, 179(3-4) :211–228, 2003. 109

[233] Peng Liu, Jun Wang, and Zhenyuan Guo. Multiple and complete stability of recurrent neural networks with sinusoidal activation function. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1) :229–240, 2020. 34

[234] Peng Liu, Zhigang Zeng, and Jun Wang. Complete stability of delayed recurrent neural networks with gaussian activation functions. *Neural Networks*, 85 :21–32, 2017. 34

[235] Paola Loreti and Riccardo March. Propagation of fronts in a nonlinear fourth order equation. *European Journal of Applied Mathematics*, 11(2) :203–213, 2000. 145

[236] Denghui Lu, Han Wang, Mohan Chen, Lin Lin, Roberto Car, E Weinan, Weile Jia, and Linfeng Zhang. 86 pflops deep potential molecular dynamics simulation of 100 million atoms with ab initio accuracy. *Computer Physics Communications*, 259 :107624, 2021. 54

[237] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet : Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv :1910.03193*, 2019. 35, 54, 55, 121

[238] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3) :218–229, 2021. 35, 55

[239] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. Deepxde : A deep learning library for solving differential equations. *SIAM Review*, 63(1) :208–228, 2021. 54

[240] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks : Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285. PMLR, 2018. 121

[241] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks : A view from the width. *Advances in neural information processing systems*, 30, 2017. 34

[242] Charles M Elliott and Hans Fritz. On approximations of the curve shortening flow and of the mean curvature flow based on the deturck trick. *IMA Journal of Numerical Analysis*, 37(2) :543–603, 2017. 23

[243] Francesco Maggi. *Sets of finite perimeter and geometric variational problems : an introduction to Geometric Measure Theory*. Number 135. Cambridge University Press, 2012. 12, 15, 17, 43, 71, 89

[244] Ravi Malladi and James A Sethian. Flows under min/max curvature flow and mean curvature : Applications in image processing. In *European Conference on Computer Vision*, pages 251–262. Springer, 1996. 11

[245] Stéphane Mallat. Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A : Mathematical, Physical and Engineering Sciences*, 374(2065) :20150203, 2016. 40, 124

[246] Carlo Mantegazza, Matteo Novaga, and Alessandra Pluda. Lectures on curvature flow of networks. In *Contemporary research in elliptic PDEs and related topics*, volume 33 of *Springer INdAM Ser.*, pages 369–417. Springer, Cham, 2019. 15, 117

[247] Simon Masnou and Giacomo Nardi. Gradient young measures, varifolds, and a generalized Willmore functional. *Advances in Calculus of Variations*, 6(4) :433–482, 2013. 144

[248] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv :1804.07612*, 2018. 127

[249] James McCoy, Glen Wheeler, and Graham Williams. Lifespan theorem for constrained surface diffusion flows. *Math. Z.*, 269(1-2) :147–178, 2011. 19, 65

[250] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4) :115–133, 1943. 32, 34

[251] Barry Merriman, James K Bence, and Stanley J Osher. Motion of multiple junctions : A level set approach. *Journal of computational physics*, 112(2) :334–363, 1994. 26

[252] Thibaut Metivet, Arnaud Sengers, Mourad Ismail, and Emmanuel Maitre. Diffusion-redistanciation schemes for 2d and 3d constrained Willmore flow : application to the equilibrium shapes of vesicles. *Journal of Computational Physics*, 436 :110288, 2021. 26, 27

[253] Stefan Metzger. On numerical schemes for phase-field models for electrowetting with electrolyte solutions. *PAMM*, 15(1) :715–718, 2015. 109

[254] Hrushikesh N Mhaskar. Neural networks for optimal approximation of smooth and analytic functions. *Neural computation*, 8(1) :164–177, 1996. 34

[255] Hrushikesh N Mhaskar and Charles A Micchelli. Approximation by superposition of sigmoidal and radial basis functions. *Advances in Applied mathematics*, 13(3) :350–373, 1992. 34

[256] Hrushikesh Narhar Mhaskar. Approximation properties of a multilayered feedforward artificial neural network. *Advances in Computational Mathematics*, 1(1) :61–80, 1993. 34, 35

[257] Alain Miranville. *The Cahn–Hilliard Equation : Recent Advances and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2019. 65

[258] Luciano Modica and Stefano Mortola. Un esempio di $\Gamma-$-convergenza. *Boll. Un. Mat. Ital. B (5)*, 14(1) :285–299, 1977. 27, 46, 118

[259] Frank Morgan. Lowersemicontinuity of energy clusters. *Proceedings of the Royal Society of Edinburgh Section A : Mathematics*, 127(4) :819–822, 1997. 17, 43, 89

[260] Roger Moser. A higher order asymptotic problem related to phase transitions. *SIAM journal on mathematical analysis*, 37(3) :712–736, 2005. 143

[261] Luca Mugnai. Gamma-convergence results for phase-field approximations of the 2d-Euler elastica functional. *ESAIM : Control, Optimisation and Calculus of Variations*, 19(3) :740–753, 2013. 144

[262] W. W. Mullins. Theory of thermal grooving. *Journal of Applied Physics*, 28(3) :333–339, 1957. 18, 65

[263] Meher Naffouti, Rainer Backofen, Marco Salvalaglio, Thomas Bottein, Mario Lodari, Axel Voigt, Thomas David, Abdelmalek Benkouider, Ibtissem Fraj, Luc Favre, et al. Complex dewetting scenarios of ultrathin silicon films for large-scale nanoarchitectures. *Science advances*, 3(11) :eaao1472, 2017. 66

[264] Yuko Nagase and Yoshihiro Tonegawa. A singular perturbation problem with integral curvature bound. *Hiroshima mathematical journal*, 37(3) :455–489, 2007. 143, 144

[265] Nicholas H Nelsen and Andrew M Stuart. The random feature model for input-output maps between banach spaces. *SIAM Journal on Scientific Computing*, 43(5) :A3212–A3243, 2021. 54, 55, 121

[266] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983. 38

[267] Britta Nestler, Frank Wendler, Michael Selzer, Björn Stinner, and Harald Garcke. Phase-field model for multiphase systems with preserved volume fractions. *Phys. Rev. E*, 78 :011604, Jul 2008. 137

[268] Amy Novick-Cohen. The Cahn-Hilliard equation. In *Handbook of differential equations : evolutionary equations. Vol. IV*, Handb. Differ. Equ., pages 201–228. Elsevier/North-Holland, Amsterdam, 2008. 65

[269] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag New York, Applied Mathematical Sciences, 2002. 23, 117

[270] S. Osher and N. Paragios. *Geometric Level Set Methods in Imaging, Vision and Graphics*. Springer-Verlag, New York, 2003. 23, 117

[271] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed : algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79 :12–49, 1988. 16, 23, 117

[272] Edouard Oudet. Approximation of partitions of least perimeter by Gamma-convergence : around Kelvin's conjecture. *Experimental Mathematics*, 20(3) :260–270, 2011. 137

[273] Tekin Evrim Ozmermer. Sinusoidal neural networks : Towards ann that learns faster. *Complex Systems Informatics and Modeling Quarterly*, (23) :44–57, 2020. 34

[274] Shaowu Pan and Karthik Duraisamy. Physics-informed probabilistic learning of linear embeddings of nonlinear dynamics with guaranteed stability. *SIAM Journal on Applied Dynamical Systems*, 19(1) :480–509, 2020. 121

[275] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. Taming the waves : sine as activation function in deep neural networks. 2016. 34

[276] Sejun Park, Chulhee Yun, Jaeho Lee, and Jinwoo Shin. Minimum width for universal approximation. *arXiv preprint arXiv :2006.08859*, 2020. 34

[277] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch : An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 129

[278] Ravi G Patel, Nathaniel A Trask, Mitchell A Wood, and Eric C Cyr. A physics-informed operator regression framework for extracting data-driven continuum models. *Computer Methods in Applied Mechanics and Engineering*, 373 :113500, 2021. 121

[279] Maurice Peemen, Bart Mesman, and Henk Corporaal. Speed sign detection and recognition by convolutional neural networks. In *Proceedings of the 8th international automotive congress*, pages 162–170. sn, 2011. 41

[280] Robert L Pego. Front migration in the nonlinear Cahn-Hilliard equation. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 422(1863) :261–278, 1989. 46, 65, 82

[281] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta numerica*, 8 :143–195, 1999. 34

[282] Paola Pozzi and Björn Stinner. On motion by curvature of a network with a triple junction. *The SMAI journal of computational mathematics*, 7 :27–55, 2021. 23, 117

[283] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. Survey of expressivity in deep neural networks. *arXiv preprint arXiv :1611.08083*, 2016. 34

[284] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR, 2017. 34

[285] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks : A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378 :686–707, 2019. 54, 121

[286] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i) : Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv :1711.10561*, 2017. 54

[287] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part ii) : Data-driven discovery of nonlinear partial differential equations. *ArXiv*, abs/1711.10566, 2017. 54

[288] Andreas Rätz, Angel Ribalta, and Axel Voigt. Surface evolution of elastically stressed films under deposition by a diffuse interface model. *Journal of Computational Physics*, 214(1) :187–208, 2006. 66

[289] E. R. Reifenberg. Solution of the Plateau problem for m-dimensional surfaces of varying topological type. *Bull. Amer. Math. Soc.*, 66 :312–313, 1960. 139

[290] Matthias Röger and Reiner Schätzle. On a modified conjecture of De Giorgi. *Mathematische Zeitschrift*, 254(4) :675–714, 2006. 143, 144

[291] Matías Roodschild, Jorge Gotay Sardiñas, and Adrián Will. A new approach for the vanishing gradient problem on sigmoid activation. *Progress in Artificial Intelligence*, 9(4) :351–360, 2020. 34

[292] Frank Rosenblatt. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6) :386, 1958. 32, 33, 124

[293] J. S. Rowlinson. Translation of J. D. van der Waals' "The thermodynamik theory of capillarity under the hypothesis of a continuous variation of density". *Journal of Statistical Physics*, 20(2) :197–200, February 1979. 27

[294] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv :1609.04747*, 2016. 38

[295] Steven J. Ruuth. Efficient algorithms for diffusion-generated motion by mean curvature. *J. Comput. Phys.*, 144(2) :603–625, 1998. 118

[296] Yu L Sachkov. Maxwell strata in the Euler elastic problem. *Journal of Dynamical and Control Systems*, 14(2) :169–234, 2008. 12

[297] Tiago Salvador and Selim Esedoğlu. The role of surface tension and mobility model in simulations of grain growth. *arXiv preprint arXiv :1907.11574*, 2019. 26, 27

[298] Marco Salvalaglio, Rainer Backofen, Roberto Bergamaschini, Francesco Montalenti, and Axel Voigt. Faceting of equilibrium and metastable nanostructures : a phase-field model of surface diffusion tackling realistic shapes. *Crystal Growth & Design*, 15(6) :2787–2794, 2015. 66

[299] Marco Salvalaglio, Rainer Backofen, Axel Voigt, and Francesco Montalenti. Morphological evolution of pit-patterned Si (001) substrates driven by surface-energy reduction. *Nanoscale research letters*, 12(1) :554, 2017. 66

[300] Marco Salvalaglio, Maximilian Selch, Axel Voigt, and Steven M. Wise. Doubly degenerate diffuse interface models of anisotropic surface diffusion. *Mathematical Methods in the Applied Sciences*, 44(7) :5406–5417, 2021. 78, 80, 101

[301] Marco Salvalaglio, Axel Voigt, and Steven M. Wise. Doubly degenerate diffuse interface models of surface diffusion. *Mathematical Methods in the Applied Sciences*, 44(7) :5385–5405, 2021. 66, 67, 69, 78, 80, 101

[302] Franco Scarselli and Ah Chung Tsoi. Universal approximation using feedforward neural networks : A survey of some existing methods, and some new results. *Neural networks*, 11(1) :15–37, 1998. 124

[303] James A Sethian. Fast marching methods. *SIAM review*, 41(2) :199–235, 1999. 25

[304] James Albert Sethian. *Level set methods and fast marching methods : evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999. 15

[305] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning : From theory to algorithms*. Cambridge university press, 2014. 32, 34, 38

[306] Yuanzhen Shao. A family of parameter-dependent diffeomorphisms acting on function spaces over a riemannian manifold and applications to geometric flows. *Nonlinear Differential Equations and Applications NoDEA*, 22(1) :45–85, 2015. 19

[307] Jie Shen, Xiaofeng Yang, and Haijun Yu. Efficient energy stable numerical schemes for a phase field moving contact line model. *J. Comput. Phys.*, 284 :617–630, 2015. 109

[308] Jaemin Shin, Darae Jeong, and Junseok Kim. A conservative numerical method for the Cahn-Hilliard equation in complex domains. *J. Comput. Phys.*, 230(19) :7441–7455, 2011. 108

[309] Jaemin Shin, Hyun Geun Lee, and June-Yub Lee. First and second order numerical methods based on a new convex splitting for phase-field crystal equation. *J. Comput. Phys.*, 327 :519–542, 2016. 78, 101

[310] Jaemin Shin, Hyun Geun Lee, and June-Yub Lee. Unconditionally stable methods for gradient flow using convex splitting Runge-Kutta scheme. *J. Comput. Phys.*, 347 :367–381, 2017. 78, 101

[311] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1) :1–48, 2019. 40

[312] David N. Sibley, Andreas Nold, Nikos Savva, and Serafim Kalliadasis. The contact line behaviour of solid-liquid-gas diffuse-interface models. *Physics of Fluids*, 25(9) :092111, 2013. 44, 109

[313] David N. Sibley, Andreas Nold, Nikos Savva, and Serafim Kalliadasis. On the moving contact line singularity : Asymptotics of a diffuse-interface model. *The European Physical Journal E*, 36(3) :26, 2013. 109

[314] Leon Simon. *Lectures on geometric measure theory*, volume 3 of *Proceedings of the Centre for Mathematical Analysis, Australian National University*. Australian National University, Centre for Mathematical Analysis, Canberra, 1983. 16

[315] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33 :7462–7473, 2020. 34

[316] Jonathan D Smith, Kamyar Azizzadenesheli, and Zachary E Ross. Eikonet : Solving the eikonal equation with deep neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 2020. 25, 121

[317] Jonthan D Smith, Zachary E Ross, Kamyar Azizzadenesheli, and Jack B Muir. Hyposvi : Hypocentre inversion with stein variational inference and physics informed neural networks. *Geophysical Journal International*, 228(1) :698–710, 2022. 54

[318] Paulo Tabuada and Bahman Gharesifard. Universal approximation power of deep residual neural networks via nonlinear control theory. *arXiv preprint arXiv :2007.06007*, 2020. 34

[319] Mickael Tanter and Mathias Fink. Time reversing waves for biomedical applications. In *Mathematical Modeling in Biomedical Imaging I*, pages 73–97. Springer, 2009. 143

[320] J Taylor. Existence and structure of solutions to a class of nonelliptic variational problems. In *Symposia Mathematica*, volume 14, pages 499–508, 1974. 17

[321] Jean Taylor. Unique structure of solutions to a class of nonelliptic variational problems. In *Proc. Symp. Pure Math. AMS*, volume 27, pages 419–427, 1975. 17

[322] Jean E Taylor. Crystalline variational problems. *Bulletin of the American Mathematical Society*, 84(4) :568–588, 1978. 17

[323] Jean E. Taylor and John W. Cahn. Linking anisotropic sharp and diffuse surface motion laws via gradient flows. *Journal of Statistical Physics*, 77(1) :183–197, 1994. 19, 46, 65

[324] Jean E Taylor, John W Cahn, and Carol A Handwerker. Overview no. 98 i—geometric models of crystal growth. *Acta Metallurgica et Materialia*, 40(7) :1443–1474, 1992. 17

[325] Knut Erik Teigen, Xiangrong Li, John Lowengrub, Fan Wang, and Axel Voigt. A diffuse-interface approach for modeling transport, diffusion and adsorption/desorption of material quantities on a deformable interface. *Communications in mathematical sciences*, 4(7) :1009–1037, 12 2009. 108

[326] Yoshihiro Tonegawa. *Brakke's Mean Curvature Flow : An Introduction*. Springer, 2019. 16, 21

[327] Alessandro Turco, François Alouges, and Antonio DeSimone. Wetting on rough surfaces and contact angle hysteresis : numerical experiments based on a phase field model. *M2AN Math. Model. Numer. Anal.*, 43(6) :1027–1044, 2009. 109

[328] Johannes Ulen, Petter Strandmark, and Fredrik Kahl. Shortest paths with higher-order regularization. *IEEE transactions on pattern analysis and machine intelligence*, 37(12) :2588–2600, 2015. 12

[329] Simone Venturi and Tiernan Casey. Svd perspectives for augmenting deeponet flexibility and interpretability. *arXiv preprint arXiv :2204.12670*, 2022. 55

[330] H.-J. Wagner. A contribution to the numerical approximation of minimal surfaces. *Computing*, 19(1) :35–58, 1977/78. 139

[331] Dong Wang, Xiao-Ping Wang, and Xianmin Xu. An improved threshold dynamics method for wetting dynamics. *J. Comput. Phys.*, 392 :291–310, 2019. 109

[332] Xiaoyu Wei, Shidong Jiang, Andreas Klöckner, and Xiao-Ping Wang. An integral equation method for the Cahn-Hilliard equation in the wetting problem. *J. Comput. Phys.*, 419 :109521, 16, 2020. 109

[333] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M Benson. U-fno—an enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163 :104180, 2022. 55

[334] S. M. Wise, C. Wang, and J. S. Lowengrub. An energy-stable and convergent finite-difference scheme for the phase field crystal equation. *SIAM J. Numer. Anal.*, 47(3) :2269–2288, 2009. 78, 101

[335] Xianmin Xu, Dong Wang, and Xiao-Ping Wang. An efficient threshold dynamics method for wetting on rough surfaces. *Journal of Computational Physics*, 330 :510–528, 2017. 44, 45

[336] Xianmin Xu and Wenjun Ying. An adaptive threshold dynamics method for three-dimensional wetting on rough surfaces. *Commun. Comput. Phys.*, 29(1) :57–79, 2021. 109

[337] Neha Yadav, Anupam Yadav, Manoj Kumar, et al. *An introduction to neural network methods for differential equations*, volume 1. Springer, 2015. 32

[338] Junxiang Yang and Junseok Kim. An unconditionally stable second-order accurate method for systems of Cahn-Hilliard equations. *Commun. Nonlinear Sci. Numer. Simul.*, 87 :105276, 17, 2020. 101, 108

[339] Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 94 :103–114, 2017. 34

[340] Thomas Young. An Essay on the Cohesion of Fluids. *Philosophical Transactions of the Royal Society of London Series I*, 95 :65–87, January 1805. 43, 89, 108

[341] B Yu. The deep ritz method : A deep learning-based numerical algorithm for solving variational problems, commun. math. *Stat*, 6 :1–12, 2018. 121

[342] Alexander Zaitzeff, Selim Esedoğlu, and Krishna Garikipati. High order, semi-implicit, energy stable schemes for gradient flows. *Journal of Computational Physics*, 447 :110688, 2021. 29, 121, 125

[343] Jiahao Zhang, Shiqi Zhang, and Guang Lin. Multiauto-deeponet : A multi-resolution autoencoder deeponet for nonlinear dimension reduction, uncertainty quantification and operator learning of forward and inverse stochastic problems. *arXiv preprint arXiv :2204.03193*, 2022. 55

[344] Kai Zhang, Yuande Zuo, Hanjun Zhao, Xiaopeng Ma, Jianwei Gu, Jian Wang, Yongfei Yang, Chuanjin Yao, and Jun Yao. Fourier neural operator for solving subsurface oil/water two-phase flow partial differential equation. *SPE Journal*, pages 1–15, 2022. 55

[345] Hongkai Zhao. A fast sweeping method for eikonal equations. *Mathematics of computation*, 74(250) :603–627, 2005. 25

[346] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and computational harmonic analysis*, 48(2) :787–794, 2020. 35

[347] Shiwei Zhou and Michael Yu Wang. Multimaterial structural topology optimization with a generalized Cahn–Hilliard model of multiphase transition. *Structural and Multidisciplinary Optimization*, 33(2) :89, 2006. 101

[348] Jingzhi Zhu, Long-Qing Chen, Jie Shen, and Veena Tikare. Coarsening kinetics from a variable-mobility Cahn-Hilliard equation : Application of a semi-implicit Fourier spectral method. *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*, 60 :3564–72, 11 1999. 101

[349] Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366 :415–447, 2018. 121

[350] Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394 :56–81, 2019. 54

[351] Carsten Zwilling. *The diffuse interface approximation of the Willmore functional in configurations with interacting phase boundaries*. PhD thesis, 2018. 145