EDPs numériques pour l'analyse d'images

Jean-Marie Mirebeau*

January 18, 2021

Abstract

L'objectif du cours est de présenter des outils et méthodes permettant la résolution numérique efficace d'équations aux dérivées partielles, en vue d'applications au traitement de l'image.

Contents

1 Introduction

Les équations aux dérivées partielles (EDPs) sont l'un des formalismes mathématiques permettant de passer du local - en décrivant un comportement à l'échelle infinitésimale - au global - par la résolution d'un système d'équations couplées définies en chaque point d'un domaine. Elles sont incontournables dans certains domaines, comme la physique des milieux continus. Nous donnons dans cette section leur pertinence dans le cadre du traitement de l'image, et donnons un aperçu des exemples et méthodes qui seront traités dans le cours.

1.1 Pertinence des EDPs en traitement d'image

De nombreuses approches mathématiques sont pertinentes dans le traitement de l'image et du signal, comme les statistiques et probabilités, l'optimisation et l'apprentissage, ou encore différentes branches de l'analyse. Voici certaines des raisons qui justifient l'intérêt des EDPs dans ce contexte.

- *Physique des dispositifs d'acquisition.* De nombreux dispositifs d'imagerie sont fondés sur la physique des milieux continus (vibrations, absorption lumineuse, ...) qui est décrite de manière directe par des EDPs. En particulier les méthodes *tomographie*, qui consistent à

*Université Paris-Saclay, ENS Paris-Saclay, CNRS, Centre Borelli, F-91190 Gif-sur-Yvette, France.

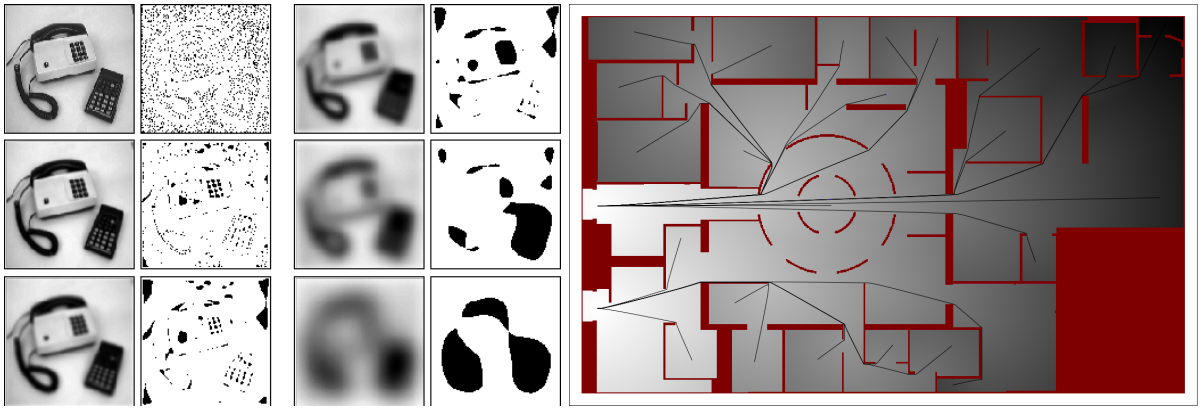

Figure 1: (Gauche) Equation de la chaleur et représentation multi-échelle d'une image. Crédit image: Tony Lindeberg. (Droite) Solution de l'équation eikonale (niveaux de grid) dans un domaine avec obstacles (rouge), et chemins minimaux.

"reconstruire volume d'un objet à partir d'une série de mesures effectuées depuis l'extérieur de cet objet" (Wikipedia), utilisée en imagerie médicale ou sismique.
Exemple : La transformée de Radon, utilisée en tomographie axiale.
Exemple : La reconstruction de relief à partir des ombres (Shape from shading).

- *Modèle interprétable et explicatif.* Par analogie avec des modèles physiques, les EDPs permettent de définir un cadre interprétable et explicatif pour le traitement de données.
  Exemple : Description multi-échelle d'une image, via un processus de diffusion.
  Exemple : Les courbes elastica de Euler, qui correspondent à la position de repos d'une barre élastique, sont utilisées pour l'extraction de contours et de courbes dans des images.

- *Reconstruction d'information globale.* On peut considérer une EDP comme un système d'équations couplées, où chaque point du domaine porte (typiquement) une inconnue et une équation. La résolution d'un tel système est un objet global, qui synthétise les contraintes imposées localement.
  Exemple : La résolution de l'équation eikonale permet de déterminer le plus court chemin dans un domaine contenant des obstacles et des zones plus ou moins rapides.

- *Structure mathématique, garanties.* Les EDPs ont une structure mathématique riche qui permet d'établir un certain nombre de garanties : terminaison de l'algorithme, robustesse au bruit, validité sur des cas simples, etc

1.2 Les EDPs considérées dans ce cours

Dans ce cours nous étudierons les schémas de discrétisation, la mise en oeuvre numérique, et les applications, de principalement deux équations. Ce sont l'équation de la chaleur, et l'équation eikonale, qui sous leur forme la plus simple s'écrivent respectivement

$$\partial_t u = \Delta u, \qquad \|\nabla u\| = 1, \tag{1}$$

avec des conditions au bord appropriées.

- L'équation de la chaleur s'interpête comme le flot gradient (descente de gradient) de l'énergie de Dirichlet $\int_\Omega \|\nabla u\|^2$. L'équation de la chaleur lisse les changements brusques

dans la fonction u; dans le cas d'une image elle élimine le bruit, mais rend flous les contours d'objets et les textures oscillantes. L'équation de la chaleur permet de séparer, au fil du temps, les différentes échelles d'une image.

- L'équation eikonale caractérise la fonction distance Euclidienne u à un point ou une sous région source. Le gradient de sa solution $\nabla u(x)$ donne la direction opposée à la source. L'équation eikonale est le modèle le plus simple de propagation de front, et remonter son gradient permet de calculer des plus courts chemins.

L'équation de la chaleur et l'équation eikonale standard (1) traitent toutes les directions d'espace de manière identique: on dit qu'elles sont *isotropes*. Plus précisément, elles sont définies en chaque point par des opérateurs différentiels Δu et $\|\nabla u\|$, qui sont invariants par rotation. Cette propriété de symétrie permet de discrétiser opérateurs y intervenant de manière particulièrement simple: pour toute fonction u assez lisse

$$\Delta u(x) = \sum_{1\leq i\leq n} \frac{u(x+he_i) - 2u(x) + u(x-he_i)}{h^2} + \mathcal{O}(h^2),$$
$$\|\nabla u(x)\|^2 = \sum_{1\leq i\leq n} \frac{\max\{0, u(x)-u(x-he_i), u(x)-u(x+he_i)\}^2}{h^2} + \mathcal{O}(h).$$

Le choix des différences finies utilisées est non-trivial, et sera justifié plus loin dans le cours.

Anisotropie. On dit qu'un problème est anisotrope lorsque les directions d'espace dans un domaine ne sont pas interchangeables. C'est un phénomène générique, dont les causes sont variées:

- *Micro-structure.* Certains milieux physiques sont micro-structurés, ce qui affecte les ondes s'y propageant de manière anisotrope, aux échelles supérieures par un effet appelé homogénéisation. Ondes sismiques dans un milieu stratifié, ondes lumineuses dans un crystal, ... En imagerie, ce type de phénomène est parfois reproduit artificiellement, par exemple lors du traitement de textures oscillantes comme dane une image d'empreinte digitale.

- *Rôle distinct des dimensions du domaine.* Certains domaines mathématiques ne correspondent pas à un milieu physique, mais à un espace d'états dont les dimensions jouent des rôles différents, ce qui crée des structures anisotropes. Par exemple l'espace $\mathbb{R}^2 \times \mathbb{S}^1$ des positions et orientations, qui corresponds aux configurations d'un véhicule simple, possède une structure dite sous-riemannienne, permettant en (x_0, x_1, θ) seulement les déplacements engendrés par $(\cos\theta, \sin\theta, 0)$ et $(0, 0, 1)$.

- *Proximité des bords ou de fissures.* La modélisation des écoulements au bord d'un domaine, ou de l'élasticité au voisinage d'une fissure, se fait souvent par l'intermédiaire de modèles réduits traitant de manière spécifique la dimension tangente.

- *Paramétrisation d'un domaine complexe.* La résolution de d'équations dans des géométries complexes, surfaces ou volumes, peut se faire par l'intermédiaire de leur paramétrisation par des domaines de référence (rectangles) permettant un traitement plus simple des conditions au bord. Dans ce cas, la paramétrisation induit en général une distortion anisotrope de l'EDP sur le domaine de référence, même si elle était isotrope sur la géométrie initiale.

Figure 2: Exemples d'anisotropies associées à une micro-structure. (Gauche) Empreinte digitale, (droite) minéral mica. Images wikipedia.

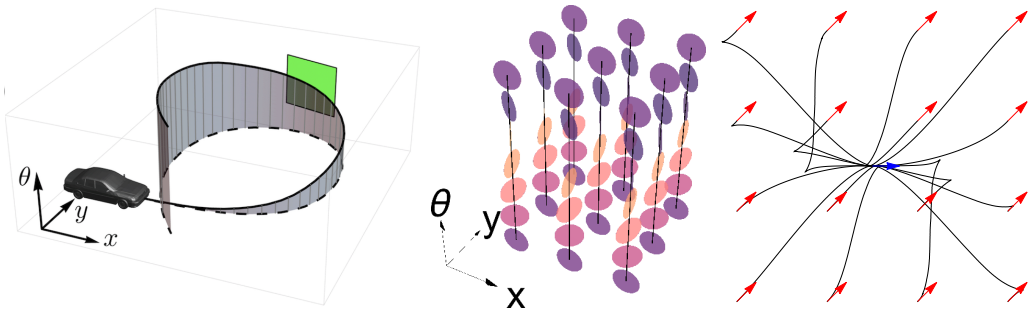

Figure 3: Exemple d'anisotropie liée au rôle distinct des dimensions du domaine. Ici le véhicule de Reeds-Shepp, dont l'espace d'états est $\mathbb{R}^2 \times \mathbb{S}^1$. (Gauche) Une trajectoire admissible. (Centre) Boule unité dans l'espace des vitesses admissibles. (Droite) Projection planaire de quelques trajectoires optimales.

- *Linéarisation d'un problème.* Certaines EDPs linéaires sont obtenues par perturbation d'une EDP non-linéaire au voisinage d'une solution. Selon les cas, l'équation linéarisée peut être anisotrope même si l'équation non-linéaire était isotrope. Par exemple, dans le cas de Monge-Ampère, on a formellement $\det(\nabla^2(u+h)) - \det(\nabla^2 u) = \mathrm{Tr}(D(u)\nabla^2 h) + \mathcal{O}(h^2)$ où $D(u)$ est la comatrice de $\nabla^2 u$.

Formellement, l'introduction d'anisotropie dans une EDP se fait par l'introduction de champs qui décrivent la géométrie locale du problème. Par exemple si D est un champ de matrices symétriques définies positives, alors on peut considérer les variantes anisotropes de l'équation de la chaleur et de l'équation eikonale définies par

$$\partial_t u = \mathrm{div}(D\nabla u), \qquad\qquad \langle \nabla u, D\nabla u \rangle = 1, \tag{2}$$

avec de nouveau des conditions au bord appropriées. Ces variantes favorisent la diffusion (chaleur) ou la propagation du front (eikonal) dans les directions associées aux grandes valeurs propres de D.

2 Différentiation numérique

L'analyse numérique et l'optimisation requièrent de calculer numériquement les dérivées de fonctions. A cet effet, plusieurs méthodes existent: *différences finies*, *différentiation automatique*, ou

simplement la différentiation symbolique suivie de l'implémentation des formules résultantes. La discussion présentée ci-dessous permet de faire un choix éclairé, et est suivie d'une description plus détaillée de la différentiation automatique.

Différences finies. On appelle différences finies les combinaisons (en général linéaires) de valeurs ponctuelles d'une fonction, destinées à approcher ses dérivées.

Par exemple, si $f : \mathbb{R} \to \mathbb{R}$ est lisse, si $x \in \mathbb{R}$ et h est petit, alors on peut considérer les différences finies *upwind*, *centrée*, et d'*ordre deux* définies par

$$\begin{aligned} \frac{f(x+h) - f(x)}{h} &= f'(x) + \mathcal{O}(h), \\ \frac{f(x+h) - f(x-h)}{2h} &= f'(x) + \mathcal{O}(h^2), \\ \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} &= f''(x) + \mathcal{O}(h^2) \end{aligned}$$

Les différences finies se caractérisent par l'ordre de la dérivée qu'elles approchent (ici 1,1,2), et leur ordre de précision (ici 1,2,2). Une autre approche conceptuellement proche, pour approcher les dérivées d'une fonction définie ponctuellement, consiste à interpoler cette fonction (par éléments finies, splines, etc), puis à différentier cette interpolation.

Intérêts: Les différences finies permettent d'approcher les dérivées de fonctions qui ne sont définies que *ponctuellement*. Avec les éléments finis et autres méthodes conceptuellement proches, elles sont à la fondation des schémas numériques pour la discrétisation des équations aux dérivées partielles. Leur structure mathématique simple et (en général) linéaire permet d'étudier mathématiquement les quantités qui en sont dérivées.

Inconvénients: L'approximation des dérivées d'une fonction par différences finies, et autres méthodes proches, nécessite de faire des compromis entre la *précision* et la *stabilité*.

Différentiation automatique. On appelle différentiation automatique les procédés informatiques permettant de calculer les dérivées de fonctions définies par des programmes, en utilisant les dérivées exactes des fonctions usuelles (exp, $\sqrt{\cdot}$, sin, etc) et en appliquant les règles de composition des différentielles.

Intérêts: La différentiation automatique permet de calculer les dérivées de fonctions complexes de manière, en général, stable et précise. Grâce à des techniques comme la surcharge des opérateurs et des fonctions usuelles, sont utilisation n'altère que peu la lisibilité des programmes. Son coût numérique est souvent raisonnable.

Inconvénients: Il existe au moins trois variétés de différentiation automatique: dense, creuse, et par rétro-propagation; elles peuvent être composées entre elles. La mise en oeuvre doit être réfléchie en fonction des conditions d'utilisation, sous peine de coût de calcul excessif.

Différentiation formelle et implémentation. Lorsqu'une fonction apparaissant dans un programme informatique possède une expression simple, il peut sembler naturel de calculer ses dérivées en la dérivant formellement et en implémentant l'expression mathématique résultante.

Intérêt: Cette approche mène, parfois, à l'implémentation la plus efficace en termes de temps de calcul.

Inconvénient: Cette méthode est **à proscrire**[1] **en première approche**, en faveur de la différentiation automatique. En effet, elle rend le programme peu lisible, peu flexible, long à

[1] Rappelons que les qualités à rechercher lors de la conception d'un programme sont, dans l'ordre:

1. La *lisibilité*, qui permet à d'autres programmeurs (voire vous-même) de continuer votre travail, et leur

écrire, et introduit de nombreux bugs,. Lorsqu'elle nécessaire, ses résultats doivent a minima être contrôlés sur des exemples par une approche alternative.

2.1 Rappels: différentielle et gradient

On rappelle pour fixer les conventions les définitions élémentaires des différentielle, gradient, hessienne d'une fonction. Dans cette sous-section, les lettres E et F désignent des espaces vectoriels normés (evn). Ce point ne sera pas rappelé systématiquement.

Définition 2.1. *Soient E, F evn. On note $\mathcal{L}(E,F)$ l'ensemble des applications linéaires continues de E dans F, qui est aussi un evn.*

Définition 2.2 (Différentielle)**.** *Une fonction $f : \Omega \to F$, où $\Omega \subseteq E$ est ouvert, est différentiable en $x \in \Omega$ s'il existe $L \in \mathcal{L}(E,F)$ telle que*

$$f(x+h) = f(x) + L(h) + o(\|h\|)$$

On note $L = \mathrm{d}f_{|x}$, appelée différentielle de f en x.

Définition 2.3 (Gradient)**.** *Soit $f : \Omega \to \mathbb{R}$, où Ω est un ouvert d'un espace de Hilbert $\mathbb{H}$, différentiable en $x \in \Omega$. Le gradient de f en x, noté $\nabla f(x) \in \mathbb{H}$, est défini par l'identité*

$$\langle \nabla f(x), v \rangle = \mathrm{d}f_{|x}(v)$$

pour tout $v \in H$. En d'autres terms $f(x+h) = f(x) + \langle \nabla f(x), h \rangle + o(h)$.

Exemple : soit H un Hilbert, et $f : H \to \mathbb{R}$ définie par $f(x) = \frac{1}{2}\|x\|^2$. On note que $f(x+h) = \frac{1}{2}\|x\|^2 + \langle x, h\rangle + \frac{1}{2}\|h\|^2$. On en déduit que $\nabla f(x) = x$, pour tout $x \in \mathbb{H}$.

Définition 2.4 (Matrice jacobienne)**.** *Soit $f : \Omega \to \mathbb{R}^n$, où $\Omega \subseteq \mathbb{R}^m$, différentiable en $x \in \Omega$. On appelle matrice jacobienne de f en x, notée $\mathrm{D}f_{|x}$, la matrice de $\mathrm{d}f_{|x}$ dans les bases canoniques de $\mathbb{R}^m$ et $\mathbb{R}^n$. Ses composantes sont appelées dérivées partielles de f*

$$\mathrm{D}f_{|x} = \left(\frac{\partial f_i}{\partial x_j}(x)\right)_{\substack{1\le i\le n\\ 1\le j\le m}}$$

En l'absence d'ambiguité (choix de base), on ne se gènera pas pour identifier $\mathrm{d}f_{|x}$ et $\mathrm{D}f_{|x}$.
Exemple: si $f : \mathbb{R}^m \to \mathbb{R}$, alors $\nabla f(x) = (\mathrm{D}f_{|x})^T$

Lemme 2.5 (Composition)**.** *Soient $f : E \to F$ et $g : F \to G$. Supposons f différentiable en $x \in E$, et g différentiable en $f(x) \in F$. Alors $g \circ f : E \to G$ est différentiable en x et*

$$\mathrm{d}(g \circ f)_{|x} = \mathrm{d}g_{|f(x)} \circ \mathrm{d}f_{|x}.$$

La différentielle d'une composée est donc la composée des différentielles; de même pour les matrices jacobiennes, sous des hypothèses adéquates, $\mathrm{D}(g \circ f)_{|x} = \mathrm{D}g_{|f(x)}\mathrm{D}f_{|x}$.

---

donne confiance quant à sa qualité.

2. La *robustesse*, c'est à dire les garanties que l'on peut apporter concernant l'éxécution du programme, et sa gestion des erreurs.
3. La *généricité*, qui permet au programme d'être utilisé au sein d'applications non considérées initialement.
4. La *rapidité de conception*, par le choix des bons outils, car le temps humain a plus de valeur que le temps machine.
5. La *rapidité d'exécution.* La recherche de cette qualité ne doit pas se faire au détriment des précédentes.

2.2 Préliminaire: coût d'un produit matriciel

Étant données deux matrices, A de taille $I \times J$ et B de taille $J \times K$, leur produit AB de taille JK est défini par

$$(AB)_{ik} = \sum_{1 \leq j \leq J} A_{ij} B_{jk} \tag{3}$$

pour tous $1 \leq i \leq I$, $1 \leq k \leq K$. Le coût du calcul de AB est par la méthode "naive" est

$$\mathcal{O}(IJK).$$

Considérons $A_1, \cdots, A_n$ des matrices de taille $I_0 \times I_1, \cdots, I_{n-1} \times I_n$. Leur produit est associatif, et peut donc être parenthésé de manière arbitraire

$$((A_1 A_2) A_3) \cdots A_n = A_1 \cdots A_n = A_1 (A_2 (\cdots A_n)). \tag{4}$$

Le coût de l'évaluation de l'expression parenthésée à gauche ou à droite est respectivement

$$\mathcal{O}(I_0(I_1 I_2 + I_2 I_3 + \cdots + I_{n-1} I_n)), \qquad \mathcal{O}((I_0 I_1 + \cdots + I_{n-2} I_{n-1}) I_n). \tag{5}$$

En particulier, si $I_0 = 1$, c'est à dire si la première matrice est un vecteur ligne, alors le parenthésage à gauche est optimal car son coût (5, gauche) correspond au coût de la lecture des données. De même manière, si $I_n = 1$, c'est à dire si la dernière matrice est un vecteur colonne, alors le parenthésage à droite est optimal.

Dans le cas général, trouver le parenthésage optimal pour minimiser le coût de calcul est un problème NP-complet.

Cas de matrices creuses Une matrice creuse possède peu de coefficients non-nuls, qui peuvent être stockés dans une structure de données adaptée, permettant une manipulation numérique efficace. Les matrices de schémas numériques pour la discrétisation des EDPs sont fréquemment de cette forme. La propriété d'être creux est partiellement compatible avec le produit matriciel, comme le montre le résultat suivant.

Lemme 2.6. *Soit A (resp. B) une matrice de taille $I \times J$ (resp. $J \times K$) dont chaque ligne au plus α (resp. β) coefficients non-nuls. Alors chaque ligne de AB contient au plus $\alpha\beta$ coefficients non-nuls.*

Proof. Soit $\mathcal{A}$ (resp. $\mathcal{B}$) la matrice obtenue en remplaçant les coefficients non-nuls de A (resp. B) par 1. Alors $\mathcal{AB}$ est une matrice dont les coefficients sont entiers, positifs, et non-nuls à chaque position où AB a un coefficient non-nul. Par ailleurs la somme des coefficients de $\mathcal{AB}$ sur la ligne d'indice i, où $1 \leq i \leq I$, vaut

$$\sum_{1 \leq k \leq K} (\mathcal{AB})_{ik} = \sum_{1 \leq j \leq J} \left(\mathcal{A}_{ij} \sum_{1 \leq k \leq K} \mathcal{B}_{jk} \right) \leq \sum_{1 \leq j \leq J} \mathcal{A}_{ij} \beta \leq \alpha\beta.$$

Le résultat annoncé s'ensuit. □

Par transposition, on obtient un résultat analogue au Lemme 2.6 pour les colonnes. Par une récurrence immédiate, si $A_1, \cdots, A_n$ sont des matrices dont chaque ligne contient au plus $\alpha_1, \cdots, \alpha_n$ coefficients non-nuls, alors leur produit $A_1 \cdots A_n$ est contient au plus $\alpha_1 \cdots \alpha_n$ coefficients non-nuls sur chaque ligne. Donc au total pas plus de

$$I_0 \alpha_1 \cdots \alpha_n \tag{6}$$

coefficients non-nuls, en notant I_0 le nombre de lignes de A_0. Avec une implémentation raisonnable du calcul du produit de matrices de creuses, la quantité (6) borne aussi le coût du calcul de $A_1 \cdots A_n$, indépendamment de l'ordre d'associativité utilisé.

Cette estimation montre avant tout que l'utilisation des produits creux doit être réservée à un nombre très faible de facteurs très creux, car le taux de remplissage augmente de manière exponentielle au fil des produits.

Interprétation en tant que Jacobienne. Soient $f_1 : \mathbb{R}^{I_1} \to \mathbb{R}^{I_0}$, $\cdots$, $f_n : \mathbb{R}^{I_n} \to \mathbb{R}^{I_{n-1}}$ des fonctions, et soit $x_n \in \mathbb{R}^{I_n}$. Supposons f_i différentiable au point x_i, où $x_{i-1} := f_i(x_i)$ pour tout $1 \leq i \leq n$. Alors

$$\mathrm{D}(f_1 \circ \cdots \circ f_n)_{|x_n} = \mathrm{D}f_{1|x_1} \cdots \mathrm{D}f_{n|x_n}.$$

Les programmes informatiques décrivent des fonctions complexes comme composée de fonctions élémentaires. Leur différentiation automatique peut donc s'interpréter[2] comme le produit matriciel des matrices Jacobiennes des étapes intermédiaires. Selon la structure de ces facteurs, on préfèrera utiliser un produit associatif à gauche, ou à droite, ou un produit creux; voire une combinaison de ces approches suivant les parties du facteur.

2.3 Trois approches de la différentiation automatique

On peut distinguer trois approches principales de la différentiation automatique: *dense*, *creuse*, et par *rétro-propagation*. Elles correspondent conceptuellement aux trois stratégies détaillées §2.2 pour le calcul d'un produit matriciel: associativité à droite, à gauche, ou produit creux, ce qui permet d'anticiper leurs forces et leurs faiblesses respectives. Cependant leur implémentation numérique s'éloigne de ce cadre, car pour plus de commodité et d'efficacité les matrices Jacobiennes des étapes intermédiaires ne sont en général pas construites explicitement.

Dans la suite, on suppose que l'on cherche à calculer la matrice jacobienne d'une fonction

$$f : \mathbb{R}^m \to \mathbb{R}^n, \tag{7}$$

définie via un programme informatique, donc comme composition de fonctions élémentaires qui ne seront pas explicitées ici. (Pour l'analogie avec le produit matriciel (5) des jacobiennes, on note que (m, n) correspondent à (I_n, I_0).)

Différentiation automatique dense (m petit).

Analogue matriciel. La différentiation automatique dense correspond au produit matriciel (4) par associativité à droite; on parle aussi de propagation *forward* puisque c'est le sens naturel de l'exécution du programme. Elle est particulièrement efficace lorsque le nombre d'entrées m de la fonction f est petit, par analogie avec (5, droite).

Utilisations. La différentiation automatique dense est idéale pour différentier des fonctions utilitaires en basse dimension (m, n petits). Elle convient aussi sur le principe pour analyser la dépendance d'une sortie en grande dimension ($n \gg 1$), par exemple la simulation numérique d'un problème physique, par rapport à un petit nombre m de paramètres.

Implémentation. La différentiation automatique dense s'implémente en replaçant les scalaires en entrée du programme définissant f par des paires $(x, v) \in \mathbb{R} \times \mathbb{R}^m$ représentant le développement limité

$$x + \langle v, h \rangle + o(h).$$

[2] Même si elle n'est pas implémentée de cette manière en apparence, voir la sous-section suivante.

La variable $h \in \mathbb{R}^m$ représente ici une perturbation *symbolique*, qui n'a pas d'existence dans le programme informatique. La surcharge des opérateurs et des fonctions usuelles permet d'appliquer les règles de calcul des développements limités sans avoir à ré-écrire la fonction. Par exemple

$$\begin{aligned}\sin(x + \langle v, h\rangle + (h)) &= \sin(x) + \cos(x)\langle v, h\rangle + o(\|h\|)\\ \big(x + \langle v, h\rangle + o(h)\big)\big(x' + \langle v', h\rangle + o(h)\big) &= xx' + \langle xv' + x'v, h\rangle + o(h).\end{aligned}$$

Différentiation automatique creuse (f simple).

Analogue matriciel. La différentiation automatique creuse correspond au produit matriciel creux (et par associativité à droite, pour la simplicité). Elle est particulièrement efficace lorsque f a une structure très simple, faisant intervenir peu d'étapes internes, chacune ne dépendant que de quelques variables. Sous ces conditions, elle permet de traiter des entrées et sorties de grande taille.

Utilisations. La différentation automatique creuse est idéale pour assembler les matrices Jacobiennes schémas numériques pour des équations aux dérivées partielles.

Implémentation. La différentiation automatique creuse s'implémente remplaçant les scalaires en entrée du programme définissant f par des paires $(x, \alpha, i) \in \mathbb{R}\times\mathbb{R}^K\times\{1, \cdots, m\}^K$ représentant le développement limité

$$x + \sum_{1\le k\le K} \alpha_k h_{i_k} + o(\|h\|).$$

De nouveau, la variable $h \in \mathbb{R}^m$ représente une perturbation *symbolique*, qui n'a pas d'existence dans le programme informatique. La surcharge des opérateurs et des fonctions usuelles permet d'appliquer les règles de calcul des développements limités. Par exemple

$$\begin{aligned}\sin\Big(x + \sum_{1\le k\le K} \alpha_k h_{i_k} + o(\|h\|)\Big) &= \sin(x) + \sum_{1\le k\le K} \cos(x)\alpha_k h_{i_k} + o(\|h\|)\\ \Big(x + \sum_{1\le k\le K} \alpha_k h_{i_k} + o(\|h\|)\Big) + \Big(x' + \sum_{1\le k\le K} \alpha'_k h_{i'_k} + o(\|h\|)\Big)&\\ = x + x' + \Big(\sum_{1\le k\le K} \alpha_k h_{i_k} + \sum_{1\le k\le K} \alpha'_k h_{i'_k}\Big) + o(\|h\|).&\end{aligned}$$

En particulier, la somme est représentée par $(x + x',\ \alpha \oplus \alpha',\ i \oplus i')$, où l'opérateur $\oplus$ désigne la concaténation de vecteurs. (Cette représentation pourra éventuellement être simplifiée en regroupant les poids associés à des indices redondants.)

Par rétro-propagation (n petit).

Analogue matriciel. La différentiation automatique par rétro-propagation correspond au produit matriciel par associativité à gauche. Pour cette raison, elle est particulièrement efficace lorsque la sortie est de petite dimension, sans limite sur la taille de l'entrée.

Utilisations. La différentiation automatique par rétro-propagation est particulièrement utile pour les problèmes d'optimisation, car la sortie est alors de dimension 1. En particulier, elle est systématiquement utilisée pour l'entrainement des réseaux de neurones.

Implémentation. La différentiation automatique par rétro-propagation nécessite de rejouer les calculs dans l'ordre inverse de leur exécution initiale, et donc de les organiser dans un historique ou un graphe orienté. Pour cette raison elle est plus complexe à mettre en oeuvre que la différentiation dense ou creuse. Par ailleurs, la conservation de la totalité des états intermédiaires des variables a un coût mémoire potentiellement important, qui peut être réduit par des

re-calculs partiels, ce qui mène à des compromis et astuces d'implémentation non-triviaux. Des libraries comme PyTorch implémentent ces techniques.

2.4 Dérivées d'ordre deux et supérieur

Rappels : définitions et propriétés élémentaires.

Définition 2.7 (Différentielle d'ordre supérieur)**.** *Soient E, F des evn, et soit $f : \Omega \to F$ où $\Omega \subseteq E$ ouvert, différentiable en tout point de Ω. Si $\mathrm{d}f : \Omega \to \mathcal{L}(E, F)$ est différentiable au point $x \in \Omega$, alors on note $d^2 f_{|x} \in \mathcal{L}(E, \mathcal{L}(E, F))$ sa différentielle.*

On note que $\mathcal{L}(E_0, L(E_1, F))$ s'identifie à $\mathcal{L}^2(E_0 \times E_1, F)$, espace vectoriel des applications bilinéaires continues de $E_0 \times E_1$ dans F.

Théorème 2.8 (Schwartz)**.** *Soit $f : \Omega \to F$, où $\Omega \subseteq E$ est ouvert, telle que $\mathrm{d}^2 f : \Omega \to \mathcal{L}^2(E \times E, F)$ existe et est continue. Alors $\mathrm{d}^2 f_{|x} \in \mathcal{L}^2(E \times E, F)$ est une forme bilinéaire* symétrique*, pour tout $x \in \Omega$.*

Sous les hypothèse du Théorème de Schwartz, on a le développement limité suivant, qui permet aussi de caractériser la différentielle seconde par identification

$$f(x + h) = f(x) + \mathrm{d}f_{|x}(h) + \frac{1}{2}\mathrm{d}^2 f_{|x}(h, h) + o(\|h\|^2).$$

Définition 2.9 (Matrice hessienne)**.** *Soit $f : \mathbb{R}^n \to \mathbb{R}$ deux fois continument différentiable au voisinage de $x \in \mathbb{R}^n$. La matrice de la forme quadratique $\mathrm{d}^2 f_{|x}$ dans la base canonique est appelée matrice hessienne, et noté $\mathrm{D}^2 f_{|x}$.*

Exemple: la Hessienne de $f : \mathbb{R}^n \to \mathbb{R}$ definie par $f(x) = \frac{1}{2}\|x\|^2$, est la matrice identité.

Différentiation automatique.

La différentiation automatique d'ordre deux, ou éventuellement supérieur, se justifie dans des cas particuliers dont voici quelques exemples:

- *Résolution de problèmes d'optimisation par la méthode de Newton.* Pour les problèmes d'optimisation ayant de bonnes propriétés mathématiques, dans l'idéal convexes, réguliers et non-contraints, l'efficacité de la méthode de Newton est souvent sans égal. Sa mise en oeuvre requiert d'évaluer les dérivées de la fonction à minimiser jusqu'à l'ordre deux.

- *Différentiation d'un programme faisant lui-même intervenir la différentiation automatique.* Exemple : trouver la géodésique joignant deux points donnés, par une méthode de tir. Les géodésiques sont des courbes obéissant aux equations de Hamilton, qui jouent un rôle fondamental en physique et en mathématiques

  $$\partial_t q = \partial_p H, \qquad \partial_t p = -\partial_q H.$$

  Il est naturel d'utiliser la différentiation automatique pour dériver le Hamiltonien H, qui encode la géométrie du problème, et implémenter ces équations. Dans le cadre des méthodes de tir, on ajuste le moment initial p_0 (la position initiale q_0 étant fixée), pour atteindre une position finale $q(1)$ donnée; on peut utiliser pour cela méthode de Newton, ce qui requiert une seconde différentiation automatique.

- *Discrétisation de problèmes variationnels.* Certaines équations aux dérivées partielles sont présentées sous forme variationnelle: par exemple trouver $u \in H^1(\Omega)$ (espace de Sobolev sur un domaine Ω) tel que pour tout $v \in H^1(\Omega)$ on ait

$$\int_\Omega \nabla u \nabla v = \int_\Omega f v.$$

  La construction automatique du schéma numérique, à partir d'une fonction implémentant une approximation numérique de ces intégrales (par différences finies, éléments finis, etc), requiert la différentiation d'ordre deux.

Les trois approches de la différentiation automatique, dense, creuse, et par rétro-propagation, s'étendent à l'ordre deux. Dans les deux premiers cas, il s'agit de remplacer les scalaires par des variables (x, v, m) ou $(x, \alpha, i, \beta, j, k)$ représentant des développements limités d'ordre deux

$$x + \langle v, h\rangle + \frac{1}{2}\langle h, mh\rangle + o(h^2), \qquad x + \sum_{1\le r\le R} \alpha_r h_{i_r} + \frac{1}{2}\sum_{1\le s\le S} \beta_s h_{j_s} h_{k_s} + o(h^2),$$

et de surcharger les opérateurs et fonctions usuelles. La différentiation automatique par rétro-propagation à l'ordre deux, bien que possible, semble peu usitée.