CSMA 2024
16ème Colloque National en Calcul des Structures
13-17 Mai 2024, Presqu'île de Giens (Var)

Pré-dimensionnement de structures aéronautiques : une nouvelle perspective grâce à la combinaison de la réduction de modèles et de l'apprentissage profond sur graphes

V. Matray[1], F. Amlani[1], F. Feyel[1,2], D. Néron[1]

[1] *Université Paris-Saclay, CentraleSupélec, ENS Paris-Saclay, CNRS*
LMPS - Laboratoire de Mécanique Paris-Saclay, 91190, Gif-sur-Yvette, France.
Mail : {victor.matray, faisal.amlani, david.neron}@ens-paris-saclay.fr
[2] *Safran Tech, Digital Sciences & Technologies Department, Rue des jeunes bois, Châteaufort, 78114, Magny-les-Hameaux, France.*
Mail : frédéric.feyel@safrangroup.com

Résumé — Ce travail présente une preuve de concept combinant la réduction de modèles et l'apprentissage profond basé sur les graphes (Graph Neural Networks) pour le pré-dimensionnement des structures aéronautiques. Notre approche utilise des données industrielles limitées pour l'entraînement des modèles d'apprentissage profond et intègre des méthodes éprouvées par l'industrie pour garantir la pertinence de la solution. L'étude se concentre sur la conception de sièges d'avions lors de simulations de crash, sans paramétrage préalable, permettant une large variété de géométries.
Mots clés — Réduction de modèles, Apprentissage profond sur graphes.

1 Intoduction et enjeux

Dans le domaine de la conception de structures aéronautiques, il serait fortement avantageux pour les bureaux d'études de disposer d'un outil de calcul leur permettant de pré-dimensionner rapidement une pièce mécanique dès la phase de conception initiale. Un tel outil permettrait d'éviter les ajustements itératifs dans le processus de conception d'un nouveau projet, ce qui conduirait à des gains de temps significatifs lors des phases de validation. De plus, il offrirait aux équipes de développement une marge de manœuvre pour proposer des conceptions originales, voire radicalement différentes de ce qui est habituellement envisagé. Dans cette optique, nous présentons ce travail qui a été mené en étroite collaboration avec Safran Tech.

Notre approche vise à fusionner deux méthodes distinctes : la réduction de modèles ou *Model Order Reduction* (MOR) [3, 7, 13] et l'apprentissage profond basé sur les graphes ou *Graph Neural Networks* (GNN) [1, 2, 4]. Alors que la première repose principalement sur des principes physiques et mathématiques pour résoudre plus efficacement les équations en jeu, la seconde, bien qu'ignorant initialement ces équations physiques, présente une capacité prometteuse d'extrapolation vers des données inconnues [6, 15] ainsi qu'un temps d'exécution compétitif.

L'apprentissage profond repose sur l'entraînement d'un modèle à partir d'une base de données [18, 17], ce qui représente une opportunité pour revaloriser les calculs haute-fidélités issus d'autres projets et conservés par les industriels. Il convient de noter que dans ce contexte, le nombre de données disponibles est souvent limité, et cette contrainte est prise en compte dans la preuve de concept présentée ici. De plus, les modèles d'apprentissage profond sont souvent confrontés à une incertitude quant à la validité de leurs sorties, ce qui entrave leur certification et suscite la réticence des industriels. C'est pourquoi l'idée d'associer ces méthodes à des solveurs déjà éprouvés et utilisés dans l'industrie apparaît comme une option intéressante.

Nous abordons ici la conception des sièges d'avion lors de la simulation d'un crash aérien. Notre attention se porte spécifiquement sur des problèmes non paramétrables, pour lesquels les approches classiques de réduction de modèles [7, 13] présentent certaines limitations. Cela nous permet d'envisager des géométries, voire des topologies, très diverses, entraînant naturellement des problèmes et des maillages de différentes tailles. Nous examinerons et comparerons deux approches distinctes. La première repose sur une méthode couramment utilisée dans la communauté de l'apprentissage profond sur les graphes

[4, 6, 17], qui consiste en une prédiction autorégressive étape par étape du champ d'intérêt. La seconde approche, plus novatrice, propose un modèle basé sur l'apprentissage profond sur graphes qui permet de déterminer a priori les modes spatiaux et les modes temporels de la solution, en vue de reconstruire le champ de solution. Cette dernière approche offre également la possibilité d'être enrichie avec d'autres modes calculés à la volée, en suivant par exemple le principe de la Proper Generalized Decomposition (PGD) [3, 13].

2 Problème traité

2.1 Simulation de crash aérien

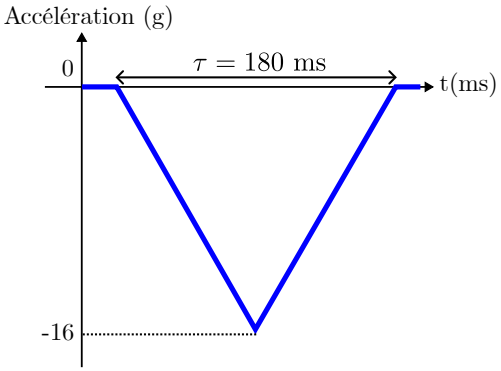

FIGURE 1 – Chargement utilisé par Safran.

L'industrie aéronautique utilise le chargement en accélération représenté dans la Figure 1 pour évaluer la résistance des sièges d'avion. Ce chargement simule la décélération subie par l'avion en cas de collision avec un obstacle. Les critères de dimensionnement sont axés à la fois sur l'intégrité mécanique des composants du siège et sur la sécurité physique du passager. Dans la preuve de concept présentée ici, la modélisation du passager n'est pas prise en compte, et l'analyse se concentre uniquement sur l'intégrité de la structure. De plus, le chargement en accélération de la Figure 1 sera appliqué sous forme d'une force d'inertie d'entraînement. De la même manière, la charge exercée au niveau de l'assise du siège par un passager de 90 kg subissant une accélération du même profil sera modélisée par un chargement équivalent. Respectivement ces chargements sont notés : $\mathbf{\Gamma}_{\text{inertie}}(t)$ et $\boldsymbol{F}_{\text{ext}}(t)$.
L'équation suivante présente le problème traité avec les conditions aux limites et les condition initiales.

$$\begin{cases} \mathbb{M}\ddot{\boldsymbol{U}}(t) + \mathbb{K}U(t) = \mathbb{M}\mathbf{\Gamma}_{\text{inertie}}(t) + \boldsymbol{F}_{\text{ext}}(t) & \boldsymbol{X} \in \Omega, t \in]0,T] \\ \boldsymbol{U}(t) = 0 & \boldsymbol{X} \in \Gamma, t \in]0,T] \\ \boldsymbol{U}(0) = \dot{\boldsymbol{U}}(0) = 0 & \boldsymbol{X} \in \Omega, t = 0 \end{cases} \tag{1}$$

2.2 Base de données

Nous avons créé 400 maillages différents pour des sièges en deux dimensions en utilisant le logiciel libre GMSH [9]. Ensuite, nous avons résolu le problème éléments finis du problème (1) en appliquant un solveur dynamique de type Newmark [16] et en implémentant de manière exacte les conditions aux limites grâce à la méthode de substitution. Suite à une analyse de convergence temporelle, nous avons sélectionné un nombre optimal de pas de temps, fixé à $N_t = 2000$. Comme illustré dans la Figure 2, les maillages de notre base de données présentent diverses géométries et des zones de rigidité variables, ce qui entraîne naturellement des maillages de tailles différentes et des réponses mécaniques variées.

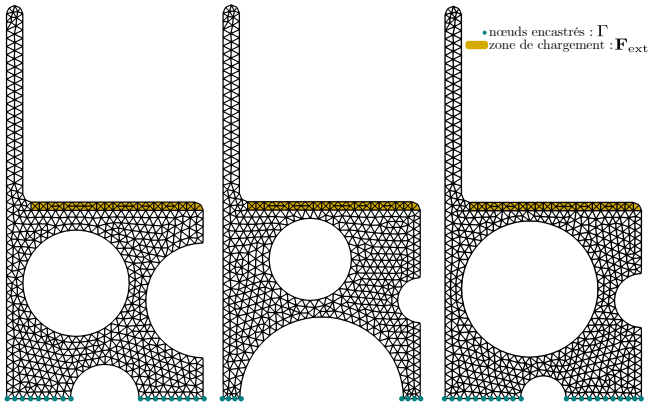

FIGURE 2 – Exemple de trois sièges de la base de données

3 Méthodologie

3.1 L'apprentissage profond sur graphes

Le formalisme des réseaux neuronaux sur graphes (ou Graph Neural Networks - GNN) offre une approche pertinente pour la mise en données des problèmes numériques utilisant des maillages. En effet, un maillage composé d'éléments d'ordre 1, par exemple, peut être vu comme un graphe composé de N nœuds v_i reliés par des arêtes e_{ij}. Ces nœuds et ces arêtes contiennent des caractéristiques, respectivement : $\boldsymbol{v}_i$ et $\boldsymbol{e}_{ij}$. Le graphe contient également les informations de connectivité du maillage sous la forme d'une matrice d'adjacence $\mathbb{A}$, qui n'est qu'une réécriture de la table de connectivité d'un problème d'éléments finis. Cette matrice d'adjacence connue *a priori*, constitue une entrée pour pour les GNN. À titre d'exemple, la matrice d'adjacence du graphe de la Figure 3 est :

$$\mathbb{A} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}_{N \times N}$$

Les Graph Neural Networks sont reconnus comme une généralisation des Réseaux de Convolution [14] (Convolutional Neural Networks - CNN) appliqués aux données structurées sous forme de graphes. Les CNN sont traditionnellement utilisés pour des données structurées sous forme de grilles cartésiennes telles que des images. Cette généralisation permet aux GNN de détecter et de comprendre les structures complexes présentes dans les données en appliquant des opérations de convolution spatiale, ce qui les rend extrêmement efficaces pour des tâches de régression par exemple.

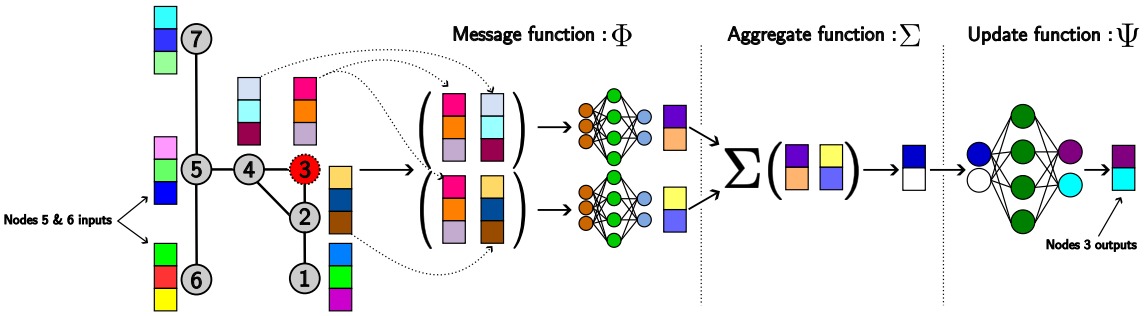

FIGURE 3 – Illustration d'une couche de convolution spatiale sur graphe en trois étapes, les réseaux neuronaux symbolisent des fonctions dont les paramètres sont ajustables par apprentissage (dans l'exemple ci-dessus on l'applique au nœud 3 du graphe possédant $N = 7$ nœuds).

Le principe fondamental des convolutions spatiales dans les GNN est illustré sur la Figure 3, en suivant les conventions bien établies dans la littérature et proposée par [10]. L'idée clé est que l'état d'un nœud $\boldsymbol{v}_i^l$ à la couche l est influencé par les états de ses nœuds voisins $\boldsymbol{v}_j^{l-1} \in \mathcal{N}(\mathbf{v}_i)$ de la couche précédente, et ce processus se propage de couche en couche. Dans cette étude, notre intérêt se limite à la prédiction des états au niveau des nœuds du graphe.

Le processus de convolution, ou transmission du message, s'effectue généralement en trois étapes distinctes :

1. TRANSMISSION DU MESSAGE : Tout d'abord, une fonction Φ de transmission du message $\boldsymbol{m}_{ij}$ est appliquée entre les nœuds v_i et v_j. Les paramètres de cette fonction sont appris par l'entraînement du modèle, et elle prend souvent la forme d'un réseau neuronal multicouche (MLP). L'objectif est de transformer les caractéristiques des nœuds voisins avant de les agréger.
2. AGRÉGATION DES CARACTÉRISTIQUES : Ensuite, les caractéristiques transformées des nœuds voisins sont agrégées à l'aide de la fonction Σ, qui doit être invariante par permutation. Cette invariance par permutation garantit que l'ordre et le nombre de nœuds voisins n'a pas d'impact sur la dimenssion du résultat de l'agrégation. Cela permet aux GNN de s'adapter à des graphes de tailles variables.

3. MISE À JOUR DES NŒUDS : Enfin, les caractéristiques agrégées sont utilisées par une fonction de mise à jour Ψ, dont les paramètres sont également ajustables par l'apprentissage. Cette fonction transforme l'information agrégée en une sortie au niveau du nœud, qui peut ensuite être associée à d'autres quantités si nécessaire.

En résumé, les GNN tirent leur puissance de la généralisation des convolutions spatiales aux graphes, grâce à des fonctions d'agrégation invariables par permutation. Cette flexibilité leur permet de traiter des graphes de tailles variables, ce qui en fait un outil puissant pour des applications de régression sur des données structurées sous forme de graphes, et donc potentiellement sur des problèmes physiques nécessitant un maillage.

Dans les approches qui ont été développées dans cette étude, nous cherchons à déterminer des quantités physiques au niveau des nœuds, telles que le déplacement $\boldsymbol{U}$ à partir d'autres caractéristiques physiques connues *a priori* : les coordonnées des nœuds $\boldsymbol{x}_i$, la nature des nœuds $\boldsymbol{n}_i$ (encastré ou libre) ou encore la force nodale généralisée au pas de temps k : $\boldsymbol{f}_i^k$. La solution en déplacement issue de l'approche par GNN sera désignée dans la suite par : $\boldsymbol{U}^{GNN}$.

3.2 Axe réduction de modèle

Nous avons opté pour l'utilisation d'un modèle réduit afin de garantir la validité physique de la solution $\boldsymbol{U}^{GNN}$ obtenue par l'apprentissage profond. L'idée sous-jacente consiste à considérer $\boldsymbol{U}^{GNN}$ comme une initialisation pour un solveur basé sur un modèle réduit. Dans ce cadre, nous évaluons le résidu du problème 1 en utilisant $\boldsymbol{U}^{GNN}$, et si ce résidu dépasse un seuil défini à $\eta_{\text{résidu}} = 10^{-3}\,\%$, nous complétons la solution obtenue par apprentissage profond avec un terme correctif $\boldsymbol{W}$.

L'approche choisie pour obtenir ce terme correctif est celle de la Proper Generalized Decomposition (PGD) temps-espace [13]. Nous avons démontré la pertinence de cette approche pour résoudre le problème décrit dans l'équation (1), en particulier pour la valeur du temps caractéristique du signal présenté dans la Figure 1. En effet, dans le cas d'un temps caractéristique très petit, on parle de dynamique rapide, on peut montrer que cette approche n'était pas adaptée [3].

Ainsi, la recherche du terme correctif $\boldsymbol{W}$ s'effectue sous la forme d'un produit de fonctions à variables séparées dans le temps et l'espace :

$$\boldsymbol{W} = \lambda \boldsymbol{\Lambda}$$

La solution recherchée est alors définie comme suit :

$$\boldsymbol{U} = \boldsymbol{U}^{\text{GNN}} + \boldsymbol{W}$$

En l'injectant dans l'équation (1), nous obtenons le nouveau second membre présenté dans :

$$\boldsymbol{F} = \mathbb{K}\boldsymbol{U}^{\text{GNN}}(t) + \mathbb{M}\ddot{\boldsymbol{U}}^{\text{GNN}}(t) - \mathbb{M}\boldsymbol{\Gamma}_{\text{inertie}}(t) - \boldsymbol{F}_{\text{ext}}(t) \tag{2}$$

Pour résoudre ce problème, nous adoptons une formulation de Galerkin temporelle de l'équation (1). Selon le respect du critère de résidu $\eta_{\text{résidu}}$, nous pouvons, si nécessaire, ajouter de nouveaux modes de manière gloutonne.

3.3 Deux approches d'apprentissage investiguées

3.3.1 Première approche : l'approche auto-regressive GNN-AR

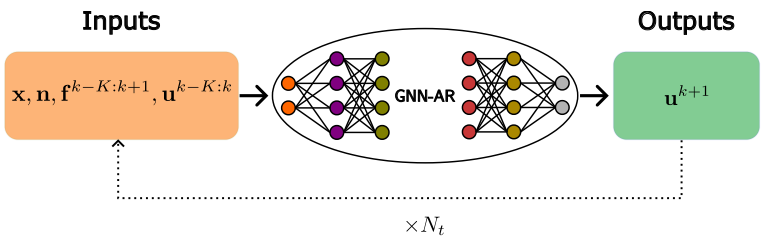

FIGURE 4 – Approche auto-régressive par GNN-AR

L'approche couramment utilisée, souvent présentée dans la littérature comme dans [6], et originellement proposée dans [17] est illustrée dans la Figure 4. L'idée principale consiste à prédire l'état des nœuds du graphe étape par étape à l'aide du même réseau GNN-AR. Ce réseau a pour

tâche d'apprendre la relation déterministe entre les déplacements précédents et le déplacement courant. Cette approche nécessite donc d'utiliser le réseau autant de fois qu'il y a de pas de temps à prédire, soit N_t fois.

Classiquement, comme proposé dans la littérature [2, 4, 15], une étape d'encodage de l'information physique est réalisée à l'aide d'un MLP, transformant les données physiques vers un espace latent de grande dimension :

$$\text{état latent du nœud : } i \quad \boldsymbol{v}_i^0 = \text{MLP}_{\text{encoder}}\left(\mathbf{x}_i, \mathbf{n}_i, \mathbf{f}_i^{k-K:k+1}, \mathbf{u}_i^{k-K:k}\right),$$

À la fin du processus de convolution, une étape symétrique de décodage est appliquée pour revenir de l'espace latent à l'espace physique des déplacements :

$$\text{état final du nœud : } i \quad \boldsymbol{y}_{\boldsymbol{i}}^{GNN} = \text{MLP}_{\text{decoder}}\left(\boldsymbol{v}_i^L\right)$$

Ces deux fonctions $\text{MLP}_{\text{encoder}}$ et $\text{MLP}_{\text{decoder}}$ sont paramétrées et ajustables par l'apprentissage.

Entre ces deux étapes, on applique le processus de convolution avec les couches suivantes :

$$\begin{aligned} \text{message sur l'arêtes } j \to i \quad & \boldsymbol{m}_{ij}^l = \Phi\left(\boldsymbol{v}_i^l, \boldsymbol{v}_j^l, \boldsymbol{u}_i^{k-K:k} - \boldsymbol{u}_j^{k-K:k}, \mathbf{f}_i^{k-K:k+1}, \mathbf{f}_j^{k-K:k+1}, \boldsymbol{n}_i, \boldsymbol{n}_j\right), \\ \text{aggrégation et mise à jour du nœud } i : \quad & \boldsymbol{v}_i^{l+1} = \Psi\left(\boldsymbol{v}_i^l, \textstyle\sum_{j\in\mathcal{N}(i)} \boldsymbol{m}_{ij}^l, \mathbf{f}_i^{k-K:k+1}, \boldsymbol{n}_i\right) \end{aligned} \tag{3}$$

L'architecture de ces couches a été initialement proposée dans [1]. Dans cette configuration, le paramètre K représente le nombre de pas de temps antérieurs utilisés pour prédire le pas de temps courant, tandis que L indique le nombre de couches de convolution appliquées. Ces deux grandeurs doivent être optimisées pour obtenir les meilleures performances d'apprentissage, on parle alors d'hyperparamètres.

Une caractéristique notable des couches que nous avons développées est la réintroduction de termes physiques à chaque couche, une approche similaire à celle utilisée dans DenseNet [11]. Ces connexions, couramment désignées sous le nom de *skip connections*, se sont avérées particulièrement efficaces pour résoudre ce type de problème, comme cela a été démontré dans d'autres travaux antérieurs [4].

3.3.2 Deuxième approche : approche par variable séparée

L'approche originale que nous présentons ici repose sur la séparabilité temporelle et spatiale du champ physique recherché : $\boldsymbol{U}$. En effet, il est observé que les solutions contenues dans la base de données sont fortement séparables en termes de temps et d'espace, une caractéristique que l'on retrouve fréquemment dans divers champs physiques, même dans des contextes non linéaires [7].

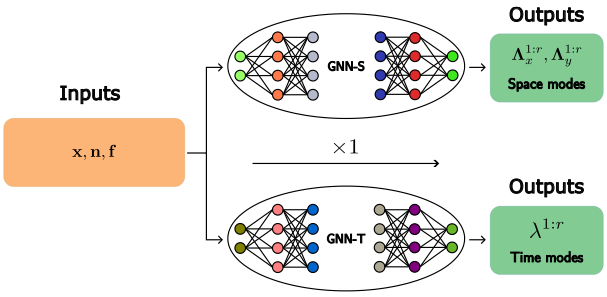

FIGURE 5 – Nouvelle approche proposée par séparation de variables.

Notre proposition consiste à déterminer, par apprentissage, les modes temporels et spatiaux de la Proper Generalized Decomposition (PGD) des solutions, plutôt que d'apprendre directement les solutions complètes. Cette approche présente plusieurs avantages par rapport à la méthode précédente. Tout d'abord, elle réduit la quantité d'information à traiter, car les modes PGD constituent également une forme compressée de la solution. Cela se traduit par une phase d'apprentissage (*offline*) beaucoup plus rapide par rapport à l'approche précédente. De plus, comme en témoigne la Figure 5, qui résume cette approche, le temps d'inférence du modèle (c'est-à-dire le temps *online*) devient nettement plus compétitif pour un nombre de paramètres constant, car en une seule étape, nous obtenons le champ solution complet.

Nous nous sommes concentrés sur l'apprentissage des trois premiers modes temporels et spatiaux ($r = 3$), ce qui permet de reconstruire le champ de solution avec une erreur relative $L2$ inférieure à $0.1\,\%$. Pour ce faire, deux réseaux ont été entraînés simultanément : GNN-S pour les modes Spatiaux, qui suit une architecture similaire à celle de la littérature présentée précédemment, et GNN-T, pour les modes Temporels, qui a nécessité un développement spécifique.

3.3.3 Apprentissage

L'un des éléments cruciaux pour le succès des modèles d'apprentissage profond réside dans le choix de la fonction de perte $\mathcal{L}$ que l'on cherche à minimiser. La fonction de perte la plus couramment utilisée et efficace pour les tâches de régression est l'erreur quadratique moyenne, que nous désignons par MSE($\bullet$). Dans la suite, nous présentons les termes que nous avons développés pour construire la fonction de perte $\mathcal{L}_{\text{GNN-S}}$ du réseau GNN-S. Les quantités $\bullet^{\text{GNN}}$ représentent les sorties du GNN-S, tandis que les quantités $\bullet^{\text{BDD}}$ désignent les sorties cibles provenant de la Base De Données.

Le premier terme est construit à partir de l'erreur quadratique moyenne des modes spatiaux :

$$\mathcal{L}_{\text{modes}} = \sum_{p=1}^{r} \eta_p \, \text{MSE}\left(\mathbf{\Lambda}_p^{\text{GNN}} - \mathbf{\Lambda}_p^{\text{BDD}}\right). \tag{4}$$

Nous introduisons les hyperparamètres $\{\eta_p\}_{p=\{1,2,3\}}$ pour pondérer l'importance des modes en fonction de leur énergie respective.
Le deuxième terme

$$\mathcal{L}_{\text{champ}} = \text{MSE}\left(\sum_{p=1}^{r} \mathbf{\Lambda}_p^{\text{GNN}}.\lambda_p^{\text{BDD}} - \boldsymbol{U}^{\text{BDD}}\right),$$

garantit la cohérence du champ reconstruit avec le champ cible. Il permet de corriger les erreurs au sein des modes spatiaux.
La fonction perte :

$$\mathcal{L}_{\text{énergie}} = \sum_{p=1}^{r} \delta_p \sqrt{(\mathbf{\Lambda}_p^{\text{GNN}} - \mathbf{\Lambda}_p^{\text{BDD}})^\top \mathbb{K} (\mathbf{\Lambda}_p^{\text{GNN}} - \mathbf{\Lambda}_p^{\text{BDD}})},$$

correspond à l'erreur énergétique du mode de déplacement. Cette erreur est plus exigeante que les précédentes en raison des propriétés de la matrice $\mathbb{K}$, mais elle aboutit à de meilleurs résultats lorsqu'elle est combinée à l'erreur de l'équation (4). Nous introduisons également les hyperparamètres $\{\delta_p\}_{p=\{1,2,3\}}$ pour pondérer l'importance des modes en fonction de leur niveau d'énergie.
La fonction de perte totale :

$$\mathcal{L}_{\text{GNN-S}} = \mathcal{L}_{\text{énergie}} + \mathcal{L}_{\text{modes}} + \kappa \mathcal{L}_{\text{champ}}$$

est une somme pondérée par des hyperparamètres des différentes fonctions de perte précédentes.

Nous minimisons $\mathcal{L}_{\text{GNN-S}}$ par apprentissage en utilisant l'algorithme du gradient stochastique Adam [12]. De plus, il est nécessaire d'optimiser les différents hyperparamètres présentés ci-dessus. Pour cela, nous utilisons une stratégie de plans d'expérience basée sur une méthode de Latin Hypercube générée à l'aide du logiciel libre Lagun [5] développé par Safran Tech.

4 Résultats

Les deux approches illustrées dans les Figures 4 et 5 ont été mises en œuvre et optimisées par apprentissage. Les données de la base de données ont été réparties conformément à la pratique habituelle : 80% (320 sièges) pour l'ensemble d'entraînement, 10% (40 sièges) pour l'ensemble de validation, dans le but de prévenir le sur-ajustement (ou *overfitting*), et enfin 10% (40 sièges) pour l'ensemble de test. Les résultats présentés ci-dessous sont issus de l'évaluation effectuée sur les données de l'ensemble de test, c'est-à-dire des données totalement inconnues du réseau.

4.1 Évolution de l'erreur

On note un autre avantage de la méthode développée ici sur la Figure 6. En effet, contrairement à l'approche auto-régressive habituellement utilisée dans la littérature, notre méthode évite l'explosion de l'erreur d'inférence (ou *rollout error explosion*) [17, 4]. Notre méthode permet d'obtenir une erreur presque constante sur l'ensemble des pas de temps, sans phénomène d'aggravation, et ici, inférieure à celle de l'approche classique.

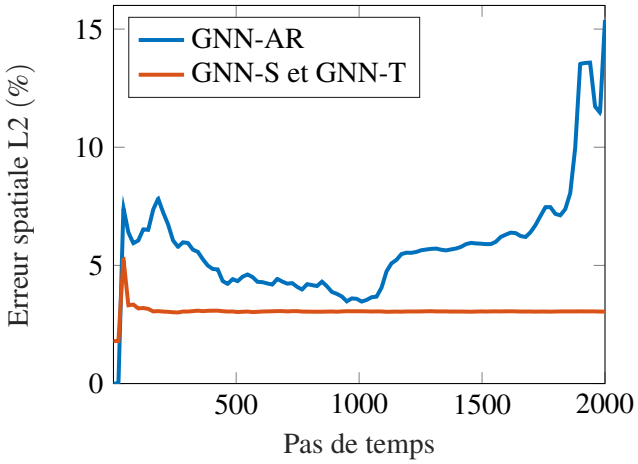

FIGURE 6 – Évolution de l'erreur spatiale L2 au cours des pas de temps pour les deux approches présentées (pour l'approche GNN-AR les deux premiers pas de temps sont connus du réseau afin de l'initialiser ($K = 2$), ce qui explique l'erreur observée).

4.2 Correction par PGD

Le terme correctif basé sur la méthode PGD, présenté dans la partie 3.2, a été développé, et nous avons constaté une différence entre les deux approches que nous avons présentées précédemment. En effet, la solution obtenue à partir de la première approche, bien que présentant une erreur globale très satisfaisante, présente un niveau de bruit local élevé. De plus, nous avons remarqué que ce champ n'était pratiquement pas séparable selon les variables temps-espace, contrairement à la solution physique. Cette faible séparabilité a des répercussions sur la construction du second membre $\boldsymbol{F}$ de l'équation (2), qui devient donc également peu séparable. Par conséquent, la résolution finale du problème avec la méthode PGD s'est avérée peu pertinente à la suite de cette première approche.

En revanche, l'approche que nous avons développée génère une solution séparable par construction. Dans ce scénario, nous avons observé qu'une correction à l'aide du terme PGD pouvait être appliquée, ce qui a entraîné un gain de temps significatif par rapport à une approche classique.

5 Conclusions et perspectives

Cet article présente une preuve de concept de l'utilisation d'un modèle d'apprentissage profond sur graphe couplé avec un modèle réduit basé sur la PGD.

Nous avons introduit une nouvelle approche, basée sur la séparabilité temps-espace du champ de solution, qui a montré des avantages significatifs par rapport aux approches présentes dans la littérature, à la fois pour la phase hors ligne (*offline*) et pour la phase en ligne (*online*) du modèle. Cette approche reste à développer et à améliorer dans le futur, en particulier en ce qui concerne son couplage avec le modèle réduit, qui reste encore délicat. Le cadre proposé permet de garantir que, *in fine*, l'équation physique soit effectivement résolue tout en garantissant un gain de temps dans sa résolution.

À l'avenir, il serait intéressant de s'affranchir les limitations imposées par les couches de convolution actuelles en s'inspirant de l'architecture U-net, largement utilisée dans la littérature sur les CNN, et de l'adapter au formalisme des graphes, comme proposé dans [8].

Enfin, bien que cette preuve de concept soit encourageante, elle devra être évaluée sur des problèmes se rapprochant davantage des problèmes industriels fortement non linéaires. Dans ce contexte, il sera nécessaire d'envisager un solveur adapté, basé sur un modèle réduit, afin de conserver les bénéfices liés au respect des équations physiques.

Remerciements

Ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS au travers de l'allocation de ressources 2023-[AD011014260] attribuée par GENCI.
Ces travaux ont également bénéficié des moyens de calcul du Mésocentre de CentraleSupélec, de l'École normale supérieure Paris-Saclay et de l'Université Paris-Saclay avec le soutien du CNRS et de la région Île-de-France (https ://mesocentre.universite-paris-saclay.fr/).

Références

[1] Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., et al. *Relational Inductive Biases, Deep Learning, and Graph Networks*. arXiv, 17 octobre 2018. http ://arxiv.org/abs/1806.01261.

[2] Bishnoi, S., Bhattoo, R., Ranu, S., et Krishnan, N. M. A. *Enhancing the Inductive Biases of Graph Neural ODE for Modeling Dynamical Systems*. arXiv, 21 septembre 2022. http ://arxiv.org/abs/2209.10740.

[3] Boucinha, L., Gravouil, A., et Ammar, A. *Space–Time Proper Generalized Decompositions for the Resolution of Transient Elastodynamic Models*. Computer Methods in Applied Mechanics and Engineering, 255 (1 mars 2013), 67-88. https ://doi.org/10.1016/j.cma.2012.11.003.

[4] Brandstetter, J., Worrall, D., et Welling, M. *Message Passing Neural PDE Solvers*. arXiv, 26 mars 2022. http ://arxiv.org/abs/2202.03376.

[5] Da Veiga, S., Bénard, C., Chazalviel, D., El Bachiri, A., Sinoquet, D., Gonon, T., Richet, Y., Baker, A. (2023). lagun. GitLab. https ://gitlab.com/drti/lagun

[6] Dalton, D., Gao, H., et Husmeier, D. *Emulation of Cardiac Mechanics Using Graph Neural Networks*. Computer Methods in Applied Mechanics and Engineering, 401 (1 novembre 2022), 115645. https ://doi.org/10.1016/j.cma.2022.115645.

[7] Daby-Seesaram, A., Fau, A., Charbonnel, P.-É., et Néron, D. *A Hybrid Frequency-Temporal Reduced-Order Method for Nonlinear Dynamics*. Nonlinear Dynamics, 111, n 15 (1 août 2023), 13669-89. https ://doi.org/10.1007/s11071-023-08513-8.

[8] Gao, H., et Ji, S. *Graph U-Nets*. arXiv, 11 mai 2019. http ://arxiv.org/abs/1905.05178.

[9] Geuzaine C. et Remacle. J.-F. *Gmsh : a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities.* International Journal for Numerical Methods in Engineering 79(11), pp. 1309-1331, 2009.

[10] Hamilton, W. L. (2020). *Graph Representation Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning, 14(3), 1-159.

[11] Huang, G., Liu, Z., van der Maaten, L., et Weinberger, K. Q. *Densely Connected Convolutional Networks*. arXiv, 28 janvier 2018. http ://arxiv.org/abs/1608.06993.

[12] Kingma, D. P., et Ba, J. « *Adam : A Method for Stochastic Optimization* ». arXiv, 29 janvier 2017. https ://doi.org/10.48550/arXiv.1412.6980.

[13] Ladevèze, P. *Nonlinear Computational Structural Mechanics : New Approaches and Non-Incremental Methods of Calculation*. Mechanical Engineering Series. New York, NY : Springer, 1999. https ://doi.org/10.1007/978-1-4612-1432-8.

[14] LeCun, Y., Kavukcuoglu, K., et Farabet, C. « *Convolutional Networks and Applications in Vision* ». Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 253-56. Paris, France : IEEE, 2010. https ://doi.org/10.1109/ISCAS.2010.5537907.

[15] Nastorg, M., Bucci, M.-A., Faney, T., Gratien, J.-M., Charpiat, G., et Schoenauer, M. *An Implicit GNN Solver for Poisson-like Problems*. arXiv, 23 février 2023. http ://arxiv.org/abs/2302.10891.

[16] Newmark, N. M., *A Method of Computation for Structural Dynamics*, American Society of Civil Engineers, 1959

[17] Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., et Battaglia, P. W. *Learning to Simulate Complex Physics with Graph Networks*. arXiv, 14 septembre 2020. https ://doi.org/10.48550/arXiv.2002.09405.

[18] You, J., Ying, R., et Leskovec, J. *Design Space for Graph Neural Networks*. arXiv, 23 juillet 2021. https ://doi.org/10.48550/arXiv.2011.08843.