LECOURTIER Frédérique 9 janvier 2024

Results : Week 1 - Week 15

Table des matières

1 Week 4 : 23/10/2023 - 27/10/2023

ATTENTION : problème de projection - carte d'erreur FEM

1.1 Reproduction résultats - Article 2104.08426

Dans un premier temps, j'ai cherché à implémenter en Python les deux méthodes présentées dans l'article 2104.08426, permettant de calculer une ADF ("Approach Distance Function"). On retrouve le calcul d'une ADF sur un polygone qui représente le bord d'une ellipse par les deux méthodes suivantes : la méthode REQ (R-equivalence) dans la Figure ?? et la méthode MVP ("Mean Value Potential") dans la Figure ??.

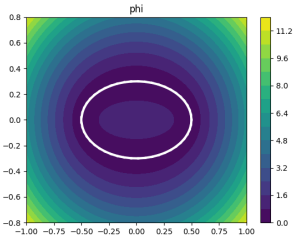

FIGURE 1 – ADF par REQ sur une ellipse.

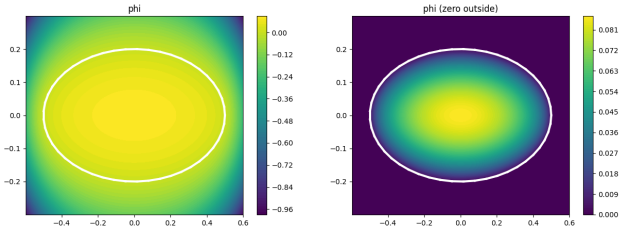

FIGURE 2 – ADF par MVP sur une ellipse.

1.2 Entraînement du PINNs à apprendre une solution unique

1.2.1 Problème considéré

EDP : On considère le problème de Poisson avec condition de Dirichlet homogène ($g = 0$), définie par
Trouver $u : \Omega \to \mathbb{R}^d (d = 1, 2, 3)$ tel que

$$\begin{cases} -\Delta u = f, & \text{dans } \Omega, \\ u = g, & \text{sur } \partial\Omega, \end{cases} \tag{$\mathcal{P}$}$$

avec Δ l'opérateur de Laplace.
Géométrie : On considère Ω comme étant un cercle de rayon r et de centre (x_0, y_0).
Pour simplifier, on va considérer que Ω est entièrement contenu dans un carré $\mathcal{O}$ (Figure ??).

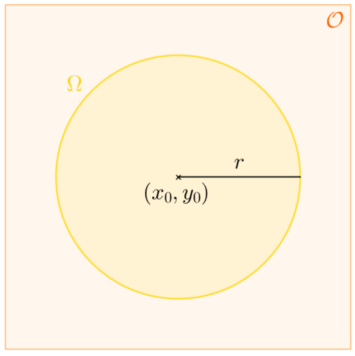

FIGURE 3 – Domaine considéré.

On considère la solution analytique u_{ex}, définie par

$$u_{ex}(x, y) = S \times \sin\left(\frac{1}{r^2}\pi((x - x_0)^2 + (y - y_0)^2)\right)$$

Ce qui nous fournit le terme source f, définie par

$$f(x, y) = \frac{4}{r^4}\pi^2 S((x - x_0)^2 + (y - y_0)^2)\sin\left(\frac{1}{r^2}\pi((x - x_0)^2 + (y - y_0)^2)\right) - \frac{4}{r^2}\pi S\cos\left(\frac{1}{r^2}\pi((x - x_0)^2 + (y - y_0)^2)\right)$$

Remarque. *On voit que sur le cercle, le problème est bien homogène.*
De plus, on notera qu'un choix simple peut être de prendre le carré $[x_0 - r - \epsilon, x_0 + r + \epsilon] \times [y_0 - r - \epsilon, y_0 + r + \epsilon]$ *où* $\epsilon > 0$ *est un paramètre fixé dans le but que* Ω *soit entièrement compris dans* $\mathcal{O}$.

1.2.2 Entraînement du PINNs

On fixe $r = \sqrt{2}/4$, $(x_0, y_0) = (0.5, 0.5)$ et $S = 0.5$ et on considère ici que l'on souhaite entraîner un PINNs à apprendre cette solution. On utilisera l'implémentation développé dans le module ScimBA [1].

1. https://sciml.gitlabpages.inria.fr/scimba/

On notera que l'article 2104.08426 présente comment imposer les conditions au bord de manière exacte. C'est pourquoi, on considérera deux cas :

- on apprend directement la solution u. La loss totale regroupe alors la loss sur le résidu et la loss sur le bord.
- on apprend w tel que $u = \phi w$ avec ϕ notre fonction levelset. La loss ne contient alors que la loss sur le résidu.

Ainsi, une première étape a été de rajouter, dans l'implémentation de ScimBa, la possibilité de définir un domaine à partir d'une fonction levelset. Ainsi pour obtenir un sampling de points à l'intérieur de Ω, il suffit de générer un échantillon de point sur le carré $\mathcal{O}$ et de ne garder que les points tels que $\phi(x, y) < 0$. Pour générer un échantillon de point sur le bord du domaine Ω, on a fait le choix pour l'instant de prendre les points tels que $|\phi(x, y)| < \epsilon$ avec $\epsilon = 1e - 5$ (Figure **??**).

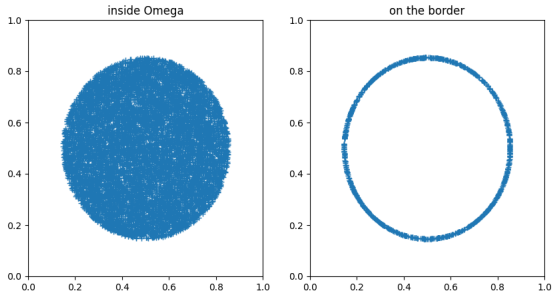

FIGURE 4 – Sampling à l'intérieur et au bord du cercle considéré.

Remarque. *Pour le sampling du bord, c'est un choix qui ne sera pas conserver, il faudra trouver une solution plus rapide-précise que celle-ci.*

On peut alors entraîner le PINNs à apprendre notre solution. On choisira la configuration suivante :

Configuration	Model parameters		Trainer parameters					Training parameters			
	Layers	Activation Function	Learning rate	Decay	w_data	w_res	w_bc	n_epochs	n_collocation	n_bc_collocation	n_data
0	[20, 20, 20, 20]	sine	0.01	0.99	0.0	0.01	10.0	1000	500	500	0

FIGURE 5 – Paramètres d'entraînement considéré pour le PINNs.

On va alors entraîner un modèle à apprendre u (Figure **??**) et un autre à apprendre w (Figure **??**) avec ces mêmes paramètres dans le but de comparer les résultats.

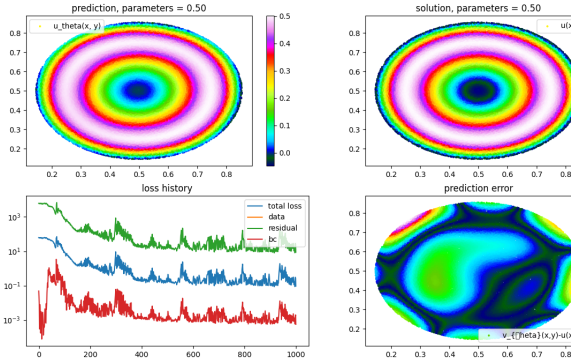

FIGURE 6 – Fin d'entraînement - Modèle sur u.

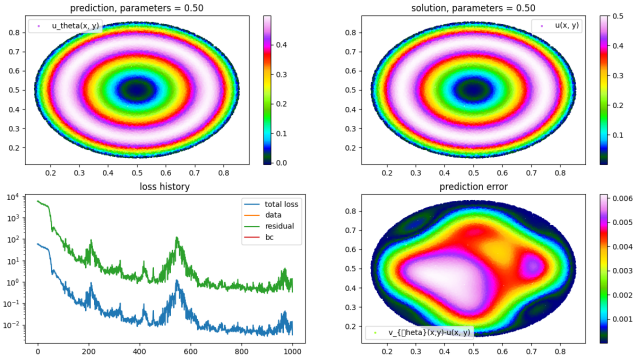

FIGURE 7 – Fin d'entraînement - Modèle sur w.

Remarque. *Il semblerait que le modèle sur w soit 10 fois plus précis en terme d'erreur.*
De plus, on remarque bien, sur la carte d'erreur, sur le modèle sur u a des erreurs au bord, ce qui risque de poser problème dans la correction.

1.2.3 Correction sur les prédictions du PINNs

On note $\tilde{\phi}$ la prédiction d'un des PINNs précédent. On ne va considérer ici que la correction par addition, on pose alors

$$\tilde{u} = \tilde{\phi} + \tilde{C}$$

et on cherche à trouver $\tilde{C} : \Omega \to \mathbb{R}^d$ solution du problème

$$\begin{cases} -\Delta \tilde{u} = f, & \text{on } \Omega, \\ \tilde{u} = g, & \text{in } \Gamma. \end{cases}$$

avec $g = 0$ dans le cas considéré. On cherche alors à trouver $\tilde{C} : \Omega \to \mathbb{R}^d$ solution du problème

$$\begin{cases} -\Delta \tilde{C} = \tilde{f}, & \text{on } \Omega, \\ \tilde{C} = 0, & \text{in } \Gamma. \end{cases}$$

avec $\tilde{f} = f + \Delta \tilde{\phi}$.
On cherche alors à tester la correction sur les deux modèles précédents (celui où on apprend u et celui où on apprend w). On testera l'utilisation de FEM et de ϕ-FEM dans les deux cas.
Résultats avec le modèle sur u :

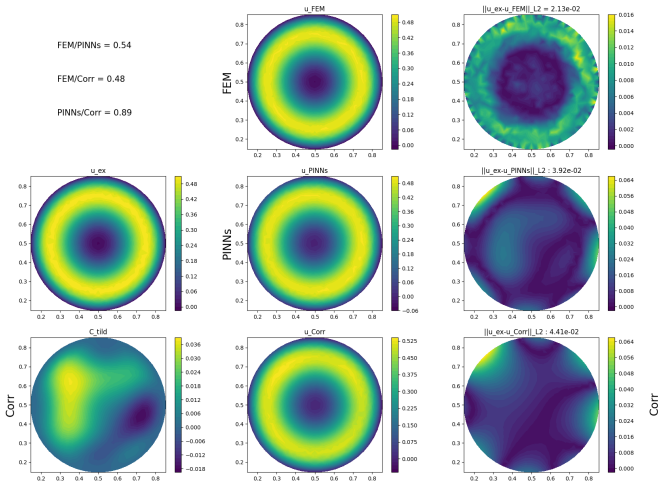

FIGURE 8 – Correction avec FEM - Modèle sur u.

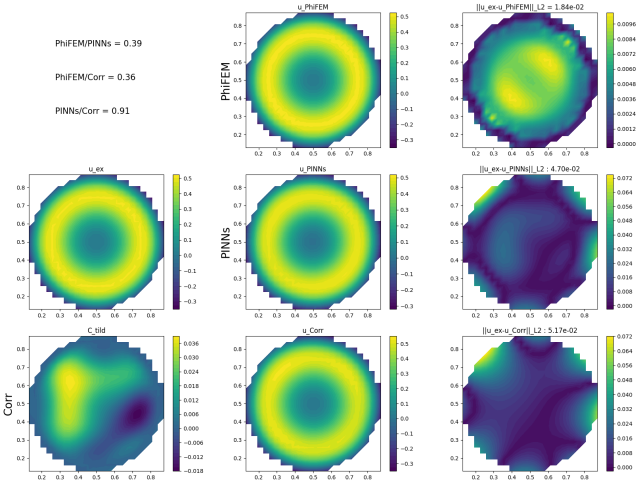

FIGURE 9 – Correction avec ϕ-FEM - Modèle sur u.

Résultats avec le modèle sur w :

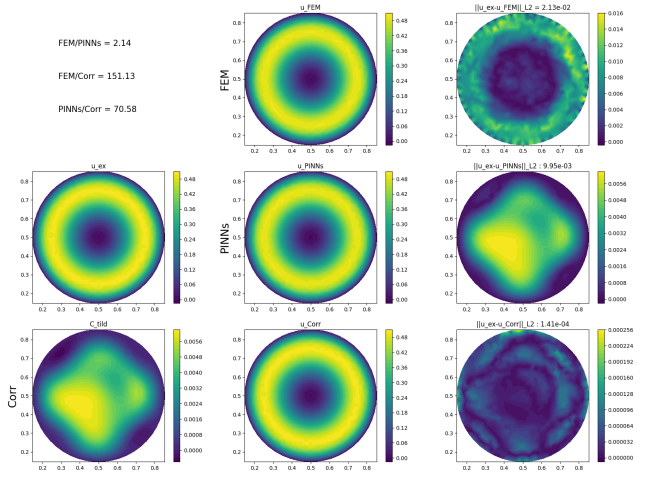

FIGURE 10 – Correction avec FEM - Modèle sur w.

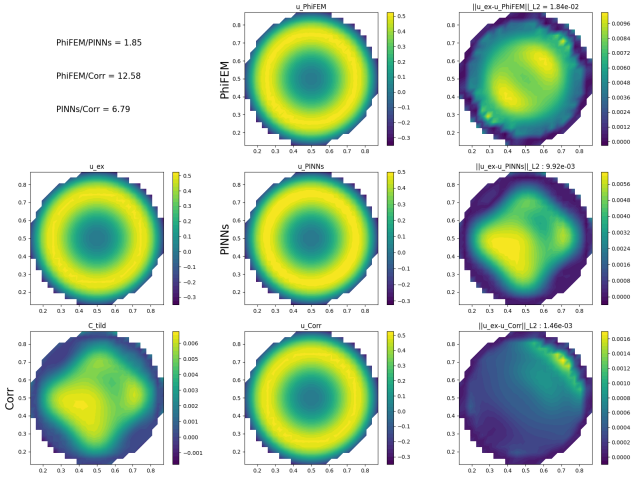

FIGURE 11 – Correction avec ϕ-FEM - Modèle sur w.