LECOURTIER Frédérique 13 juillet 2023

Suivi Stage PhiFEM : 06/02/2023 - 28/07/2023

Table des matières

1 Semaine 1 : 06/02/2023 - 10/02/2023

Résumé

Pendant cette première semaine, j'ai du me familiariser avec le code "phifem" écrit par Vincent Vigon et les codes de génération des données fournit par Killian. L'idée étant de comprendre comment générer les données avec FEniCS pour ensuite les faire apprendre par un FNO. Après la réunion du 07/02, il semblerait que le sujet du stage porte sur l'entrainement d'un FNO puis la correction/certification des prédictions.

1.1 Génération des données

Le code fournit par Killian ("Data_Generation_moving_ellipse_poisson") a pour but de faire varier la levelset. J'ai donc repris ce code afin de générer dans un premier temps les données solution d'un problème de Poisson avec condition de Dirichlet homogène ("DataGen_PhiFEM_f_gaussienne").

On considère Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x, y) = -1/8 + (x - 1/2)^2 + (y - 1/2)^2$ et le domaine fictif $O = (0, 1)^2$.

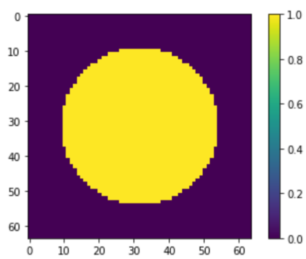
On souhaite résoudre

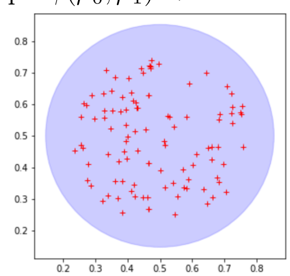
$$\begin{cases} -\Delta u & = f\,, \quad \text{dans } \Omega\,, \\ u & = 0\,, \quad \text{sur } \Gamma\,, \end{cases}$$

où

$$f(x, y) = \exp\left(-\frac{(x-\mu_0)^2 + (y-\mu_1)^2}{2\sigma^2}\right)\,,$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5 - \sqrt{2}/4, 0.5 + \sqrt{2}/4])$ à condition que $\phi(\mu_0, \mu_1) < -0.05$.





La fonction *create_ data* renvoie le nombre donné de F et les paramètres associés choisis uniformément.

Formulation faible :

$$\int_{\Omega_h} \nabla(\bar{\phi} w)\nabla(\bar{\phi} v) - \int_{\partial\Omega_h} \frac{\partial}{\partial n}(\bar{\phi} w)\bar{\phi} v + G_h(w, v) = \int_{\Omega_h} f\bar{\phi} v + G_h^{rhs}(v)$$

avec

$$G_h(w, v) = \sigma h \sum_{E \in \mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial}{\partial n}(\bar{\phi} w)\right]\left[\frac{\partial}{\partial n}(\bar{\phi} v)\right] + \sigma h^2 \sum_{T \in \mathcal{T}_h^\Gamma} \int_T \Delta(\bar{\phi} w)\Delta(\bar{\phi} v)$$

et

$$G_h^{rhs}(v) = -\sigma h^2 \sum_{T \in \mathcal{T}_h^\Gamma} \int_T f \Delta(\bar{\phi} v)$$

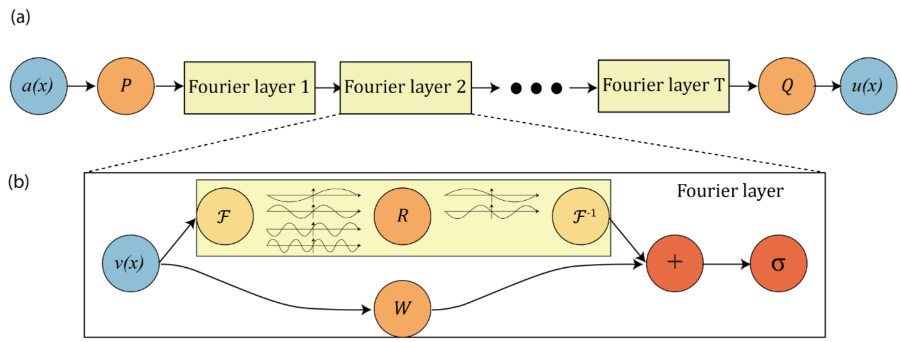
On utilise FEniCS (solveur d'EDP) pour résoudre le problème.

On va finalement stocker les résultats au format npy dans les fichiers "F.npy", "agentParams.npy" et "U.npy"

1.2 FNO

De la même manière que pour la génération des données, il a fallu reprendre le code de Vincent Vigon ("phifem") afin de l'adapter au problème considéré (phifem_f_gaussienne).

Une première étape fut donc la lecture d'article sur les FNO (Fourier Neural Operator). Voici un schéma descriptif de ce type de réseau de neurones.



(a) The full architecture of neural operator: start from input a. 1. Lift to a higher dimension channel space by a neural network P. 2. Apply four layers of integral operators and activation functions. 3. Project back to the target dimension by a neural network Q. Output u. **(b) Fourier layers**: Start from input v. On top: apply the Fourier transform $\mathcal{F}$; a linear transform R on the lower Fourier modes and filters out the higher modes; then apply the inverse Fourier transform $\mathcal{F}^{-1}$. On the bottom: apply a local linear transform W.

L'idée étant que le réseau nous retourne u.

Remarque. *ERREUR : il doit nous retourner w car on connait déjà la levelset et pour la correction, ça n'a pas de sens de faire $\phi u = \phi^2 w$.*

1.3 Correction

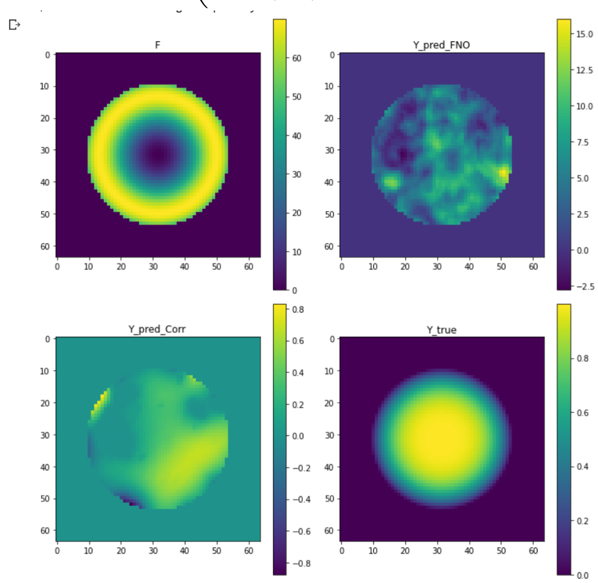
On veut en fait résoudre le même problème sauf que cette fois-ci, on définit une nouvelle levelset $\bar{\phi} = \phi u$ (où u est la sortie du FNO).

On résout alors le nouveau problème $z = \bar{\phi} C$:

$$\begin{cases} -\Delta z & = f\,, \quad \text{dans } \Omega\,, \\ z & = 0\,, \quad \text{sur } \Gamma\,, \end{cases}$$

Remarque. *L'idée étant d'appliquer la correction sur un millage plus grossier que le réseau. Ainsi FNO+Corr plus précis que Phifem classique et plus rapide.*

Les résultats obtenus (en prenant ici $u_{ex} = cos\left(\frac{\pi}{2} \times \left(\frac{4}{\sqrt{2}}\right)^2 \times \left((x-0.5)^2 + (y-0.5)^2\right)\right)$) ne sont pas bons :



Remarque. *ERREUR : C'est logique ! ! ! Le réseau n'a pas été entrainé pour ça ! (+ pb remarque d'avant.)*

2 Semaine 2 : 13/02/2023 - 17/02/2023

Résumé

(Les résultats en semaine 1 ont été obtenus en début de semaine.)

Après la semaine dernière, on s'est dit qu'une bonne idée serait de prendre une solution manufacturée (analytique) afin de pouvoir comparer les erreurs avec FEM classique, PhiFEM, le FNO et le FNO+correction. On a choisi de prendre u comme une gaussienne et ainsi le f associé. Attention, il a fallut normaliser les F pour le FNO. On a également eut l'idée d'utiliser une solution sur-raffinée (à la place d'une solution exacte) mais ça n'a pas encore été testé.

2.1 Génération de données

On considère toujours Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x,y) = -1/8+(x-1/2)^2+(y-1/2)^2$ et le domaine fictif $O = (0,1)^2$ ("DataGen_PhiFEM_u_gaussienne").

On souhaite résoudre

$$\begin{cases} -\Delta u & = f\,, \quad \text{dans } \Omega\,, \\ u & = g\,, \quad \text{sur } \Gamma\,, \end{cases}$$

Notre solution analytique est

$$u_{ex}(x,y) = \exp\left(-\frac{(x-\mu_0)^2+(y-\mu_1)^2}{2\sigma^2}\right),$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([-0.9, 0.9])$.

Ainsi

$$f(x,y) = \left(\frac{2\sigma^2-(x-\mu_0)^2-(y-\mu_1)^2}{\sigma^4}\right) * \exp\left(-\frac{(x-\mu_0)^2+(y-\mu_1)^2}{2\sigma^2}\right)$$

et

$$g(x,y) = u_{ex}(x,y)$$

Pour l'apprentissage du FNO, on va normaliser f par $\max_f ||f||_{L^2(\Omega_h)}$

Formulation faible :

On utiliser une méthode directe pour inclure les conditions de Dirichlet homogène :

$$\int_{\Omega_h} \nabla(\bar{\phi}w)\nabla(\bar{\phi}v) - \int_{\partial\Omega_h} \frac{\partial}{\partial n}(\bar{\phi}w)\bar{\phi}v + G_h(w,v) = \int_{\Omega_h} f\bar{\phi}v - \left(\int_{\Omega_h} \nabla(g)\nabla(\bar{\phi}v) - \int_{\partial\Omega_h} \frac{\partial g}{\partial n}\bar{\phi}v\right) + G_h^{rhs}(v)$$

avec

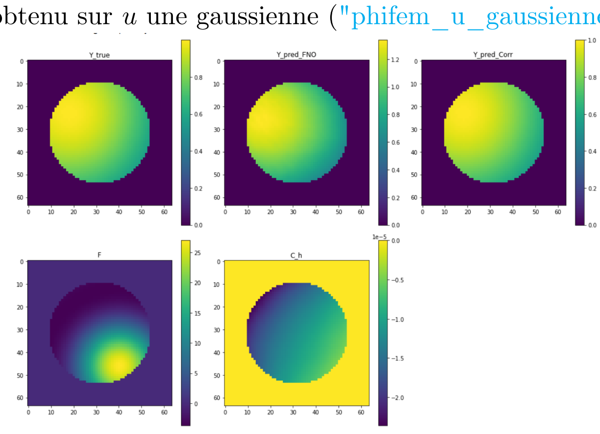
$$G_h(w,v) = \sigma h \sum_{E\in\mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial}{\partial n}(\bar{\phi}w)\right]\left[\frac{\partial}{\partial n}(\bar{\phi}v)\right] + \sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \Delta(\bar{\phi}w)\Delta(\bar{\phi}v)$$

et

$$G_h^{rhs}(v) = -\sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T f\Delta(\bar{\phi}v) - \sigma h \sum_{E\in\mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial g}{\partial n}\right]\left[\frac{\partial}{\partial n}(\bar{\phi}v)\right] - \sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \Delta(g)\Delta(\bar{\phi}v)$$

2.2 Résultat

Voici un exemple de résultats obtenu sur u une gaussienne ("phifem_u_gaussienne").



Conclusion

C'est en fin de semaine qu'on s'est rendue compte du problème du FNO : qu'il faut retourner w et pas u. Les modifications ont été faites mais les résultats ne semblent toujours pas convaincant (résultats supprimés malencontreusement).

En fait, on a remarqué que prendre $g = u_{ex}$ sur Ω entier posait problème pour la correction, car on obtient $w = 0$ et donc le problème de correction n'a plus de sens. Une solution à celà a été proposé par Emmanuel : il faudrait prendre $g = u_{ex}$ sur Γ et étendre la solution (par exemple en utilisant un nouveau réseau de neurones qui nous fournirait une solution lisse). Mais pour l'instant, on va simplement choisir une autre solution analytique, où on n'est pas obligé de prendre $g = u_{ex}$ sur tout le domaine.

On a également eut l'idée de se ramener à un problème homogène (passage du problème en f au problème en $\tilde{f} = f + \Delta g$)

Un autre problème constaté est qu'il faut utilisé le ϕ initial pour la génération des espaces $\mathcal{F}_h^\Gamma$ et $\mathcal{T}_h^\Gamma$.

3 Semaine 3 : 20/02/2023 - 24/02/2023

Résumé

Pour cette semaine, on considère une nouvelle solution analytique (solution trigonométrique : $\sin * \cos$). Avec cette nouvelle solution, on va pouvoir prendre g différent de u_{ex} sur Ω comme souhaité pendant la semaine précédente.

A défaut de pouvoir faire tourner les entrainements (car plus d'units dispo sur colab) et en attente d'une solution avec v100, on va s'intéresser uniquement au problème de correction où on prendra comme $\bar{\phi}$ notre u_{ex} ($-g$ si non homogène). En lui donnant comme nouvelle levelset notre solution exacte, C doit être très proche de 1 (à l'erreur machine).

3.1 Génération des données

On considère toujours Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x,y) = -1/8+(x-1/2)^2+(y-1/2)^2$ et le domaine fictif $O = (0,1)^2$.

On souhaite résoudre

$$\begin{cases} -\Delta u &= f\,, \quad \text{dans } \Omega\,, \\ u &= g\,, \quad \text{sur } \Gamma\,, \end{cases}$$

Notre solution analytique est

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos(k_2(x^2+y^2))\,,$$

avec $k_1, k_2 \sim \mathcal{U}([0.1, 0.5])$.

Ainsi

$$g(x,y) = \cos(k_2(x^2+y^2))$$

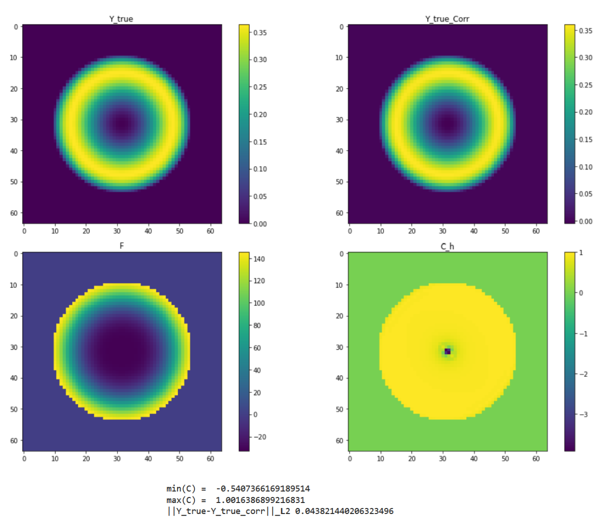
On considérera également la solution analytique au problème homogène :

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2+(y-0.5)^2\right)\right)\,,$$

Les formulation faibles sont les mêmes que précédemment.

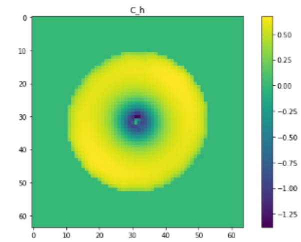
3.2 Correction

Dans un premier temps, on a pris notre nouvel levelset comme étant notre solution exacte sur une image 64*64 puis on interpoler cette levelset ("u_trigo_homogene"). En prenant $\phi = u_{ex}$, on espère obtenir C très proche de 1 (en augmentant le degré d'interpolation, on espère atteindre l'erreur machine). Les résultats obtenus n'étaient pas bon :

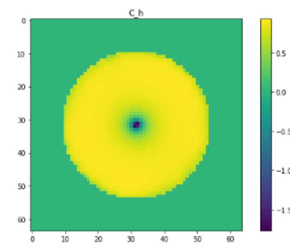


En fin de semaine, on s'est rendue compte que l'interpolation de ϕ posait problème pour la correction. Lorsque $\bar{\phi} = u_{ex}$, on obtient bien l'erreur machine mais en lui donnant notre levelset sous la forme d'une image nb_vert×nb_vert (ce qui est plus proche de ce que le réseau fournira), les résultats deviennent incohérents. On a donc tenter deux approches ("u_trigo_homogene_modif") : griddata (de scipy) et extrapolate (de FEniCS). L'option du extrapolate n'a pas fonctionné (nan ?). Pour l'autre option avec le griddata, on a constaté que interpoler avec tous les points de l'image était trop lourd, on a donc décidé de prendre pour chaque point (x,y) un certains nombres de points d'interpolations autour de lui. Voici les résultats obtenus avec différents nb_vert et différents nombres de points d'interpolation :

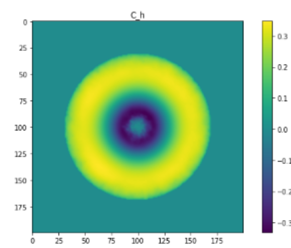
```
nb_vert =  64  ; nb_pts_inter =  20
min(C) =  -1.3784305135502903
max(C) =  0.6775565517686649
||Y_true-Y_true_corr||_L2 0.4496541682615938
```




```
nb_vert =  64  ; nb_pts_inter =  50
min(C) =  -1.767988511885549
max(C) =  0.9318459960999035
||Y_true-Y_true_corr||_L2 0.10616354494514865
```




```
nb_vert =  200  ; nb_pts_inter =  20
min(C) =  -0.3312977177567371
max(C) =  0.34758316279438156
||Y_true-Y_true_corr||_L2 0.7271864833222375
```



Ces résultats ont été obtenus la semaine suivante.

Conclusion

L'idée pour la semaine prochaine et d'effectuer quelques tests pour essayer de comprendre ce qui pose problème avec l'interpolation. Une fois ce type de problème réglé, on espère pouvoir entrainer le modèle sur la v100.

4 Semaine 4 : 27/02/2023 - 03/03/2023

Résumé

On cherche à comprendre les problèmes d'interpolation de phi dans le cadre de la correction sur la solution exacte du problème de poisson avec condition de dirichlet homogène. Après la réunion du 28/02 avec Emmanuel et Vanessa, les points suivants sont à traiter :

- tester avec solution analytique + perturbation !
- enlever les termes de stabilisation afin de voir si le problème est dans la dérivée seconde de phi
- visualiser le ϕ et le ϕ' EF et le comparer au ϕ et ϕ' analytique
- comprendre le problème du extrapolate

Après la réception d'un ordinateur, on a passé la moitié de la semaine à essayer d'installer ce dont on avait besoin. En fin de semaine, les installations ont enfin été faite et je peux maintenant générer des données et entraîner le modèle sur ce nouveau PC.

4.1 Génération des données

On considère toujours Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x,y) = -1/8+(x-1/2)^2+(y-1/2)^2$ et le domaine fictif $O = (0,1)^2$.

On souhaite résoudre

$$\begin{cases} -\Delta u &= f\,, \quad \text{dans } \Omega\,, \\ u &= 0\,, \quad \text{sur } \Gamma\,, \end{cases}$$

Notre solution analytique est

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1\frac{\pi}{2}\right)} \times \sin\left(k_1\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right)\,,$$

avec $k_1 \sim \mathcal{U}([0.1, 0.5])$.

4.2 Correction

On cherche à comprendre les problèmes d'interpolation de phi dans le cadre de la correction sur la solution exacte du problème de poisson avec condition de Dirichlet homogène ("tests_interpolation"). On va effectuer plusieurs tests :

Solution analytique + Perturbation.

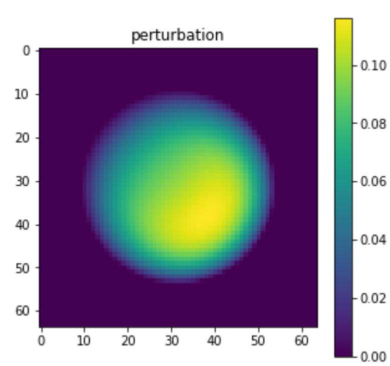
Pour simuler la solution fournit par le FNO on va considérer comme nouvelle level-set notre solution analytique plus une petite perturbation. La perturbation choisie est définie par :

$$P(x,y) = \epsilon \times \sin(k_1(x+y)) \times \cos(4\pi((x-0.5)^2+(y-0.5)^2))$$

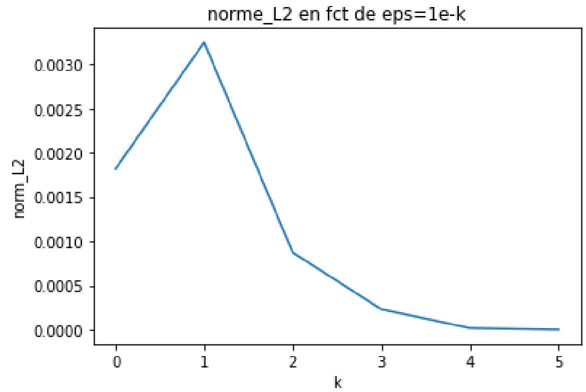
On a alors

$$\Phi(x,y) = u_{ex}(x,y) + P(x,y)$$

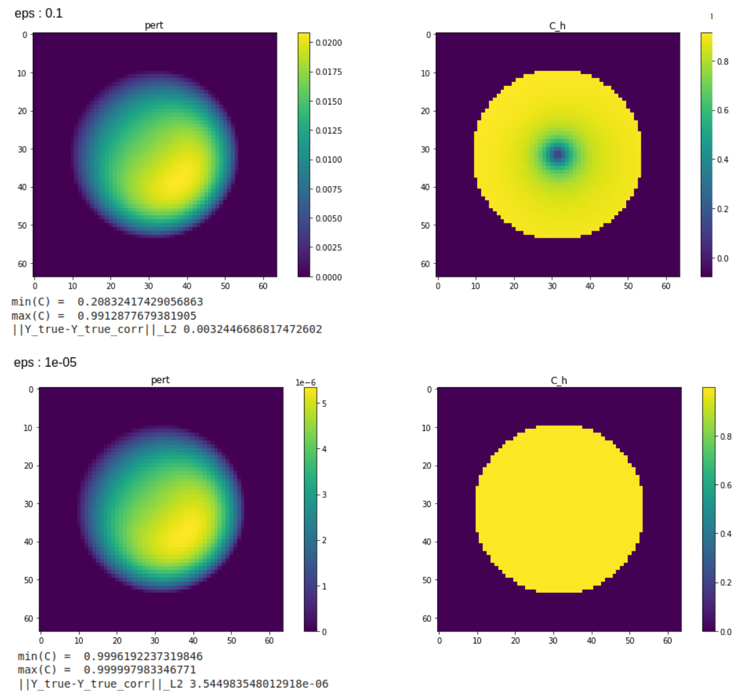
En prenant $\epsilon = 1$, on a :



On obtient en normes L_2 les différentes erreurs en fonction de $\epsilon = 1e - k$ entre la solution initiale fournit Φ et le solution après correction ΦC :

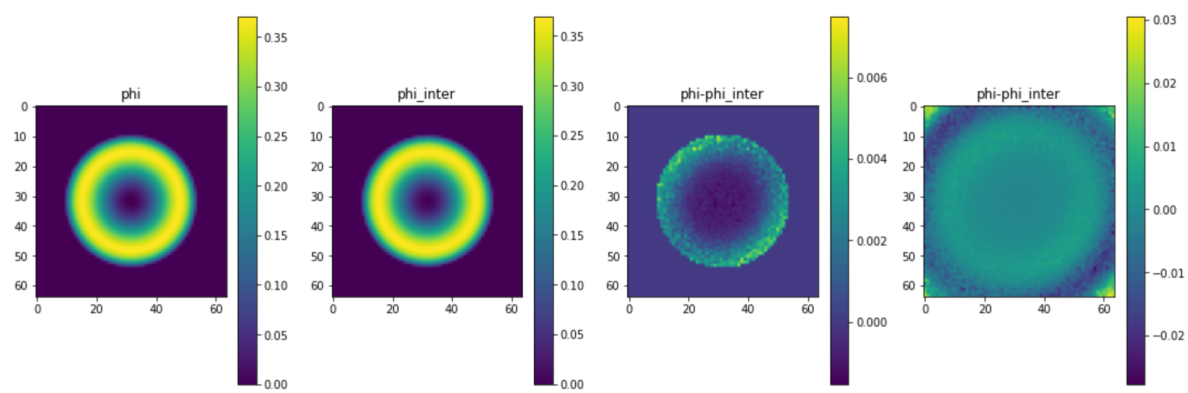


Il semblerait que plus la perturbation est petite et plus l'erreur est petite. De plus, C est de plus en plus proche de 1 :



Griddata

On a comparé Φ exact et notre Φ interpolé avec griddata (pour nb_vert=64 et nb_pts=20). Voici les résultats obtenus :



Il semblerait que sur le bord du cercle, les résultats soit moins bons. On a pas cherché à comprendre plus loin car on va utiliser la fonction extrapolate de FEniCS.

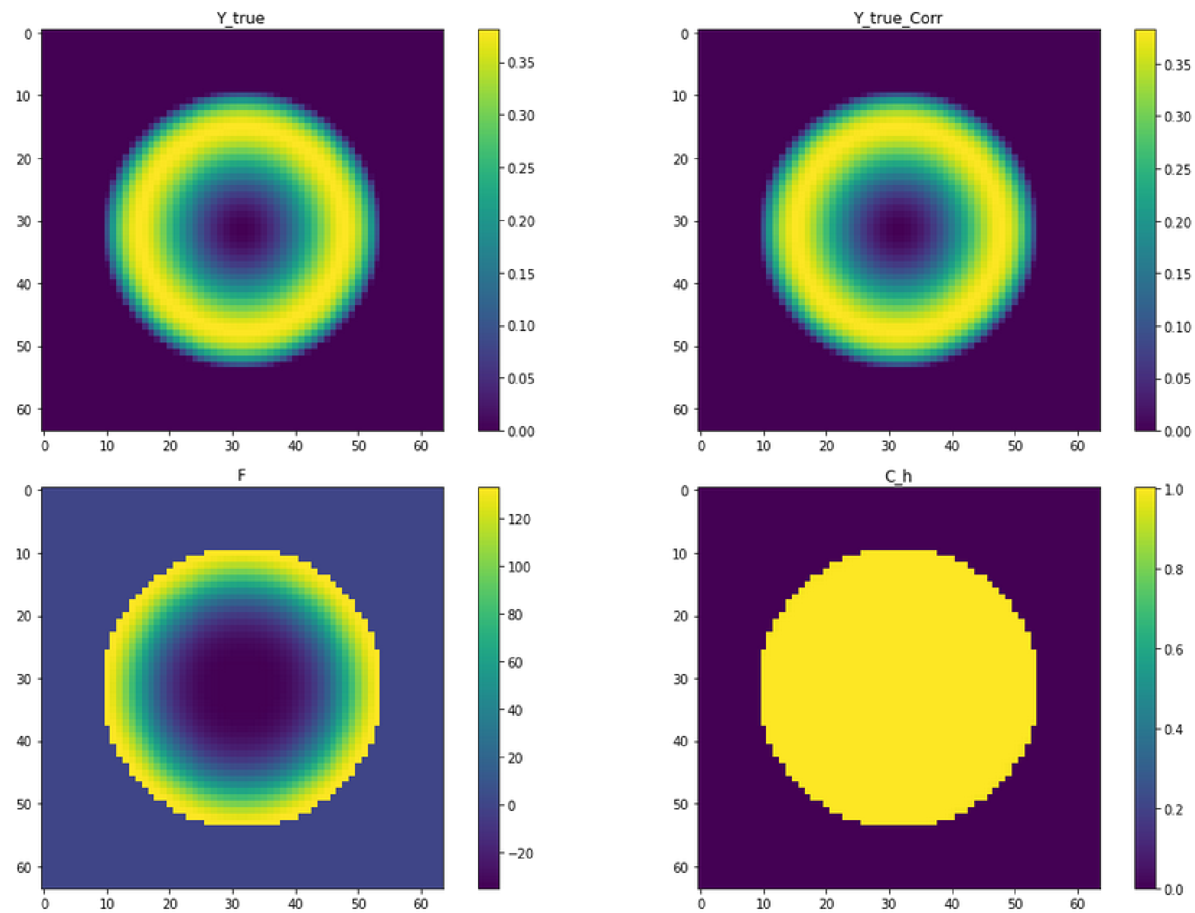
Extrapolate de FEniCS

En milieu de semaine, on s'est rendu compte que le problème avec la foncion extrapolate que l'on avai eut la semaine dernière était que l'on pouvait n'incrémenter le degré d'interpolation que de 1.

```python
def get_phi_fct(Y, degree = 2):
    Y_np = Y[0,:,:,0]

    nb_vert = Y_np.shape[0]
    coeff = 1

    boxmesh = RectangleMesh(Point(0, 0), Point(1,1), nb_vert-1, nb_vert-1)

    V = FunctionSpace(boxmesh, "CG", 1)
    v2d = vertex_to_dof_map(V)

    phi_fct = Function(V)
    phi_fct.vector()[v2d] = Y_np.flatten()

    for i in range(2,degree+1):
        V2 = FunctionSpace(boxmesh, "CG", i)
        phi_fct2 = Function(V2)
        phi_fct2.extrapolate(phi_fct)
        phi_fct = phi_fct2

    return phi_fct
```

Voici les résultats obtenus pour une degré 2 :

```
degree =  2
num of cell in the ghost penalty: 302
||phi_ex-phi_inter||_L2 0.0002493552722685459
solver duration: 22.493684768676758
```




```
min(C) =  0.994576987356184
max(C) =  1.0032304281199327
||Y_true-Y_true_corr||_L2 0.00044278483969670816
```

Conclusion

La semaine prochaine, on peut enfin entraîner le modèle car les installations ont été faites sur le nouveau pc.

5 Semaine 5 : 06/03/2023 - 10/03/2023

Résumé

Pendant cette semaine, on a globalement cherché à comprendre les problèmes liés à l'interpolation de ϕ. On s'est donc concentré sur la précision de la correction lorsque l'on prend la solution analytique en entrée. On a testé avec deux solutions analytiques : la solution trigonométrique considérées précédemment et une solution polynomiale. La partie où l'on va utilisé le FNO sera considéré dans un second temps.

Dans toute la suite, les erreurs calculées en norme L2 sont les erreurs relatives calculées de la manière suivante :

$$||y_{true} - y_{pred}||^2_{L^2(\Omega_h),rel} = \frac{\int_{\Omega_h}(y_{true} - y_{pred})^2}{\int_{\Omega_h} y_{true}^2}$$

5.1 Solution analytique trigonométrique

On considère la solution analytique (au problème de Poisson avec conditions de Dirichlet homogène) :

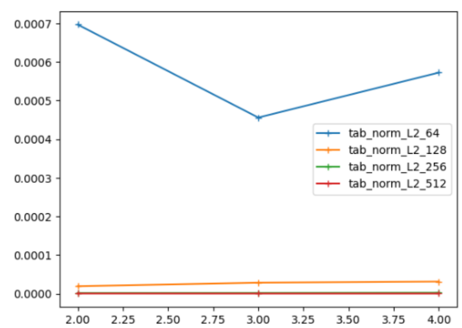
$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 ((x-0.5)^2 + (y-0.5)^2)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 ((x-0.5)^2 + (y-0.5)^2)\right),$$

avec $k_1 \sim \mathcal{U}([0.1, 1])$.

Extrapolate en faisant varier nb_vert

On va calculer l'erreur en norme L2 pour différents degré d'extrapolation (de 2 à 4) et pour différents nb_vert. On calculera les facteurs multiplicatifs entre les résultats obtenus pour différentes valeurs de nb_vert.

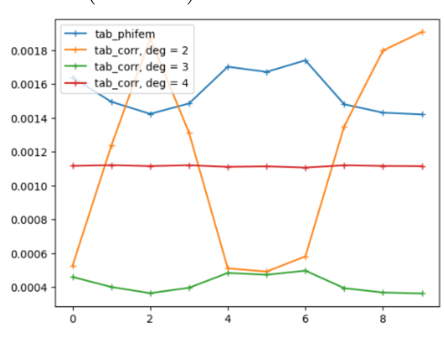
Attention : Les résultats obtenus pour chaque valeur de nb_vert n'ont pas été calculés pour les mêmes valeurs du paramètres k_1.

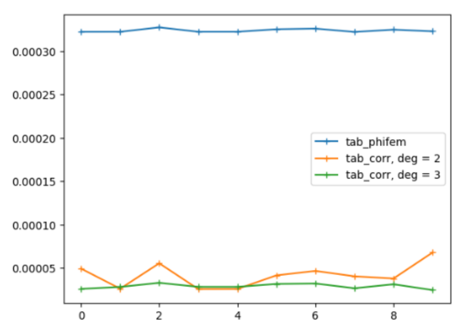



```
tab_norm_L2_64 :  [0.0006967652600855427, 0.00045597122067689814, 0.0005724634964613069]

tab_norm_L2_128 :  [1.922158215675266e-05, 2.823458823293494e-05, 3.11003321120869e-05]
facteur 64-128 :  [36.24911073 16.14938447 18.40698982]

tab_norm_L2_256 :  [1.4710081923791521e-06, 1.6045669223902737e-06, 2.1645064826912723e-06]
facteur 64-256 :  [473.66511192 284.1708964  264.47760773]
facteur 128-256 :  [13.06694433 17.5963918  14.36832477]

tab_norm_L2_512 :  [2.1037824077334138e-07, 1.0974522716743716e-07, 1.1186391627112441e-07]
facteur 64-512 :  [3311.96447657 4154.81595369 5117.49915025]
facteur 128-512 :  [ 91.36677865 257.27395133 278.01933947]
facteur 256-512 :  [ 6.99220693 14.62083558 19.34946098]
```

Comparaison avec PhiFEM

On veut comparer les erreurs en normes L2 pour différents degré d'interpolation. On fait une moyenne des résultats obtenus sur 10 valeurs de paramètres k_1 (nb_data=10). Les comparaisons entre PhiFEM et la correction (pour différents degré d'interpolation) sont effectuées pour les mêmes valeurs de paramètres. On testera avec nb_vert=64 (à gauche) et nb_vert=128 (à droite).






```
[0.00154893 0.00154893 0.00154893]
[0.00115899 0.00042031 0.00111569]
facteur :  [1.33644568 3.68519288 1.38831814]
```

```
[0.00032393 0.00032393]
[4.15350306e-05 2.88233974e-05]
facteur :  [ 7.79897413 11.23846107]
```

Stratégie P1 fin -> Pk grossier

Méthode précédente (avec le extrapolate) : On part d'une grille nb_vert $\times$ nb_vert. On associe alors les valeurs au noeuds aux valeurs des degré de liberté $\mathbb{P}^1$ puis on va extrapoler en $\mathbb{P}^k$ nb_vert $\times$ nb_vert.

$$[\bar{\phi} = \phi w]_{ij\ grossier} \quad \longrightarrow \quad \mathbb{P}^1{}_{grossier} \quad \overset{extra}{\longrightarrow} \quad \mathbb{P}^k{}_{grossier}$$

On va considérer ici une nouvelle méthode : On part d'une grille fine nb_vert_fine $\times$ nb_vert_fine. On associe alors les valeurs aux noeuds aux valeurs des degrés de liberté $\mathbb{P}^1$ (nb_vert_fine $\times$ nb_vert_fine) puis on interpole en $\mathbb{P}^k$ nb_vert_coarse $\times$ nb_vert_coarse.

$$[\bar{\phi} = \phi w]_{ij\ fin} \quad \longrightarrow \quad \mathbb{P}^1{}_{fin} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^k{}_{grossier}$$

On effectuera plusieurs comparaisons (avec nb_data=10) :

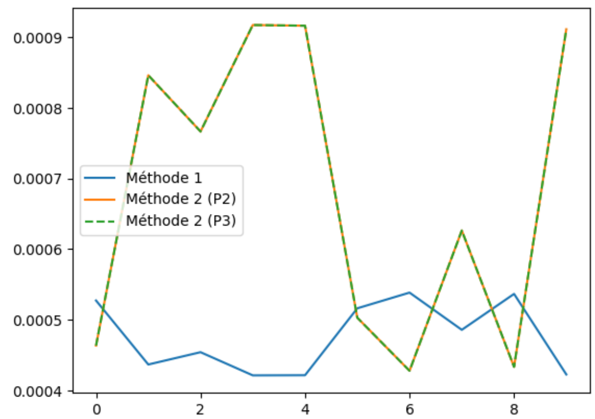
- **1er test :**

On va comparer les méthodes suivantes :

$$[\bar{\phi} = \phi w]_{ij\ 64} \quad \longrightarrow \quad \mathbb{P}^1{}_{64} \quad \overset{extra^3}{\longrightarrow} \quad \mathbb{P}^3{}_{64}$$

$$[\bar{\phi} = \phi w]_{ij\ 256} \quad \longrightarrow \quad \mathbb{P}^1{}_{256} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^2{}_{64}$$

$$[\bar{\phi} = \phi w]_{ij\ 256} \quad \longrightarrow \quad \mathbb{P}^1{}_{256} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^3{}_{64}$$



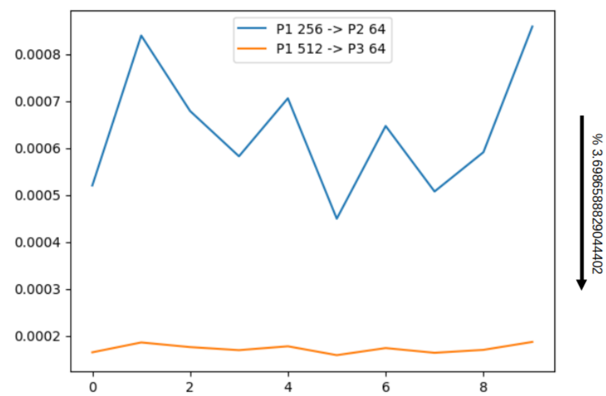
(Les courbes orange et verte sont très proches.)

- **2ème test :**

On va comparer les méthodes suivantes :

$$[\bar{\phi} = \phi w]_{ij\ 256} \quad \longrightarrow \quad \mathbb{P}^1{}_{256} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^2{}_{64}$$

$$[\bar{\phi} = \phi w]_{ij\ 512} \quad \longrightarrow \quad \mathbb{P}^1{}_{512} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^3{}_{64}$$



- **3ème test :**

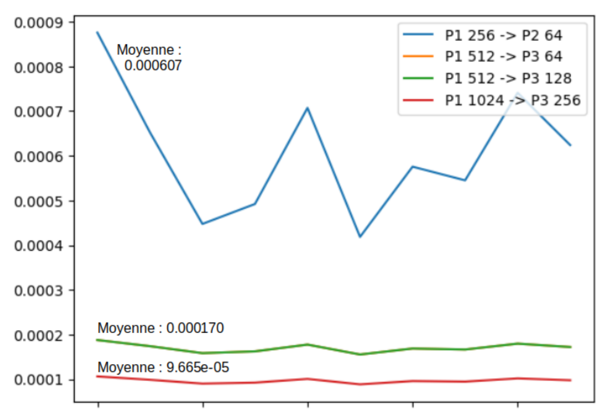
On va comparer les méthodes suivantes :

$$[\bar{\phi} = \phi w]_{ij\ 256} \quad \longrightarrow \quad \mathbb{P}^1{}_{256} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^2{}_{64}$$

$$[\bar{\phi} = \phi w]_{ij\ 512} \quad \longrightarrow \quad \mathbb{P}^1{}_{512} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^3{}_{64}$$

$$[\bar{\phi} = \phi w]_{ij\ 512} \quad \longrightarrow \quad \mathbb{P}^1{}_{512} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^3{}_{128}$$

$$[\bar{\phi} = \phi w]_{ij\ 1024} \quad \longrightarrow \quad \mathbb{P}^1{}_{1024} \quad \overset{inter}{\longrightarrow} \quad \mathbb{P}^3{}_{256}$$



(Les courbes orange et verte sont très proches.)

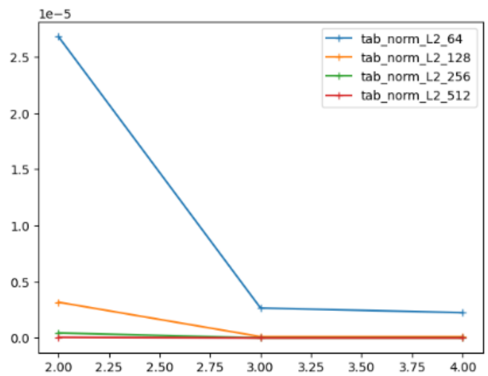
5.2 Solution analytique polynomiale

On considère la solution analytique (au problème de Poisson avec conditions de Dirichlet homogène) :

$$u_{ex}(x,y) = (1 + k_1 * x + k_2 * y + x^2 + y^2 + x^3 + y^3) \times \phi(x,y)$$

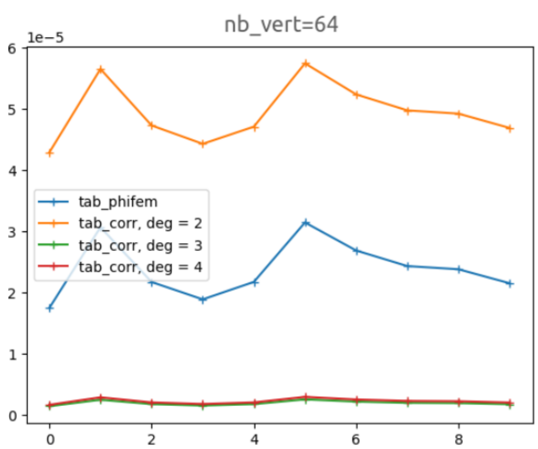
avec $k_1, k_2 \sim \mathcal{U}(1,5)$
On va effectuer exactement les mêmes tests que dans le cas trigonométrique.

Extrapolate en faisant varier nb_vert

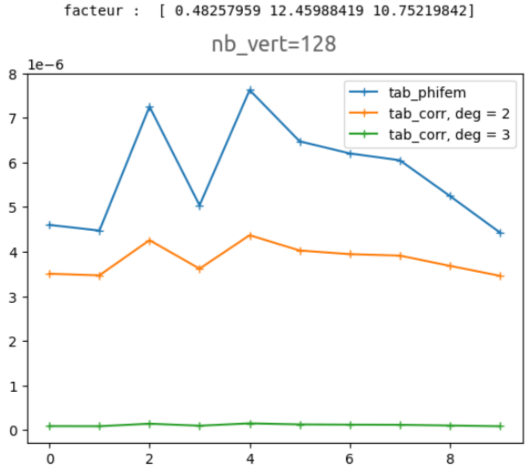



```
tab_norm_L2_64 :  [2.6795787869257804e-05, 2.6620051945242716e-06, 2.251929379799515e-06]

tab_norm_L2_128 :  [3.1816265809951308e-06, 1.1787885864954612e-07, 1.1458184372097631e-07]
facteur 64-128 :  [ 8.42204048 22.58254979 19.65345736]

tab_norm_L2_256 :  [4.45770025132257e-07, 5.792778233797276e-09, 8.63884715265421e-09]
facteur 64-256 :  [ 60.11123754 459.53859911 260.67475671]
facteur 128-256 :  [ 7.13737219 20.34927869 13.26355724]

tab_norm_L2_512 :  [5.892283603687297e-08, 3.271917059060034e-10, 5.05223428330941e-10]
facteur 64-512 :  [ 454.76066109 8135.91893215 4457.29404758]
facteur 128-512 :  [ 53.99649431 360.27459291 226.79439887]
facteur 256-512 :  [ 7.56531856 17.70453874 17.09906285]
```

Comparaison avec PhiFEM



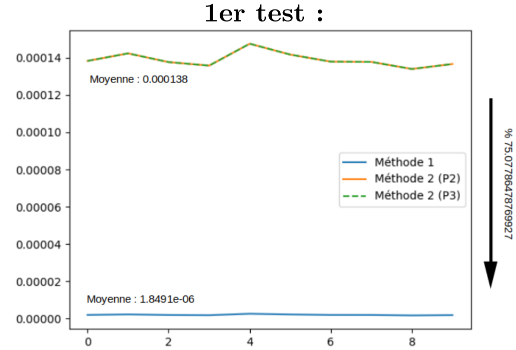

```
[2.37940622e-05 2.37940622e-05 2.37940622e-05]
[4.93059856e-05 1.90965356e-06 2.21294857e-06]
facteur :  [ 0.48257959 12.45988419 10.75219842]
```



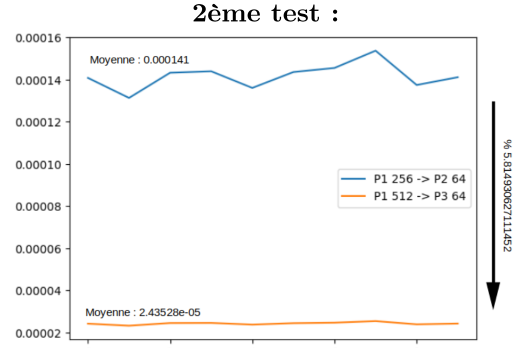

```
[5.74036956e-06 5.74036956e-06]
[3.82656316e-06 1.16131386e-07]
facteur :  [ 1.50013715 49.42995808]
```

Stratégie P1 fin -> Pk grossier

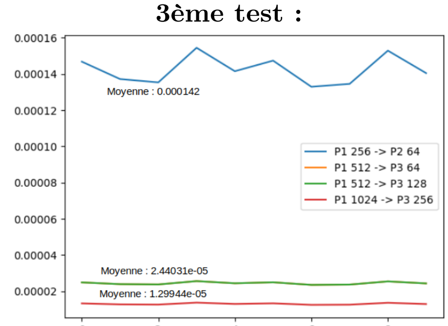
1er test :



2ème test :



3ème test :

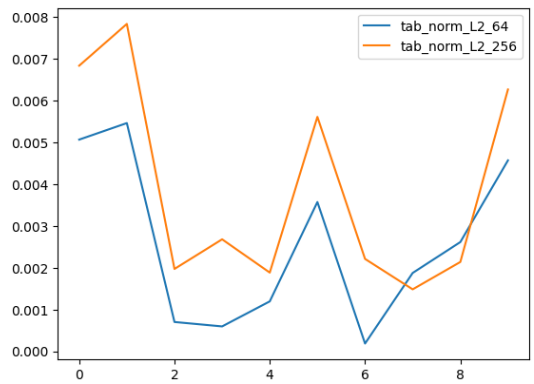


5.3 Tests sur le FNO

On veut maintenant comparer la première stratégie utilisée (avec le extrapolate à degré fixé égal à 3) et la nouvelle stratégie (P1 fin puis interpolation en Pk grossier). on considère ici que la solution que l'on va introduire dans la correction/certification est la sortie d'un FNO.

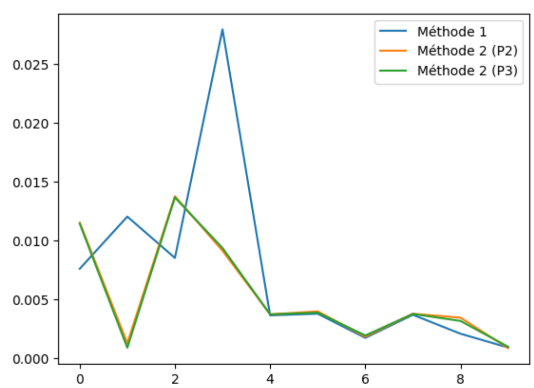
Appel avec nb_vert plus fin

On entraine le réseau FNO avec nb_vert=64. Avant de mettre en place la stratégie, on cherche à voir si la précision de ce réseau (entrainé avec nb_vert=64) est meilleure lorsque l'on prend nb_vert plus petit. On va alors comparer l'erreur pour nb_data=10 lorsqu'on l'on appelle le réseau avec nb_vert=64 et nb_vert=256. Voici les résultats obtenus :



Comparaison des 2 stratégies

On va maintenant comparer la première méthode utilisée (avec le extrapolate à degré fixé égal à 3) et la nouvelle méthode (P1 fin puis interpolation en Pk grossier). On prendra nb_vert=256. Voici les résultats obtenus (pour $k = 2$ et $k = 3$) :



Conclusion

Pour l'instant, on ne va s'intéresser qu'à la partie avec la solution analytique. Il semblerait que la première méthode avec le extrapolate de FEniCS soit meilleure que la deuxième stratégie proposée. C'est pourquoi, vendredi, on a commencé (Killian et moi) à s'intéresser au code source du extrapolate afin de comprendre pourquoi les résultats obtenus sont meilleurs. On considérera le FNO dans un second temps.

6 Semaine 6 : 13/03/2023 - 17/03/2023

Résumé

Après les tests de la semaine dernière sur la partie de correction/certification du modèle où l'on prend la solution analytique comme nouvelle level-set, il semblerait que la méthode avec les meilleurs résultats soit celle où l'on utilise la méthode extrapolate de FEniCS. C'est pourquoi, cette semaine on s'est intéressé en détail au code source de cette fonction FEniCS (Extrapolation). Étant donné que je n'étais pas présente mardi, mercredi après-midi et jeudi car j'étais malade, c'est tout ce qui a été fait cette semaine. De plus, une grosse partie de la méthode reste encore floue : la construction de la matrice A (pour la résolution du système linéaire dans compute_coefficients).

6.1 Extrapolate - FEniCS

Pour illustrer les explications, nous considérerons un domaine rectangulaire maillés uniformément par des triangles. Dans la suite, nous considérerons également *cell0* comme étant une des cellules de maillage. On prendra comme exemple une extrapolation de $\mathbb{P}^1$ vers $\mathbb{P}^2$.

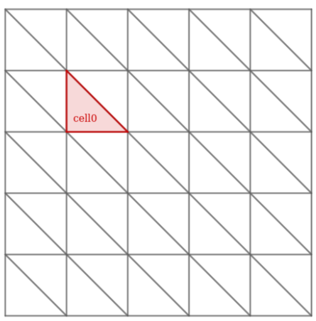


FIGURE 1 – Cell0

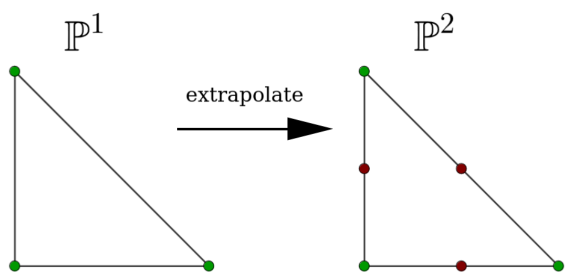


FIGURE 2 – Extrapolation de $\mathbb{P}^1$ vers $\mathbb{P}^2$

Corps de la fonction **extrapolate** qui a pour but d'extrapoler la fonction v en une fonction w :

```
void Extrapolation::extrapolate(Function& w, const Function& v)
```

On cherche à calculer la valeur en chacun des degrés de liberté associé à la fonction w. Pour cela, on va parcourir toutes les cellules du maillage dans le but de construire le tableau *coefficients* (de taille le nombre total de degrés de liberté associés à w). On appelle alors sur chacune des cellules la fonction **compute_coefficients** 6.1 qui va compléter le tableau *coefficients* aux indices associées à ses degrés de liberté. Comme les degrés de liberté d'une cellule peuvent être communs à ceux d'une autre, on finira par faire une moyenne des coefficients en chacun des degrés de liberté en utilisant la fonction **average_coefficients** 6.1, ce qui nous donne alors la fonction w.

On notera que deux espaces de fonctions sont créés dans la fonction extrapolate : l'espace V est notre espace de départ (dans l'exemple l'espace $\mathbb{P}^1$) et W est l'espace d'arrivée (espace $\mathbb{P}^2$). On prendra $v \in V$ et $w \in W$.

compute_coefficients

Corps de la fonction :

```
void Extrapolation::compute_coefficients(
std::vector<std::vector<double>>& coefficients,
const Function& v,
const FunctionSpace& V,
const FunctionSpace& W,
const Cell& cell0,
const std::vector<double>& coordinate_dofs0,
const ufc::cell& c0,
const Eigen::Ref<const Eigen::Matrix<dolfin::la_index, Eigen::Dynamic, 1>> dofs,
std::size_t& offset)
```

Cette fonction a pour but de compléter le tableau *coefficients* aux indices associées aux degrés de liberté d'une cellule donnée *cell0*. Autrement dit, on cherche à déterminer les valeurs aux degrés de liberté de la cellule *cell0* en utilisant l'information que nous apporte les cellule voisines à celle-ci.

On commence par construire les tableaux *cell2dof2row* et *unique_ dofs* en utilisant la fonction **build_unique_dofs** 6.1. L'ensemble *unique_ dofs* contient tous les degrés de liberté de notre espace de départ V (associé à la fonction v) des cellules voisines à *cell0*. Le dictionnaire *cell2dof2row* permet d'associer à chaque degré de liberté (unique) d'une cellule donnée un numéro de ligne unique.

Ensuite, on définit N le nombre de degré de liberté associé à un élément de W (dans notre cas $N = 6$) et M le nombre de degré de liberté (unique) des cellules voisines à la cellule courante *cell0* (dans notre cas les nœuds des cellules voisines et donc $M = 12$). Attention : il faut que $M \geq N$ pour avoir suffisamment de degré de libertés pour pouvoir construire l'extrapolation.

On peut maintenant créer la matrice A (de taille $M \times N$) et le vecteur b (de taille M). En parcourant les cellules voisines de la cellule courante *cell0*, on va compléter la matrice A et le vecteur b en utilisant la fonction **add_cell_equations** 6.1. A noter que la cellule courante *cell0* est inclue dans ses cellules voisines.

On pourra ensuite résoudre le système linéaire $Ax = b$ qui nous donnera la valeur en chacun des degrés de liberté de la cellule courante *cell0*. Ces valeurs sont alors ajoutées au tableau global *coefficients* qui nous fournit après avoir utilisé la fonction **average_coefficients** 6.1 les valeurs en chaque degré de liberté de w.

build_unique_dofs

Corps de la fonction :

```
void Extrapolation::build_unique_dofs(
std::set<std::size_t>& unique_dofs,
std::map<std::size_t, std::map<std::size_t, std::size_t>>& cell2dof2row,
const Cell& cell0,
const FunctionSpace& V)
```

Cette fonction a pour but de compléter les tableaux *cell2dof2row* et *unique_ dofs* donnés en entrée. A noter que au total, on a le même nombre de degré de liberté dans *cell2dof2row* et *unique_ dofs*.

On commence par remplir un ensemble contenant les cellules voisines à *cell0*. Pour être plus précis, les cellules voisines à *cell0* sont les cellules ayant un nœud commun avec la cellule courante *cell0* (Figure 3).

En parcourant ensuite chacune de ces cellules, on va pouvoir compléter les tableaux *cell2dof2row* et *unique_ dofs* donnés en entrée en appelant la fonction **compute_unique_dofs** 6.1.

Dans le cas de notre exemple, on va numéroter tous les nœuds des cellules voisines à *cell0* et on supposera que le parcours des cellules est effectuées dans un ordre précis (Figure 4).

Alors le set *unique_ dofs* contiendra tous les noeuds des cellules voisines à *cell0* :

$$unique\_dofs = \{n1, n2, n3, \ldots, n12\}$$

Et le dictionnaire *cell2dof2row* associé à *cell0* est construit de la manière suivante :

$$cell2dof2row = \{ \begin{array}{ll} "cell0" : \{"n1" : 0, "n2" : 1, "n3" : 2\}, & \\ "cell1" : \{"n4" : 3\}, & \\ "cell2" : \{"n5" : 4\}, & \\ \ldots & \} \end{array}$$

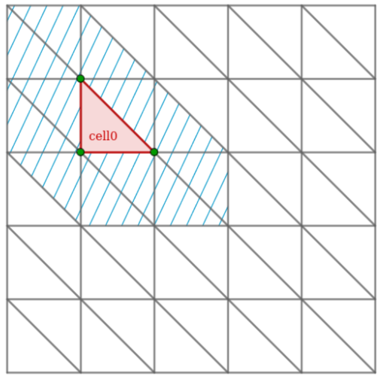


FIGURE 3 – Cellules voisines à Cell0

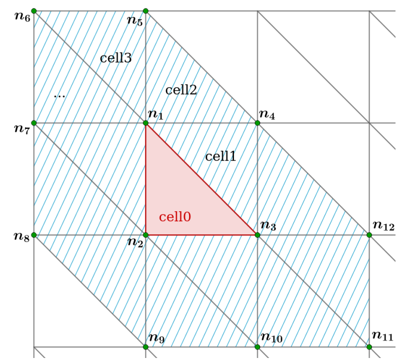


FIGURE 4 – Parcours des cellules voisines

compute_unique_dofs

Corps de la fonction :

```
std::map<std::size_t, std::size_t> Extrapolation::compute_unique_dofs(
const Cell& cell,
const FunctionSpace& V,
std::size_t& row,
std::set<std::size_t>& unique_dofs)
```

Cette fonction a pour but de traiter chacune des cellules voisines afin de compléter les tableaux *unique_ dofs* et *cell2dof2row* créés dans **compute_coefficients** 6.1.

Pour une cellule *cell* (voisine à *cell0*), on va parcourir chacun de ses degrés de liberté associé à l'espace V. Autrement dit, on parcourt tous les degrés de liberté dont la valeur est connue (dans notre cas les degrés de liberté $\mathbb{P}^1$ et donc les noeuds de la cellule). Si ce degré de liberté fait partie de *unique_ dofs*, on ne fait rien. Sinon, on l'ajoute à *unique_ dofs*. On va également créer un tableau *dof2row* qui a pour but d'associer un degré de liberté à un numéro de ligne unique. Ce dictionnaire *dof2row* est retourné par la fonction et permet de remplir le dictionnaire plus général *cell2dof2row* créé dans **compute_coefficients** 6.1.

add_cell_equations

Corps de la fonction :

```
void Extrapolation::add_cell_equations(
Eigen::MatrixXd& A,
Eigen::VectorXd& b,
const Cell& cell0,
const Cell& cell1,
const std::vector<double>& coordinate_dofs0,
const std::vector<double>& coordinate_dofs1,
const ufc::cell& c0,
const ufc::cell& c1,
const FunctionSpace& V,
const FunctionSpace& W,
const Function& v,
std::map<std::size_t, std::size_t>& dof2row)
```

Cette fonction a pour but de remplir une partie de la matrice A et du vecteur b à partir de la cellule courante *cell0* et d'une de ses cellules voisines *cell1*.

On commence par créer les fonctions de base Φ_j associées aux degrés de liberté de la cellule courante *cell0*.
On va ensuite parcourir les degrés de liberté associés à la cellule voisine *cell1* dans le dictionnaire *cell2dof2row* (c'est le tableau *dof2row* donné en argument). On évalue alors Φ_j en chacun des degrés de liberté de la cellule

voisine *cell1*. On peut alors compléter $A(row, j)$ (où row nous ai donné par le dictionnaire *dof2row*). On complète également $b(row)$ par la valeur aux degrés de liberté associés à l'espace V (les nœuds dans notre cas).

average_coefficients

Corps de la fonction :

```
void Extrapolation::average_coefficients(
Function& w,
std::vector<std::vector<double>>& coefficients)
```

Cette fonction a pour but de faire la moyenne pour chaque degré de liberté des coefficients calculés. Elle associe ensuite ces valeurs au vecteur w.

Conclusion

La semaine prochaine, il faudrait continuer à essayer de comprendre la partie de construction de la matrice A.

7 Semaine 7 : 20/03/2023 - 24/03/2023

Résumé

Lundi : Après discussion avec Michel des résultats obtenus les semaines précédentes, on a décidé de se concentrer sur la comparaison des erreurs PhiFEM, FNO et avec la correction (avec la méthode extrapolate de FEniCS). On a également parlé d'entrainer le modèle avec des données plus fines (nb_vert=128). Sur ma machine, ça n'a pas été possible (OOM : Out Of Memory). Je me suis donc connectée à Titan sur lequel l'entrainement a également crashé (Noyau mort). Je pense que le problème vient de l'installation de tensorflow (car j'ai cloné sur Titan l'environnement conda phifem) : il faudrait tester d'installer htop pour voir si la RAM est pleine. C'est un problème qui est mis de côté pour l'instant.

Mardi : Après discussion avec Michel et Emmanuel, Emmanuel a proposé de regarder les facteurs de division d'erreur sur la solution analytique pour différents $\mathbb{P}^k$. (Ensuite, on a une réunion d'équipe sur les FNO) Puis, une fois ces résultats obtenus, Michel a proposé de vérifier les pentes de convergence au niveau de l'interpolation de la level-set et sur la correction. Il a également proposé une nouvelle idée pour faire en sorte que la correction soit moins couteuse que PhiFEM classique : On entraine le réseau en 32×32 $\mathbb{P}^2$ puis on applique la correction sur du $\mathbb{P}^1$ puis on compare les erreurs (PhiFEM, FNO et FNO+Corr).

Pour le reste de la semaine, on s'est alors concentré sur cette nouvelle idée pour le FNO. La première complexité a été de pouvoir faire le passage d'une solution $\mathbb{P}^2$ de Numpy à FEniCS et de FEniCS à Numpy. Au vue des résultats obtenus sur la solution analytique trigonométrique, on pense qu'il est possible que les données soient "trop simple" (ne varie pas assez) pour que la correction ait un intérêt. En effet, il semblerait qu'un entrainement de seulement 500 époques nous donnent déjà de très bons résultats avec le FNO. La correction a alors du mal à corriger la solution obtenue par le FNO car elle est déjà très bonne. Pour justifier cette hypothèse, on choisit de changer le problème considéré : on repasse à Poisson avec condition de Dirichlet homogène où on prend f gaussienne. Comme l'on a pas de solution analytique pour ce problème, on considérera une solution sur-raffinée comme solution de référence.

On considérera dans la suite deux normes. La première norme est une norme relative calculée avec FEniCS :

$$||y_{true} - y_{pred}||^2_{L^2(\Omega_h),rel} = \frac{\int_{\Omega_h}(y_{true} - y_{pred})^2}{\int_{\Omega_h} y_{true}^2}$$

La seconde est calculée à partir des "images" représentées par des tableaux Numpy ou des tenseurs :

$$||y_{true} - y_{pred}||^2_2 = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(y_{true}(i,j) - y_{pred}(i,j))^2 * mask$$

où N est nb_vert.

(PS : j'aurais du mettre la seconde norme en norme relative.)

7.1 Comparaison PhiFEM, FNO et FNO+Corr

On considère la solution analytique trigonométrique suivante :

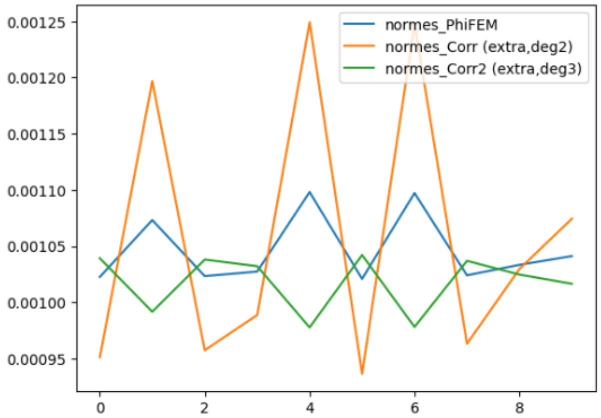
$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1\frac{\pi}{2}\right)} \times \sin\left(k_1\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right),$$

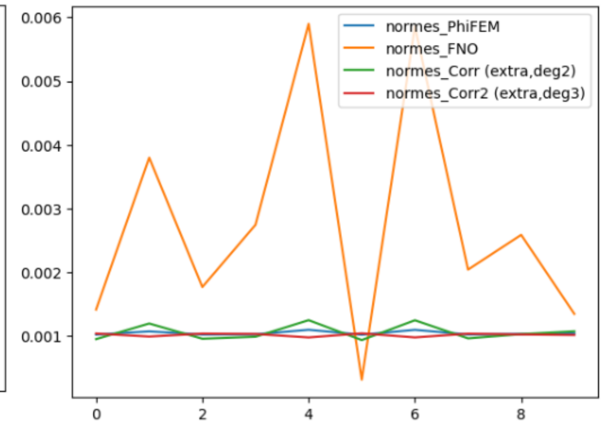
On entraîne notre FNO avec nb_vert=64, nb_data=1000 et sur 1000 époques. On calcule alors les normes relatives suivantes :

$$||u_{ex} - u_{PhiFEM}||_2,\ ||u_{ex} - u_{FNO}||_2,\ ||u_{ex} - u_{Corr}||_2$$

où la méthodes choisi pour la correction est une extrapolation de degré 2 et 3.

On obtient les résultats suivants :





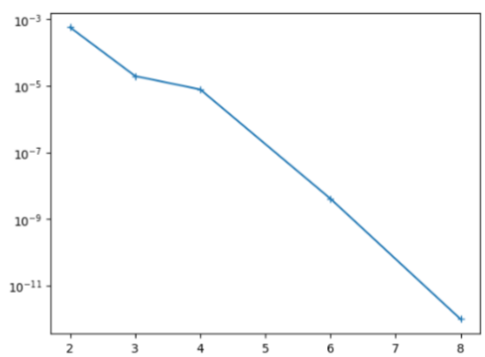

```
||Y_true-Y_PhiFEM||_L2 :  [0.00102271 0.00107332 0.00102346 0.00102749 0.00109824 0.00102099
 0.00109734 0.00102417 0.00103351 0.00104125]
||Y_true-Y_pred_FNO||_L2 :  [0.00141527 0.00379858 0.00176916 0.00274529 0.00589883 0.00031555
 0.0058568  0.00204237 0.00258738 0.00134767]
||Y_true-Y_corr||_L2,deg=2 :  [0.00095133 0.00119688 0.00095755 0.00098889 0.0012493  0.00093661
 0.00124785 0.00096333 0.00102987 0.00107454]
||Y_true-Y_corr||_L2,deg=3 :  [0.00103936 0.00099179 0.00103816 0.0010323  0.0009778  0.00104225
 0.00097825 0.00103706 0.00102482 0.00101666]
```

7.2 Facteurs de division de l'erreur sur la solution analytique

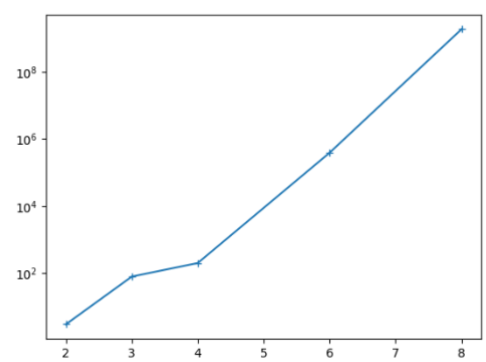
On considère encore la solution analytique trigonométrique suivante :

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1\frac{\pi}{2}\right)} \times \sin\left(k_1\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2\left((x-0.5)^2+(y-0.5)^2\right)\right),$$

On cherche ici à déterminer les facteurs de division entre PhiFEM et la correction appliqué à la solution analytique. Voici les résultats obtenus :



```
Moyenne des erreurs L2 :
PhiFEM :  0.0015266256880406305
degré :  [2, 3, 4, 6, 8]
Inter :  [6.95339849e-05 1.30249776e-06 2.44578907e-08
    5.71454551e-12 6.29371730e-15]
Corr :  [5.73218942e-04 1.96955223e-05 7.80800690e-06
    4.07615388e-09 9.77961484e-13]
```



```
degré :  [2, 3, 4, 6, 8]
Facteur phifem-Corr :  [2.93037042e+00 7.76899093e+01
    1.96517935e+02 3.81979992e+05 1.83037565e+09]
```

7.3 Pentes de convergence (interpolation et correction)

On veut vérifier ici les pentes des droites de convergence sur la norme L^2 de la différence entre la solution exacte et son interpolation :

$$||u_{ex} - I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}$$

Dans un second temps on cherche à déterminer numériquement la pente des droites de convergence sur la norme L^2 de la différence entre la solution exacte et la correction appliquée à son interpolation :

$$||u_{ex} - CI_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^?$$

On traitera dans un premier temps le cas avec la solution analytique trigonométrique puis le cas où f est gaussienne où on prend comme solution de référence une solution sur-raffinée.

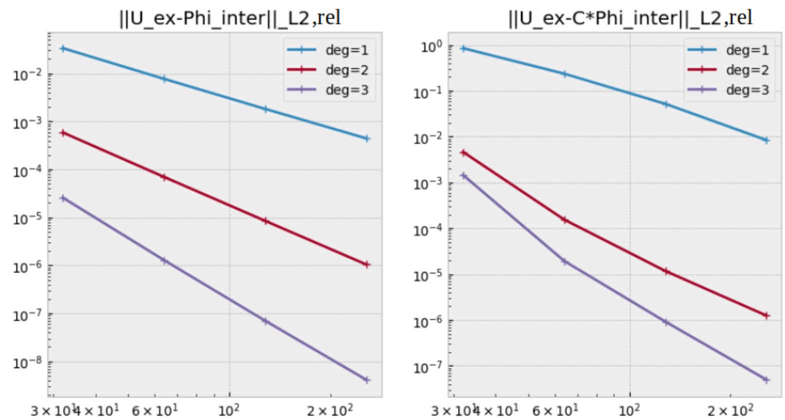
Solution analytique trigonométrique

On considère encore la solution analytique trigonométrique suivante :

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2+(y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2+(y-0.5)^2\right)\right),$$

On considère la solution exacte u_{ex} interpolé à un ordre assez élevé (de degré 8 par exemple). On interpole cette solution exacte (notre nouvelle level-set que l'on note $I_h u_{ex}$) dans $\mathbb{P}^k$ avec $k \in \{1,2,3\}$ et on fixe $C \in \mathbb{P}^1$.

On obtient les résultats suivants :



On obtient

$$||u_{ex} - I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}, \quad ||u_{ex} - C I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}$$

f gaussienne

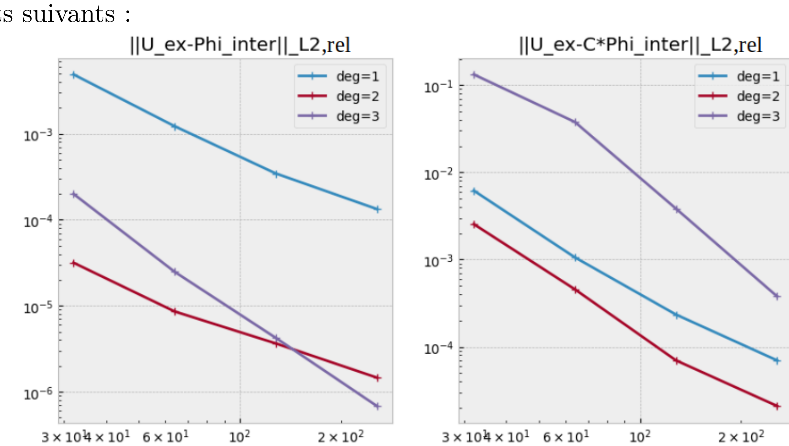
On considère cette fois-ci f gaussienne :

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2+(y-\mu_1)^2}{2\sigma^2}\right),$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5-\sqrt{2}/4, 0.5+\sqrt{2}/4])$ à condition que $\phi(\mu_0,\mu_1) < -0.05$.

Dans un premier temps, on considère la solution de référence u_{ex} comme étant une solution sur-raffinée obtenue par les EF standard (avec $h_{ex} \approx 0.006$ car $h_{ex} << h_{FNO}$). On interpole cette solution exacte (notre nouvelle level-set que l'on note $I_h u_{ex}$) dans $\mathbb{P}^k$ avec $k \in \{1,2,3\}$ et on fixe $C \in \mathbb{P}^1$.

On obtient les résultats suivants :



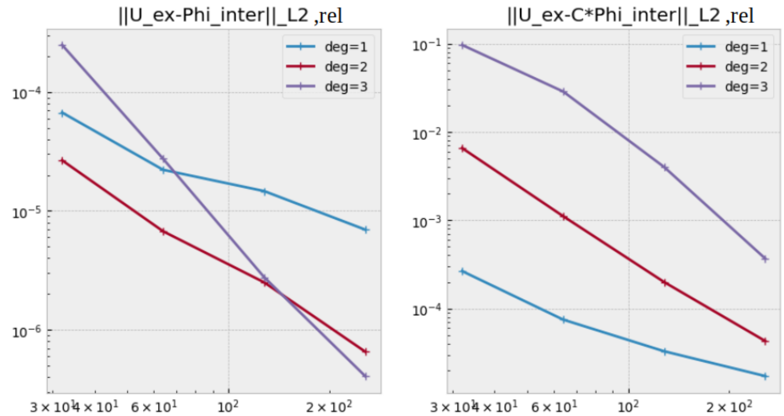
On obtient

$$||u_{ex} - I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}, \quad ||u_{ex} - C I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}$$

Cependant, il semblerait qu'il y ait un problème en $\mathbb{P}^3$ car l'erreur est plus grande qu'avec $k = 1, 2$. De plus la différence des erreurs en $\mathbb{P}^1$ et en $\mathbb{P}^2$ est très petite.

Dans un second temps, on considère toujours la solution de référence u_{ex} comme étant une solution sur-raffinée obtenue par les EF standard (avec $h \leq 0.006$). On interpole cette fois-ci cette solution exacte (notre nouvelle level-set que l'on note $I_h u_{ex}$) dans $\mathbb{P}^{k+1}$ avec $k \in \{1, 2, 3\}$ et on prend $C \in \mathbb{P}^k$.

On obtient les résultats suivants :



On obtient

$$||u_{ex} - I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}, \quad ||u_{ex} - C I_h u_{ex}||_{L^2(\Omega_h),rel} \sim h^{k+1}$$

Cependant, il semblerait qu'il y ait un problème car l'erreur $\mathbb{P}^1$ est meilleure que l'erreur $\mathbb{P}^2$ qui elle-même est meilleure que l'erreur $\mathbb{P}^3$.

7.4 Nouvelle idée FNO : Entraînement en 32×32 $\mathbb{P}^2$

On commence par générer *nb_data* données en $\mathbb{P}^2$ avec $nb\_vert = 32$. Une première difficulté est de faire la conversion des résultats FEniCS en "image" Numpy. Ensuite, on entraîne le FNO avec ces données sur un certains nombres d'époques. On peut alors utiliser le FNO sur de nouvelles données (un échantillon test par exemple). On va ensuite corriger la sortie du FNO où une seconde difficulté est le passage de notre image Numpy (où chaque pixel représente la valeur de la solution en chacun des degré de liberté $\mathbb{P}^2$) à une Expression Fenics. Après la correction, on obtient la solution $\mathbb{P}^1$ (car $C \in \mathbb{P}^1$).

Conversion FEniCS->Numpy :

Cette conversion est effectuée lors de la génération des données. Le solveur PhiFEM nous fournit une expression FEniCS. Dans le cas $\mathbb{P}^1$, la fonction *compute_vertex_values* de FEniCS nous permet de récupérer la valeur de la solution aux nœuds de notre maillage. En $\mathbb{P}^2$, c'est un peu plus compliqué, il faudra récupérer la valeur en chaque degré de liberté manuellement. Pour cela, on commence par récupérer les coordonnées de nos degrés de liberté en utilisant la fonction *tabulate_dof_coordinates* de FEniCS. La complexité de cette méthode est que la numérotation FEniCS n'est pas la même que celle que l'on souhaiterais. C'est pour cela que l'on va devoir créer un mapping. Pour cela, on commence par récupérer le vecteur des coordonnées dont la première colonne contient x et la deuxième contient y. On va rajouter une colonne contenant une nouvelle indexation de nos coordonnée. Puis, on trie cette matrice selon les ordonnées puis selon les abscisses. On obtient alors les coordonnées triées de bas en haut et de gauche à droite. La colonne contenant les indices est alors notre mapping. On peut alors ordonner l'expression et on obtient le tableau Numpy.

Conversion Numpy->FEniCS :

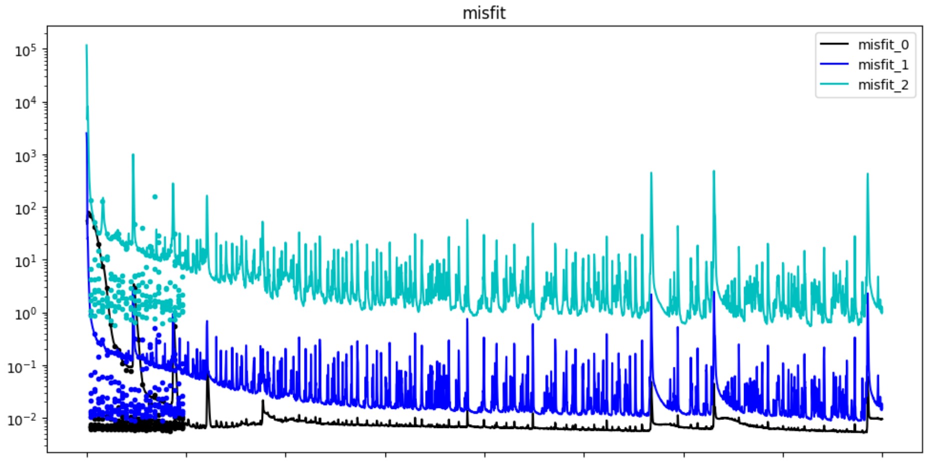
Cette conversion est utilisée pour convertir notre sortie de FNO en une fonction FEniCS pour la Correction. De la même manière, que pour la conversion dans le sens inverse, on crée un mapping qui nous permet d'associer chaque degré de liberté FEniCS aux valeurs Numpy.

7.4.1 Solution analytique trigonométrique

On considère encore la solution analytique trigonométrique suivante :

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2 + (y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2 + (y-0.5)^2\right)\right),$$

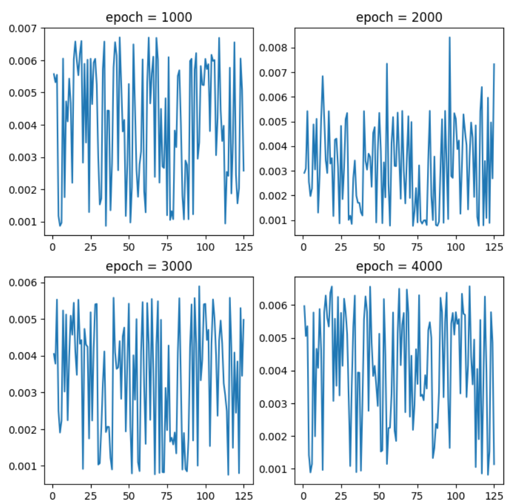
Après entraînement sur 4000 époques voici les misfits obtenus :

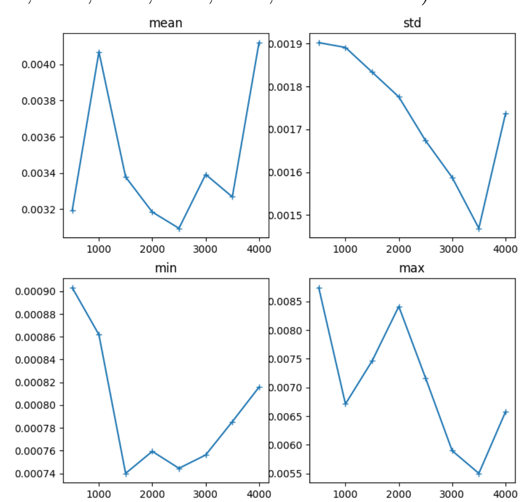


$\boxed{\text{A RELANCER EN UNE TRAITE !}}$

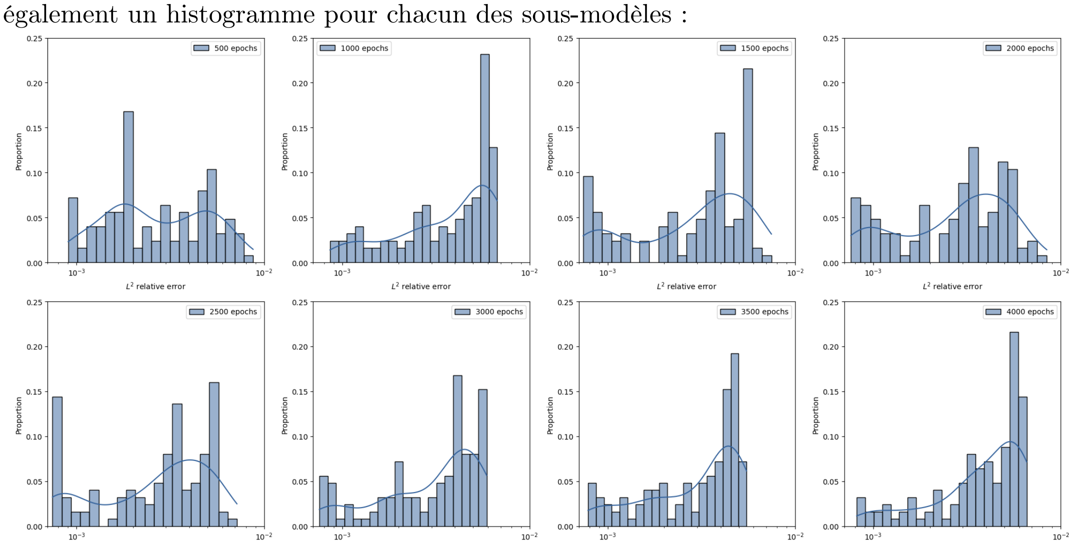
Échantillon de validation :

On commence par calculer les erreurs en norme L^2 sur l'échantillon de validation. On considérera des "sous-modèle" qui seront sauvegarder toutes les 500 époques. A gauche, on a les erreurs sur l'échantillon de validations toutes les 1000 époques. A droite, il y a la moyenne, l'écart-type, le minimum et le maximum de l'erreur sur l'échantillon de validation pour chacun des sous-modèle (500,1000,1500,2000,2500,3000,3500 et 4000).

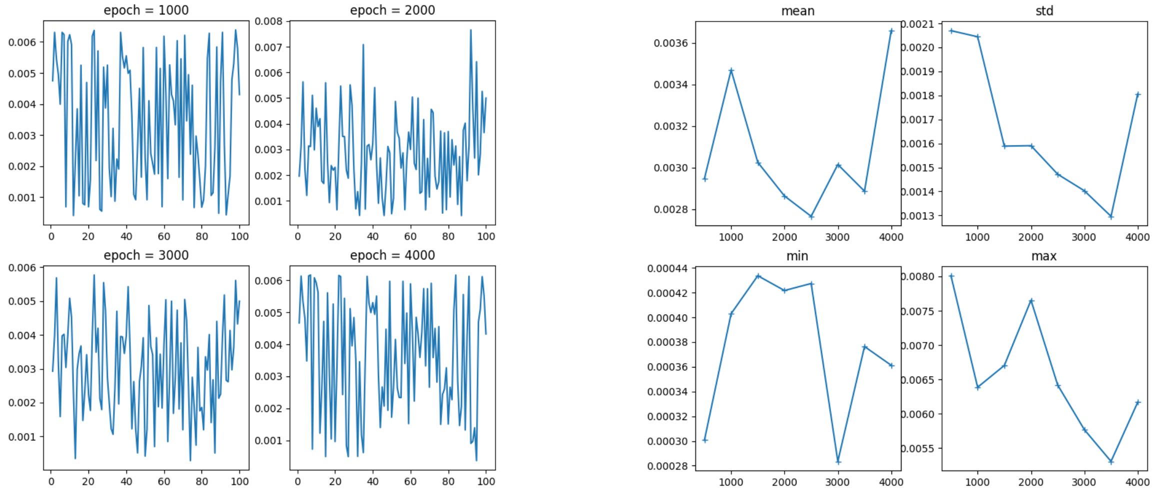


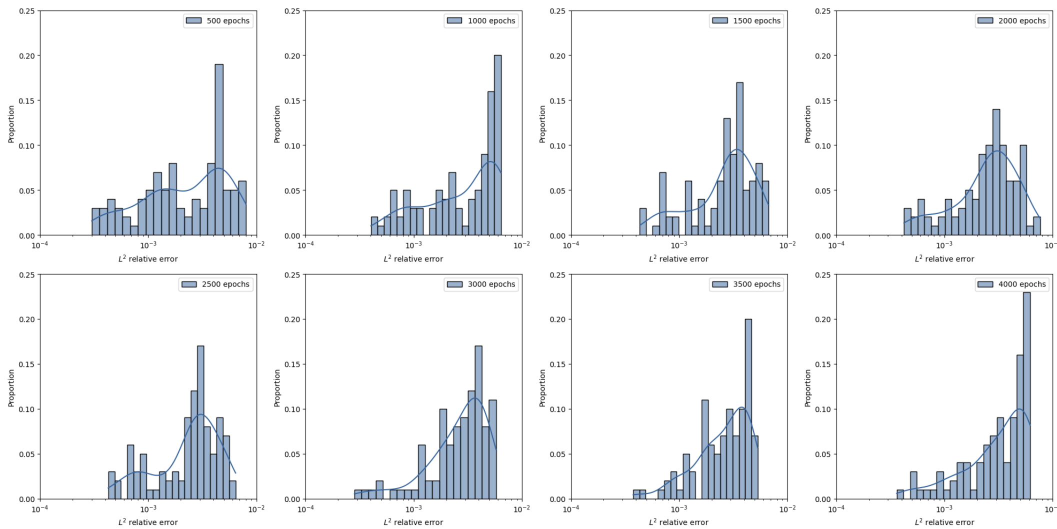


On réalise également un histogramme pour chacun des sous-modèles :



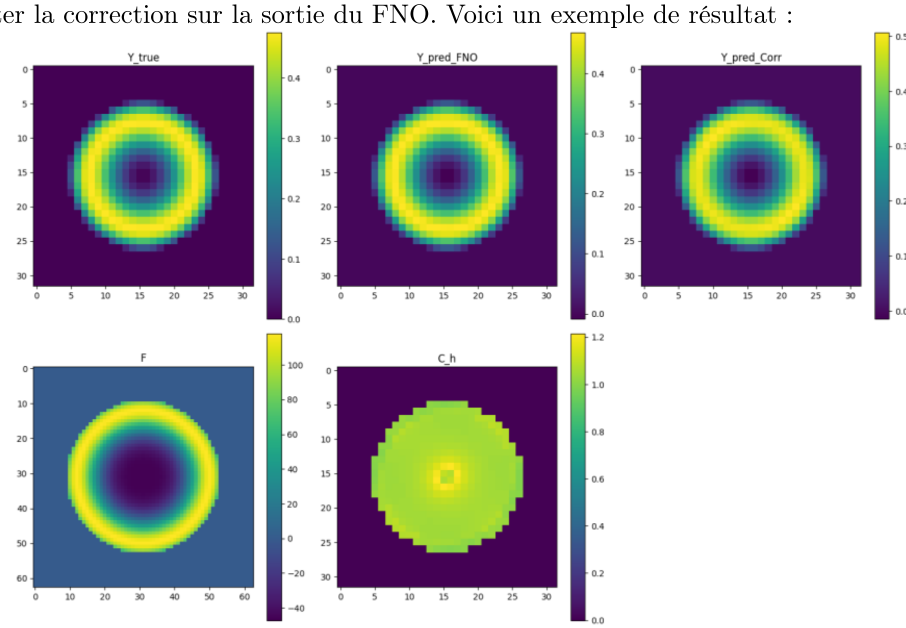
Échantillon test : On s'intéresse exactement aux même résultats mais cette fois-ci sur un nouvel échantillon (un échantillon test de taille 100) :





Résultat après correction :

On cherche à tester la correction sur la sortie du FNO. Voici un exemple de résultat :




```
||Y_true - Y_pred_FNO||_L2 :  [0.00522729]
||Y_true - Y_pred_Corr||_L2 :  [0.01061496]
```

On pense que les données précédentes ne varient pas assez et qu'elles sont donc trop facile à apprendre par le FNO. En effet, au bout de déjà 500 époques les erreurs semblent très bonnes. Ainsi la correction a du mal à être meilleure que la sortie du FNO. On va donc considérer un nouveau problème plus compliqué. On considère toujours le problème de poisson avec condition de Dirichlet homogène. On prend f gaussienne et notre solution de référence sera une solution sur-raffinée obtenue par les éléments finis standard.

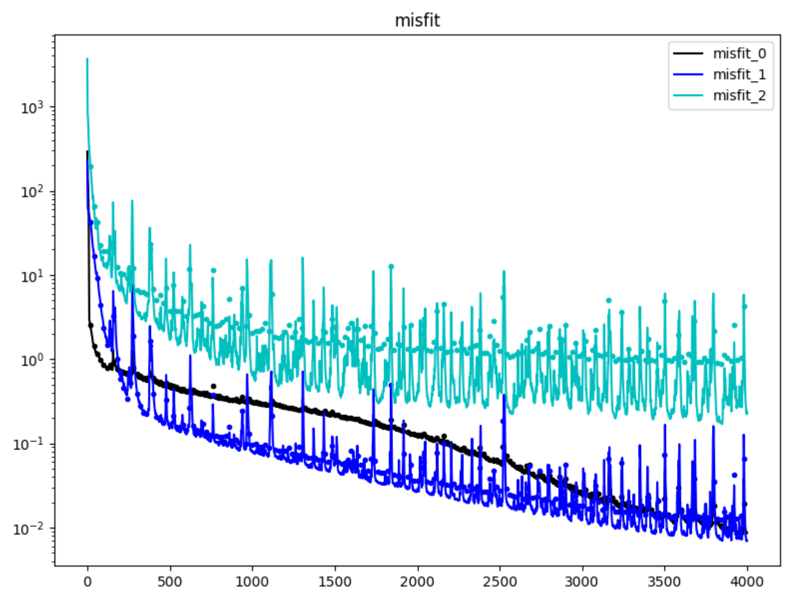
7.4.2 f gaussienne

On considère cette fois-ci f gaussienne :

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2+(y-\mu_1)^2}{2\sigma^2}\right),$$

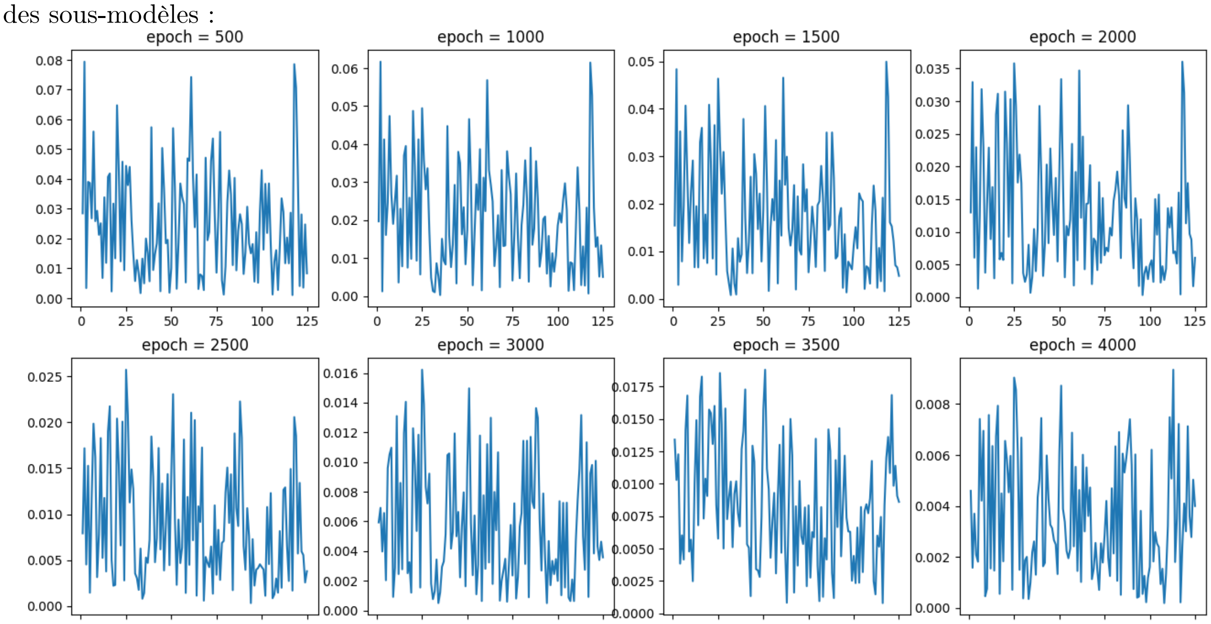
avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5-\sqrt{2}/4, 0.5+\sqrt{2}/4])$ à condition que $\phi(\mu_0,\mu_1) < -0.05$.

Après entraînement sur 4000 époques voici les misfits obtenus :

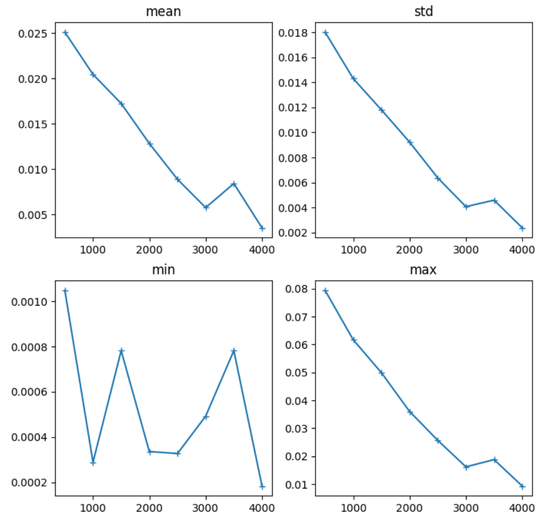


Échantillon de validation :

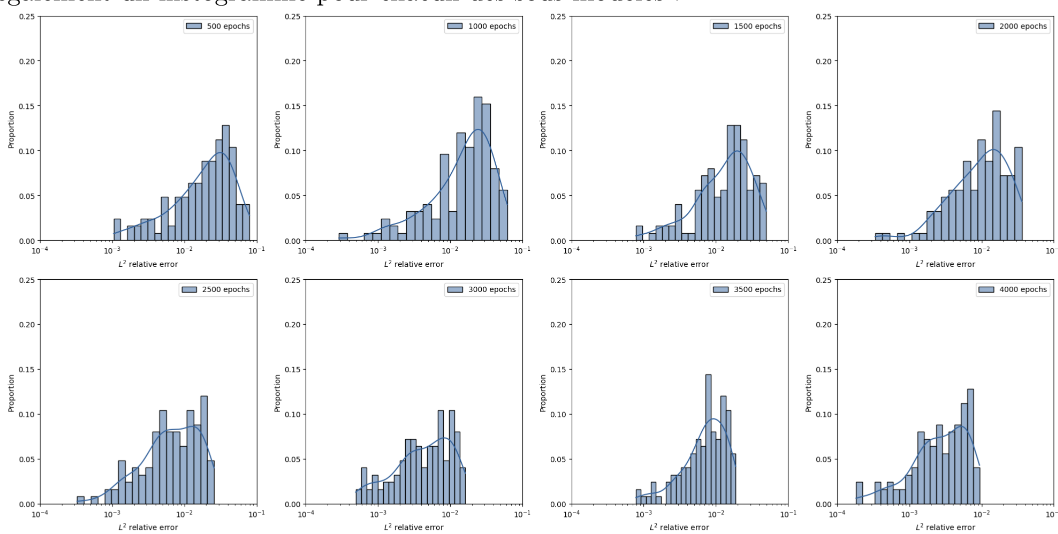
On commence par calculer les erreurs en norme L^2 sur l'échantillon de validation. On considérera des "sous-modèle" qui seront sauvegarder toutes les 500 époques. Voici les erreurs obtenus sur l'échantillon de validation pour chacun des sous-modèles :



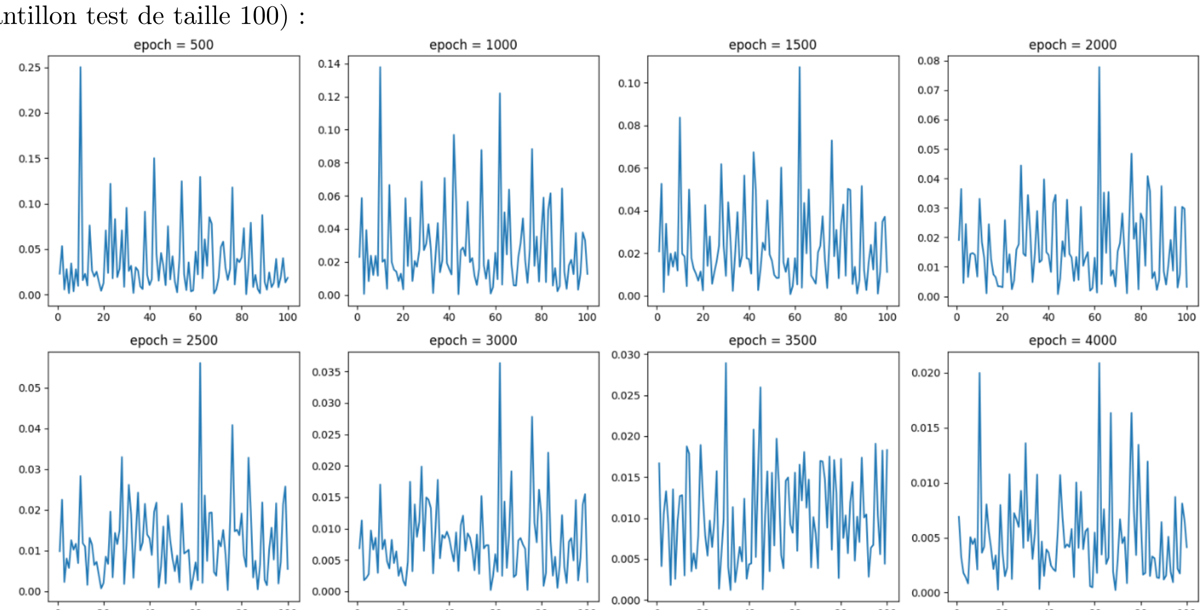
Voici la moyenne, l'écart-type, le minimum et le maximum de l'erreur sur l'échantillon de validation pour chacun des sous-modèle (500,1000,1500,2000,2500,3000,3500 et 4000) :

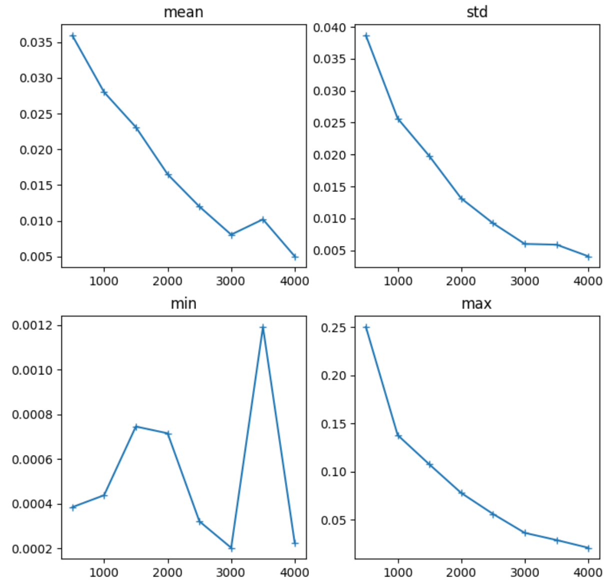


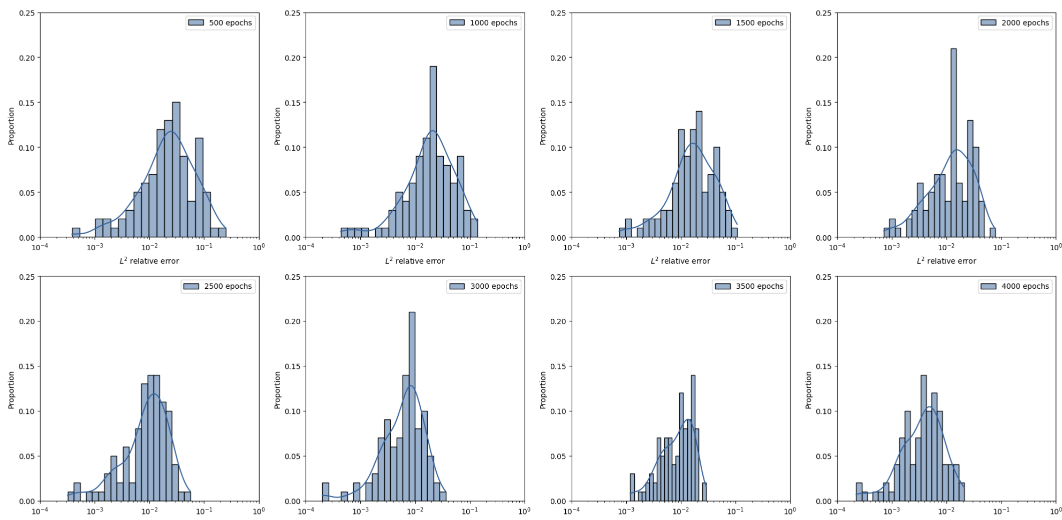
On réalise également un histogramme pour chacun des sous-modèles :



Échantillon test : On s'intéresse exactement aux même résultats mais cette fois-ci sur un nouvel échantillon (un échantillon test de taille 100) :

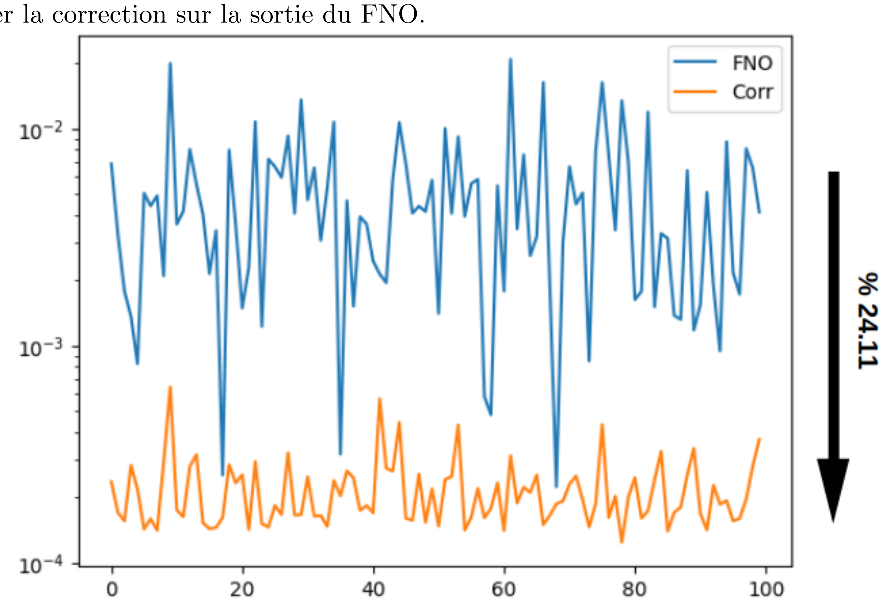






Résultat après correction :

On cherche à tester la correction sur la sortie du FNO.



8 Semaine 8 : 27/03/2023 - 31/03/2023

Résumé

L'idée principale de la semaine et de tester de rehausser la solution. Deux méthodes ont été proposées : rehausser par une constante m (proposée par Emmanuel) ou rehausser par la level-set initial ϕ (proposée par Michel).

J'ai également essayé de faire des boxplots pour comparer FEM standard, ϕ-FEM, FNO et FNO+Corr (pour différents nombres d'époques).

8.1 Rehaussement

On considère toujours le problème initial

$$\begin{cases} -\Delta u = f & \Omega \\ u = 0 & \Gamma \end{cases}$$

On prend encore la solution analytique trigonométrique suivante :

$$u_{ex}(x,y) = \frac{1}{\sin\left(k_1 \frac{\pi}{2}\right)} \times \sin\left(k_1 \frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2 + (y-0.5)^2\right)\right) \times \cos\left(\frac{\pi}{2}\left(\frac{4}{\sqrt{2}}\right)^2 \left((x-0.5)^2 + (y-0.5)^2\right)\right)$$

avec $k_1 \sim \mathcal{U}([0.1, 1])$.

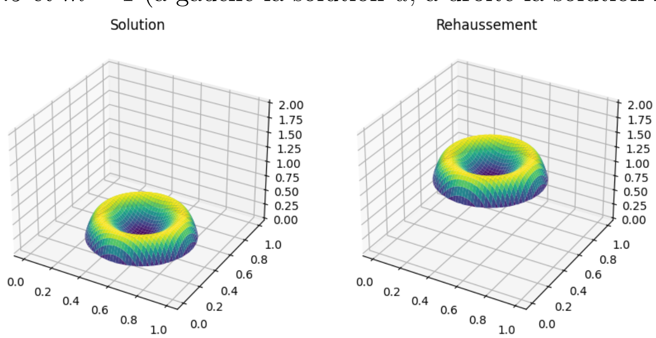
Par une constante

On considère la solution rehaussée par m :

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

avec m une constante.

Voici la solution pour $k_1 = 0.5$ et $m = 1$ (à gauche la solution u, à droite la solution rehaussée $\tilde{\phi}$) :



On réécrit alors le problème par

$$\begin{cases} -\Delta(\tilde{\phi} - m) = f & \Omega \\ \tilde{\phi} = m & \Gamma \end{cases}$$

Donc

$$\begin{cases} -\Delta\tilde{\phi} = f & \Omega \\ \tilde{\phi} = m & \Gamma \end{cases}$$

On veut appliquer la correction, on pose alors $\tilde{u} = \tilde{\phi}C$ avec $\tilde{\phi}$ la level-set.

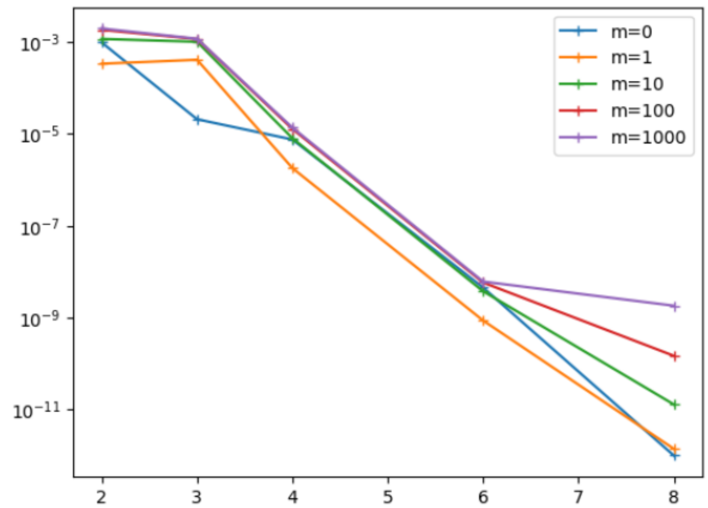
On a

$$\begin{cases} -\Delta\tilde{u} = f & \Omega \\ \tilde{u} = m & \Gamma \end{cases}$$

On se ramène alors à la correction du problème initial en posant

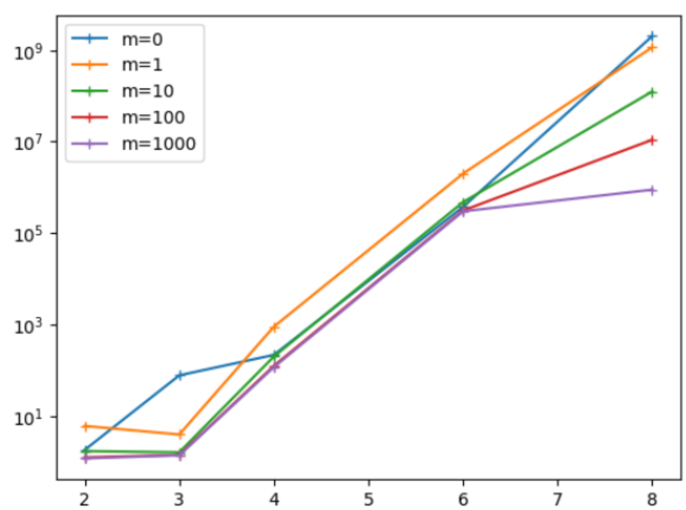
$$u_c = \tilde{\phi}C - m$$

On prend nb_data=10. On cherche à comparer la moyenne des erreurs PhiFEM sur les nb_data données avec la Correction pour différents degré d'intepolation de la levelset et avec la Correction par rehaussement pour différents degré d'interpolation et différents m. Voici les résultats obtenus :




```
degré :  [2, 3, 4, 6, 8]

Moyenne des erreurs L2 pour nb_data=10 :

PhiFEM :  0.0015873927273033282
Inter :  [6.85861612e-05 1.35160548e-06 2.45462225e-08 6.37926110e-12 6.29531285e-15]
Corr :  [9.68218587e-04 2.06743938e-05 7.49024943e-06 4.48122324e-09 9.89821122e-13]

Rehaussement :
m =  1  :  [3.34660431e-04 4.07822986e-04 1.78652616e-06 8.59018250e-10 1.38554186e-12]
m =  10  :  [1.15914758e-03 1.00680000e-03 7.98022826e-06 3.73253986e-09 1.28146952e-11]
m =  100  :  [1.82749310e-03 1.15130698e-03 1.27085003e-05 5.77658051e-09 1.46714047e-10]
m =  1000  :  [1.95365731e-03 1.16916526e-03 1.35870515e-05 6.05634042e-09 1.80513593e-09]
```

Et les facteurs obtenus :



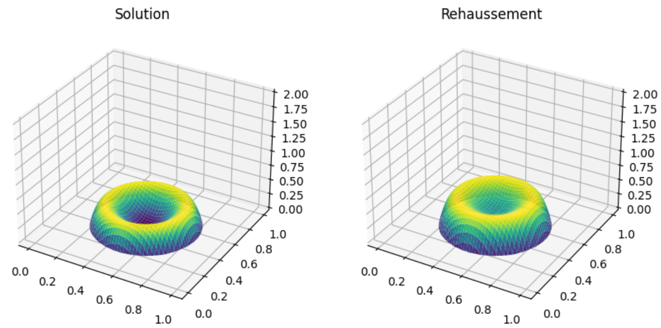

```
degré :  [2, 3, 4, 6, 8]

Facteurs entre PhiFEM et - :

Corr :  [1.80587121e+00 7.71956552e+01 2.15961523e+02 3.71349197e+05 2.01596384e+09]

Rehaussement :
m =  1  :  [6.04593538e+00 3.93079230e+00 8.95708589e+02 1.98707264e+06 1.14901703e+09]
m =  10  :  [1.70401921e+00 1.60235629e+00 1.98869249e+02 4.73817317e+05 1.24076403e+08]
m =  100  :  [1.24032009e+00 1.40243295e+00 1.24734589e+02 3.10563403e+05 1.08974158e+07]
m =  1000  :  [1.18131458e+00 1.38108163e+00 1.16659385e+02 2.97921007e+05 8.84913250e+05]
```

Par la level-set ϕ

On considère la solution rehaussée par une constante multipliée par ϕ :

$$\tilde{\phi} = u - \alpha\phi$$

avec α une constante positive.
Voici la solution pour $k_1 = 0.5$ et $\alpha = 2$ (à gauche la solution u, à droite la solution rehaussée $\tilde{\phi}$) :



On réécrit alors le problème par

$$\begin{cases} -\Delta(\tilde{\phi} + \alpha\phi) = f & \Omega \\ \tilde{\phi} = 0 & \Gamma \end{cases}$$

Donc

$$\begin{cases} -\Delta\tilde{\phi} = \tilde{f} & \Omega \\ \tilde{\phi} = 0 & \Gamma \end{cases}$$

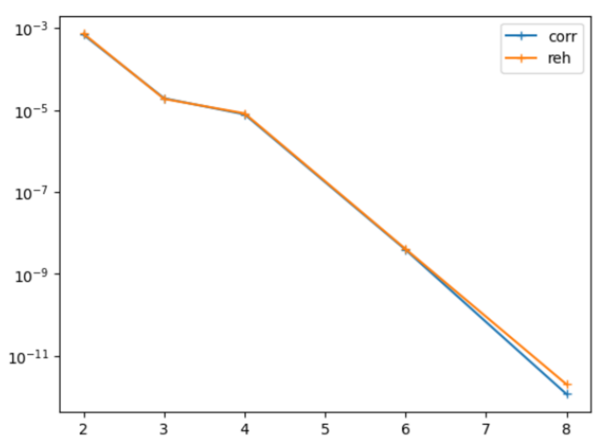
avec $\tilde{f} = f + \alpha\Delta\phi$.
On veut appliquer la correction, on pose alors $\tilde{u} = \tilde{\phi}C$ avec $\tilde{\phi}$ la level-set.
On a

$$\begin{cases} -\Delta\tilde{u} = \tilde{f} & \Omega \\ \tilde{u} = 0 & \Gamma \end{cases}$$

On se ramène alors à la correction du problème initial en posant

$$u_c = \tilde{\phi}C + \alpha\phi$$

Voici les résultats obtenus :




```
degré :  [2, 3, 4, 6, 8]

Moyenne des erreurs L2 :

PhiFEM :  0.0015145849148183242
Inter :  [7.02513795e-05 1.28686357e-06 2.43868839e-08 5.53322372e-12 6.30631390e-15]
Corr :  [6.86856780e-04 1.93371524e-05 7.87957409e-06 3.91170644e-09 1.17201576e-12]
Rehaussement :  [7.27581723e-04 1.87422781e-05 8.29325966e-06 4.07358125e-09 2.03008432e-12]
```

8.2 Résultats avec le FNO

On considère cette fois-ci f gaussienne :

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2+(y-\mu_1)^2}{2\sigma^2}\right),$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5-\sqrt{2}/4, 0.5+\sqrt{2}/4])$ à condition que $\phi(\mu_0,\mu_1) < -0.05$.

Dans un premier temps, on considère la solution de référence u_{ref} comme étant une solution sur-raffinée $\mathbb{P}^1$ obtenue par les EF standard (avec $h_{ref} \approx 0.006$ car $h_{ref} << h_{FNO}$).

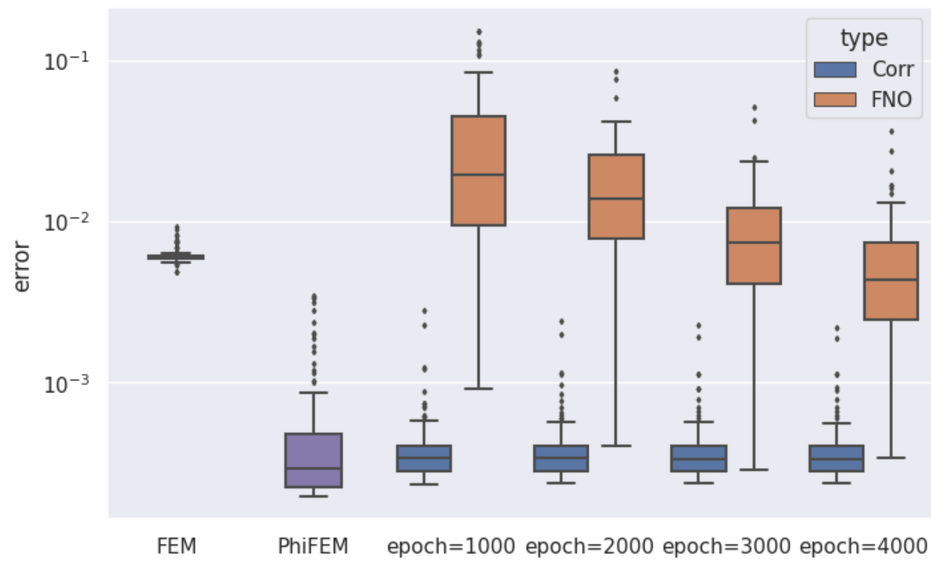
On cherche à afficher des boxplots (boite à moustache) sur les erreurs en norme L^2 pour FEM standard, PhiFEM, le FNO et le FNO corrigé pour différents nombres d'époques. On prendra un nouvel échantillon test avec nb_data=100.

On va donc comparer

$$||u_{ref} - u_{FEM}||_{L^2,rel}, \quad ||u_{ref} - u_{\phi-FEM}||_{L^2,rel}, \quad ||u_{ref} - u_{FNO}||_{L^2,rel}, \quad ||u_{ref} - Cu_{FNO}||_{L^2,rel}$$

où u_{FEM} est la solution $\mathbb{P}^1$ obtenue par la méthode des éléments finis standard avec une taille de maillage comparable à celle utilisée pour le FNO, $u_{\phi-FEM}$ est la solution $\mathbb{P}^1$ obtenue par ϕ-FEM, u_{FNO} est la solution $\mathbb{P}^2$ obtenue par le FNO et C est la correction $\mathbb{P}^1$ obtenue en prenant comme level-set u_{FNO}.

Voici les résultats obtenus :



Conclusion

Les résultats pour le rehaussement n'ont pas l'air bon : il faudra en parler la semaine prochaine avec Emmanuel et Michel.

Pour la partie avec le FNO, la semaine prochaine il faudrait comparer les temps d'exécution entre PhiFEM et FNO+Corr (ratio temps/erreur ou erreur/temps).

9 Semaine 9 : 03/04/2023 - 07/04/2023

Résumé

Après les résultats obtenus avec le FNO, on va essayer de comparer les temps d'exécution et les erreurs obtenus pour FEM, Phi-FEM, le FNO et le FNO+corr à différentes époques.

Après discussion avec Emmanuel, on va considérer une level-set du type $\tilde{\phi} = u_{ex} + \epsilon * P$. On va tester le rehaussement avec FEM puis avec PhiFEM pour différents m sur cette solution perturbée.

(Vendredi est férié)

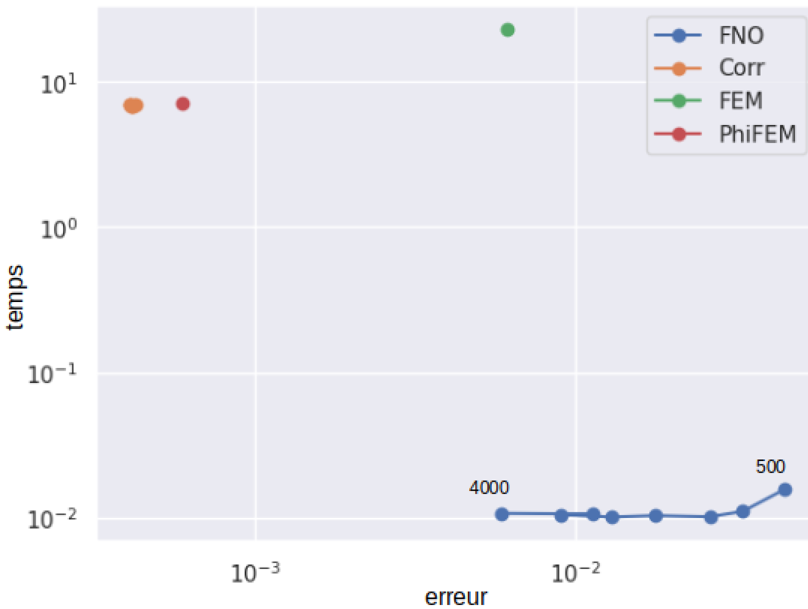
9.1 Temps d'exécution (avec FNO)

On considère, comme pour les boxplots de la semaine dernière, f gaussienne :

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2 + (y-\mu_1)^2}{2\sigma^2}\right),$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5 - \sqrt{2}/4, 0.5 + \sqrt{2}/4])$ à condition que $\phi(\mu_0, \mu_1) < -0.05$.

Voici les résultats obtenus :




```
time_FNO :  [0.01556826 0.01101136 0.01010323 0.01029396 0.0100584  0.01044846 0.01055074 0.01066899]
time_Corr :  [6.96440387 7.03112602 6.89074326 6.87651873 6.84414625 6.97967672 6.94006777 6.97081423]
time_FEM :  22.981932878494263
time_PhiFEM :  7.10906720161438

errors_FNO :  [0.04441629 0.03298442 0.02606437 0.01761629 0.01285544 0.00901223 0.01128896 0.00587092]
errors_Corr :  [0.00042691 0.00042341 0.00041927 0.000415   0.00041312 0.00041249 0.00041029 0.00041118]
errors_FEM :  0.006137759508301814
errors_PhiFEM :  0.0005956599945233883
```

9.2 Rehaussement avec FEM

On se place sur le carré $[0,1]^2$.
On souhaite résoudre le problème de Poisson avec condition de Dirichlet non homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = g & \Gamma \end{cases}$$

On considère la solution analytique suivante :

$$u_{ex}(x,y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

S est l'amplitude du signal, f la fréquence du signal et φ la phase à l'origine.
On pose alors

$$f(x,y) = 8\pi^2 S f^2 \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi), \quad g(x,y) = u_{ex}(x,y)$$

On considère qu'après une utilisation du FNO, on obtient une solution du type

$$u_p = u_{ex} + \epsilon P(x,y)$$

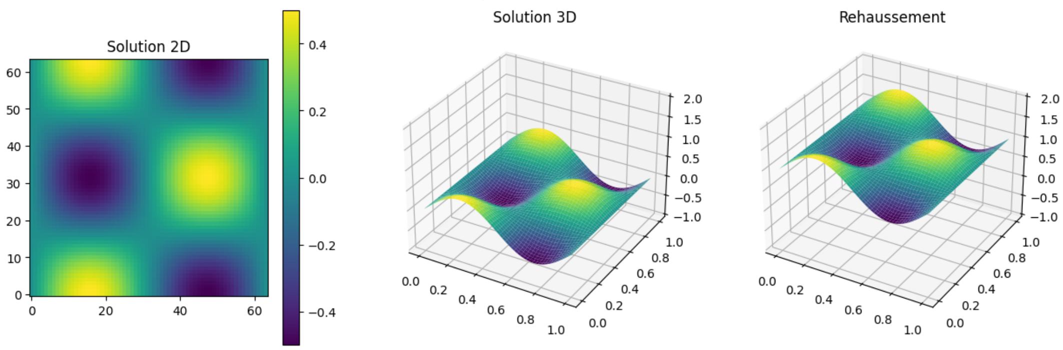
avec ϵ petit et P la perturbation définie par

$$P(x,y) = S_p \times \sin(2\pi f_p x + \varphi_p) \times \sin(2\pi f_p y + \varphi_p)$$

On considère alors la solution rehaussée par m :

$$\tilde{\phi} = u_p + m$$

avec m une constante.
Voici la solution pour $S = 0.5$, $f = 1$, $\varphi = 0$ et $m = 1$ (à gauche la solution u en 2d, au milieu la solution u en 3D et à droite la solution exacte rehaussée $u + m$ en 3D) :



On se ramène alors on problème

$$\begin{cases} -\Delta(u_p C) = f & \Omega \\ C = 1 & \Gamma \end{cases}$$

Et donc

$$\begin{cases} -\Delta(\tilde{\phi} C) = f & \Omega \\ C = 1 & \Gamma \end{cases}$$

On obtient alors

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

Et donc

$$u_C = \tilde{\phi} C - m$$

On obtient alors la formulation variationnelle suivante (avec comme fonction test $\tilde{\phi} v$) :

$$\int_\Omega \nabla(\tilde{\phi} C) \cdot \nabla(\tilde{\phi} v) = \int_\Omega f \tilde{\phi} v$$

Remarque. *Attention u et u_p doivent avoir les mêmes conditions aux bords donc P doit être nulle au bord. On prendra 10 comme degré de quadrature et 10 comme degré d'intepolation.*

Test 1 : $g = 0$

On prend $S, S_p = 0.5$, $\epsilon = 10^{-3}$ et $\varphi, \varphi_p = 0$. Ainsi $g = 0$ sur Γ.
Voici les résultats obtenus en faisant varier f, f_p et m (à gauche les erreurs en norme L^2, à droite les facteurs avec FEM) :

	FEM	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	0.011958	0.000032	0.000050	0.000013	0.000012	0.000012
f=4 , fp=2	0.047539	0.000878	0.000036	0.000012	0.000012	0.000012
f=6 , fp=2	0.103958	0.000947	0.000087	0.000013	0.000012	0.000012
f=8 , fp=2	0.176819	0.000972	0.000186	0.000014	0.000012	0.000012
f=2 , fp=4	0.011958	0.000056	0.000059	0.000048	0.000048	0.000048
f=4 , fp=4	0.047539	0.000046	0.000593	0.000070	0.000048	0.000048
f=6 , fp=4	0.103958	0.000824	0.000085	0.000048	0.000048	0.000048
f=8 , fp=4	0.176819	0.000899	0.000120	0.000048	0.000048	0.000048
f=2 , fp=6	0.011958	0.000100	0.000108	0.000104	0.000104	0.000104
f=4 , fp=6	0.047539	0.000714	0.000151	0.000104	0.000104	0.000104
f=6 , fp=6	0.103958	0.000057	0.001678	0.000268	0.000107	0.000104
f=8 , fp=6	0.176819	0.000810	0.000183	0.000105	0.000104	0.000104
f=2 , fp=8	0.011958	0.000166	0.000179	0.000177	0.000177	0.000177
f=4 , fp=8	0.047539	0.000113	0.000365	0.000177	0.000177	0.000177
f=6 , fp=8	0.103958	0.000739	0.000338	0.000179	0.000177	0.000177
f=8 , fp=8	0.176819	0.000066	0.002511	0.000758	0.000192	0.000177

	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	372.411086	237.086766	953.321916	998.527278	999.887492
f=4 , fp=2	54.163986	1313.159982	3902.587042	3974.125289	3975.472738
f=6 , fp=2	109.810333	1193.065886	8166.664616	8688.294588	8693.679712
f=8 , fp=2	181.843203	948.868056	12601.778690	14760.578465	14786.546488
f=2 , fp=4	214.770822	201.191150	251.310633	251.530848	251.536658
f=4 , fp=4	1044.480429	80.123509	681.428836	993.484765	999.851238
f=6 , fp=4	126.199975	1217.335781	2173.378142	2186.622568	2186.785326
f=8 , fp=4	196.765333	1473.875790	3664.247747	3718.367537	3719.363626
f=2 , fp=6	119.006821	110.434417	115.004705	115.025190	115.025683
f=4 , fp=6	66.627007	314.377864	455.517624	457.264320	457.290006
f=6 , fp=6	1835.390795	61.944099	387.511328	971.851341	999.624184
f=8 , fp=6	218.177004	964.053534	1689.687439	1700.711541	1700.855000
f=2 , fp=8	72.224401	66.968845	67.622143	67.627823	67.627962
f=4 , fp=8	422.017905	130.221914	268.613626	268.855900	268.858662
f=6 , fp=8	140.657846	307.320364	580.703766	587.851448	587.935689
f=8 , fp=8	2665.337366	70.412995	233.275836	919.568937	999.002878

On prend toujours $S, S_p = 0.5$, $\varphi, \varphi_p = 0$. On fixe cette fois-ci $f = 8$ et $f_p = 3$ et on prend $\epsilon = 10^{-4}$.
On obtient les résultats suivants :

	FEM	Corr	m=1	m=10	m=100	m=1000
erreurs	0.176819	0.000094	0.000012	0.000003	0.000003	0.000003
facteurs	NaN	1881.268027	14165.933652	63284.082873	65648.180268	65671.992163

Remarque. *On a le même type de résultat avec $S = 1$ et $S_p = 0.25$.*

Test 2 : $g \neq 0$

On prend $S, S_p = 0.5$, $\epsilon = 10^{-3}$, $\varphi = 0.25$ et $\varphi_p = 0$.
Voici les résultats obtenus en faisant varier f, f_p et m (à gauche les erreurs en norme L^2, à droite les facteurs avec FEM) :

	FEM	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	0.011924	0.000252	0.000046	0.000012	0.000012	0.000012
f=6 , fp=2	0.103889	0.000947	0.000087	0.000013	0.000012	0.000012
f=2 , fp=6	0.011924	0.000488	0.000106	0.000104	0.000104	0.000104
f=6 , fp=6	0.103889	0.000278	0.001525	0.000246	0.000106	0.000104

	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	47.262514	257.995536	957.889396	995.910553	997.098834
f=6 , fp=2	109.740225	1192.276850	8161.032017	8682.496127	8687.900006
f=2 , fp=6	24.428833	112.210148	114.683082	114.703057	114.703523
f=6 , fp=6	373.960118	68.136169	423.128584	976.332107	999.023591

9.3 Rehaussement avec PhiFEM

On souhaite effectuer le même type de tests avec PhiFEM.
On se place encore sur le carré $[0, 1]^2$. On prend alors

$$\phi(x, y) = ||x - 0.5||_\infty - 0.5$$

On considérera le domaine environnant $\mathcal{O} = [-0.5, 1.5]^2$.
On considère encore la solution analytique suivante :

$$u_{ex}(x, y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

et pour $p = 0$, $g(x, y) = 0$.
De la même manière que pour FEM, on va considérer la solution rehaussée par m :

$$\tilde{\phi} = u_p + m$$

Test

Comme pour le Test 1 de FEM, on prend $S, S_p = 0.5$, $\epsilon = 10^{-2}$ et $\varphi, \varphi_p = 0$. Ainsi $g = 0$ sur Γ.
Voici les résultats obtenus en faisant varier f, f_p et m (à gauche les erreurs en norme L^2, à droite les facteurs avec PhiFEM) :

	PhiFEM	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	0.015834	9.159368e-13	5.970404e-10	2.815364e-09	5.288582e-08	3.941090e-07
f=6 , fp=2	0.109548	9.456905e-03	1.109036e-02	1.019081e-02	1.014757e-02	1.014110e-02
f=2 , fp=6	0.015834	6.610428e-04	1.991425e-02	4.450450e-02	4.948660e-02	4.994433e-02
f=6 , fp=6	0.109548	3.427688e-09	1.433677e-08	8.365243e-09	1.177021e-08	6.346846e-08

	Corr	m=1	m=10	m=100	m=1000
f=2 , fp=2	1.728750e+10	2.652125e+07	5.624232e+06	2.994046e+05	4.017735e+04
f=6 , fp=2	1.158395e+01	9.877801e+00	1.074972e+01	1.079552e+01	1.080241e+01
f=2 , fp=6	2.395345e+01	7.951220e-01	3.557900e-01	3.199706e-01	3.170381e-01
f=6 , fp=6	3.195983e+07	7.641075e+06	1.309566e+07	9.307252e+06	1.726028e+06

Conclusion

Il semblerait qu'il y ait un problème avec PhiFEM. On va donc de nouveau mettre le FNO de côté.

10 Semaine 10 : 10/04/2023 - 14/04/2023

Résumé

(Lundi est férié)

Après discussion avec Michel et Emmanuel mardi matin, on a discuté de quelques points qu'il faudrait traité suite aux résultats obtenus avec PhiFEM :

- On garde la fonction $\phi_c(x,y) = ||x-0.5||_\infty - 0.5$ pour construire les ensembles utilisés par PhiFEM. On utilise la levelset suivante pour PhiFEM : $\phi(x,y) = x(1-x)y(1-y)$.
- On effectue les courbes de convergence de PhiFEM sur le carré et sur le cercle pour ce problème. On étudiera également les erreurs d'interpolation.
- On cherchera ensuite à comparer les résultats FEM et PhiFEM. En effet, la semaine dernière on a considéré le même problème pour les deux méthodes mais les résultats obtenus sont très différents.
- Pour finir, Michel a proposé une analyse pour la sortie du FNO : la décomposition en série de Fourier de la solution (ce qui nous permettrait de déterminer dans lequel des cas on se trouve, parmis ceux observés sur la solution analytique et ainsi d'avoir une idée de la forme de la perturbation en sortie du FNO). Problème : Je ne comprends pas ce qui nous garantit que l'on peut faire cette décomposition, la sortie n'est pas forcément périodique !

On a également remarqué qu'il faut penser à prendre un epsilon de tolérance pour la construction des espaces (car si la la levelset se situe exactement à l'intersection de deux cellules il peut y avoir des problèmes d'arrondis).

Dans la suite, on construira les ensembles $\mathcal{T}_h^\Gamma$ et $\mathcal{F}_h^\Gamma$ en utilisant la fonction :

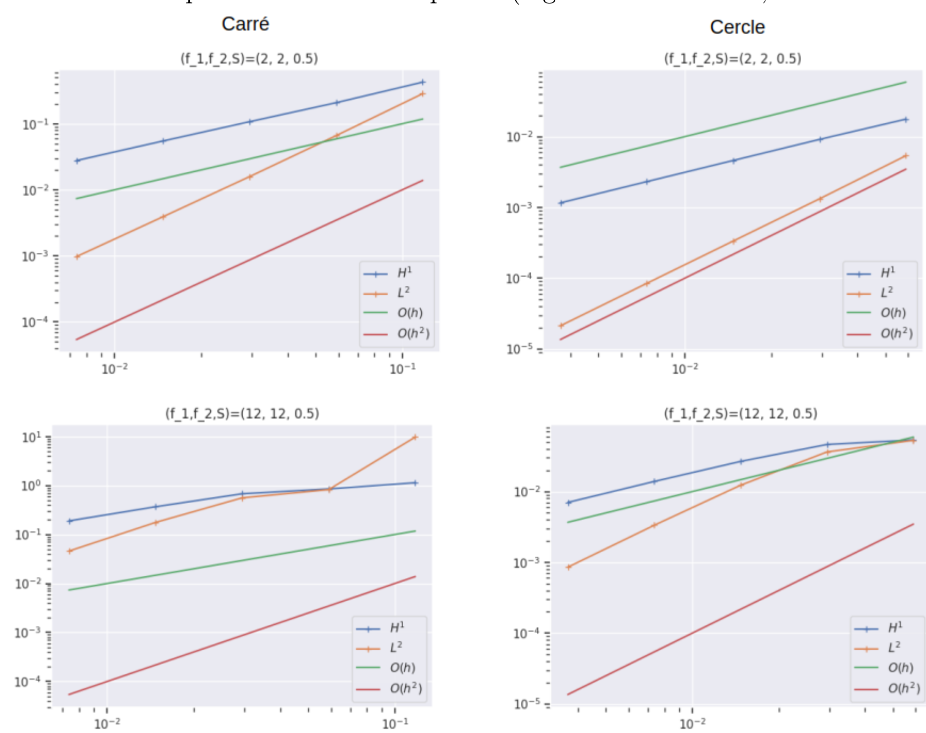
$$\phi_c(x,y) = ||x-0.5||_\infty - 0.5$$

Cependant, dans la résolution PhiFEM, on considérera :

$$\phi(x,y) = x(1-x)y(1-y)$$

10.1 Convergence PhiFEM

On obtient les résultats suivant pour deux cas de fréquence (à gauche sur le carré, à droite sur le cercle) :



10.2 Comparaison FEM-PhiFEM

Les résultats de la semaine dernière était plutôt étranges. C'est pour cela qu'on a raisonnée en deux parties : On commence par essayer de comprendre pourquoi la correction (sans rehaussement $m = 0$) fournit d'aussi bon résultat. Puis on s'intéressera au rehaussement.

Correction avec PhiFEM

Il semblerait que lorsque l'on prend $f = f_p$, PhiFEM fournit les mêmes résultats que si on lui fournissait la solution exacte comme levelset. Si $f = f_p$ alors $P = u_e x$ et donc

$$u_p = u_{ex} + \epsilon P = (1 + \epsilon) u_{ex}$$

Il semblerait que faire varier ϵ n'ait alors pas d'effet sur l'erreur L^2 PhiFEM contraiement à FEM standard.

Cependant, en prenant des fréquences différents pour la solution analytique et la perturbation, on obtient les résultats attendus, à savoir que plus ϵ est petit et meilleure est l'erreur.

Voici les résultats obtenus pour différents ϵ :

		eps=1.0	eps=0.1	eps=0.01	eps=0.001	eps=0.0
FEM	f=2, fp=2	5.440797e-02	5.440797e-03	5.440797e-04	5.440797e-05	2.460038e-13
	f=2, fp=4	6.720384e-01	1.091674e-02	1.045260e-03	1.045152e-04	2.460038e-13
PhiFEM	f=2, fp=2	1.509278e-10	1.509269e-10	1.509280e-10	1.509279e-10	1.509278e-10
	f=2, fp=4	6.870713e-01	7.943090e-03	6.873758e-04	6.866112e-05	1.509278e-10

Rehaussement

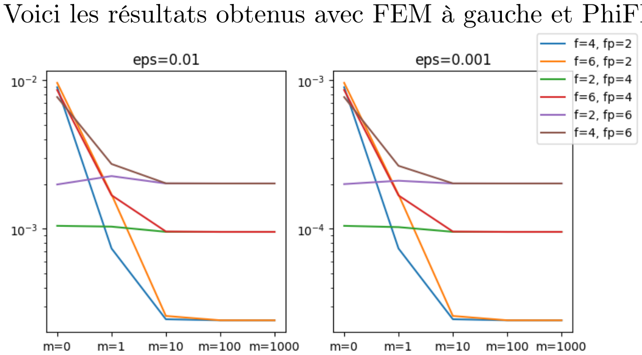
On cherche ici à comparer les erreurs FEM et PhiFEM lorsque l'on rehausse le problème. On testera pour différentes fréquences (avec $f \neq f_p$), différents rehaussement m ainsi que différents ϵ.

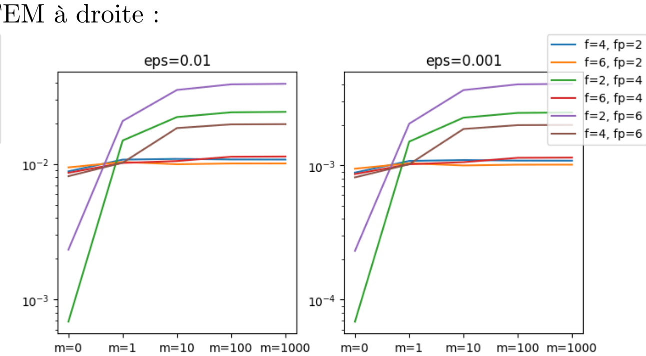
Voici les résultats numériques obtenus :

			FEM/PhiFEM	Corr	m=1	m=10	m=100	m=1000
FEM	f=4, fp=2	eps=0.01	0.095196	0.008953	0.000736	0.000246	0.000242	0.000242
		eps=0.001	0.095196	0.000895	0.000074	0.000025	0.000024	0.000024
	f=6, fp=2	eps=0.01	0.201494	0.009584	0.001701	0.000259	0.000242	0.000242
		eps=0.001	0.201494	0.000959	0.000170	0.000026	0.000024	0.000024
	f=2, fp=4	eps=0.01	0.024178	0.001045	0.001033	0.000953	0.000952	0.000952
		eps=0.001	0.024178	0.000105	0.000102	0.000095	0.000095	0.000095
	f=6, fp=4	eps=0.01	0.201494	0.008578	0.001679	0.000957	0.000952	0.000952
		eps=0.001	0.201494	0.000858	0.000167	0.000096	0.000095	0.000095
	f=2, fp=6	eps=0.01	0.024178	0.001985	0.002260	0.002015	0.002015	0.002015
		eps=0.001	0.024178	0.000200	0.000210	0.000201	0.000201	0.000201
	f=4, fp=6	eps=0.01	0.095196	0.007688	0.002725	0.002020	0.002015	0.002015
		eps=0.001	0.095196	0.000769	0.000265	0.000202	0.000201	0.000201
PhiFEM	f=4, fp=2	eps=0.01	0.161997	0.008844	0.010793	0.010926	0.010822	0.010807
		eps=0.001	0.161997	0.000884	0.001080	0.001097	0.001087	0.001085
	f=6, fp=2	eps=0.01	0.341761	0.009455	0.010373	0.009977	0.010110	0.010112
		eps=0.001	0.341761	0.000946	0.001037	0.001000	0.001013	0.001014
	f=2, fp=4	eps=0.01	0.035073	0.000687	0.014939	0.022258	0.024124	0.024310
		eps=0.001	0.035073	0.000069	0.001505	0.002269	0.002461	0.002481
	f=6, fp=4	eps=0.01	0.341761	0.008626	0.010205	0.010538	0.011340	0.011380
		eps=0.001	0.341761	0.000863	0.001021	0.001058	0.001140	0.001144
	f=2, fp=6	eps=0.01	0.035073	0.002334	0.020820	0.035286	0.038820	0.039172
		eps=0.001	0.035073	0.000231	0.002046	0.003640	0.004018	0.004056
	f=4, fp=6	eps=0.01	0.161997	0.008146	0.010241	0.018461	0.019655	0.019726
		eps=0.001	0.161997	0.000815	0.001021	0.001869	0.001996	0.002003

			Corr	m=1	m=10	m=100	m=1000
FEM	f=4, fp=2	eps=0.01	10.632588	129.268768	387.223595	393.748857	393.742152
		eps=0.001	106.315008	1293.647125	3872.070426	3937.472946	3937.419987
	f=6, fp=2	eps=0.01	21.024327	118.426204	778.838663	832.835214	833.388314
		eps=0.001	210.201187	1187.929347	7788.825049	8328.329635	8333.880044
	f=2, fp=4	eps=0.01	23.130747	23.415962	25.377976	25.397508	25.397846
		eps=0.001	231.331464	235.947955	253.776456	253.973450	253.978287
	f=6, fp=4	eps=0.01	23.489579	119.976882	210.472466	211.708833	211.669409
		eps=0.001	234.912870	1203.206236	2105.810805	2117.086604	2116.692744
	f=2, fp=6	eps=0.01	12.181702	10.700304	11.996710	11.999594	11.999237
		eps=0.001	121.116911	114.984299	120.003046	119.995268	119.992261
	f=4, fp=6	eps=0.01	12.381564	34.934546	47.135628	47.251756	47.245743
		eps=0.001	123.831430	359.222973	471.745833	472.517487	472.457027
PhiFEM	f=4, fp=2	eps=0.01	18.316687	15.009515	14.826504	14.968608	14.989810
		eps=0.001	183.165374	149.995930	147.673054	149.090512	149.304431
	f=6, fp=2	eps=0.01	36.146991	32.946014	34.253728	33.803678	33.798343
		eps=0.001	361.453844	329.551955	341.869791	337.222122	337.171858
	f=2, fp=4	eps=0.01	51.025173	2.347820	1.575775	1.453875	1.442759
		eps=0.001	510.819897	23.302661	15.460967	14.249544	14.138847
	f=6, fp=4	eps=0.01	39.619802	33.487976	32.430948	30.136784	30.032409
		eps=0.001	396.180757	334.864495	323.028383	299.760695	298.709164
	f=2, fp=6	eps=0.01	15.026924	1.684592	0.993972	0.903491	0.895373
		eps=0.001	151.733542	17.142409	9.634366	8.729383	8.648069
	f=4, fp=6	eps=0.01	19.886431	15.818603	8.775022	8.242179	8.212396
		eps=0.001	198.837296	158.713128	86.658958	81.169034	80.863217

Voici les résultats obtenus avec FEM à gauche et PhiFEM à droite :





Variation des termes dans la formulation variationnelle

On cherche ici à comparer les erreurs PhiFEM avec tous les termes dans la formulation variationnelle, sans les termes de stabilisation et sans le terme de bord lorsque l'on rehausse le problème. On testera pour différentes fréquences (avec $f \neq f_p$), différents rehaussement m et on prendre $\epsilon = 1e - 2$.

Voici les résultats numériques obtenus :

Erreurs | Facteurs

Original

Erreurs :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	0.161997	0.008844	0.010793	0.010926	0.010822	0.010807
	f=6, fp=2	eps=0.01	0.341761	0.009455	0.010373	0.009977	0.010110	0.010112
	f=2, fp=4	eps=0.01	0.035073	0.000687	0.014939	0.022258	0.024124	0.024310
	f=6, fp=4	eps=0.01	0.341761	0.008626	0.010205	0.010538	0.011340	0.011380
	f=2, fp=6	eps=0.01	0.035073	0.002334	0.020820	0.035286	0.038820	0.039172
	f=4, fp=6	eps=0.01	0.161997	0.008146	0.010241	0.018461	0.019655	0.019726

Facteurs :

			Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	18.316687	15.009515	14.826504	14.968608	14.989810
	f=6, fp=2	eps=0.01	36.146991	32.946014	34.253728	33.803678	33.798343
	f=2, fp=4	eps=0.01	51.025173	2.347820	1.575775	1.453875	1.442759
	f=6, fp=4	eps=0.01	39.619802	33.487976	32.430948	30.136784	30.032409
	f=2, fp=6	eps=0.01	15.026924	1.684592	0.993972	0.903491	0.895373
	f=4, fp=6	eps=0.01	19.886431	15.818603	8.775022	8.242179	8.212396

Sans termes de stabilisation

Erreurs :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	0.161997	0.008819	0.010765	0.010043	0.010069	0.010067
	f=6, fp=2	eps=0.01	0.341761	8.796988	0.010451	0.010841	0.009836	0.009836
	f=2, fp=4	eps=0.01	0.035073	16.315081	0.019181	0.049081	0.053328	0.053573
	f=6, fp=4	eps=0.01	0.341761	2862.638567	0.010190	0.015019	0.010642	0.010660
	f=2, fp=6	eps=0.01	0.035073	186.507886	0.031005	0.109391	0.119442	0.120032
	f=4, fp=6	eps=0.01	0.161997	0.007672	0.020063	0.021493	0.025796	0.025814

Facteurs :

			Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	18.368859	15.048981	16.130528	16.088604	16.092506
	f=6, fp=2	eps=0.01	0.038850	32.700661	31.523922	34.745397	34.746356
	f=2, fp=4	eps=0.01	0.002150	1.828553	0.714597	0.657691	0.654687
	f=6, fp=4	eps=0.01	0.000119	33.540299	22.755172	32.113470	32.060513
	f=2, fp=6	eps=0.01	0.000188	1.131233	0.320625	0.293643	0.292202
	f=4, fp=6	eps=0.01	21.114121	8.074494	7.537316	6.280052	6.275598

Sans terme de bord

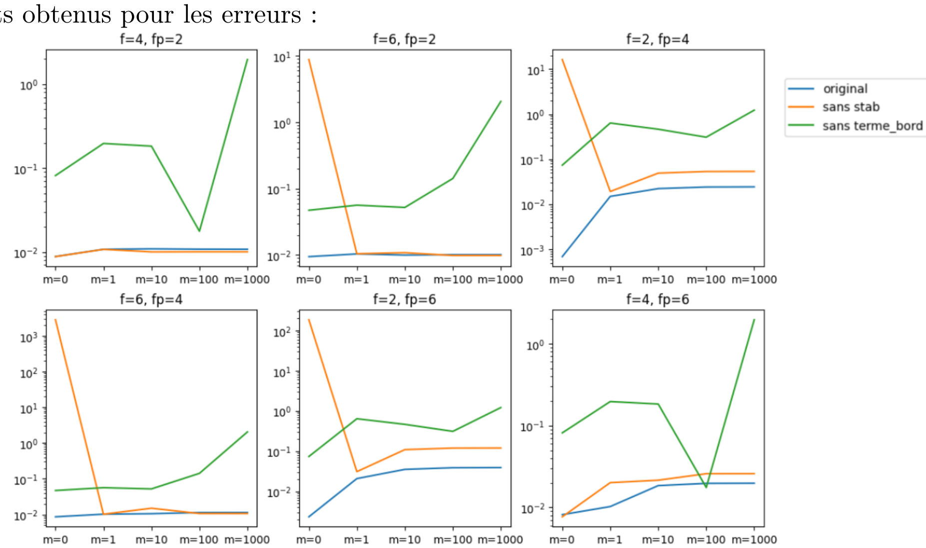
Erreurs :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	0.161997	0.081483	0.196589	0.182752	0.017697	1.966101
	f=6, fp=2	eps=0.01	0.341761	0.047200	0.056342	0.052084	0.142108	2.057690
	f=2, fp=4	eps=0.01	0.035073	0.073736	0.635883	0.462911	0.308058	1.226936
	f=6, fp=4	eps=0.01	0.341761	0.047042	0.056392	0.052125	0.142112	2.057717
	f=2, fp=6	eps=0.01	0.035073	0.073761	0.641461	0.465598	0.311316	1.216335
	f=4, fp=6	eps=0.01	0.161997	0.081346	0.196461	0.183095	0.017474	1.965036

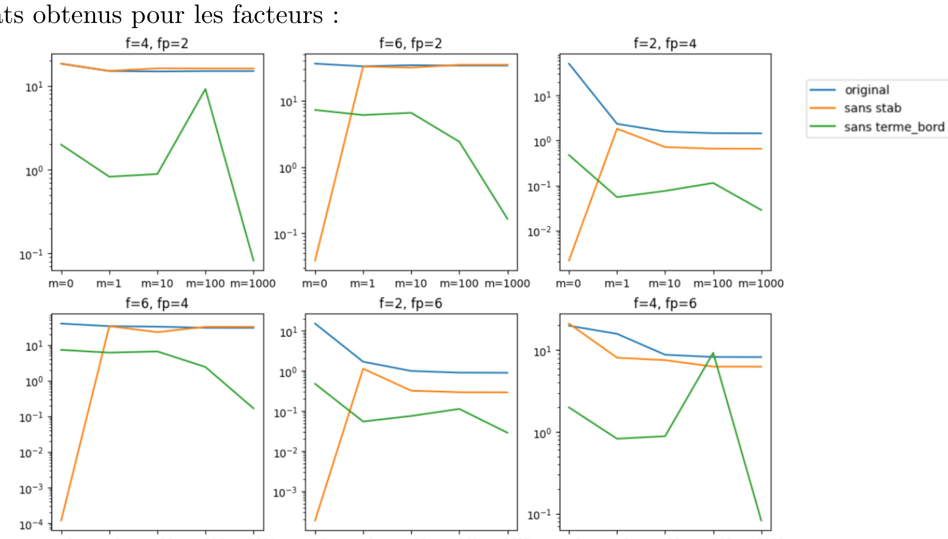
Facteurs :

			Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=4, fp=2	eps=0.01	1.988107	0.824041	0.886434	9.153871	0.082395
	f=6, fp=2	eps=0.01	7.240683	6.065790	6.561775	2.404928	0.166089
	f=2, fp=4	eps=0.01	0.475661	0.055157	0.075767	0.113854	0.028586
	f=6, fp=4	eps=0.01	7.265061	6.060448	6.556577	2.404864	0.166087
	f=2, fp=6	eps=0.01	0.475503	0.054677	0.075330	0.112662	0.028835
	f=4, fp=6	eps=0.01	1.991454	0.824579	0.884771	9.270583	0.082440

Voici les résultats obtenus pour les erreurs :



Voici les résultats obtenus pour les facteurs :



11 Semaine 11 : 17/04/2023 - 21/04/2023

Résumé

Après discussion avec Emmanuel vendredi 14/04, les points suivants vont être traités cette semaine :

- On cherchera dans un premier temps à comprendre pourquoi FEM n'a pas l'air de fonctionner aussi bien que PhiFEM en prenant $f = f_p$ et en faisant varier ϵ.
- On va ensuite tester le rehaussement avec FEM et PhiFEM avec la solution exacte (sans perturbation, $\epsilon = 0$).
- On ecrira ensuite les formulations pour le rehaussement avec PhiFEM "au propre".
- On testera ensuite d'appliquer les conditions limites pour PhiFEM différemment : on les impose de manière forte sur notre bord approché Γ_h.

(Cette semaine, Michel est en vacance.)

11.1 Correction et Rehaussement avec FEM

Les résultats obtenus la semaine dernière pour FEM (dans la partie 10.2) semblait considérablement moins bon que PhiFEM. C'est pourquoi cette partie avait pour but de comprendre d'où provenait les problèmes dans les résultats. On s'est rendu compte qu'il fallait penser à bien distinguer les cas pour imposer les conditions aux bords sur C.

Distinction des cas

On considère ici le problème de Poisson avec condition de Dirichlet homogène et non homogène.
On considère toujours qu'après une utilisation du FNO, on obtient une solution du type

$$u_p = u_{ex} + \epsilon P(x,y)$$

avec u_{ex} la solution analytique définie par

$$u_{ex}(x,y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

et P la perturbation définie par

$$P(x,y) = S_p \times \sin(2\pi f_p x + \varphi_p) \times \sin(2\pi f_p y + \varphi_p)$$

avec $\varphi_p = 0$ pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).
On cherche alors à corriger cette solution avec et sans rehaussement.
On notera dans la suite $\tilde{u}$ la solution corrigée :

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

- **Problème homogène :**

$$\begin{cases} -\Delta u = f & \Omega \\ u = 0 & \Gamma \end{cases}$$

  — **Correction :** $(m = 0)$
  On pose

$$\tilde{\phi} = u_p$$

  Et donc $\tilde{\phi} = 0$ sur γ.
  On veut

$$\begin{cases} -\Delta(\tilde{\phi} C) = f & \Omega \\ \tilde{u} = 0 & \Gamma \end{cases}$$

  Ce qui signifie que $\tilde{\phi} C = 0$ sur Γ et donc on ne sait pas la valeur de C sur Γ :

$$C = ? \quad \Gamma$$

  Remarque. *Si $f = f_p$, on a $C = \frac{u_{ex}}{(1+\epsilon)u_{ex}} = \frac{1}{1+\epsilon}$ sur Ω et donc on peut prendre $C = \frac{1}{1=\epsilon}$ sur Γ.*

  Une solution pour éviter ce problème est de rehausser la solution !

— **Rehaussement :**
On pose
$$\tilde{\phi} = u_p + m$$
On veut
$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{u} = m & \Gamma \end{cases}$$
Ainsi peu importe f et f_p, on a
$$\begin{cases} C = \dfrac{u_{ex} + m}{\tilde{\phi}} & \Omega \\ C = 1 & \Gamma \end{cases}$$

- **Problème non homogène :**
$$\begin{cases} -\Delta u = f & \Omega \\ u = g & \Gamma \end{cases}$$

Remarque. *Dans le cas non homogène, on ne peut pas avoir $P = u_{ex}$ car $P = 0$ sur Γ.*

— **Correction :** $(m = 0)$
On pose
$$\tilde{\phi} = u_p$$
Et donc $\tilde{\phi} = 0$ sur γ.
On veut
$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{u} = g & \Gamma \end{cases}$$
Ainsi peu importe f et f_p, on a
$$\begin{cases} C = \dfrac{u_{ex}}{u_{ex} + \epsilon P} & \Omega \\ C = 1 & \Gamma \end{cases}$$

— **Rehaussement :**
On pose
$$\tilde{\phi} = u_p + m$$
On veut
$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{u} = g + m & \Gamma \end{cases}$$
Ainsi peu importe f et f_p, on a
$$\begin{cases} C = \dfrac{u_{ex} + m}{\tilde{\phi}} & \Omega \\ C = 1 & \Gamma \end{cases}$$

Problème initial

On se place dans le cas où on avait obtenu les problèmes la semaine précédente, c'est-à-dire dans le cas du problème de Poisson avec condition de Dirichlet homogène avec $f = f_p$.

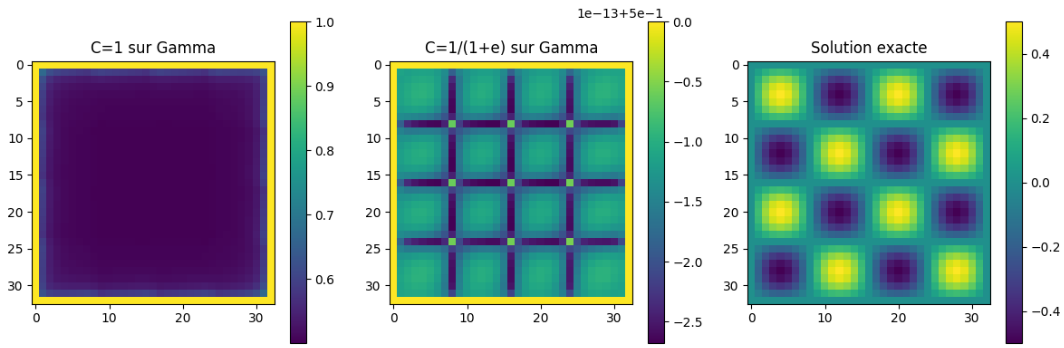
Au vue de la sous-section précédente, il semblerait que le problème obtenue vienne de la condition de bord imposée à FEM pour C (qui était $C = 1$ sur Γ). En effet, voici les résultats obtenus avec cette condition :

			eps=1	eps=0.1	eps=0.01	eps=0.001	eps=0.0001	eps=1e-05	eps=0.0
FEM	f=2	error	0.065043	0.006504	0.000650	0.000065	0.000007	6.504257e-07	2.426953e-13
		min(C)	0.500034	0.909097	0.990100	0.999001	0.999900	9.999900e-01	1.000000e+00
		max(C)	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000e+00	1.000000e+00
	f=4	error	0.093298	0.009330	0.000933	0.000093	0.000009	9.329765e-07	1.523907e-11
		min(C)	0.500000	0.909091	0.990099	0.999001	0.999900	9.999900e-01	1.000000e+00
		max(C)	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000e+00	1.000000e+00

En modifiant cette condition par $C = \frac{1}{1+\epsilon}$, on a

			eps=1	eps=0.1	eps=0.01	eps=0.001	eps=0.0001	eps=1e-05	eps=0.0
FEM	f=2	error	2.426949e-13	2.424979e-13	2.424428e-13	2.427277e-13	2.429016e-13	2.425055e-13	2.426953e-13
		min(C)	5.000000e-01	9.090909e-01	9.900990e-01	9.990010e-01	9.999000e-01	9.999900e-01	1.000000e+00
		max(C)	5.000000e-01	9.090909e-01	9.900990e-01	9.990010e-01	9.999000e-01	9.999900e-01	1.000000e+00
	f=4	error	1.523907e-11	1.523904e-11	1.523905e-11	1.523903e-11	1.523904e-11	1.523907e-11	1.523907e-11
		min(C)	5.000000e-01	9.090909e-01	9.900990e-01	9.990010e-01	9.999000e-01	9.999900e-01	1.000000e+00
		max(C)	5.000000e-01	9.090909e-01	9.900990e-01	9.990010e-01	9.999000e-01	9.999900e-01	1.000000e+00

En prenant $f = 2$ et $\epsilon = 1$, on obtient les résultats suivants (à gauche le C obtenu par FEM en imposant $C = 1$ sur Γ, au milieu le C obtenu par FEM en imposant $C = \frac{1}{1+\epsilon}$ sur Γ et à droite notre solution analytique u_{ex}) :

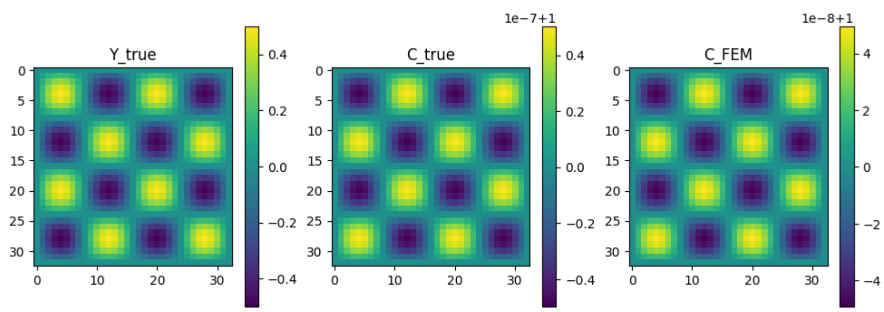


Résultats supplémentaires

En tant que vérification supplémentaire des résultats obtenus dans la distinction des cas, on va comparer le C obtenu par FEM et le C analytique dans les différents cas considérés où l'on rehausse la solution. On considerera dans la suite $S = 0.5$ et $m = 100$.

- **Problème homogène :** $\varphi = 0$

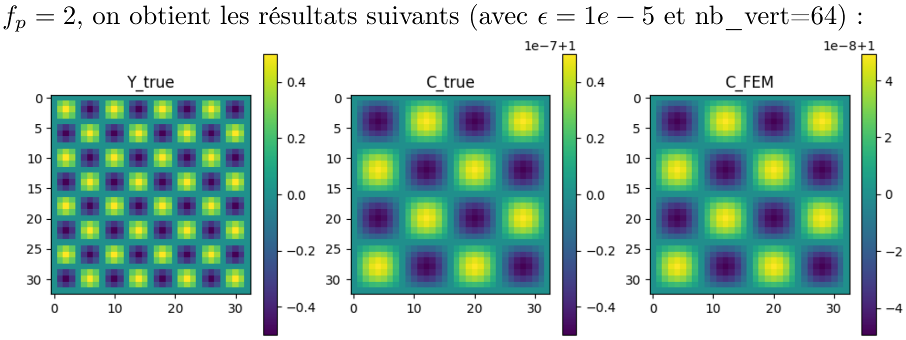
  Pour $f = f_p = 2$, on obtient les résultats suivants (avec $\epsilon = 1e-5$ et nb_vert=64) :



  En faisant varier ϵ, on obtient les résultats numérique suivants :

	eps=1	eps=0.1	eps=0.01	eps=0.001	eps=0.0001	eps=0.0
||C_true-C_FEM||_2	0.00139	0.000137	0.000014	0.000001	1.366798e-07	4.798681e-14

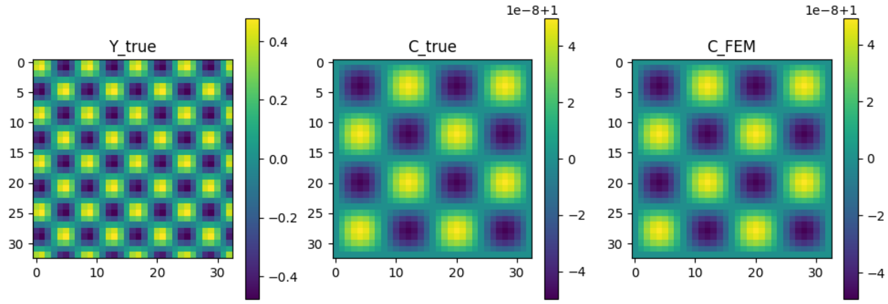
  Pour $f = 4$ et $f_p = 2$, on obtient les résultats suivants (avec $\epsilon = 1e-5$ et nb_vert=64) :



  En faisant varier ϵ, on obtient les résultats numérique suivants :

	eps=1	eps=0.1	eps=0.01	eps=0.001	eps=0.0001	eps=0.0
||C_true-C_FEM||_2	0.001369	0.000135	0.000014	0.000001	1.353618e-07	1.828914e-13

- **Problème non homogène :** $\varphi = 1$
  Pour $f = 4$ et $f_p = 2$, on obtient les résultats suivants (avec $\epsilon = 1e-5$ et nb_vert=64) :



  En faisant varier ϵ, on obtient les résultats numérique suivants :

	eps=1	eps=0.1	eps=0.01	eps=0.001	eps=0.0001	eps=0.0
||C_true-C_FEM||_2	0.001366	0.000135	0.000014	0.000001	1.351064e-07	1.213764e-13

11.2 Rehaussement PhiFEM

On considère le problème de Poisson avec condition de Dirichlet homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = 0 & \Gamma \end{cases}$$

On se place encore sur le carré $[0,1]^2$. Pour construire nos ensembles, on utilisera

$$\phi_c(x) = ||x-0.5||_\infty - 0.5$$

On considérera le domaine environnant $\mathcal{O} = [-0.5, 1.5]^2$.
Pour PhiFEM, on prendra la levelset

$$\phi(x,y) = x(1-x)y(1-y)$$

On considère encore la solution analytique suivante :

$$u_{ex}(x,y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

et pour $p = 0$, $g(x,y) = 0$.
De la même manière que pour FEM, on va considérer la solution rehaussée par m :

$$\tilde{\phi} = u_p + m$$

avec

$$u_p = u_{ex} + \epsilon P$$

Remarque. *Ici $\tilde{\phi} = m$ sur Γ (et $u_p = 0$ sur Γ).*

On cherche alors à résoudre le problème

$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{u} = m & \Gamma \end{cases}$$

On testera d'imposer les conditions aux bords de deux manières :

— 1ère méthode : Comme $\tilde{\phi} = m$ sur Γ (et pas 0), on ne fais rien !

— 2ème méthode : On cherche à imposer $C = 1$ sur Γ_h (2 cas à différencier : nb_vert=64 et nb_vert=101).

On testera dans un premier temps les 2 méthodes pour $\epsilon = 0$ (c'est-à-dire sans perturbation) puis pour différents $\epsilon \neq 0$ (avec perturbation).

Sans perturbation ($\epsilon = 0$)

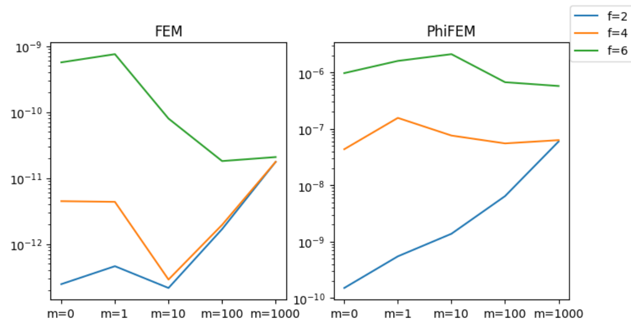
On testera pour différentes fréquences avec $f = f_p$. On prendra $\epsilon = 0$ et on fera varier m.

- **Ancienne version :** (avec ajout des termes en g)
  Voici les résultats numériques obtenus (à gauche les erreurs et à droite les facteurs) :

		FEM/PhiFEM	Corr	m=1	m=10	m=100	m=1000
FEM	f=2	0.024178	2.460038e-13	4.602080e-13	2.150931e-13	1.678049e-12	1.758857e-11
	f=4	0.095196	4.455779e-12	4.349959e-12	2.886173e-13	1.948447e-12	1.760685e-11
	f=6	0.201494	5.677455e-10	7.554587e-10	7.969937e-11	1.807977e-11	2.078066e-11
PhiFEM	f=2	0.035073	1.509278e-10	5.488548e-10	1.378591e-09	6.401034e-09	5.951891e-08
	f=4	0.161997	4.342085e-08	1.559231e-07	7.584557e-08	5.505268e-08	6.288545e-08
	f=6	0.341761	9.690887e-07	1.594400e-06	2.100021e-06	6.689421e-07	5.727617e-07

		Corr	m=1	m=10	m=100	m=1000
FEM	f=2	9.828165e+10	5.253637e+10	1.124055e+11	1.440819e+10	1.374623e+09
	f=4	2.136453e+10	2.188426e+10	3.298334e+11	4.885718e+10	5.406737e+09
	f=6	3.549021e+08	2.667175e+08	2.528176e+09	1.114473e+10	9.696231e+09
PhiFEM	f=2	2.323857e+08	6.390300e+07	2.544154e+07	5.479344e+06	5.892827e+05
	f=4	3.730866e+06	1.038957e+06	2.135885e+06	2.942588e+06	2.576071e+06
	f=6	3.526618e+05	2.143506e+05	1.627415e+05	5.108970e+05	5.966889e+05

  En traçant les erreurs FEM et PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :

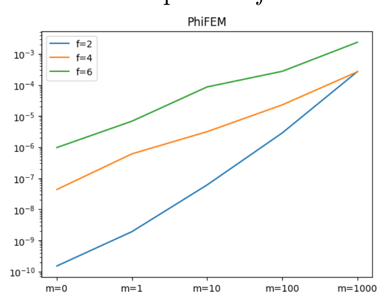


- **1ère méthode :** (on fait rien !)
  Voici les résultats numériques obtenus (à gauche les erreurs et à droite les facteurs) :

		PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	0.035073	1.509278e-10	1.923416e-09	6.094305e-08	0.000003	0.000268
	f=4	0.161997	4.342085e-08	6.084181e-07	3.141666e-06	0.000023	0.000264
	f=6	0.341761	9.690887e-07	6.829327e-06	8.621124e-05	0.000274	0.002350

		Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	2.323857e+08	1.823499e+07	575512.150134	12259.593982	130.911383
	f=4	3.730866e+06	2.662600e+05	51564.160261	7091.371928	612.688077
	f=6	3.526618e+05	5.004308e+04	3964.222270	1246.127708	145.449883

  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :

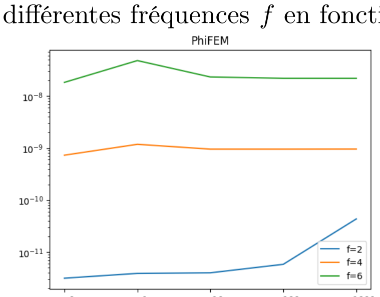


- **2ème méthode : nb_vert=101**
  Voici les résultats numériques obtenus (à gauche les erreurs et à droite les facteurs) :

		PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	0.014206	3.152929e-12	3.899212e-12	3.999729e-12	5.829417e-12	4.344627e-11
	f=4	0.067444	7.306653e-10	1.182244e-09	9.557133e-10	9.556407e-10	9.589799e-10
	f=6	0.152678	1.834458e-08	4.820489e-08	2.331348e-08	2.199666e-08	2.198982e-08

		Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	4.505738e+09	3.643370e+09	3.551809e+09	2.436998e+09	3.269849e+08
	f=4	9.230558e+07	5.704786e+07	7.056979e+07	7.057515e+07	7.032940e+07
	f=6	8.322805e+06	3.167279e+06	6.548929e+06	6.940979e+06	6.943139e+06

  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :



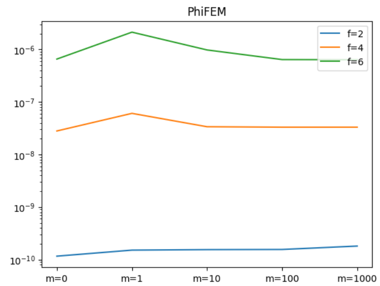
- **2ème méthode : nb_vert=64**

  Voici les résultats numériques obtenus (à gauche les erreurs et à droite les facteurs) :

		PhiFEM	Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	0.035073	1.164885e-10	1.517215e-10	1.550573e-10	1.561263e-10	1.812841e-10
	f=4	0.161997	2.806979e-08	6.074009e-08	3.379375e-08	3.313619e-08	3.314758e-08
	f=6	0.341761	6.552125e-07	2.131420e-06	9.771107e-07	6.408918e-07	6.372419e-07

		Corr	m=1	m=10	m=100	m=1000
PhiFEM	f=2	3.010894e+08	2.311700e+08	2.261968e+08	2.246481e+08	1.934724e+08
	f=4	5.771236e+06	2.667058e+06	4.793709e+06	4.888835e+06	4.887155e+06
	f=6	5.216026e+05	1.603441e+05	3.497664e+05	5.332578e+05	5.363121e+05

  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :



Avec perturbation ($\epsilon \neq 0$)

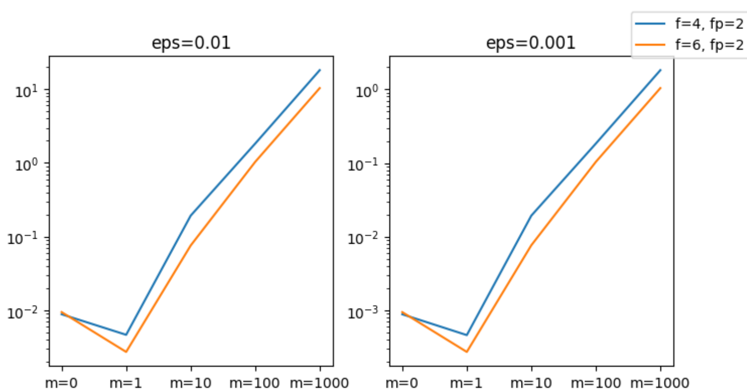
On testera pour différentes fréquences avec $f \neq f_p$. On prendra $\epsilon = 0.01$ et $\epsilon = 0.001$ et on fera varier m.

- **1ère méthode :** (on fait rien !)

  Voici les résultats numériques obtenus :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
f=4, fp=2	eps=0.01	error	0.161997	0.008844	0.004653	0.192623	1.831358	18.155628
		min(C)	NaN	0.996790	0.992952	0.994678	0.995512	0.995613
		max(C)	NaN	1.002834	1.004777	0.995971	0.995652	0.995627
	eps=0.001	error	0.161997	0.000884	0.000461	0.019322	0.183870	1.822954
		min(C)	NaN	0.999693	0.999295	0.999466	0.999549	0.999559
		max(C)	NaN	1.000296	1.000478	0.999596	0.999563	0.999561
f=6, fp=2	eps=0.01	error	0.341761	0.009455	0.002725	0.075771	1.033636	10.365274
		min(C)	NaN	0.998505	0.996125	0.997535	0.997417	0.997465
		max(C)	NaN	1.000806	1.006047	0.998751	0.997545	0.997479
	eps=0.001	error	0.341761	0.000946	0.000273	0.007655	0.103835	1.040995
		min(C)	NaN	0.999851	0.999611	0.999752	0.999741	0.999745
		max(C)	NaN	1.000082	1.000601	0.999873	0.999753	0.999747

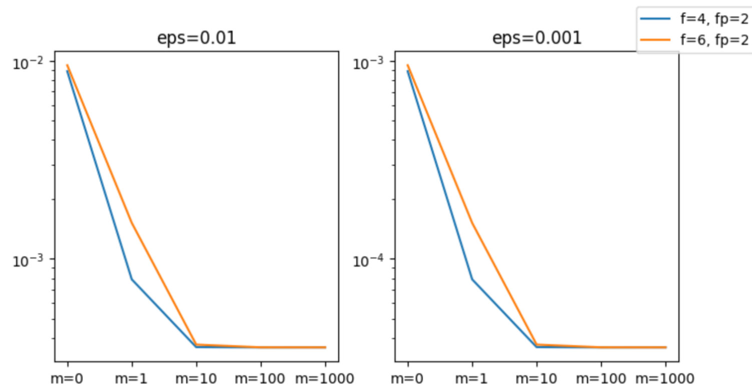
  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :



- **2ème méthode : nb_vert=101**
  Voici les résultats numériques obtenus :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
f=4, fp=2	eps=0.01	error	0.067444	0.008879	0.000789	0.000358	0.000357	0.000357
		min(C)	NaN	0.995732	0.994487	0.999505	0.999950	0.999995
		max(C)	NaN	1.002894	1.005662	1.000496	1.000050	1.000005
	eps=0.001	error	0.067444	0.000888	0.000079	0.000036	0.000036	0.000036
		min(C)	NaN	0.999615	0.999447	0.999951	0.999995	1.000000
		max(C)	NaN	1.000306	1.000565	1.000050	1.000005	1.000000
f=6, fp=2	eps=0.01	error	0.152678	0.009518	0.001522	0.000369	0.000358	0.000357
		min(C)	NaN	0.998231	0.995343	0.999527	0.999951	0.999995
		max(C)	NaN	1.001065	1.008036	1.000520	1.000050	1.000005
	eps=0.001	error	0.152678	0.000952	0.000152	0.000037	0.000036	0.000036
		min(C)	NaN	0.999828	0.999534	0.999953	0.999995	1.000000
		max(C)	NaN	1.000107	1.000800	1.000052	1.000005	1.000000

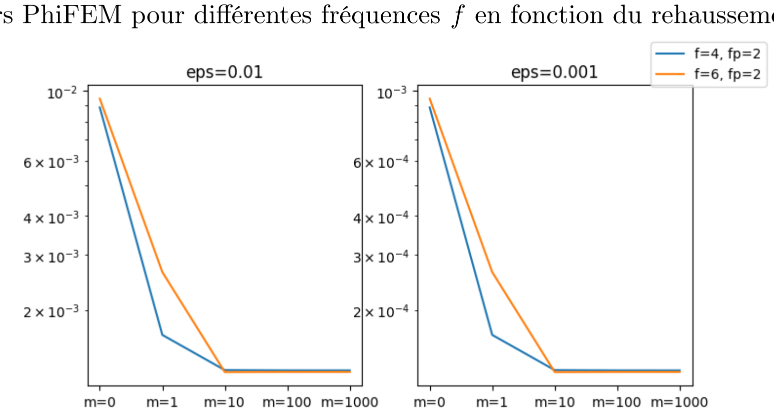
  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :



- **2ème méthode : nb_vert=64**
  Voici les résultats numériques obtenus :

			PhiFEM	Corr	m=1	m=10	m=100	m=1000
f=4, fp=2	eps=0.01	error	0.161997	0.008866	0.001667	0.001287	0.001285	0.001284
		min(C)	NaN	0.996787	0.994838	0.999498	0.999950	0.999995
		max(C)	NaN	1.002836	1.005253	1.000504	1.000050	1.000005
	eps=0.001	error	0.161997	0.000887	0.000167	0.000129	0.000128	0.000128
		min(C)	NaN	0.999693	0.999483	0.999950	0.999995	0.999999
		max(C)	NaN	1.000296	1.000524	1.000050	1.000005	1.000001
f=6, fp=2	eps=0.01	error	0.341761	0.009456	0.002646	0.001269	0.001270	0.001270
		min(C)	NaN	0.998505	0.996316	0.999517	0.999950	0.999995
		max(C)	NaN	1.000809	1.005964	1.000522	1.000051	1.000005
	eps=0.001	error	0.341761	0.000946	0.000264	0.000127	0.000127	0.000127
		min(C)	NaN	0.999851	0.999631	0.999952	0.999995	0.999999
		max(C)	NaN	1.000082	1.000594	1.000052	1.000005	1.000001

  En traçant les erreurs PhiFEM pour différentes fréquences f en fonction du rehaussement m, on obtient :



12 Semaine 12 : 24/04/2023 - 28/04/2023

Résumé

Cette semaine, j'ai corrigé certains des problèmes obtenus la semaine dernière. Les bons résultats sont présentés en semaine 11. J'ai également rédigé un document expliquant l'intérêt du rehaussement.

12.1 Explication rehaussement pour FEM

On considère le problème de Poisson avec condition de Dirichlet homogène ou non homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = g & \Gamma \end{cases} \tag{$\mathcal{E}_1$}$$

On a ainsi une EDP que l'on souhaite résoudre sur un domaine Ω. On note Γ le bord de Ω, c'est-à-dire $\Gamma = \partial\Omega$. Dans notre cas, on souhaite appliquer une correction à la sortie d'un FNO. On considère ici que l'on possède une solution analytique u et qu'après une utilisation du FNO, on obtient une solution du type

$$u_p(x,y) = u(x,y) - \epsilon P(x,y) \tag{1}$$

avec P la perturbation (tel que $P = 0$ sur Γ) et ϵ petit.
On supposera que $||P||_{H^{k+1}(\Omega)} \leq 1$.
On souhaite ainsi résoudre le problème suivant

$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{u} = g & \Gamma \end{cases} \tag{$\mathcal{E}_2$}$$

avec $\tilde{\phi} = u_p$ et $\tilde{u} = \tilde{\phi}C$.
Ce document a pour but d'expliquer l'intérêt de rehausser la solution (avec FEM) afin de réduire le plus possible l'erreur $||u - u_c||_{L^2}$ où u_c est la solution obtenu après la correction.

Principe général de FEM

La démarche générale de la méthode des éléments finis consiste à écrire la formulation variationnelle de cette EDP et ainsi à se ramener à un problème du type

$$\text{Trouver } u \in V \text{ tel que } a(u,v) = l(v),\ \forall v \in V \tag{$\mathcal{P}$}$$

On définit alors un maillage du domaine Ω, grâce auquel on va définir un espace d'approximation V_h, sous-espace vectoriel de V de dimension fini N_h. On écrit alors le problème approché

$$\text{Trouver } u_h \in V_h \text{ tel que } a(u_h,v_h) = l(v_h),\ \forall v_h \in V \tag{$\mathcal{P}_h$}$$

On considère une base $(\varphi_1, \dots, \varphi_{N_h})$ de V_h. En décomposant u_h sur cette base sous la forme

$$u_h = \sum_{i=1}^{N_h} \mu_i \varphi_i \tag{2}$$

le problème ($\mathcal{P}_h$) se réécrit

$$\text{Trouver } \mu_1, \dots, \mu_{N_h} \text{ tels que } \sum_{i=1}^{N_h} \mu_i a(\varphi_i, v_h) = l(v_h),\ \forall v_h \in V$$

ou encore

$$\text{Trouver } \mu_1, \dots, \mu_{N_h} \text{ tels que } \sum_{i=1}^{N_h} \mu_i a(\varphi_i, \varphi_j) = l(\varphi_j),\ \forall j \in \{1, \dots, N_h\}$$

La résolution de l'EDP consiste alors à résoudre le système linéaire suivant :

$$A\mu = b$$

avec

$$A = (a(\varphi_i, \varphi_j))_{1\leq i,j\leq N_h}, \quad \mu = (\mu_i)_{1\leq i\leq N_h} \quad \text{et} \quad b = (l(\varphi_j))_{1\leq j\leq N_h}$$

Le théorème de convergence de FEM nous donne l'inégalité suivante :

$$||u - u_h||_{L^2(\Omega)} \leq ch^{k+1}|u|_{H^{k+1}(\Omega)} \tag{3}$$

Application à la correction

On considère à présent uniquement le problème ($\mathcal{E}_2$). On peut alors effectuer le même type de raisonnement dans le cas de la correction. Ainsi par (2), la décomposition de u_h sur la base $(\varphi_1, \ldots, \varphi_{N_h})$ de V_h s'écrit pour ce problème

$$u_h = \sum_{i=1}^{N_h} \mu_i(\varphi_i \tilde{\phi}(x)) = \left(\sum_{i=1}^{N_h} \mu_i \varphi_i\right) \tilde{\phi}(x) \tag{4}$$

Ainsi par (4) on en déduit

$$\mu_i = \frac{u(x_i)}{\tilde{\phi}(x_i)}$$

Et ainsi l'inégalité (3) se réécrit pour le problème ($\mathcal{E}_2$) :

$$||u - u_h||_0 \leq ch^{k+1}||\tilde{\phi}||_\infty \left|C\right|_{k+1} \tag{5}$$

avec $C = \frac{u}{\tilde{\phi}}$ la solution exacte du problème.

Remarque. *On notera que si $\epsilon = 0$ (c'est-à-dire qu'il n'y a pas de perturbation), alors $C = 1$.*

On considère une solution $\mathbb{P}^1$ $(k = 1)$, ainsi

$$|C|_{H^2(\Omega)} = \left|\frac{u}{\tilde{\phi}}\right|_{H^2(\Omega)} = \left|\left|\left(\frac{u}{\tilde{\phi}}\right)''\right|\right|_{L^2(\Omega)} = \epsilon \left|\left|\left(\frac{P}{\tilde{\phi}}\right)''\right|\right|_{L^2(\Omega)} \tag{6}$$

car

$$\left(\frac{u}{\tilde{\phi}}\right)'' = \left(\frac{\tilde{\phi} + \epsilon P}{\tilde{\phi}}\right)'' = \left(1 + \epsilon\frac{P}{\tilde{\phi}}\right)'' = \epsilon\left(\frac{P}{\tilde{\phi}}\right)''$$

avec

$$\left(\frac{P}{\tilde{\phi}}\right)'' = \frac{P''\tilde{\phi} - P\tilde{\phi}''}{\tilde{\phi}^2} + \frac{2(P\tilde{\phi}' - P'\tilde{\phi})\tilde{\phi}'}{\tilde{\phi}^3}$$

Rehaussement

L'idée du rehaussement est la suivante : on considère cette fois

$$\hat{\phi} = \tilde{\phi} + m$$

avec m une constante.
On souhaite alors résoudre le problème suivant

$$\begin{cases} -\Delta(\hat{\phi}C) = f & \Omega \\ \hat{u} = g + m & \Gamma \end{cases} \tag{$\mathcal{E}_3$}$$

Dans le cas où la solution s'annule, rehausser le problème permet d'améliorer la correction. Autrement dit si la solution peut-être nulle, il faut rehausser le problème.
Si la solution ne s'annule pas, rehausser le problème permet dans certains cas de diminuer l'erreur.
En effet, l'inégalité (5) se réécrit pour le problème ($\mathcal{E}_3$) :

$$||\hat{u} - \hat{u_h}||_0 \leq ch^{k+1}||\hat{\phi}||_\infty \left|C\right|_{k+1} \tag{7}$$

avec $C = \frac{u+m}{\hat{\phi}}$ la solution exacte du problème.
De la même manière que (6), on a :

$$|C|_{H^2(\Omega)} = \left|\frac{u+m}{\hat{\phi}}\right|_{H^2(\Omega)} = \left|\left|\left(\frac{u+m}{\hat{\phi}}\right)''\right|\right|_{L^2(\Omega)} = \epsilon\left|\left|\left(\frac{P}{\hat{\phi}}\right)''\right|\right|_{L^2(\Omega)} = \epsilon\left|\left|\left(\frac{P}{\tilde{\phi}+m}\right)''\right|\right|_{L^2(\Omega)} \tag{8}$$

avec

$$\begin{aligned}\left(\frac{P}{\hat{\phi}}\right)'' &= \frac{P''(\tilde{\phi}+m) - P\tilde{\phi}''}{(\tilde{\phi}+m)^2} + \frac{2(P\tilde{\phi}' - P'(\tilde{\phi}+m))\tilde{\phi}'}{(\tilde{\phi}+m)^3} \\ &= \frac{P''\tilde{\phi} - P\tilde{\phi}''}{(\tilde{\phi}+m)^2} + \frac{2(P\tilde{\phi}' - P'\tilde{\phi})\tilde{\phi}'}{(\tilde{\phi}+m)^3} + \frac{mP''}{(\tilde{\phi}+m)^2} - \frac{2mP'\tilde{\phi}'}{(\tilde{\phi}+m)^3}\end{aligned}$$

Or

$$\left|\left|\left(\frac{P}{\tilde{\phi}+m}\right)''\right|\right|_{L^2(\Omega)} = \left|\left|\left(\frac{P}{m\left(1+\frac{\tilde{\phi}}{m}\right)}\right)''\right|\right|_{L^2(\Omega)} = \frac{1}{m}\left|\left|\left(\frac{P}{1+\frac{\tilde{\phi}}{m}}\right)''\right|\right|_{L^2(\Omega)}$$

Alors, pour m suffisamment grand :

$$||\hat{\phi}||_\infty \sim m$$

et

$$\left|\left|\left(\frac{P}{1+\frac{\tilde{\phi}}{m}}\right)''\right|\right|_{L^2(\Omega)} \sim ||P''||_{L^2(\Omega)}$$

car

$$\begin{aligned}\left(\frac{P}{1+\frac{\tilde{\phi}}{m}}\right)'' = m\left(\frac{P}{\hat{\phi}}\right)'' &= \frac{m(P''\tilde{\phi} - P\tilde{\phi}'')}{(\tilde{\phi}+m)^2} + \frac{2m(P\tilde{\phi}' - P'\tilde{\phi})\tilde{\phi}'}{(\tilde{\phi}+m)^3} + \frac{m^2P''}{(\tilde{\phi}+m)^2} - \frac{2m^2P'\tilde{\phi}'}{(\tilde{\phi}+m)^3} \\ &= \frac{m(P''\tilde{\phi} - P\tilde{\phi}'')}{m^2\left(1+\frac{\tilde{\phi}}{m}\right)^2} + \frac{2m(P\tilde{\phi}' - P'\tilde{\phi})\tilde{\phi}'}{m^3\left(1+\frac{\tilde{\phi}}{m}\right)^3} + \frac{m^2P''}{m^2\left(1+\frac{\tilde{\phi}}{m}\right)^2} - \frac{2m^2P'\tilde{\phi}'}{m^3\left(1+\frac{\tilde{\phi}}{m}\right)^3} \\ &= \frac{P''\tilde{\phi} - P\tilde{\phi}''}{m\left(1+\frac{\tilde{\phi}}{m}\right)^2} + \frac{2(P\tilde{\phi}' - P'\tilde{\phi})\tilde{\phi}'}{m^2\left(1+\frac{\tilde{\phi}}{m}\right)^3} + \frac{P''}{\left(1+\frac{\tilde{\phi}}{m}\right)^2} - \frac{2P'\tilde{\phi}'}{m\left(1+\frac{\tilde{\phi}}{m}\right)^3}\end{aligned}$$

Donc pour m suffisamment grand, (7) devient

$$\left|\left|\frac{u+m}{\hat{\phi}} - C_h\right|\right|_{L^2(\Omega)} \leq ch^{k+1}\epsilon\,||P''||_{L^2(\Omega)} \tag{9}$$

Et ainsi, quand m est grand, l'erreur ne dépend plus de la solution mais dépend uniquement de P.
Quand m est petit, l'erreur est dominée par les dérivées et dérivées secondes de la solution perturbée $\tilde{\phi}$.

Résultats numériques

On prend ici la solution analytique suivante

$$u_{ex}(x,y) = S \times \sin(2\pi fx + p) \times \sin(2\pi fy + p)$$

et P la perturbation définie par

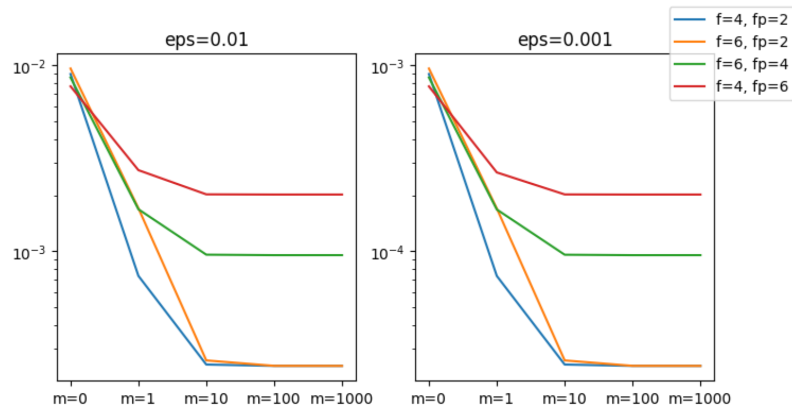
$$P(x,y) = S \times \sin(2\pi f_p x + p_p) \times \sin(2\pi f_p y + p_p)$$

avec $p_p = 0$ pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).
On cherche alors à corriger cette solution avec et sans rehaussement.
On prendra $S = 0.5$ et $p = 0$ (c'est-à-dire $g = 0$). On fera varier ϵ, f et f_p.

Voici les résultats obtenus (avec à gauche les erreurs et à droite les facteurs obtenus en comparant avec l'erreur FEM classique) :

			FEM	Corr	m=1	m=10	m=100	m=1000
FEM	f=4, fp=2	eps=0.01	0.095196	0.008953	0.000736	0.000246	0.000242	0.000242
		eps=0.001	0.095196	0.000895	0.000074	0.000025	0.000024	0.000024
	f=6, fp=2	eps=0.01	0.201494	0.009584	0.001701	0.000259	0.000242	0.000242
		eps=0.001	0.201494	0.000959	0.000170	0.000026	0.000024	0.000024
	f=6, fp=4	eps=0.01	0.201494	0.008578	0.001679	0.000957	0.000952	0.000952
		eps=0.001	0.201494	0.000858	0.000167	0.000096	0.000095	0.000095
	f=4, fp=6	eps=0.01	0.095196	0.007688	0.002725	0.002020	0.002015	0.002015
		eps=0.001	0.095196	0.000769	0.000265	0.000202	0.000201	0.000201

			Corr	m=1	m=10	m=100	m=1000
FEM	f=4, fp=2	eps=0.01	10.632588	129.268768	387.223595	393.748857	393.742152
		eps=0.001	106.315008	1293.647125	3872.070426	3937.472946	3937.419987
	f=6, fp=2	eps=0.01	21.024327	118.426204	778.838663	832.835214	833.388314
		eps=0.001	210.201187	1187.929347	7788.825049	8328.329635	8333.880044
	f=6, fp=4	eps=0.01	23.489579	119.976882	210.472466	211.708833	211.669409
		eps=0.001	234.912870	1203.206236	2105.810805	2117.086604	2116.692744
	f=4, fp=6	eps=0.01	12.381564	34.934546	47.135628	47.251756	47.245743
		eps=0.001	123.831430	359.222973	471.745833	472.517487	472.457027



Application à ϕ-FEM

On souhaitera par la suite obtenir le même type de résultats avec ϕ-FEM. L'idée décrite ici sera la même mais pour l'instant les résultats obtenus ne sont pas ceux attendus.

13 Semaine 13 : 01/05/2023 - 05/05/2023

13.1 Comparaison de 2 méthodes de correction

Résumé

On considère le problème de Poisson avec condition de Dirichlet homogène ou non homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = g & \Gamma \end{cases} \tag{$\mathcal{E}_1$}$$

On a ainsi une EDP que l'on souhaite résoudre sur un domaine Ω. On note Γ le bord de Ω, c'est-à-dire $\Gamma = \partial\Omega$.

Dans notre cas, on souhaite appliquer une correction à la sortie d'un FNO. On considère ici que l'on possède une solution analytique u et qu'après une utilisation du FNO, on obtient une solution du type

$$\tilde{\phi}(x,y) = u_p(x,y) = u(x,y) - \epsilon P(x,y)$$

avec P la perturbation (tel que $P = 0$ sur Γ) et ϵ petit.

Ce document a pour but de comparer deux méthodes de correction de la solution obtenue.

13.1.1 Correction avec FEM

Présentation des 2 méthodes

- **Méthode 1 :** On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ C = 1 & \Gamma \end{cases} \tag{$\mathcal{C}_1$}$$

avec $\tilde{u} = \tilde{\phi}C$.

Dans un autre document, on a présenté l'intérêt de rehausser le problème et de se ramener au problème suivant

$$\begin{cases} -\Delta(\hat{\phi}C) = f & \Omega \\ \hat{u} = g + m & \Gamma \end{cases} \tag{$\mathcal{C}_1^{\mathcal{R}}$}$$

avec $\hat{u} = \hat{\phi}C + m$ où $\hat{\phi} = \tilde{\phi} + m$ (m une constante).

- **Méthode 2 :** On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta C = \tilde{f} & \Omega \\ C = 0 & \Gamma \end{cases} \tag{$\mathcal{C}_2$}$$

avec $\tilde{u} = \tilde{\phi} + C$ et $\tilde{f} = f + \Delta\tilde{\phi}$.

Remarque. *On notera que dans ce cas rehausser le problème n'a aucun intérêt.*
En effet, la décomposition de C_h sur la base $(\varphi_1, \ldots, \varphi_{N_h})$ de V_h s'écrit pour ce problème

$$C_h = \sum_{i=1}^{N_h} C_i \varphi_i$$

avec $C_i = u(x_i) - \tilde{\phi}(x_i)$. Et donc, on a l'inégalité suivante

$$||(u - \tilde{\phi}) - C_h||_{L^2(\Omega)} \leq ch^{k+1}|u - \tilde{\phi}|_{H^{k+1}(\Omega)}$$

Alors pour $k = 1$

$$|u - \tilde{\phi}|_{H^2(\Omega)} = ||(u - \tilde{\phi})''||_{L^2(\Omega)} = ||(\tilde{\phi} + \epsilon P - \tilde{\phi})''||_{L^2(\Omega)} = ||P''||_{L^2(\Omega)}$$

Et ainsi, en prenant $\hat{\phi} = \tilde{\phi} + m$ on obtient le même résultat.

Résultats numériques

On se place sur le carré $[0,1]^2$. On considère ici la solution analytique suivante

$$u_{ex}(x,y) = S \times \sin(2\pi f x + p) \times \sin(2\pi f y + p)$$

et P la perturbation définie par

$$P(x,y) = S \times \sin(2\pi f_p x + p_p) \times \sin(2\pi f_p y + p_p)$$

avec $p_p = 0$ pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).
On cherche alors principalement à comparer les erreurs en norme L^2 obtenus avec les problèmes 14.1 et $\mathcal{C}_2$.
On prendra $S = 0.5$ et $p = 0$ (c'est-à-dire $g = 0$). On fera varier ϵ, f et f_p.
Voici les résultats obtenus :

			FEM	Corr	m=1000	Corr v2
FEM	f=4, fp=2	eps=0.01	0.170794	0.009114	0.000455	0.000455
		eps=0.001	0.170794	0.000911	0.000045	0.000045
	f=6, fp=2	eps=0.01	0.340309	0.009562	0.000455	0.000455
		eps=0.001	0.340309	0.000957	0.000045	0.000045
	f=8, fp=2	eps=0.01	0.511393	0.009871	0.000455	0.000455
		eps=0.001	0.511393	0.000987	0.000045	0.000045
	f=2, fp=4	eps=0.01	0.045487	0.001398	0.001708	0.001708
		eps=0.001	0.045487	0.000140	0.000171	0.000171
	f=2, fp=6	eps=0.01	0.045487	0.002909	0.003403	0.003403
		eps=0.001	0.045487	0.000292	0.000340	0.000340
	f=2, fp=8	eps=0.01	0.045487	0.004749	0.005114	0.005114
		eps=0.001	0.045487	0.000471	0.000511	0.000511

FIGURE 5 – Résultats FEM pour nb_vert=64

Il semblerait ici que les résultats obtenus pour les problèmes 14.1 avec $m = 1000$ (avant-dernière colonne) et $\mathcal{C}_2$ (dernière colonne) soient très proches.

Explication

On cherche ici à comprendre pourquoi on obtient des résultats aussi proches avec les 2 méthodes.

Méthode 1

On cherche à résoudre le problème

$$\begin{cases} -\Delta(\hat{\phi} C) = f & \Omega \\ \hat{u} = g + m & \Gamma \end{cases} \tag{$\mathcal{C}_1^{\mathcal{R}}$}$$

avec $\hat{u} = \hat{\phi} C + m$ où $\hat{\phi} = \tilde{\phi} + m$ (m une constante).
La décomposition de $\hat{u_h}$ sur la base $(\varphi_1, \dots, \varphi_{N_h})$ de V_h s'écrit pour ce problème

$$\hat{u_h} = C_h \hat{\phi} = \left(\sum_{i=1}^{N_h} C_i \varphi_i \right) \hat{\phi}(x) \tag{1}$$

Or

$$C_i = \frac{u(x_i) + m}{\hat{\phi}(x_i)} = \frac{u(x_i) + m}{\tilde{\phi}(x_i) + m} \tag{2}$$

avec

$$u(x_i) = \tilde{\phi}(x_i) + \epsilon P(x_i) \tag{3}$$

et

$$\tilde{\phi}(x) = \tilde{\phi}(x_i) + (x - x_i)\tilde{\phi}'(x_i) \tag{4}$$

De plus

$$\sum_{i=1}^{N_h} \varphi_i = 1 \tag{5}$$

Avec les 4 relations précédentes, on peut développer 1 :

$$
\begin{aligned}
\hat{u_h} &= \left(\sum_{i=1}^{N_h} C_i \varphi_i\right) \hat{\phi}(x) \\
&= \left(\sum_{i=1}^{N_h} \frac{u(x_i)+m}{\tilde{\phi}(x_i)+m} \varphi_i\right) \hat{\phi}(x) \quad \text{par } 2 \\
&= \left(\sum_{i=1}^{N_h} \frac{\tilde{\phi}(x_i)+m+\epsilon P(x_i)}{\tilde{\phi}(x_i)+m} \varphi_i\right) \hat{\phi}(x) \quad \text{par } 3 \\
&= \sum_{i=1}^{N_h} \left(1+\epsilon \frac{P(x_i)}{\tilde{\phi}(x_i)+m}\right) \varphi_i \hat{\phi}(x) \\
&= \left(\sum_{i=1}^{N_h} \varphi_i\right) \hat{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \frac{\hat{\phi}(x)}{\tilde{\phi}(x_i)+m} \varphi_i \\
&= \hat{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \frac{\tilde{\phi}(x_i)+m+(x-x_i)\tilde{\phi}'(x_i)}{\tilde{\phi}(x_i)+m} \varphi_i \quad \text{par } 4 \text{ et } 5 \\
&= \hat{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \left(1+\frac{(x-x_i)\tilde{\phi}'(x_i)}{\tilde{\phi}(x_i)+m}\right) \varphi_i \\
&= \tilde{\phi}(x) + m + \epsilon \sum_{i=1}^{N_h} P(x_i) \left(1+\frac{(x-x_i)\tilde{\phi}'(x_i)}{\tilde{\phi}(x_i)+m}\right) \varphi_i
\end{aligned}
$$

Ainsi

$$u_h = \hat{u_h} - m = \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \left(1+\frac{(x-x_i)\tilde{\phi}'(x_i)}{\tilde{\phi}(x_i)+m}\right) \varphi_i$$

et finalement

$$u_h \xrightarrow[m\to\infty]{} \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i)\varphi_i \tag{6}$$

Remarque. *Pour le problème $\mathcal{C}_1$, (équivalent au problème 14.1 avec $m=0$), on a*

$$u_h = \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \left(1+\frac{(x-x_i)\tilde{\phi}'(x_i)}{\tilde{\phi}(x_i)}\right) \varphi_i$$

Méthode 2
On cherche à résoudre le problème

$$\begin{cases} -\Delta C = \tilde{f} & \Omega \\ C = 0 & \Gamma \end{cases} \tag{$\mathcal{C}_2$}$$

avec $\tilde{u} = \tilde{\phi} + C$ et $\tilde{f} = f + \Delta\tilde{\phi}$.
La décomposition de u_h sur la base $(\varphi_1, \ldots, \varphi_{N_h})$ de V_h s'écrit pour ce problème

$$u_h = C_h + \tilde{\phi} = \left(\sum_{i=1}^{N_h} C_i \varphi_i\right) + \tilde{\phi}(x) \tag{7}$$

Or

$$C_i = u(x_i) - \tilde{\phi}(x_i) \tag{8}$$

avec

$$u(x_i) = \tilde{\phi}(x_i) + \epsilon P(x_i) \tag{9}$$

Avec les 2 relations précédentes, on peut développer 7 :

$$\begin{aligned} u_h &= \tilde{\phi}(x) + \sum_{i=1}^{N_h} C_i \varphi_i \\ &= \tilde{\phi}(x) + \sum_{i=1}^{N_h} (u(x_i) - \tilde{\phi}(x_i))\varphi_i \quad \text{par } 8 \\ &= \tilde{\phi}(x) + \sum_{i=1}^{N_h} (\tilde{\phi}(x_i) + \epsilon P(x_i) - \tilde{\phi}(x_i))\varphi_i \quad \text{par } 9 \\ u_h &= \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i)\varphi_i \end{aligned} \tag{10}$$

Ainsi par 6 et 10, il semblerait que pour le problème $\mathcal{E}_1$, les 2 méthodes proposées soit équivalentes (en prenant m grand).

13.1.2 Correction avec ϕ-FEM

Remarque. *Le rehaussement avec ϕ-FEM n'étant pas encore fonctionnel, on comparera seulement la seconde méthode avec la méthode classique (pour $m = 0$). On testera par la suite d'imposer les conditions au bord par la méthode duale et finalement on pourra comparer le rehaussement avec la nouvelle méthode de correction.*

Présentation des 2 méthodes

- **Méthode 1 :** On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ C = 1 & \Gamma \end{cases} \tag{$\mathcal{C}_1$}$$

  avec $\tilde{u} = \tilde{\phi}C$.

- **Méthode 2 :** On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\phi C) = \tilde{f} & \Omega \\ \tilde{C} = 0 & \Gamma \end{cases} \tag{$\mathcal{C}_2$}$$

  avec $\tilde{u} = \tilde{\phi} + \tilde{C}$ où $\tilde{C} = \phi C$ et $\tilde{f} = f + \Delta\tilde{\phi}$.

Résultats numériques

Nous allons considérer deux cas tests : le premier sera de considérer comme géométrie un carré (similaire au cas test de FEM) et le second sera de considérer un cercle.

1er cas test : le carré

On se place sur le carré $[0,1]^2$. On prend alors

$$\phi_c(x,y) = ||x - 0.5||_\infty - 0.5$$

pour construire les ensembles $\mathcal{F}_h^\Gamma$ et $\mathcal{T}_h^\Gamma$ nécessaire à ϕ-FEM.
On considérera alors le domaine environnant $\mathcal{O} = [-0.5, 1.5]^2$ et on prendra la levelset

$$\phi(x,y) = x(1-x)y(1-y)$$

On considère la même solution analytique que celle utilisée dans le cas test de FEM :

$$u_{ex}(x,y) = S \times \sin(2\pi f x + p) \times \sin(2\pi f y + p)$$

et P la perturbation définie par

$$P(x,y) = S \times \sin(2\pi f_p x + p_p) \times \sin(2\pi f_p y + p_p)$$

avec $p_p = 0$ pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).
On cherche alors principalement à comparer les erreurs en norme L^2 obtenus avec les problèmes $\mathcal{C}_1$ et $\mathcal{C}_2$.
On prendra $S = 0.5$ et $p = 0$ (c'est-à-dire $g = 0$). On fera varier ϵ, f et f_p.
Voici les résultats obtenus :

			PhiFEM	Corr	Corr v2	facteurs
PhiFEM	f=4, fp=2	eps=0.01	0.161997	0.008844	0.000450	19.671352
		eps=0.001	0.161997	0.000884	0.000045	19.671159
	f=6, fp=2	eps=0.01	0.341761	0.009455	0.000445	21.259183
		eps=0.001	0.341761	0.000946	0.000044	21.250331
	f=8, fp=2	eps=0.01	0.521955	0.009495	0.000439	21.607885
		eps=0.001	0.521955	0.000950	0.000044	21.490152
	f=2, fp=4	eps=0.01	0.035073	0.000687	0.001868	0.367876
		eps=0.001	0.035073	0.000069	0.000187	0.367467
	f=2, fp=6	eps=0.01	0.035073	0.002334	0.003959	0.589615
		eps=0.001	0.035073	0.000231	0.000396	0.583925
	f=2, fp=8	eps=0.01	0.035073	0.004498	0.006052	0.743180
		eps=0.001	0.035073	0.000421	0.000605	0.696009

Figure 6 – Résultats sur le carré (nb_vert=64)

			PhiFEM	Corr	Corr v2	facteurs
PhiFEM	f=4, fp=2	eps=0.01	0.042117	0.008710	0.000158	55.065688
		eps=0.001	0.042117	0.000871	0.000016	55.064206
	f=6, fp=2	eps=0.01	0.096873	0.009457	0.000158	59.882736
		eps=0.001	0.096873	0.000946	0.000016	59.905757
	f=8, fp=2	eps=0.01	0.169108	0.009651	0.000158	61.246306
		eps=0.001	0.169108	0.000965	0.000016	61.247752
	f=2, fp=4	eps=0.01	0.008772	0.000183	0.000472	0.387565
		eps=0.001	0.008772	0.000018	0.000047	0.386982
	f=2, fp=6	eps=0.01	0.008772	0.000661	0.001098	0.601830
		eps=0.001	0.008772	0.000066	0.000110	0.602117
	f=2, fp=8	eps=0.01	0.008772	0.001356	0.001938	0.699478
		eps=0.001	0.008772	0.000134	0.000194	0.690160

Figure 7 – Résultats sur le carré (nb_vert=128)

Il semblerait que les résultats obtenus pour le problème $\mathcal{C}_2$ (colonne "Corr v2") soient meilleurs que ceux obtenus pour le problème $\mathcal{C}_1$ (colonne "Corr"). La colonne "facteur" contient les coefficients "Corr"/"Corr v2".

2nd cas test : le cercle
On considère Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$. On prend

$$\phi(x, y) = -1/8 + (x - 1/2)^2 + (y - 1/2)^2$$

On considère le domaine fictif $\mathcal{O} = (0, 1)^2$.
On considère toujours la solution analytique suivante :

$$u_{ex}(x, y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

On prend dans ce cas la perturbation P définie par

$$P(x, y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi) \times \cos(4\pi((x - 0.5)^2 + (y - 0.5)^2))$$

pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).
On cherche comme pour le cas test du carré à comparer les erreurs en norme L^2 obtenues avec les problèmes $\mathcal{C}_1$ et $\mathcal{C}_2$.
On prendra $S = 0.5$ et $p = 0$ (attention ici $g \neq 0$). On fera varier ϵ, f et f_p.

Remarque. *Ici, on prend 2 fois moins de nœuds que dans le cas test du carré pour avoir des cas comparables (car le domain $\mathcal{O}$ est deux fois plus petit sur la longueur et sur la largeur). ATTENTION : Le pb est non homogène ici !*

Voici les résultats obtenus :

			PhiFEM	Corr	Corr v2	facteurs
PhiFEM	f=4, fp=2	eps=0.01	0.012251	0.006089	0.000299	20.396088
		eps=0.001	0.012251	0.000609	0.000030	20.398810
	f=6, fp=2	eps=0.01	0.024760	0.006622	0.000302	21.961475
		eps=0.001	0.024760	0.000662	0.000030	22.052286
	f=8, fp=2	eps=0.01	0.036114	0.006653	0.000297	22.366257
		eps=0.001	0.036114	0.000665	0.000030	22.537918
	f=2, fp=4	eps=0.01	0.003165	0.009198	0.001293	7.112455
		eps=0.001	0.003165	0.000933	0.000129	7.215171
	f=2, fp=6	eps=0.01	0.003165	0.003155	0.002576	1.224708
		eps=0.001	0.003165	0.000307	0.000258	1.191953
	f=2, fp=8	eps=0.01	0.003165	0.007592	0.003775	2.011383
		eps=0.001	0.003165	0.000728	0.000377	1.927707

Figure 8 – Résultats sur le cercle (nb_vert=32)

			PhiFEM	Corr	Corr v2	facteurs
PhiFEM	f=4, fp=2	eps=0.01	0.003184	0.006018	0.000068	87.999890
		eps=0.001	0.003184	0.000602	0.000007	88.004480
	f=6, fp=2	eps=0.01	0.007356	0.006716	0.000071	94.941980
		eps=0.001	0.007356	0.000672	0.000007	94.992693
	f=8, fp=2	eps=0.01	0.012282	0.006719	0.000069	97.292153
		eps=0.001	0.012282	0.000672	0.000007	97.341512
	f=2, fp=4	eps=0.01	0.000767	0.009725	0.000337	28.885409
		eps=0.001	0.000767	0.000988	0.000034	29.349433
	f=2, fp=6	eps=0.01	0.000767	0.005168	0.000753	6.865843
		eps=0.001	0.000767	0.000525	0.000075	6.978587
	f=2, fp=8	eps=0.01	0.000767	0.008193	0.001287	6.367135
		eps=0.001	0.000767	0.000831	0.000129	6.458621

Figure 9 – Résultats sur le cercle (nb_vert=64)

Il semblerait que les résultats obtenus pour le problème $\mathcal{C}_2$ (colonne "Corr v2") soient meilleurs que ceux obtenus pour le problème $\mathcal{C}_1$ (colonne "Corr"). La colonne "facteur" contient les coefficients "Corr"/"Corr v2". On obtient alors le même type de résultat que pour le cas test du carré.

Explication On cherche ici à expliciter la forme de la solution dans le problème $\mathcal{C}_2$.
La décomposition de u_h sur la base $(\varphi_1, \ldots, \varphi_{N_h})$ de V_h s'écrit pour ce problème

$$u_h = \phi(x) C_h + \tilde{\phi}(x) \tag{1}$$

avec

$$C_h = \sum_{i=1}^{N_h} C_i \varphi_i \tag{2}$$

Or

$$C_i = \frac{u(x_i) - \tilde{\phi}(x_i)}{\phi(x_i)} \tag{3}$$

avec

$$u(x_i) = \tilde{\phi}(x_i) + \epsilon P(x_i) \tag{4}$$

et

$$\phi(x) = \phi(x_i) + (x - x_i)\phi'(x_i) \tag{5}$$

Les relations 1 et 2 deviennent alors :

$$\begin{aligned}
u_h &= \tilde{\phi}(x) + \phi(x) C_h \\
&= \tilde{\phi}(x) + \phi(x) \sum_{i=1}^{N_h} C_i \varphi_i \\
&= \tilde{\phi}(x) + \phi(x) \sum_{i=1}^{N_h} \frac{u(x_i) - \tilde{\phi}(x_i)}{\phi(x_i)} \varphi_i \quad \text{par } 3 \\
&= \tilde{\phi}(x) + \phi(x) \sum_{i=1}^{N_h} \frac{\tilde{\phi}(x_i) + \epsilon P(x_i) - \tilde{\phi}(x_i)}{\phi(x_i)} \varphi_i \quad \text{par } 4 \\
&= \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \frac{\phi(x)}{\phi(x_i)} \varphi_i \\
u_h &= \tilde{\phi}(x) + \epsilon \sum_{i=1}^{N_h} P(x_i) \left(1 + \frac{(x - x_i)\phi'(x_i)}{\phi(x_i)}\right) \varphi_i \quad \text{par } 5
\end{aligned} \tag{6}$$

Conclusion

On s'est rendu compte en fin de semaine que les résultats obtenus avaient été obtenus en prenant $\tilde{\phi}$ de degré 10 (et pas de degré 2 comme on pourrait avoir avec le FNO). Les résultats obtenus avec $\tilde{\phi}$ de degré 2 seront à la semaine suivante, ainsi que les résultats obtenus avec le FNO.

14 Semaine 14 : 08/05/2023 - 12/05/2023

Résumé

On souhaite faire fonctionner le rehaussement avec PhiFEM en utilisant la méthode duale. On commencera par faire les courbes de convergence puis on comparera pour différent cas tests, les erreurs en norme L^2 des méthodes suivantes : Φ-FEM par méthode directe, Φ-FEM par méthode duale, Correction par multiplication sans rehaussement, Correction par multiplication avec rehaussement par méthode duale, Correction par addition.

Après discussion avec Emmanuel, il semblerait que les résultats obtenus pour $\tilde{\phi}$ de degré 2 ne soient pas aberrant. On va alors continuer les tests sur le FNO en lui appliquant les mêmes méthodes de correction que celles testées sur la solution analytique.

Après relecture du document sur le rehaussement par Michel, il m'a demandé de faire la théorie sur l'erreur d'interpolation mis dans le document pour le rehaussement (inégalité 7 qui m'avait été dite par Emmanuel). Jeudi, on a alors discuté de ça avec Michel et vendredi j'ai alors relu ce que Michel avait fait pour essayer de bien comprendre.

14.1 Présentation : méthode duale pour ϕ-FEM

On considère le problème de Poisson avec condition de Dirichlet homogène ou non homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = g & \Gamma \end{cases}$$

La formulation variationnelle PhiFEM pour ce problème en utilisant la méthode duale est :

$$\int_{\Omega_h} \nabla u \nabla v - \int_{\partial\Omega_h} \frac{\partial u}{\partial n} v + \frac{\gamma}{h^2} \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \left(u - \frac{1}{h}\phi p\right)\left(v - \frac{1}{h}\phi q\right) + G_h(u,v) = \int_{\Omega_h} fv + \frac{\gamma}{h^2} \sum_{T\in\mathcal{T}_h^\Gamma} \int_T g\left(v - \frac{1}{h}\phi q\right) + G_h^{rhs}(v)$$

avec

$$G_h(u,v) = \sigma h \sum_{E\in\mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial u}{\partial n}\right]\left[\frac{\partial v}{\partial n}\right] + \sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \Delta u \Delta v$$

et

$$G_h^{rhs}(v) = -\sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T f\Delta v$$

Pour la correction, on considère le problème suivant :

$$\begin{cases} -\Delta(\hat{\phi}C) = f & \Omega \\ \hat{u} = g + m & \Gamma \end{cases}$$

avec $\hat{u} = \hat{\phi}C + m$ où $\hat{\phi} = \tilde{\phi} + m$ (m une constante).
La formulation variationnelle PhiFEM pour ce problème en utilisant la méthode duale est :

$$\begin{aligned} \int_{\Omega_h} \nabla(\hat{\phi}C)\nabla(\hat{\phi}v) - \int_{\partial\Omega_h} \frac{\partial}{\partial n}(\hat{\phi}C)\hat{\phi}v + \frac{\gamma}{h^2} \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \left(\hat{\phi}C - \frac{1}{h}\phi p\right)\left(\hat{\phi}v - \frac{1}{h}\phi q\right) + G_h(C,v) \\ = \int_{\Omega_h} f\hat{\phi}v + \frac{\gamma}{h^2} \sum_{T\in\mathcal{T}_h^\Gamma} \int_T (g+m)\left(\hat{\phi}v - \frac{1}{h}\phi q\right) + G_h^{rhs}(v) \end{aligned}$$

avec

$$G_h(C,v) = \sigma h \sum_{E\in\mathcal{F}_h^\Gamma} \int_E \left[\frac{\partial}{\partial n}(\hat{\phi}C)\right]\left[\frac{\partial}{\partial n}(\hat{\phi}v)\right] + \sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T \Delta(\hat{\phi}C)\Delta(\hat{\phi}v)$$

et

$$G_h^{rhs}(v) = -\sigma h^2 \sum_{T\in\mathcal{T}_h^\Gamma} \int_T f\Delta(\hat{\phi}v)$$

Résultats

On considère Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$. On prend

$$\phi(x, y) = -1/8 + (x - 1/2)^2 + (y - 1/2)^2$$

On considère le domaine fictif $O = (0, 1)^2$.
On considère toujours la solution analytique suivante :

$$u_{ex}(x, y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi)$$

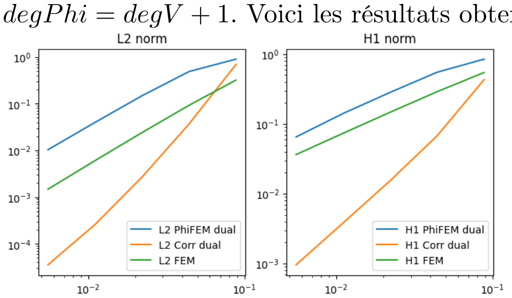
On prend dans ce cas la perturbation P définie par

$$P(x, y) = S \times \sin(2\pi f x + \varphi) \times \sin(2\pi f y + \varphi) \times \cos(4\pi((x - 0.5)^2 + (y - 0.5)^2))$$

pour que $P = 0$ sur Γ (et donc $u_p = u_{ex}$ sur Γ).

Résultats de convergence :

On cherchera à comparer pour un seul cas test les erreurs en norme L^2 et H^1 de ϕ-FEM (sans correction) en utilisant la méthode duale, de la correction avec un rehaussement en utilisant la méthode duale et l'erreur FEM standard (sans correction). On prendra $S = S_p = 0.5$, $f = 4$, $f_p = 2$ et $p = p_p = 0$ (à noter que sur le cercle ce problème n'est pas homogène). On choisit $\epsilon = 1e-2$ et $m = 1000$ pour le rehaussement. On prendra $degV = 1$ et $degPhi = degV + 1$. Voici les résultats obtenus pour différentes tailles de maillage h :



PhiFEM

h	L2	H1
0.088388	0.896113	0.848248
0.044194	0.485780	0.550728
0.022097	0.147528	0.283167
0.011049	0.039635	0.141269
0.005524	0.010338	0.065197

Correction

h	L2	H1
0.088388	0.689443	0.431876
0.044194	0.036653	0.067565
0.022097	0.002680	0.015365
0.011049	0.000259	0.003817
0.005524	0.000036	0.000959

FEM

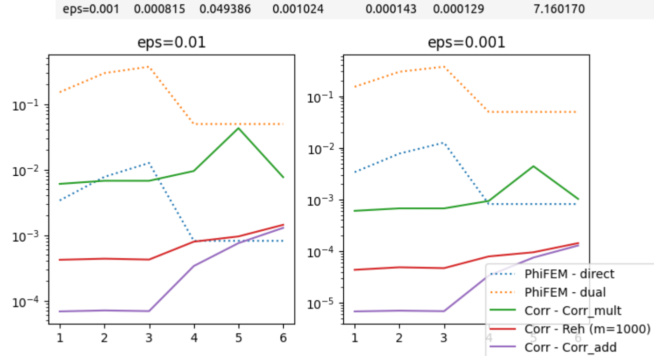
h	L2	H1
0.088142	0.318042	0.544746
0.044176	0.092040	0.289509
0.022096	0.024006	0.147011
0.011046	0.006033	0.073511
0.005524	0.001477	0.036407

Comparaison des méthodes :

On cherche cette fois-ci à compléter le document de la semaine dernière pour ϕ-FEM en comparant les erreurs en norme L^2 pour les méthodes suivantes : Φ-FEM par méthode directe, Φ-FEM par méthode duale, Correction par multiplication sans rehaussement ($\tilde{u} = \tilde{\phi} C$), Correction par multiplication avec rehaussement par méthode duale ($\tilde{u} = \tilde{\phi} C + m$), Correction par addition ($\tilde{u} = \tilde{\phi} + \phi C$).

Remarque. *On prendra ici* $\tilde{\phi}$ *de degré 10 et* $\sigma = \gamma = 20$.

			PhiFEM		Corr			Facteur	
			direct	dual	Corr_mult	Reh (m=1000)	Corr_add	Corr_mult/Reh	Corr_mult/Corr_add
PhiFEM	f=4, fp=2	eps=0.01	0.003372	0.151095	0.006028	0.000419	0.000068	14.382901	88.145595
		eps=0.001	0.003372	0.151095	0.000603	0.000044	0.000007	13.818186	88.148858
	f=6, fp=2	eps=0.01	0.007717	0.295148	0.006728	0.000436	0.000071	15.447764	95.118201
		eps=0.001	0.007717	0.295148	0.000673	0.000049	0.000007	13.805241	95.168640
	f=8, fp=2	eps=0.01	0.012569	0.369731	0.006723	0.000422	0.000069	15.922067	97.344606
		eps=0.001	0.012569	0.369731	0.000672	0.000047	0.000007	14.320722	97.393895
	f=2, fp=4	eps=0.01	0.000815	0.049386	0.009475	0.000792	0.000337	11.967641	28.142119
		eps=0.001	0.000815	0.049386	0.000936	0.000079	0.000034	11.826681	27.809971
	f=2, fp=6	eps=0.01	0.000815	0.049386	0.042686	0.000953	0.000753	44.807890	56.704457
		eps=0.001	0.000815	0.049386	0.004396	0.000095	0.000075	46.156493	58.392909
	f=2, fp=8	eps=0.01	0.000815	0.049386	0.007632	0.001431	0.001287	5.333027	5.930548
		eps=0.001	0.000815	0.049386	0.001024	0.000143	0.000129	7.160170	7.959555



14.2 Résultats FNO

On considère le problème de Poisson avec condition de Dirichlet homogène :

$$\begin{cases} -\Delta u = f & \Omega \\ u = 0 & \Gamma \end{cases}$$

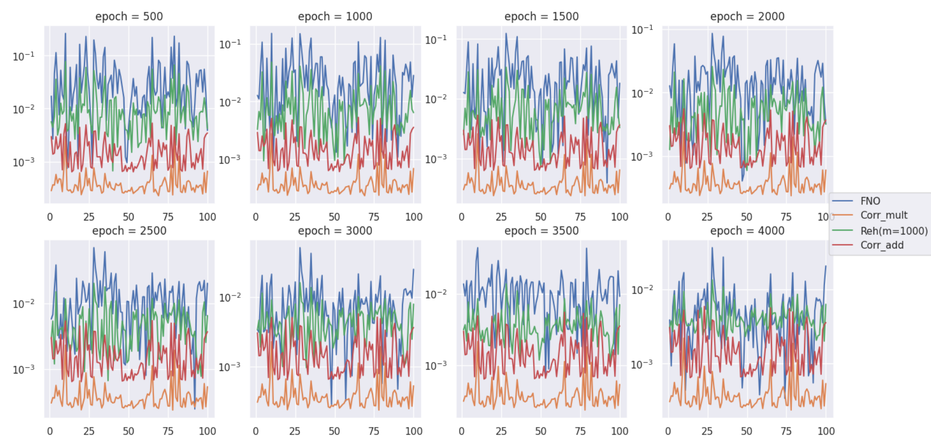
On considère cette fois-ci f gaussienne :

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2 + (y-\mu_1)^2}{2\sigma^2}\right),$$

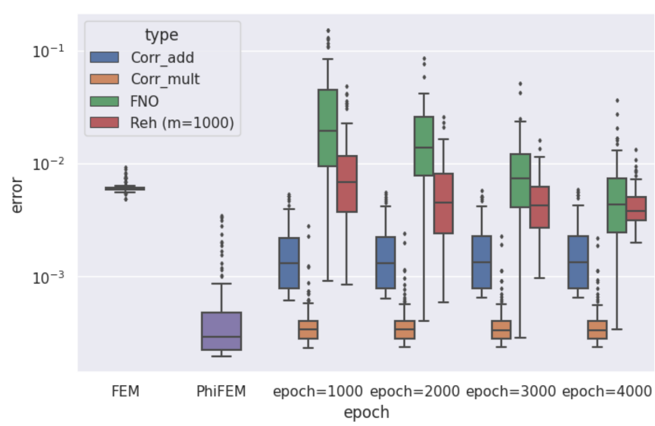
avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5 - \sqrt{2}/4, 0.5 + \sqrt{2}/4])$ à condition que $\phi(\mu_0, \mu_1) < -0.05$.
On considère la solution de référence u_{ref} comme étant une solution sur-raffinée $\mathbb{P}^1$ obtenue par les EF standard (avec $h_{ref} \approx 0.006$ car $h_{ref} << h_{FNO}$).

On cherche à appliquer sur le FNO les 3 "types " de correction : Correction par multiplication sans rehaussement ($\tilde{u} = \tilde{\phi}C$), Correction par multiplication avec rehaussement par méthode duale ($\tilde{u} = \tilde{\phi}C + m$), Correction par addition ($\tilde{u} = \tilde{\phi} + \phi C$).

Erreurs en norme L^2 à différentes epochs



Représentation en boxplots



Conclusion

Plusieurs autres idées de tests ont été proposé par Michel et Emmanuel. Ces tests seront expliqué dans le Résumé de la semaine prochaine. La semaine prochaine, il faudra également continuer la théorie sur le rehaussement en posant quelques questions à Michel.

Vanessa a également proposée une nouvelle idée qui consiste à combiner les deux méthodes de correction, en posant quelque chose comme $\tilde{f} = \hat{\phi}C_1 + C_2$. Il semblerait d'après Michel et Killian que le problème soit mal posé.

15 Semaine 15 : 15/05/2023 - 19/05/2023

Résumé

(Killian est en vacances pour les deux prochaines semaines)

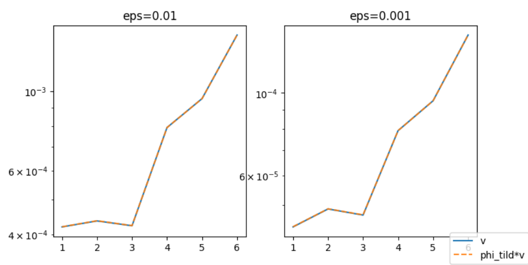
Au tout début de la semaine, j'ai fait un petit test où je comparais la méthode duale pour le rehaussement en prenant comme fonction test v ou $\tilde{\phi}v$. On s'est rendu compte que cela nous donnait les mêmes résultats.

Comme Emmanuel était de retour cette semaine, on a fait une réunion mardi avec Michel. Après avoir montré mes résultats sur le FNO, on a cherché des solutions afin de comprendre pourquoi ces résultats sont très différents des résultats obtenus sur la solution analytique. Dans un premier temps, ils ont proposés de tester d'implémenter un réseau multiperceptron qui nous permet d'obtenir une solution en tout point de notre domaine et donc de pouvoir tester la correction avec $\tilde{\phi}$ de de plus haut degré (10 par exemple). Par la suite, ils ont proposés une seconde idée, plus simple, qui consiste à construire une solution analytique à partir de la sortie du FNO. Ils ont proposés de tester avec Fourier, Legendre ou Hermite. J'ai alors commencé à tester avec Fourier cette semaine, les explications et résultats seront mis dans le suivi de la semaine suivante.

De plus, j'ai repris les explications de Michel sur la preuve de l'inégalité obtenu pour le rehaussement et ait rédigé un petit document qui n'est pas encore terminé. Je vais mettre ce document ici.

15.1 Comparaison fonction test - Rehaussement

Résultats obtenus :



15.2 Estimation d'erreur - Problème rehaussé (Modifié !)

On se place ici dans le cadre de FEM standard.

Problème initital :

On considère initialement le problème de Poisson avec condition de Dirichlet homogène ou non homogène :

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

Problème considéré :

Dans notre cas, on souhaite appliquer une correction à la sortie d'un FNO. On va considérer ici que l'on possède une solution analytique u_{ex} et qu'après une utilisation du FNO, on obtient une solution du type

$$\tilde{\phi}(x,y) = u_{ex}(x,y) - \epsilon P(x,y)$$

avec P la perturbation (tel que $P = 0$ sur Γ) et ϵ petit.

On considère

$$\hat{\phi} = \tilde{\phi} + m = u_{ex} - \epsilon P + m = \widehat{u_{ex}} - \epsilon P$$

avec $\widehat{u_{ex}} = u_{ex} + m$ et m une constante.

On souhaite alors résoudre le problème suivant :

$$\begin{cases} -\Delta(\hat{\phi}C) = f & \Omega \\ (\hat{\phi}C) = g + m & \Gamma \end{cases} \tag{$\mathcal{C}$}$$

On pose alors

$$\hat{u} = \hat{\phi}C$$

But du document :

Démonter la propriété suivante :

$$||\hat{u} - \hat{u_h}||_0 \leq ch^{k+1}||\hat{\phi}||_\infty \, |C|_{k+1} \tag{1}$$

Problèmes variationnels :

Problème variationnel :

$$\text{Trouver } \hat{u} \in V \text{ tel que } a(\hat{u}, v) = l(v), \ \forall v \in V$$

Problème variationnel approché :

$$\text{Trouver } \hat{u_h} \in V_h \text{ tel que } a(\hat{u_h}, v_h) = l(v_h), \ \forall v_h \in V_h$$

15.2.1 Partie 1 : norme H^1

Comme V_h est un sous espace vectoriel de V, en posant $v = v_h$, on obtient :

$$a(\hat{\phi}C, \hat{\phi}v_h) - a(\hat{\phi}C_h, \hat{\phi}v_h) = 0 \quad \forall v_h \in V_h$$

On a alors l'orthogonalité de Galerkin : (ATTENTION : Abus de notation sur v_h !)

$$a(\hat{u} - \hat{u_h}, v_h) = 0 \quad \forall v_h \in V_h$$

On a alors

$$\begin{aligned}
\nu||\hat{u} - \hat{u_h}||_1^2 &\leq \alpha a(\hat{u} - \hat{u_h}, \hat{u} - \hat{u_h}) && \text{par coercivité} \\
&= \alpha a(\hat{u} - \hat{u_h}, \hat{u} - I_h\hat{u} + I_h\hat{u} - \hat{u_h}) && \\
&= \alpha a(\hat{u} - \hat{u_h}, \hat{u} - I_h\hat{u}) && \text{par orthogonalité de Galerkin en prenant } v_h = \hat{u_h} - I_h\hat{u} \\
&\leq \alpha|\hat{u} - \hat{u_h}|_1|\hat{u} - I_h\hat{u}|_1 && \text{par continuité} \\
&\leq \alpha||\hat{u} - \hat{u_h}||_1|\hat{u} - I_h\hat{u}|_1 &&
\end{aligned}$$

Ainsi

$$||\hat{u} - \hat{u_h}||_1 \leq \alpha|\hat{u} - I_h\hat{u}|_1$$

Or

$$|\hat{u} - I_h\hat{u}|_1 = |(C - I_hC)\hat{\phi}|_1$$

En posant $A = C - I_hC$, on a

$$|A\hat{\phi}|_1 = ||(A\hat{\phi})'||_0 = ||A'\hat{\phi} + A\hat{\phi}'||_0 \leq ||A'\hat{\phi}||_0 + ||A\hat{\phi}'||_0$$

Comme

$$||A'\hat{\phi}||_0 = \sqrt{\int_\Omega (A'\hat{\phi})^2} \leq \max_\Omega \hat{\phi} \sqrt{\int_\Omega (A')^2} = ||\hat{\phi}||_\infty |A|_1 \leq ||\hat{\phi}||_\infty ||A||_1$$

et

$$||A\hat{\phi}'||_0 = \sqrt{\int_\Omega (A\hat{\phi}')^2} \leq \max_\Omega \hat{\phi}' \sqrt{\int_\Omega (A)^2} = ||\hat{\phi}'||_\infty ||A||_0 \leq \alpha||\hat{\phi}||_\infty ||A||_1$$

Ainsi

$$|A\hat{\phi}|_1 \leq \alpha||\hat{\phi}||_\infty ||A||_1$$

Donc

$$|\hat{u} - I_h\hat{u}|_1 = |(C - I_hC)\hat{\phi}|_1 \leq \alpha||\hat{\phi}||_\infty ||C - I_hC||_1$$

Finalement en utilisant l'inégalité d'interpolation, on obtient

$$\boxed{||\hat{u} - \hat{u_h}||_1 \leq \alpha h^k ||\hat{\phi}||_\infty |C|_{k+1}} \tag{2}$$

15.2.2 Partie 2 : norme L^2

On applique la méthode de dualité d'Aubin-Nitsche. On considère le problème dual :
Soit $\hat{z} \in H_0^1(\Omega)$ solution du problème

$$\begin{cases} -\Delta \hat{z} = \hat{e_h} & \text{dans } \Omega \\ \hat{z} = 0 & \text{sur } \Gamma \end{cases}$$

avec $\hat{e_h} = \hat{u} - \hat{u_h}$. Alors

$$a(u,v) = -\int_\Omega \Delta u \cdot v = \int_\Omega \nabla u \cdot \nabla v$$

et comme $e_h \in H_0^1(\Omega)$

$$a(\hat{z}, \hat{e_h}) = \int_\Omega (-\Delta \hat{z}) \cdot \hat{e_h} = \int_\Omega \hat{e_h}^2 = ||\hat{e_h}||_0^2 \tag{3}$$

De plus, par les propriétés de régularité :

$$\hat{z} \in H^2(\Omega) \tag{4}$$

et

$$||\hat{z}||_2 \leq \alpha ||\hat{e_h}||_0 = \alpha ||\hat{u} - \hat{u_h}||_0 \tag{5}$$

Ainsi

$$\begin{aligned} ||\hat{u} - \hat{u_h}||_0^2 &= ||\hat{e_h}||_0^2 \\ &= a(\hat{z}, \hat{e_h}) && \text{par } 3 \\ &= a(\hat{z} - I_h\hat{z}, \hat{e_h}) && \text{par orthogonalité de Galerkin} \\ &\leq \alpha |\hat{z} - I_h\hat{z}|_1 |\hat{e_h}|_1 && \text{par continuité} \\ &\leq \alpha h |\hat{z}|_2 |\hat{e_h}|_1 && \text{par } 4 \text{ et par inégalité d'interpolation} \\ &\leq \alpha \cdot h|\hat{z}|_2 \cdot h^k ||\hat{\phi}||_\infty |C|_{k+1} && \text{par } 2 \\ &\leq \alpha h^{k+1} ||\hat{u} - \hat{u_h}||_0 ||\hat{\phi}||_\infty |C|_{k+1} && \text{par } 5 \end{aligned}$$

Finalement

$$\boxed{||\hat{u} - \hat{u_h}||_0 \leq \alpha h^{k+1} ||\hat{\phi}||_\infty |C|_{k+1}} \tag{6}$$

Remarque. *On notera que*

$$||\hat{u} - \hat{u_h}||_0 = ||u + m - (u_h + m)||_0 = ||u - u_h||_0$$

Conclusion

Les tests effectués avec Fourier pour obtenir une solution analytique à partir de la solution produite par le FNO seront mis à la semaine prochaine.

Il faudra continuer la preuve commencé cette semaine mais ce n'est pas une priorité.

(Ce document a été modifié.)

16 Semaine 16 : 22/05/2023 - 26/05/2023

Résumé

On a entraîné un FNO avec des solutions $\mathbb{P}^2$ où nb_vert=32. Une fois le réseau entraîné, on a cherché à appliquer divers types de correction où les résultats obtenus ne semblaient pas correspondre avec ceux obtenus sur la solution analytique. C'est pourquoi, on cherche ici à récupérer la sortie du FNO pour ensuite construire une solution analytique en utilisant les polynômes de Legendre. Finalement, on testera à nouveau les différents types de correction sur une solution $\mathbb{P}^{10}$. La première étape consistait alors à tester en 1D puis en 2D. J'ai testé dans un premier temps sur une solution analytique avec des séries/transformées de Fourier. N'ayant pas abouti, je suis passé aux polynômes de Legendre.

16.1 Transformée de Fourier

16.1.1 Transformée de Fourier en 1D

Transformée de Fourier discrète

La Transformée de Fourier discrète (en 1D) est définie par :

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{-2i\pi x\frac{u}{N}}$$

L'inverse de la Transformée de Fourier discrète (en 1D) est définie par :

$$f(x) = \sum_{u=0}^{N-1} F(u)e^{2i\pi\frac{u}{N}x}$$

Montrons que f est bien la fonction réciproque de F. On a

$$\begin{aligned} f(x) &= \sum_{u=0}^{N-1} F(u)e^{2i\pi\frac{u}{N}x} \\ &= \frac{1}{N}\sum_{u=0}^{N-1}\sum_{x'=0}^{N-1} f(x')e^{-2i\pi x'\frac{u}{N}}e^{2i\pi\frac{u}{N}x} \\ &= \frac{1}{N}\sum_{x'=0}^{N-1} f(x')\underbrace{\sum_{u=0}^{N-1} e^{2i\pi\frac{u}{N}(x-x')}}_{:= S} \end{aligned}$$

Alors

- Si $x = x'$: $S = \sum_{u=0}^{N-1} 1 = N$
- Si $x \neq x'$:

$$S = \sum_{u=0}^{N-1}\left(e^{\frac{2i\pi}{N}(x-x')}\right)^u = \frac{1-\left(e^{\frac{2i\pi}{N}(x-x')}\right)^N}{1-e^{\frac{2i\pi}{N}(x-x')}} = \frac{1-e^{2i\pi(x-x')}}{1-e^{\frac{2i\pi}{N}(x-x')}} = 0$$

car

$$e^{2i\pi(x-x')} = cos(2\pi(x-x')) + isin(2\pi(x-x')) = 1+0 = 1$$

On en déduit que

$$f(x) = \frac{1}{N}\times Nf(x) = f(x)$$

Voici l'implémentation Python de la transformée de Fourier discrète et de son inverse :

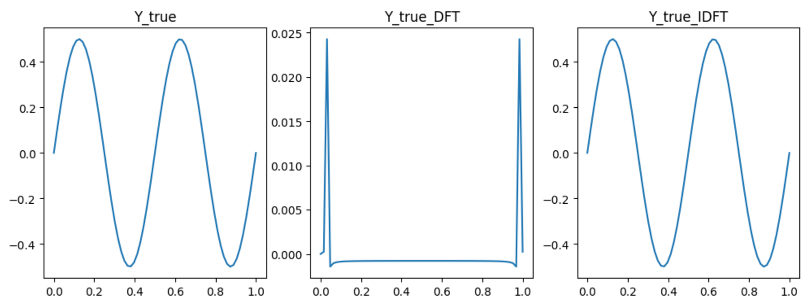
```python
# Discrete Fourier Transform
def DFT(f):
    N = f.shape[0]
    F = np.zeros(N,dtype=complex)
    for u in range(N):
        for x in range(N):
            F[u]+=f[x]*cmath.exp(-2j*np.pi*x*u/N)
    return 1/N*F
```

```
# Inverse Discrete Fourier Transform
def IDFT(F):
    N = F.shape[0]
    f = np.zeros(N,dtype=complex)
    for x in range(N):
        for u in range(N):
            f[x]+=F[u]*cmath.exp(2j*np.pi*u/N*x)
    return f
```

Résultats :
On prendra

$$u_{ex}(x) = S\sin(2\pi f x + p)$$

avec $S = 0.5$,$f = 2$ et $p = 0$.
On peut y appliquer la DFT puis reconstruire la fonction originale en y appliquant la DFT inverse.
Voici les résultats obtenus avec nb_vert=64 et $x \in [0, 1]$:



Transformée de Fourier continue

La Transformée de Fourier continue (en 1D) est définie par :

$$F(u) = \int_0^L f(x)e^{-2i\pi ux}dx$$

L'inverse de la Transformée de Fourier continue (en 1D) est définie par :

$$f(x) = \int_0^L F(u)e^{2i\pi ux}du$$

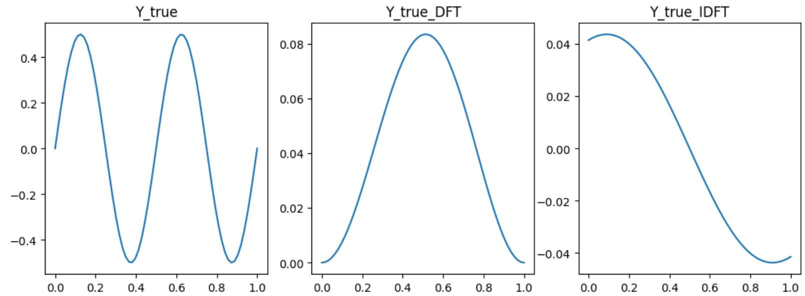
Par la méthode des rectangles, on a :

$$F(u) \approx \frac{L}{N}\sum_{k=0}^{N-2} f(x_k)e^{-2i\pi ux_k} \quad \text{et} \quad f(x) \approx \frac{L}{N}\sum_{k=0}^{N-2} F(u_k)e^{2i\pi u_k x}$$

Voici l'implémentation Python de la transformée de Fourier continue et de son inverse :

```
# Fourier Transform
def DFT(f,XX):
    N = f.shape[0]
    F = np.zeros(N,dtype=complex)
    for (k,u) in enumerate(XX):
        for (l,x) in enumerate(XX[:-1]):
            F[k]+=f[l]*cmath.exp(-2j*np.pi*x*u)
    return 1/N*F

# Inverse Fourier Transform
def IDFT(F,XX):
    N = F.shape[0]
    f = np.zeros(N,dtype=complex)
    for (k,x) in enumerate(XX):
        for (l,u) in enumerate(XX[:-1]):
            f[k]+=F[l]*cmath.exp(2j*np.pi*u*x)
    return 1/N*f
```

Résultats :
En prenant le même cas test que pour la transformée de Fourier discrète, voici les résultats obtenus :



Remarque. *Il semblerait que la discrétisation du problème ne fonctionne pas. En effet dans la preuve que f est bien la fonction réciproque de F faite au-dessus (pour le cas discret), lorsque $x \neq x'$, on obtenait une série géométrique en faisant passer en exposant u. Dans notre cas, lorsque $x_k \neq x$, on ne peut pas mettre en exposant u, ce n'est pas un entier. Il faudrait trouver une solution à ce problème !*

16.1.2 Transformée de Fourier en 2D

Transformée de Fourier discrète

La Transformée de Fourier discrète (en 2D) est définie par :

$$F(u,v) = \frac{1}{N^2}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{-i\frac{2\pi}{N}(ux+vy)}$$

L'inverse de la Transformée de Fourier discrète (en 2D) est définie par :

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{i\frac{2\pi}{N}(ux+vy)}$$

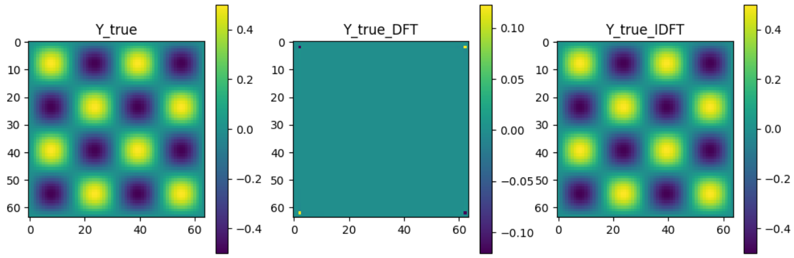
Voici l'implémentation Python de la transformée de Fourier discrète et de son inverse :

```
# Discrete Fourier Transform
def DFT(img):
    N = img.shape[0]
    img_DFT = np.zeros([N,N],dtype=complex)
    for u in range(N):
        for v in range(N):
            for x in range(N):
                for y in range(N):
                    img_DFT[u,v]+=img[x,y]*cmath.exp(-2j*np.pi/N*(u*x+v*y))
    return 1/N**2*img_DFT

# Inverse Discrete Fourier Transform
def IDFT(img_DFT): # Inverse Discrete Fourier Transform
    N = img_DFT.shape[0]
    img = np.zeros([N,N],dtype=complex)
    for x in range(N):
        for y in range(N):
            for u in range(N):
                for v in range(N):
                    img[x,y]+=img_DFT[u,v]*cmath.exp(2j*np.pi/N*(u*x+v*y))
    return img
```

Résultats :
On prendra

$$u_{ex}(x,y) = S\sin(2\pi f x + p)\sin(2\pi f y + p)$$

avec $S = 0.5$,$f = 2$ et $p = 0$.
On peut y appliquer la DFT puis reconstruire l'image originale en y appliquant la DFT inverse.
Voici les résultats obtenus avec nb_vert=64 et $x, y \in [0, 1]$:



Transformée de Fourier continue
Même problème que dans le cas 1D. A régler !

16.2 Polynômes de Legendre

16.2.1 Polynômes de Legendre en 1D

On cherche à décomposer une fonction en une série de polynômes de Legendre de la manière suivante :

$$f(x) = \sum_{p=0}^{P-1} \alpha_p P_p(x) \tag{1}$$

où les polynômes de Legendre sont définis pour tout $x \in \mathbb{R}$ par

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n}[(x^2-1)^n]$$

On notera que les polynômes de Legendre sont orthogonaux dans l'espace $L^2(]-1,1[)$ et plus précisément

$$\int_{-1}^{1} P_n(x) P_m(x) dx = \frac{2}{2n+1}\delta_{nm} \tag{2}$$

On pose $X = (x_0, \ldots, x_{N-1})$.
Ainsi

$$f(x_i) = \sum_{p=0}^{P-1} \alpha_p P_p(x_i), \quad \forall i \in \{0, \ldots, N-1\}$$

On pose

$$\widetilde{F_N} = \begin{pmatrix} f(x_0) \\ \vdots \\ f(x_{N-1}) \end{pmatrix} \in \mathbb{R}^N, \quad \widetilde{\alpha_P} = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{P-1} \end{pmatrix} \in \mathbb{R}^P, \quad \widetilde{P_N} = \begin{pmatrix} P_0(x_0) & \ldots & P_{P-1}(x_0) \\ \vdots & \ddots & \vdots \\ P_0(x_{N-1}) & \ldots & P_{P-1}(x_{N-1}) \end{pmatrix} \in \mathcal{M}_{N,P}(\mathbb{R})$$

Par la méthode des rectangles (à gauche), 2 devient

$$\begin{aligned} \int_{-1}^{1} P_n(x) P_m(x) dx &= \sum_{i=0}^{N-2} \int_{x_i}^{x_{i+1}} P_n(x) P_m(x) dx \\ &\approx \sum_{i=0}^{N-2} (x_{i+1} - x_i) P_n(x_i) P_m(x_i) \\ &= \frac{2}{N-1} \sum_{i=0}^{N-2} P_n(x_i) P_m(x_i) \approx \frac{2}{2n+1}\delta_{nm} \end{aligned}$$

On en déduit PB 1

$$\frac{2}{N-1} \widetilde{P_N}^T \widetilde{P_N} = \widetilde{D_P} \tag{3}$$

avec

$$\widetilde{D_P} = diag\left(\frac{2}{2p+1}, p = 0, \dots, P-1\right) \in \mathcal{M}_P(\mathbb{R})$$

On cherche à déterminer $\widetilde{\alpha_P}$. Pour cela, on va réécrire le problème 1 sous la forme matricielle suivante

$$\begin{aligned}
& \widetilde{P_N}\widetilde{\alpha_P} = \widetilde{F_N} \\
\iff \quad & \widetilde{P_N}^T\widetilde{P_N}\widetilde{\alpha_P} = \widetilde{P_N}^T\widetilde{F_N} \\
\iff \quad & \frac{N-1}{2}\widetilde{D_P}\widetilde{\alpha_P} = \widetilde{P_N}^T\widetilde{F_N} \\
\iff \quad & \boxed{\widetilde{\alpha_P} = \frac{2}{N-1}\widetilde{D_P}^{-1}\widetilde{P_N}^T\widetilde{F_N}}
\end{aligned}$$

avec

$$\widetilde{D_P}^{-1} = diag\left(\frac{2p+1}{2}, p = 0, \dots, P-1\right) \in \mathcal{M}_P(\mathbb{R})$$

Autrement dit

$$\forall p \in \{0, \dots, P-1\}, \; \alpha_p = \frac{2p+1}{2} < f(X), P_p(X) >_{L^2(]-1,1[)}$$

<u>Résultats :</u>
On prendra

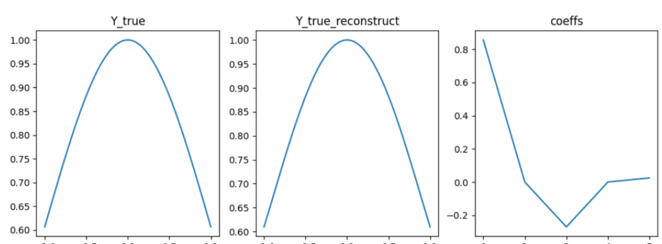
$$u(x) = exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

avec $\mu = 0$ et $\sigma = 1$.
On prendra $P = 5$ et $N = 100$ et $x \in [-1, 1]$.
On commence par vérifier l'orthogonalité des polynômes de Legendre :

```
diag_exact :  [2.         0.66666667 0.4        0.28571429 0.22222222]
diag :  [2.         0.66680271 0.4004081  0.28653027 0.22358128]
max hors diag 0.000883985926341084
```

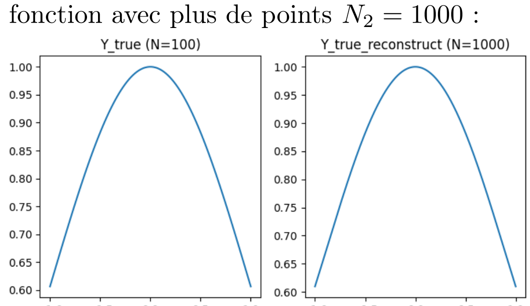
Remarque. *A noter que pour calculer le produit scalaire sur $L^2(]-1,1[)$, on utilisera la méthode des trapèzes.*

Voici les résultats obtenus pour la reconstruction de la fonction en utilisant la décomposition en une série de polynômes de Legendre :



max(|Y_true-Y_true_reconstruct|)=0.00296

On peut également reconstruire la fonction avec plus de points $N_2 = 1000$:



En faisant varier N et P, on remarque que l'approximation semble dépendre énormément de P :

	N=100	N=200	N=400	N=600	N=800	N=1000
p=2	0.249073	0.249089	0.249092	0.249093	0.249093	0.249094
p=4	0.021579	0.021718	0.021753	0.021759	0.021761	0.021762
p=6	0.002964	0.001566	0.001221	0.001158	0.001135	0.001125
p=8	0.007179	0.001747	0.000405	0.000157	0.000071	0.000032
p=10	0.019482	0.004825	0.001201	0.000534	0.000300	0.000193

16.2.2 Polynômes de Legendre en 2D

On cherche à décomposer une fonction en une série de polynômes de Legendre de la manière suivante :

$$f(x,y) = \sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} \alpha_{p,q} P_p(x) P_q(y) \tag{4}$$

où les polynômes de Legendre sont définis pour tout $x \in \mathbb{R}$ par

$$P_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}[(x^2-1)^n]$$

On notera que les polynômes de Legendre sont orthogonaux dans l'espace $L^2(]-1,1[)$ et plus précisément

$$\int_{-1}^{1} P_n(x)P_m(x)dx = \frac{2}{2n+1}\delta_{nm} \tag{5}$$

On pose $X = (x_0, \ldots, x_{N-1})$ et $Y = (y_0, \ldots, y_{M-1})$.
Ainsi

$$f(x_i, y_j) = \sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} \alpha_{p,q} P_p(x_i) P_q(y_j), \quad \forall i \in \{0,\ldots,N-1\}, j \in \{0,\ldots,M-1\}$$

On pose

$$\widetilde{F_{N,M}} = \begin{pmatrix} f(x_0,y_0) & \ldots & f(x_0,y_{M-1}) \\ \vdots & \ddots & \vdots \\ f(x_{N-1},y_0) & \ldots & f(x_{N-1},y_{M-1}) \end{pmatrix} \in \mathcal{M}_{N,M}(\mathbb{R})$$

$$\widetilde{\alpha_{P,Q}} = \begin{pmatrix} \alpha_{0,0} & \ldots & \alpha_{0,Q-1} \\ \vdots & \ddots & \vdots \\ \alpha_{P-1,0} & \ldots & \alpha_{P-1,Q-1} \end{pmatrix} \in \mathcal{M}_{P,Q}(\mathbb{R})$$

$$\widetilde{P_N} = \begin{pmatrix} P_0(x_0) & \ldots & P_{P-1}(x_0) \\ \vdots & \ddots & \vdots \\ P_0(x_{N-1}) & \ldots & P_{P-1}(x_{N-1}) \end{pmatrix} \in \mathcal{M}_{N,P}(\mathbb{R})$$

$$\widetilde{P_M} = \begin{pmatrix} P_0(y_0) & \ldots & P_{Q-1}(y_0) \\ \vdots & \ddots & \vdots \\ P_0(y_{M-1}) & \ldots & P_{Q-1}(y_{M-1}) \end{pmatrix} \in \mathcal{M}_{M,Q}(\mathbb{R})$$

Par la méthode des rectangles (à gauche), 5 devient

$$\begin{aligned}\int_{-1}^{1} P_n(x)P_m(x)dx &= \sum_{i=0}^{N-2}\int_{x_i}^{x_{i+1}} P_n(x)P_m(x)dx \\ &\approx \sum_{i=0}^{N-2}(x_{i+1}-x_i)P_n(x_i)P_m(x_i) \\ &= \frac{2}{N-1}\sum_{i=0}^{N-2} P_n(x_i)P_m(x_i) \approx \frac{2}{2n+1}\delta_{nm}\end{aligned}$$

On en déduit PB 1

$$\frac{2}{N-1}\widetilde{P_N}^T\widetilde{P_N} = \widetilde{D_P} \quad \text{et} \quad \frac{2}{M-1}\widetilde{P_M}^T\widetilde{P_M} = \widetilde{D_Q} \tag{6}$$

avec

$$\widetilde{D_P} = diag\left(\frac{2}{2p+1}, p=0,\ldots,P-1\right) \in \mathcal{M}_P(\mathbb{R}) \quad \text{et} \quad \widetilde{D_Q} = diag\left(\frac{2}{2q+1}, q=0,\ldots,Q-1\right) \in \mathcal{M}_Q(\mathbb{R})$$

On cherche à déterminer $\widetilde{\alpha_{P,Q}}$. Pour cela, on va réécrire le problème 4 sous la forme matricielle suivante

$$\begin{aligned}
& \widetilde{P_N}\widetilde{\alpha_{P,Q}}\widetilde{P_M}^T = \widetilde{F_{N,M}} \\
\Longleftrightarrow\quad & \widetilde{P_N}^T\widetilde{P_N}\widetilde{\alpha_{P,Q}}\widetilde{P_M}^T\widetilde{P_M} = \widetilde{P_N}^T\widetilde{F_{N,M}}\widetilde{P_M} \\
\Longleftrightarrow\quad & \frac{(N-1)(M-1)}{4}\widetilde{D_P}\widetilde{\alpha_{P,Q}}\widetilde{D_Q} = \widetilde{P_N}^T\widetilde{F_{N,M}}\widetilde{P_M} \\
\Longleftrightarrow\quad & \boxed{\widetilde{\alpha_{P,Q}} = \frac{4}{(N-1)(M-1)}\widetilde{D_P}^{-1}\widetilde{P_N}^T\widetilde{F_{N,M}}\widetilde{P_M}\widetilde{D_Q}^{-1}}
\end{aligned}$$

avec

$$\widetilde{D_P}^{-1} = diag\left(\frac{2p+1}{2}, p=0,\dots,P-1\right) \in \mathcal{M}_P(\mathbb{R}) \quad \text{et} \quad \widetilde{D_Q}^{-1} = diag\left(\frac{2q+1}{2}, q=0,\dots,Q-1\right) \in \mathcal{M}_Q(\mathbb{R})$$

Problèmes :

1. $\widetilde{P_N}^T\widetilde{P_N}$ fait un cran de trop ! On a un rectangle supplémentaire !
2. Choisir P et Q : si on les prend trop grand, résultats incohérents !!!

Remarque. *Pour $x \in [a,b]$, on fait un changement de variable pour se ramener dans l'intervalle $[-1,1]$. On pose alors*

$$\tilde{x} = \frac{2}{b-a}x + \frac{a+b}{a-b}$$

Ainsi

$$\tilde{f}(\tilde{x}) = f(x)$$

Résultats :
On prendra

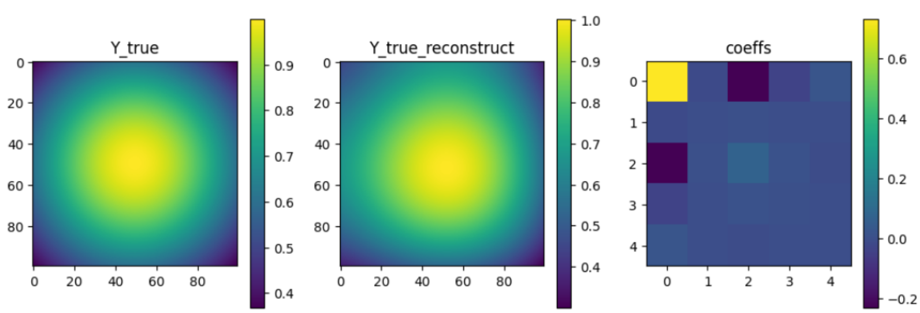
$$u(x) = exp\left(-\frac{(x-\mu)^2+(y-\mu)^2}{2\sigma^2}\right)$$

avec $\mu = 0$ et $\sigma = 1$.
On prendra $P, Q = 5$ et $N = 100$ et $x \in [-1,1]$.
On commence par vérifier l'orthogonalité des polynômes de Legendre :

```
diag_exact :  [2.         0.66666667 0.4        0.28571429 0.22222222]
diag :  [2.         0.66680271 0.4004081  0.28653027 0.22358128]
diff :  0.00135905948651735
max hors diag 0.0008839859263412319
```

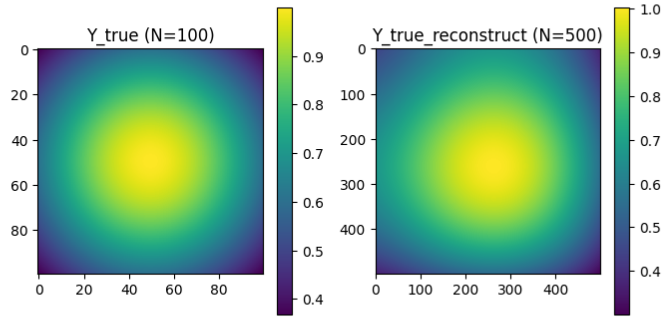
Remarque. *A noter que contrairement au cas 1D, pour calculer le produit scalaire sur $L^2(]-1,1[)$, on utilise cette fois le produit matricielle $\frac{2}{N-1}P_N^T P_N$. En fait on ne considérera pas réellement P_N^T mais une matrice $P_N^{T,-}$ qui est égale à P_N^T avec sa dernière colonne nulle.*

Voici les résultats obtenus pour la reconstruction de l'image en utilisant la décomposition en une série de polynômes de Legendre :



max(|Y_true-Y_true_reconstruct|)=0.0820

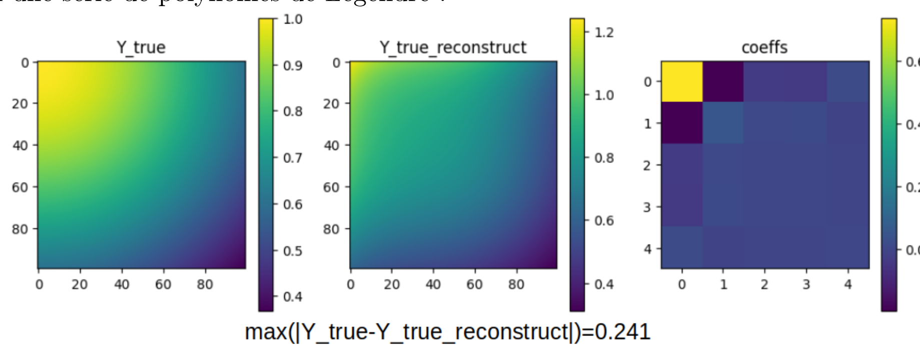
On peut également reconstruire la fonction avec plus de points $N_2 = 500$:



En faisant varier N et P, on remarque que l'approximation semble dépendre énormément de P :

	N=100	N=200	N=400	N=600	N=800	N=1000
p=2	0.395968	0.379936	0.372036	0.369420	0.368115	0.367334
p=4	0.093632	0.060594	0.043462	0.037662	0.034745	0.033037
p=6	0.176989	0.083843	0.041304	0.027702	0.021007	0.017024
p=8	0.328121	0.147647	0.069914	0.045750	0.033987	0.027031
p=10	0.559439	0.238945	0.110070	0.071366	0.052779	0.041869

En prenant cette fois-ci $x \in [0,1]$, voici les résultats obtenus pour la reconstruction de l'image en utilisant la décomposition en une série de polynômes de Legendre :



Il semblerait donc qu'il y ait un problème ici.

16.3 Test sur le FNO

16.3.1 Explication

On considère Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x,y) = 1/8 - (x-1/2)^2 - (y-1/2)^2$ et le domain fictif $O = (0,1)^2$.
On souhaite résoudre

$$\begin{cases} -\Delta u & = f\,, \quad \text{dans } \Omega\,, \\ u & = 0\,, \quad \text{sur } \Gamma\,, \end{cases}$$

où

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2 + (y-\mu_1)^2}{2\sigma^2}\right),$$

avec $\sigma \sim \mathcal{U}([0.1, 0.6])$ et $\mu_0, \mu_1 \sim \mathcal{U}([0.5 - \sqrt{2}/4, 0.5 + \sqrt{2}/4])$.
On cherche à tester et comparer 2 méthodes de correction sur le FNO :

- **Méthode 1 :**
  Sans rehaussement :
  On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\tilde{\phi}C) = f & \Omega \\ \tilde{\phi}C = 0 & \Gamma \end{cases}$$

  On pose $\tilde{u} = \tilde{\phi}C$. On utilise alors la formulation Φ-FEM classique (en homogène).
  Avec rehaussement :

On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\hat{\phi}C) = f & \Omega \\ \hat{\phi}C = m & \Gamma \end{cases}$$

On pose $\hat{u} = \hat{\phi}C$. On utilise alors la formulation Φ-FEM en imposant la condition au bord par la méthode duale.

- **Méthode 2 :**
  On souhaite résoudre le problème suivant

$$\begin{cases} -\Delta(\phi C) = \tilde{f} & \Omega \\ \tilde{C} = 0 & \Gamma \end{cases}$$

avec $\tilde{u} = \tilde{\phi} + \tilde{C}$ où $\tilde{C} = \phi C$ et $\tilde{f} = f + \Delta\tilde{\phi}$. On utilise alors la formulation Φ-FEM classique (en homogène).

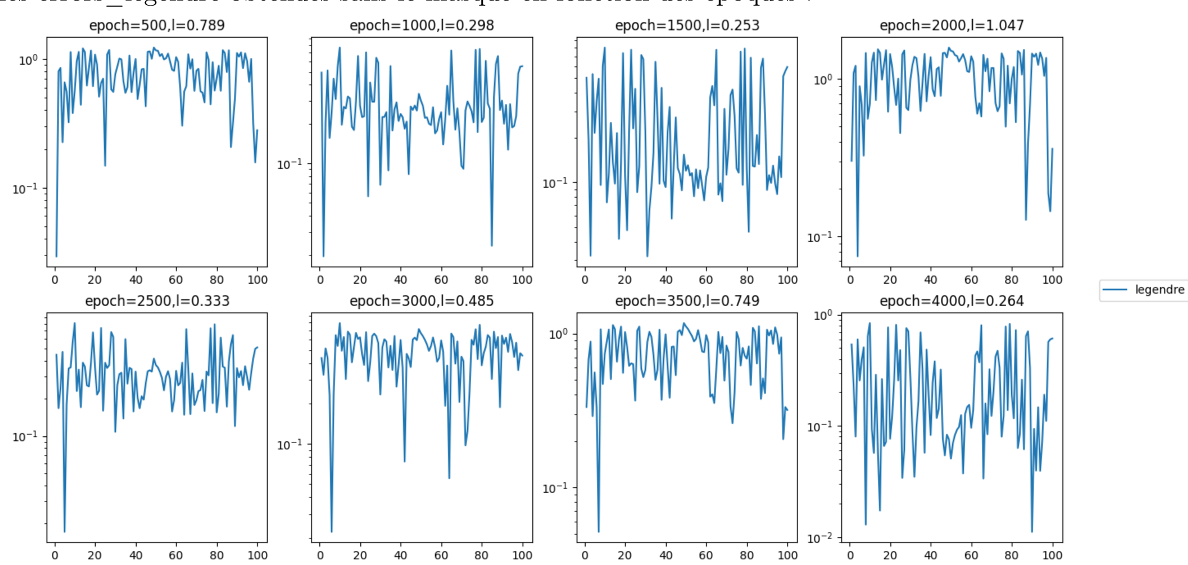
16.3.2 Applications

Voici les étapes effectuées :

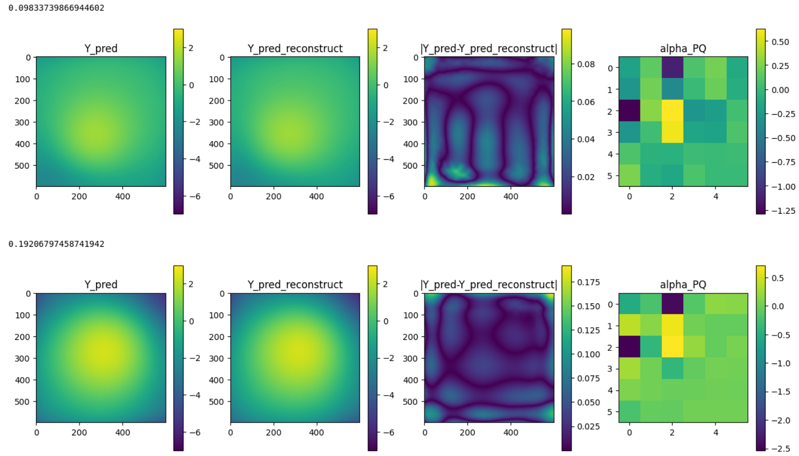
- On va créer une classe prenant en paramètre nb_vert. On récupère alors nb_dofs=2nb_vert-1.
- On récupère les paramètres *params* de l'ancien échantillon test ainsi que le nombre de données nb_data.
- On crée nos nouveaux inputs X_test pour l'échantillon test de taille (nb_data,nb_dofs,nb_dofs,$nb\_channels$).
- On génère nos nb_data solutions sur-raffinées Y_test avec FEniCS.
- On sépare notre échantillon test en batch pour éviter les Out Of Memory. Ainsi, pour chaque batch, on récupère la prédiction du FNO Y_pred et les erreurs associées $errors\_FNO$ (par rapport aux solutions sur-raffinées).
- On calcule $\widetilde{P_N}$ et $\widetilde{P_M}$ puis $\widetilde{\alpha_{P,Q}}$.
- On reconstruit alors les solutions en utilisant les polynômes de Legendre.
- On calcule ensuite les deux erreurs suivantes :
  — max(|Y_pred-Y_pred_reconstruct|) -> errors_reconstruct
  — $||Y\_test - \phi Y\_pred\_reconstruct||_{L^2}$ -> errors_legendre

Remarque. *On notera que les* Y_pred *sont* w *(et non* u*), il manque la multiplication par* ϕ*.*

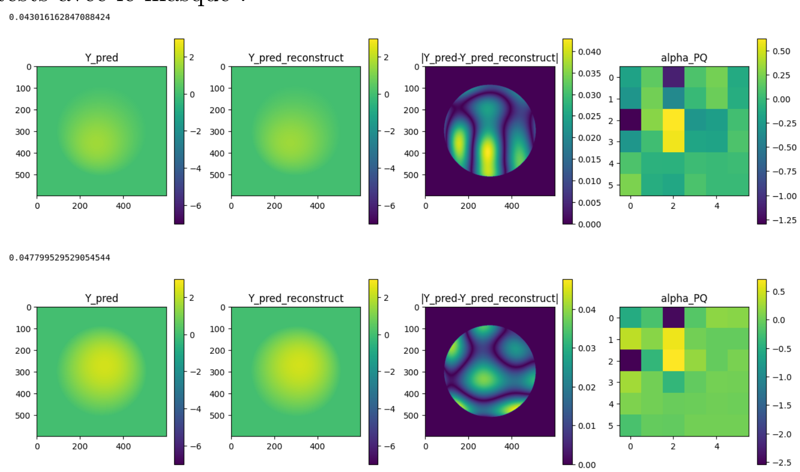
Voici les errors_legendre obtenues sans le masque en fonction des époques :



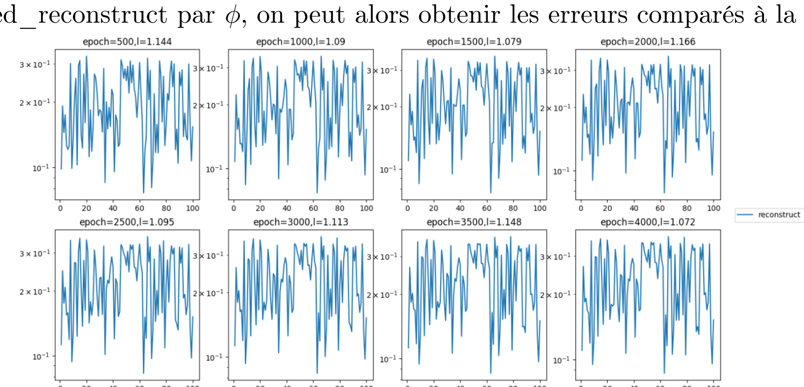
Et voici deux cas tests sans le masque (epoch=500, data=0,1)) :



Après avoir visualisé les cas test, il semblerait que les grosses erreurs se fassent aux bords, ce qui ne devrait pas posé problème étant donné que l'on va appliquer un masque.
Voici les 2 mêmes cas tests avec le masque :



En multipliant Y_pred_reconstruct par ϕ, on peut alors obtenir les erreurs comparés à la solution sur-raffinée :



Conclusion

Il semblerait que les polynômes de Legendre permettent bien d'obtenir une solution analytique à partir de la solution produite par le FNO. La semaine prochaine, il faudra tester la correction en $\mathbb{P}^1 0$. On pourra également tester de décomposer directement $\phi\times$Y_pred en série de polynômes de Legendre. Attention : Il y avait une petite erreur dans la méthode des rectangles. Cette erreur a été remarqué la semaine d'après, il faut changer N en $N-1$. Tous les résultats ont donc du être modifiés !

17 Semaine 17 : 29/05/2023 - 02/06/2023

Résumé

Cette semaine était assez courte. En effet, lundi était férié et le jeudi et le vendredi, il y a eut le déménagement. Globalement, cette semaine j'ai relancé mes résultats pour Legendre (avec les coefficients calculés sous formes matriciels). J'ai testé pour différents P et ait considéré 3 types d'erreurs.

17.1 Résultats FNO

Cette semaine, on va essayer d'approcher par série de polynômes de Legendre directement la solution prédite (c'est-à-dire la sortie du FNO multipliée par la levelset ϕ). On considérera

$$\widetilde{\alpha_{P,Q}} = \frac{4}{(N-1)(M-1)} \widetilde{D_P}^{-1} \widetilde{P_N}^T \widetilde{F_{N,M}} \widetilde{P_M} \widetilde{D_Q}^{-1}$$

En fait on ne multiplie pas réellement par P_N^T à gauche mais par une matrice $P_N^{T,-}$ qui est égale à P_N^T avec sa dernière colonne nulle. De la même manière, on ne multiplie pas réellement par P_M à droite mais par une matrice P_M^- qui est égale à P_M avec sa dernière ligne nulle. On considérera les trois erreurs suivantes :

— max(|Y_pred-Y_pred_reconstruct|) -> errors_reconstruct

— max(|(Y_pred-Y_pred_reconstruct)*mask|) -> errors_reconstruct_mask

— $||Y\_test - Y\_pred\_reconstruct||_{L^2}$ -> errors_legendre

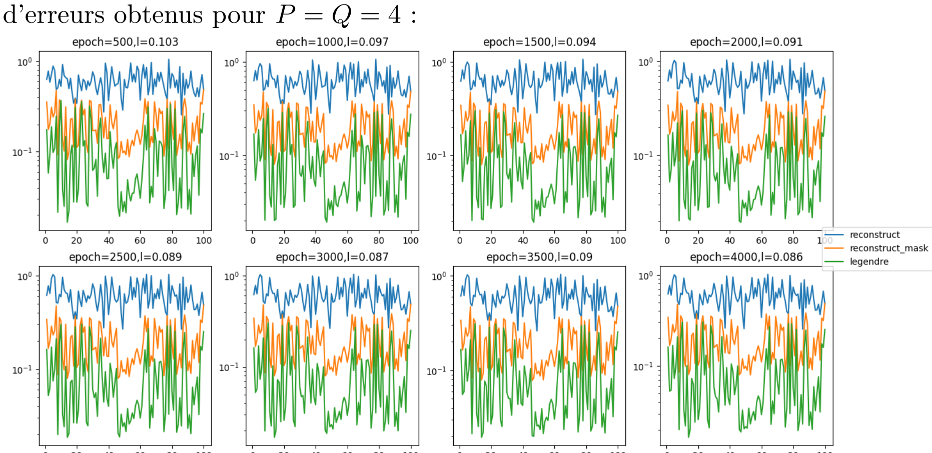
Résultats :

On considère Ω le cercle de rayon $\sqrt{2}/4$ et de centre $(0.5, 0.5)$ avec $\Phi(x,y) = 1/8 - (x-1/2)^2 - (y-1/2)^2$ et le domaine fictif $O = (0,1)^2$. On prend

$$f(x,y) = \exp\left(-\frac{(x-\mu_0)^2 + (y-\mu_1)^2}{2\sigma^2}\right),$$

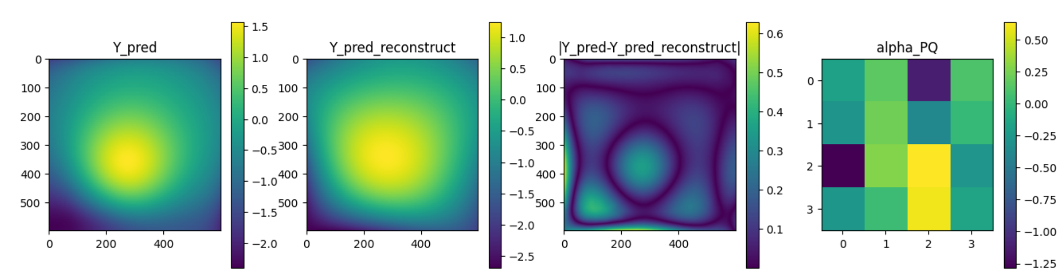
On considère $N = M = 599$ (=2*nb_vert-1 avec nb_vert=300).

Voici les 3 types d'erreurs obtenus pour $P = Q = 4$:

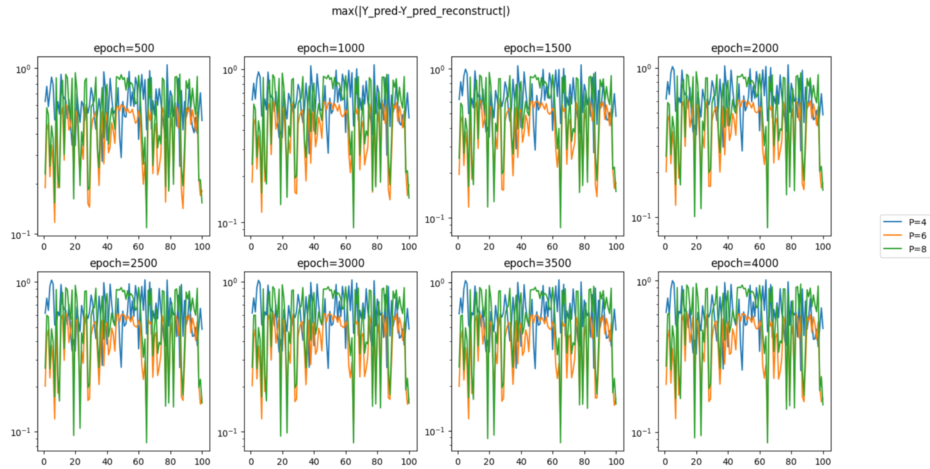


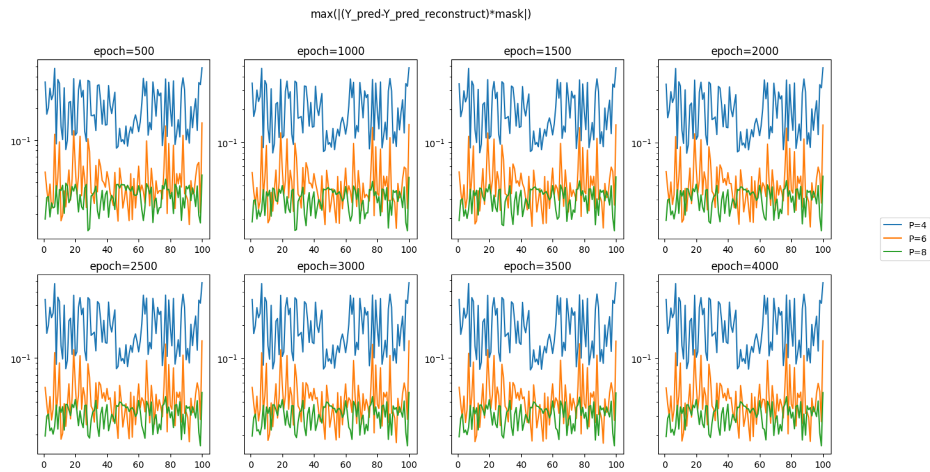
Voici également les résultats pour epoch=0 et data=0 :

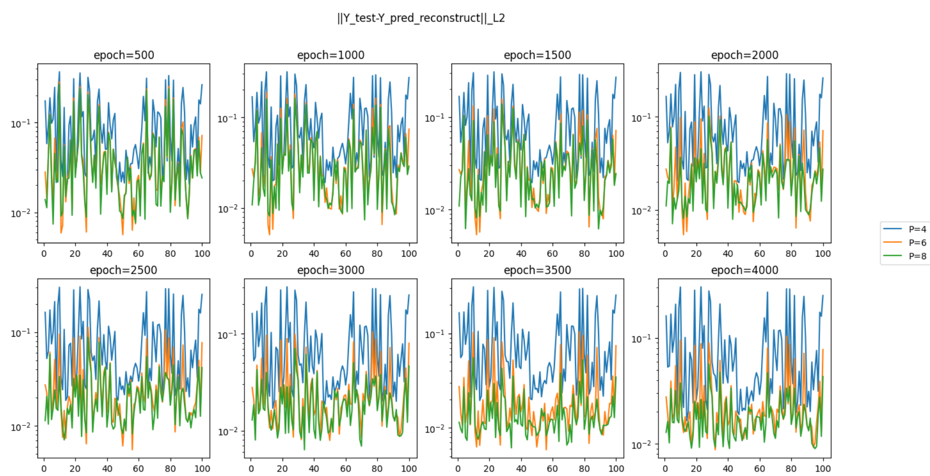
```
errors_reconstruct :  0.6295466908835694
errors_reconstruct_mask :  0.35415143914596126
```



En testant pour différents P ($P \in \{4, 6, 8\}$), on obtient les résultats suivants :







Conclusion

Les résultats n'étant pas très bon, je pense qu'il serait mieux d'approcher w par série de polynôme de Legendre plutôt que u afin d'éviter les erreurs au bord du domaine. Je pense également qu'un problème est le changement de variable (sur la solution analytique ça ne fonctionne pas non plus en considérant les problèmes matriciels.)

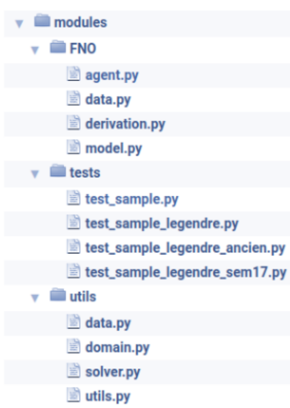
18 Semaine 18 : 05/06/2023 - 09/06/2023

Résumé

Lundi j'ai commencé à regarder comment appliquer la correction en $\mathbb{P}^{10}$ à partir de l'expression analytique obtenu par série de polynômes de Legendre. En fin de joournée, Emmanuel est passé et m'a expliqué une meilleure méthode pour calculer les coefficients des séries. Dans la journée de mardi, j'ai pris le temps de trier les fichiers et mettre au propre tout le code. J'ai séparé les fonctions qui été toutes dans un même jupyter en différents fichiers pythons. J'ai également créé un fichyier de configuration "config.json" qui regroupe une partie des paramètres. Pour le reste de la semaine, j'ai implémenté et tester la nouvelle méthode pour calculer les coefficients de Legendre. J'ai commencé sur solution analytique 2D avec et sans changement de variables puis directement sur le FNO (sur w). Pour le FNO, j'ai considéré les mêmes erreurs que la semaine précédente en comparant w et en comparant $u = \phi w$. On compare également errors_legendre et errors_FNO. En toute fin de semaine, j'ai également continué les tests sur la correction.

18.1 Séparation des fichiers

J'ai séparé le code en plusieurs modules, comme on peut le voir sur l'image de gauche. A droite, on peut voir les attributs et les fonctions de la classe **test_sample_legendre** :



Remarque. *J'ai rapidement généré une documentation doxygen.*

18.2 Nouvelle méthode de calcul des coefficients de Legendre

- En 1D, on a

$$f(x) = \sum_{p=0}^{P-1} \alpha_p P_p(x)$$

Ainsi pour tout $q = 0, \dots, P-1$

$$\begin{aligned}\int_{-1}^{1} f(x) P_q(x) dx &= \langle f, P_q \rangle_{L^2([-1,1])} \\ &= \sum_{p=0}^{P-1} \alpha_p \langle P_p, P_q \rangle_{L^2([-1,1])} \\ &= \alpha_q \langle P_q, P_q \rangle_{L^2([-1,1])}\end{aligned}$$

par orthogonalité des polynômes de Legendre dans $L^2([-1,1])$.
On en déduit

$$\alpha_q = \frac{\langle f, P_q\rangle_{L^2([-1,1])}}{\langle P_q, P_q\rangle_{L^2([-1,1])}} = \frac{2q+1}{2}\langle f, P_q\rangle_{L^2([-1,1])}$$

- En 2D, on a

$$f(x,y) = \sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} \alpha_{p,q}P_p(x)P_q(y)$$

Montrons dans un premier temps que les polynômes

$$Q_{p,q}(x,y) = P_p(x)P_q(y)$$

sont orthogonaux dans l'espace $L^2([-1,1]^2)$:

$$\begin{aligned}
\int_{-1}^{1}\int_{-1}^{1} Q_{p,q}(x,y)Q_{p',q'}(x,y)dxdy &= \int_{-1}^{1}\int_{-1}^{1} P_p(x)P_q(y)P_{p'}(x)P_{q'}(y)dxdy \\
&= \int_{-1}^{1} P_p(x)P_{p'}(x)dx \times \int_{-1}^{1} P_q(y)P_{q'}(y)dy \\
&= \frac{2}{2p+1}\delta_{pp'}\frac{2}{2q+1}\delta_{qq'} \\
&= \frac{4}{(2p+1)(2q+1)}\delta_{(p,q)(p',q')}
\end{aligned}$$

Ainsi pour tout $p' = 0,\ldots,P-1$,$q' = 0,\ldots,Q-1$

$$\begin{aligned}
\int_{-1}^{1}\int_{-1}^{1} f(x,y)Q_{p',q'}(x,y)dxdy &= \langle f, Q_{p,q}\rangle_{L^2([-1,1]^2)} \\
&= \sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} \alpha_{p,q}\langle Q_{p,q}, Q_{p',q'}\rangle_{L^2([-1,1]^2)} \\
&= \alpha_{p',q'}\langle Q_{p',q'}, Q_{p',q'}\rangle_{L^2([-1,1]^2)}
\end{aligned}$$

par orthogonalité des polynômes $Q_{p,q}$ dans $L^2([-1,1]^2)$.
On en déduit

$$\alpha_{p',q'} = \frac{\langle f, Q_{p',q'}\rangle_{L^2([-1,1]^2)}}{\langle Q_{p',q'}, Q_{p',q'}\rangle_{L^2([-1,1]^2)}} = \frac{(2p'+1)(2q'+1)}{4}\langle f, Q_{p',q'}\rangle_{L^2([-1,1])}$$

Remarque. *Pour calculer le produit scalaire sur $L^2([-1,1]^2)$, on va utiliser la méthode des trapèzes :*

$$\begin{aligned}
\int_{-1}^{1}\int_{-1}^{1} g(x,y)dydx &= \int_{-1}^{1} \frac{2}{N-1}\sum_{i=0}^{N-1}\frac{g(x_i,y)+g(x_{i+1},y)}{2} \\
&= \frac{1}{(N-1)^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left[g(x_i,y_i)+g(x_{i+1},y_i)+g(x_i,y_{i+1})+g(x_{i+1},y_{i+1})\right]
\end{aligned}$$

ATTENTION : Il faut implémenter ce calcule de manière vectorielle sinon c'est beaucoup trop long (car double boucle sur N) :

```
def trapez_method(self,g):
    N=self.nb_dofs
    sum_ij = g[:,:,:N-1,:N-1]+g[:,:,1:,:N-1]+g[:,:,:N-1,1:]+g[:,:,1:,1:]
    return 1/(N-1)**2 * np.sum(sum_ij,axis=(2,3))
```

On utilisera également la fonction legval de Numpy pour l'évaluation des polynômes de Legendre en nos degré de liberté.

18.2.1 Résultats sur la solution analytique

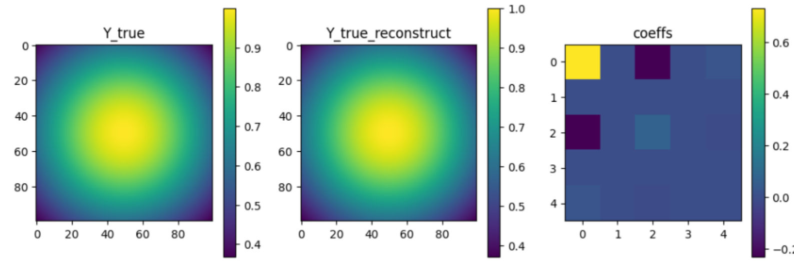
On prendra

$$u(x) = exp\left(-\frac{(x-\mu)^2 + (y-\mu)^2}{2\sigma^2}\right)$$

avec $\mu = 0$ et $\sigma = 1$. On prendra $P, Q = 5$ et $N = 100$.
On commence par vérifier l'orthogonalité des polynômes de Legendre :

```
diff_diag :  0.0027181189730220434
max hors diag 0.0017679718526824823
```

On testera également de faire varier P et N.
Sans changement de variable :
En prenant $x \in [-1, 1]$, voici les résultats obtenus pour epoch=0 et data=0 :

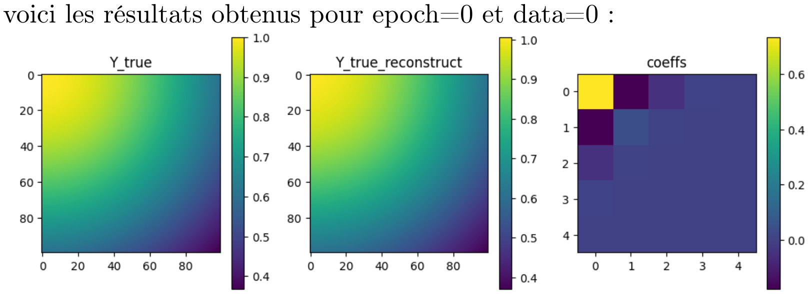


max|Y_true-Y_true_reconstruct|=0.0036

Et en faisant varier N et P :

	N=100	N=200	N=400	N=600	N=800	N=1000	N=1200	N=1400	N=1600	N=1800	N=2000	N=5000	N=10000
p=2	0.364178	0.364205	0.364211	0.364213	0.364213	0.364213	0.364213	0.364213	0.364214	0.364214	0.364214	0.364214	0.364214
p=3	0.026878	0.026967	0.026989	0.026993	0.026994	0.026995	0.026995	0.026995	0.026995	0.026996	0.026996	0.026996	0.026996
p=4	0.026878	0.026967	0.026989	0.026993	0.026994	0.026995	0.026995	0.026995	0.026995	0.026996	0.026996	0.026996	0.026996
p=5	0.003604	0.001903	0.001483	0.001406	0.001379	0.001366	0.001359	0.001355	0.001353	0.001351	0.001350	0.001345	0.001344
p=6	0.003604	0.001903	0.001483	0.001406	0.001379	0.001366	0.001359	0.001355	0.001353	0.001351	0.001350	0.001345	0.001344
p=7	0.008760	0.002122	0.000491	0.000191	0.000086	0.000046	0.000039	0.000035	0.000032	0.000030	0.000031	0.000045	0.000048
p=8	0.008760	0.002122	0.000491	0.000191	0.000086	0.000046	0.000039	0.000035	0.000032	0.000030	0.000031	0.000045	0.000048
p=9	0.024012	0.005876	0.001459	0.000648	0.000364	0.000234	0.000163	0.000120	0.000092	0.000073	0.000059	0.000011	0.000004
p=10	0.024012	0.005876	0.001459	0.000648	0.000364	0.000234	0.000163	0.000120	0.000092	0.000073	0.000059	0.000011	0.000004

Avec changement de variable :
En prenant $x \in [0, 1]$, voici les résultats obtenus pour epoch=0 et data=0 :



Et en faisant varier N et P :

	N=100	N=200	N=400	N=600	N=800	N=1000	N=1200	N=1400	N=1600	N=1800	N=2000	N=5000	N=10000
p=2	0.127232	0.127184	0.127172	0.127169	0.127169	0.127168	0.127168	0.127168	0.127168	0.127168	0.127168	0.127168	0.127168
p=3	0.017979	0.017350	0.017195	0.017166	0.017156	0.017152	0.017149	0.017148	0.017147	0.017146	0.017145	0.017144	0.017143
p=4	0.000969	0.001553	0.001825	0.001875	0.001892	0.001900	0.001905	0.001907	0.001909	0.001910	0.001911	0.001914	0.001914
p=5	0.005967	0.001227	0.000396	0.000295	0.000260	0.000267	0.000286	0.000298	0.000305	0.000310	0.000314	0.000327	0.000328
p=6	0.008643	0.002148	0.000546	0.000251	0.000148	0.000100	0.000075	0.000059	0.000049	0.000042	0.000037	0.000019	0.000017
p=7	0.023525	0.005801	0.001444	0.000643	0.000363	0.000234	0.000163	0.000121	0.000093	0.000075	0.000061	0.000013	0.000006
p=8	0.029367	0.007231	0.001796	0.000797	0.000448	0.000286	0.000199	0.000146	0.000112	0.000088	0.000071	0.000011	0.000003
p=9	0.063349	0.015507	0.003847	0.001706	0.000959	0.000613	0.000426	0.000313	0.000239	0.000189	0.000153	0.000024	0.000006
p=10	0.075355	0.018410	0.004565	0.002024	0.001137	0.000728	0.000505	0.000371	0.000284	0.000224	0.000182	0.000029	0.000007

18.2.2 Résultats sur le FNO

On va régénérer les données en prenant

$$\alpha_{p,q} = \frac{(2p+1)(2q+1)}{4} \langle f, Q_{p,q} \rangle_{L^2([-1,1])}$$

De plus, on cherchera à décomposer en série de polynômes de Legendre la sortie du FNO (w) plutôt que la solution (c'est-à-dire la prédiction du FNO multipliée par la levelset ϕ).
On considérera alors

$$w(x,y) = \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} \alpha_{p,q} P_p(x) P_q(y)$$

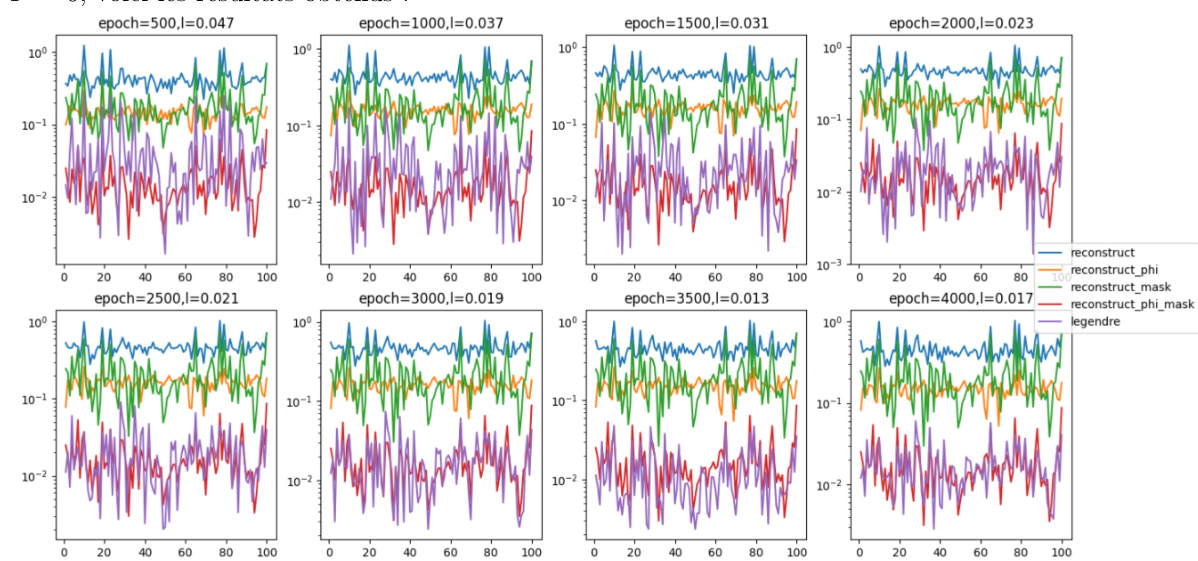
Ainsi

$$u(x,y) = w(x,y) \times \phi(x,y)$$

On considérera les cinq erreurs suivantes :

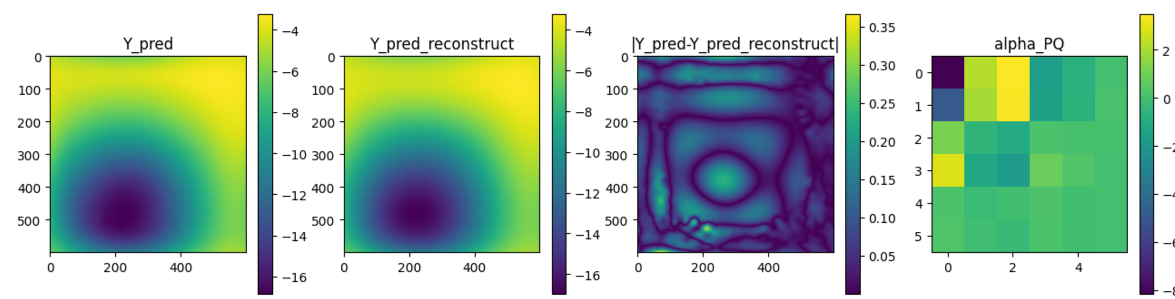
— max(|Y_pred-Y_pred_reconstruct|) -> errors_reconstruct
— max(|(Y_pred-Y_pred_reconstruct)*phi|) -> errors_reconstruct_phi
— max(|(Y_pred-Y_pred_reconstruct)*mask|) -> errors_reconstruct_mask
— max(|(Y_pred-Y_pred_reconstruct)*phi*mask|) -> errors_reconstruct_phi_mask
— $||Y\_test - Y\_pred\_reconstruct||_{L^2}$ -> errors_legendre

Pour $P = 6$, voici les résultats obtenus :

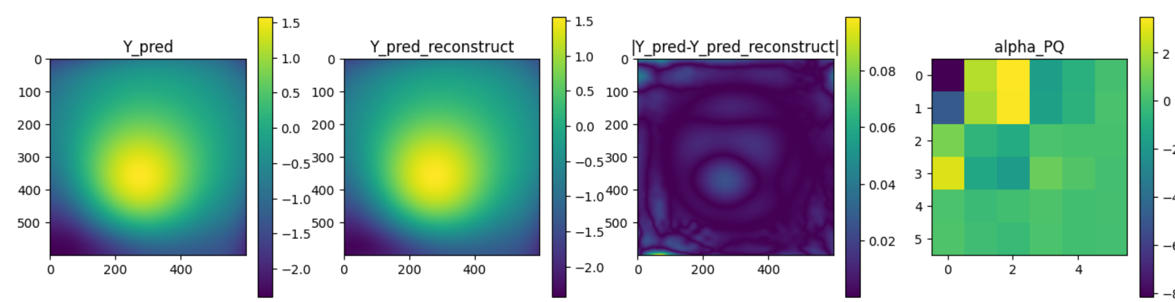


Pour epoch=0 et data=0, on a

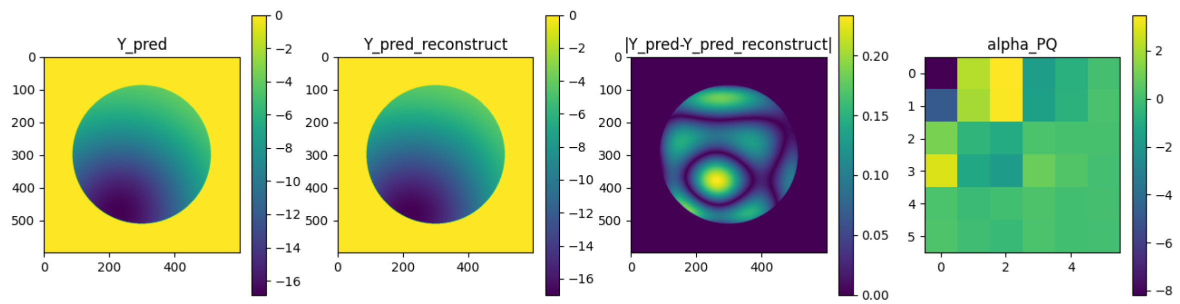
errors_reconstruct : 0.3661062124457999



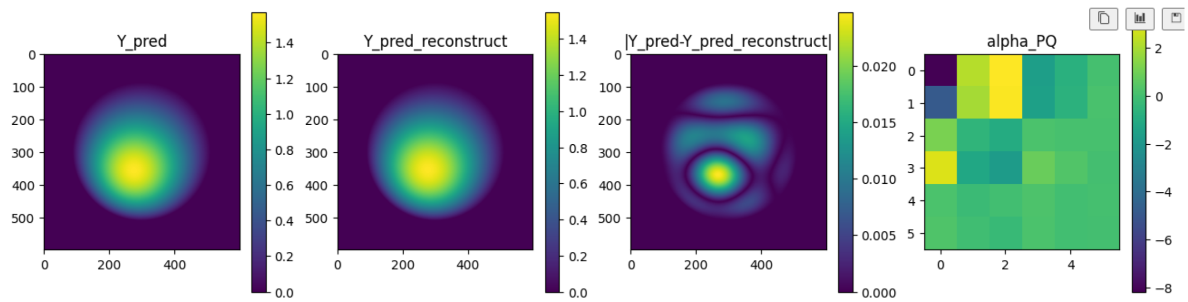
errors_reconstruct_phi : 0.09895795604737012



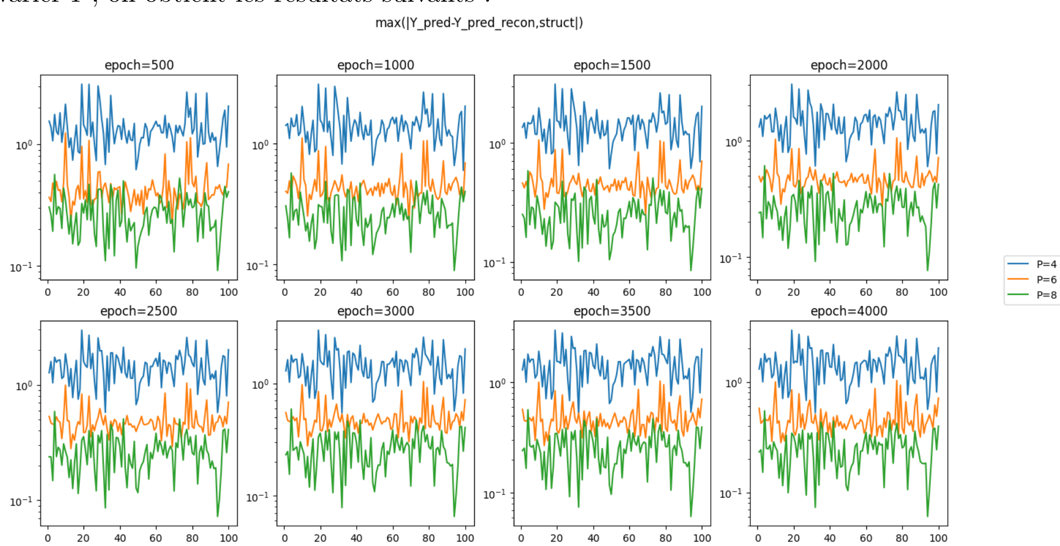
```
errors_reconstruct_mask :  0.23418376549237152
```

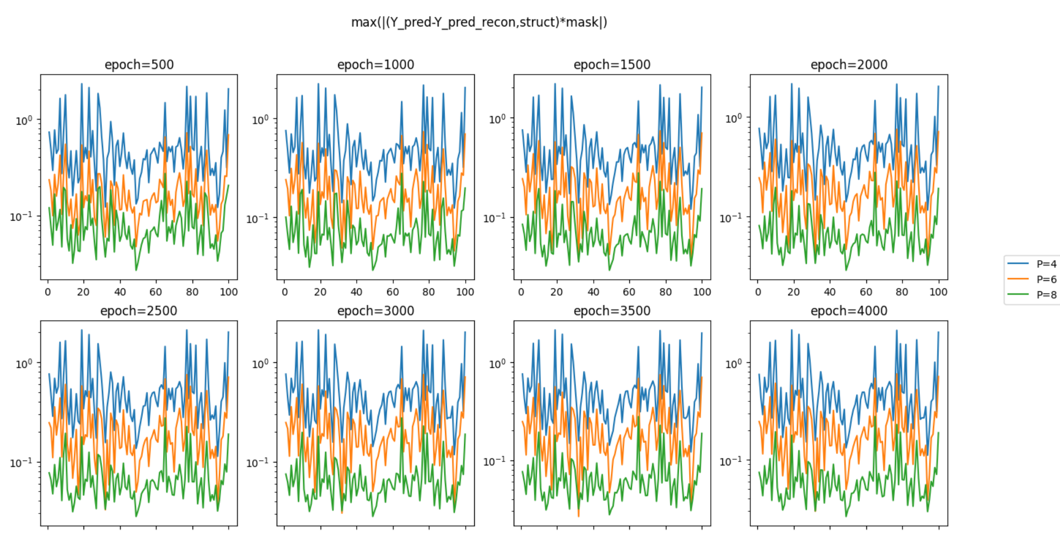


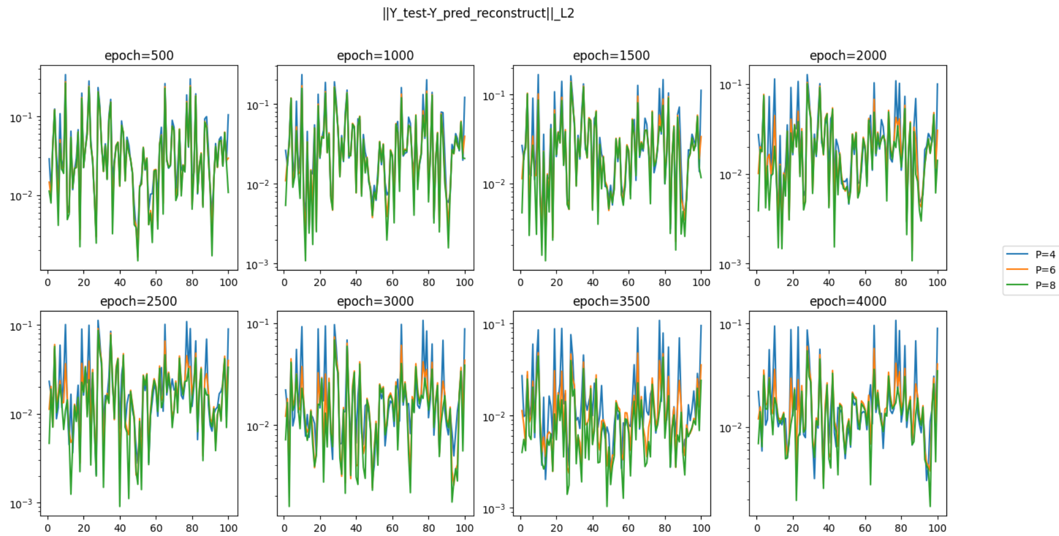

```
errors_reconstruct_phi_mask :  0.024772918720191273
```



En faisant varier P, on obtient les résultats suivants :







18.3 Correction de haut degré

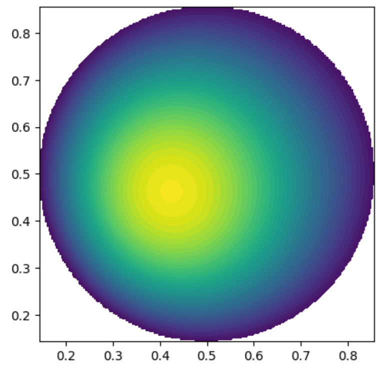
A ce stade, on a $\alpha_{p,q}$ pour chaque époque (sous-entendu celles enregistrées par le modèle) et pour chaque data. On souhaite appliquer les différents types de correction (par multiplication avec et sans rehaussement et par addition) en prenant

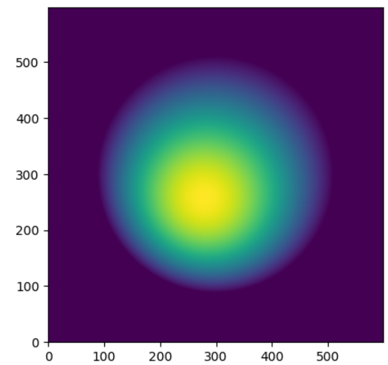
$$\tilde{\phi}(x,y) = \left(\sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} \alpha_{p,q} P_p(x) P_q(y) \right) \times \phi(x,y)$$

où x, y sont les degrés de liberté associés à $\mathbb{P}^k$ avec k assez grand (on aimerait prendre 10 par exemple).
On utilisera ensuite la fonction *construct_one_function(XX,YY,epoch,data)* de la classe *test_sample_legendre* qui pour une donnée évalue $\tilde{\phi}$ en les XX,YY.
Dans le cas de correction où on utilisait directement la prédiction du FNO (multipliée par ϕ), on récupérait la prédiction du FNO (dans $\mathbb{P}^2$) et à partir de celle-ci on générait une fonction FEniCS $\tilde{\phi}$ auquel on pouvait appliquer les différentes corrections.
Dans le cas présent, on possède une expression analytique de la solution et donc il nous suffit de récupérer les degrés de liberté associés à $\mathbb{P}^k$ et d'évaluer $\tilde{\phi}$ en ces points.
Voici un exemple d'implémentation (pour $k = 6$) :

```
nb_vert = 300
P = 4
test_legendre = test_sample_legendre(nb_vert,P)

solver = CorrSolver(nb_cell=test_legendre.nb_vert-1, params=test_legendre.params,
   Y_test=test_legendre.Y_test, V_ex=test_legendre.V_ex, dx_ex=test_legendre.dx_ex)

V_phi = FunctionSpace(solver.mesh,"CG",6)
XXYY = V_phi.tabulate_dof_coordinates().T
XX,YY = XXYY
XX = test_legendre.get_new_coord(XX)
YY = test_legendre.get_new_coord(YY)

epoch = 0
data = 5
f = test_legendre.construct_one_function(XX,YY,epoch,data)
f_FEniCS = Function(V_phi)
f_FEniCS.vector()[np.arange(0,f.shape[0],1)] = f
f_FEniCS = f_FEniCS * PhiExpr(degree=6,domain=solver.mesh)
```

Voici les résultats obtenus (à gauche la solution prédite par le FNO pour nb_vert=300 et à droite la solution FEniCS) :





Conclusion

Pour l'instant nous sommes bloqués ici car on n'a pas assez de RAM pour gérer d'aussi grosse résolution de problème variationnels. La semaine prochaine, il faudra effectués plusieurs tests dans le but de voir les limitations.

19 Semaine 19 : 12/06/2023 - 16/06/2023

Résumé

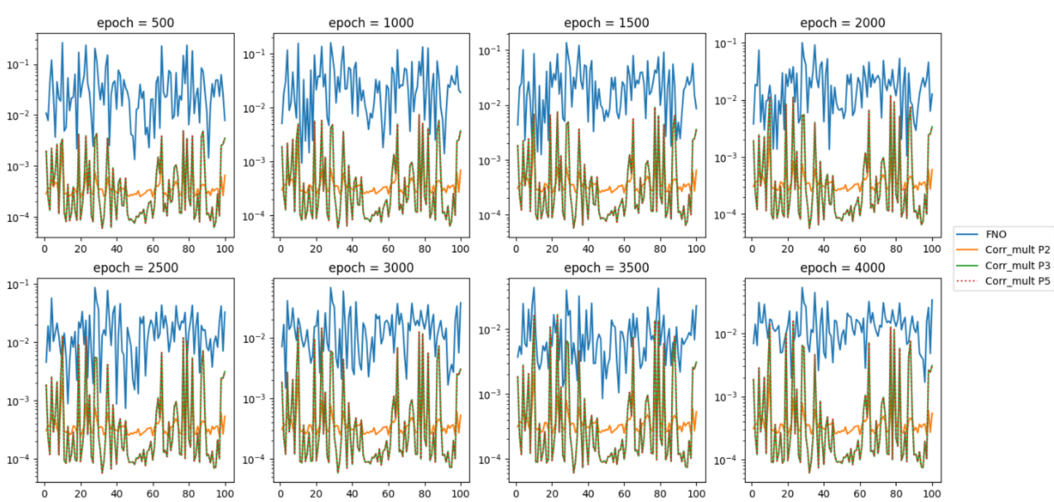
- Cette semaine, on a commencé par essayer de comprendre pourquoi le kernel se bloquait. On a tenté de résoudre un problème de correction sur une solution analytique avec nb_vert=300 et en degré 10 et il n'y a pas eut de soucis. C'est pourquoi, je ne comprends pas pourquoi on a ce problème !
- Comme on ne peut pas faire du haut degré avec Legendre, j'ai testé avec des plus petits degrés. J'ai remarqué que la projection de phi_tild sur notre espace de fonction V_ex pour calculer la norme L2 d'erreur prend beaucoup de temps. Avec un degré 5, on est sur environ 15s (que l'on doit multiplier par nb_data=100 et nb_epochs=8). Par la suite, j'ai eut un autre problème avec la correction par multiplication. En prenant P=4 polynômes de Legendre, j'obtiens exactement les mêmes erreurs pour $\tilde{\phi}$ de degré 3 ou 5. En prenant P=6 et un degré 5, mes erreurs sont moins bonnes qu'avec P=4.
- J'ai alors testé la convergence de la décomposition en série de Polynômes de Legendre, tout d'abord avec la méthode des trapèzes pour différents nombres de points N. Puis avec une méthode de quadrature de scipy (scipy.integrate.quad) qui prend une expression analytique de notre fonction en paramètre (et pas un ensemble de valeurs à des points connus). Il semblerait alors qu'avec la méthode de scipy, on est bien la convergence attendue.
- J'ai également corrigé la preuve sur l'inégalité pour le rehaussement avec FEM et ait commencé à générer une documentation sphinx du code.

19.1 Test pour le kernel

Test pour voir si le kernel se bloque sur la solution analytique en $\mathbb{P}^{10}$ avec nb_vert=300 :

```
params : [[0.5, 4, 0.0]]
params_perturbation : [[0.5, 2, 0.0]]
num of cell in the ghost penalty: 1446
#### eps =  0.01
### PhiFEM Corr
0/1:Solving linear variational problem.
Building point search tree to accelerate distance queries.
Computed bounding box tree with 141975 nodes for 70988 points.
||Y_true-Y_true_corr||_L2 [0.006122752719941293]
```

19.2 Résultat Correction $\mathbb{P}^3$ et $\mathbb{P}^5$ (Legendre)



19.3 Test de convergence 1D (Legendre)

Test de convergence de la décomposition en une série de polynôme de Legendre sur la solution analytique suivante

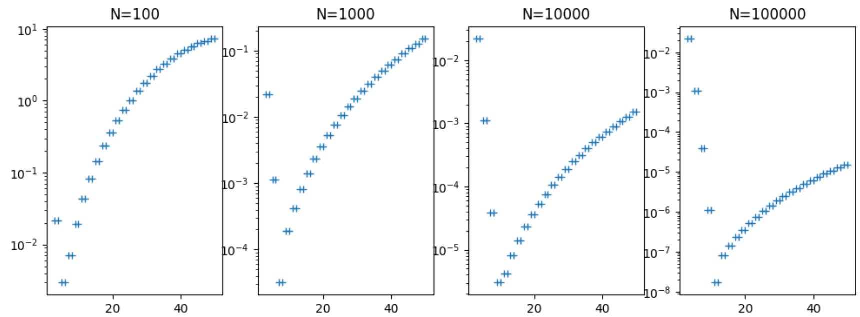
$$u_{ex}(x) = \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

avec $\mu = 0$ et $\sigma = 1$.

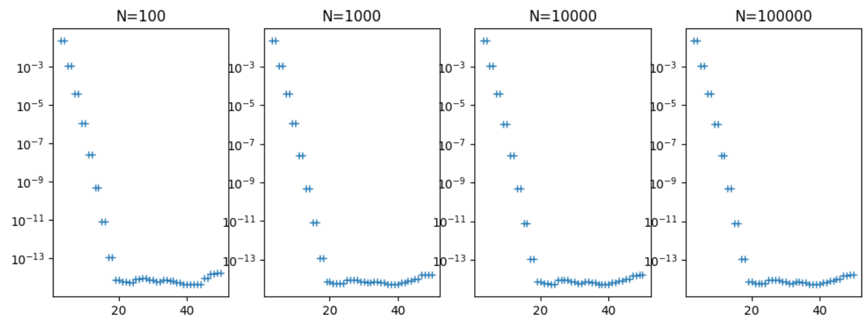
On comparera pour différents nombres de points N, l'erreur max(|Y_true-Y_true_reconsrtruct|) en faisant varier le nombre de polynômes de Legendre P. On a commencé par test en considérant une solution discrétisée en nos N^2 nœuds et en utilisant la méthode des trapèzes pour calculer les coefficients α_p. Etant donné que les résultats n'étaient pas bons, on a testé avec une une méthode de quadrature de scipy (scipy.integrate.quad) qui prend une expression analytique de notre fonction en paramètre.

Remarque. *On notera que contrairement à la méthode des trapèzes, avec scipy le paramètre N n'influence pas le calcul des coefficients. On calcule juste l'erreur en norme infini sur des solutions avec différents nombres de points.*

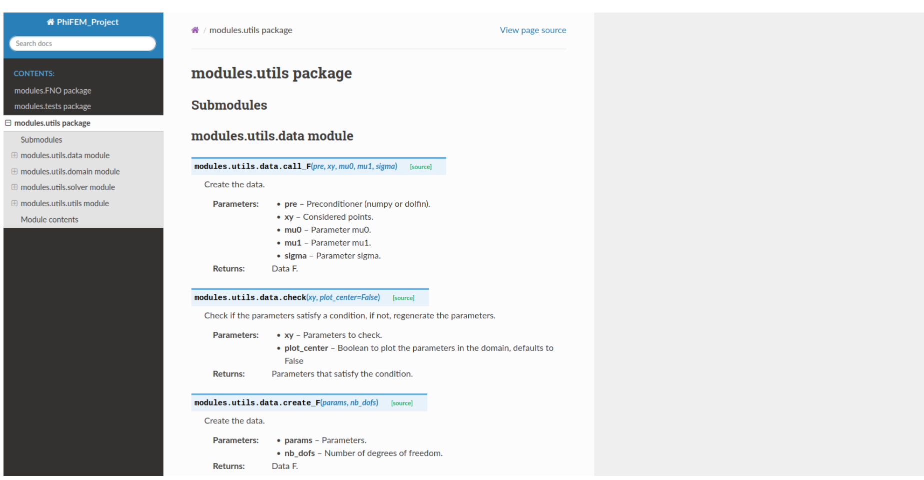
Résultats obtenus avec la méthode des trapèzes :



Résultats obtenus avec la méthode de scipy :



19.4 Documentation sphinx



Conclusion

Après une réunion mercredi avec Emmanuel et Michel, Emmanuel a proposé de tester les 2 idées suivantes. La première, tenter de corriger en $\mathbb{P}^{10}$ une solution analytique décomposée en série de polynômes de Legendre. La seconde, implémenter un réseau de neurones multi-perceptron avec $||w_\theta(x)\phi(x) - u(x)||_{H^1}$. Ces deux idées seront abordés dans le suivi de la semaine prochaine.

20 Semaine 20 : 19/06/2023 - 23/06/2023

Résumé

Cette semaine on s'est concentré sur l'implémentation d'un réseau multi-perceptron. On a commencé par implémenter un modèle capable d'approcher directement la solution u à partir d'une loss $mae(y\_true - y\_pred)$ (où mae = mean absolute error). Puis on a tenté d'approcher la solution w à partir de la loss $mae(y\_true - \phi y\_pred)$. Le second modèle fut plus compliqué car on a due créer une boucle d'entraînement personnalisée. L'idée étant que ces 2 modèles soient capables de prédire une solution seulement à partir d'un couple de point (x, y). On considère alors y_true comme étant une solution analytique associée à une force f. On testera de faire varier différents paramètres afin de tester plusieurs configurations des modèles. On souhaitera ensuite comparer quels paramétrisation est la meilleure. Pour cela on comparer les loss calculées pendant l'entraînement en fonction des époques ainsi que la norme L^2 calculée sur un échantillon test.

On souhaitera dans un second temps appliqué une correction à la sortie du réseau en $\mathbb{P}^{10}$.

Après avoir discuté avec Emmanuel vendredi matin, il semblerait que le problème au niveau de la correction puisse venir des dérivées premières et secondes de la prédiction du modèles. C'est pourquoi il a conseillé de les afficher pour voir si le problème vient de là.

On a aussi commencé à préparer le rapport de stage : page de garde + début de plan + lecture sur le FNO. Il faudra par la suite commencer la rédaction. A noter que j'ai demandé à Ouiza et je n'aurais pas la possibilité de venir en août, je n'ai donc pas réellement d'autres choix que de commencer dès maintenant.

20.1 Référence - Réseau Multi-perceptron

- Cours - Emmanuel Franck

  > On appelle un réseau totalement connecté ou Multi-Perceptron un réseau ou les matrices sont pleines. On parle d'un Perceptron simple si il y a une seule couche qui va de l'espace d'entrée à celle de sortie. Les couches qui ne concerne pas l'espace de sortie sont appelées couche cachée. Un des ingrédients essentiels des réseaux de neurones est la fonction non linéaire qui intervient entre chaque partie linéaire on parle de fonction d'activation.

- Wikipedia - Perceptron Multicouche

20.2 Configuration des 2 modèles

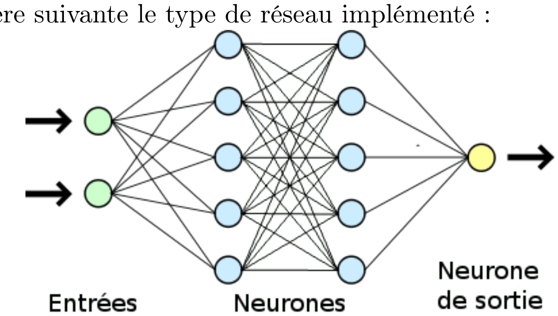
Pour chaque modèle, on va faire varier plusieurs paramètres afin de voir lequel est le plus performant, on aura au total 12 modèles où l'on va faire varier :

- le nombre de couches cachées du réseau (nb_hidden_layers $\in \{3, 4, 5\}$)
- le nombre de neurones dans chaque couche qui sera identique dans chacune (units $\in \{20, 60\}$)
- le taux d'apprentissage (lr $\in \{0.01, 0.001\}$)

On entraînera le réseau sur 4000 époques avec comme batch_size=16 pour chacune des configurations possibles. Ainsi voici les différentes configurations du modèle :

	1	2	3	4	5	6	7	8	9	10	11	12
nb_hidden_layers	3.00	3.000	3.00	3.000	4.00	4.000	4.00	4.000	5.00	5.000	5.00	5.000
units	20.00	20.000	60.00	60.000	20.00	20.000	60.00	60.000	20.00	20.000	60.00	60.000
lr	0.01	0.001	0.01	0.001	0.01	0.001	0.01	0.001	0.01	0.001	0.01	0.001

On peut représenter de la manière suivante le type de réseau implémenté :



20.3 Modèle 1 : $u = \phi w$

Dans un premier temps, j'ai implémenté un réseau capable d'apprendre directement la solution u.

20.3.1 Implémentation du modèle

Voici les étapes principales effectuées :

- Input X : (nb_pts,2), Output Y : (nb_pts,1)
- On crée un modèle keras séquentiel composé de *nb_hidden_layers* couche dense de *units* neurones. A noter qu'on rajoute une couche Dense à la fin de 1 neurones.
- On compile le modèle en lui donnant un optimiseur (on prend Adam auquel on fournit le taux d'appresntissage *lr*) ainsi que la loss (dans notre cas "mae" fournit par tensorflow).
- On fit le modèle à partir de X_train, Y_train, une *batch_size* et un nombre d'époques totales.

Remarque. *On prendra comme X_train les coordonnées (x,y) en* $\mathbb{P}^2$ *avec nb_vert=32 (pour se ramener au même cas que le FNO) et Y_train l'évalution de la solution analytique en chacun de ces degrés de liberté.*
On sauvegardera également le modèle à différents nombres d'époques sous la forme de checkpoints et à la fin de l'apprentissage, on sauvegardera l'historique (les loss).

20.3.2 Résultats

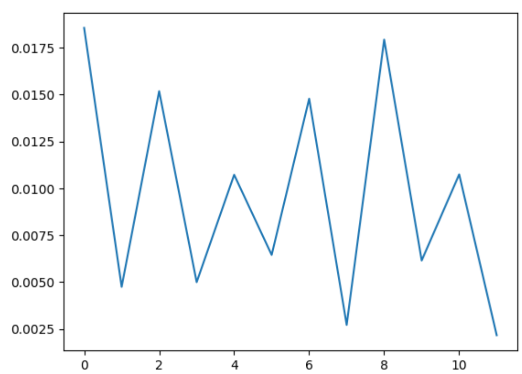
On considère ici la solution analytique suivante

$$u_{ex}(x,y) = S \times \sin(2\pi f x + p) \times \sin(2\pi f y + p)$$

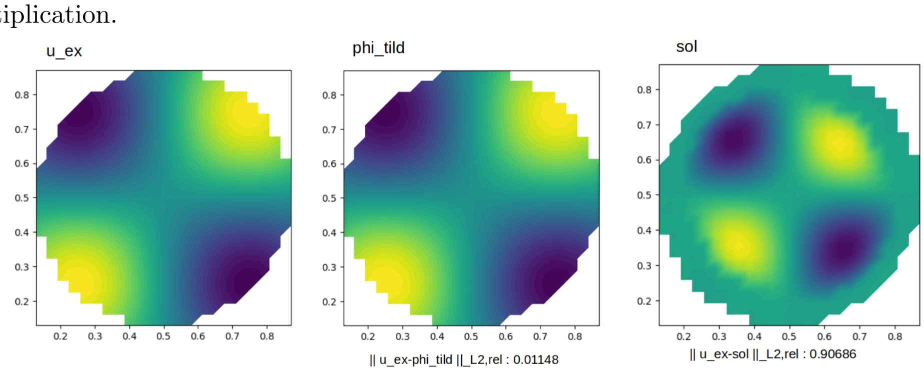
avec $S = 0.5$, $f = 1$ et $p = 0$.

Remarque. *A noter qu'on se place ici dans le cas d'une* ***solution non homogène****.*

On a testé différents modèles. Voici les loss finales obtenus pour chacun des 12 configurations possibles :



Il semblerait que les 12 modèles soient plus ou moins capable d'apprendre correctement la solution. On considérera dans la suite le dernier modèle (car c'est celui qui a la meilleure loss finale). On cherche maintenant à corriger la prédiction du modèle 12 en l'injectant dans un solveur PhiFEM. On considère alors la solution en $\mathbb{P}^{10}$ que l'on va corriger par multiplication.



Il semblerait alors que le solveur ne nous donne pas du tout la bonne solution. Une première hypothèse est que les conditions au bord ne soit pas correctement vérifiées sur phi_tild et donc C n'est pas imposé faiblement à 1.

20.4 Modèle 2 : w

Pour être sûre de bien imposer les conditions aux bord, on peut entraîner le modèle à apprendre w plutôt que u et ainsi e multiplier par ϕ ensuite (qui est nulle au bord).

20.4.1 Implémentation du modèle

Voici les étapes principales effectuées :

- Input X : (nb_pts,2), Output Y : (nb_pts,1)
- On crée un modèle keras séquentiel composé de *nb_hidden_layers* couche dense de *units* neurones. A noter qu'on rajoute une couche Dense à la fin de 1 neurones.
- On définit une fonction de loss :

  ```
  def loss_mae(y_true,y_pred,xy):
      return tf.reduce_mean(tf.abs(y_true-y_pred*call_phi(None,xy)))
  ```

  ainsi qu'un optimiseur (on prend Adam auquel on fournit le taux d'appresntissage *lr*).
- On fit le modèle en utilisant une boucle d'entraînement personnalisée :
  — On commence par une première boucle sur les époques.
  — Pour chaque époque, on shuffle notre X_train et le Y_train associé.
  — On effectue alors une seconde boucle sur les batch dans laquelle on va réaliser une descente de gradient à partir de la loss précédente.

Remarque. *Comme précédemment, on prendra comme X_train les coordonnées (x,y) en $\mathbb{P}^2$ avec nb_vert=32 (pour se ramener au même cas que le FNO) et Y_train l'évaluation de la solution analytique en chacun de ces degrés de liberté.*
On sauvegardera également le modèle à différents nombres d'époques sous la forme de checkpoints et à la fin de l'apprentissage, on sauvegardera l'historique (les loss).

20.4.2 Résultats

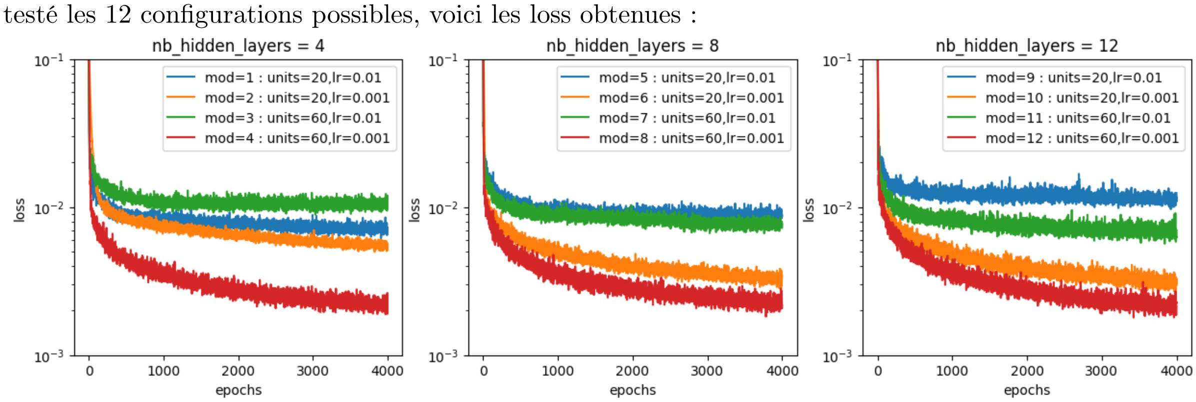
On considère ici la solution analytique suivante

$$u_{ex}(x,y) = S \times \sin(2\pi f x + p) \times \sin(2\pi f y + p) \times \cos(4\pi((x-0.5)^2 + (y-0.5)^2))$$

avec $S = 0.5$, $f = 1$ et $p = 0$.

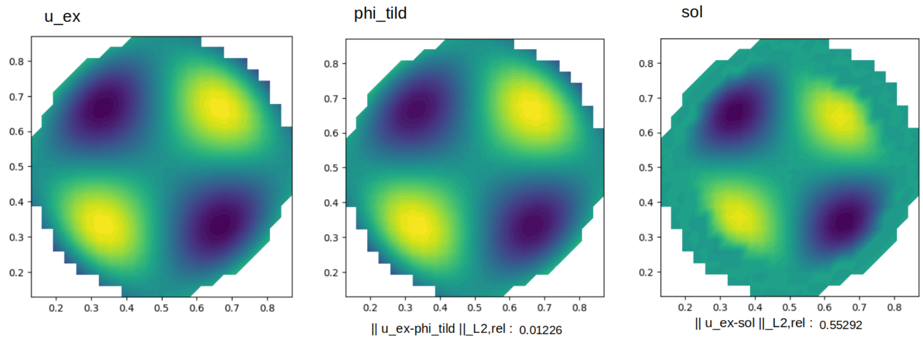
Remarque. *A noter qu'on se place ici dans le cas d'une* ***solution homogène****.*

On a testé les 12 configurations possibles, voici les loss obtenues :

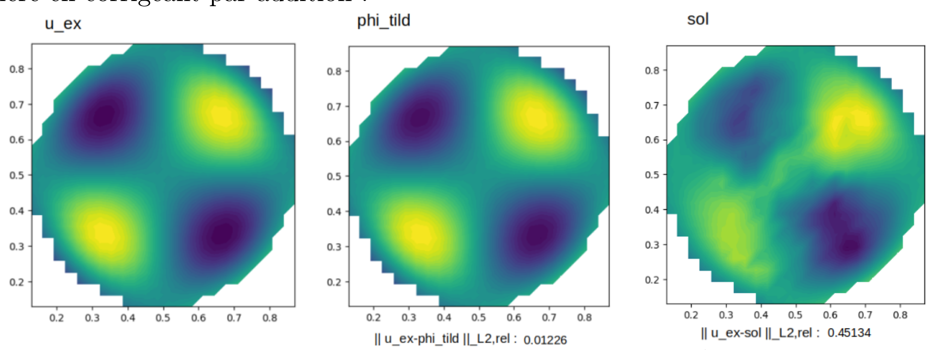


Remarque. *Il faudrait rajouter une comparaison des modèles sur un échantillon test !!*

Il semblerait que les 12 modèles soient plus ou moins capable d'apprendre correctement la solution. On considérera dans la suite le modèle 8 (dernier modèle lancé au moment des tests sur la correction). On cherche maintenant à corriger la prédiction du modèle 8 en l'injectant dans un solveur PhiFEM. On considère alors la solution en $\mathbb{P}^{10}$ que l'on va corriger par multiplication.



De la même manière en corrigeant par addition :

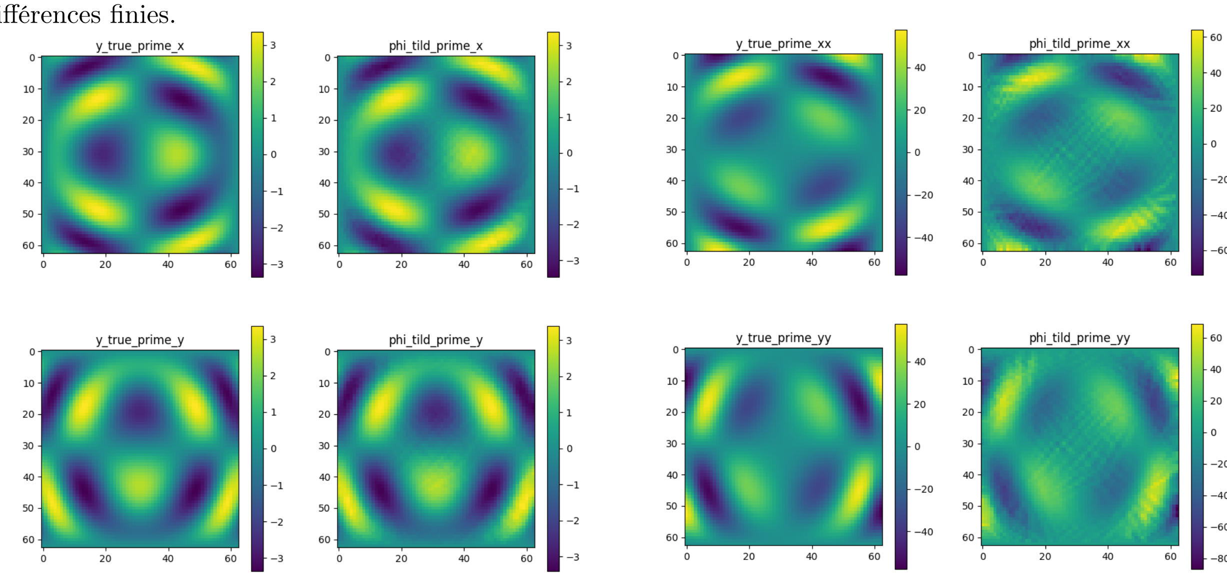


Il semblerait alors que le solveur ne nous donne pas du tout la bonne solution. On dirait que le problème est toujours due au bord mais là c'est bizarre.

Remarque. *En gardant le même code et en prenant une solution perturbée, on obtient les erreurs attendues. Il semblerait donc que le problème ne vienne pas de là.*

20.4.3 Dérivées premières et secondes

Emmanuel pense que le réseau peut avoir du mal à apprendre les dérivées premières et seconde de la solution et que c'est pour ça que la correction ne fonctionne pas. C'est pourquoi, on va afficher ces informations. Pour cela, on utilmisera la classe implémentée par Vincent Vigon pour le FNO qui permet de calculer les dérivées souhaitées par différences finies.



On calcule également les erreurs en normes L^2, H^1 et H^2 (en directe et pas relative). On considérera ces erreurs sur le carré et sur le cercle. Voici les résultats obtenus :

```
||y_true-phi_tild||_L2 : 0.00215408076944087
||y_true-phi_tild||_H1 : 0.08863015252174998
||y_true-phi_tild||_H2 : 4.781150126733509
||y_true-phi_tild||_L2_omega : 0.0010654892116334564
||y_true-phi_tild||_H1_omega : 0.0406639806153612
||y_true-phi_tild||_H2_omega : 2.1473005363805187
```

Remarque. *Il semblerait que le réseau ait vraiment du mal à approcher correctement les dérivées secondes.*

20.4.4 IPP pour la correction par additivité

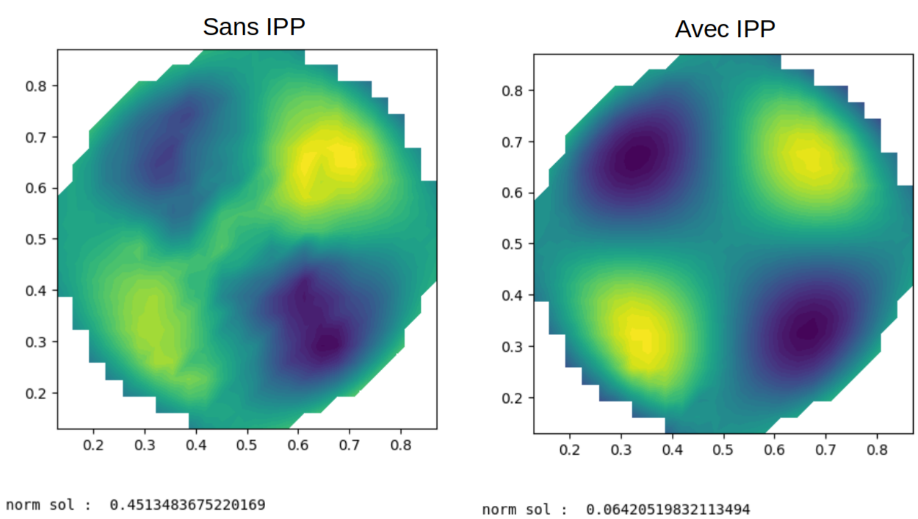
Comme les dérivées semblent ne pas être bien approchés par le réseau, on va modifier la formulation variationnelle de la manière suivante.
On a posé $\tilde{f} = f + \Delta\tilde{\phi}$ ainsi dans le terme à droite, on a

$$\begin{aligned}\int_{\Omega_h} \tilde{f}\phi v &= \int_{\Omega_h} f\phi v + \int_{\Omega_h} \Delta\tilde{\phi}\phi v \\ &= \int_{\Omega_h} f\phi v - \int_{\Omega_h} \nabla\tilde{\phi}\cdot\nabla(\phi v) + \int_{\partial\Omega_h} \frac{\partial\tilde{\phi}}{\partial n}\phi v\end{aligned}$$

Sur une solution analytique perturbée, il semblerait que ce ne soit pas nécessaire :

	f=4,fp=2	f=6,fp=2	f=8,fp=2	f=2,fp=4	f=2,fp=6	f=2,fp=8
add	0.000831	0.000828	0.000834	0.002578	0.004688	0.007026
add v2	0.000831	0.000828	0.000834	0.002578	0.004688	0.007026

Par contre sur le réseau, il semblerait que ça soit bien bénéfique



Conclusion

La semaine prochaine, on relancera des entraînement avec la loss qui prend en compte les dérivées premières. On vérifiera de nouveau le comportement des dérivées.

21 Semaine 21 : 26/06/2023 - 30/06/2023

Résumé

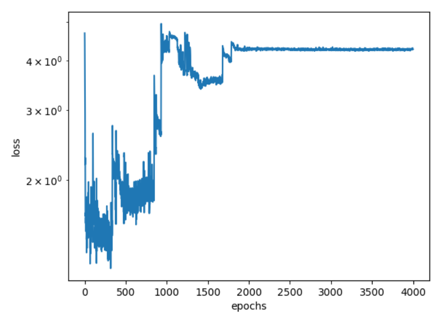
Cette semaine, j'ai enfin compris comment faire les dérivées. En fait, Emmanuel a expliqué que le modèle est en fait une fonction et donc qu'il peut être dérivée, ainsi on connaît les dérivées de $\phi(x,y)w_\theta(x,y)$. L'idée est donc de calculer les dérivées avant le passage en P10 directement avec Tensorflow puis en P10 avec FEniCS.

On a également voulu tester si le problème venait de PhiFEM donc j'ai du tester la correction avec FEM.

Pour finir, j'ai commencé à tester l'implémentation de la loss H1 (avec directement la dérivée par tensorflow), ça tourne mais les loss ne diminue pas.

En fin de semaine, j'ai commencé à rédiger la partie sur les FNO dans le rapport. Il a également été question d'inscription à l'ED (non aboutit, en attente des résultats), j'ai donc dû faire des modifications à mon CV.

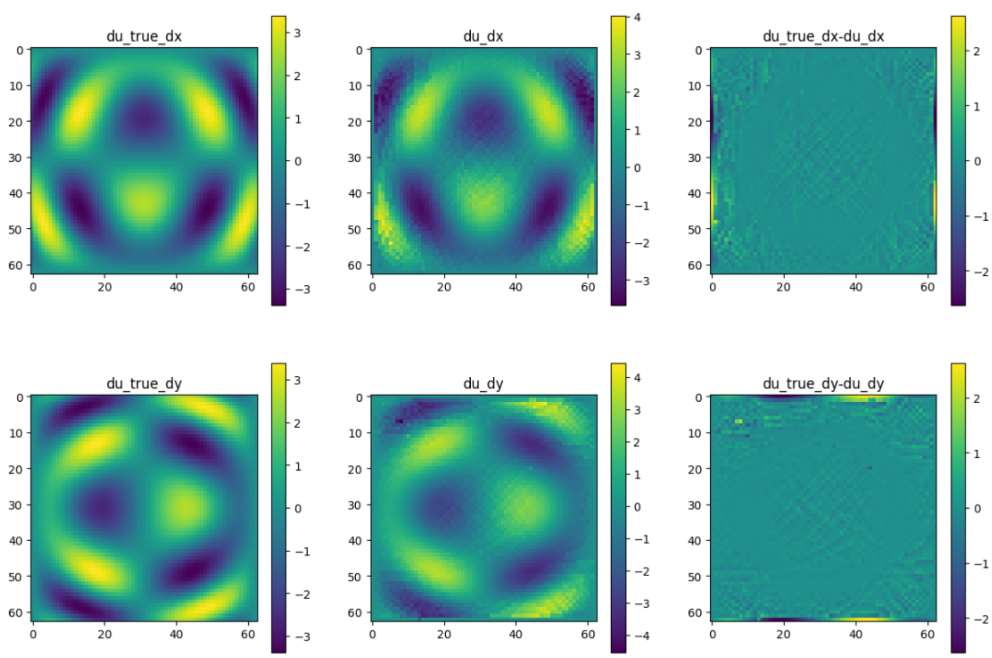
21.1 Entraînement Loss H1



21.2 Dérivées

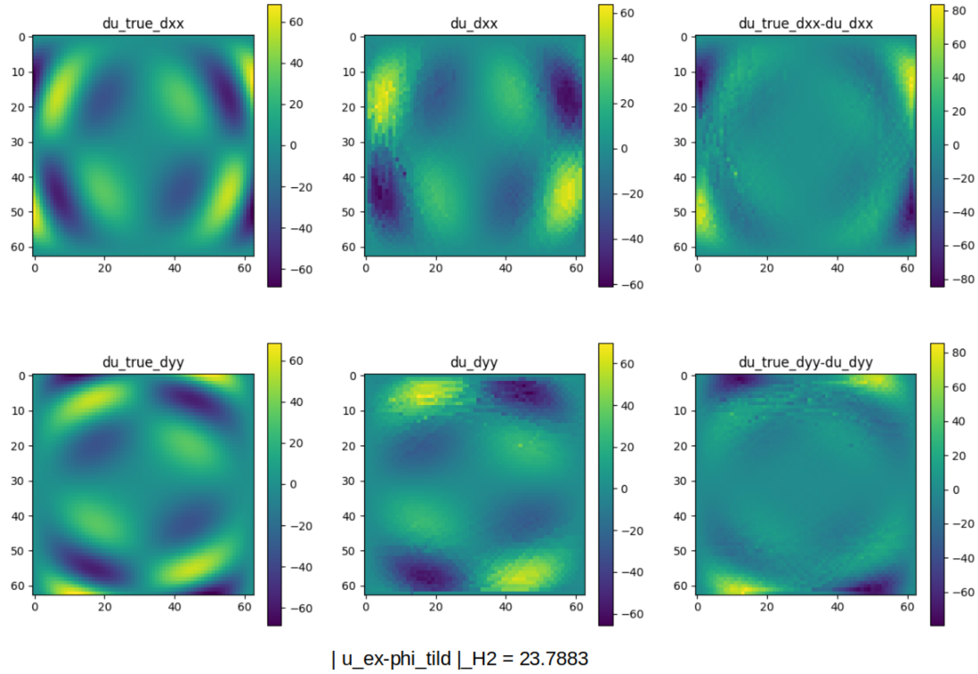
Tensorflow

On va comparer à gauche les dérivées analytiques (calculées avec sympy), au milieu les dérivées du modèle et à droite la différence des deux. Pour le calcul de ces dérivées, on considère un échantillon de points (x,y) en $\mathbb{P}^2$ avec nb_vert=32. On utilise alors GradientTape de tensorflow pour calculer la dérivée de la prédiction du modèle (multipliée par ϕ).
Voici les dérivées premières :



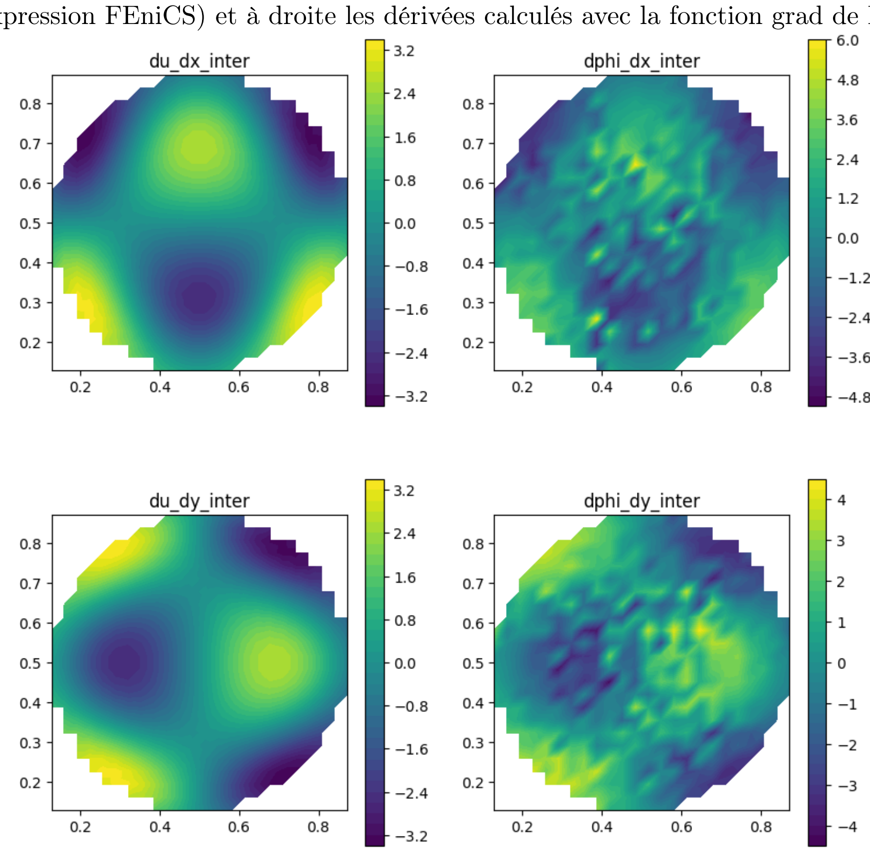
| u_ex-phi_tild |_H1 = 0.4827

Voici les dérivées secondes :

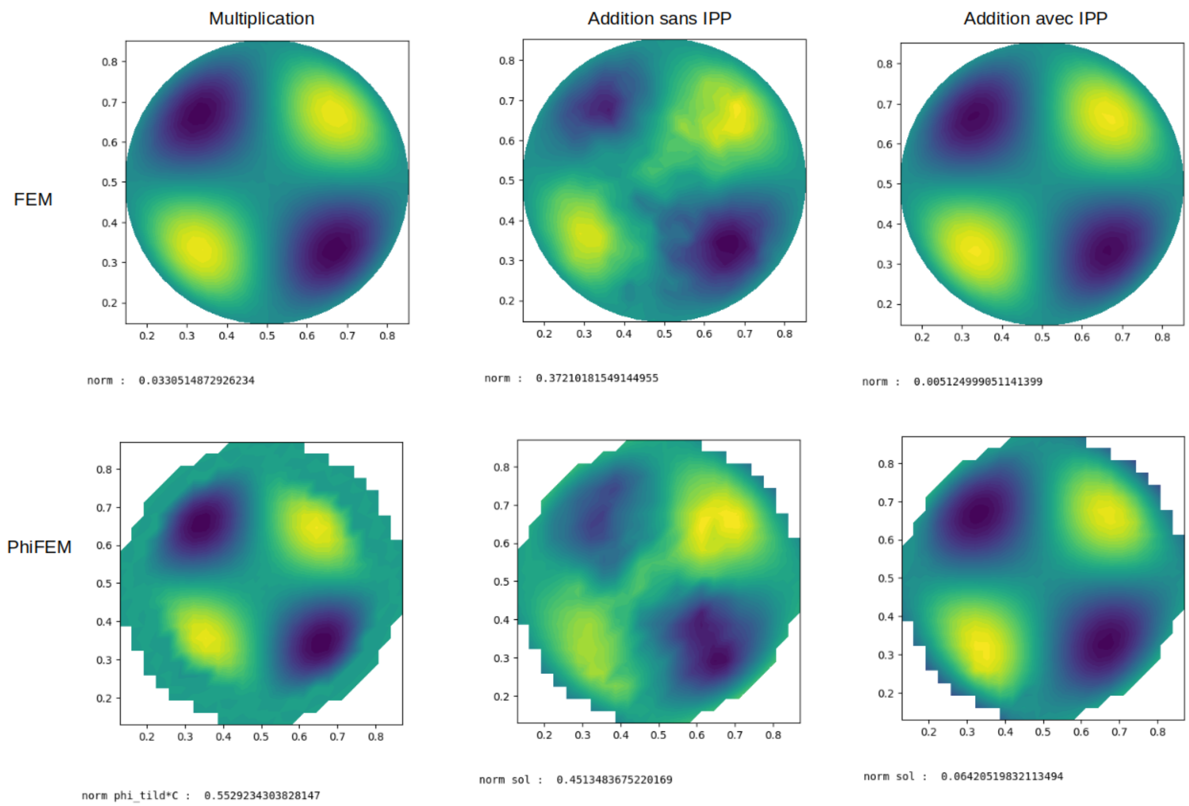


FEniCS

Ici on considère du $\mathbb{P}^{10}$ avec nb_vert=32. On ne s'intéressera ici qu'au dérivée première. A gauche, la solution analytique (avec expression FEniCS) et à droite les dérivées calculés avec la fonction grad de FEniCS.



21.3 Comparaison FEM-PhiFEM



Il semblerait que la correction avec FEM fonctionne mieux qu'avec PhiFEM mais on obtient quand même pas les résultats attendus.

22 Semaine 22-23 : 03/07/2023 - 14/07/2023

Résumé

Je regroupe ici 2 semaines car durant ces 2 dernières semaines, j'ai passé beaucoup de temps sur le rapport dont je n'ai pas encore la date de rendu. Notamment sur la mise en place de la conversion (en python) du rapport latex en un rapport antora et sur la ci permettant de mettre en ligne ce rapport antora à chaque push.

Après discussion avec Emmanuel, il aimerait que je refasse les mêmes tests sur un carré que ceux effectués sur le cercle. J'ai également essayer de continuer à entraîner avec une loss H1, cette fois sur le carré mais ça ne fonctionne pas. Ces résultats sont obtenus sur les 2 semaines.

22.1 Résultats sur le carré

On considère ici la solution analytique suivante

$$u_{ex}(x,y) = S \times \sin(2\pi f x + p) \times \sin(2\pi f y + p)$$

avec $S = 0.5$, $f = 1$ et $p = 0$. Solution **homogène** sur notre domaine, le carré $[0,1]^2$. On prendra $\mathcal{O} = [-0.5, 1.5]^2$.

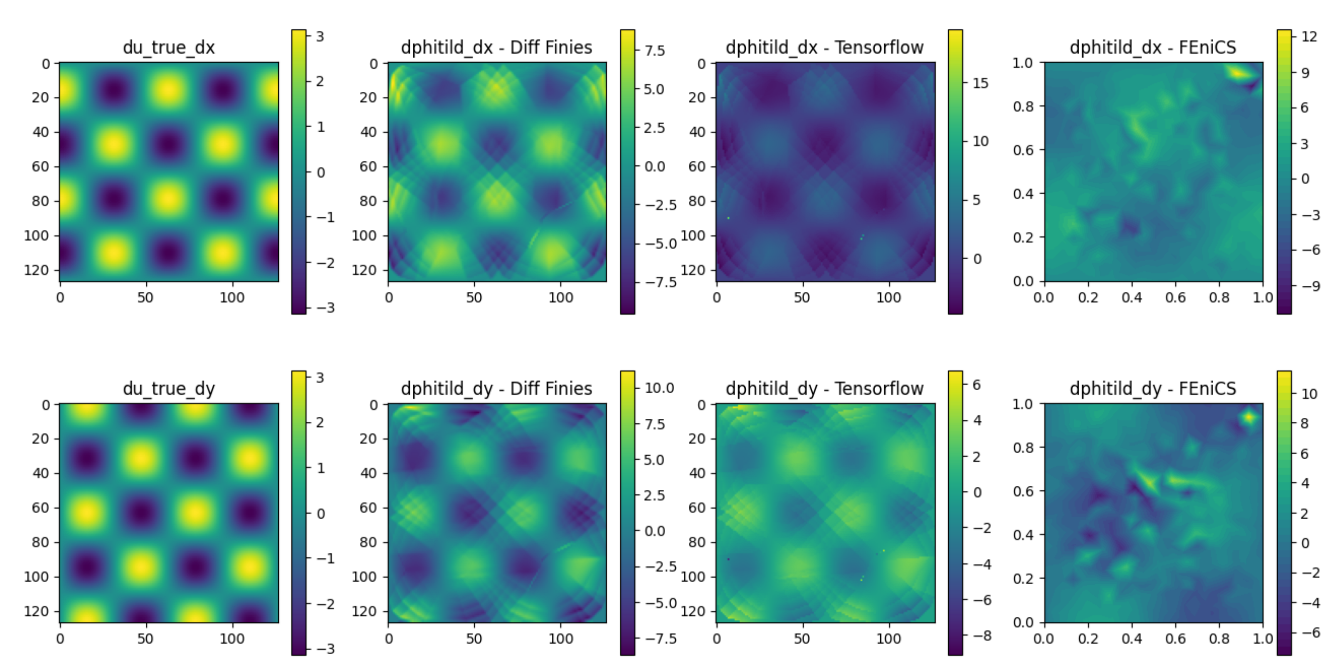
Dérivées

Voici les normes H1 et H2, obtenus avec les différentes méthodes :

```
||y_true-phi_tild||_L2 : 0.01440731535342013
### Différences finis :
||y_true-phi_tild||_H1 : 2.237165231760222
||y_true-phi_tild||_H2 : 90.66799465709268
### Tensorflow :
||y_true-phi_tild||_H1 : 0.6076613060500831
||y_true-phi_tild||_H2 : 7.925625390876353
### FEniCS :
||u_ex-phi_tild||_L2 : 0.012043277209272478
||u_ex-phi_tild||_H1 : 0.6661866563445389
||u_ex-phi_tild||_H2 : 656.9648518678864
```

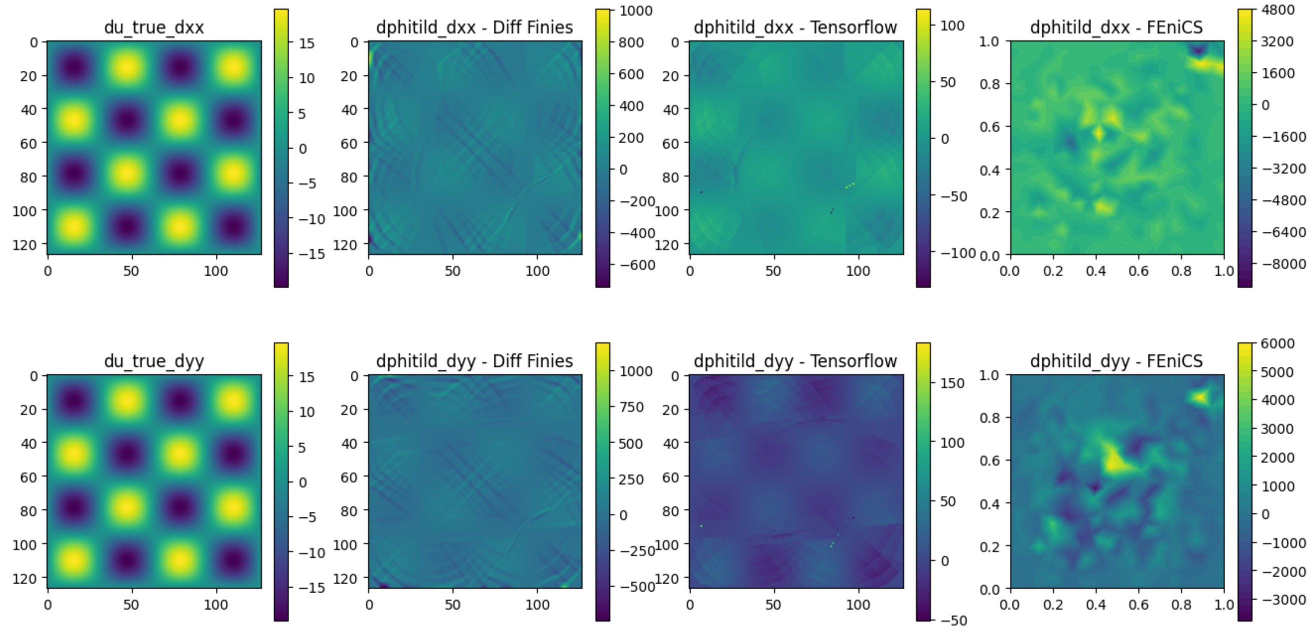
Voici les dérivées premières :

```
Dérivées premières :
```

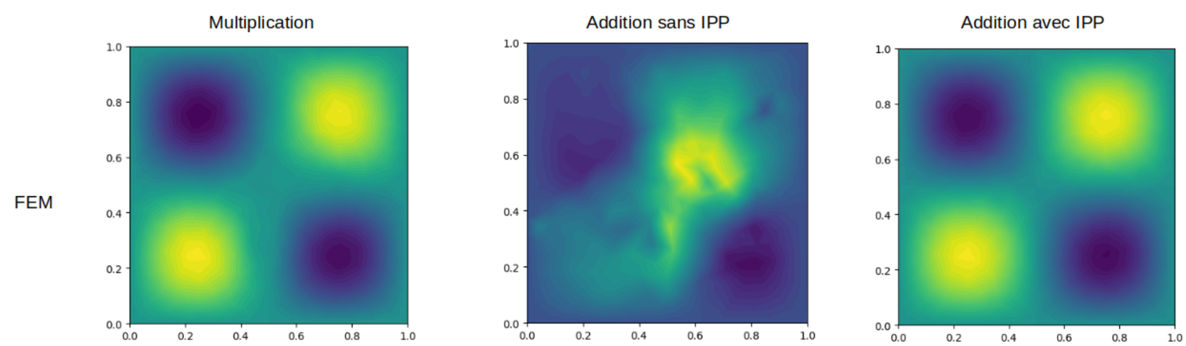


Voici les dérivées secondes :

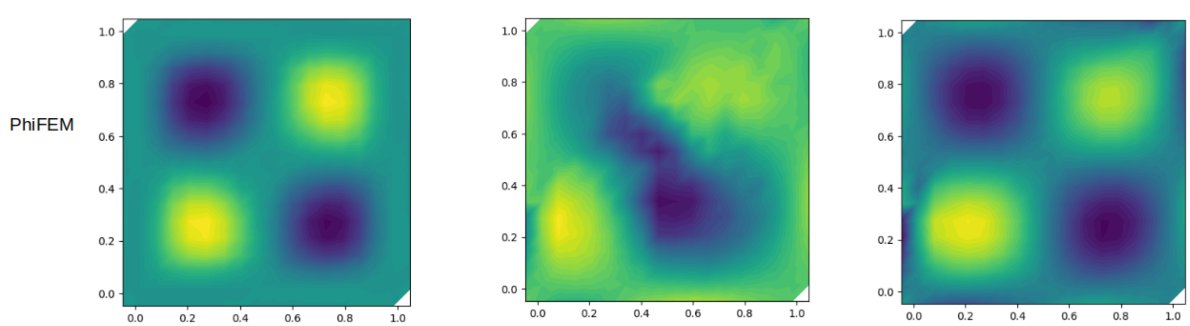
```
Dérivées secondes :
```



Comparaison FEM-PhiFEM



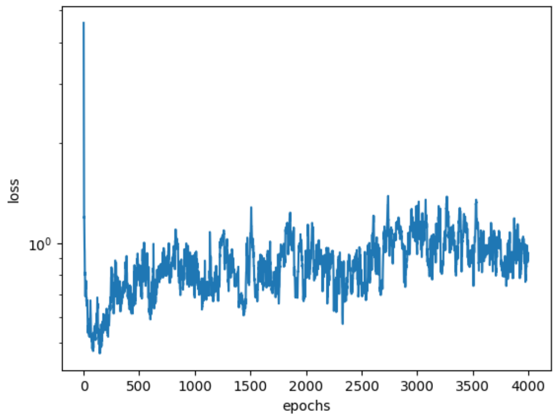

```
norm :  0.13183670211920356
norm :  1.696209732359107
norm :  0.01863286720106959
```



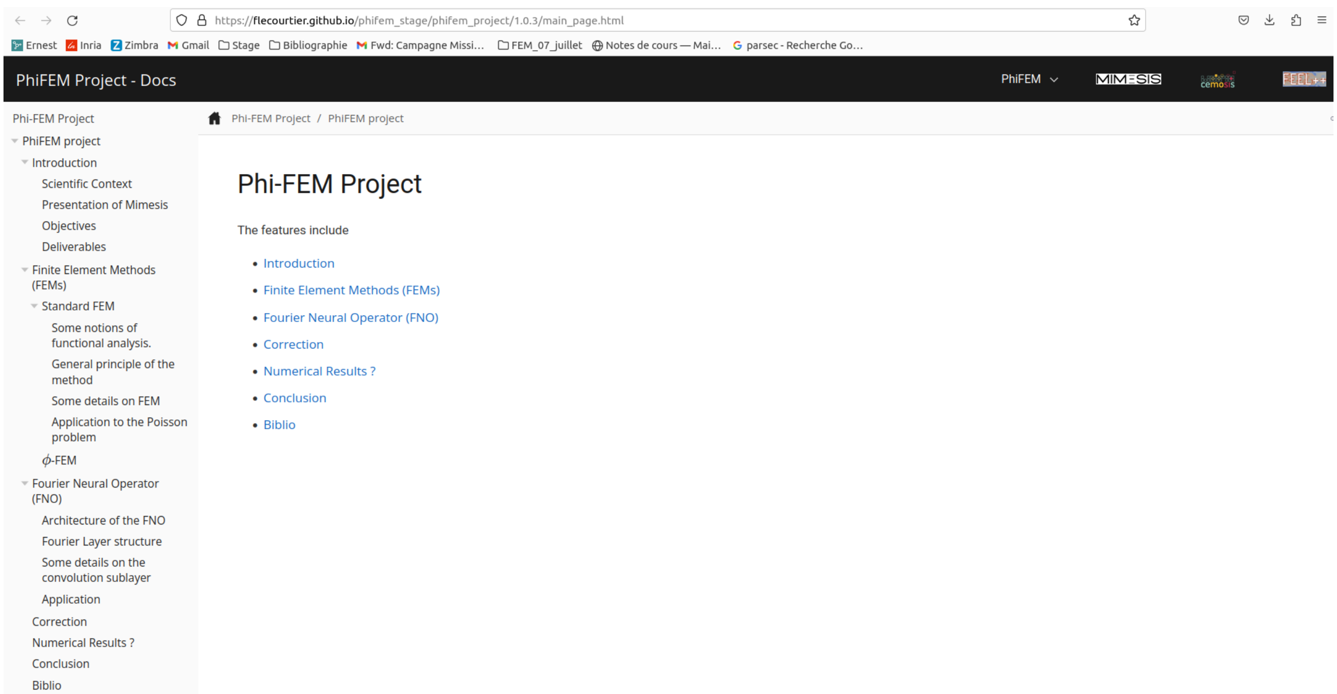

```
norm phi_tild*C :  0.3500309930168719
norm sol :  2.048758089279502
norm sol :  0.2399808730886504
```

22.2 Entraînement - loss H1

```
num_dense : 2
model_dir : /home/lecourtier/Bureau/Modele_dense/model_carre/sol_analytique_w_loss_H1/model_2/
4000/4000 [==============================] - 4460s 1s/step - loss: 0.9292
```



22.3 Documentation Antora



Conclusion

Pour la doc antora, il faut encore faire quelques modifications. Notamment enlever cemosis, FEEL++...