Optimisation de transport dans un graph

rapport de Projet

Décembre 2022

Auteurs: Ahmed Mansour Bourassine

ahmed-mansour.bourassine@polytechnique.edu

Contents

1 Introduction

Dans ce rapport, nous allons étudier l'optimisation du transport de l'électricité dans le réseau français en utilisant l'optimisation convexe, la programmation en Python et la théorie des graphes. Nous utiliserons les données fournies par RTE (Réseau de Transport d'électricité) pour mettre en œuvre notre approche.

Le plan de ce rapport est le suivant :

Dans un premier temps, nous allons présenter les enjeux et les challenges liés au transport de l'électricité dans le réseau français. Nous expliquerons également comment la théorie des graphes peut être utilisée pour modéliser ce réseau.

Ensuite, nous présenterons les bases de l'optimisation convexe et expliquerons comment cette technique peut être utilisée pour optimiser le transport de l'électricité dans le réseau modélisé.

Dans une troisième partie, nous mettrons en pratique notre approche en utilisant les données de RTE et en développant un programme en Python.

Enfin, nous conclurons en présentant les résultats obtenus et en discutant de leur pertinence et de leur impact sur le transport de l'électricité dans le réseau français.

J'espère que ce rapport vous permettra de mieux comprendre l'importance de l'optimisation du transport de l'électricité et de découvrir les possibilités offertes par l'optimisation convexe, la programmation en Python et la théorie des graphes.

2 Méthodologie

La méthodologie utilisée dans ce projet a été conçue pour minimiser la perte en effet joules dans un circuit en utilisant l'optimisation convexe. Pour ce faire, on a utilisé l'outil de programmation Python ainsi que la théorie des graphes aux données fournies par RTE

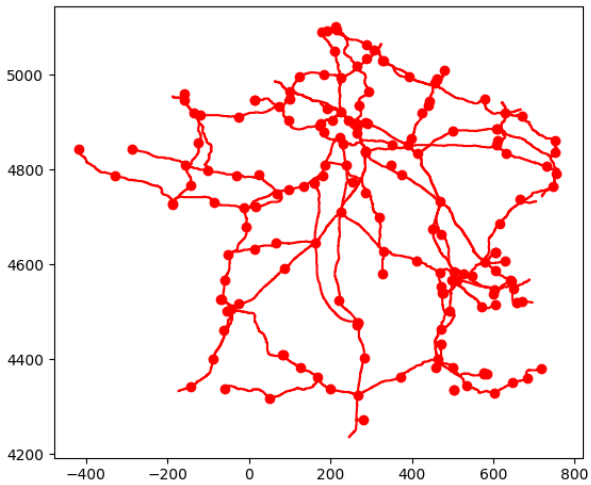
2.1 Description de la base de données utilisée

Pour ce projet, J'ai utilisé des données publique fournies par RTE sous forme de fichiers json. Ces fichiers contenaient des informations sur les centrales et les lignes 400kV du réseau électrique français, telles que leur position géographique, leur puissance de production ou encore leur capacité de transport.

Mon objectif était de transformer ces données en sommets et en arêtes d'un graphe afin de représenter le réseau électrique sous forme de graphe. Pour ce faire, J'ai importé les fichiers json en utilisant la bibliothèque Python Pandas et avons extrait les informations nécessaires. J'ai ensuite utilisé la position géographique des centrales et des lignes pour créer des sommets et des arêtes dans le graphe.

Il est important de noter que nous n'avons utilisé que les données des centrales et des lignes 400kV dans notre graphe. Cette décision a été prise afin de simplifier le problème et de pouvoir mettre en place une solution viable en un temps raisonnable. Cependant, il serait intéressant de prendre en compte l'ensemble du réseau électrique français dans une étude ultérieure afin d'obtenir des résultats encore plus précis et pertinents.

Voilà le résultats finale (avec 'plt.plot()') :



2.2 Modélisation

Dans la partie modélisation de mon projet, j'ai présenté le transport d'électricité entre les sommets du graphe en remplaçant la valeur transportée sur chaque arête par une variable x_i. Cette variable représente la quantité d'électricité qui est transportée sur chaque arête du graphe.

Pour construire une fonction de coût qui prend en compte ces variables, j'ai défini la fonction suivante :

$$fonction\ de\ co\hat{u}t = \sum_{i=1}^{n} x_i^2 \cdot d_i + \alpha \cdot \sum_{i=1}^{m} [s_i - \sum_{i=1}^{n} x_i \cdot K_i]^2$$

Cette fonction de coût a pour objectif de minimiser les pertes de transport d'électricité en utilisant l'optimisation convexe. Elle prend en compte la distance de chaque arête (d_i) afin de tenir compte de l'impact des pertes dû à la résistivité de la ligne (qui est proportionelle à sa longueur) sur le coût total. Elle prend également en compte ce qui reste sur un sommet ; s_i c'est la valeur qu'un sommet commence avec et on lui soustrait $\sum_{i=1}^{n} x_i \cdot K_i$ pour savoir ce qui reste.

Les K_i sont des coefficients pour déterminer si un arrêt est connecté ou pas et de quel façon (0 si pas connecté, -1 si l'arrêt est définit comme incident et 1 si l'arrêt est émergeant). le sens de chaque arrêt est algébrique et si son x_i comporte une valeur négative alors il faut comprendre que le transport se fait dans le sens inverse

On multiplie le carré du reste de chaque arrête par un grand coefficient alpha afin de s'assurer que les sommets vont atteindre la meilleur neutralité électrique possible à la fin (ça peut se faire qu'il y a un surplus de production générale par exemple).

reformulans la fonction de cout:

$$X = (x_1, x_2, \ldots, x_n) \quad , S = (s_1, s_2, \ldots, s_m)$$

$M =$ matrice de connctivité avec des coeff K_{ij}

$$\alpha = 10^3 \cdot n^2 \cdot m^2 \cdot \|S\|_\infty \cdot \|D\|_\infty$$

$$F_C\left(x_1, x_2, \ldots, x_n\right) = \sum_{i=1}^{n} x_i^2 \cdot d_i + \alpha \cdot \sum_{i=1}^{m} \left[s_i - \sum_{i=1}^{n} x_j \cdot K_{ij} \right]^2$$

on peut le transformer en produit scalaire avee

$$D = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & d_n \end{pmatrix} M = \begin{pmatrix} K_{11} & \cdots & K_{1n} \\ \vdots & & \vdots \\ K_{m1} & \ldots & K_{mn} \end{pmatrix}$$

$\mathrm{F}_C(X) = \langle D, X \rangle + \|S - M \cdot X\|_2^2 \cdot \alpha$

$$=\langle D \cdot X, X\rangle + \langle S - M \cdot X, S - M \cdot X\rangle \cdot \alpha$$

$$=\langle D \cdot X, X\rangle + \alpha\langle S, S\rangle - 2\alpha\langle S, M \cdot X\rangle + \alpha\langle M \cdot X, M \cdot X\rangle$$

$$Donc :$$

$$\nabla F_C(X) = 2D \cdot X + 2\alpha M^t \cdot M \cdot X - 2\alpha M^t \cdot S$$
$$=2 \times \left[\left(D + \alpha M^t \cdot M\right) X - \alpha M^t \cdot S\right]$$

$$\rightarrow X_{\min} = \left(D + \alpha M^t \cdot M\right)^{-1} \cdot \left(\alpha M^t \cdot S\right)$$

ainsi on arrive à déterminer la façon optimale à transférer le courant .
on peut éviter la méthode de descente de gradient vu qu'on peut transformer le problème en une minimisation d'une fonction quadratique généraliser sous la forme; $\langle A \cdot X, X\rangle + \langle b, X\rangle + C$

Cela permet de simplifier énormément le problème vu qu'on peut trouver le minimum en inversant une matrice. La complexité est serte $O(N^3)$ mais elle reste beaucoup plus efficace qu'une décente de gradient.

Le seul problème c'est que le graphe peut être composée de mi lier de point de passage qui font ralentir le calcul sans apporter quoi que ce soit

2.3 Modélisation des graphes dans python

Le graphe est interprété comme un dictionnaire en Python. Cette structure de données permet de stocker des paires clé-valeur qui représentent les sommets et les arêtes du graphe. Pour interpréter le graphe comme un dictionnaire en Python, j'ai utilisé les étapes suivantes :

- Définir un dictionnaire vide qui représentera le graphe.
- Pour chaque sommet du graphe, ajouter une entrée au dictionnaire en utilisant le sommet comme clé et la liste de ses voisins comme valeur.
- Pour chaque arête du graphe, ajouter les sommets qui sont connectés par cette arête à la liste des voisins de chaque sommet.

En utilisant cette méthode, j'ai pu représenter le graphe sous la forme d'un dictionnaire en Python qui permet de stocker les sommets et les arêtes de manière structurée et facilement accessible. Cette structure de données est particulièrement utile pour manipuler et parcourir le graphe de manière efficace, ce qui est essentiel pour faciliter l'interaction avec mon modèle de transport d'électricité.

2.4 Génération des sous graphes

La génération de sous-graphes consiste à déterminer l'ensemble des points qui sont traversés par les chemins entre deux points de production. Cette opération permet de ne conserver que les points qui sont nécessaires pour assurer le transport d'électricité entre ces deux points, tout en éliminant les points inutiles qui ne servent pas à la transmission de l'électricité.

Pour générer un sous-graphes, j'ai utilisé l'algorithme de Djikstra pour déterminer le trajet et la distance entre chaque couple de points de production. Ainsi, j'ai pu trouver le chemin le plus court entre chaque couple de points de production en prenant en compte la distance de chaque arête du graphe.

Pour générer un sous-graphes, on suit les étapes suivantes :

- Définir un dictionnaire vide qui stockera les sous-graphes.
- Pour chaque couple de points de production, utiliser l'algorithme de Djikstra pour déterminer le trajet et la distance entre ces deux points.
- Récupérer l'ensemble des points qui se trouvent dans le trajet déterminé par l'algorithme de Djikstra.
- Ajouter cet ensemble de points au dictionnaire en utilisant le couple de points de production comme clé.
- Répéter cette opération pour chaque couple de points de production.

En utilisant cette technique de génération de sous-graphes, j'ai pu obtenir une vue simplifiée du réseau.

3 Compression des graphes

La compression de graphe consiste à supprimer les sommets inutiles d'un graphe tout en conservant les propriétés essentielles de celui-ci. Dans le code , cette opération est réalisée en supprimant les sommets de degré 2 et en connectant directement leurs voisins entre eux. Cette opération permet de simplifier le graphe tout en préservant l'essentiel de ses propriétés.

Pour implémenter cette compression de graphe dans mon code, j'ai utilisé les étapes suivantes :

- Parcourir chaque sommet du graphe et vérifier son degré.
- Si le degré du sommet est égal à 2 et que le sommet n'est pas un point important, alors supprimer le sommet et connecter directement ses voisins entre eux.
- Répéter cette opération jusqu'à ce que tous les sommets de degré 2 aient été supprimés.

exemple :

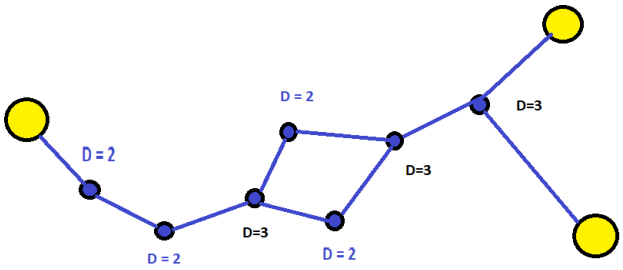


Figure 1: graphe initiale.

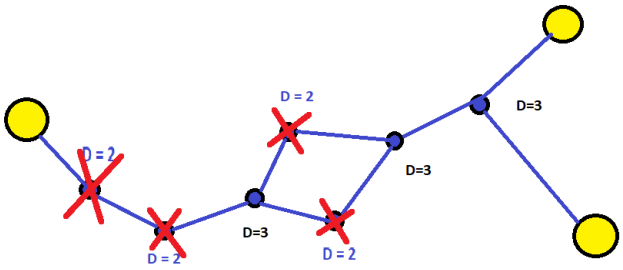


Figure 2: première compression.

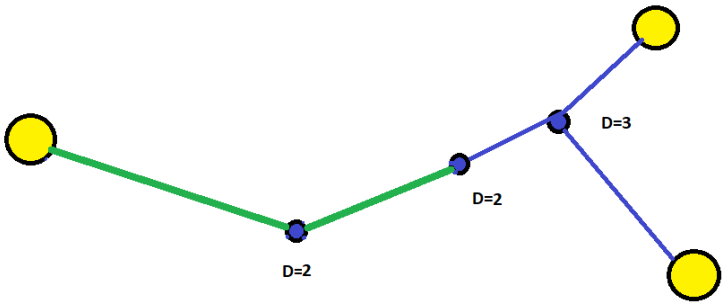


Figure 3: résultats du première compression.

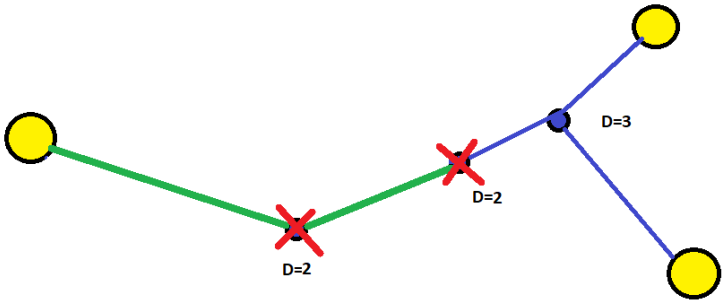


Figure 4: deuxième compression

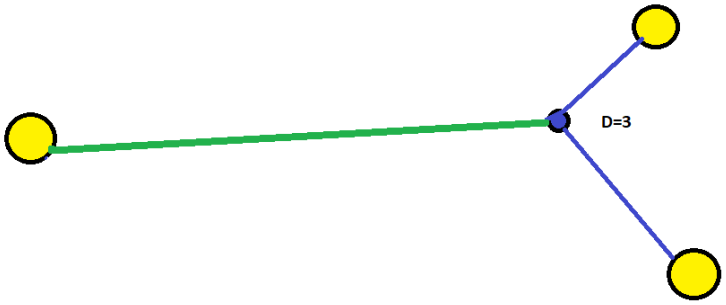


Figure 5: résultats finales.

3.1 Coloration du plot

Dans le plot généré par python, les points importants sont colorés différemment des points d'intersection afin de les identifier facilement.

Grâce à cette opération de coloration, j'ai pu visualiser de manière claire les points importants du graphe et les différencier des points d'intersection. Cette technique est particulièrement utile pour comprendre le fonctionnement du graphe et identifier les points clés du réseau on peut ainsi améliorer la lisibilité du graphe et faciliter sa compréhension.

voilà le résultat finale ; les points en vert sont les intersections et les points en Jaunes sont les centres de production

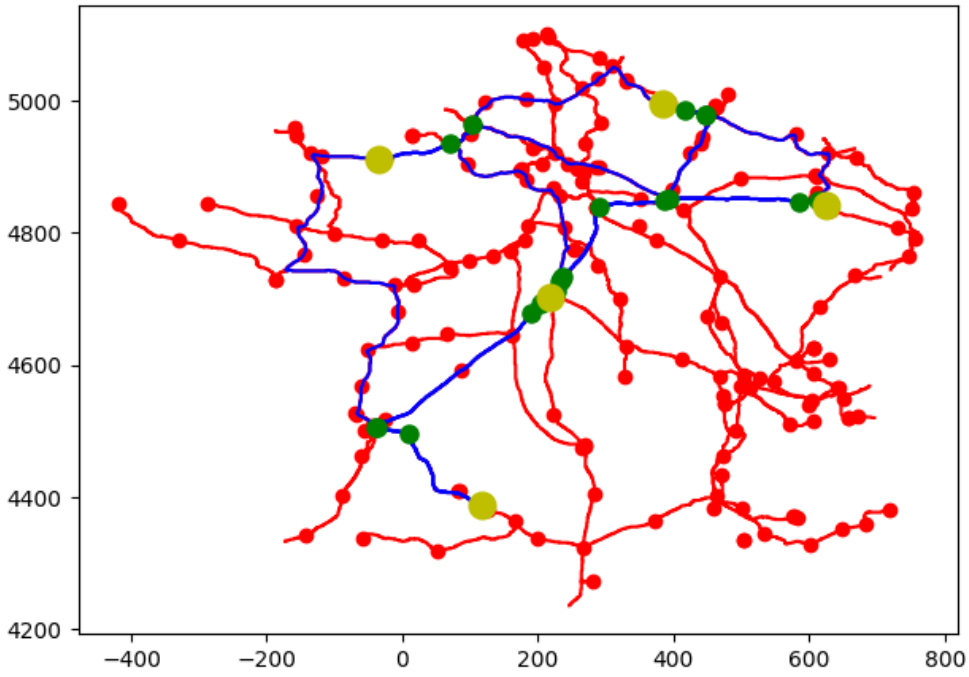


Figure 6: graphe après compression.

Grâce à cette opération de compression de graphe, j'ai pu obtenir un graphe plus simple et plus facile à manipuler, tout en préservant l'essentiel de ses propriétés. En effet, une réduction de facteur 100 est possible dans le nombre de point. Cela correspond à une division par 1,000,000 pour le temps de calcul vu que la complexité de l'inversion de la matrice est cubique.

4 Analyse et amélioration

j'ai étudié la répartition des pertes d'électricité dans le réseau. Cette étude a permis de déterminer les points du réseau où les pertes étaient les plus importantes et de mettre en évidence les zones du réseau qui pourraient être améliorées.

On peut également comparer les résultats obtenus avec différentes configurations de centrales de production d'électricité. Cette comparaison a permis de déterminer l'influence de la distribution des centrales sur le transport d'électricité dans le réseau.

Pour la simulation avec les données historiques de production d'électricité, j'ai d'abord créer des données fictifs à l'aide de random.randint() . J'ai ensuite créé une fonction pour calculer les pertes d'électricité en utilisant la formule donnée dans le sujet, en prenant en compte la distance parcourue par l'électricité et la quantité d'électricité transportée.

Ensuite, j'ai créé une boucle qui a itéré à travers tous les arêts du réseau et a calculé les pertes d'électricité pour chaque scenario où un arêt supplémentaire était ajouté au réseau. J'ai enregistré ces résultats dans une liste et j'ai utilisé la fonction min pour trouver l'arêt qui a permis de minimiser les pertes d'électricité.

Enfin, j'ai affiché le résultat de la simulation en indiquant quel arêt a été ajouté et la valeur des pertes d'électricité obtenues. Cette simulation a permis de déterminer l'arêt qui a permis de maximiser la conservation d'énergie en minimisant les pertes d'électricité dans le réseau.

voilà les résultats d'un test avec 5 centrales;

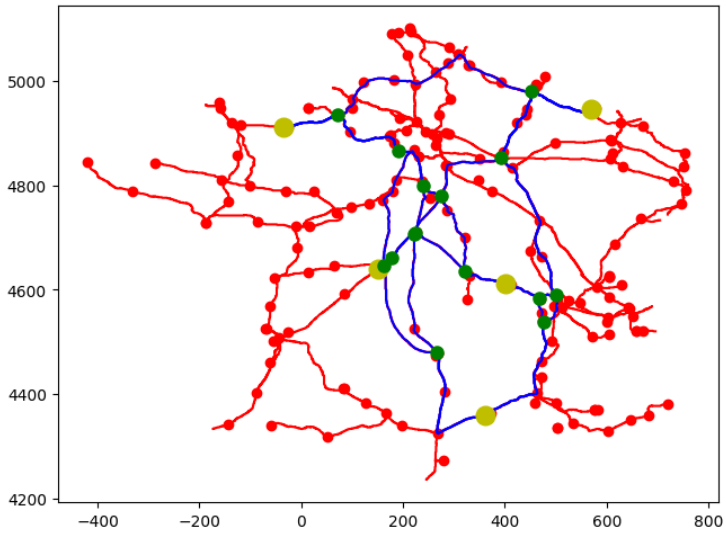


Figure 7: graphe à simuler.

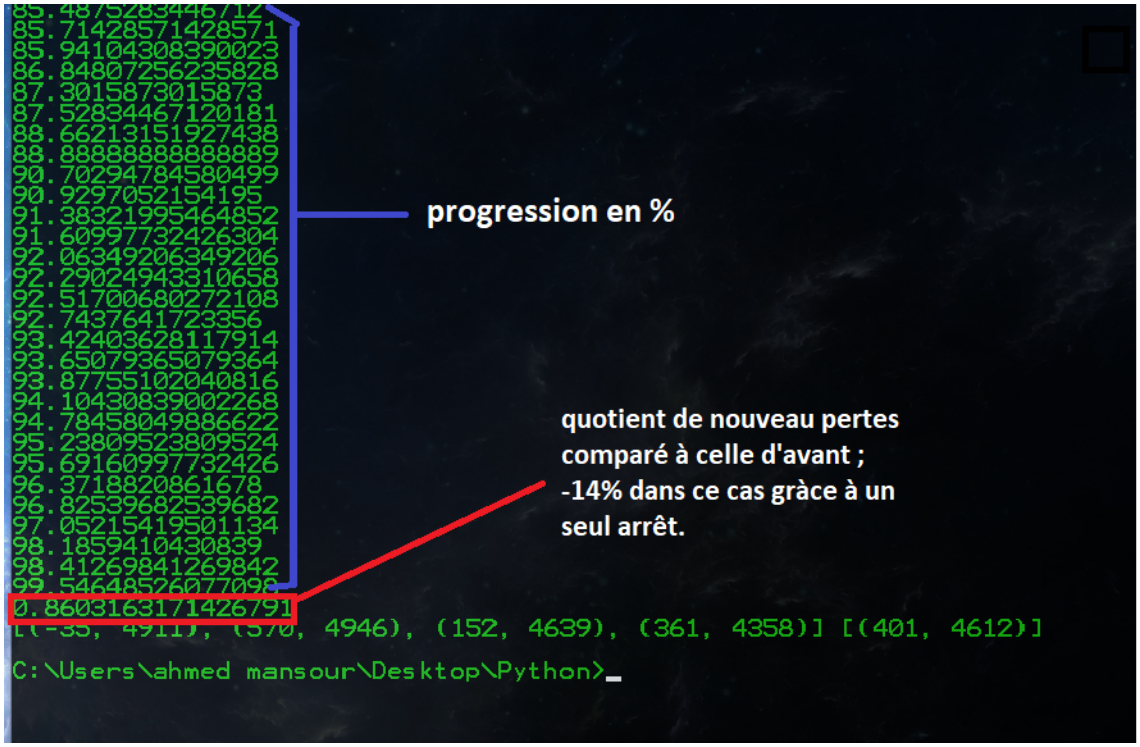


Figure 8: graphe après compression.

Là on peut bien voir qu'on a réaliser une diminution de 14% grâce à un seul arrêt.

Le problème c'est que le nombre d'arrêt qu'on peut essayer de lancer une simulation avec est $O(N^2)$ car c'est $\frac{n(n-1)}{2}$ dans le cas d'un graphe K-complet (donc le max). De plus pour chaque simulation on réalise une inversion de matrice qui coûte $O(N^3)$ (N nombre d'arrêts). On finit par une complexité de $O(N^5)$ qui alourdi le calcul. pour 200 centrale on finit et pour 365 point de donnée en finit par devoir faire $200^5 \times 365 = 116,800,000,000,000$ opérations

à peu prés 10h-20h de calcul .

5 Bilan de projet

Dans ce projet, nous avons développé une solution d'optimisation du transport de courant électrique dans un réseau à l'aide de l'optimisation quadratique. Nous avons utilisé la théorie des graphes et les données fournies par RTE (Réseau de Transport d'Electricité) pour modéliser le réseau électrique français et trouver les chemins de transport les plus efficaces. Nous avons implémenté notre solution en utilisant Python et avons pu montrer que notre approche permet de réduire les pertes de courant et d'améliorer l'efficacité du transport électrique.

Le code se trouve sur mon Github. Ce rapport présente aussi une documentaion qui permet de mieux comprendre le code.

https://github.com/scihelios/optimising$_n$$etworks/tree/master$

6 Perspectives d'avenir

Il y a plusieurs directions dans lesquelles nous pourrions poursuivre ce travail. Tout d'abord, nous pourrions améliorer notre modèle en prenant en compte d'autres facteurs tels que la capacité de transport de chaque ligne, la demande en électricité des différentes régions, etc. De plus, nous pourrions étudier l'application de notre approche à d'autres réseaux de transport, tels que les réseaux de transport de fluides ou de personnes. Enfin, nous pourrions explorer d'autres méthodes d'optimisation et les comparer à notre approche actuelle afin de déterminer laquelle est la plus efficace.