TP 3 – Systèmes et Traitements Répartis GIS 5
Opérations entre matrices et vecteurs

Il existe d'autres fonctions, plus élaborées, qui peuvent s'avérer utiles : `MPI_Allgather`, `MPI_Allreduce`, `MPI_ReduceScatter`, `MPI_Alltoall`. *N'hésitez pas à les découvrir dans la documentation MPI. Vous pouvez utiliser certaines d'entre elles dans ce TP.*

Vous pouvez récupérer un code source sur la page web de Charles Bouillaguet, vous permettant de vous concentrer plus sur l'essentiel de l'exercice que sur de la programmation rudimentaire en C.

Exercice 1 : Produit matrice – vecteur *(A*x)*
Le but de cet exercice est d'écrire un code le plus optimal possible effectuant un produit matrice – vecteur (*A*x*) de trois façons différentes. Les coefficients de la matrice *A* et du vecteur *x* seront des doubles construits aléatoirement sur les processus qui en auront besoin (créer des fonctions à cet effet).

1ère méthode : Distribuer *A* par blocs de lignes sur les différents processus.
2ème méthode : Distribuer *A* par blocs de colonnes sur les différents processus.

Comparer ces méthodes, en terme de communication et de temps d'exécution.

Exercice 2 : *A*(A*x+x) + x*
Écrire un code effectuant l'opération *A*(A*x+x) + x* pour une matrice *A* et un vecteur *x* étant des doubles construits aléatoirement à l'aide des deux méthodes de l'exercice 1.

Exercice 3 : Calcul du vecteur propre dominant d'une matrice stochastique *(A*...*A*A*x)*
Dans cette exercice, *A* est une matrice stochastique, c'est-à-dire une matrice telle que la somme des coefficients de chaque ligne vaut 1 et chaque coefficient est positif ou nul. Les coefficients de la matrice stochastique *A* et du vecteur *x* unitaire (de norme 1) seront des doubles construits aléatoirement par chaque processus (reprendre les fonctions de l'exercice 1).
Le but de cet exercice est de calculer le vecteur propre dominant de *A* qui s'obtient à l'aide de l'algorithme suivant :

```
tant que ||y - x|| > epsilon
       x = y
       y = A*x
       y = y / ||y||
return y
```

1) Créer une fonction retournant la norme d'un vecteur.
2) Écrire deux versions du code, inspirées chacune d'une méthode de l'exercice 1, les plus optimales possible, calculant le vecteur propre dominant de *A*.

Exercice 4 : 3ème méthode – distribuer A par blocs sur les différents processus (facultatif)
Refaire les exercices 1 à 3 avec cette méthode. Cet exercice est difficile et facultatif.

Voici des pistes de réflexions :
1) Pour bien gérer les communications, on peut créer de nouveaux communicateurs qui communiqueront entre eux et pas avec les autres (des familles de processus). On pourra s'intéresser aux fonctions suivantes : `MPI_Cart_Create`, `MPI_Cart_coords`, `MPI_Comm_split`.
2) On peut aussi considérer une matrice à 3 dimensions : les 2 premières dimensions sont les coordonnées des processus, la 3ème la sous-matrice de A.