|  |
| --- |
| ***#1 Lecture et connectivité du domaine global jusqu’à elem2elem***  Entrées :   * *Fichier SU2 global*   Sorties :   * *npoin\_g, nelem\_g, ncells\_g* * *nbound\_g (nombre de condition limite)* * *ghost2node\_g int[nb de condition limite][nghost][nnoel+1]* * *node2elem\_g* * *node2node\_g* * *elem2elem\_g* |

|  |  |
| --- | --- |
| ***#2 Calcul du nombre d’élément par zone*** | |
| ***Entrées :***   * *nelem\_g* * *elem2zone* | ***Sorties :***   * *zone2nelem* |
| Initialisation de zone2nelem= [0,0,0,0]  For *ielem* à *nelem\_g*  izone = elem2zone[*ielem*]  zone2nelem[*zone*] += 1 | |

|  |  |
| --- | --- |
| ***#3 Création des éléments locals de chaque zone***  ***Description :*** | |
| ***Entrées :***   * ***nzone, nelem\_g*** * ***zone2nelem*** | ***Sorties :***   * ***zone2elem*** |
| Initialisation :  For izone à nzone  zone2elem[izone] = new int[zone2nelem[*izone*]]  int indexZone = [0,0,0,0]  For *ielem* à *nelem\_g*  izone = elem2zone[*ielem*]  idz = indexZone [*izone*]  zone2elem[*zone*][*idz*] = ielem  indexZone [*zone*] += 1 | |

|  |  |
| --- | --- |
| **#4 Création des nœuds locaux**  **Description :**  *Pour chaque nœud, vérifier les zones connectées et enregistre le nœud localement pour chaque zone.*  *Cette étape doit être fait* ***deux fois****: une fois pour calculer zone2npoin et initialiser zone2node et une fois pour calculer zone2node.* | |
| **Entrées :**   * *elem2zone* * *node2elem\_g* * *npoin\_g* | **Sorties :**   * *zone2npoin* * *zone2node* |
| Initialisation de checkZone  [0,0,0,0] et indexZone [0,0,0,0]  For *inode* à *npoin\_g*  nelemArnode = fonction existante de node2elem\_g (nombre d’élément autour du nœud)  for *ielem* à nelemArnode  ielem\_g = fonction existante (numéro d’élément global)  izone = elem2zone[ielem\_g]  if checkzone[*izone*] == 0 *//Vérifie si on n’a pas déjà compter le nœud dans la zone*  idz = IndexZone[*izone*]  zone2node[izone][idz] = inode  IndexZone[i*zone*] += 1  checkzone[izone] = 1  Reset de checkzone à [0,0,0,0] | |

|  |  |
| --- | --- |
| **#5 Connectivité Élément Global 🡪 Local** | |
| **Entrées :**   * *nzone* * *zone2nelem* * *zone2elem* | **Sorties :**   * *elemglobal2local = [nelem\_g][2]* |
| For izone à nzone  For ielemZ à zone2nelem[izone]  ielemG= zone2elem[izone][ielemZ]  elemglobal2local[ielemG][0] = ielemZ  elemglobal2local[ielemG][1] = izone | |

|  |  |
| --- | --- |
| **#6 Connectivité Noeuds Global 🡪 Local**  **Description :**  *Pour chaque nœud global, cette matrice donne les nœuds locaux associées dans chaque zone (colonne). Si le nœud global n’est pas dans la zone, alors l’indice -1 doit être enregistré dans la matrice.* | |
| **Entrées :**   * *nzone* * *zone2npoin* * *zone2node* | **Sorties :**   * *nodeglobal2local = [npoin\_g][nzone]* |
| For *izone* à nzone  For *inode\_z* à zone2npoin[*izone*]  inode\_g= zone2node[*izone*][*inode\_z*]  nodeglobal2local[inode\_g][*izone*] = *inode\_z* | |

|  |  |
| --- | --- |
| **#7 Création de Coord pour chaque zone** | |
| **Entrées :**   * *nzone* * *coord\_g* * *zone2npoin* * *zone2node* | **Sorties :**   * *zone2coord = double[nzone][npoin\_z][3]* |
| For *izone* à nzone  For *inode* à zone2npoin[*izone*]  inode\_g = zone2node[*izone*][*inode*]  zone2coord[*izone*][*inode*][0] = coord\_g[inode\_g][0]  zone2coord[*izone*][*inode*][1] = coord\_g[inode\_g][1]  zone2coord[*izone*][*inode*][2] = coord\_g[inode\_g][2] | |

|  |  |  |  |
| --- | --- | --- | --- |
| **#8 Création de elem2node pour chaque zone**  **Description :**  *Pour chaque élément global, le code détermine ses nœuds locaux et la zone et remplis elem2node pour cette zone. En même temps, le code vérifie si les éléments voisins appartiennent à une autre zone, si oui, alors le code détermine les nœuds locaux de la face qui touche à la zone et les enregistre dans belem2node.* | | | |
| **Entrées :**   * *nelem\_g* * *elem2zone* * *elem2node\_g* * *elem2elem\_g* * *elemglobal2local* * *nodeglobal2local* * *elem2vtk* | | **Sorties :**   * *elem2node = int [nzone][nelem][nnoel]* * *belem2node = vector int[nzone][…idz][nnofa+2]*   *(1ère colonne = vtk, 2e colonne = zone)*   * *zone2markelem = int[nzone][nzone-1]* * *zone2idmark = vector int[nzone][nzone-1][…]* | |
| Initialisation indexZone = [0,0,0,0], izone2jzone(voir exemple)  For *ielem* à nelem\_g  ielemLocal = elemglobal2local[*ielem*][0]  izone = elemglobal2local[*ielem*][1]  nnoel = function existante elem2vtk (nombre de noeud de l’élément)  nfael = function existante elem2vtk (nombre de face de l’élément)  for *inoel* à nnoel  inode\_g = elem2node\_g [*ielem*][*inoel*]  inode\_z = nodeglobal2local[inode\_g][izone]  elem2node[izone][ielemLocal][*inoel*] = inode\_z (elem2node de izone)  for *ifael* à nfael  jelem = elem2elem\_g[ielem][*ifael*] //*élément voisin*  if jelem <= nelem\_g //*si ce n’est pas un ghostcell*  jzone = elem2zone[jelem] // *de quelle zone s’agit-il*  jelemLocal = elemglobal2local[jelem][0]  if jzone != izone //*détection d’une frontière avec une autre zone*  idz = indexZone[izone]  nnofa = fonction de elem2vtk //*nb de nœud par face*  belem2node[izone][idz] = new int[nnofa +2]  belem2node[izone][idz][0] = vtk de la face  belem2node[izone][idz][end] = jelemLocal // num element local dans jzone  ijzone = izone2jzone[izone][jzone]  zone2markelem[izone][ijzone] += 1  zone2idmark[izone][ijzone].pushback(idz)  for *inofa* à nnofa //*Assignation de nœuds locaux*  inoel = fonction Getlpofa(ielem, *inofa*, ifael)  inode = elem2node\_g[ielem,inoel]  inodeLocal = nodeglobal2local[inode][izone]  belem2node[izone][idz][*inofa*+1] = inodeLocal  indexZone[izone] += 1 | | | |
| **#9 Écriture des fichiers** | | | |
| **Entrées :**   * *nzone, nelem\_g, ngcells\_g* * *zone2nelem* * *zone2ncells* * *zone2npoin* | * *zone2coord* * *nbound* * *ghost2node\_g* * *elem2node* | | * *belem2node* * *zone2markelem* * *zone2idmark* |
| **Sorties :**   * *nzone fichier SU2* | | | |
| - Ouverture des nzone fichiers  - initialisation de zone2jzone (voir exemple)  - initialisation de zone2ghost = vector<vector<vector<int>>>  For ighost à ngcells\_g  Btype = ghost2node\_g[ighost][1]  ielem = nelem\_g + ighost  jelem = elem2elem\_g[ielem][0]  jzone = elem2zone[jelem]  zone2ghost[jzone][Btype].pushback(jelem)  For *izone* à nzone  For inode à zone2npoin[*izone*]  Ligne = Coord[inode][0 à 2]  Écriture de Ligne //dans fichier izone  For ielem à zone2nelem[*izone*]  Ligne elem2node[ielem][0 à nnoel]  Écriture de Ligne //dans fichier izone  For ib à nbound  Écriture de MARKER\_TAG = ib //dans fichier izone  Écriture de MARKER\_ELEM = zone2ghost[izone][ib].size() //dans fichier izone  For ighost à zone2ghost[izone][ib].size()  jelem = zone2ghost[izone][ib][ighost]  nnoel = function elem2vtk (nb de nœuds pour l’élément)    for inoel à nnoel+1  écriture de ghost2node\_g[ighost][inoel+1] //dans fichier izone    For ijzone à nzone-1  jzone = zone2jzone[izone][ijzone]  Écriture de MARKER\_TAG = jzone  Écriture de MARKER\_ELEM = zone2markerElem[izone][ijzone]  For ibelem à zone2markelem[izone][ijzone]  idz = zone2idmark[izone][ijzone][ibelem]  Ligne = belem2node[izone][idz][0 à nnofa+2]  Écriture de Ligne //dans fichier izone | | | |

|  |  |
| --- | --- |
| **#10 Connectivité zone2zone** | |
| **Entrées :**   * *nzone* * zone2jzone * zone2markelem | **Sorties :**   * zone2zone *= int[nzone][nzone]* |
| Initialisation de matrice zone2zone = int[nzone][nzone] = -1  For *izone* à nzone  int istep = 0  For ijzone à nzone-1  jzone = zone2jzone[izone][ijzone]  nmark = zone2markelem[izone][jzone]  if nmark > 0  zone2zone[izone][istep] = jzone  istep += 1 | |

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Description** | **Sorties** | **Pseudo code** |
| 1 | Connectivité du maillage global jusqu’à elem2elem | *node2elem, elem2elem, etc.* | Non |
| 2 | Séparation en zones avec MÉTIS | *elem2zone* | Non |
| 3 | Génération d’éléments locaux aux zones | *zone2elem* | Oui |
| 4 | Génération des nœuds locaux au zones | *zone2node* | Oui |
| 5 | Connectivité entre éléments globaux et locaux | *elemglobal2local* | Oui |
| 6 | Connectivité entre nœuds globaux à locaux | *nodeglobal2local* | Oui |
| 7 | Connectivité entre les frontières | *À déterminer* | Oui |
| 8 | Écriture des fichiers Su2 pour toutes les zones | *nzone fichiers Su2* | Oui |
| 9 | Connectivité pour chaque zone avec code actuel | *À déterminer* | Oui |

***#1 Connectivité du maillage global jusqu’à elem2elem***

***#2 Séparation en zones avec MÉTIS 🡪 elem2zone***

***#3 Génération d’éléments locaux aux zones 🡪 zone2elem***

***#4 Génération des nœuds locaux au zones 🡪 zone2node***

***#5 Connectivité entre éléments globaux et locaux 🡪 elemglobal2local***

***#6 Connectivité entre nœuds globaux à locaux 🡪 nodeglobal2local***

***#7 Connectivité entre les frontières 🡪 ?***

***#7 Écriture des fichiers Su2 pour toutes les zones 🡪 fichiers Su2***

***#8 Connectivité pour chaque zone avec code actuel***