Manuel utilisateur PNM\_Matlab

Tristan Agaësse

Juin 2014

# Avant-Propos

Ce document est le manuel utilisateur de PNM\_Matlab, un code réseau de pores programmé sous Matlab en 2013 à l’IMFT par T. Agaesse.

Table des matières

[Avant-Propos 1](#_Toc392861554)

[Principales fonctionnalités 1](#_Toc392861555)

[Getting started 3](#_Toc392861556)

[Installation 3](#_Toc392861557)

[Premiers pas 3](#_Toc392861558)

[Tutoriels 4](#_Toc392861559)

[1) Prise en main 4](#_Toc392861560)

[2) Basic Physics 7](#_Toc392861561)

[Documentation détaillée 12](#_Toc392861562)

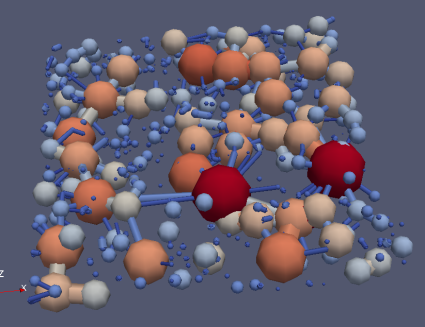
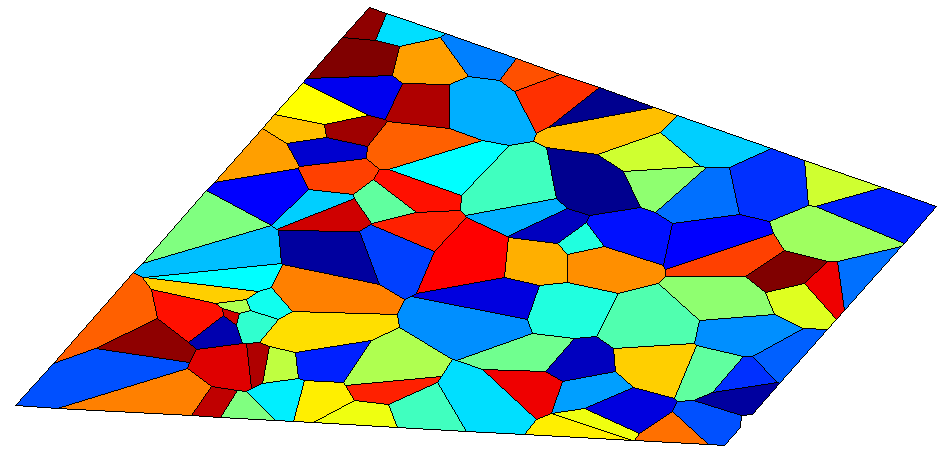
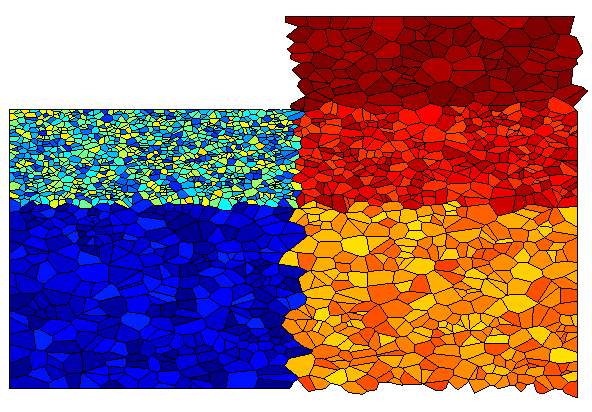
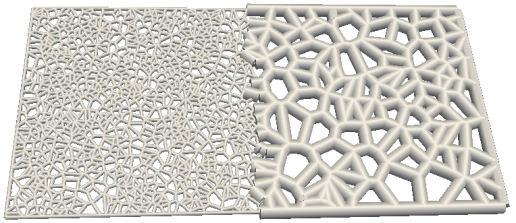
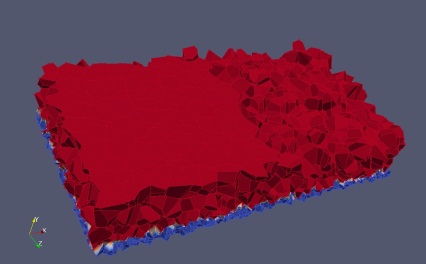
[Fichiers géométrie macroscopique 12](#_Toc392861563)

[Liste des fonctions 12](#_Toc392861564)

# Principales fonctionnalités

**Création de réseaux de pores morphologiques**

* Extrait d’une image 3D avec Avizo
* Généré à partir d’un maillage aléatoire : 2D, 3D, géométrie complexe, anisotropie

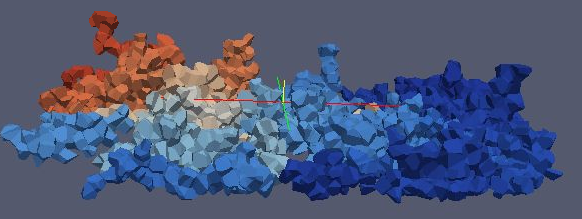
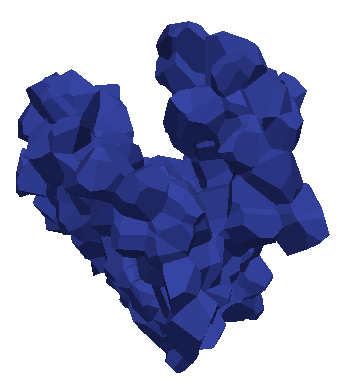
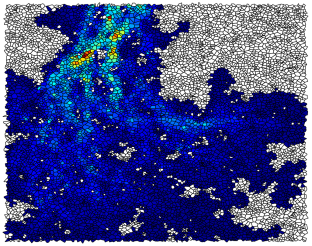


**Simulations physiques**

* Invasion percolation et transports de type laplacien (diffusion, perméabilité…)
* Les physiques mélangeant invasion diphasique et transports laplaciens sont facilement implémentables grâce aux fonctions de haut niveau disponibles dans le code.

**Outils de visualisation**

* Visualisation avec Matlab ou export vers Paraview.

****

**Programmation orientée objet**

* Des objets et des fonctions de haut niveau pour représenter les quantités physiques : le réseau (=le matériau), les clusters d’eau liquide…

**Utilisation conviviale**

* Auto-complétion pour retrouver facilement le nom des fonctions
* Aide interactive en ligne de commande
* Fonctions documentées et code commenté

**Compatibilité avec l’environnement Matlab**

* Utilisation aisée des fonctions Matlab : boites à outils mathématiques et graphiques, documentation exhaustive et importante communauté d’utilisateurs.
* intégration dans l’environnement de développement Matlab : accès graphique aux variables en mémoire, outils de débogage et de profilage du code, sauvegarde du contenu de la mémoire au format .mat…
* commande par scripts ou en ligne de commande

# Getting started

## Installation

Le dossier PoreNetworkMatlabCode contenant le code est composé de quatre sous-dossiers :

* Sources : le code proprement dit
* Simulations : dossier où sont stockés les résultats de simulation
* ExternalLibraries : librairies externes utilisées par le code
* Documentation : manuel utilisateur et tutoriels.

Pour utiliser le PNM\_Matlab il faut ouvrir Matlab, se mettre (fenêtre CurrentFolder dans l’interface de Matlab) au niveau du dossier contenant PNM\_Matlab puis cliquer avec le bouton droit sur le dossier PNM\_Matlab et sélectionner ‘Add To Path / Selected Folder and Sub-folders’.

## Premiers pas

Interface Matlab (double cliquer sur objets dans workspace)

Ligne de commande

network=CreateNework(‘exemple2D’)

Objets en mémoire : network et visualisateur.

Taper la commande : class(network);

On voit que c’est un objet de la classe PoreNetworkMeshFibrous.

doc PoreNetworkMeshFibrous

La documentation nous indique que cette classe est un cas particulier (sous-classe) de la classe générale PoreNetwork. Ce réseau de pore-ci est défini à partir d’un maillage (d’où le Mesh), et la position des parties solides est défini implicitement par des fibres situées sur les arêtes du maillage (d’où le Fibrous).

La documentation nous indique aussi que cette classe a des méthodes et des propriétés. Les propriétés des objets de PNM\_Matlab doivent être considérées comme privées, c’est-à-dire qu’elles sont gérées par la classe elle-même et qu’on ne doit pas s’en préoccuper. Les propriétés sont une affaire d’implémentation interne au code, l’utilisateur n’a pas besoin de s’en préoccuper.

Ce qui nous intéresse plutôt ce sont les méthodes des objets, c’est-à-dire des fonctionnalités qui leur sont associées. Ce sont-elles qui vont nous permettre d’interagir avec eux pour construire des simulations. La documentation permet de cliquer sur le nom des méthodes pour avoir plus d’informations. Il est possible aussi de taper methods(network) pour voir une liste des méthodes de l’objet network. On peut également utiliser l’auto complétion pour voir les méthodes et propriétés de l’objet : taper network. puis taper sur tab, un menu déroulant s’affiche alors.

Pour visualiser le réseau, on va utiliser une méthode du viewer. Tapez help viewer.View . Cela donne accès à l’aide de la méthode View du viewer. Ici on voit que la méthode View peut prendre plusieurs options. Tapez visualisateur.View(‘Network’) pour voir le réseau (avec une couleur différente pour chaque cellule). visualisateur.View(‘Boundaries’) affiche les liens situés sur les frontières, qui vont être utiles pour définir des conditions limites.

# Tutoriels

Les tutoriels présentés dans cette partie sont disponibles sous forme de scripts Matlab dans le dossier PNM\_Matlab/Documentation. On peut les exécuter dans Matlab. Cette partie a été réalisée avec l’outil publish de Matlab.

**Pour avoir une version à jour des tutorials**, le mieux est d’aller les chercher dans le dossier tutorial et de les exécuter avec clic droit, run.

## 1) Prise en main

%Pour creer un reseau de pores il faut avoir un fichier geometrie

%macroscopique qui decrit la forme des materiaux poreux.

%Nous allons utiliser la fonction createNetwork pour construire un reseau

%de pores ainsi que son outil de visualisation a partir du fichier de

%geometrie macroscopique '1block2D';

[ network,viewer ]=CreateNetwork('1block2D');

Generation du reseau...

Reseau genere. Duree : 0 minutes 9.001929 s.

%La fonction CreateNetwork a cree en memoire deux objets, network et

%viewer. Nous allons voir que ces objets appartiennent a des classes.

class(network)

class(viewer)

ans =

PoreNetworkMeshFibrous

ans =

Viewer

On voit que le reseau est un objet de la classe PoreNetworkMeshFibrous. Ouvrons la documentation de cette classe a l'aide de la fonction 'doc'.

doc PoreNetworkMeshFibrous

%

% La documentation nous indique que cette classe est un cas particulier

% (sous-classe) de la classe générale PoreNetwork. Ce réseau de pore-ci est

% défini à partir d’un maillage (d’où le Mesh), et la position des parties

% solides est défini implicitement par des fibres situées sur les arêtes du

% maillage (d’où le Fibrous).

%

% La documentation nous indique aussi que cette classe a des méthodes

% et des propriétés. Les propriétés des objets de PNM\_Matlab doivent être

% considérées comme privées, c’est-à-dire qu’elles sont gérées par la classe

% elle-même et qu’on ne doit pas s’en préoccuper. Les propriétés sont une

% affaire d’implémentation interne au code, l’utilisateur n’a pas besoin de

% s’en préoccuper.

%

% Ce qui nous intéresse plutôt ce sont les méthodes des objets,

% c’est-à-dire des fonctionnalités qui leur sont associées. Ce sont-elles qui

% vont nous permettre d’interagir avec eux pour construire des simulations.

% La documentation permet de cliquer sur le nom des méthodes pour avoir plus

% d’informations.

%Pour voir une liste des méthodes de l’objet network il est possible aussi de taper

methods(network)

Methods for class PoreNetworkMeshFibrous:

AddNewEdgeData GetNumberOfEdges

AddNewLinkData GetNumberOfLinks

AddNewPoreData GetNumberOfPores

AddNewVerticeData GetNumberOfVertices

ComputeAllLinkDiameter GetPoreCenter

ComputeAllPoreVolume GetPoreDataList

ComputeExtrudePolygonParFibre GetPoreDiameter

ComputeLinkDiameter GetPoresFrontiere

ComputePoreVolume GetPoresOfLink

CreateVoidCluster GetPoresVoisinsOfPore

ExportToBabe GetVertice

ExportToParaview GetVerticeDataList

FindComposantesConnexes GetVerticesOfEdge

GetAllVerticesCoordinates GetVerticesOfEdgeNumber

GetCommonFace GetVerticesOfLink

GetDimension GetVerticesOfLinkNumber

GetEdgeDataList GetVerticesOfPore

GetEdgesOfLink GetVerticesOfPoreNumber

GetEdgesOfPore PoreNetworkMeshFibrous

GetEdgesOfVertice PrivateBabeOutputStruct

GetFrontiereOfLink PrivateInternalOutputStruct

GetLinkCenter PrivateVTKOutputStructBallAndStick

GetLinkDataList PrivateVTKOutputStructPolydataMesh

GetLinkDiameter RemoveEdgeData

GetLinksFrontiere RemoveLinkData

GetLinksOfEdge RemovePoreData

GetLinksOfPore RemoveVerticeData

GetNeighbourEdgesOfVerticeInPore

GetNeighbourLinksOfEdgeInPore

%On peut également utiliser

% l’auto complétion pour voir les méthodes et propriétés de l’objet : taper

% network. puis taper sur tab, un menu déroulant s’affiche alors.

%Pour visualiser le réseau, on va utiliser la méthode View du viewer.

help viewer.View

Input : (option, varargin)

'Network': affichage du maillage

'PoreList': affiche une liste de pores : liste\_cellules=varargin{1}

'PoreField' : affichage d'un champ scalaire sur les cellules, champ=varargin{1}

'LinkField' : affichage d'un champ scalaire sur les liens, champ=varargin{1}

'Boundaries': affichage des faces du maillage, avec code couleur pour les fronti�res

'Edges' : affichage des ar�tes du maillage, avec code couleur pour les fronti�res

'Sommets' : affichage des fibres avec des disques

%On voit que la méthode View peut prendre plusieurs options. Pour voir le

%réseau (avec une couleur différente pour chaque cellule), tapez

viewer.View('Network')

%Pour afficher les liens situés sur les frontières, qui vont être utiles

%pour définir des conditions limites.

figure; %cree une nouvelle figure

viewer.View('Boundaries')

## 2) Basic Physics

%Dans ce tutoriel on va calculer une invasion percolation et de la diffusion.

%Les algorithmes pour simuler une physique ont un nom commençant par Compute. Pour retrouver un algorithme il est pratique d’utiliser l’autocomplétion de Matlab : taper Compute puis taper sur tab pour avoir accès à la liste des fonctions dont le nom commence par compute.

[ network,viewer ]=CreateNetwork('3block2D');

viewer.View('Network')

figure;colorbar;

viewer.View('Boundaries')

Generation du reseau...

Reseau genere. Duree : 0 minutes 11.403237 s.

%La fonction ComputeInvasionPercolation prend en arguement le reseau, une

%liste de liens d'entree pour l'eau (inletLink), une liste de liens de

%sortie (outletLink) et une option concernant les angles de contact.

%Ici nous choisissons de faire rentrer l'eau par la frontiere du bas et la

%faire sortir par la frontiere du haut. Les numeros de ces frontieres sont

%accessibles soit dans le fichier de geometrie macroscopique, soit avec la

%fonction viewer.View('Boundaries'). Les liens correspondants à ces

%frontieres sont donnes par la methode GetLinksFrontiere du reseau.

help ComputeInvasionPercolation

inletLink=network.GetLinksFrontiere(2);

outletLink=network.GetLinksFrontiere([4,5,6]);

[cluster,breakthroughPressure,invasionPressureList]=ComputeInvasionPercolation(network,inletLink,outletLink,'hydrophobic');

class(cluster)

ComputeInvasionPercolation Calcule l'invasion de percolation sur un r�seau

de pores, avec un unique cluster

Input : network,inletLink,outletLink,wettability , ( varargin ) :

- network

- inletLink : liste des liens d'injection

- outletLink : liste des liens de percée possible

- wettability : 'currentWettability', 'hydrophobic', 'hydrophilic' or 'random'.

- varargin (optionnel) : clusterOptions (voir ClusterMonophasique)

Output : [cluster,breakthroughPressure,invasionPressureList]

Calcul du diam�tre des liens...

Calcul du diam�tre des liens termin�. Dur�e : 0 minutes 1.276982 s.

Calcul d'invasion percolation...

Calcul d'invasion percolation termin�. Dur�e : 0 minutes 2.574683 s.

ans =

ClusterMonophasique

%ComputeInvasionPercolation renvoit un objet de type ClusterMonophasique.

%Ce type d'objet decrit la repartition d'un phase dans le reseau. La classe

%ClusterMonophasique est celle qui gere les deplacements des frontieres

%entre phases. Pour avoir la liste des pores envahis, utiliser la methode

%GetInvadedPores du cluster.

viewer.View('PoreList',cluster.GetInvadedPores)

%Nous allons maintenant faire un calcul de diffusion. En utilisant

%l'autocompletion a partir du mot Compute, on trouve la fonction

%ComputeDiffusion. help ou doc ComputeDiffusion nous donne la liste des parametres

%d'entree et de sortie de cette fonction.

%

%Nous allons calculer la diffusion dans les pores qui n'ont pas ete envahis

%lors de l'invasion percolation. Il est possible d'indiquer la region dans

%laquelle a lieu la diffusion avec un objet cluster, qui contient une

%liste de pores envahis. Ici le cluster a donner a ComputeDiffusion est le

%complementaire du cluster obtenu par invasion percolation.

complementaryCluster=cluster.GetComplementaryCluster;

newInletLink=network.GetLinksFrontiere([4,5,6]);

newOutletLink=network.GetLinksFrontiere([1,2,3]);

[ concentrations, debits, fluxSurfaciques, diffusionCoefficient ]=ComputeDiffusion(network,complementaryCluster, newInletLink, newOutletLink);

diffusionCoefficient

figure

viewer.View('PoreField',concentrations)

Calcul des conductances de Diffusion...

Calcul des conductances de diffusion termin�. Dur�e : 0 minutes 1.133461 s.

Elapsed time is 0.087034 seconds.

diffusionCoefficient =

6.9366e-10

%Calculons maintenant la permeabilite du reseau de pores. Nous allons

%utiliser la fonction ComputePermeabilite dans une region correspondant au

%reseau tout entier. Pour cela il nous faut un cluster qui recouvre tout le

%reseau. Le moyen le plus simple de l'obtenir utilise les fonctions

%CreateVoidCluster du reseau et GetComplementaryCluster du cluster.

voidCluster=network.CreateVoidCluster;

fullCluster=voidCluster.GetComplementaryCluster;

[ pressions, debits, vitessesMoyennes, permeabilityCoefficient ]=ComputePermeability(network, fullCluster, newOutletLink, newInletLink);

permeabilityCoefficient

figure;

viewer.View('PoreField',pressions)

Calcul des conductances Stokes...

Calcul des conductances Stokes termin�. Dur�e : 0 minutes 1.242516 s.

permeabilityCoefficient =

3.2821e-13

## 3) Tutoriel Wettability and ClusterShape

%Dans ce tutoriel nous allons voir comment faire une invasion percolation

%en définissant nous même les angles de contact. Nous allons apprendre à

%gérer les poreData et les linkData qui contiennent les propriétés des

%pores et des liens. Enfin nous allons exporter le réseau et les résultats

%de simulation vers Paraview pour visualiser les résultats.

%Commençons par créer un réseau 3D.

[ network,viewer ]=CreateNetwork('1block3D');

inletLink=network.GetLinksFrontiere(1);outletLink=network.GetLinksFrontiere(2);

G�n�ration du r�seau...

R�seau g�n�r�. Dur�e : 0 minutes 43.703342 s.

%L'objet network contient des data associées aux pores et liens. Ce peut

%etre les propriétées géométriques des pores et liens, des résultats de

%simulations...

%On peut accéder aux poresData et linkData directement dans l'interface

%graphique de Matlab en double cliquant sur l'objet network. Il est aussi

%possible de les manipuler en ligne de commande.

%

%Ces données peuvent servir dans les algorithmes physiques

%(conductances, tailles...). C'est le cas pour l'invasion percolation qui

%utilise les diametres des liens et les angles de contact dans les liens.

%Pour faire une invasion percolation avec des angles de contact

%personnalises, il faut ajouter a la liste des link data un tableau

%contenant les angles de contact. On utilise la fonction AddNewLinkData

%avec le nom 'ContactAngle'.

contactAngle=80;

theta=contactAngle\*pi/180\*ones(1,network.GetNumberOfLinks);

network.AddNewLinkData(theta,'ContactAngle');

network.GetLinkDataList

network.GetLinkDataList.ContactAngle;

ans =

ContactAngle: [1x9975 double]

%Calculons donc une invasion percolation avec l'angle de contact de 80°

%defini a l'etape precedente. Pour visualiser les pores envahis dans

%Paraview ultérieurement, il faut rajouter une liste de pores envahis aux

%poreData du réseau. On voit aussi que la fonction

%ComputeInvasionPercolation a rajoute la liste des diametres de liens qui

%lui manquait.

[cluster,breakthroughPressure,invasionPressureList]=ComputeInvasionPercolation(network,inletLink,outletLink,'currentWettability');

network.AddNewPoreData(cluster.GetInvadedPoresBooleans,'InvadedPores\_80');

network.GetPoreDataList

network.GetLinkDataList

Calcul du diam�tre des liens...

Calcul du diam�tre des liens termin�. Dur�e : 1 minutes 55.974487 s.

Calcul d'invasion percolation...

Calcul d'invasion percolation termin�. Dur�e : 0 minutes 18.020365 s.

ans =

InvadedPores\_80: [1x1096 double]

ans =

ContactAngle: [1x9975 double]

Diameter: [1x9975 double]

%Faisons maintenant une invasion percolation avec un angle de contact

%uniforme de 110°. On retire d'abord l'ancien linkData 'ContactAngle', on

%ajoute le nouveau puis .

contactAngle=110;

network.RemoveLinkData('ContactAngle');

theta=contactAngle\*pi/180\*ones(1,network.GetNumberOfLinks);

network.AddNewLinkData(theta,'ContactAngle');

[cluster,breakthroughPressure,invasionPressureList]=ComputeInvasionPercolation(network,inletLink,outletLink,'currentWettability');

network.AddNewPoreData(cluster.GetInvadedPoresBooleans,'InvadedPores\_110');

network.GetPoreDataList

network.GetLinkDataList

Calcul d'invasion percolation...

Calcul d'invasion percolation termin�. Dur�e : 0 minutes 8.248737 s.

ans =

InvadedPores\_80: [1x1096 double]

InvadedPores\_110: [1x1096 double]

ans =

Diameter: [1x9975 double]

ContactAngle: [1x9975 double]

%Calculons maintenant la diffusion. Pour visualiser les concentrations dans

%Paraview ultérieurement, il faut rajouter les concentrations aux data du

%réseau.

concentrations=ComputeDiffusion(network, cluster, inletLink, outletLink);

network.AddNewPoreData(concentrations,'DiffusionConcentrations');

Calcul des conductances de Diffusion...

Calcul des conductances de diffusion termin�. Dur�e : 0 minutes 1.714278 s.

Elapsed time is 0.023504 seconds.

%La fonction ComputeDiffusion utilise les conductances de diffusion des

%liens. Si 'ConductancesDiffusion' n'est pas encore defini dans les

%linkData du reseau, la fonction la calcule et l'ajoute au reseau pour une

%future utilisation.

network.GetLinkDataList

ans =

Diameter: [1x9975 double]

ContactAngle: [1x9975 double]

ConductancesDiffusion: [1x9975 double]

%On peut visualiser le reseau dans Matlab avec le viewer. Neanmoins les

%fonctionnalites du viewer sont moins etendues en 3D qu'en 2D et le rendu est moins bon.

%Cela est du aux limitations des outils de visualisation de Matlab.

viewer.View('Network')

%Il est possible d'exporter le reseau dans Paraview pour une meilleure visualisation. Pour cela on ecrit un

%fichier .vtk. Dans ce fichier il y aura des informations sur la geometrie

%du reseau et les data associes aux pores, liens etc... Il y a deux types

%de rendu d'un reseau, Ball and Stick et maillage. Le rendu maillage est

%disponible seulement pour les reseaux des classes PoreNetworkMesh et

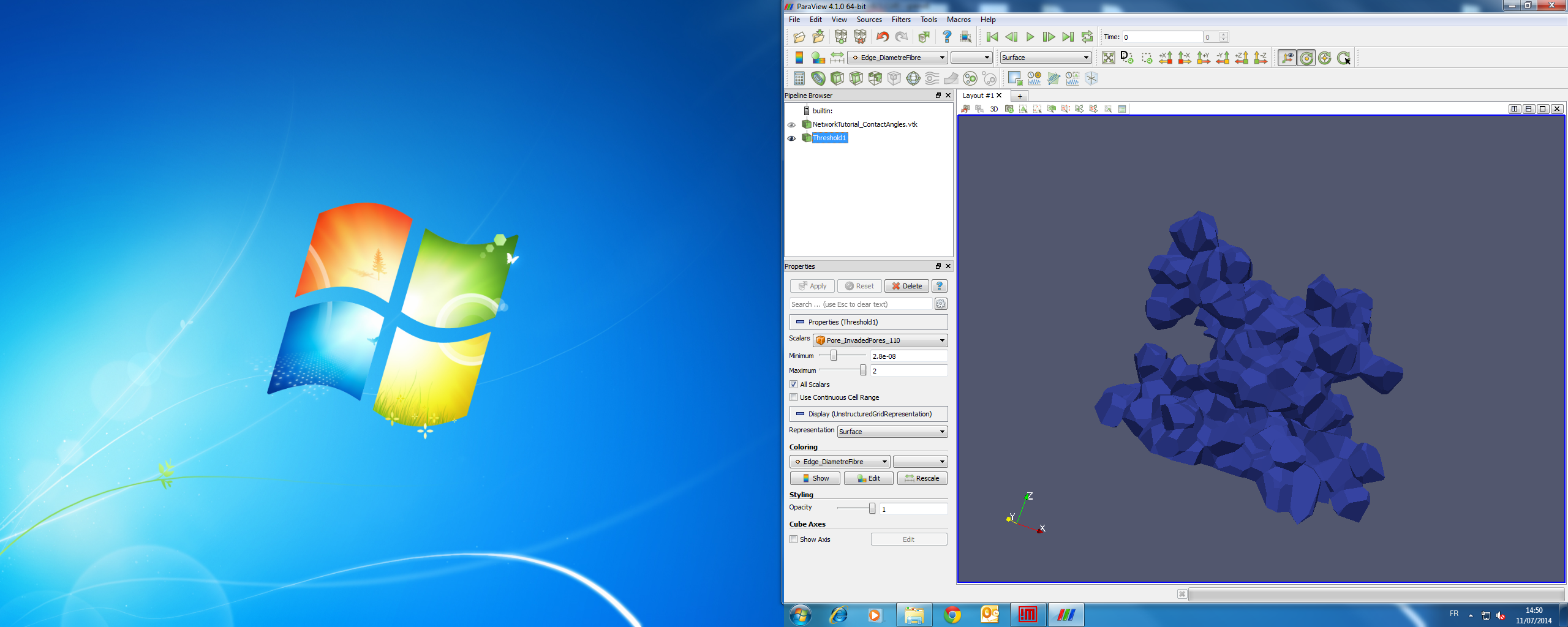
%classes derivees.

network.ExportToParaview('NetworkTutorial\_Wettability')

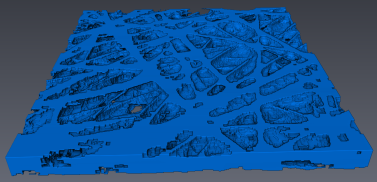
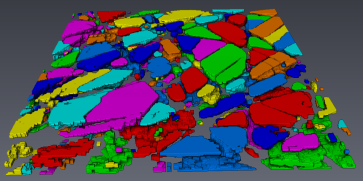
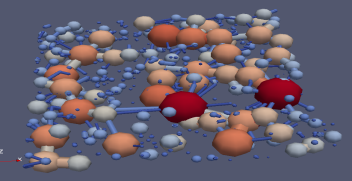
G�n�ration du fichier VTK...

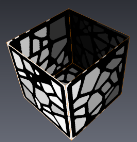
Fichier VTK g�n�r�. Dur�e : 0 minutes 43.368388 s.

Une fois le fichier VTK chargé dans Paraview, on peut visualiser le réseau et les données associées. Pour voir les pores envahis on peut utiliser le filtre threshold pour visualiser seulement les pores qui ont une donnée InvadedPore égale à 1.



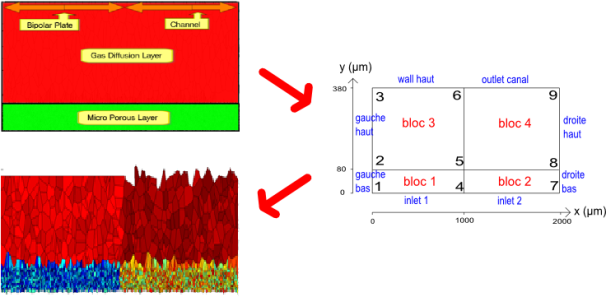
## 4) Utilisation d’un réseau de pores extrait d’une image 3D



# Documentation détaillée

## Fichiers géométrie macroscopique



Cet outil est inspiré du générateur de géométrie BlockMesh de openFoam. http://www.openfoam.org/docs/user/blockMesh.php

## Liste des fonctions

La documentation détaillées des fonctions est accessible directement dans Matlab en tapant doc +nom de la fonction ou de la classe, par exemple doc PoreNetworkMeshFibrous. On peut naviguer dans cette documentation d’une fonction à l’autre.

## Implémentation interne

La façon dont est codé le réseau (graph) est inspiré du codage d’un maillage dans les fichiers openFoam. De cette façon on a une implémentation efficace et économe en mémoire.

http://www.openfoam.org/docs/user/mesh-description.php