Manipulation et visualisation des fibres MRtrix par Matlab (ou Octave)

Pierre Besson

Formation “IRM de diffusion cérébrale”
TP du 01/12/2016
Marseille

Introduction

Au cours de ce TP, vous allez :

- Importer un jeu de fibres .tck
- Sauvegarder des fibres en .tck ou les exporter en .vtk
- Visualiser des fibres via le logiciel Paraview
- Mesurer et visualiser une donnée d'intérêt sur les fibres
- Sélectionner un faisceau d'intérêt
- Délimiter une portion de faisceau d'intérêt

L'objectif de ce TP est de montrer que les fibres MRtrix peuvent être manipulées facilement avec seulement quelques fonctions Matlab / Octave. Inutile d'être expert en Matlab, même si les utilisateurs avancés pourront bien entendu aller plus loin et lire le contenu des fonctions.

Tout au long de ce TP, une police différente sera utilisée pour indiquer des lignes de commande à entrer dans le terminal ou dans Octave.

```
$ Ceci est une ligne de commande du terminal
```

Lorsque la ligne commence par le symbole $, il s'agit d'une commande à entrer dans le terminal.

```
>> Ceci est une commande Octave
```

Pour une commande Octave, la ligne commence par >>

Dans Octave, vous pouvez afficher l'aide d'une fonction avec la commande :

```
>> help nom_de_fonction.m
```

Une petite description indiquera comment utiliser la fonction.

Préparation des données

- Le jeu de données est issu du Human Connectome Project (HCP), disponible librement sur ce site

  https://db.humanconnectome.org/ (inscription requise).

- Sujet 113922 sélectionné au hasard (homme, 31-35 ans)
- IRM de diffusion: résolution isotrope $1.25 \times 1.25 \times 1.25$ mm^3, 288 volumes: 6 volumes à b = 0 s/mm2, environ 90 directions pour 3 valeurs de b (1000, 2000, 3000 s/mm^2), LR/RL.
- Téléchargement des données de diffusion prétraitées (corrigées pour les distorsions et alignées avec l'image T1w)
- Tractographie avec MRtrix3 en suivant ce tutoriel (multi-shell, multi-tissue)

  http://mrtrix.readthedocs.io/en/latest/tutorials/hcp_connectome.html

- Utilisation de l'ACT, dynamic seed

```
$ tckgen WM_FODs.mif 10000.tck -act 5TT.mif -backtrack -crop_at_gmwmi -seed_dynamic WM_FODs.mif
-maxlength 250 -number 10000 -cutoff 0.06
```

- Seulement 10 000 fibres générées pour garder les traitements rapides et légers, mais dans la pratique utiliser beaucoup plus de fibres (bien supérieur à 1 million). Nécessite d'avoir un ordinateur rapide et bien équipé en RAM.

Importer et exporter un jeu de fibres

Ouvrir un terminal, se placer dans le répertoire d'intérêt

```
$ cd /chemin/mon/repertoire
```

Lancer Octave

```
$ octave
```

ou

```
$ qtoctave
```

pour avoir une interface graphique.

Pour importer les fibres, utiliser la commande

```
>> Fib = load_tck('10000.tck');
```

La variable Fib est créée et stocke les informations relatives aux fibres du fichier 10000.tck. Pour voir le contenu de la variable Fib, entrer la commande :

```
>> Fib
```

Octave affiche alors le contenu de la variable (mes annotations en jaune):

```
Fib =
  scalar structure containing the fields:  Fib est une structure
    fiber =                                fiber est un champ de la structure
      1x10000 struct array containing the fields: taille du champ = nombre de fibres
        coord                   tableau de coordonnées des fibres
        nFiberLength            nombre de points des fibres
        rgbFiberColor           couleur de la fibre
        rgbPointColor           couleur des points de la fibre
        length                  longueur de la fibre (mm)
        cumlength               longueur le long de la fibre (mm)
    cumlength =
      scalar structure containing the fields:
        type = point            indique que cumlength est une donnée par point
    length =
      scalar structure containing the fields:
        type = cell             indique que length est une donnée par fibre
```

Toutes les informations de la première fibre peuvent être visualisées en entrant la commande:

```
>> Fib.fiber(1)
```

Octave affiche en entier l'ensemble des attributs de la première fibre, beaucoup de lignes puisque la fibre est constituée de 201 points (Fib.fiber(1).nFiberLength = 201).

Pour sauvegarder les fibres au format d'origine de MRtrix (.tck), utiliser cette commande:

```
>> save_tck(Fib, 'copy_10000.tck')
```

Le fichier copy_10000.tck est généré et contient un jeu de fibres strictement identique à 10000.tck

Pour exporter les fibres au format .vtk, utiliser la commande:

```
>> export_tck_to_vtk(Fib, '10000.vtk')
```

Le fichier 10000.vtk est visualisable grâce au logiciel Paraview. Pour le lancer, entrer dans le terminal :

```
$ paraview
```

Le logiciel Paraview se lance, cliquez sur Open et sélectionnez le fichier 10000.vtk

Paraview est très pratique pour de nombreuses raisons car il permet de visualiser simultanément: volumes, surfaces, fibres, graphes, décours temporels, champs vectoriels, etc… Il permet aussi de visualiser les données dans des plans arbitraires. Mais surtout, Paraview permet de visualiser des fibres avec une information mappée dessus. Par exemple, la longueur cumulative des fibres :

```
>> export_tck_to_vtk(Fib, '10000_cumlength.vtk', 'cumlength');
```

Ouvrez le fichier 10000_cumlength.vtk dans Paraview.

Mesurer et visualiser un signal le long des fibres

La fonction sample_tck.m permet d'interpoler un volume à chacun des points de toutes les fibres. Elle a besoin d'une structure de fibres et un volume dans un format compatible MRtrix.

```
>> Fib = sample_tck(Fib, 'dti_FA.nii', 'FA');
```

Le premier argument est une structure de fibres, le second une image, le troisième le nom du signal mesuré (choisi par l'utilisateur). La variable Fib est utilisée à la fois en entrée et en sortie. La nouvelle variable Fib contient de nouveau champs:

```
>> Fib
Fib =
  scalar structure containing the fields:
    fiber =
      1x10000 struct array containing the fields:
        coord
        nFiberLength
        rgbFiberColor
        rgbPointColor
        length
        cumlength
        FA                FA mesurée à chaque point
        FA_mean           FA moyenne le long de chaque fibre
        FA_max            FA maximum le long de chaque fibre
    cumlength =
      scalar structure containing the fields:
        type = point
    length =
      scalar structure containing the fields:
        type = cell
    FA =
      scalar structure containing the fields:
        type = point    Indique que FA est une mesure par point
    FA_mean =
      scalar structure containing the fields:
        type = cell     Indique que FA_mean est une mesure par fibre
    FA_max =
      scalar structure containing the fields:
        type = cell     Indique que FA_max est une mesure par fibre
```

On peut ajouter les valeurs de MD à la même structure :

```
>> Fib = sample_tck(Fib, 'dti_MD.nii', 'MD');
```

Vous pouvez vérifier que les champs MD, MD_mean et MD_max sont ajoutés à la structure Fib.

Une conversion au format .vtk vous permet de visualiser les données désirées, par exemple FA, FA_mean, MD, MD_mean :

```
export_tck_to_vtk(Fib, '10000_fa_md.vtk', {'FA'; 'FA_mean'; 'MD'; 'MD_mean'});
```

Paraview permet de visualiser chacun des champs indiqués.

Extraire un faisceau d'intérêt

L'extraction de jolis faisceaux nécessite souvent plusieurs ROIs bien placées.

En utilisant mrview, nous pouvons tracer une première région d'intérêt en choisissant une région très riche en fibres (par exemple le corps calleux ou un faisceau cortico-spinal). Sauvegardez le mask en le nommant roi1.mif

Dans Octave, échantillonnez la valeur de la ROI le long des fibres avec la commande:

```
>> Fib = sample_tck(Fib, 'roi1.mif', 'roi1');
```

Les champs roi1, roi1_mean et roi1_max sont ajoutés à Fib.

Le critère d'exclusion des fibres peut être défini par la valeur de roi1_max :

si Fib.fiber(i).roi1_max > seuil, on garde la fibre

si Fib.fiber(i).roi1_max <= seuil, on supprime la fibre.

Dans Octave, le filtrage des fibres s'effectue en quelques lignes :

```
>> Fib_roi1 = Fib;           Copie la structure Fib dans Fin_roi1

>> excluded_fiber = cat(1, Fib_roi1.fiber.roi1_max); crée un vecteur regroupant roi1_max de toutes les
                                                     fibres

>> excluded_fiber = excluded_fiber <= 0;  Le seuil est placé à 0

>> Fib_roi1.fiber(excluded_fiber) = [];   Supprime les fibres
```

La structure Fib_roi1 contient désormais un nombre de fibres différent de 10000, et toutes devraient passer par la ROI. Nous pouvons sauvegarder les fibres au format .tck et le vérifier dans mrview

```
>> save_tck(Fib_roi1, 'fibres_roi1.tck');
```

En général, une ROI ne suffit pas à obtenir un faisceau bien défini, il faut placer d'autres ROI et éliminer les fibres de non-intérêt.

Extraction d’une portion d’un faisceau

Une fois qu’un joli faisceau est extrait, nous pouvons en extraire une portion d’intérêt afin de placer une origine commune à toutes les fibres et à mesurer quelque chose le long des fibres.

Sauvegarder le faisceau au format .vtk, l’ouvrir avec Paraview et placer un plan de coupe des fibres. Noter les coordonnées de l’origine du plan [o_x o_y o_z] et du vecteur normal au plan [n_x n_y n_z].

Dans Octave, utiliser la fonction crop_fibers_plan.m pour couper les fibres par le plan :

```
>> Fib_crop = crop_fibers_plan(Fib, [o_x o_y o_z], [n_x n_y n_z]);
```

Assurez-vous que le faisceau ait été bien coupé et l’origine des fibres placé au bon endroit

```
>> export_tck_to_vtk(Fib_crop, ‘fibres_crop.vtk’, ‘cumlength’);
```

La coupe de ce nouveau faisceau par un second plan assure l’extraction d’une portion bien définie.

```
>> Portion_finale = crop_fibers_plan(Fib_crop, [o_x2 o_y2 o_z2], [n_x2 n_y2 n_z2]);
```

S’assurer à nouveau que la portion finale soit jolie.

Si tout est bon, échantillonner un volume d’intérêt sur la portion finale puis tracer tous les profiles. Par exemple en utilisant la FA :

```
>> Portion_finale = sample_tck(Portion_finale, ‘dti_FA.nii’, ‘FA’);
>> for ii = 1 : length(Portion_finale.fiber)
hold on, plot(Portion_finale.fiber(ii).cumlength, Portion_finale.fiber(ii).FA)
end
```