Notas para crear un pipeline de fMRI

Conversión a NIfTI-1

Que bonito, hay dos tipos de formato diferentes. A ver como hacemos.

ep2d_pace_moco_p2

Este es facil y se convierte con *dcm2nii*. El DICOM consiste en una serie de scaners completos (150). Básicamente lo que hecho es localizar los fMRI de este tipo y escribirlos en un archivo.

```
for x in `find ./ -name Img00001.dcm`; do if [[ `dckey -k
"SeriesDescription" "$x" 2>&1 | grep ep2d_pace_moco_p2` ]]; then echo $x |
awk -F"/" {'print $2,$3'} | sed 's/F/0/;s/Serie(//;s/)//'; fi; done | sort >
~/data/facehbi/ep2d_pace_moco_p2.loc
```

y despues un script en perl para automatizar la conversión

moco2nii

dcm_moco_2nii.pl

```
#!/usr/bin/perl

# Copyright 2015 O. Sotolongo <osotolongo@fundacioace.com>

use strict; use warnings;
use File::Slurp qw(read_file);
use Data::Dump qw(dump);
use NEURO qw(populate check_subj load_study print_help get_petdata
get_pair achtung shit_done);

my $study;
@ARGV = ("-h") unless @ARGV;
while (@ARGV and $ARGV[0] =~ /^-/) {
    $_ = shift;
    last if /^--$/;
    if (/^-e/) { $study = shift; chomp($study);}
    if (/^-h/) { print_help $ENV{'PIPEDIR'}.'/doc/dcm_moco.hlp'; exit;}
}

unless ($study) { print_help $ENV{'PIPEDIR'}.'/doc/dcm_moco.hlp';
exit;}

my %std = load_study($study);
```

```
my $data_dir=$std{'DATA'};
my $ifile = 'ep2d_pace_moco_p2.loc';
my $dest_dir = 'fmri_conv';
my %locs = map { my ( $key, $value ) = split " "; $key => $value}
read_file $data_dir.'/'.$ifile;

foreach my $subject (sort keys %locs) {
    my ($short) = $subject =~ /0(.*)/;
    #print "$short\n";
    my $orig =
'"/nas/corachan/facehbi/F'.$short.'/Serie('.$locs{$subject}.')/Img00001
.dcm"';
    #print "$orig\n";
    my $order = 'mkdir '.$data_dir.'/'.$dest_dir.'/f'.$short.' &&
dcm2nii -o '.$data_dir.'/'.$dest_dir.'/f'.$short.' '.$orig;
    print "$order\n";
    system($order);
}
```

que se ejecuta con

```
dcm_moco_2nii.pl -e facehbi
```

y otro script para organizarlo, por aquello de la vagancia de cada cual

moco organizer ;-)

moco_org.pl

```
#!/usr/bin/perl

# Copyright 2015 O. Sotolongo <osotolongo@fundacioace.com>

use strict; use warnings;
use File::Slurp qw(read_file);
use Data::Dump qw(dump);
use NEURO qw(populate check_subj load_study print_help get_petdata
get_pair achtung shit_done);

my $study;
@ARGV = ("-h") unless @ARGV;
while (@ARGV and $ARGV[0] =~ /^-/) {
    $_ = shift;
    last if /^--$/;
    if (/^-e/) { $study = shift; chomp($study);}
    if (/^-h/) { print_help $ENV{'PIPEDIR'}.'/doc/dcm_moco.hlp'; exit;}
}
```

```
unless ($study) { print_help $ENV{'PIPEDIR'}.'/doc/dcm_moco.hlp';
exit;}

my %std = load_study($study);

my $data_dir=$std{'DATA'};
my $ifile = 'ep2d_pace_moco_p2.loc';
my $orig_dir = 'fmri_conv';
my $dest_dir = 'fmri';
my %locs = map { my ( $key, $value ) = split " "; $key => $value}
read_file $data_dir.'/'.$ifile;

foreach my $subject (sort keys %locs) {
    my ($short) = $subject =~ /0(.*)/;
    #my $puaf = sprintf("%04d", $locs{$subject});
    my $order = 'cp '.$data_dir.'/'.$orig_dir.'/f'.$short.'/*.nii.gz
'.$data_dir.'/'.$dest_dir.'/smc'.$subject.'s'.sprintf("%04d",
$locs{$subject}).'.nii.gz';
    print "$order\n";
    system($order);
}
```

ep2d_fid_bold_p2

Por lo pronto la mejor manera de convertir de DICOM a *nifti* parece ser utilizando **spm**. Básicamente abrimos *matlab*, llamamos el spm y hacemos ahí la conversión. Pero siguiendo al viejo Jack (*The Ripper*), vayamos por partes.

Los pasos a seguir aqui son especificos para el proyecto **FACEHBI** pero aplicables a cualquier otro. Primero hay que encontrar donde estan las imágenes fMRI de cada sujeto.

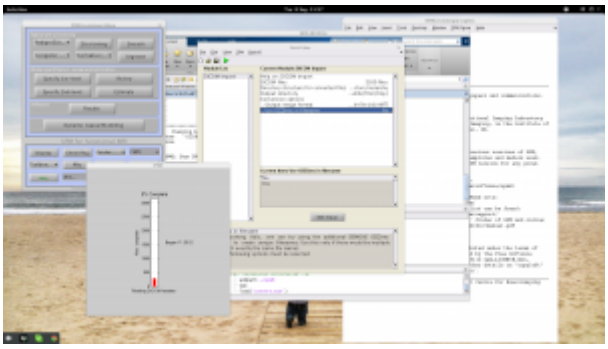
```
$ cd ~/data/facehbi/fmri/
$ for x in /nas/corachan/facehbi/*; do for y in `ls $x`; do a=(`ls "$x/$y" |
head -n 1`) ; if [[ `dckey -k "SeriesDescription" "$x/$y/$a" 2>&1 | grep
bold` ]]; then echo "$x - $y"; fi; done; done 2>&1| grep -v "No such file or
directory"
```

que da un resultado como este:

```
/nas/corachan/facehbi/F005 - Serie(22)
/nas/corachan/facehbi/F009 - Serie(21)
/nas/corachan/facehbi/F010 - Serie(22)
/nas/corachan/facehbi/F011 - Serie(24)
/nas/corachan/facehbi/F012 - Serie(21)
/nas/corachan/facehbi/F013 - Serie(22)
/nas/corachan/facehbi/F015 - Serie(10)
```

Esto se puede escribir a un file o lo que sea pero de entrada ya tenemos con que empezar a trabajar.

Ahora ya abrimos matlab y cargamos el SPM. Lamentablemente aqui hay que ir despacio por cada archivo. La conversion demora mucho. Lo mas parecido es hacer un batch pero todavia no entiendo muy bien que podria hacer para editarlo externamente. asi que ahi vamos,



Después, por cada sujeto, queda un grupo de archivos así:

```
$ ls f005/
sD417843-0024-00001-000001-01.nii  sD417843-0024-00012-000474-01.nii
sD417843-0024-00023-000947-01.nii  sD417843-0024-00034-001420-01.nii
sD417843-0024-00045-001893-01.nii  sD417843-0024-00056-002366-01.nii
sD417843-0024-00067-002839-01.nii
sD417843-0024-00002-000044-01.nii  sD417843-0024-00013-000517-01.nii
sD417843-0024-00024-000990-01.nii  sD417843-0024-00035-001463-01.nii
sD417843-0024-00046-001936-01.nii  sD417843-0024-00057-002409-01.nii
sD417843-0024-00068-002882-01.nii
sD417843-0024-00003-000087-01.nii  sD417843-0024-00014-000560-01.nii
sD417843-0024-00025-001033-01.nii  sD417843-0024-00036-001506-01.nii
sD417843-0024-00047-001979-01.nii  sD417843-0024-00058-002452-01.nii
sD417843-0024-00069-002925-01.nii
sD417843-0024-00004-000130-01.nii  sD417843-0024-00015-000603-01.nii
sD417843-0024-00026-001076-01.nii  sD417843-0024-00037-001549-01.nii
sD417843-0024-00048-002022-01.nii  sD417843-0024-00059-002495-01.nii
sD417843-0024-00070-002968-01.nii
sD417843-0024-00005-000173-01.nii  sD417843-0024-00016-000646-01.nii
sD417843-0024-00027-001119-01.nii  sD417843-0024-00038-001592-01.nii
sD417843-0024-00049-002065-01.nii  sD417843-0024-00060-002538-01.nii
sD417843-0024-00071-003011-01.nii
sD417843-0024-00006-000216-01.nii  sD417843-0024-00017-000689-01.nii
sD417843-0024-00028-001162-01.nii  sD417843-0024-00039-001635-01.nii
sD417843-0024-00050-002108-01.nii  sD417843-0024-00061-002581-01.nii
sD417843-0024-00072-003054-01.nii
sD417843-0024-00007-000259-01.nii  sD417843-0024-00018-000732-01.nii
sD417843-0024-00029-001205-01.nii  sD417843-0024-00040-001678-01.nii
sD417843-0024-00051-002151-01.nii  sD417843-0024-00062-002624-01.nii
sD417843-0024-00073-003097-01.nii
sD417843-0024-00008-000302-01.nii  sD417843-0024-00019-000775-01.nii
sD417843-0024-00030-001248-01.nii  sD417843-0024-00041-001721-01.nii
sD417843-0024-00052-002194-01.nii  sD417843-0024-00063-002667-01.nii
sD417843-0024-00074-003140-01.nii
sD417843-0024-00009-000345-01.nii  sD417843-0024-00020-000818-01.nii
sD417843-0024-00031-001291-01.nii  sD417843-0024-00042-001764-01.nii
```

```
sD417843-0024-00053-002237-01.nii  sD417843-0024-00064-002710-01.nii
sD417843-0024-00010-000388-01.nii  sD417843-0024-00021-000861-01.nii
sD417843-0024-00032-001334-01.nii  sD417843-0024-00043-001807-01.nii
sD417843-0024-00054-002280-01.nii  sD417843-0024-00065-002753-01.nii
sD417843-0024-00011-000431-01.nii  sD417843-0024-00022-000904-01.nii
sD417843-0024-00033-001377-01.nii  sD417843-0024-00044-001850-01.nii
sD417843-0024-00055-002323-01.nii  sD417843-0024-00066-002796-01.nii
```

Pero algunos barridos no son correctos por lo que algunas de las imagenes estan cortadas. Para identificarlas intentamos unirlas,

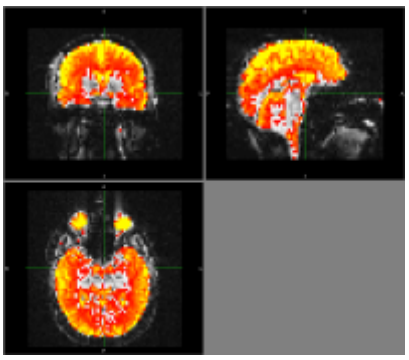
```
$ fslmerge -t f005 `for x in f005/*; do echo $x; done | sed ':a;N;$!ba;s/\n/
/g'`
Error in size-match along non-concatenated dimension for input file:
f005/sD417843-0024-00074-003140-01.nii
```

El error nos dice la imagen que esta mal, asi que la borramos,

```
$ rm f005/sD417843-0024-00074-003140-01.nii
```

y lo intetamos de nuevo hasta tener la imagen correcta,

```
$ ls -l f005.nii.gz
-rw-rw---- 1 osotolongo osotolongo 13786881 Sep  9  2015 f005.nii.gz
```



En este punto tenemos las imagenes correctas convertidas a NIfTI y podemos empezar el preprocesamiento.

Preprocesamiento con AFNI

Esto no es más que una colección de pasos tomados de aquí y probados.

Primero vamos a copiar el archivo a un formato de nombre familiar 🙂

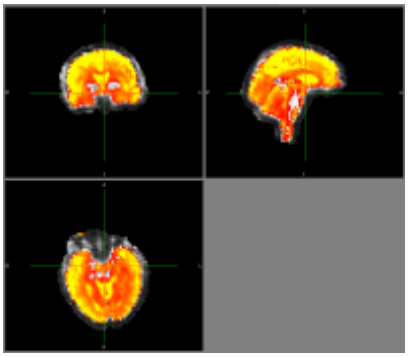
```
$ cp fmri_conv/f005.nii.gz fmri/smc0005s0022.nii.gz
```

y ahora hacemos: *Slice-timing correction*, *Motion correction (realignment)*, *Despiking*

```
$ 3dTshift -prefix fmri/smc0005_st.nii.gz -tpattern altplus
fmri/smc0005s0022.nii.gz
$ 3dvolreg -prefix fmri/smc0005_mc.nii.gz -1Dfile fmri/smc0005_motion.1D -
Fourier -twopass -zpad 4 -base fmri/smc0005_st.nii.gz[0]
fmri/smc0005_st.nii.gz
$ 3dDespike -prefix fmri/smc0005_ds.nii.gz -ssave
fmri/smc0005_spikiness.nii.gz fmri/smc0005_mc.nii.gz
```

Esto de alguna manera podría meterse en un script. A ver,

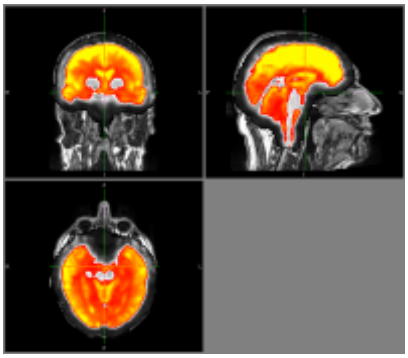
```
#!/bin/sh
# Ejemplo: func_pre.sh smc0014s0006 ~/data/facehbi/fmri

idf=$1
shift
tdir=$1
shift
id=${idf%s0*}

3dTshift -prefix ${tdir}/${id}_st.nii.gz -tpattern altplus
${tdir}/${idf}.nii.gz
3dvolreg -prefix ${tdir}/${id}_mc.nii.gz -1Dfile ${tdir}/${id}_motion.1D -
Fourier -twopass -zpad 4 -base ${tdir}/${id}_st.nii.gz[0]
${tdir}/${id}_st.nii.gz
3dDespike -prefix ${tdir}/${id}_ds.nii.gz -ssave
${tdir}/${id}_spikiness.nii.gz ${tdir}/${id}_mc.nii.gz
fslreorient2std ${tdir}/${id}_ds.nii.gz ${tdir}/${id}_ro.nii.gz
```



Coregistro al espacio anatomico

Ahora hay que mover la imagen funcional al espacio anatomico del sujeto

```
$ flirt -omat working/smc0005_func2anat.mat -cost corratio -dof 12 -interp
trilinear -ref working/smc0005_brain.nii.gz -in fmri/smc0005s0022_ro.nii.gz
$ flirt -out working/smc0005_func_inA.nii.gz -interp trilinear -applyxfm -
init working/smc0005_func2anat.mat -ref working/smc0005_brain.nii.gz -in
fmri/smc0005s0022_ro.nii.gz
```



Reducción de ruido

Esto lo voy a quitar por ahora. Vamos a ver que pasa si confiamos en la estadística.

Primero se segmenta el cerebro anatomico para tener el CSF

```
$ fast -o working/smc0005_segm_A -t 1 -n 3 -g -p
working/smc0005_brain.nii.gz
```

y ahora que tenemos la mascara sacamos la señal en el CSF

```
$ fslmaths working/smc0005_func_inA.nii.gz -mas
working/smc0005_segm_A_seg_0.nii.gz working/smc0005_csf_A.nii.gz
$ 3dmaskave -quiet -mask working/smc0005_csf_A.nii.gz
working/smc0005_func_inA.nii.gz > working/smc0005_func_csf.1D
```

Ya tenemos el ruido por el movimiento de cuando se hizo la correcion de movimiento y ahora sacamos tambien la derivada (por aquello de las corrientes inducidas, autoinducidas, etc). Sumamos todo y aplicamos la correcion.

```
$ cp fmri/smc0005s0022_motion.1D working/smc0005_motion.1D
$ 1d_tool.py -write working/smc0005_motion_deriv.1D -derivative -infile
working/smc0005_motion.1D
$ 1dcat working/smc0005_func_csf.1D working/smc0005_motion.1D
working/smc0005_motion_deriv.1D > working/smc0005_noise.1D
$ 3dBandpass -prefix working/smc0005_cl_inA.nii.gz -mask
working/smc0005_func_inA.nii.gz[0] -ort working/smc0005_noise.1D 0.02 99999
working/smc0005_func_inA.nii.gz
```

Corregistro al espacio estandard

Ahora lo ponemos en el espacio MNI standard, pero esto ya sabiamos hacerlo, ¿no?

```
${FSLDIR}/bin/flirt -ref ${FSLDIR}/data/standard/MNI152_T1_2mm_brain -in
${wdir}/${id}_brain -omat ${wdir}/${id}_tmp_aff_struct2mni.mat
${FSLDIR}/bin/fnirt --in=${wdir}/${id}_struc --
aff=${wdir}/${id}_tmp_aff_struct2mni.mat --
cout=${wdir}/${id}_tmp_nonlinear_transf --config=T1_2_MNI152_2mm
```

```
${FSLDIR}/bin/applywarp --ref=${FSLDIR}/data/standard/MNI152_T1_2mm --
in=${wdir}/${id}_func_inA --out=${wdir}/${id}_func_inMNI --
warp=${wdir}/${id}_tmp_nonlinear_transf
```

Procesamiento (the atlas way)

Al construir la red lo ideal sería calcular la interaccion entre todos lo voxels. *Pero todavía no se me ha ocurrido una manera en que pueda calcularlo con los recursos disponibles*. La solucion por ahora es utilizar un atlas y calcular la interacción entre las diferentes regiones de interés (ROI). El seleccionado *a priori* es el atlas de *Destrieux*. (Ver aqui las ROI).

Lo primero entonces es calcular la serie temporal media en cada una de las ROI. Esto se hace usando **3dmaskave** de *AFNI*,

```
my $order =  '3dmaskave -quiet -mrange '.$roik.' '.$roik.' -mask
'.$ENV{'PIPEDIR'}.$atlas{$parcelation}{'file'}.'
'.$w_dir.'/'.$subject.'_func_inMNI.nii.gz';
my @lines = qx/$order/;
```

y una vez guardado en memoria toda la información se calcula la interacción entre dos nodos de la red como,

<latex> r(x_{1},x_{2})=\frac{\left<V(x_{1},t)V(x_{2},t)\right> - \left<V(x_{1},t)\right>\left<V(x_{2},t)\right>}{\sigma\left(V(x_{1})\right)\sigma\left(V(x_{2})\right)} </latex>

siendo

<latex> \sigma^{2}\left(V(x)\right)=\left<V(x,t)^{2}\right> - \left<V(x,t)\right>^{2} </latex>

😎 Afortunadamente esta es una matriz triangular y

solo hay que calcular la mitad

```
my @corrs;
for (my $i=0; $i<$n; $i++){
    for (my $j=$i; $j<$n; $j++){
        my $vx1 = 0; my $vx2 = 0; my $pvx = 0; my $v2x1 = 0; my $v2x2 = 0;
        for (my $t=0; $t<$ttime; $t++){
            $vx1 += $roi_vt{$roi_codes[$i]}[$t];
            $vx2 += $roi_vt{$roi_codes[$j]}[$t];
            $pvx += $roi_vt{$roi_codes[$i]}[$t]*$roi_vt{$roi_codes[$j]}[$t];
            $v2x1 +=
$roi_vt{$roi_codes[$i]}[$t]*$roi_vt{$roi_codes[$i]}[$t];
            $v2x2 +=
$roi_vt{$roi_codes[$j]}[$t]*$roi_vt{$roi_codes[$j]}[$t];
        }
        $vx1 /= $ttime; $vx2 /= $ttime; $pvx /= $ttime; $v2x1 /= $ttime;
$v2x2 /= $ttime;
```

```
        my $s1 = sqrt($v2x1 - $vx1*$vx1);
        my $s2 = sqrt($v2x2 - $vx2*$vx2);
        $corrs[$i][$j] = ($pvx - $vx1*$vx2)/($s1*$s2);
    }
}
```

Pero para propositos de graficar y demas voy a escribir todo a disco (a no ser que despues necestie ahorrar espacio 😕)

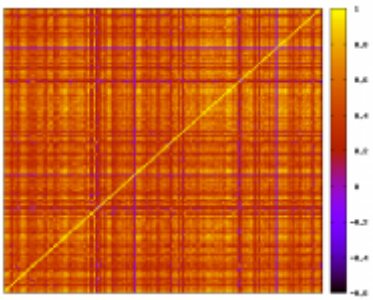
Pero por ahora asi va

```
my $ofile = $w_dir.'/'.$subject.'_'.$parcelation.'.network';
my $mfile = $w_dir.'/'.$subject.'_'.$parcelation.'.map';
open ODF, ">$ofile" or die "Couldn't open ofile";
open MDF, ">$mfile" or die "Couldn't open ofile";
for (my $i=0; $i<$n; $i++){
    print MDF "$i, $roi_codes[$i], $roi_vt{$roi_codes[$i]}\n";
    for (my $j=0; $j<$i; $j++){
        print ODF "$corrs[$j][$i] ";
    }
    for (my $j=$i; $j<$n; $j++){
        print ODF "$corrs[$i][$j] ";
    }
    print ODF "\n";
}
close MDF;
close ODF;
```

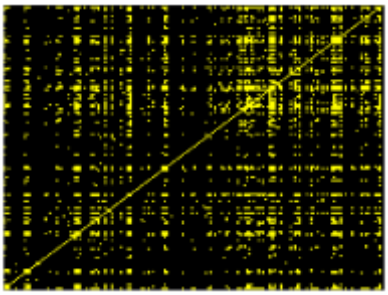
Esto esta todo junto en *func_getnet.pl* que se llama desde *func_nii2net.pl*

Ya tengo la red y ahora que?

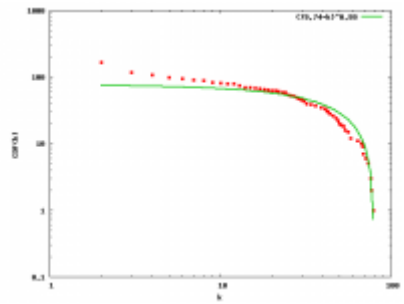
Lo que tienes son las interacciones entre los distintos nodos que has definido.



para convertirla en una red discreta hay que fijar un umbral, por ejemplo $r>0.7$ y quedan definidos los enlaces entre los nodos,



Y ahora se pueden calcular varias cosas! El coeficiente de clustering de cada nodo, <latex> C_{i}=\frac{2E_{i}}{k_{i}(k_{i}-1)} </latex> y por consiguiente el coeficiente de clustering medio. La *Degree distribution* $P(k)$,



Wavelets Analysis

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2827259/

https://cran.r-project.org/web/packages/wavelets/wavelets.pdf

Libro

Statistical Analysis of fMRI Data F. Gregory Ashby

Toolbox

Conn + SPM