Segmentacion del hipocampo

Hippocampal Subfields (Freesurfer)

de XNAT a FS

El primer paso es crear el proyecto, como si se fuera a ejecutar la segmentacion de FS pero sin hacerlo. Basicamente es ejecutar *make_proj.pl* y *update_mri_db.pl* de la manera usual.

El siguiente paso es bajar de XNAT el directorio completo de procesamiento de todos los sujetos y moverlo al directorio usual de FS.

```
$ xnatapic list_subjects --project_id facemopead --label > xnat_subjects.list
$ for x in `awk -F"," '{print $2}' xnat_subjects.list`; do e=$(xnatapic list_experiments --project_id facemopead --subject_id ${x} --modality MRI --label); echo "${x},${e}"; done > xnat_sub_exp.list
$ mkdir pull; cd pull
$ for l in `cat ../xnat_sub_exp.list`; do s=$(echo ${l} | awk -F"," '{print $1}'); e=$(echo ${l} | awk -F"," '{print $2}'); mkdir -p fstgz/${s}; xnatapic get_fsresults --experiment_id ${e} --all-tgz fstgz/${s}/; done
$ okd='/nas/data/mopead/pull'; for x in $(ls -d fstgz/*); do f=$(ls ${x}/*.tar.gz 2>/dev/null); if [[ -e ${f} ]]; then tn=$(echo ${x} | awk -F"/" '{print $2}'); d='mopead_'${tn}; echo "${x},${d},${f}"; mkdir ${d}; cd ${d}; tar xzvf ${okd}/${f} --strip-components=1; cd ${okd} ; fi; done
$ for x in `cat ../mopead_mri.csv`; do id=$(echo ${x} | awk -F";" '{print $1}'); pid=$(echo ${x} | awk -F";" '{print $2}'); mv mopead_${pid} /nas/data/subjects/mopead_${id}; done
```

Nota: Esto tarda un rato pero siempre mucho menos que recalcular la segmentacion.

extraer HSF (v6)

Si ya tenemos la segmentacion inicial, para sacar los HSF de cada sujeto solo hay que hacer (**Para la version 6 de FS**),

```
$ recon-all -s <subject> -hippocampal-subfields-T1
```

Asi que lo que hacemos es aprovechar el codigo escrito para hacer la paralelizacion de FS pero cambiar la ejecucion. (ok, pero voy a aprovechar y convertir el codigo para que use el modulo nuevo de SLURM).

```
my %ptask;
$ptask{'job_name'} = 'hsf_recon_'.$study;
$ptask{'cpus'} = 4;
$ptask{'time'} = '3:0:0';
```

```
foreach my $pkey (sort @plist){
        my $subj = $study."_".$pkey;
        my $ok_subj = check_fs_subj($subj);
        if($ok_subj){
                my $order;
                $ptask{'command'} = "recon-all -s ".$subj." -hippocampal-
subfields-T1 -itkthreads 4";
                $ptask{'filename'} = $outdir.'/'.$subj.'fs_orders.sh';
                $ptask{'output'} = $outdir.'/fs_recon-slurm-'.$subj.'-%j';
                send2slurm(\%ptask);
                sleep(10);
        }
}
```

Nota: Aqui el uso del parametro *-itkthreads 4* acelera notablemente los calculos.

y luego, aviso cuando termine,

```
my %warn;
$warn{'filename'} = $outdir.'/fs_recon_end.sh';
$warn{'job_name'} = 'hsf_recon_'.$study;
$warn{'mailtype'} = 'END'; #email cuando termine
$warn{'output'} =  $outdir.'/fs_recon_end-%j';
$warn{'dependency'} = 'singleton';
send2slurm(\%warn);
```

Lo demas es basicamente los mismo que antes

hsfrecon.pl

```
#!/usr/bin/perl

# Copyright 2021 O. Sotolongo <asqwerty@gmail.com>

# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
use strict; use warnings;
use File::Basename qw(basename);
use Data::Dump qw(dump);
use SLURM qw(send2slurm);
use NEURO4 qw(populate check_fs_subj load_project print_help
check_or_make get_subjects get_list);
```

```
my $cfile;

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

my $study = shift;
unless ($study) { print_help $ENV{'PIPEDIR'}.'/doc/recon.hlp'; exit;}
my $debug = 1;
my $stage = "-autorecon-all";

my %std = load_project($study);
my $data_dir=$std{'DATA'};
my $mri_db = $std{'DATA'}.'/'.$study.'_mri.csv';
my $bids_path = $std{'DATA'}.'/bids';
#open debug file
my $logfile = "$std{'DATA'}/.debug.controlled.log";
$debug ? open DBG, ">$logfile" :0;
#open slurm file
my $orderfile = "$std{'DATA'}/mri_orders.sh";
my $conffile = "$std{'DATA'}/mri_orders.conf";
my $outdir = "$std{'DATA'}/slurm";
check_or_make($outdir);

#get subjects from database or file
my @plist;
my @iplist = get_subjects($mri_db);

if ($cfile){
        my @cuts = get_list($data_dir."/".$cfile);
        foreach my $cut (sort @cuts){
                if(grep {/$cut/} @iplist){
                        push @plist, $cut;
                }
        }
}else{
        @plist = @iplist;
}
my %ptask;
$ptask{'job_name'} = 'hsf_recon_'.$study;
$ptask{'cpus'} = 4;
$ptask{'time'} = '3:0:0';
foreach my $pkey (sort @plist){
        my $subj = $study."_".$pkey;
        my $ok_subj = check_fs_subj($subj);
        if($ok_subj){
```

```
                my $order;
                $ptask{'command'} = "recon-all -s ".$subj." -
hippocampal-subfields-T1 -itkthreads 4";
                $ptask{'filename'} = $outdir.'/'.$subj.'fs_orders.sh';
                $ptask{'output'} = $outdir.'/fs_recon-slurm-'.$subj.'-
%j';
                send2slurm(\%ptask);
                sleep(10);
        }
}

$debug ? close DBG:0;
my %warn;
$warn{'filename'} = $outdir.'/fs_recon_end.sh';
$warn{'job_name'} = 'hsf_recon_'.$study;
$warn{'mailtype'} = 'END'; #email cuando termine
$warn{'output'} =  $outdir.'/fs_recon_end-%j';
$warn{'dependency'} = 'singleton';
send2slurm(\%warn);
```

y lo lanzo de la manera usual

```
$ ls -d /nas/data/subjects/mopead_0*  | awk -F"_" '{print $2}' > done.txt
$ hsfrecon.pl -cut pull/done.txt mopead
```

extraer HSF (v7)

https://surfer.nmr.mgh.harvard.edu/fswiki/HippocampalSubfieldsAndNucleiOfAmygdala

en la version 7.2 de Freesurfer el comando para segmentar el hipocampo ha cambiado asi que voy a poner un nuevo switch para calcular con la v6 (porsiaca),

```
my $legacy = 0;

@ARGV = ("-h") unless @ARGV;
while (@ARGV and $ARGV[0] =~ /^-/) {
    $_ = shift;
    last if /^--$/;
    if (/^-cut/) { $cfile = shift; chomp($cfile);}
    if (/^-old/) { $legacy = 1;}
    if (/^-h/) { print_help $ENV{'PIPEDIR'}.'/doc/recon.hlp'; exit;}
}
```

y luego,

```
               if ($legacy) {
                       $ptask{'command'} = "recon-all -s ".$subj." -
hippocampal-subfields-T1 -itkthreads 4";
```

```
                } else {
                        $ptask{'command'} = "segmentHA_T1.sh ".$subj;
                }
```

Sacando las metricas

Los resutlados de la segmentacion quedan almacenados en el subdirectorio *mri* de cada sujeto, en los archivos *lh.hippoSfVolumes-T1.v10.txt* y *rh.hippoSfVolumes-T1.v10.txt*.

```
my $hsf_i_data = ".hippoSfVolumes-T1.v10.txt";
my @hemis = ("lh", "rh");
```

Hay una herramienta para sacar esto a una tabla pero esto es sencillo y es mejor sacarlo a mi manera,

```
my %metrics;
my %headers;
foreach my $subject (@fspnames){
        foreach my $hemi (@hemis){
                my $hsfdp =
$subj_dir.'/'.$subject.'/mri/'.$hemi.$hsf_i_data;
                if (-e $hsfdp){
                        open IDF, "<$hsfdp";
                        while(<IDF>){
                                my ($hs_k, $hs_v) = /(\S+)\s+(\S+)/;
                                $metrics{$subject}{$hemi}{$hs_k} = $hs_v;
                                $headers{$hs_k} = 1;
                        }
                        close IDF;
                }
        }
}
```

Esto despues se escribe a disco en un csv y aprovecho apra escribir la suma de ambos hemisferios,

```
open ODF, ">$fsout";
print ODF "Subject";
foreach my $hemi (sort @hemis){
        foreach my $tag (sort keys %headers){
                print ODF ",$hemi.$tag";
        }
}
foreach my $tag (sort keys %headers){
        print ODF ",whole.$tag";
}
print ODF "\n";
foreach my $subject (sort @fspnames){
        my $hsfdp = $subj_dir.'/'.$subject.'/mri/lh'.$hsf_i_data;
        if(-e $hsfdp){
```

```
                print ODF "$subject";
                foreach my $hemi (sort @hemis){
                        foreach my $tag (sort keys %headers){
                                print ODF
",$metrics{$subject}{$hemi}{$tag}";
                        }
                }
                foreach my $tag (sort keys %headers){
                        my $whole =
$metrics{$subject}{'lh'}{$tag}+$metrics{$subject}{'rh'}{$tag};
                        print ODF ",$whole";
                }
                print ODF "\n";
        }
}
close ODF;
```

Esto esta tambien escrito encima de un script anterior pero con muchos mas cambios

hsf_metrics.pl

```
#!/usr/bin/perl

# Copyright 2021 O. Sotolongo <asqwerty@gmail.com>

# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
use strict; use warnings;
use NEURO4 qw(get_subjects check_fs_subj load_project print_help
shit_done check_or_make);
#use FSMetrics qw(fs_file_metrics);
use File::Basename qw(basename);
use Data::Dump qw(dump);

my $hsf_i_data = ".hippoSfVolumes-T1.v10.txt";
my @hemis = ("lh", "rh");
# print help if called without arguments
@ARGV = ("-h") unless @ARGV;
while (@ARGV and $ARGV[0] =~ /^-/) {
    $_ = shift;
    last if /^--$/;
    if (/^-h/) { print_help $ENV{'PIPEDIR'}.'/doc/hsf_metrics.hlp';
```

```
exit;}
}

my $study = shift;
unless ($study) { print_help $ENV{'PIPEDIR'}.'/doc/fs_metrics.hlp';
exit;}
my %std = load_project($study);
my $db = $std{DATA}.'/'.$study.'_mri.csv';
my $fsout = $std{DATA}.'/'.$study.'_hsf_metrics.csv';
my @plist = get_subjects($db);
my $subj_dir = $ENV{'SUBJECTS_DIR'};
#my %stats = fs_file_metrics();

my @fspnames;
foreach my $pkey (@plist){
        my $subj = $study."_".$pkey;
        if(check_fs_subj($subj)){
                push @fspnames, $subj;
        }
}
my %metrics;
my %headers;
foreach my $subject (@fspnames){
        foreach my $hemi (@hemis){
                my $hsfdp =
$subj_dir.'/'.$subject.'/mri/'.$hemi.$hsf_i_data;
                if (-e $hsfdp){
                        open IDF, "<$hsfdp";
                        while(<IDF>){
                                my ($hs_k, $hs_v) = /(\S+)\s+(\S+)/;
                                $metrics{$subject}{$hemi}{$hs_k} =
$hs_v;
                                $headers{$hs_k} = 1;
                        }
                        close IDF;
                }
        }
}
open ODF, ">$fsout";
print ODF "Subject";
foreach my $hemi (sort @hemis){
        foreach my $tag (sort keys %headers){
                print ODF ",$hemi.$tag";
        }
}
foreach my $tag (sort keys %headers){
        print ODF ",whole.$tag";
}
print ODF "\n";
foreach my $subject (sort @fspnames){
        my $hsfdp = $subj_dir.'/'.$subject.'/mri/lh'.$hsf_i_data;
```

```
        if(-e $hsfdp){
                print ODF "$subject";
                foreach my $hemi (sort @hemis){
                        foreach my $tag (sort keys %headers){
                                print ODF
",$metrics{$subject}{$hemi}{$tag}";
                        }
                }
                foreach my $tag (sort keys %headers){
                        my $whole =
$metrics{$subject}{'lh'}{$tag}+$metrics{$subject}{'rh'}{$tag};
                        print ODF ",$whole";
                }
                print ODF "\n";
        }
}
close ODF;

my $zfile=$std{DATA}."/".$study."_hsf_results.tgz";
system("tar czf $zfile $fsout");
shit_done basename($ENV{_}), $study, $zfile;
```

y al ejecutarlo,

```
$ hsf_metrics.pl mopead
```

deja los resultados en el archivo *mopead_hsf_metrics.csv*,

```
head mopead_hsf_metrics.csv
Subject,lh.CA1,lh.CA3,lh.CA4,lh.GC-ML-
DG,lh.HATA,lh.Hippocampal_tail,lh.Whole_hippocampus,lh.fimbria,lh.hippocampa
l-
fissure,lh.molecular_layer_HP,lh.parasubiculum,lh.presubiculum,lh.subiculum,
rh.CA1,rh.CA3,rh.CA4,rh.GC-ML-
DG,rh.HATA,rh.Hippocampal_tail,rh.Whole_hippocampus,rh.fimbria,rh.hippocampa
l-
fissure,rh.molecular_layer_HP,rh.parasubiculum,rh.presubiculum,rh.subiculum,
whole.CA1,whole.CA3,whole.CA4,whole.GC-ML-
DG,whole.HATA,whole.Hippocampal_tail,whole.Whole_hippocampus,whole.fimbria,w
hole.hippocampal-
fissure,whole.molecular_layer_HP,whole.parasubiculum,whole.presubiculum,whol
e.subiculum
mopead_0001,449.417399,136.550033,159.203691,180.117640,48.898191,313.070721
,2276.662038,58.591722,144.137698,382.559011,56.468258,202.711162,289.074210
,589.944861,194.255342,240.100806,277.550573,61.633290,406.669721,3062.57650
7,76.109079,158.912734,519.045332,53.013319,245.554280,398.699905,1039.36226
,330.805375,399.304497,457.668213,110.531481,719.740442,5339.238545,134.7008
01,303.050432,901.604343,109.481577,448.265442,687.774115
mopead_0002,570.850319,208.264971,249.568023,289.812260,56.069344,390.857972
```

```
,3009.035855,79.800306,160.138675,496.908364,54.800668,247.570133,364.533496
,618.382260,271.944290,298.421522,339.695352,67.015750,419.979746,3318.27414
1,71.821230,146.474951,558.433172,46.349086,242.476110,383.755622,1189.23257
9,480.209261,547.989545,629.507612,123.085094,810.837718,6327.309996,151.621
536,306.613626,1055.341536,101.149754,490.046243,748.289118
mopead_0003,736.087744,232.871935,291.788096,341.398685,58.837049,472.581220
,3734.598253,107.558269,242.305178,633.964770,57.009950,341.410812,461.08972
2,769.204595,226.712653,277.319436,315.109843,70.625900,542.014004,3842.4049
98,98.548523,211.208520,657.066300,54.664028,331.673962,499.465752,1505.2923
39,459.584588,569.107532,656.508528,129.462949,1014.595224,7577.003251,206.1
06792,453.513698,1291.03107,111.673978,673.084774,960.555474
```

Formato de resultados

Ahora solo hay que cambiar los codigos de neuroimagen por los codigos del proyecto. Para eso se usa el archivo que contiene la base de datos de mris (en este caso *mopead_mri.csv*),

```
$ sed 's/mopead_//' mopead_hsf_metrics.csv > mopead_hsf_tmp.csv
$ sed 's/;/,/; 1iSubject,PID' mopead_mri.csv > mopead_codes.csv
$ join -t, mopead_codes.csv mopead_hsf_tmp.csv | sed 's/[^,]*,//' >
mopead_hsf.csv
```

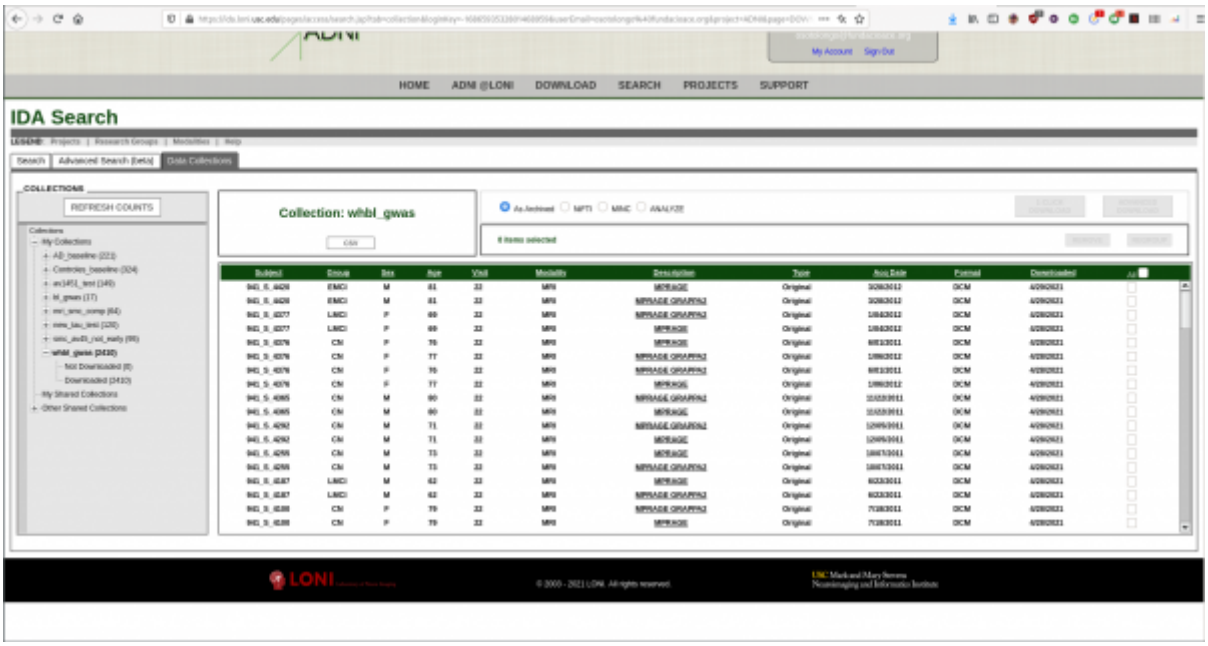
y ya queda correcto,

```
[osotolongo@brick03 mopead]$ head mopead_hsf.csv
PID,lh.CA1,lh.CA3,lh.CA4,lh.GC-ML-
DG,lh.HATA,lh.Hippocampal_tail,lh.Whole_hippocampus,lh.fimbria,lh.hippocampa
l-
fissure,lh.molecular_layer_HP,lh.parasubiculum,lh.presubiculum,lh.subiculum,
rh.CA1,rh.CA3,rh.CA4,rh.GC-ML-
DG,rh.HATA,rh.Hippocampal_tail,rh.Whole_hippocampus,rh.fimbria,rh.hippocampa
l-
fissure,rh.molecular_layer_HP,rh.parasubiculum,rh.presubiculum,rh.subiculum,
whole.CA1,whole.CA3,whole.CA4,whole.GC-ML-
DG,whole.HATA,whole.Hippocampal_tail,whole.Whole_hippocampus,whole.fimbria,w
hole.hippocampal-
fissure,whole.molecular_layer_HP,whole.parasubiculum,whole.presubiculum,whol
e.subiculum
24A8DVSH,449.417399,136.550033,159.203691,180.117640,48.898191,313.070721,22
76.662038,58.591722,144.137698,382.559011,56.468258,202.711162,289.074210,58
9.944861,194.255342,240.100806,277.550573,61.633290,406.669721,3062.576507,7
6.109079,158.912734,519.045332,53.013319,245.554280,398.699905,1039.36226,33
0.805375,399.304497,457.668213,110.531481,719.740442,5339.238545,134.700801,
303.050432,901.604343,109.481577,448.265442,687.774115
24N223T4,570.850319,208.264971,249.568023,289.812260,56.069344,390.857972,30
09.035855,79.800306,160.138675,496.908364,54.800668,247.570133,364.533496,61
8.382260,271.944290,298.421522,339.695352,67.015750,419.979746,3318.274141,7
1.821230,146.474951,558.433172,46.349086,242.476110,383.755622,1189.232579,4
80.209261,547.989545,629.507612,123.085094,810.837718,6327.309996,151.621536
```

```
,306.613626,1055.341536,101.149754,490.046243,748.289118
27FAKA55,736.087744,232.871935,291.788096,341.398685,58.837049,472.581220,37
34.598253,107.558269,242.305178,633.964770,57.009950,341.410812,461.089722,7
69.204595,226.712653,277.319436,315.109843,70.625900,542.014004,3842.404998,
98.548523,211.208520,657.066300,54.664028,331.673962,499.465752,1505.292339,
459.584588,569.107532,656.508528,129.462949,1014.595224,7577.003251,206.1067
92,453.513698,1291.03107,111.673978,673.084774,960.555474
```

Ejemplo ADNI

El objetivo ahora es intentar replicar los resultados en ADNI. La lista de los sujetos con GWAS de ADNI me la dan desde la estratosfera. Lo mio es entrar en ADNI, seleccionar solo T1 en baseline de los sujetos de esta lista y montar una colección.



Aqui hay unas 2400 imagenes pero porque los MPRAGE estan repetidos. O sea, hay varios T1raw para cada sujeto en cada fecha. Bajar esto tiene su historia pero con paciencia y **curl** todo se puede. Basicamente es mandar a bajar los nifti, los dicom pesan demasiado y se queda a la mitad.

Y los metadatos?

Pero aunque no tengamos los DICOM, los datos imprescindibles estan en el CSV que podemos bajar de adni,

```
[osotolongo@brick03 ~]$ head -n 1 /old_nas/adni/whbl_gwas_5_05_2021.csv |
sed 's/"//g;s/,/\n/g' | cat -n
     1 Image Data ID
     2 Subject
     3 Group
     4 Sex
```

```
     5 Age
     6 Visit
     7 Modality
     8 Description
     9 Type
    10 Acq Date
    11 Format
    12 Downloaded

[osotolongo@brick03 ~]$ head /old_nas/adni/whbl_gwas_5_05_2021.csv
"Image Data
ID","Subject","Group","Sex","Age","Visit","Modality","Description","Type","A
cq Date","Format","Downloaded"
"I294080","941_S_4420","EMCI","M","81","22","MRI","MPRAGE","Original","3/28/
2012","DCM","4/29/2021"
"I294078","941_S_4420","EMCI","M","81","22","MRI","MPRAGE
GRAPPA2","Original","3/28/2012","DCM","4/29/2021"
"I275239","941_S_4377","LMCI","F","69","22","MRI","MPRAGE
GRAPPA2","Original","1/04/2012","DCM","4/28/2021"
"I275228","941_S_4377","LMCI","F","69","22","MRI","MPRAGE","Original","1/04/
2012","DCM","4/28/2021"
"I269043","941_S_4376","CN","F","76","22","MRI","MPRAGE","Original","6/01/20
11","DCM","4/28/2021"
"I276870","941_S_4376","CN","F","77","22","MRI","MPRAGE
GRAPPA2","Original","1/06/2012","DCM","4/28/2021"
"I269040","941_S_4376","CN","F","76","22","MRI","MPRAGE
GRAPPA2","Original","6/01/2011","DCM","4/28/2021"
"I276860","941_S_4376","CN","F","77","22","MRI","MPRAGE","Original","1/06/20
12","DCM","4/28/2021"
"I268046","941_S_4365","CN","M","80","22","MRI","MPRAGE
GRAPPA2","Original","11/22/2011","DCM","4/28/2021"
```

El proximo paso es hacer un proyecto y crear una estructura BIDS para estos pollos.
Afortunadamente, el zip se descomprime (*7za x whbl_gwas.zip*) con un directorio por cada usuario,

Asi que sacar las fechas seria algo como,

```
[osotolongo@brick03 ~]$ awk -F"," '{print $2","$10}'
/old_nas/adni/whbl_gwas_5_05_2021.csv | sed 's/"//g' | uniq | sed
's/\,\([^\/]\)\//,0\1\//' | sed 's/,\(.*\)\/\(.*\)\/\(.*\)$/,\3\2\1/' >
/old_nas/adni/whbl_gwas_dates.csv
```

---

montando el proyecto

```
[osotolongo@brick03 gadni]$ ls /old_nas/adni/ADNI/ | head
002_S_0295
002_S_0413
002_S_0559
```

```
002_S_0619
002_S_0685
002_S_0729
002_S_0782
002_S_0816
002_S_0938
002_S_0954
```

asi que es muy sencillo hacer el proyecto correctamente.

```
[osotolongo@brick03 gadni]$ head gadni_mri.csv
0001;002_S_0295
0002;002_S_0413
0003;002_S_0559
0004;002_S_0619
0005;002_S_0685
0006;002_S_0729
0007;002_S_0782
0008;002_S_0816
0009;002_S_0938
0010;002_S_0954
```

Pero crear la estructura BIDS, tiene su truco. Para aprovechar todas las imagenes y hacer la correccion de movimiento correcta en FS, tengo que hacer primero un *conform*. Algunas imagenes tiene tamaño de voxel distinto y FS no se lo traga. Asi que en lugar de copiarlas he de hacer un,

```
$ mri_convert --conform src_img.nii bids_img.nii.gz
```

que lo que hace es convertir todas las imagenes a un tamaño de voxel de 1x1x1 y compactarlas.

Esto lo puedo meter en un script que recorra la DB del proyecto y ponga las imagenes en su sitio,

adni2bids.pl

```
#!/usr/bin/perl
use strict;
use warnings;
use File::Find::Rule;
use File::Path qw(make_path);
use Data::Dump qw(dump);
my $idir = '/old_nas/adni/ADNI';
my $odir = '/nas/data/gadni/bids';
my $rules = '/nas/data/gadni/gadni_mri.csv';

my %subjects;
open IRF, "<$rules" or die "\nNo rules file\n\n";
while (<IRF>){
    my ($skey, $svalue) = /(\d{4});(.*)/;
    $subjects{$skey} = $svalue;
}
#dump %subjects;
```

```
foreach my $subject (sort keys %subjects){
    my $opath = $odir.'/sub-'.$subject.'/anat';
    make_path($opath) unless (-d $opath);
    my $ipath = $idir.'/'.$subjects{$subject};
    my @niftis = find(file => 'name' => '*.nii', in => $ipath);
    my $count = 1;
    foreach my $isrc (@niftis){
        my $itrg = $opath.'/sub-'.$subject.'_run-'.sprintf("%02d",
$count).'_T1w.nii.gz';
        my $order = 'mri_convert --conform '.$isrc.' '.$itrg;
        print "$order\n";
        system($order);
        $count++;
    }
}
```

Al ejecutar el script, la estructura de ADNI,

```
[osotolongo@brick03 gadni]$ tree /old_nas/adni/ADNI/002_S_0295
/old_nas/adni/ADNI/002_S_0295
├── MP-RAGE
│   └── 2006-04-18_08_20_30.0
│       └── S13408
│           └── ADNI_002_S_0295_MR_MP-
RAGE__br_raw_20060418193713091_1_S13408_I13722.nii
└── MP-RAGE_REPEAT
    └── 2006-04-18_08_28_40.0
        └── S13407
            └── ADNI_002_S_0295_MR_MP-
RAGE_REPEAT_br_raw_20060418194821744_1_S13407_I13721.nii

6 directories, 2 files
```

queda en formato BIDS correcto,

```
[osotolongo@brick03 gadni]$ tree bids/sub-0001/
bids/sub-0001/
└── anat
    ├── sub-0001_run-01_T1w.nii.gz
    └── sub-0001_run-02_T1w.nii.gz

1 directory, 2 files
```

Correcion de movimiento en el pipeline

Ahora he de enviar un *recon-all* pero con varias T1. Esto es primera vez que tenemos la oprtunidad de hacerlo así que hemos de cambiar el pipeline. Basta con editar dos archivos. Primero la biblioteca de funciones *NEURO4.pm*, en la funcion *check_subj* se ha de cambiar,

```
                my @t1 = find(file => 'name' => "sub-$subj*_T1w.nii.gz", in
=>  $subj_dir);
                if (@t1 && -e $t1[0] && -f $t1[0]){
                        $mri{'T1w'} = $t1[0];
                }
```

a

```
                my @t1 = find(file => 'name' => "sub-$subj*_T1w.nii.gz", in
=>  $subj_dir);
                if (@t1 && -e $t1[0] && -f $t1[0]){
                         $mri{'T1w'} = \@t1;
                }
```

y ahora en el script de ejecucion (*precon.pl*) debo tomar este array, es decir que el commando anterior,

```
$ptask{'command'} = "recon-all ".$stage." -i ".$nifti{'T1w'}." -subjid
".$subj;
```

ahora se convierte en,

```
my $t1char = join ' -i ', @{$nifti{'T1w'}};
$ptask{'command'} = "recon-all ".$stage." -i ".$t1char." -subjid ".$subj;
```

Por supuesto que si hay un solo T1 el resultado es el mismo que antes. 😉

Nota: *preview de la v0.5*. He aprovechado que tenia que cambiar el script y cambiado todas las ejecuciones de *sbatch* por llamadas a la libreria *SLURM.pm*. Por aquello de hacerlo modular, etc. 😃

Procesando

Entonces,

```
$ precon.pl gadni
```

genera unas mil y algo reconstrucciones de FS. Las que fallen las eliminio de *gadni_mri.csv* y luego,

```
$ hsfrecon.pl gadni
```

me lanza las segmentaciones del hipocampo. Y las metricas se recogen como,

```
% hsf_metrics.pl gadni
```

TADA!!!!!

y ahora junta y demas

```
$ awk -F"," '{print "gadni_"$1","$65}'
```

```
/nas/data/gadni/fsrecon/aseg_stats.csv | sed 's/gadni_Subject/Subject/'>
gadni_icv.csv
$ join -t, gadni_hsf_metrics_wpid.csv gadni_icv.csv >
gadni_hsf_metrics_wicv.csv
```

```
  497  head gadni_hsf_metrics_wicv.csv
  498  wc -l gadni_hsf_metrics_wicv.csv
  499  history
  500  history | gregp f2cehbi
  501  history | grep f2cehbi
  502  sed 's/;/,/;'1iSubject,PSubject'' /nas/data/epad/epad_codes.csv.csv >
epad_codes.csv
  503  sed 's/;/,/;'1iSubject,PSubject'' /nas/data/epad/epad_codes.csv >
epad_codes.csv
  504  head epad_codes.csv
  505  join -t, -1 2 -2 1 epad_codes.csv /nas/data/epad/epad_internos.csv
  506  history | grep f2cehbi
  507  join -t, -1 2 -2 1 epad_codes.csv /nas/data/epad/epad_internos.csv  |
sed 's/,\(.*\),/,epad_\1,/;s/epad_Subject/Subject/' > epad_internos.csv
  508  head epad_internos.csv
  509  sed 's/;/,/;'1iSubject,PSubject'' /nas/data/mopead/mopead_mri.csv >
mopead_codes.csv
  510  join -t, -1 2 -2 1 mopead_codes.csv
/nas/data/mopead/mopead_internos.csv  | sed
's/,\(.*\),/,mopead_\1,/;s/mopead_Subject/Subject/' > mopead_internos.csv
  511  head mopead_internos.csv
  512  ls *_internos.csv
  513  head bioface_internos.csv
  514  join -t, -1 2 -2 1 bioface_internos.csv
/nas/data/bioface/bioface_hsf_metrics.csv > bioface_hsf.csv
  515  history | grep gadni
  516  awk -F"," '{print "bioface_"$1","$65}'
/nas/data/bioface/fsrecon/aseg_stats.csv | sed 's/bioface_Subject/Subject/'
> bioface_icv.csv
  517  head bioface_icv.csv
  518  join -t, bioface_hsf.csv bioface_icv.csv > bioface_hsf_wicv.csv
  519  head bioface_hsf_wicv.csv
  520  history | grep bioface
  521  join -t, -1 2 -2 1 facehbi_internos.csv
/nas/data/facehbi/facehbi_hsf_metrics.csv > facehbi_hsf.csv
  522  head facehbi_hsf.csv
  523  wc -l facehbi_hsf.csv
  524  history | grep bioface
  525  awk -F"," '{print "facehbi_"$1","$65}'
/nas/data/facehbi/fsrecon/aseg_stats.csv | sed 's/facehbi_Subject/Subject/'
> facehbi_icv.csv
  526  join -t, facehbi_hsf.csv facehbi_icv.csv > facehbi_hsf_wicv.csv
  527  wc -l facehbi_hsf_wicv.csv
  528  join -t, -1 2 -2 1 f2cehbi_internos.csv
/nas/data/f2cehbi/f2cehbi_hsf_metrics.csv > f2cehbi_hsf.csv
  529  awk -F"," '{print "f2cehbi_"$1","$65}'
```

```
/nas/data/f2cehbi/fsrecon/aseg_stats.csv | sed 's/f2cehbi_Subject/Subject/'
> f2cehbi_icv.csv
  530  join -t, f2cehbi_hsf.csv f2cehbi_icv.csv > f2cehbi_hsf_wicv.csv
  531  join -t, -1 2 -2 1 epad_internos.csv
/nas/data/epad/epad_hsf_metrics.csv > epad_hsf.csv
  532  awk -F"," '{print "epad_"$1","$65}'
/nas/data/epad/fsrecon/aseg_stats.csv | sed 's/epad_Subject/Subject/' >
epad_icv.csv
  533  join -t, epad_hsf.csv epad_icv.csv > epad_hsf_wicv.csv
  534  join -t, -1 2 -2 1 mopead_internos.csv
/nas/data/mopead/mopead_hsf_metrics.csv > mopead_hsf.csv
  535  awk -F"," '{print "mopead_"$1","$65}'
/nas/data/mopead/fsrecon/aseg_stats.csv | sed 's/mopead_Subject/Subject/' >
mopead_icv.csv
  536  join -t, mopead_hsf.csv mopead_icv.csv > mopead_hsf_wicv.csv
  537  ls *_wicv.csv
  538  body=$(mktemp)
  539  tmp1=$(mktemp)
  540  tmp2=$(mktemp)
  541  tail -n +2 facehbi_hsf_wicv.csv > $body
  542  cat bioface_hsf_wicv.csv $body > $tmp1
  543  tail -n +2 f2cehbi_hsf_wicv.csv > $body
  544  cat $tmp1 $body > $tmp2
  545  tail -n +2 epad_hsf_wicv.csv > $body
  546  cat $tmp2 $body > $tmp1
  547  tail -n +2 mopead_hsf_wicv.csv > $body
  548  cat $tmp1 $body > $tmp2
  549  cat $tmp1 $body > all_wicv.csv
```