VBM FACEHBI visita basal

Data

Queremos analizar la variabilidad, en el volumen de materia gris, que aportan las variables,

- Edad
- Genero
- APOEe4
- Centiloid
- SCDplus_Total (con el sumatorio sería suficiente)
- COMPOSITE_executive_fluency
- COMPOSITE_executive_processing speed
- COMPOSITE_executive_attention
- COMPOSITE_memory_FNAME professions
- COMPOSITE_memory_FNAME names
- COMPOSITES_memory_WMS
- COMPOSITE_memory_RBANS
- COMPOSITE_gnosis
- COMPOSITE_praxis
- COMPOSITE_languge_naming

¿Que tenemos?

```
[osotolongo@brick03 facehbi]$ head -n 1
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | sed 's/,/\n/g' | cat -n
     1  Subject
     2  SubjID
     3  code_facehbi
     4  N_Interno
     5  Date
     6  edat
     7  Anyos_Escolaridad_FAC
     8  Sex_1H_0M
     9  SCDplus_Memorycomplaint
    10  SCDplus_APOE4
    11  SCDplus_concernsaboutcognition
    12  SCDplus_feelingworsethancontemporarypeers
    13  SCDplus_informantcorroboratessymptoms
    14  SCDplus_onset60Y
    15  SCDplus_onsetwithinlast5Y
    16  SCDplus_SUVRgt1.35
    17  SCDplus_SUVRgt1.45
    18  SCDplus_Total7
    19  Q_QSM_JPO
    20  SUVR
    21  Centilod
    22  APOE
```

```
23  COMPOSITE_executive_fluency
24  COMPOSITE_executive_processing speed
25  COMPOSITE_executive_attention
26  COMPOSITE_memory_FNAME professions
27  COMPOSITE_memory_FNAME names
28  COMPOSITES_memory_WMS
29  COMPOSITE_memory_RBANS
30  COMPOSITE_gnosis
31  COMPOSITE_praxis
32  COMPOSITE_languge_naming
```

Voy a empezar por las tres primeras, despues es solo repetir. Vamos a meter todo junto a ver que pasa.

```
[osotolongo@brick03 facehbi]$ awk -F"," {'print $2","$6","$7","$8","$10'}
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | head
SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,SCDplus_APOE4
/nas/data/subjects/facehbi_0001,71,8,0,1
/nas/data/subjects/facehbi_0002,70,12,1,0
/nas/data/subjects/facehbi_0003,70,8,0,0
/nas/data/subjects/facehbi_0004,76,16,0,0
/nas/data/subjects/facehbi_0005,68,20,1,0
/nas/data/subjects/facehbi_0006,64,14,0,1
/nas/data/subjects/facehbi_0007,59,19,1,0
/nas/data/subjects/facehbi_0008,55,16,0,0
/nas/data/subjects/facehbi_0009,67,16,0,0
```

Aqui esta todo, solo hay que depurarlo un poco,

```
[osotolongo@brick03 facehbi]$ awk -F"," {'print $2","$6","$7","$8","$10'}
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,SCDplus_APOE4/
Subject,Age,Education,Gender,APOE/' > data_apoe.csv
[osotolongo@brick03 facehbi]$ head data_apoe.csv
Subject,Age,Education,Gender,APOE
0001,71,8,0,1
0002,70,12,1,0
0003,70,8,0,0
0004,76,16,0,0
0005,68,20,1,0
0006,64,14,0,1
0007,59,19,1,0
0008,55,16,0,0
0009,67,16,0,0
```

Este va a ser mi primer modelo. De aqui saco, la lista de sujetos,

```
[osotolongo@brick03 facehbi]$ awk -F"," {'print $1'} data_apoe.csv | tail -n
+2 > subjects.list
```

el *design.mat*,

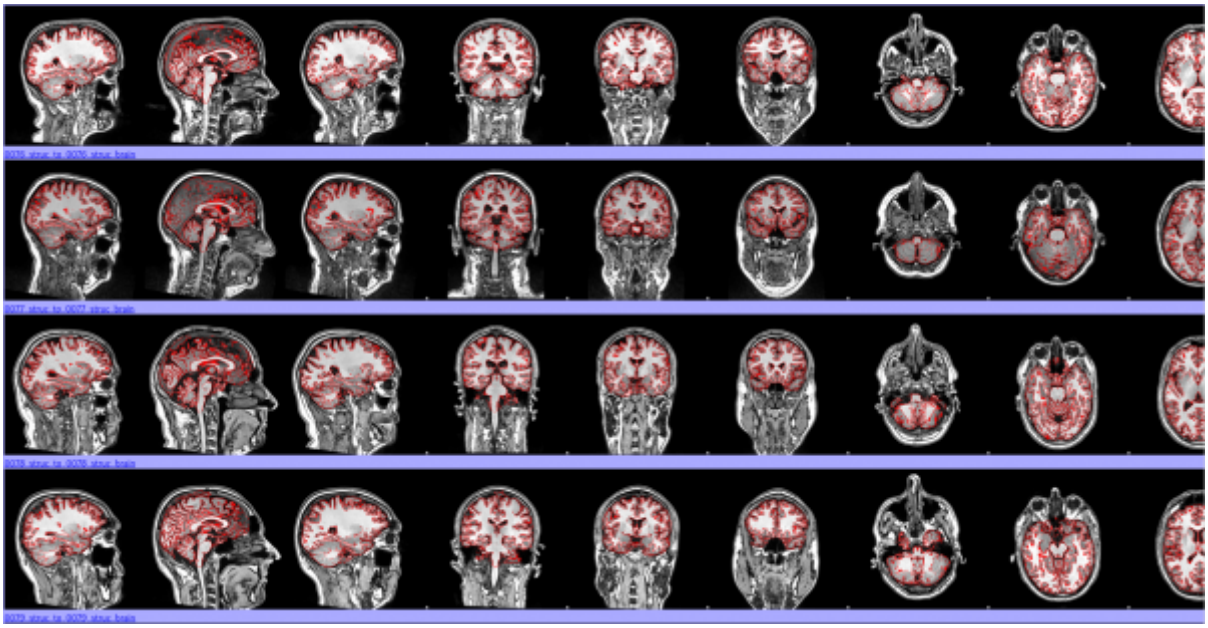
```
[osotolongo@brick03 facehbi]$ awk -F"," {'print $2,$3,$4,$5'} data_apoe.csv
| tail -n +2 > data_apoe.dat
[osotolongo@brick03 facehbi]$ Text2Vest data_apoe.dat design_apoe.mat
[osotolongo@brick03 facehbi]$ head design_apoe.mat
/NumWaves 4
/NumPoints 198
/Matrix
71 8 0 1
70 12 1 0
70 8 0 0
76 16 0 0
68 20 1 0
64 14 0 1
59 19 1 0
```

y de paso ya hago el *design.con*,

```
[osotolongo@brick03 facehbi]$ cat contrast.txt
0 0 0 1
0 0 0 -1
[osotolongo@brick03 facehbi]$ Text2Vest contrast.txt design.con
[osotolongo@brick03 facehbi]$ cat design.con
/NumWaves 4
/NumPoints 2
/Matrix
0 0 0 1
0 0 0 -1
```

Preparando

```
[osotolongo@brick03 facehbi]$ mv design.con vbmcomps/
[osotolongo@brick03 facehbi]$ mv design_apoe.mat vbmcomps/
[osotolongo@brick03 facehbi]$ mv subjects.list vbmcomps/
[osotolongo@brick03 facehbi]$ cd vbmcomps/
[osotolongo@brick03 vbmcomps]$ mkdir struc
[osotolongo@brick03 vbmcomps]$ for x in `cat subjects.list`; do
get_fsbrain.sh facehbi ${x} struc; done
[osotolongo@brick03 vbmcomps]$ cd struc/
[osotolongo@brick03 struc]$ for x in `cat ../subjects.list`; do a="$a
${x}_struc ${x}_struc_brain"; done
[osotolongo@brick03 struc]$ slicesdir -o -e 0.1 $a
```



```
[osotolongo@brick03 struc]$ cd ..
[osotolongo@brick03 vbmcomps]$ fslvbm_2_template -n
```

Nota: El *0107* no se registra bien a la plantilla. Dado que es uno de 197 sujetos, la opcion mas viable parece ser eliminarlo del VBM. Hay que eliminar los archivos, editar las matrices, etc. **Y correr todo de nuevo** 😕

```
[osotolongo@brick03 vbmcomps]$ fslvbm_3_proc
[osotolongo@brick03 vbmcomps]$ fslmaths stats/GM_merg.nii.gz -s 4.0
stats/GM_merg_s4.nii.gz
```

Analisis

Y ahora es que viene el primer analisis,

```
[osotolongo@brick03 vbmcomps]$ randomise -i stats/GM_merg_s4.nii.gz -o
stats/fslvbm_apoe -m stats/GM_mask -d design_apoe.mat -t design.con -n 5000
-T -V
```

Escogiendo otra variable

Primero voy a quitar el pollo que no salio bien,

```
[osotolongo@brick03 vbmcomps]$ sed '/^107,/d'
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv >
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv
```

Revisa!!!!!

```
[osotolongo@brick03 vbmcomps]$ diff
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv
104d103
<
107,/nas/data/subjects/facehbi_0107,F112,20131187,23.07.2015,52,15,1,1,0,1,0
,1,0,1,0,0,4,14,0.955694548,-8.296456382,e3e3,0.875136,-1.236862,1.306347,1.
725823,2.119303,1.758764,0.725239,0.168598,0.515105,1.00369
```

y ahora reestructuro los datos,

```
[osotolongo@brick03 vbmcomps]$ awk -F"," {'print $2","$6","$7","$8","$18'}
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,SCDplus_Total7
/Subject,Age,Education,Gender,SCDTotal/' > data_scd.csv
[osotolongo@brick03 vbmcomps]$ head data_scd.csv
Subject,Age,Education,Gender,SCDTotal
0001,71,8,0,5
0002,70,12,1,3
0003,70,8,0,2
0004,76,16,0,3
0005,68,20,1,3
0006,64,14,0,6
0007,59,19,1,2
0008,55,16,0,3
0009,67,16,0,2
[osotolongo@brick03 vbmcomps]$ awk -F"," {'print $2,$3,$4,$5'} data_scd.csv
| tail -n +2 > data_scd.dat
[osotolongo@brick03 vbmcomps]$ Text2Vest data_scd.dat design_scd.mat
[osotolongo@brick03 vbmcomps]$ head design_scd.mat
/NumWaves 4
/NumPoints 197
/Matrix
71 8 0 5
70 12 1 3
70 8 0 2
76 16 0 3
68 20 1 3
64 14 0 6
59 19 1 2
```

y ejecutamos de nuevo,

```
[osotolongo@brick01 vbmcomps]$ randomise -i stats/GM_merg_s4.nii.gz -o
stats/fslvbm_scd -m stats/GM_mask -d design_scd.mat -t design.con -n 5000 -T
-V
```

workaround para los NA

```
[osotolongo@brick02 vbmcomps]$ awk -F"," {'print $2","$6","$7","$8","$21'}
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,Centilod/Subje
ct,Age,Education,Gender,Centiloid/' > data_centiloid.csv
[osotolongo@brick02 vbmcomps]$ head data_centiloid.csv
Subject,Age,Education,Gender,Centiloid
0001,71,8,0,-5.593250248
0002,70,12,1,12.05680811
0003,70,8,0,-7.342333945
0004,76,16,0,10.12495194
0005,68,20,1,-8.16586361
0006,64,14,0,37.36310815
0007,59,19,1,75.89280769
0008,55,16,0,-17.19557899
0009,67,16,0,3.613350513
```

Mira aqui

```
[osotolongo@brick02 vbmcomps]$ grep NA data_centiloid.csv
0096,64,15,0,NA
```

Por cada *missing* (NA) he de añadir una columna marcandolo. Por suerte aqui es solo uno,

```
[osotolongo@brick02 vbmcomps]$ awk -F"," {'print $2,$3,$4,$5" 0"'}
data_centiloid.csv | tail -n +2 | sed 's/NA 0/0 1/' > data_centiloid.dat
```

y tenemos entonces esto,

```
...
63 16 1 -4.004566361 0
81 12 0 -3.596007943 0
64 15 0 0 1
69 14 0 -20.89228368 0
61 18 0 -14.04746122 0
...
```

Lo demas es mas de lo mismo,

```
[osotolongo@brick02 vbmcomps]$ Text2Vest data_centiloid.dat
design_centiloid.mat
[osotolongo@brick02 vbmcomps]$ head design_centiloid.mat
/NumWaves 5
/NumPoints 197
/Matrix
71 8 0 -5.593250248 0
70 12 1 12.05680811 0
70 8 0 -7.342333945 0
76 16 0 10.12495194 0
```

```
68 20 1 -8.16586361 0
64 14 0 37.36310815 0
59 19 1 75.89280769 0
```

Pero ahora el contraste tiene que cambiar tambien

```
[osotolongo@brick02 vbmcomps]$ cat centiloid_contrast.txt
0 0 0 1 0
0 0 0 -1 0
[osotolongo@brick02 vbmcomps]$ Text2Vest centiloid_contrast.txt
design_centiloid.con
[osotolongo@brick02 vbmcomps]$ head design_centiloid.con
/NumWaves 5
/NumPoints 2
/Matrix
0 0 0 1 0
0 0 0 -1 0
```

y,

```
[osotolongo@brick02 vbmcomps]$ randomise -i stats/GM_merg_s4.nii.gz -o
stats/fslvbm_centiloid -m stats/GM_mask -d design_centiloid.mat -t
design_centiloid.con -n 5000 -T -V
```

Y asi sucesivamente para cada una de las variables de interes.

Automagico por una lista de variables

Tenemos el siguiente problema. Hay una lista de variables sobre las que queremos correr el analisis. Cada una por separado. En cada caso debemos escoger las covriable, la variable a estudiar, hacer la matriz correspondiente y ejecutar el randomize. Como es Como es algo repetitivo, voy a hacer un script que lo haga.

Lo primero es saber cuales varaibles quiero analizar,

```
[osotolongo@brick03 vbmcomps]$ head -n 1
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv | sed 's/,/\n/g' | cat -n
     1 Subject
     2 SubjID
     3 code_facehbi
     4 N_Interno
     5 Date
     6 edat
     7 Anyos_Escolaridad_FAC
     8 Sex_1H_0M
     9 SCDplus_Memorycomplaint
    10 SCDplus_APOE4
    11 SCDplus_concernsaboutcognition
    12 SCDplus_feelingworsethancontemporarypeers
```

```
    13 SCDplus_informantcorroboratessymptoms
    14 SCDplus_onset60Y
    15 SCDplus_onsetwithinlast5Y
    16 SCDplus_SUVRgt1.35
    17 SCDplus_SUVRgt1.45
    18 SCDplus_Total7
    19 Q_QSM_JPO
    20 SUVR
    21 Centilod
    22 APOE
    23 COMPOSITE_executive_fluency
    24 COMPOSITE_executive_processing speed
    25 COMPOSITE_executive_attention
    26 COMPOSITE_memory_FNAME professions
    27 COMPOSITE_memory_FNAME names
    28 COMPOSITES_memory_WMS
    29 COMPOSITE_memory_RBANS
    30 COMPOSITE_gnosis
    31 COMPOSITE_praxis
    32 COMPOSITE_languge_naming
```

De aqui escogemos los numeros adecuados para las variables,

```
# Static covariables
stcov = [6, 7, 8]
# Variables a procesar
dynvars = list(range(9,18))+list(range(23,32))
```

Ahora , para cada variable debemos crear una matriz de diseño,

```
for v in dynvars:
  ltw = ''
  with open(datafile) as f:
    reader = csv.reader(f)
    for row in reader:
      if not row[0] == 'Subject':
        for sv in stcov:
          ltw += row[sv-1]+' '
        ltw += row[v]+'\n'
  f.close()
  tempfile = 'tempdata_'+str(v)+'.dat'
  df = open(tempfile, 'w')
  df.write(ltw)
  df.close()
  os.system('Text2Vest '+df.name+' design_var'+str(v)+'.mat')
```

y ejecutar un *randomise*. Pero voy a hacerlo usando slurm, asi que he de crear y ejecutar un archivos sbatch para cada variable,

```
  bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'+'#SBATCH -c
'+str(cpus)+'\n'+'#SBATCH --mem-per-cpu=4G\n'
```

```
  bcontent += '#SBATCH --time='+time+'\n'+'#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT\n'
  bcontent += '#SBATCH --mail-user='+os.environ.get('USER')+'\n'+'#SBATCH -o
'+wdir+'/'+randomise+'-%j'+'\n'
  bcontent += 'srun randomise -i '+templatefile+' -o
stats/fslvbm_var'+str(v)+' -m stats/GM_mask -d '+'design_var'+str(v)+'.mat -
t desig.con -n 5000 -T -V'
  sbfilename = outdir+'/run_var_'+str(v)+'.sh'
  sbf = open(sbfilename, 'w')
  sbf.write(bcontent)
  sbf.close()
  os.system('sbatch '+sbfile
```

y de paso me aviso por correo cuando termine,

```
bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'
bcontent += '#SBATCH --mail-type=END\n'
bcontent += '#SBATCH --mail-user='+os.environ.get('USER')+'\n'+'#SBATCH -o
'+wdir+'/'+randomise+'-%j'+'\n'
bcontent += ':\n'
bfilename = outdir+'/run_var_'+str(v)+'.sh' sbf = open(sbfilename, 'w')
sbf.write(bcontent)
sbf.close()
os.system('sbatch --dependency=singleton '+sbfilename)
```

Al script hay que indicarle cual es la base de datos y donde esta el template para ejecutar el randomise,

```
[osotolongo@brick03 vbmcomps]$ ./pvbm.py -d
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv -t stats/GM_merg_s4.nii.gz
Submitted batch job 1118
Submitted batch job 1119
Submitted batch job 1120
Submitted batch job 1121
Submitted batch job 1122
Submitted batch job 1123
Submitted batch job 1124
Submitted batch job 1125
Submitted batch job 1126
Submitted batch job 1127
Submitted batch job 1128
Submitted batch job 1129
Submitted batch job 1130
Submitted batch job 1131
Submitted batch job 1132
Submitted batch job 1133
Submitted batch job 1134
Submitted batch job 1135
Submitted batch job 1136
```

y ahi van,

```
[osotolongo@brick03 vbmcomps]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
              1136      fast randomis osotolon PD       0:00      1
(Dependency)
              1118      fast randomis osotolon  R       0:37      1 brick02
              1119      fast randomis osotolon  R       0:34      1 brick02
              1120      fast randomis osotolon  R       0:34      1 brick02
              1121      fast randomis osotolon  R       0:34      1 brick02
              1122      fast randomis osotolon  R       0:34      1 brick02
              1123      fast randomis osotolon  R       0:34      1 brick02
              1124      fast randomis osotolon  R       0:33      1 brick02
              1125      fast randomis osotolon  R       0:31      1 brick02
              1126      fast randomis osotolon  R       0:31      1 brick02
              1127      fast randomis osotolon  R       0:31      1 brick02
              1128      fast randomis osotolon  R       0:28      1 brick02
              1129      fast randomis osotolon  R       0:28      1 brick02
              1130      fast randomis osotolon  R       0:28      1 brick02
              1131      fast randomis osotolon  R       0:25      1 brick02
              1132      fast randomis osotolon  R       0:25      1 brick02
              1133      fast randomis osotolon  R       0:25      1 brick02
              1134      fast randomis osotolon  R       0:25      1 brick03
              1135      fast randomis osotolon  R       0:22      1 brick03
```

Clusters

Ahora, con muy poco esfuerzo podemos pedir al script que saque los clusters. Necesitaremos el *jobid* del radomise, asi que hay que cambiar un poco la primera llamada,

```
  bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'+'#SBATCH -c
'+str(cpus)+'\n'+'#SBATCH --mem-per-cpu=4G\n'
  bcontent += '#SBATCH --time='+time+'\n'+'#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT\n'
  bcontent += '#SBATCH --mail-user='+os.environ.get('USER')+'\n'+'#SBATCH -o
'+outdir+'/randomise-%j'+'\n'
  bcontent += 'randomise -i '+wdir+'/'+templatefile+' -o
'+wdir+'/stats/fslvbm_var'+str(v)+' -m '+wdir+'/stats/GM_mask -d '
  bcontent += wdir+'/design_var'+str(v)+'.mat -t '+wdir+'/design.con -n 5000
-T\n'
  sbfilename = outdir+'/run_var_'+str(v)+'.sh'
  sbf = open(sbfilename, 'w')
  sbf.write(bcontent)
  sbf.close()
  jobid = subprocess.getoutput('sbatch --parsable '+sbfilename) # capturo el
jobid
  #Aqui lanzo la busqueda de clusters
  bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'+'#SBATCH -c
'+str(cpus)+'\n'+'#SBATCH --mem-per-cpu=4G\n'
```

```
  bcontent += '#SBATCH --time='+time+'\n'+'#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT\n'
  bcontent += '#SBATCH --mail-user='+os.environ.get('USER')+'\n'+'#SBATCH -o
'+outdir+'/cluster-%j'+'\n'
  bcontent += 'cluster --
in='+wdir+'/stats/fslvbm_var'+str(v)+'_tfce_corrp_tstat1 --thresh=0.95 -o
'+wdir+'/stats/clusters_var'+str(v)+' >
'+wdir+'/stats/clusters_var'+str(v)+'_index.txt\n'
  sbfilename = outdir+'/cluster_var_'+str(v)+'.sh'
  sbf = open(sbfilename, 'w')
  sbf.write(bcontent)
  sbf.close()
  os.system('sbatch --dependency=afterok:'+jobid+' '+sbfilename)
```

Full code here

pvbm.py

```
#!/usr/bin/python3

"""
This program executes FSL randomise command over a given set of
external
variables. You should supply the FSL template for the group of
subjects,
the variables list, the full curated data and the design matrices. The
script only  parallelize the set of randomise calls over the cluster.

"""

import os
import sys
import getopt
import csv
import re
import tempfile
import subprocess

# Static covariables
stcov = [6, 7, 8]
# Variables a procesar
dynvars = list(range(9,18))+list(range(23,32))

#Entorno de trabajo
time = '12:0:0'
cpus = 4
wdir = os.environ.get('PWD')
outdir = wdir+'/slurm'
if not os.path.isdir(outdir): os.mkdir(outdir)
```

```
# Get CLI inputs
short_args = 'd:t:'
long_args = ['data=', 'template=']

try:
  args, values = getopt.getopt(sys.argv[1:], short_args, long_args)
except getopt.error as err:
  print(str(err))
  sys.exit(2)

for a,v in args:
  if a in ('--data', '-d'):
    datafile = v
  elif a in ('--template', '-t'):
    templatefile = v
for v in dynvars:
  ltw = ''
  with open(datafile) as f:
    reader = csv.reader(f)
    for row in reader:
      if not row[0] == 'Subject':
        for sv in stcov:
          ltw += row[sv-1]+' '
        ltw += row[v]+'\n'
  f.close()
  tempfile = 'tempdata_'+str(v)+'.dat'
  df = open(tempfile, 'w')
  df.write(ltw)
  df.close()
  os.system('Text2Vest '+df.name+' design_var'+str(v)+'.mat')
  bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'+'#SBATCH -c
'+str(cpus)+'\n'+'#SBATCH --mem-per-cpu=4G\n'
  bcontent += '#SBATCH --time='+time+'\n'+'#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT\n'
  bcontent += '#SBATCH --mail-
user='+os.environ.get('USER')+'\n'+'#SBATCH -o '+outdir+'/randomise-
%j'+'\n'
  bcontent += 'randomise -i '+wdir+'/'+templatefile+' -o
'+wdir+'/stats/fslvbm_var'+str(v)+' -m '+wdir+'/stats/GM_mask -d '
  bcontent += wdir+'/design_var'+str(v)+'.mat -t '+wdir+'/design.con -n
5000 -T\n'
  sbfilename = outdir+'/run_var_'+str(v)+'.sh'
  sbf = open(sbfilename, 'w')
  #bcontent = '#!/bin/bash\n:\n'
  sbf.write(bcontent)
  sbf.close()
  jobid = subprocess.getoutput('sbatch --parsable '+sbfilename)
  bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'+'#SBATCH -c
'+str(cpus)+'\n'+'#SBATCH --mem-per-cpu=4G\n'
  bcontent += '#SBATCH --time='+time+'\n'+'#SBATCH --mail-
type=FAIL,TIME_LIMIT,STAGE_OUT\n'
```

```
  bcontent += '#SBATCH --mail-
user='+os.environ.get('USER')+'\n'+'#SBATCH -o '+outdir+'/cluster-
%j'+'\n'
  bcontent += 'cluster --
in='+wdir+'/stats/fslvbm_var'+str(v)+'_tfce_corrp_tstat1 --thresh=0.95
-o '+wdir+'/stats/clusters_var'+str(v)+' >
'+wdir+'/stats/clusters_var'+str(v)+'_index.txt\n'
  sbfilename = outdir+'/cluster_var_'+str(v)+'.sh'
  sbf = open(sbfilename, 'w')
  sbf.write(bcontent)
  sbf.close()
  os.system('sbatch --dependency=afterok:'+jobid+' '+sbfilename)
bcontent = '#!/bin/bash\n'+'#SBATCH -J randomise\n'
bcontent += '#SBATCH --mail-type=END\n'
bcontent += '#SBATCH --mail-user='+os.environ.get('USER')+'\n'+'#SBATCH
-o '+outdir+'/randomise-%j'+'\n'
bcontent += ':\n'
bfilename = outdir+'/run_var_'+str(v)+'.sh'
sbf = open(sbfilename, 'w')
sbf.write(bcontent)
sbf.close()
os.system('sbatch --dependency=singleton '+sbfilename)
```

Ahora, si hacemos ,

```
[osotolongo@brick03 vbmcomps]$ ./pvbm.py -d
datacomb_freesurfer_neuro_v0_scdSUMCC_ok.csv -t stats/GM_merg_s4.nii.gz
```

e lanzan todos los comandos *randomise* en el cluster y posteriormente se sacan los clusters correspondientes al resultado de cada variable,

Los *p-values* quedan, segun el codigo de cada variable, en,

```
[osotolongo@brick03 vbmcomps]$ ls stats/fslvbm_var*_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var10_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var24_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var11_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var25_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var12_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var26_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var13_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var27_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var14_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var28_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var15_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var29_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var16_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var30_tfce_corrp_tstat1.nii.gz
```

```
stats/fslvbm_var17_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var31_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var23_tfce_corrp_tstat1.nii.gz
stats/fslvbm_var9_tfce_corrp_tstat1.nii.gz
```

y las imagenes y los reports de clusters por cada variable,

```
[osotolongo@brick03 vbmcomps]$ ls stats/clusters_var*
stats/clusters_var10_index.txt stats/clusters_var16_index.txt
stats/clusters_var27_index.txt
stats/clusters_var10.nii.gz    stats/clusters_var16.nii.gz
stats/clusters_var27.nii.gz
stats/clusters_var11_index.txt stats/clusters_var17_index.txt
stats/clusters_var28_index.txt
stats/clusters_var11.nii.gz    stats/clusters_var17.nii.gz
stats/clusters_var28.nii.gz
stats/clusters_var12_index.txt stats/clusters_var23_index.txt
stats/clusters_var29_index.txt
stats/clusters_var12.nii.gz    stats/clusters_var23.nii.gz
stats/clusters_var29.nii.gz
stats/clusters_var13_index.txt stats/clusters_var24_index.txt
stats/clusters_var30_index.txt
stats/clusters_var13.nii.gz    stats/clusters_var24.nii.gz
stats/clusters_var30.nii.gz
stats/clusters_var14_index.txt stats/clusters_var25_index.txt
stats/clusters_var31_index.txt
stats/clusters_var14.nii.gz    stats/clusters_var25.nii.gz
stats/clusters_var31.nii.gz
stats/clusters_var15_index.txt stats/clusters_var26_index.txt
stats/clusters_var9_index.txt
stats/clusters_var15.nii.gz    stats/clusters_var26.nii.gz
stats/clusters_var9.nii.gz
```

Basicamente solo queda inspeccionar los clusters

```
[osotolongo@brick03 vbmcomps]$ for x in stats/clusters_var*_index.txt; do
echo ${x}; cat ${x}; done
stats/clusters_var10_index.txt
Cluster Index   Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var11_index.txt
Cluster Index   Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
15 310    0.955    69    24    27    67.6    24.2    30.4
14 293    0.954    77    63    29    76.3    59.6    26.1
13 282    0.952    39    75    53    40.9    74.7    50.9
12 91    0.952    35    93    47    36.1    93.1    48.1
11 72    0.951    38    86    55    39.1    86.6    55.8
10 70    0.951    80    42    37    78.6    43.5    37.7
9  45    0.95    36    94    40    37    93.4    41.2
8  39    0.951    64    23    40    64.6    22.9    40.4
```

```
7  31    0.951    76    33    37    74.9    33.5    37.9
6  11    0.95    45    85    52    45.5    85.7    52.2
5  10    0.95    77    35    30    76.9    35.7    31
4  8    0.95    39    87    49    39.6    87    49.4
3  6    0.95    52    80    18    51.7    79.5    18.7
2  3    0.95    52    73    49    52    73.3    49.3
1  1    0.95    64    36    46    64    36    46
stats/clusters_var12_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var13_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var14_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
1  163251    0.997    20    35    16    44.4    51.2    37.3
stats/clusters_var15_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var16_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var17_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var23_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var24_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var25_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
1  137391    0.992    44    46    30    45    54.7    37.9
stats/clusters_var26_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
1  165183    0.998    43    61    44    45    52.8    37.8
stats/clusters_var27_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var28_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var29_index.txt
Cluster Index    Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
1  97509    0.991    45    24    54    44    50.8    41.8
stats/clusters_var30_index.txt
```

```
Cluster Index   Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var31_index.txt
Cluster Index   Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
stats/clusters_var9_index.txt
Cluster Index   Voxels    MAX    MAX X (vox)    MAX Y (vox)    MAX Z (vox)
COG X (vox)    COG Y (vox)    COG Z (vox)
```

 EH, algun resultado si que hay!!!!!

Plantillas con matchs en numero de sujetos

Para variables dicotomicas quiero hacer una plantilla con la misma cantidad de sujetos en abos valores. Ejemplo,

```
[osotolongo@brick03 vbmcomps]$ awk -F"," {'print $2","$6","$7","$8","$9'}
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,.*/Subject,Age
,Education,Gender,Var9/' | head
Subject,Age,Education,Gender,Var9
0001,71,8,0,0
0002,70,12,1,0
0003,70,8,0,0
0004,76,16,0,0
0005,68,20,1,0
0006,64,14,0,0
0007,59,19,1,1
0008,55,16,0,0
0009,67,16,0,0
[osotolongo@brick03 vbmcomps]$ awk -F"," {'print $2","$6","$7","$8","$9'}
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,.*/Subject,Age
,Education,Gender,Var9/' | tail -n +2 | awk -F"," {'if ($5==1) print'} | wc
-l
34
[osotolongo@brick03 vbmcomps]$ awk -F"," {'print $2","$6","$7","$8","$9'}
datacomb_freesurfer_neuro_v0_scdSUMCC_fixed.csv | sed
's/.*facehbi_//;s/SubjID,edat,Anyos_Escolaridad_FAC,Sex_1H_0M,.*/Subject,Age
,Education,Gender,Var9/' | tail -n +2 | awk -F"," {'if ($5==0) print'} | wc
-l
164
```

Aqui tengo que tomar todos los sujetos con *Var9==1* y escoger aleatoriamente *34* con *Var9==0*.

Empezar por aqui: https://pynative.com/python-random-sample/