ANEXOS

GLOSARIO

Avconv: Librería que permite la edición de video y audio.

Datasets: Estructura del HDF5 que contiene información, usualmente matrices de números.

Groups: Estructura del HDF5 que puede contener otros *groups* o *datasets.*

HDF5: 1. Hierarchical Data Format. Formato de archivo caracterizado por estructurar la información usando *groups* y *datasets*. 2. Librería que permite leer y crear archivos con formato HDF5.

Indels: Inserciones y deleciones.

Librería: Conjunto de scripts que componen un determinado programa.

Matplotlib: Libreria de Python para la creacion de distintos tipos de graficas en 2D y 3D.

NumPy: Numeric Python.Librería para Python escrita en C, C++ y Fortran que contiene diversas funciones para la creacion de matrices y cálculos matemáticos.

ACP: Principal Component Analysis. Es una técnica usada para reducir las dimensiones de un conjunto de datos, utilizada para facilitar la búsqueda de la causa de la variabilidad de los datos.

Python: Lenguaje de programación diseñado en 1991 por Guido Van Rossum.

Script: Archivo de texto plano que contiene las órdenes (código) para realizar alguna tarea. Pueden ser considerados mini programas.

SNP: Single Nucleotide Polymorphism. Cambio de un nucleótido en un punto del genoma. A veces los indels se engloba en este término.

Ubuntu: Sistema operativo libre basado en GNU/Linux.

Variables: En programación, un conjunto de caracteres (usualmente una palabra que recuerde su contenido) que contiene la información que se le haya asignado.

VCF: Variant Call Format. Formato de archivo que almacena información sobre SNP e indels de varias muestras de una determinada especie o variedad con respecto al genotipo de referencia.

FILTROS_ACP.PY

```
import h5py
import allel
import numpy as np

#For create the graphic
from draw_ACP import draw_ACP
import sklearn
from sklearn.cluster import KMeans




def open_HDF5(filename):
	h5_file = h5py.File(filename, 'r')
	return h5_file


def extract_genotype_data(h5_file):

	'''Transform the genotype data in h5_file to a GenotypeArray'''


	genotypes = h5_file['calldata']['genotype']
	#genotypes = h5_file['calldata']['GT']
	return allel.model.GenotypeArray(genotypes)




def filter_snps_by_maf(genotype_array, max_freq=1):

	'''Eliminates the SNP with a allelic frequencie bigger than max_freq'''

	af = genotype_array.count_alleles().to_frequencies()
	maf = np.amax(af, axis=1)
	is_polimorf = maf < max_freq
	return genotype_array[is_polimorf, :, :]


def calculate_total_alleles(genotype_array):

	'''With the genotype array, guess the ploidy and the number of samples to calculate'''
	'''how many alleles per SNP have the array'''

	g_ploidy = genotype_array.ploidy
```

```
    g_samples = genotype_array.n_samples
    return g_ploidy*g_samples


def filter_snps_by_missing_calls(genotype_array, num_al_snp, max_missing):

    '''Eliminate SNP whith less missing data than max_missing'''
    '''num_al_snp is the total number of alleles per SNP'''

    al = genotype_array.count_alleles()
    suma = np.sum(al, axis=1)
    min_all = num_al_snp-max_missing
    filt = suma > min_all

    return genotype_array[filt, :, :]


def filter_snps_by_min_calls(genotype_array, min_calls):

    '''Eliminates SNP whith less data than min_calls'''

    al = genotype_array.count_alleles()
    suma = np.sum(al, axis=1)
    filt = suma >= min_calls
    return genotype_array[filt, :, :]



def filter_gn(gn, num_al_snp):

    '''Eliminates the rows with only 2'''
    '''This filter is requiered if you have any row whith all 2, because ACP falls'''
    '''Works with ploidy==2'''
    suma = np.sum(gn, axis=1)
    div =suma/num_al_snp
    filt = div<1 #If all are 2, div will be 1.
    return gn[filt, :]



def filter_(genotype_array, filters=True, num_al_snp=None, max_missing=0,
min_calls=1):

    '''Uses at least maf filter'''
    '''If filters==2 filts with missing call, 3 with min calls, and 4 with both'''

    if filters:
    g_filt = filter_snps_by_maf(genotype_array)

    if filters==2:
```

```
        g_filt = filter_snps_by_missing_calls(g_filt, num_al_snp, max_missing)
        elif filters==3:
        g_filt = filter_snps_by_min_calls(g_filt, min_calls)
        elif filters==4:
        g_filt = filter_snps_by_missing_calls(g_filt, num_al_snp, max_missing)
        g_filt = filter_snps_by_min_calls(g_filt, min_calls)

        return g_filt




def obtain_ACP_points(ACP):
        '''Obtain axes X, Y, Z like the first, second and third components'''

        X = []
        Y = []
        Z = []
        valores=ACP[0] #ACP[0] are point data, ACP[1] is the object
        for i in range(len(valores)):
        X.append(ACP[0][i][0])
        Y.append(ACP[0][i][1])
        Z.append(ACP[0][i][2])
        #i is the sample number
        #The third number is the component


        return X, Y, Z

def calculate_Kmeans(gn, n_clusters):

        KM_cluster = KMeans(n_clusters=n_clusters)
        KM_fit = KM_cluster.fit(gn)
        centr = KM_fit.cluster_centers_
        X, Y, Z = centr[0], centr[1], centr[2]

        return X, Y, Z

def main(filename=None, filters=None, max_freq=1):

        filename = '/home/felipe/Documentos/HDF5/ril_call2d.HDF5'
        #filename = '/home/felipe/Documentos/HDF5/calldata_2d_apeki.HDF5'
        #filename = '/home/felipe/Documentos/HDF5/tom_call2d.HDF5'

        h5_file = open_HDF5(filename)
        genotype_array = extract_genotype_data(h5_file)
        num_al_snp = calculate_total_alleles(genotype_array)
        g_filt = filter_(genotype_array, filters=2, num_al_snp=num_al_snp,
max_missing=30)
```

```
        #g_filt = g_filt[:450000, :, :]
        gn = g_filt.to_n_alt()

        gn = filter_gn(gn, num_al_snp)


        #Deleting the variables with genotypes that we don't need, increases speed and
        #the max number of snp that can calculate the ACP function

        del(genotype_array)
        del(g_filt)


        ACP = allel.stats.decomposition.ACP(gn)


        X, Y, Z = obtain_ACP_points(ACP)

        #X, Y, Z = calculate_Kmeans(gn, n_clusters=5)

        return X, Y, Z

if __name__=='__main__':

        proj = main(filename = '/home/felipe/Documentos/HDF5/tom_call2d.HDF5')
        draw_ACP(proj)
```

DRAW_ACP.PY

```
from matplotlib.figure import Figure
from matplotlib.backends.backend_agg import FigureCanvasAgg as FigureCanvas
from mpl_toolkits.mplot3d import Axes3D
import os



def draw_ACP(projections, color='blue', dir_ ='.', opt_ax = 'off'):
	X, Y, Z = projections

	fig = Figure()

	axes = fig.add_subplot(111, projection='3d', xticks=[], yticks=[], zticks=[])
	axes.scatter3D(X, Y, Z, color=color)
	axes.axis(opt_ax)

	for i, angle in enumerate(range(0, 360, 1)):
		fname = 'ril_comp_%04d.png' % (i)
		fpath = os.path.join(dir_, fname)
		fhand = open(fpath, 'w')
		if not opt_ax=='off':
			axes.set_xlabel('componente 1')
			axes.set_ylabel('componente 2')
			axes.set_zlabel('componente 3')
		axes.view_init(azim=angle)
		canvas = FigureCanvas(fig)
		canvas.print_figure(fhand)
		fhand.close()
```

TIEMPOS FILTROS

				Numpy			

	t(s)	tiempo antes primer filtro	filtro maf=1	filtro maaf=1+ miss=100	filtro maaf=1+ miss=100+ min calls=1	filtro maaf=1+ miss=100+ min calls=1+gn	
	t1	0,94	0,745	0,745	0,691	0,794	
	t2	0,726	0,701	0,693	0,696	0,712	
ril	t3	0,743	0,684	0,703	0,703	0,708	
	t4	0,724	0,687	0,687	0,718	0,701	
	t5	0,733	0,679	0,7	0,703	0,704	
	tmedio	0,7732	0,6992	0,7056	0,7022	0,7238	
	t1	0,759	0,738	0,773	0,737	0,751	
	t2	0,73	0,733	0,734	0,732	0,748	
apeki	t3	0,723	0,701	0,738	0,745	0,757	
	t4	0,726	0,769	0,743	0,737	0,752	
	t5	0,751	0,711	0,732	0,789	0,763	
	tmedio	0,7378	0,7304	0,744	0,748	0,7542	
	t1	4,45	27,27	13,066	12,595	11,926	
	t2	6,224	24,103	14,646	12,134	12,23	
tomate	t3	4,49	17,549	13,726	12,966	11,972	
	t4	4,491	9,205	12,197	12,181	12,894	
	t5	4,416	9,032	12,149	11,882	12,167	
	tmedio	4,8142	17,4318	13,1568	12,3516	12,2378	

		For y np.vstack	
t(s)	af	af(sin np.vstack)	gn
t1	1,298	0,91	0,943
t2	0,925	0,894	0,912
t3	0,891	0,903	0,905
t4	0,903	0,91	0,91
t5	0,888	0,915	0,901
tmedio	0,981	0,9064	0,9142
t1	170,599	1,337	2,837
t2	166,062	1,339	2,955
t3	167,672	1,451	2,822
t4	167,939	1,329	2,959
t5	168,082	1,369	2,853
tmedio	168,0708	1,365	2,8852
t1	16min+(parado a mano (ctrlC) a 16minutos)	30,973	31,51
t2	90min+(parado a mano (ctrlC) a 90 minutos)	28,51	31,949
t3		25,253	30,676
t4		19,113	33,041
t5		19,404	34,977
tmedio		24,6506	32,4306

TIEMPOS POR MUESTRAS

100k SNP							
nºmuestras	50	100	150	200	250	300	348
t1(s)	9	17	26	32	43	51	62
t2(s)	9	17	26	32	43	51	62
t3(s)	9	18	26	32	44	51	63
t4(s)	9	17	26	32	43	51	62
t5(s)	9	19	26	32	44	52	62
tmedio	9	17,6	26	32	43,4	51,2	62,2
150k SNP							
nºmuestras	50	100	150	200	250	300	348
t1(s)	11	24	36	45	61	71	87
t2(s)	11	24	36	45	61	71	87
t3(s)	11	24	36	45	60	71	86
t4(s)	11	23	36	45	61	71	86
t5(s)	11	24	36	45	61	71	86
tmedio	11	23,8	36	45	60,8	71	86,4

TIEMPOS NPY HDF5

		ril.vcf	
tamaño comprimido	504,5kB	nºmuestras	153
tamaño descomprimido	2,5MB	nºSNP	943
		vcf2npy (s)	vcfnp2hdf5 (s)
	t1	1	1,335
	t2	0,144	0,774
variants	t3	0,148	0,704
	t4	0,154	0,701
	t5	0,163	0,773
	tamaño	223 K	477K
	tamaño comprimido(tar.gz)	74K	151K
	tamaño comprimido (gz)	75K	152K
	t1	3,581	0,707
	t2	0,153	0,721
calldata	t3	0,149	0,729
	t4	0,148	0,744
	t5	0,143	0,73
	tamaño	6,4M	1,5M
	tamaño comprimido(tar.gz)	438K	1,5M
	tamaño comprimido (gz)	437K	1,5M
	t1	3,445	0,707
	t2	0,183	0,701
calldata_2d	t3	0,153	0,719
	t4	0,15	0,759
	t5	0,152	0,756
	tamaño	6,4M	1,5M
	tamaño comprimido(tar.gz)	437K	1,3M
	tamaño comprimido (gz)	437K	1,3M

	suma del tamaño de los 3 ficheros	13M	3,5M
	tamaño de los 3 ficheros comprimidos(tar.gz)	948K	2,9M
	suma del tamaño de los tres ficheros compromidos (gz)	948K	2,9M

		tomate_apeki	
tamaño comprimido	9,7MB	nºmuestras	12
tamaño descomprimido	35,6MB	nºSNP	41682
		vcf2npy (s)	vcfnp2hdf5 (s)
	t1	6,528	1,211
	t2	0,15	1,221
variants	t3	0,151	1,232
	t4	0,149	1,243
	t5	0,146	1,271
	tamaño	9,6M	4,9M
	tamaño comprimido(tar.gz)	2,7M	4,5M
	tamaño comprimido (gz)	2,7M	4,5M
	t1	14,3	2,067
	t2	0,152	1,841
calldata	t3	0,148	1,915
	t4	0,155	1,875
	t5	0,15	1,912
	tamaño	22M	12M
	tamaño comprimido(tar.gz)	4,7M	12M
	tamaño comprimido (gz)	4,7M	12M
	t1	13,9	2,165
	t2	0,165	2,079
calldata_2d	t3	0,151	2,037
	t4	0,151	2,17
	t5	0,154	2,043
	tamaño	22M	11M

	tamaño comprimido(tar.gz)	4,7M	11M
	tamaño comprimido (gz)	4,7M	11M
	suma del tamaño de los 3 ficheros	53,6M	27,9M
	tamaño de los 3 ficheros comprimidos(tar.gz)	12M	27M
	suma del tamaño de los tres ficheros compromidos (gz)	12M	27M

		tomatos	
tamaño comprimido	1,7GB	nºmuestras	348
tamaño descomprimido	7,1GB	nºSNP	868978
		vcf2npy (s)	vcfnp2hdf5 (s)
	t1	4m51	19,292
	t2	3,115s	19,302
variants	t3	0,183s	19.281
	t4	0,144s	19,391
	t5	0,149s	19,386
	tamaño	200M	124M
	tamaño comprimido(tar.gz)	80M	123M
	tamaño comprimido (gz)	81M	123M
	t1	55m23s	45,231
	t2	1,426	48,694
calldata (solo GT)	t3	0,152	42,106
	t4	0,142	38,956
	t5	0,149	39,118
	tamaño	577M	46M
	tamaño comprimido(tar.gz)	23M	43M

	tamaño comprimido (gz)	23M	43M
	t1	53m56s	28,458
	t2	1,476	35,977
calldata_2d(solo GT)	t3	0,151	22,815
	t4	0,16	19,376
	t5	0,153	18,578
	tamaño	577M	46M
	tamaño comprimido(tar.gz)	23M	44M
	tamaño comprimido (gz)	23M	44M
	suma del tamaño de los 3 ficheros	1354M	216M
	tamaño de los 3 ficheros comprimidos(tar.gz)	125M	209M
	suma del tamaño de los tres ficheros compromidos (gz)	126M	209M

VELOCIDADES

SNP	10000	50000	80000	100000	150000	200000	300000	350000	400000	450000	500000	550000	600000
t1(s)	20	36	52	62	87	111	160	188	274	327	541	662	699
t2(s)	17	37	52	62	87	111	160	192	260	303	550	752	689
t3(s)	17	37	53	62	86	111	178	189	256	313	526	726	765
t4(s)	17	36	52	63	86	112	166	191	262	334	551	841	909
t5(s)	17	38	53	62	86	111	157	189	277	313	590	620	837
tmedio (s)	17,6	36,8	52,4	62,2	86,4	111,2	164,2	189,8	265,8	318	551,6	720,2	779,8

Sin usar del				
SNP	100000	200000	300000	400000
t1(s)	68	114	226	360
t2(s)	61	288	250	421
t3(s)	62	190	246	288
t4(s)	61	142	237	316
t5(s)	61	121	233	314
tmedio(s)	62,6	171	238,4	339,8

TIEMPOS INOUT

Ril				
	CHUNK=200	CHUNK=1000	CHUNK=200	CHUNK=1000
shuffle y fletcher	TRUE	TRUE	FALSE	FALSE
t1(s)	8,6	8,5	8,6	8,6
t2(s)	8,6	8,6	8,5	8,6
t3(s)	8,5	8,5	8,6	8,5
t4(s)	8,5	8,6	8,5	8,6
t5(s)	8,6	8,8	8,6	8,5
tamaño hdf5	517K	511K	483K	485K

Apeki					
	CHUNK=200	CHUNK=10000	CHUNK=25000	CHUNK=200	CHUNK=25000
shuffle y fletcher	TRUE	TRUE	TRUE	FALSE	FALSE
t1(s)	357	353	369	353	374
t2(s)	361	359	364	350	357
t3(s)	346	355	367	375	363
t4(s)	348	362	368	390*	359
t5(s)	347	357	373	399*	363
tamaño hdf5	4.3M	3.4M	3.3M	4.2M	3.4M
	*Ejecutadas a la vez				

tomatos				

kept_field=['GT']				kept_field=None
	CHUNK=200	CHUNK=200		CHUNK=200
shuffle y fletcher	TRUE	FALSE		TRUE
t1(s)	1813	1887		10613
t2(s)	1907	1884		11450*
t3(s)	1873	1891		11053*
t4(s)	1886	1872		11393*
t5(s)	1914	2068*		11229*
tamaño hdf5	38M	38M		
	*Con Google Drive abierto			

VIDEOS

https://www.dropbox.com/s/fjw0pjup9edpbue/Videos.zip?dl=0