

Apéndice A

Rutinas en Matlab.

A.1. Espectro de energía cinética turbulenta.

Se muestra a continuación la rutina realizada en Matlab para la obtención del espectro de energía. Primero el programa transforma los valores de tensión, procedentes del anemómetro, en velocidad mediante la curva de calibración de la sonda de hilo caliente. Luego realiza la transformada discreta de Fourier utilizando el algoritmo para la transformada rápida fft (Fast Fourier Transform) que incorpora Matlab.

```
%PROGRAMA PARA OBTENER EL ESPECTRO DE ENERGÍA.  
%Datos iniciales.  
 $n = 10000$ ; %Tamaño de muestras  
 $fs = 9000$ ; Frecuencia de muestreo en Hz  
 $Ts = 1/fs$ ;  
 $T = n/fs$ ; %Tiempo de muestreo  
 $x = \text{zeros}(n, 1)$ ;  $F = \text{zeros}(n, 1)$ ;  
%Vector de frecuencias (sólo las  $n/2$  primeras)  
 $Fe = fs * (0 : n/2 + 1)/n$ ;  
 $F = Fe(2 : n/2 + 1)$ ;  
%Carga de archivos de datos.  
load c:s1.txt; load c:s2.txt; load c:s3.txt; load c:s4.txt; load c:s5.txt; load c:s6.txt;  
load c:s7.txt; load c:s8.txt; load c:s9.txt; load c:s10.txt; load c:s11.txt; load c:s12.txt;  
load c:s13.txt; load c:s14.txt; load c:s15.txt; load c:s16.txt; load c:s17.txt; load  
c:s18.txt; load c:s19.txt; load c:s20.txt; load c:s21.txt; load c:s22.txt; load c:s23.txt;
```

```

load c:s24.txt; load c:s25.txt; load c:s26.txt; load c:s27.txt; load c:s28.txt; load
c:s29.txt; load c:s30.txt; load c:s31.txt; load c:s32.txt; load c:s33.txt; load c:s34.txt;
load c:s35.txt; load c:s36.txt; load c:s37.txt; load c:s38.txt; load c:s39.txt; load
c:s40.txt; load c:s41.txt; load c:s42.txt; load c:s43.txt; load c:s44.txt; load c:s45.txt;
load c:s46.txt; load c:s47.txt; load c:s48.txt; load c:s49.txt; load c:s50.txt;
e = [s1s2s3s4s5s6s7s8s9s10s11s12s13s14s15s16s17s18s19s20s21s22s23s24s25
s26s27s28s29s30s31s32s33s34s35s36s37s38s39s40s41s42s43s44s45s46s47s48s49s50];
%Aplicamos curva de calibración
u = e.^2 * 0,1073 - e. * 2,6878 + 19,576;
E = zeros(n/2 + 1, 1);
for i = 1 : 50;
x = u(1 : n, i);
%Transformada discreta de Fourier de u
X = fft(x);
W = X(1 : n/2 + 1)Ts;
Y = abs(W);
Z = Y.^2;
E = E + 2/T * Z; Espectro
end
Emedio = E(2 : n/2 + 1)/50;
U = u;
Ufinal = zeros(500000, 1);
for i = 1 : 500000;
Ufinal(i) = U(i);
end;
Umedia = mean(Ufinal);
K1 = 2 * pi * F/Umedia;
%Representación de  $E_{11}$  junto a  $K_1^{-5/3}$ 
recta = K1.^(-5/3);
loglog(F, Emedio, F, recta)

```

A.2. Obtención de ϵ , Re_λ y η

El siguiente programa hace un estudio espectral y obtiene las magnitudes necesarias para caracterizar la turbulencia. El programa parte de los valores obtenidos en el programa para la obtención del espectro de energía. En este programa se ha hecho uso de la herramienta cftool de Matlab, que permite realizar operaciones numéricas

como interpolaciones polinómicas, por splines, además de integrar.

```
%ESTUDIO ESPECTRAL.
%Previamente se habrá ejecutado el programa para la obtención del espectro de
energía.
%Obtenemos  $E_{11}(K_1)$  a partir de  $E_{11}(f)$ 
EK = Umedia * Emedio / (2 * pi);
%Obtenemos  $K_1^2 * E_{11}(K_1)$ 
K1 = K1';
K1cuadrado = K1;
K1cuadrado = K1cuadrado.^2;
EKcuadrado = K1cuadrado.*EK;
%Obtención de  $T_u$ 
Desvtipica = std(Ufinal);
Tu = Desvtipica/Umedia;
%Obtención de  $\epsilon$ .
%Extrapolación
%Frecuencia en la que corto el espectro.
frec2 = input(' Teclea frecuencia de corte');
%Frecuencia a la que empiezo a interpolar.
frec1 = input(' Teclea frecuencia para comenzar la recta');
kmax = round(frec2 * fs/n);
kmin = round(frec1 * fs/n);
intervalok = kmax - kmin;
for i = 1 : intervalok;
K1disipalog(i) = log10(K1(i + kmin - 1));
Edisipalog(i) = log10(EKcuadrado(i + kmin - 1));
end
K1disipalog = Fdisipalog';
Edisipalog = Edisipalog';
cftool(K1disipalog, Edisipalog);
recta2 = fit(Fdisipalog, Edisipalog, 'poly1');
Kdefi = (fs/n : fs/n : 10000);
Edefi = zeros(1, 10000/(fs/n));
for i = 1 : kmax
Edefi(i) = Emedio(i);
end
for i = kmax : 50000/(fs/n)
Edefi(i) = 10^recta2(log10(Kdefi(i)));
end
loglog(K1defi, Edefi)
```

```

K1defi = 2 * pi * Fdefi / Umedia;
K1defic cuadrado = K1defi.^2;
EKdefi = Umedia * Edefi / (2 * pi);
EKcuadradodefi = K1defic cuadrado * EKdefi;
K1defi = K1defi';
EKcuadradodefi = EKcuadradodefi';
EKdefi = EKdefi';
%Para la integración numérica aproximo la función mediante spline
cftool(K1defi, EKcuadradodefi);
Curvaajustada = fit(K1defi, EKcuadradodefi, 'cubicspline');
plot(K1defi, Curvaajustada(K1defi));
%Integración
integral = integrate(Curvaajustada, 2 * pi * 1000 / Umedia, fs / n);
epsilon = 15 * 1,5 * 10(-5) * integral
kolmogorov = ((1,5 * 10(-5))3 / epsilon)(1/4)
uprima = std(Ufinal)
landa = (15 * (1,5 * 10(-5)) / epsilon)0,5 * uprima
Relanda = uprima * landa / (1,5 * 10(-5))
Umedia
Tu

```