Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

Máster en Ingeniería Industrial

MÁSTER EN INGENIERÍA INDUSTRIAL

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA

PRE-EVALUACIÓN DEL

COMPORTAMIENTO DINÁMICO DE ESTRUCTURAS ESBELTAS.

ANEXOS:

CÓDIGO

Departamento: Construcciones Arquitectónicas, Ingeniería del Terreno y Mecánica de Medios Continuos y Teoría de Estructuras

Autor: D. Juan Miguel Alario Bercianos

Tutor: D. Antonio Foces Mediavilla

Valladolid, Junio de 2018

ÍNDICE

CÓDIGO ANDROID

CLASES JAVA

MAIN ACTIVITY

```
1  package com.example.jalab.pruebaaccelgraphview;
2
3  import android.app.ProgressDialog;
4  import android.bluetooth.BluetoothAdapter;
5  import android.bluetooth.BluetoothDevice;
6  import android.bluetooth.BluetoothSocket;
7  import android.content.Context;
8  import android.content.DialogInterface;
9  import android.graphics.Color;
10 import android.hardware.Sensor;
11 import android.hardware.SensorEvent;
12 import android.hardware.SensorEventListener;
13 import android.hardware.SensorManager;
14 import android.os.Environment;
15 import android.os.Handler;
16 import android.support.v7.app.AlertDialog;
17 import android.support.v7.app.AppCompatActivity;
18 import android.os.Bundle;
19 import android.util.DisplayMetrics;
20 import android.util.Log;
21 import android.view.Display;
22 import android.view.LayoutInflater;
23 import android.view.View;
24 import android.view.Window;
25 import android.view.WindowManager;
26 import android.widget.AdapterView;
27 import android.widget.ArrayAdapter;
28 import android.widget.Button;
29 import android.widget.CheckBox;
30 import android.widget.EditText;
31 import android.widget.LinearLayout;
32 import android.widget.ListView;
33 import android.widget.Spinner;
34 import android.widget.TextView;
35 import android.widget.Toast;
36 import android.widget.ToggleButton;
37
38
39 import com.jjoe64.graphview.DefaultLabelFormatter;
40 import com.jjoe64.graphview.GraphView;
41
42 import com.jjoe64.graphview.series.DataPoint;
43 import com.jjoe64.graphview.series.DataPointInterface;
44 import com.jjoe64.graphview.series.LineGraphSeries;
45 import com.jjoe64.graphview.series.OnDataPointTapListener;
46 import com.jjoe64.graphview.series.PointsGraphSeries;
47 import com.jjoe64.graphview.series.Series;
48
49
50 import java.io.BufferedReader;
51 import java.io.DataInputStream;
52 import java.io.File;
53 import java.io.FileInputStream;
54 import java.io.FileOutputStream;
55 import java.io.IOException;
56 import java.io.InputStream;
57 import java.io.InputStreamReader;
58 import java.io.OutputStream;
59 import java.text.NumberFormat;
60 import java.text.SimpleDateFormat;
61 import java.util.ArrayList;
62 import java.util.Calendar;
63 import java.util.List;
64 import java.util.StringTokenizer;
65 import java.util.UUID;
66
67 import uk.me.berndporr.iirj.Butterworth;
68
69
70 public class MainActivity extends AppCompatActivity implements SensorEventListener{
71
72     // Crea instancia del sensor, probee varios métodos
73     // de acceso y listado de sensores
74     private SensorManager mSensorManager;
75     private Sensor mAccelerometer;
76     private int[] sensorType;
77     private boolean mInitialized;
```

Main Activity

```
78      private float x,y,z;
79      private CheckBox xRec, yRec, zRec, bf, g;
80      // Instancias de los objetos del Acelerómetro
81      private AccelColection accelColection, accelLoaded, accelFFTed, accelButted, accelRMSTrend,
82                      accelDetrended;
83      private AccelColection resampled;
84      private String acelerationaLoaded=" ", accelerationStringRecording ,
85              acelerationLoadedName=" ", acelerationaLoadedAux=" ";
86      private  Spinner frecuencies;
87      // Botones
88      private ToggleButton saveToggleButton;
89      private ToggleButton startToggleButton;
90      private Button loadButton, saveButton, fftButton, fftSaveButton, butterButton, butterSetButton,
91              butterSaveButton, sfSelecctor, arSelecctor, detrendButton, detrendSaveButton,
92              rsmButton, rsmSetButton, rsmSaveButton;
93
94      // Cosas para la progressbar
95      private ProgressDialog progressBar;
96      private int progressBarStatus = 0;
97      private Handler progressBarHandler = new Handler();
98      private int contRead=0, contiG=0, contRes=0, marcRes=0, contSave=0;
99      // butterwhorth filter:
100     private Butterworth butterworthX,butterworthY,butterworthZ;
101     //RMS
102     private double[] RMSX, RMS_T, RMSY, RMSZ;
103     private double fs, T2=0.25, win=0.5, meanX, meanY, meanZ, MTVVX=0, MTVVY=0, MTVVZ=0,
104             MTVVX_T=0, MTVVY_T=0, MTVVZ_T=0;
105     private int Ndat, Npts, N=0, contRms;
106     private int[] posi;
107
108     // Frecuencia de muestreo:
109     private int frecMuest=2;
110     // butterworth filter:
111     private double cutoff=0;
112     private int order=3;
113     // Timer:
114     private long initialTime=0, stopTime, initialRecordedTime;
115
116     private final Handler mHandler = new Handler(); // Handler para los timmers
117     // Timers
118     private Runnable mTimer1;
119     // Gráficas
120     private LineGraphSeries<DataPoint> mSeries1;
121     private LineGraphSeries<DataPoint> mSeries2;
122     private LineGraphSeries<DataPoint> mSeries3;
123     private double graph1LastXValue = 0d;
124     private GraphView graph1;
125     private boolean compCharged=false;
126
127     private LineGraphSeries<DataPoint> mSeries1_2;
128     private LineGraphSeries<DataPoint> mSeries2_2;
129     private LineGraphSeries<DataPoint> mSeries3_2;
130     private double graph1LastXValue_2 = 0d;
131     private GraphView graph1_2;
132     private TextView samplingGraph;
133
134     private LineGraphSeries<DataPoint> mSeries1Loaded;
135     private LineGraphSeries<DataPoint> mSeries2Loaded;
136     private LineGraphSeries<DataPoint> mSeries3Loaded;
137     private double graph1LastXValueLoaded = 0d; //Seguramente no haga falta
138     private PointsGraphSeries<DataPoint> mSeries1LoadedMTVVX;
139     private PointsGraphSeries<DataPoint> mSeries1LoadedMTVVY;
140     private PointsGraphSeries<DataPoint> mSeries1LoadedMTVVZ;
141
142     private TextView loadedData;
143     private SimpleDateFormat df;
144
145     // Para guardar en .txt:
146     private String filename = "SampleFile";
147     private String filepath = "MyFileStorage";
148     private File myExternalFile;
149     private String myData = "";
150     private boolean saveable;
151     private ListView savedListView;
152     private ArrayAdapter savedListViewAdpter=null;
153
154     // Cosas Bluetooth
```

Main Activity

```
155     private static final String TAG = "Jon";
156     private BluetoothAdapter mBluetoothAdapter = null;
157     private BluetoothSocket btSocket = null;
158     private OutputStream outStream = null;
159     private static String address = "98:D3:31:60:1E:37";
160     private static final UUID MY_UUID = UUID
161             .fromString("00001101-0000-1000-8000-00805F9B34FB");
162     private InputStream inStream = null;
163     Handler handler = new Handler();
164     byte delimiter = 10;
165     boolean stopWorker = false;
166     int readBufferPosition = 0;
167     byte[] readBuffer = new byte[1024];
168     private boolean ard=false;
169 // Parámetros para calcular la media móvil desde el
170     int zmean[]={0,0,0};
171     int ymean[]={0,0,0};
172     int xmean[]={0,0,0};
173     int sumZ, sumY, sumX;
174
175     @Override
176     protected void onCreate(Bundle savedInstanceState) {
177         super.onCreate(savedInstanceState);
178         setContentView(R.layout.activity_main);
179
180         loadedData=(TextView) findViewById(R.id.textView2);
181         // Selectores ejes:
182         xRec=(CheckBox) findViewById(R.id.xCheckBox);
183         yRec=(CheckBox) findViewById(R.id.yCheckBox);
184         zRec=(CheckBox) findViewById(R.id.zCheckBox);
185         // Selector filtro:
186         bf=(CheckBox)findViewById(R.id.butterFilterCheckButton) ;
187         bf.setEnabled(false);
188         // Date Format:
189         df= new SimpleDateFormat("yyyy-MM-dd HH.mm.ss");
190         //Filtro:--> lo dejo creado y lo defino en startToogleButton
191         butterworthX = new Butterworth();
192         butterworthY = new Butterworth();
193         butterworthZ = new Butterworth();
194         // Toggle button para guardar:
195         saveToggleButton=(ToggleButton) findViewById(R.id.saveToggleButton);
196         saveToggleButton.setText("SAVE");
197         saveToggleButton.setTextOff("SAVE");
198         saveToggleButton.setTextOn("SAVING");
199         saveToggleButton.setEnabled(false);
200         // Toggle button para comenzar:
201         startToggleButton=(ToggleButton) findViewById(R.id.startToggleButton);
202         startToggleButton.setText("START");
203         startToggleButton.setTextOff("START");
204         startToggleButton.setTextOn("STOP");
205         // .txt guardados:
206         savedListView=(ListView) findViewById(R.id.savedListView);
207         savedFiles();
208         // Inicializo y declaro Sensores
209         mInitialized = false;
210         mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
211         sensorType=new int[2];
212         sensorType[0]=Sensor.TYPE_LINEAR_ACCELERATION;
213         sensorType[1]=Sensor.TYPE_ACCELEROMETER;
214         mAccelerometer = mSensorManager.getDefaultSensor(sensorType[0]);
215         mSensorManager.registerListener(this,mAccelerometer,SensorManager.SENSOR_DELAY_FASTEST);
216         g=(CheckBox) findViewById(R.id.gCheckBox);
217         g.setOnClickListener(new View.OnClickListener() {
218             @Override
219             public void onClick(View view) {
220                 changgSFSensor();
221             }
222         });
223         // Cosas de gráficas
224         graph1 = (GraphView) findViewById(R.id.graph);
225         mSeries1 = new LineGraphSeries<>();
226         mSeries2 = new LineGraphSeries<>();
227         mSeries3 = new LineGraphSeries<>();
228         mSeries1.setColor(Color.parseColor("#182ce6"));
229         mSeries2.setColor(Color.parseColor("#de1a13"));
230         mSeries3.setColor(Color.parseColor("#000000"));
231
```

Main Activity

```
232         graph1_2 = (GraphView) findViewById(R.id.graph2);
233         mSeries1_2 = new LineGraphSeries<>();
234         mSeries2_2 = new LineGraphSeries<>();
235         mSeries3_2 = new LineGraphSeries<>();
236         mSeries1_2.setColor(Color.parseColor("#182ce6"));
237         mSeries2_2.setColor(Color.parseColor("#de1a13"));
238         mSeries3_2.setColor(Color.parseColor("#000000"));
239
240         // En función de la frecuencia seleccionada, cargo unos gráficos u otros:
241         if(frecMuest!=2) {
242             putAxisGraph2();
243         }
244         if(frecMuest==2) {
245             putAxisGraph1_2();
246         }
247         // Propiedades des los gráficos:
248         graph1_2.getViewport().setXAxisBoundsManual(true);
249         graph1_2.getViewport().setMinX(0);
250         graph1_2.getViewport().setMaxX(1800);
251         graph1_2.getGridLabelRenderer().setHorizontalLabelsVisible(false);
252         samplingGraph=(TextView) findViewById(R.id.textViewSamplingGraph);
253
254         // Selector de frecuencias, mediante un Spinner (desplegable con opciones):
255         frecuencies =(Spinner) findViewById(R.id.frecuenciesSelector);
256         ArrayAdapter<CharSequence> adapterFrecc = ArrayAdapter.createFromResource(this,
257                 R.array.frecuencies, android.R.layout.simple_spinner_item);
258         adapterFrecc.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
259         frecuencies.setAdapter(adapterFrecc);
260         frecuencies.setOnItemSelectedListener(
261                 new AdapterView.OnItemSelectedListener() {
262                     @Override
263                     public void onItemSelected(AdapterView<?> adapterView,
264                                                View view, int i, long l) {
265                         String item=(String) adapterView.getItemAtPosition(i);
266                         if(item.equals("Max")) {
267                             // Que la frecuencia Máxima sea 2 es a efectos de código, en verdad no
268                             // va a 2, va a full power, que es aprox. 4,9 ms== 204.1 hz
269                             frecMuest=2;
270                             bf.setEnabled(false);
271                             bf.setChecked(false);
272                             putAxisGraph1_2();
273                         }
274                         else {
275                             putAxisGraph2();
276                             bf.setEnabled(true);
277                             switch(item){
278                                 case "5 ms":
279                                     frecMuest = 5;
280                                     break;
281                                 case "10 ms":
282                                     frecMuest = 10;
283                                     break;
284                                 case "50 ms":
285                                     frecMuest = 50;
286                                     break;
287                                 case "75 ms":
288                                     frecMuest = 75;
289                                     break;
290                                 case "100 ms":
291                                     frecMuest = 100;
292                                     break;
293                             }
294                         }
295
296                         samplingGraph.setText("Sampling at "+ adapterView.getItemAtPosition(i));
297
298                     }
299
300                     @Override
301                     public void onNothingSelected(AdapterView<?> adapterView) {
302
303                     }
304                 }
305         );
306         // Radio buttons para seleccionar ejes:
307         xRec.setOnClickListener(new View.OnClickListener() {
308             @Override
```

Main Activity

```
309             public void onClick(View view) {
310                 if(mSeries1Loaded!=null && mSeries2Loaded!=null && mSeries3Loaded!=null)
311                     putAxisGraph1();
312                 if(startToggleButton.isChecked()) {
313                     if (frecMuest == 2) {
314                         putAxisGraph1_2();
315                     } else {
316                         putAxisGraph2();
317                     }
318                 }
319             }
320         });
321         yRec.setOnClickListener(new View.OnClickListener() {
322             @Override
323             public void onClick(View view) {
324                 if(mSeries1Loaded!=null && mSeries2Loaded!=null && mSeries3Loaded!=null)
325                     putAxisGraph1();
326                 if(startToggleButton.isChecked()) {
327                     if (frecMuest == 2) {
328                         putAxisGraph1_2();
329                     } else {
330                         putAxisGraph2();
331                     }
332                 }
333             }
334         });
335         zRec.setOnClickListener(new View.OnClickListener() {
336             @Override
337             public void onClick(View view) {
338                 if(mSeries1Loaded!=null && mSeries2Loaded!=null && mSeries3Loaded!=null)
339                     putAxisGraph1();
340                 if(startToggleButton.isChecked()) {
341                     if (frecMuest == 2) {
342                         putAxisGraph1_2();
343                     } else {
344                         putAxisGraph2();
345                     }
346                 }
347             }
348         });
349         // Botón para la guardar la FFT:
350         fftSaveButton=(Button) findViewById(R.id.saveFFTButton);
351         fftSaveButton.setEnabled(false);
352         fftSaveButton.setOnClickListener(new View.OnClickListener() {
353             @Override
354             public void onClick(View view) {
355                 fftSaveButton.setEnabled(false);
356                 saveDataFile(accelFFTed.getStringAccel(),accelFFTed.getName());
357                 savedFiles();
358             }
359         });
360         // Boton para hacer la FFT:
361         fftButton=(Button)findViewById(R.id.fftButton);
362         fftButton.setEnabled(false);
363         fftButton.setOnClickListener(new View.OnClickListener() {
364             @Override
365             public void onClick(View view) {
366
367                 if (accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
368                         accelLoaded.get(2).getTime() - accelLoaded.get(1).getTime() &&
369                         accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
370                                 accelLoaded.get(3).getTime() - accelLoaded.get(2).getTime()
371                         && accelLoaded.getName().indexOf("FFT")==-1) {
372                     //Toast.makeText(getApplicationContext(), "Resampled", Toast.LENGTH_SHORT).show
    ();
373                     fftProces();
374                     graph1.removeAllSeries();
375                     chargeGraphLoaded(accelFFTed,2);
376                     fftButton.setEnabled(false);
377                     fftSaveButton.setEnabled(true);
378                 }
379                 else {
380                     Toast.makeText(getApplicationContext(),"NOT POSSIBLE",Toast.LENGTH_SHORT)
381                             .show();
382                     fftButton.setEnabled(false);
383                 }
384
```

Main Activity

```
385
386             }
387         });
388         // Elementos para el filtro butterworth:
389         butterButton=(Button) findViewById(R.id.butterButton);
390         butterSetButton=(Button) findViewById(R.id.butterSetButton);
391         butterButton.setEnabled(false);
392         butterSetButton.setOnClickListener(new View.OnClickListener() {
393             @Override
394             public void onClick(View view) {
395                 butterInputDialog();
396             }
397         });
398         butterSaveButton=(Button) findViewById(R.id.butterSaveButton);
399         butterButton.setOnClickListener(new View.OnClickListener() {
400             @Override
401             public void onClick(View view) {
402                 // Primero comprueba sí el registro está euiespaciado o no (resampled)
403                 if (accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
404                         accelLoaded.get(2).getTime() - accelLoaded.get(1).getTime()
405                         && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
406                         accelLoaded.get(3).getTime() - accelLoaded.get(2).getTime()
407                         && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
408                         accelLoaded.get(4).getTime() - accelLoaded.get(3).getTime()
409                         && accelLoaded.getName().indexOf("BUTTER") == -1 &&
410                         accelLoaded.getName().indexOf("FFT") == -1) {
411                     // Cómo para este filtro la frecuencia de corte no puede ser mayor a la mitad de
412                     // la frecuencia de muestreo (por las propiedades del filtro):
413                     if(cutoff>(1000 / (2*(accelLoaded.get(1).getTime()
414                             - accelLoaded.get(0).getTime())))){
415                         Toast.makeText(getApplicationContext(),"NOT POSSIBLE: Cutoff> fsampling/2"
416                                 ,Toast.LENGTH_LONG).show();
417                     }
418                     else{
419
420                         butterButton.setEnabled(false);
421                         butterSaveButton.setEnabled(true);
422                         if(cutoff==0) cutoff=30; // frecuencia de corte por defecto:
423                         // Establezco los filtros para el caso de filtrado en tiempo real:
424                         butterworthX.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
425                                 - accelLoaded.get(0).getTime())), cutoff);
426                         butterworthY.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
427                                 - accelLoaded.get(0).getTime())), cutoff);
428                         butterworthZ.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
429                                 - accelLoaded.get(0).getTime())), cutoff);
430                         // Construyo la collección acelerción con la acelaración filtrada
431                         accelButted = butterEx();
432                         accelLoaded=accelButted;
433
434                     }
435
436                 }
437                 else{
438                     Toast.makeText(getApplicationContext(),"NOT POSSIBLE",Toast.LENGTH_SHORT)
439                             .show();
440                     butterButton.setEnabled(false);
441                 }
442             }
443         });
444         // Guardado de el registro filtrado:
445         butterSaveButton.setEnabled(false);
446         butterSaveButton.setOnClickListener(new View.OnClickListener() {
447             @Override
448             public void onClick(View view) {
449                 saveDataFile(accelButted.getStringAccel(),accelButted.getName());
450                 savedFiles();
451                 butterSaveButton.setEnabled(false);
452             }
453         });
454         // Botón para guardar el archivo:
455         saveButton=(Button) findViewById(R.id.saveButton);
456         saveButton.setEnabled(false);
457         saveButton.setOnClickListener(new View.OnClickListener() {
458             @Override
459             public void onClick(View view) {
460                 saveDataFile(resampled.getStringAccel());
461                 saveButton.setEnabled(false);
```

Main Activity

```
462                 savedFiles();
463             }
464         });
465         // Botones (toggle button) para iniciar/guardar registro
466         saveToggleButton.setOnClickListener(new View.OnClickListener() {
467             @Override
468             public void onClick(View view) {
469                 // Inicio del guardado:
470                 if(saveToggleButton.isChecked()){
471                     marcRes=0;
472                     contRes=0;
473                     contSave=0;
474                     accelColection=new AccelColection(10000);
475                     initialRecordedTime=System.currentTimeMillis();
476                     accelerationStringRecording="\t"+"Time"+"\t"+"X"+"\t"+"Y"+"\t"+"Z"+"\n";
477                     frecuencies.setEnabled(false);
478                     startToggleButton.setEnabled(false);
479                     bf.setEnabled(false);
480                 }
481                 // Fin más guardado:
482                 if(!saveToggleButton.isChecked()){
483                     frecuencies.setEnabled(true);
484                     startToggleButton.setEnabled(true);
485                     bf.setEnabled(true);
486                     // En función de la frecuencia de muestro (máxima u otra) útilizo un método
487                     // u otro:
488                     // Para frecuencia máxima, guardo el archivot al cual:
489                     if(frecMuest==2)saveDataFile(accelColection.getStringAccel());
490                     // En el resto de caso, resampleo y habilito la opción de guardar:
491                     if(frecMuest!=2){
492
493                         progSave(view);
494                         saveButton.setEnabled(true);
495
496                     }
497                     // recargo los archivos guardados/almacenados:
498                     savedFiles();
499                 }
500             }
501         });
502         //  Comienza el registro en tiempo real:
503         startToggleButton.setOnClickListener(new View.OnClickListener() {
504             @Override
505             public void onClick(View view) {
506                 // Habilito el filtro si corresponde:
507                 if(bf.isChecked()){
508                     if (cutoff==0) cutoff=30;
509                     butterworthX.lowPass(order, (1000/(frecMuest)) , cutoff);
510                     butterworthY.lowPass(order, (1000/(frecMuest)) , cutoff);
511                     butterworthZ.lowPass(order, (1000/(frecMuest)) , cutoff);
512                 }
513                 bf.setEnabled(false);
514                 // Inicio el registro en tiempo real:
515                 if(startToggleButton.isChecked()){
516                     if(initialTime==0) initialTime = System.currentTimeMillis();
517                     if(initialTime!=0) initialTime=initialTime+System.currentTimeMillis()-stopTime;
518                     timer(true);
519                     butterSetButton.setEnabled(false);
520                     saveToggleButton.setEnabled(true);
521                     g.setEnabled(false);
522                     if(sfSelecctor.isEnabled()) {
523                         ard=true;
524                         stopWorker=false;
525                     }
526                     if(!sfSelecctor.isEnabled()) ard=false;
527                 }
528                 // Finalizo el registro y habilito su guardado:
529                 if(!startToggleButton.isChecked()){
530                     saveToggleButton.setEnabled(false);
531                     butterSetButton.setEnabled(true);
532                     timer(false);
533                     g.setEnabled(true);
534                     stopTime=System.currentTimeMillis();
535                     if(frecMuest!=2) bf.setEnabled(true);
536                 }
537             }
538         });
```

```
539         // Boton para cargar el registro seleccionado de la listview:
540         loadButton=(Button) findViewById(R.id.loadButton);
541         loadButton.setEnabled(false);
542         loadButton.setOnClickListener(new View.OnClickListener() {
543             @Override
544             public void onClick(View view) {
545                 graph1=null;
546                 progLoad(view);
547                 loadButton.setEnabled(false);
548                 loadedData.setText("Loaded Data: "+acelerationLoadedName.replaceAll(".txt",""));
549                 fftButton.setEnabled(true);
550                 butterButton.setEnabled(true);
551                 detrendButton.setEnabled(true);
552                 rsmButton.setEnabled(true);
553             }
554         });
555         // Acciónes a ejecutar al pinchr sobre un elemento de la listview:
556         savedListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
557             @Override
558             public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
559                 // En resumen, alamacena el nombre del archivo seleccionado para cargarlo:
560                 acelerationLoadedName=String.valueOf(savedListView.getItemAtPosition(i));
561                 loadButton.setEnabled(true);
562                 fftButton.setEnabled(false);
563                 fftSaveButton.setEnabled(false);
564                 butterButton.setEnabled(false);
565                 butterSaveButton.setEnabled(false);
566                 detrendSaveButton.setEnabled(false);
567                 detrendButton.setEnabled(false);
568                 rsmButton.setEnabled(false);
569                 rsmSaveButton.setEnabled(false);
570                 Toast.makeText(getApplicationContext(),acelerationLoadedName,Toast.LENGTH_SHORT)
571                         .show();
572             }
573         });
574         // Boton para el detrend, eliminar la tendencia lineal:
575         detrendButton=(Button) findViewById(R.id.detrendButton) ;
576         detrendButton.setEnabled(false);
577         detrendButton.setOnClickListener(new View.OnClickListener() {
578             @Override
579             public void onClick(View view) {
580                 // Ejecuto lo que corresponda:
581                 if(accelLoaded.getName().indexOf("FFT")==-1) {
582                     if (accelLoaded.getName().indexOf("DETR") == -1) {
583                         // Lo construyo a través de una función declarada más adelante:
584                         accelDetrended = detrend(accelLoaded);
585                         chargeGraphLoaded(accelDetrended, 1);
586                         detrendButton.setEnabled(false);
587                         detrendSaveButton.setEnabled(true);
588                     } else {
589                         detrendButton.setEnabled(false);
590                         Toast.makeText(getApplicationContext(), "ALREADY DETRENDED",
591                                 Toast.LENGTH_LONG).show();
592                     }
593                 }
594                 else{
595                     detrendButton.setEnabled(false);
596                     Toast.makeText(getApplicationContext(), "NOT POSSIBLE", Toast.LENGTH_LONG)
597                             .show();
598                 }
599             }
600         });
601         // Guardado del archivo tras el detrend
602         detrendSaveButton=(Button) findViewById(R.id.detrendSaveButton);
603         detrendSaveButton.setEnabled(false);
604         detrendSaveButton.setOnClickListener(new View.OnClickListener() {
605             @Override
606             public void onClick(View view) {
607                 detrendSaveButton.setEnabled(false);
608                 saveDataFile(accelDetrended.getStringAccel(),accelDetrended.getName());
609                 savedFiles();
610             }
611         });
612         // Ejecución del RMS Trend:
613         rsmSetButton=(Button) findViewById(R.id.rsmSetButton) ;
614         rsmSetButton.setOnClickListener(new View.OnClickListener() {
615             @Override
```

Main Activity

```
616             public void onClick(View view) {
617                 rsmInputDialog();
618             }
619         });
620         rsmButton=(Button) findViewById(R.id.rsmButton);
621         rsmButton.setEnabled(false);
622         rsmButton.setOnClickListener(new View.OnClickListener() {
623             @Override
624             public void onClick(View view) {
625                 if(accelLoaded!=null){
626                     // Ejecuto si no ha sido RMS Trend o FFT antes:
627                     if(accelLoaded.getName().indexOf("RMS")==-1
628                             && accelLoaded.getName().indexOf("FFT")==-1) {
629                         progRms(view);
630                         rsmSaveButton.setEnabled(true);
631                     }else{
632                         Toast.makeText(getApplicationContext(),"NOT POSSIBLE", Toast.LENGTH_LONG)
633                                 .show();
634                     }
635                 }
636 
637                 rsmButton.setEnabled(false);
638 
639             }
640         });
641         // Guardado del RMS Trend
642         rsmSaveButton=(Button) findViewById(R.id.rsmSaveButton);
643         rsmSaveButton.setEnabled(false);
644         rsmSaveButton.setOnClickListener(new View.OnClickListener() {
645             @Override
646             public void onClick(View view) {
647                 rsmSaveButton.setEnabled(false);
648                 saveDataFile(accelRMSTrend.getStringAccel(),accelRMSTrend.getName());
649                 savedFiles();
650             }
651         });
652 
653         CheckBt();
654         // No es necesario aquí, peor lo dejo por si acaso se necesita revisar:
655         // BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);
656         // Log.e("Jon", device.toString());
657 
658         // Selector de Smartphone sensors como imput de datos:
659         sfSelecctor=(Button) findViewById(R.id.sfSelector);
660         sfSelecctor.setOnClickListener(new View.OnClickListener() {
661             @Override
662             public void onClick(View view) {
663                 ard=false;
664             }
665         });
666         sfSelecctor.setEnabled(false);
667         // Selector de Arduino como imput de datos:
668         arSelecctor=(Button) findViewById(R.id.arSellector);
669         arSelecctor.setOnClickListener(new View.OnClickListener() {
670             @Override
671             public void onClick(View view) {
672                 bluetoothInputDialog();
673             }
674         });
675         sfSelecctor.setOnClickListener(new View.OnClickListener() {
676             @Override
677             public void onClick(View view) {
678                 try {
679                     ard=false;
680                     inStream.close();
681                     btSocket.close();
682 
683                     sfSelecctor.setEnabled(false);
684                     arSelecctor.setEnabled(true);
685                 } catch (IOException e) {
686                     e.printStackTrace();
687                 }
688 
689             }
690         });
691 
692     }
```

Main Activity

```
693     // Funicón para alternar entre los dos tipos de sensores que ofrece el Smartphone:
694     private void changgSFSensor(){
695         // Primero elimino el aterior, para que no se solapen los registros
696         mAccelerometer=null;
697         mSensorManager.unregisterListener(this);
698         //mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
699         if(!g.isChecked()){
700             // Sensor TYPE_LINEAR_ACCELERATION (SIN GRAVEDAD):
701             mAccelerometer = mSensorManager.getDefaultSensor(sensorType[0]);
702         }
703         else{
704             // Sensor TYPE_ACCELEROMETER (CON GRAVEDAD):
705             mAccelerometer = mSensorManager.getDefaultSensor(sensorType[1]);
706         }
707         mSensorManager.registerListener(this,mAccelerometer,SensorManager.SENSOR_DELAY_FASTEST);
708
709     }
710     // Parte del ciclo de vida de la actividad:
711     protected void onResume() {
712         super.onResume();
713         mSensorManager.registerListener(this, mAccelerometer, SensorManager.SENSOR_DELAY_FASTEST);
714
715     }
716     protected void onPause() {
717
718         super.onPause();
719
720         mSensorManager.unregisterListener(this);
721
722     }
723     // Funciones para cargar los diferentes grupos de datos en los múltiples gráficos habilitados:
724     // lo hacen en función de los ejes seleccionados, así como de qué tipo de procesado hayan
725     // recivido los datos.
726     public void putAxisGraph1(){
727         graph1.removeAllSeries();
728
729         if(xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
730             graph1.addSeries(mSeries1Loaded);
731             graph1.addSeries(mSeries2Loaded);
732             graph1.addSeries(mSeries3Loaded);
733             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
734                 graph1.addSeries(mSeries1LoadedMTVVX);
735                 graph1.addSeries(mSeries1LoadedMTVVY);
736                 graph1.addSeries(mSeries1LoadedMTVVZ);
737             }
738         }
739         if(xRec.isChecked() && !yRec.isChecked() && !zRec.isChecked()){
740             graph1.addSeries(mSeries1Loaded);
741             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
742                 graph1.addSeries(mSeries1LoadedMTVVX);
743             }
744         }
745         if(!xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
746             graph1.addSeries(mSeries2Loaded);
747             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
748                 graph1.addSeries(mSeries1LoadedMTVVY);
749             }
750         }
751         if(!xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
752             graph1.addSeries(mSeries3Loaded);
753             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
754                 graph1.addSeries(mSeries1LoadedMTVVZ);
755             }
756         }
757         if(xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
758             graph1.addSeries(mSeries1Loaded);
759             graph1.addSeries(mSeries2Loaded);
760             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
761                 graph1.addSeries(mSeries1LoadedMTVVX);
762                 graph1.addSeries(mSeries1LoadedMTVVY);
763             }
764         }
765         if(xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
766             graph1.addSeries(mSeries1Loaded);
767             graph1.addSeries(mSeries3Loaded);
768             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
769                 graph1.addSeries(mSeries1LoadedMTVVX);
```

Main Activity

```
770                 graph1.addSeries(mSeries1LoadedMTVVZ);
771             }
772         }
773         if(!xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
774             graph1.addSeries(mSeries2Loaded);
775             graph1.addSeries(mSeries3Loaded);
776             if(mSeries1LoadedMTVVX!=null && mSeries1LoadedMTVVY!=null && mSeries1LoadedMTVVZ!=null){
777                 graph1.addSeries(mSeries1LoadedMTVVY);
778                 graph1.addSeries(mSeries1LoadedMTVVZ);
779             }
780         }
781
782     }
783     public void putAxisGraph2(){
784         graph1_2.removeAllSeries();
785         if(xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
786             graph1_2.addSeries(mSeries1_2);
787             graph1_2.addSeries(mSeries2_2);
788             graph1_2.addSeries(mSeries3_2);
789         }
790         else if(xRec.isChecked() && !yRec.isChecked() && !zRec.isChecked()){
791             graph1_2.addSeries(mSeries1_2);
792         }
793         else if(!xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
794             graph1_2.addSeries(mSeries2_2);
795         }
796         else if(!xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
797             graph1_2.addSeries(mSeries3_2);
798         }
799         else if(xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
800             graph1_2.addSeries(mSeries1_2);
801             graph1_2.addSeries(mSeries2_2);
802         }
803         else if(xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
804             graph1_2.addSeries(mSeries1_2);
805             graph1_2.addSeries(mSeries3_2);
806         }
807         else if(!xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
808             graph1_2.addSeries(mSeries2_2);
809             graph1_2.addSeries(mSeries3_2);
810         }
811     }
812     public void putAxisGraph1_2(){
813         graph1_2.removeAllSeries();
814         if(xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
815             graph1_2.addSeries(mSeries1);
816             graph1_2.addSeries(mSeries2);
817             graph1_2.addSeries(mSeries3);
818         }
819         else if(xRec.isChecked() && !yRec.isChecked() && !zRec.isChecked()){
820             graph1_2.addSeries(mSeries1);
821         }
822         else if(!xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
823             graph1_2.addSeries(mSeries2);
824         }
825         else if(!xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
826             graph1_2.addSeries(mSeries3);
827         }
828         else if(xRec.isChecked() && yRec.isChecked() && !zRec.isChecked()){
829             graph1_2.addSeries(mSeries1);
830             graph1_2.addSeries(mSeries2);
831         }
832         else if(xRec.isChecked() && !yRec.isChecked() && zRec.isChecked()){
833             graph1_2.addSeries(mSeries1);
834             graph1_2.addSeries(mSeries3);
835         }
836         else if(!xRec.isChecked() && yRec.isChecked() && zRec.isChecked()){
837             graph1_2.addSeries(mSeries2);
838             graph1_2.addSeries(mSeries3);
839         }
840     }
841     // Timer para dar apariencia de que se registra a la frecuencia sellecionada. Esto solo afecta
842     // a la gráfica, la frecuencia en verdad se postprocesa, y el registro se toma a full power
843     public void timer(boolean state){
844         if(state) {
845             mTimer1 = new Runnable() {
846                 @Override
```

Main Activity

```
847                 public void run() {
848
849                     graph1LastXValue_2 = System.currentTimeMillis() - initialTime;
850
851                     mSeries1_2.appendData(new DataPoint(graph1LastXValue_2, x),
852                             true, 300);
853
854                     mSeries2_2.appendData(new DataPoint(graph1LastXValue_2, y),
855                             true, 300);
856
857                     mSeries3_2.appendData(new DataPoint(graph1LastXValue_2, z),
858                             true, 300);
859
860                     mHandler.postDelayed(this, frecMuest / 3);
861                 }
862             };
863             mHandler.postDelayed(mTimer1, frecMuest / 3);
864         }
865         else{
866             mHandler.removeCallbacks(mTimer1);
867         }
868     }
869     // Programación por eventos:
870     // cada vez que detecta un cambio en el acelerómetro, registra el nuevo valor.
871     @Override
872     public void onSensorChanged(SensorEvent event) {
873         if(!ard) {
874             x = event.values[0];
875             y = event.values[1];
876             z = event.values[2];
877             // En el caso de aplicar el filtro en tiempo real, filtra la señal:
878             if(bf.isChecked() ){
879                 x = (float) butterworthX.filter(x);
880                 y = (float) butterworthY.filter(y);
881                 z = (float) butterworthZ.filter(z);
882             }
883         }
884         // Guardado de registros:
885         if(startToggleButton.isChecked() && !ard) {
886
887             if(saveToggleButton.isChecked()&& !ard) {
888                 AccelDataSimple acc = new AccelDataSimple();
889                 acc.setData(System.currentTimeMillis() - initialRecordedTime, x, y, z);
890                 accelColection.add(acc);
891
892             }
893             graph1LastXValue =(System.currentTimeMillis()-initialTime);
894
895             mSeries1.appendData(new DataPoint(graph1LastXValue, x),
896                     true, 350);
897             mSeries2.appendData(new DataPoint(graph1LastXValue, y),
898                     true, 350);
899             mSeries3.appendData(new DataPoint(graph1LastXValue, z),
900                     true, 350);
901         }
902     }
903
904
905
906     //Comprobaciones de almacenamiento externo
907     private static boolean isExternalStorageReadOnly() {
908         String extStorageState = Environment.getExternalStorageState();
909         if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(extStorageState)) {
910             return true;
911         }
912         return false;
913     }
914     //Comprobaciones almacenamiento externo
915     private static boolean isExternalStorageAvailable() {
916         String extStorageState = Environment.getExternalStorageState();
917         if (Environment.MEDIA_MOUNTED.equals(extStorageState)) {
918             return true;
919         }
920         return false;
921     }
922
923     //Cargar archivo guardado
```

Main Activity

```
924     private String readSavedFile(Context context, String name) {
925
926         String ret = "";
927         File myExternalFileReaded=new File(getExternalFilesDir(filepath),name);
928         try {
929             FileInputStream fis = new FileInputStream(myExternalFileReaded);
930             DataInputStream in = new DataInputStream(fis);
931             BufferedReader br = new BufferedReader(new InputStreamReader(in));
932             String strLine;
933             ret="";
934             while ((strLine = br.readLine()) != null) {
935                 ret = ret + strLine;
936             }
937             in.close();
938         } catch (IOException e) {
939             e.printStackTrace();
940         }
941         return ret;
942     }
943     // Clase para guardar archivos
944     public void saveDataFile(String string){
945         String name=df.format(Calendar.getInstance().getTime()) + ".txt";
946         if(bf.isChecked())name.replaceAll(".txt",".BUTTER.txt");
947
948         myExternalFile = new File(getExternalFilesDir(filepath),
949                 name.replaceAll(" ","-"));
950         try {
951             FileOutputStream fos = new FileOutputStream(myExternalFile);
952             fos.write(string.getBytes());
953             fos.close();
954         } catch (IOException e) {
955             e.printStackTrace();
956         }
957         Toast.makeText(this, "SAVED at " + getExternalFilesDir(filepath), Toast.LENGTH_LONG).show();
958     }
959     // Función alternativa para almacenar datos:
960     public void saveDataFile(String string, String name, String op){
961
962         myExternalFile = new File(getExternalFilesDir(filepath),
963                 name.substring(0,name.indexOf(".txt"))+ op +".txt");
964         try {
965             FileOutputStream fos = new FileOutputStream(myExternalFile);
966             fos.write(string.getBytes());
967             fos.close();
968         } catch (IOException e) {
969             e.printStackTrace();
970         }
971         Toast.makeText(this, "SAVED at " + getExternalFilesDir(filepath), Toast.LENGTH_LONG).show();
972     }
973     // Función para guardar archivo, ahora con nombre:
974     public void saveDataFile(String string, String name){
975
976         myExternalFile = new File(getExternalFilesDir(filepath),name);
977         try {
978             FileOutputStream fos = new FileOutputStream(myExternalFile);
979             fos.write(string.getBytes());
980             fos.close();
981         } catch (IOException e) {
982             e.printStackTrace();
983         }
984         Toast.makeText(this, "SAVED at " + getExternalFilesDir(filepath), Toast.LENGTH_LONG)
985                 .show();
986     }
987     // Función alternativa para guardar:
988     public void saveDataFile2(){ //Actualmente en desuso
989
990         myExternalFile = new File(getExternalFilesDir(filepath),
991                 df.format(Calendar.getInstance().getTime()) + ".txt");
992         try {
993             FileOutputStream fos = new FileOutputStream(myExternalFile);
994             fos.write(resampled.getStringAccel().getBytes());
995             fos.close();
996         } catch (IOException e) {
997             e.printStackTrace();
998         }
999         Toast.makeText(this, "SAVED at " + getExternalFilesDir(filepath), Toast.LENGTH_LONG)
1000                .show();
```

```
1001    }
1002
1003    // Clase para añadir objetos a la list view:
1004    public void savedFiles(){
1005        String caca = String.valueOf(getExternalFilesDir(filepath));
1006        File file=new File(caca/*Environment.getExternalStorageDirectory().getAbsolutePath()+
1007                File.separator+"Android"+File.separator+"data"+File.separator+
1008                "com.example.jalab.pruebaaccelgraphview"+File.separator+
1009                "files"+File.separator+"MyFileStorage"*/);
1010        List items = new ArrayList<String>();
1011        String arr[]=file.list();
1012
1013        if(arr!=null) {
1014            for (String i : arr) {
1015                if (i.endsWith(".txt")) {
1016                    items.add(i);
1017                }
1018            }
1019        }
1020        savedListViewAdpter=new ArrayAdapter(this, android.R.layout.simple_list_item_1,
1021                items);
1022        savedListView.setAdapter(savedListViewAdpter);
1023
1024    }
1025
1026    // Cargar archivo desde String versión 1: Ahora en desuso por meter la progressbar
1027    // , pero funcional
1028    private AccelData loadAccel (String data){
1029        double time=0.0;
1030        float []accelData=new float[3];
1031        AccelData accel;
1032        String data2=" ", aux=" ";
1033        // Me quedo con el string hasta el primer tiempo:
1034        data2=data.substring(data.indexOf("Z")+1);
1035        //numero de datos en el registro
1036        int conti=(data2.length()-data2.replaceAll("\t","").length())/4;
1037        accel=new AccelData(conti);
1038        for(int k=0; k<conti; k++){
1039            for(int i=0; i<4;i ++){
1040                data2=data2.substring(data2.indexOf("\t")+1);// Quito la primera tabulación
1041                // Me quedo con el primer dígito
1042                if(data2.indexOf("\t")!=-1)aux=data2.substring(0,data2.indexOf("\t"));
1043                if(data2.indexOf("\t")==-1)aux=data2;
1044                // Guardo:
1045                if(i==0) time=Double.parseDouble(aux);
1046                if(i==1) accelData[0]=Float.parseFloat(aux);
1047                if(i==2) accelData[1]=Float.parseFloat(aux);
1048                if(i==3) accelData[2]=Float.parseFloat(aux);
1049                // Quito dato registrado
1050                if(data2.indexOf("\t")!=-1) data2=data2.substring(data2.indexOf("\t"));
1051            }
1052            accel.setData(k,time,accelData);
1053        }
1054        return accel;
1055
1056    }
1057    // Cargar archivo desde String versión 2 --> Habilitada para la progressbar (secuencial):
1058    private int loadAccel2 (int conti, int contRead){
1059        double time=0.0;
1060        float []accelData=new float[3];
1061        String aux=" ";
1062        for(int i=0; i<4;i ++){
1063            // Quito la primera tabulación
1064            acelerationaLoadedAux=acelerationaLoadedAux
1065                    .substring(acelerationaLoadedAux.indexOf("\t")+1);
1066            // Me quedo con el primer dígito
1067            if(acelerationaLoadedAux.indexOf("\t")!=-1)aux=acelerationaLoadedAux
1068                    .substring(0,acelerationaLoadedAux.indexOf("\t"));
1069            if(acelerationaLoadedAux.indexOf("\t")==-1)aux=acelerationaLoadedAux;
1070            // Guardo:
1071            if(i==0) time=Double.parseDouble(aux);
1072            if(i==1) accelData[0]=Float.parseFloat(aux);
1073            if(i==2) accelData[1]=Float.parseFloat(aux);
1074            if(i==3) accelData[2]=Float.parseFloat(aux);
1075            if(acelerationaLoadedAux.indexOf("\t")!=-1)
1076                acelerationaLoadedAux=acelerationaLoadedAux
1077                        .substring(acelerationaLoadedAux.indexOf("\t")); // Quito dato registrado
```

Main Activity

```
        }

        // acceleration.setData(contRead,time,accelData);
        AccelDataSimple acc=new AccelDataSimple();
        acc.setData(time,accelData[0],accelData[1],accelData[2]);
        accelLoaded.add(acc);
        return (contRead+1)*100/conti;
    }
    // Progressbar para cargar los archivos:
    public void progLoad(View view){
        // Como a veces tarda, pongo una Progress Bar, así se notamenos que es una chapusilla
        contRead=0;
        progressBar = new ProgressDialog(view.getContext());
        progressBar.setCancelable(false);
        progressBar.setMessage("LOADING FILE :"+acelerationLoadedName);
        progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
        progressBar.setProgress(0);
        progressBar.setMax(100);
        progressBar.show();
        //reset progress bar status
        progressBarStatus = 0;

        new Thread(new Runnable() {
            public void run() {

                //Creo el string con el archivo guardado auxiliar y le quito hasta la "Z":
                acelerationaLoaded=readSavedFile(getApplicationContext(),acelerationLoadedName);
                acelerationaLoadedAux=acelerationaLoaded
                        .substring(acelerationaLoaded.indexOf("Z")+1);
                // numero de vectores almacenado: será el
                // numero de tabulaciones / 4 (4 tabulaciones por vector)
                //Esto es así porque al escribir SI escribe \n, pero NO lo lee... Su puta madre
                contiG=(acelerationaLoadedAux.length()-acelerationaLoadedAux.replaceAll("\t","")
                        .length())/4;//numero de datos en el registro
                // creo la instancia de aceleración ya con el tamaño oportuno
                // acceleration=new AccelData(contiG);
                accelLoaded= new AccelColection(contiG);
                accelLoaded.setName(acelerationLoadedName);
                if(acelerationLoadedName.endsWith(".FFT.txt"))
                    accelLoaded.setName(acelerationLoadedName);
                while (progressBarStatus < 100) {
                    // Ejecuto la función de transcricción
                    progressBarStatus = loadAccel2(contiG, contRead);
                    contRead++;
                    //El resto son movidas de la progress bar
                    progressBarHandler.post(new Runnable() {
                        public void run() {
                            progressBar.setProgress(progressBarStatus);
                        }
                    });
                }
                if(acelerationLoadedName.endsWith(".FFT.txt"))chargeGraphLoaded(accelLoaded,2);
                else chargeGraphLoaded(accelLoaded,1);
                putAxisGraph1();
                // más movidas de la progres brass
                if (progressBarStatus >= 100) {
                    // un segundo para obserbar lo maravilloso que es el 100%
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    // Cierra el dialog
                    progressBar.dismiss();
                }
            }
        }).start();

    }
    //Guardar con barra de carga, para que no se pete esperando:
    public void progSave(View view){
        contSave=0;
        marcRes=0;
        contRes=0;
        // Como a veces tarda, pongo una Progress Bar, así se notamenos que es una chapusilla
        progressBar = new ProgressDialog(view.getContext());
        progressBar.setCancelable(false);
```

```
1155        progressBar.setMessage("SAVINGING AND RESAMPLING FILE :");
1156        progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
1157        progressBar.setProgress(0);
1158        progressBar.setMax(100);
1159        progressBar.show();
1160        //reset progress bar status
1161        progressBarStatus = 0;
1162        // En funcion de si puedo resamplear o no a esa frecuencia de muestreo (sólo útil en el
1163        // registro mediante arduino, mediante SF siempre es posible), determino uno u otro
1164        // método:
1165        if(accelColection.getTimeMean()<frecMuest){
1166            contRes=(int) accelColection.get(accelColection.length()-1).getTime()/frecMuest;
1167        }
1168        else{
1169            contRes=(int) accelColection.get(accelColection.length()-1).getTime()/
1170                    (int)(accelColection.getTimeMean()+1);
1171        }
1172        // Ejecuto el resample del registro:
1173        resampled=new AccelColection(contRes);
1174        // Ejecuto secuencialmente, en otro hilo, el guardado:
1175        new Thread(new Runnable() {
1176            public void run() {
1177
1178                while (progressBarStatus < 100) {
1179                    // Ejecuto la función de transcripción
1180                    progressBarStatus = resample();
1181                    contSave++;
1182                    //El resto son movidas de la progress bar
1183                    progressBarHandler.post(new Runnable() {
1184                        public void run() {
1185                            progressBar.setProgress(progressBarStatus);
1186                        }
1187                    });
1188                }
1189                // más movidas de la progres brass
1190                if (progressBarStatus >= 100) {
1191                    // un segundo para obserbar lo maravilloso que es el 100%
1192                    try {
1193                        Thread.sleep(1000);
1194                    } catch (InterruptedException e) {
1195                        e.printStackTrace();
1196                    }
1197                    // Cierra el dialog
1198                    progressBar.dismiss();
1199                }
1200            }
1201        }).start();
1202    }
1203    // Función detrend: --> Elimina la tendiencia lineal
1204    public AccelColection detrend(AccelColection data){
1205        AccelColection detrended=new AccelColection(data.length());
1206        detrended.setName(data.getName().replaceAll(".txt",".DETREND.txt"));
1207        AccelDataSimple acc;
1208        for(int i=0; i<data.length(); i++){
1209            acc=new AccelDataSimple();
1210            acc.setData(data.get(i).getTime(),data.get(i).getX()-data.getXMean(),
1211                    data.get(i).getY()-data.getYMean(),
1212                    data.get(i).getZ()-data.getZMean());
1213            detrended.add(acc);
1214        }
1215        accelLoaded=detrended;
1216        return detrended;
1217    }
1218    // Función alternativa para guardar el resample sin progressbar:
1219    public void saveResample(){
1220        contSave=0;
1221        marcRes=0;
1222        contRes=0;
1223        contRes=(int) accelColection.get(accelColection.length()-1).getTime()/frecMuest;
1224        resampled=new AccelColection(contRes);
1225        while (contSave<contRes){
1226            int i=resample();
1227            contSave++;
1228        }
1229        saveDataFile2();
1230    }
1231    // Función para la FFT: debido a su complegidad, no ha sido posible hacerlo mediantae
```

Main Activity

```
1232    // progressbar, con lo que si son un número elevado de puntos se congelará el programa hasta
1233    // que se termine la ejecuccion del bucle:
1234    public void fftProces(){
1235        // Primera instancia de múltiplos de dos:
1236        int NFFT=2;
1237        // Busco el siguiente múltiplo de dós al número de datos:
1238        while(NFFT<accelLoaded.length()) NFFT=NFFT*2;
1239        // Instancio para cada eje el array de objeto complejo correspondiente
1240        Complex[] compx= new Complex[NFFT];
1241        Complex[] compy= new Complex[NFFT];
1242        Complex[] compz= new Complex[NFFT];
1243        // Introduccó los datos a las instancias creadas:
1244        for(int i=0; i<NFFT; i++){
1245            if(i<accelLoaded.length()) {
1246                compx[i] = new Complex(i, 0);
1247                compx[i] = new Complex(accelLoaded.get(i).getX(), 0);
1248                compy[i] = new Complex(i, 0);
1249                compy[i] = new Complex(accelLoaded.get(i).getY(), 0);
1250                compz[i] = new Complex(i, 0);
1251                compz[i] = new Complex(accelLoaded.get(i).getZ(), 0);
1252            }
1253            else{
1254                compx[i] = new Complex(i, 0);
1255                compx[i] = new Complex(0, 0);
1256                compy[i] = new Complex(i, 0);
1257                compy[i] = new Complex(0, 0);
1258                compz[i] = new Complex(i, 0);
1259                compz[i] = new Complex(0, 0);
1260            }
1261        }
1262        // Ejecuto la FFT, que es una clase aparte.
1263        Complex[] yX=FFT.fft(compx);
1264        Complex[] yY=FFT.fft(compy);
1265        Complex[] yZ=FFT.fft(compz);
1266        // Instancio y nombro la colección procesada.
1267        accelFFTed=new AccelColection(NFFT);
1268        accelFFTed.setName(accelLoaded.getName().replaceAll(".txt",".FFT.txt"));
1269        // calcuo la frecuencia:
1270        double fhz= 1000/(accelLoaded.get(1).getTime()-accelLoaded.get(0).getTime());
1271        double point=fhz/(NFFT-1);
1272        double cont=0;
1273        // Inroduzco los puntos:
1274        for(int i=0; i<NFFT; i++){
1275            AccelDataSimple acc=new AccelDataSimple();
1276
1277            acc.setData(cont,
1278                (float) yX[i].abs()/(float) accelLoaded.length(),
1279                (float) yY[i].abs()/(float) accelLoaded.length(),
1280                (float) yZ[i].abs()/(float) accelLoaded.length());
1281            cont+=point;
1282
1283            accelFFTed.add(acc);
1284        }
1285        accelLoaded=accelFFTed;
1286        Toast.makeText(this,"FFT Calculated",Toast.LENGTH_SHORT).show();
1287    }
1288    // Función para resamplear el registro:
1289    private int resample(){
1290        // De nuevo, para el arduino, en funcion de la frecuencia máxima:
1291        int timePront;
1292        if(accelColection.getTimeMean()<frecMuest){
1293            timePront=frecMuest;
1294        }
1295        else{
1296            timePront=(int) accelColection.getTimeMean()+1;
1297        }
1298        // Instancio la colección:
1299        AccelDataSimple acc=new AccelDataSimple();
1300        // El punto incial se toma como inicio del resampleo.
1301        if(contSave==0) {
1302            acc.setData(accelColection.get(0).getTime(),accelColection.get(0).getX(),
1303                    accelColection.get(0).getY(),accelColection.get(0).getZ());
1304            resampled.add(acc);
1305        }
1306        // El resto se calculan:
1307        else{
1308            // el resampleo consiste en una interpolación lineal:
```

```
            double time=resampled.get(contSave-1).getTime()+timePront;
            float x, y, z;
            while(accelColection.get(marcRes).getTime()<=time)
                marcRes++;
            x=linearInterpolation(accelColection.get(marcRes-1).getX(),
                    accelColection.get(marcRes).getX(),accelColection.get(marcRes-1).getTime(),
                    accelColection.get(marcRes).getTime(), time);
            y=linearInterpolation(accelColection.get(marcRes-1).getY(),
                    accelColection.get(marcRes).getY(),accelColection.get(marcRes-1).getTime(),
                    accelColection.get(marcRes).getTime(), time);
            z=linearInterpolation(accelColection.get(marcRes-1).getZ(),
                    accelColection.get(marcRes).getZ(),accelColection.get(marcRes-1).getTime(),
                    accelColection.get(marcRes).getTime(), time);
            acc.setData(time,x,y,z);
            resampled.add(acc);
        }
        if((-1)*resampled.get(resampled.length()-1).getTime()+ accelColection.get(
                accelColection.length()-1).getTime()>=timePront){
            return contSave*100/(contRes-1);
        }
        else return 100;
    }
    // Interpolación linear para el resampling:
    private float linearInterpolation(float xi0, float xi1, double ti0, double ti1, double ti){
        return xi0+(xi1-xi0)*(((float) ti- (float) ti0))/((float) ti1- (float) ti0);
    }
    // Función para cargar los gráficos en función de los datos de entrada:
    public void chargeGraphLoaded(AccelColection acc, int n){

        graph1 = (GraphView) findViewById(R.id.graph);
        graph1.getViewport().setXAxisBoundsManual(true);
        graph1.getViewport().setMinX(0);
        if(n==1 || n==3){
            graph1.getViewport().setMaxX(1800);
            graph1.getGridLabelRenderer().setHorizontalLabelsVisible(false);
        }
        // En caso de FFT, solo se represneta la mitad:
        if(n==2){
            graph1.getViewport().setMaxX(acc.get(acc.length()-1).getTime()/2);
            graph1.getGridLabelRenderer().setHorizontalLabelsVisible(true);
        }
        graph1.getViewport().setScalable(true);
        graph1.getViewport().setScalableY(true);
        graph1.getViewport().setScrollable(true);
        graph1.getViewport().setScrollableY(true);
        DataPoint[] points1=new DataPoint[acc.length()/n];
        DataPoint[] points2=new DataPoint[acc.length()/n];
        DataPoint[] points3=new DataPoint[acc.length()/n];
        int u=0;
        for(int i=0; i< acc.length()/n ; i++){

            points1[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getX());
            points2[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getY());
            points3[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getZ());
            //u+=acc.get(acc.length()-1).getTime()/(acc.length()-1);
        }
        if(n==1 || n==3) {
            mSeries1Loaded = new LineGraphSeries<>(points1);
            mSeries2Loaded = new LineGraphSeries<>(points2);
            mSeries3Loaded = new LineGraphSeries<>(points3);
            // En el caso del RMS Trend, represento como puntos gordos los máximos
            if(acc.getName().indexOf("RMS")!=-1){
                if(acc.getMTVVX()==0 && acc.getMTVVY()==0 && acc.getMTVVZ()==0){
                    acc.generateMTVV();
                }
                DataPoint pointX[]=new DataPoint[2];
                DataPoint pointY[]=new DataPoint[2];
                DataPoint pointZ[]=new DataPoint[2];
                MTVVX=acc.getMTVVX();
                MTVVY=acc.getMTVVY();
                MTVVZ=acc.getMTVVZ();
                pointX[0]=new DataPoint
                        (acc.getMTVVX_T(),acc.getMTVVX());
                pointX[1]=new DataPoint
                        (acc.getMTVVX_T()+1,acc.getMTVVX());
                pointY[0]=new DataPoint
                        (acc.getMTVVY_T(),acc.getMTVVY());
```

Main Activity

```
                pointY[1]=new DataPoint
                        (acc.getMTVVY_T()+1,acc.getMTVVY());
                pointZ[0]=new DataPoint
                        (acc.getMTVVZ_T(),acc.getMTVVZ());
                pointZ[1]=new DataPoint
                        (acc.getMTVVZ_T()+1,acc.getMTVVZ());
                mSeries1LoadedMTVVX=new PointsGraphSeries<>(pointX);
                mSeries1LoadedMTVVY=new PointsGraphSeries<>(pointY);
                mSeries1LoadedMTVVZ=new PointsGraphSeries<>(pointZ);
                mSeries1LoadedMTVVX.setSize(50);
                mSeries1LoadedMTVVY.setSize(50);
                mSeries1LoadedMTVVZ.setSize(50);
                mSeries1LoadedMTVVX.setColor(Color.parseColor("#182ce6"));
                mSeries1LoadedMTVVY.setColor(Color.parseColor("#de1a13"));
                mSeries1LoadedMTVVZ.setColor(Color.parseColor("#000000"));
                // Estos puntos gordos además se pueden tocar para desplegar en pantaya su valro:
                mSeries1LoadedMTVVX.setOnDataPointTapListener(new OnDataPointTapListener() {
                    @Override
                    public void onTap(Series series, DataPointInterface dataPointInterface) {
                        Toast.makeText(getApplicationContext(),"MTVV X= "+MTVVX,Toast.LENGTH_LONG)
                                .show();
                    }
                });
                mSeries1LoadedMTVVY.setOnDataPointTapListener(new OnDataPointTapListener() {
                    @Override
                    public void onTap(Series series, DataPointInterface dataPointInterface) {
                        Toast.makeText(getApplicationContext(),"MTVV Y= "+MTVVY,Toast.LENGTH_LONG)
                                .show();
                    }
                });
                mSeries1LoadedMTVVZ.setOnDataPointTapListener(new OnDataPointTapListener() {
                    @Override
                    public void onTap(Series series, DataPointInterface dataPointInterface) {
                        Toast.makeText(getApplicationContext(),"MTVV Z= "+MTVVZ,Toast.LENGTH_LONG)
                                .show();
                    }
                });

            }
            if(acc.getName().indexOf("RMS")==-1  ){
                mSeries1LoadedMTVVX=null;
                mSeries1LoadedMTVVY=null;
                mSeries1LoadedMTVVZ=null;
            }
        }
        if(n==2) {
            mSeries1Loaded = new LineGraphSeries<>(points1);
            mSeries2Loaded = new LineGraphSeries<>(points2);
            mSeries3Loaded = new LineGraphSeries<>(points3);

        }
        mSeries1Loaded.setColor(Color.parseColor("#182ce6"));
        mSeries2Loaded.setColor(Color.parseColor("#de1a13"));
        mSeries3Loaded.setColor(Color.parseColor("#000000"));
        putAxisGraph1();
    }
    // Función para procesar el butterworth filtter sobre el registro:
    public AccelColection butterEx(){
        // Instancio nueva colección:
        AccelColection acc=new AccelColection(accelLoaded.length());
        // Instancio dato:
        AccelDataSimple ac=new AccelDataSimple();
        // Instancio puntos:
        DataPoint[] points1=new DataPoint[accelLoaded.length()];
        DataPoint[] points2=new DataPoint[accelLoaded.length()];
        DataPoint[] points3=new DataPoint[accelLoaded.length()];
        // Variables:
        float x,y,z,X,Y,Z;
        // Y filtro:
        for(int i=0; i<accelLoaded.length(); i++){
            x=accelLoaded.get(i).getX();
            y=accelLoaded.get(i).getY();
            z=accelLoaded.get(i).getZ();
            // EL filtro es una función de una biblioteca descargada:
            X = (float) butterworthX.filter(x);
            Y = (float) butterworthY.filter(y);
            Z = (float) butterworthZ.filter(z);
```

```
1463            ac.setData(accelLoaded.get(i).getTime(), X, Y, Z);
1464            acc.add(ac);
1465            points1[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getX());
1466            points2[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getY());
1467            points3[i]=new DataPoint(acc.get(i).getTime(), acc.get(i).getZ());
1468        }
1469        // Por ultimo habilito el guardado y cargo el gráfico:
1470        acc.setName(accelLoaded.getName().replaceAll(".txt",".BUTTER.txt"));
1471        mSeries1Loaded = new LineGraphSeries<>(points1);
1472        mSeries2Loaded = new LineGraphSeries<>(points2);
1473        mSeries3Loaded = new LineGraphSeries<>(points3);
1474        mSeries1Loaded.setColor(Color.parseColor("#182ce6"));
1475        mSeries2Loaded.setColor(Color.parseColor("#de1a13"));
1476        mSeries3Loaded.setColor(Color.parseColor("#000000"));
1477        putAxisGraph1();
1478        return acc;
1479    }
1480    // Ciclo de vida de la actividad:
1481    @Override
1482    public void onAccuracyChanged(Sensor sensor, int i) {
1483
1484    }
1485    // Ventana del butterfilter:
1486    protected void butterInputDialog() {
1487
1488        LayoutInflater layoutInflater = LayoutInflater.from(MainActivity.this);
1489        View promptView = layoutInflater.inflate(R.layout.input_butter_dialog, null);
1490        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(MainActivity.this);
1491        alertDialogBuilder.setView(promptView);
1492
1493        final Spinner orederSpinner = (Spinner) promptView.findViewById(R.id.orderSpinner);
1494        final EditText cutoffEditText = (EditText) promptView.findViewById(R.id.cutoffEditText);
1495        final Button applybutter=(Button) promptView.findViewById(R.id.applyButter);
1496        ArrayAdapter<CharSequence> adapterOrder = ArrayAdapter.createFromResource(this,
1497                R.array.order, android.R.layout.simple_spinner_item);
1498        adapterOrder.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
1499        orederSpinner.setAdapter(adapterOrder);
1500        orederSpinner.setSelection(adapterOrder.getPosition(String.valueOf(order)));
1501
1502        applybutter.setOnClickListener(new View.OnClickListener() {
1503            @Override
1504            public void onClick(View view) {
1505                try{
1506                    // Almaceno valores introducidos:
1507                    cutoff=Double.parseDouble(cutoffEditText.getText().toString());
1508                    cutoffEditText.setText("");
1509                    cutoffEditText.setHint(String.valueOf(cutoff));
1510                }
1511                catch(NumberFormatException e){
1512                    Toast.makeText(getApplicationContext(),"Invalid cutoff",Toast.LENGTH_SHORT)
1513                            .show();
1514                }
1515            }
1516        });
1517
1518        if(cutoff==0){
1519            cutoff=30;
1520            cutoffEditText.setHint(String.valueOf(cutoff));
1521        }
1522        if(cutoff!=0) cutoffEditText.setHint(String.valueOf(cutoff));
1523        orederSpinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
1524            @Override
1525            public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {
1526                // <Selecciono posible orden del filtro:
1527                String item=(String) adapterView.getItemAtPosition(i);
1528                order=Integer.parseInt(item);
1529            }
1530
1531            @Override
1532            public void onNothingSelected(AdapterView<?> adapterView) {
1533
1534            }
1535        });
1536        // Desplego la ventana
1537        alertDialogBuilder.setCancelable(false)
1538                .setPositiveButton("APPLY", new DialogInterface.OnClickListener() {
1539                    public void onClick(DialogInterface dialog, int id) {
```

Main Activity

```
                        if (!cutoffEditText.getText().toString().equals(""))
                            cutoff = Double.parseDouble(cutoffEditText.getText().toString());
                            dialog.cancel();
                        // Funcionamento alternativo para posible revisión:
                    /*if(accelLoaded!=null) {
                        if (accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                                accelLoaded.get(2).getTime() - accelLoaded.get(1).getTime()
                                && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                                accelLoaded.get(3).getTime() - accelLoaded.get(2).getTime()
                                && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                                accelLoaded.get(4).getTime() - accelLoaded.get(3).getTime()
                                && accelLoaded.getName().indexOf("BUTTER") == -1 &&
                                accelLoaded.getName().indexOf("FFT") == -1) {

                            if (!cutoffEditText.getText().toString().equals(""))
                                cutoff = Double.parseDouble(cutoffEditText.getText().toString());
                            butterButton.setEnabled(false);
                            butterSaveButton.setEnabled(true);
                            butterworthX.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                                    - accelLoaded.get(0).getTime())), cutoff);
                            butterworthY.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                                    - accelLoaded.get(0).getTime())), cutoff);
                            butterworthZ.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                                    - accelLoaded.get(0).getTime())), cutoff);
                            /*accelButted = butterEx();
                            chargeGraphLoaded(accelButted, 1);
                            putAxisGraph1();
                            dialog.cancel();*/
                            /*doButter(dialog);

                        } else {
                            Toast.makeText(getApplicationContext(), "NOT POSSIBLE",
                                    Toast.LENGTH_SHORT).show();
                        }
                    } else {
                        Toast.makeText(getApplicationContext(), "NOT POSSIBLE",
                                Toast.LENGTH_SHORT).show();
                    }*/
                    }
                })
                .setNegativeButton("Cancel",
                        new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog, int id) {
                                dialog.cancel();
                            }
                        });

        // create an alert dialog
        AlertDialog alert = alertDialogBuilder.create();
        alert.show();
    }
    // Versión alternativa para ejecutar el butterworht filter, apra posible revisión:
    public void doButter(){

        if (accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                accelLoaded.get(2).getTime() - accelLoaded.get(1).getTime()
                && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                accelLoaded.get(3).getTime() - accelLoaded.get(2).getTime()
                && accelLoaded.get(1).getTime() - accelLoaded.get(0).getTime() ==
                accelLoaded.get(4).getTime() - accelLoaded.get(3).getTime()
                && accelLoaded.getName().indexOf("BUTTER") == -1 &&
                accelLoaded.getName().indexOf("FFT") == -1) {

            butterButton.setEnabled(false);
            butterSaveButton.setEnabled(true);
            butterworthX.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                    - accelLoaded.get(0).getTime())), cutoff);
            butterworthY.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                    - accelLoaded.get(0).getTime())), cutoff);
            butterworthZ.lowPass(order, (1000 / (accelLoaded.get(1).getTime()
                    - accelLoaded.get(0).getTime())), cutoff);
                    /*accelButted = butterEx();
                    chargeGraphLoaded(accelButted, 1);
                    putAxisGraph1();
                    dialog.cancel();*/
            accelButted = butterEx();
            accelLoaded=accelButted;
```

```
1617            chargeGraphLoaded(accelButted, 1);
1618            putAxisGraph1();
1619
1620        } else {
1621            Toast.makeText(getApplicationContext(), "NOT POSSIBLE",
1622                    Toast.LENGTH_SHORT).show();
1623        }
1624
1625
1626        //dialog.cancel();
1627
1628    }
1629    // Ventana de entrada para la conexxión bluetooth:
1630    protected void bluetoothInputDialog() {
1631        LayoutInflater layoutInflater = LayoutInflater.from(MainActivity.this);
1632        View promptView = layoutInflater.inflate(R.layout.bluetooth_config_dialog, null);
1633        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(MainActivity.this);
1634        alertDialogBuilder.setView(promptView);
1635
1636
1637
1638        alertDialogBuilder.setCancelable(false).setNegativeButton("Cancel", new DialogInterface
1639                .OnClickListener(){
1640            public void onClick(DialogInterface dialog, int id) {
1641                dialog.cancel();
1642            }
1643        }).setPositiveButton("Connect",new DialogInterface.OnClickListener() {
1644            public void onClick(DialogInterface dialog, int id) {
1645                try{
1646                    // lanzo la conexión bluetooth:
1647                    Connect();
1648                    dialog.cancel();
1649                }
1650                catch(Exception e){
1651                    Toast.makeText(getApplicationContext(),"Connection Error",Toast.LENGTH_SHORT
1652                    ).show();
1653                }
1654            }
1655        });
1656        AlertDialog alert = alertDialogBuilder.create();
1657        alert.show();
1658
1659        final TextView currentAddress=(TextView) promptView.findViewById(R.id.currentAddress);
1660        final EditText macAddressEt=(EditText) promptView.findViewById(R.id.macAddressEt);
1661        Button saveAddress=(Button) promptView.findViewById(R.id.saveAddress);
1662
1663        currentAddress.setText("Current Address :"+address);
1664        macAddressEt.setHint(address);
1665
1666        saveAddress.setOnClickListener(new View.OnClickListener() {
1667            @Override
1668            public void onClick(View view) {
1669                // Almaceno posible nueva dirección:
1670                address= macAddressEt.getText().toString();
1671                currentAddress.setText("Current Address :"+address);
1672                macAddressEt.setHint(address);
1673            }
1674        });
1675
1676    }
1677    // Ventana de entrada de datos del RMS Trend: Ventana de cálculo y T2:
1678    protected void rsmInputDialog() {
1679        LayoutInflater layoutInflater = LayoutInflater.from(MainActivity.this);
1680        View promptView = layoutInflater.inflate(R.layout.rsm_trend_data_dialog, null);
1681        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(MainActivity.this);
1682        alertDialogBuilder.setView(promptView);
1683
1684
1685        alertDialogBuilder.setCancelable(false).setNegativeButton("CANCEL", new DialogInterface
1686                .OnClickListener(){
1687            public void onClick(DialogInterface dialog, int id) {
1688                dialog.cancel();
1689            }
1690        }).setPositiveButton("OK",new DialogInterface.OnClickListener() {
1691            public void onClick(DialogInterface dialog, int id) {
1692                dialog.cancel();
1693            }
```

Main Activity

```
        });
        AlertDialog alert = alertDialogBuilder.create();
        alert.show();

        final EditText winEdit=(EditText) promptView.findViewById(R.id.winEditText);
        final EditText t2Edit=(EditText) promptView.findViewById(R.id.t2EditText);
        Button apply=(Button) promptView.findViewById(R.id.rsmtrednApplyButton);
        winEdit.setHint(String.valueOf(win));
        t2Edit.setHint(String.valueOf(T2));
        apply.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                try{
                    win=Double.parseDouble(winEdit.getText().toString());
                    T2=Double.parseDouble(t2Edit.getText().toString());
                }
                catch (NumberFormatException e){
                    Toast.makeText(getApplicationContext(),"INVALID VALUES",Toast.LENGTH_SHORT)
                            .show();
                }

            }
        });

    }
    // Progressbar de ejecución del RMS Trend:
    public void progRms(View view){
        // Como a veces tarda, pongo una Progress Bar, así se notamenos que es una chapusilla
        contRms=0;
        progressBar = new ProgressDialog(view.getContext());
        progressBar.setCancelable(false);
        progressBar.setMessage("Calculating RMS_Trend:"+acelerationLoadedName);
        progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
        progressBar.setProgress(0);
        progressBar.setMax(100);
        progressBar.show();
        // Ejecuto en un hilo diferente el calculo del RMS Trend:
        new Thread(new Runnable() {
            public void run() {

                // Anulo everything por si habia algo:
                RMSX=null; RMSY=null; RMSZ=null;
                RMS_T=null;
                meanX=0;meanY=0;meanZ=0; MTVVX=0; MTVVY=0; MTVVZ=0;
                MTVVX_T=0; MTVVY_T=0; MTVVZ_T=0;
                Ndat=0; Npts=0; N=0;
                fs=1000/(accelLoaded.get(1).getTime()-accelLoaded.get(0).getTime());
                posi=null;
                //Calculo
                Ndat=(int) Math.ceil(win*fs); // Número de datos para la RMS_trend
                Npts=(int) Math.ceil(T2*fs); // Número de puntos en cada cálculo
                // Posiciones de los vectores Accel-Time que ocuparán los cálculos:
                posi=new int[((accelLoaded.length()-Ndat)/Npts)+1];

                //reset progress bar status
                RMSX=new double[posi.length]; // La medai esta --> root mean square
                RMSY=new double[posi.length];RMSZ=new double[posi.length];
                RMS_T=new double[posi.length];// Tiempo para el que se calcula el RMS
                progressBarStatus = 0;
                // Instancia
                accelRMSTrend=new AccelColection(posi.length);
                accelRMSTrend.setName(accelLoaded.getName().replaceAll(".txt",".RMS_Trend.txt"));
                for(int i=0; i<posi.length; i++){
                    posi[i]=Ndat+Npts*i;
                }
                while (progressBarStatus < 100) {
                    // Ejecuto la función de transcricción
                    progressBarStatus = rmsTrendProcess(contRms);
                    contRms++;
                    //El resto son movidas de la progress bar
                    progressBarHandler.post(new Runnable() {
                        public void run() {
                            progressBar.setProgress(progressBarStatus);
                        }
                    });
                }
                chargeGraphLoaded(accelRMSTrend,1);
```

```
1771                putAxisGraph1();
1772                // más movidas de la progres brass
1773                if (progressBarStatus >= 100) {
1774                    // un segundo para obserbar lo maravilloso que es el 100%
1775                    try {
1776                        Thread.sleep(1000);
1777                    } catch (InterruptedException e) {
1778                        e.printStackTrace();
1779                    }
1780                    // Cierra el dialog
1781                    progressBar.dismiss();
1782                }
1783            }
1784        }).start();
1785
1786    }
1787    // Proceso secuancial del RMS Trend:
1788    private int rmsTrendProcess(int i){
1789        //RMS_T[i]=accelLoaded.get([posi[i]]-win/2).getTime();
1790        // Tiempo de calculo:
1791        RMS_T[i]=accelLoaded.get(posi[i]-1).getTime()-win/2;
1792        meanX=0;meanY=0;meanX=0;
1793        N=0;
1794        // Calculo los sumatorios al cuadrado
1795        for(int k=posi[i]-Ndat; k<posi[i]; k++){
1796            meanX+=accelLoaded.get(k).getX()*accelLoaded.get(k).getX();
1797            meanY+=accelLoaded.get(k).getY()*accelLoaded.get(k).getY();
1798            meanZ+=accelLoaded.get(k).getZ()*accelLoaded.get(k).getZ();
1799            N++;
1800        }
1801        // Las medias
1802        meanX=meanX/N;
1803        meanY=meanY/N;
1804        meanZ=meanZ/N;
1805        // Y por último las raizes cuadradas, así como el máixmo de cada eje
1806        RMSX[i]=Math.sqrt(meanX);
1807        if(RMSX[i]>MTVVX){ MTVVX =RMSX[i];MTVVX_T=RMS_T[i]; }
1808        RMSY[i]=Math.sqrt(meanY);
1809        if(RMSY[i]>MTVVY){ MTVVY =RMSY[i];MTVVY_T=RMS_T[i]; }
1810        RMSZ[i]=Math.sqrt(meanZ);
1811        if(RMSZ[i]>MTVVZ){ MTVVZ =RMSZ[i];MTVVZ_T=RMS_T[i]; }
1812        // Almaceno lso resultados:
1813        AccelDataSimple acc=new AccelDataSimple();
1814        acc.setData(RMS_T[i],(float) RMSX[i], (float) RMSY[i], (float) RMSZ[i]);
1815        accelRMSTrend.add(acc);
1816        if (i*100/(RMS_T.length-1)==100){
1817            accelRMSTrend.setMTVVX(MTVVX);
1818            accelRMSTrend.setMTVVY(MTVVY);
1819            accelRMSTrend.setMTVVZ(MTVVZ);
1820            accelRMSTrend.setMTVVX_T(MTVVX_T);
1821            accelRMSTrend.setMTVVY_T(MTVVY_T);
1822            accelRMSTrend.setMTVVZ_T(MTVVZ_T);
1823        }
1824        return i*100/(RMS_T.length-1);
1825    }
1826
1827    //BLUETOOTH:
1828    // Chequeo inicial del estado del bluetooth:
1829    private void CheckBt() {
1830        mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
1831
1832        if (!mBluetoothAdapter.isEnabled()) {
1833            Toast.makeText(getApplicationContext(), "Bluetooth Disabled !",
1834                    Toast.LENGTH_SHORT).show();
1835        }
1836
1837        if (mBluetoothAdapter == null) {
1838            Toast.makeText(getApplicationContext(),
1839                    "Bluetooth null !", Toast.LENGTH_SHORT)
1840                    .show();
1841        }
1842        else{
1843            Toast.makeText(getApplicationContext(),
1844                    "Bluetooth ok !", Toast.LENGTH_SHORT)
1845                    .show();
1846        }
1847    }
```

Main Activity

```
1848    // Procesamiento de la conexxión:
1849    public void Connect() {
1850        Log.d(TAG, address);
1851        BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);
1852        // Toast.makeText(this,"Connecting to ... ",Toast.LENGTH_SHORT).show();
1853        Log.d(TAG, "Connecting to ... " + device);
1854        mBluetoothAdapter.cancelDiscovery();
1855        try {
1856            // COnnecto y habilito Shocket de entrada:
1857            btSocket = device.createRfcommSocketToServiceRecord(MY_UUID);
1858            btSocket.connect();
1859            sfSelecctor.setEnabled(true);
1860            arSelecctor.setEnabled(false);
1861            ard=true;
1862            // Toast.makeText(this,"Connection made.",Toast.LENGTH_SHORT).show();
1863            Log.d(TAG, "Connection made.");
1864        } catch (IOException e) {
1865            try {
1866                btSocket.close();
1867                inStream.close();
1868                Toast.makeText(this,"Unable to end the connection",Toast.LENGTH_SHORT).show();
1869            } catch (IOException e2) {
1870                Log.d(TAG, "Unable to end the connection");
1871            }
1872            Log.d(TAG, "Socket creation failed");
1873        }
1874        // Comienzo el hilo con el registro de datos de datos:
1875        beginListenForData();
1876    }
1877    // Ciclo de vida de la actividad:
1878    @Override
1879    protected void onDestroy() {
1880        super.onDestroy();
1881
1882        try {
1883            btSocket.close();
1884        } catch (IOException e) {
1885        }
1886    }
1887    // Hilo con el registro de datos:
1888    public void beginListenForData()   {
1889        try {
1890            inStream = btSocket.getInputStream();
1891            // Toast.makeText(this,"Input Stream up",Toast.LENGTH_SHORT).show();
1892        } catch (IOException e) {
1893        }
1894        final float X,Y,Z;
1895        Thread workerThread = new Thread(new Runnable()
1896        {
1897            public void run()
1898            {
1899                while(!Thread.currentThread().isInterrupted() && !stopWorker)
1900                {
1901                    try
1902                    {
1903                        // Cuenta el número de bits del registro de entrada que han sido escritos
1904                        int bytesAvailable = inStream.available();
1905                        if(bytesAvailable > 0)
1906                        {
1907                            // Instancio un array de bytes con el tamaño pertinente
1908                            byte[] packetBytes = new byte[bytesAvailable];
1909                            // Leo la entrada:
1910                            inStream.read(packetBytes);
1911                            // Leo cada byte registrado:
1912                            for(int i=0;i<bytesAvailable;i++)
1913                            {
1914                                byte b = packetBytes[i];
1915                                if(b == delimiter)
1916                                {
1917                                    // Proceso:
1918                                    byte[] encodedBytes = new byte[readBufferPosition];
1919                                    // Paso de Bytes a String, para procesar el registro:
1920                                    System.arraycopy(readBuffer, 0, encodedBytes, 0,
1921                                            encodedBytes.length);
1922                                    final String data = new String(encodedBytes, "US-ASCII");
1923                                    readBufferPosition = 0;
1924                                    handler.post(new Runnable()
```

```
                                    {
                                        public void run()
                                        {
                                            // Identificador de inicio del registro:
                                            int kk=data.indexOf("#");
                                            if(ard && kk!=-1) {
                                                try {
                                                    // Método de entrada directa de datos:
                                                    /*x = (float) ((Integer.parseInt(
                                                            data.substring(1, data.indexOf(" "))) *
                                                            (3.3 / 1023) - 1.65) * (9.81 / 0.8));
                                                    y = (float) ((Integer.parseInt(data.substring(
                                                            data.indexOf(" ") + 1,
                                                            data.indexOf(" ",
                                                                    data.indexOf(" ") + 1))) *
                                                            (3.3 / 1023) - 1.65) * (9.81 / 0.8));
                                                    z = (float) ((Integer.parseInt(data.substring(
                                                            data.indexOf(" ",
                                                                    data.indexOf(" ") + 1) + 1,
                                                            data.length() - 2))
                                                            * (3.3 / 1023) - 1.65) * (9.81 / 0.8));
                                                    */
                                                    // Método con media móvil, up now:
                                                    // Inicializo el sumatorio de los tres últimos
                                                    // puntos para la media móvil:
                                                    sumZ=0;
                                                    sumY=0;
                                                    sumX=0;
                                                    // Desplazo las dos primeras posiciones y las
                                                    // sumo:
                                                    for (int i=0; i<2; i++){
                                                        sumZ+=zmean[i];
                                                        zmean[i+1]=zmean[i];
                                                        sumY+=ymean[i];
                                                        ymean[i+1]=ymean[i];
                                                        sumX+=xmean[i];
                                                        xmean[i+1]=xmean[i];
                                                    }
                                                    // Añado el nuebo registro y lo sumo:
                                                    // Cada nuevo registro ha de ser extraido del
                                                    // string de entrada, almazenado tal que :
                                                    // #XXXX YYYY ZZZZ, como se especifica en la
                                                    // memoria:
                                                    xmean[0]=Integer.parseInt(data.substring(1,
                                                            data.indexOf(" ")));
                                                    sumX+=xmean[0];
                                                    ymean[0]=Integer.parseInt(data.substring(
                                                            data.indexOf(" ") + 1,
                                                            data.indexOf(" ",
                                                                    data.indexOf(" ") + 1)));
                                                    sumY+=ymean[0];
                                                    zmean[0]=Integer.parseInt(data.substring(
                                                            data.indexOf(" ",
                                                                    data.indexOf(" ") + 1) + 1,
                                                            data.length() - 2));
                                                    sumZ+=zmean[0];
                                                    // Y por ultimo se procesa mara pasarlo de
                                                    // la lectura analógica digitalizada que es
                                                    // a m/s^2, tal y como se describe en la
                                                    // memoria.
                                                    x = (float) (((sumX/3.0) * (3.3 / 1023) - 1.65)
                                                            * (9.81 / 0.8));
                                                    y = (float) (((sumY/3.0) * (3.3 / 1023) - 1.65)
                                                            * (9.81 / 0.8));
                                                    z = (float) (((sumZ/3.0) * (3.3 / 1023) - 1.65)
                                                            * (9.81 / 0.8));
                                                    // En el caso de Filtrado en tiempo real:
                                                    // NO FUNCIONA--> Futuras revisiones:
                                                    if (startToggleButton.isChecked() &&
                                                            bf.isChecked()) {
                                                        x = (float) butterworthX.filter(x);
                                                        y = (float) butterworthY.filter(y);
                                                        z = (float) butterworthZ.filter(z);
                                                    }
                                                    // Almaceno en los objetos graficados/bles:
                                                    if (startToggleButton.isChecked() &&
                                                            !bf.isChecked()) {
```

Main Activity

```
                                                    graph1LastXValue =
                                                            (System.currentTimeMillis()
                                                            - initialTime);
                                                    mSeries1.appendData(new DataPoint(
                                                                    graph1LastXValue, x),
                                                            true, 350);
                                                    mSeries2.appendData(new DataPoint(
                                                                    graph1LastXValue, y),
                                                            true, 350);
                                                    mSeries3.appendData(new DataPoint(
                                                                    graph1LastXValue, z),
                                                            true, 350);
                                                    if (saveToggleButton.isChecked()) {
                                                        AccelDataSimple acc =
                                                                new AccelDataSimple();
                                                        acc.setData(System.currentTimeMillis()
                                                                - initialRecordedTime, x, y, z);
                                                        accelColection.add(acc);
                                                    }
                                                }
                                            } catch(Exception e){

                                            }
                                        }
                                    }
                                });
                            }
                            else
                            {
                                readBuffer[readBufferPosition++] = b;
                            }
                        }
                    }
                }
                catch (IOException ex)
                {
                    stopWorker = true;
                }
            }
        }
    });

    workerThread.start();
}
}
```

```java
package com.example.jalab.pruebaaccelgraphview;

// Clase para calcular la FAST FURIER TRANSFORMATION (FFT):
public class FFT {

    public static Complex[] fft(Complex[] x) {
        int n = x.length;

        // Instancia inicial:
        if (n == 1) return new Complex[] { x[0] };

        // Longitud de la instancia no es múltiplo de dos:
        if (n % 2 != 0) {
            throw new IllegalArgumentException
                    ("n is not a power of 2");
        }

        // fft de los términos pares
        Complex[] even = new Complex[n/2];
        for (int k = 0; k < n/2; k++) {
            even[k] = x[2*k];
        }
        Complex[] q = fft(even);

        // fft de los términos impares:
        Complex[] odd  = even;  // Instancio a partir del par
        for (int k = 0; k < n/2; k++) {
            odd[k] = x[2*k + 1];
        }
        Complex[] r = fft(odd);

        // Los convino
        Complex[] y = new Complex[n];
        for (int k = 0; k < n/2; k++) {
            double kth = -2 * k * Math.PI / n;
            Complex wk = new Complex(Math.cos(kth), Math.sin(kth));
            y[k]       = q[k].plus(wk.times(r[k]));
            y[k + n/2] = q[k].minus(wk.times(r[k]));
        }
        return y;
    }

    // Transformada inversa:
    public static Complex[] ifft(Complex[] x) {
        int n = x.length;
        Complex[] y = new Complex[n];

        for (int i = 0; i < n; i++) {
            y[i] = x[i].conjugate();
        }

        y = fft(y);

        for (int i = 0; i < n; i++) {
```

```
 55             y[i] = y[i].conjugate();
 56         }
 57
 58         for (int i = 0; i < n; i++) {
 59             y[i] = y[i].scale(1.0 / n);
 60         }
 61
 62         return y;
 63
 64     }
 65
 66     // Convolución circular de x e y
 67     public static Complex[] cconvolve(Complex[] x, Complex[] y) {
 68
 69
 70         if (x.length != y.length) {
 71             throw new IllegalArgumentException
 72                     ("Dimensions don't agree");
 73         }
 74
 75         int n = x.length;
 76
 77         Complex[] a = fft(x);
 78         Complex[] b = fft(y);
 79
 80         Complex[] c = new Complex[n];
 81         for (int i = 0; i < n; i++) {
 82             c[i] = a[i].times(b[i]);
 83         }
 84
 85         return ifft(c);
 86     }
 87
 88
 89     // Convolución linear de x e y
 90     public static Complex[] convolve(Complex[] x, Complex[] y) {
 91         Complex ZERO = new Complex(0, 0);
 92
 93         Complex[] a = new Complex[2*x.length];
 94         for (int i = 0;        i <   x.length; i++) a[i] = x[i];
 95         for (int i = x.length; i < 2*x.length; i++) a[i] = ZERO;
 96
 97         Complex[] b = new Complex[2*y.length];
 98         for (int i = 0;        i <   y.length; i++) b[i] = y[i];
 99         for (int i = y.length; i < 2*y.length; i++) b[i] = ZERO;
100
101         return cconvolve(a, b);
102     }
103
104
105
106 }
107
```

```java
package com.example.jalab.pruebaaccelgraphview;


import java.util.Objects;
// Clase para construir y gestionar números complejos:
public class Complex {
    private final double re;   // Parte real
    private final double im;   // Parte imaginaria

    // Constructor de clase a partir de datos de entrada
    public Complex(double real, double imag) {
        re = real;
        im = imag;
    }
    // Getter de la parte real
    public double getRe(){
        return re;
    }
    // Getter de la parte imaginaria
    public double getIm(){
        return im;
    }

    // Devuelve un String con el complejo:
    public String toString() {
        if (im == 0) return re + "";
        if (re == 0) return im + "i";
        if (im <  0) return re + " - " + (-im) + "i";
        return re + " + " + im + "i";
    }

    // devuelve abs/modulus/magnitude
    public double abs() {
        return Math.hypot(re, im);
    }

    // devuelve ángulo/phase/argument0,
    // normalizado para estar entre -pi y pi
    public double phase() {
        return Math.atan2(im, re);
    }

    // devuelve la suma con el complejo de entrada
    public Complex plus(Complex b) {
        Complex a = this;
        double real = a.re + b.re;
        double imag = a.im + b.im;
        return new Complex(real, imag);
    }

    // devuelve la resta con el complejo de entrada
    public Complex minus(Complex b) {
        Complex a = this;
        double real = a.re - b.re;
```

Complex

```
55          double imag = a.im - b.im;
56          return new Complex(real, imag);
57      }
58
59      // devuelve la multiplicación con el complejo de entrada
60      public Complex times(Complex b) {
61          Complex a = this;
62          double real = a.re * b.re - a.im * b.im;
63          double imag = a.re * b.im + a.im * b.re;
64          return new Complex(real, imag);
65      }
66
67      // devuelve la multiplicación con el escalar de entrada
68      public Complex scale(double alpha) {
69          return new Complex(alpha * re, alpha * im);
70      }
71
72      // devuelve el conjugado del complejo de entrada
73      public Complex conjugate() {
74          return new Complex(re, -im);
75      }
76
77      // devuelve el recíproco de este
78      public Complex reciprocal() {
79          double scale = re*re + im*im;
80          return new Complex(re / scale, -im / scale);
81      }
82
83      // getters alternativos:
84      public double re() { return re; }
85      public double im() { return im; }
86
87      // devuelve la división de complejos
88      public Complex divides(Complex b) {
89          Complex a = this;
90          return a.times(b.reciprocal());
91      }
92
93      // Devuelve la exponencial de este
94      public Complex exp() {
95          return new Complex(Math.exp(re) * Math.cos(im),
96                  Math.exp(re) * Math.sin(im));
97      }
98
99      // devuelve el seno compljeo de este
100     public Complex sin() {
101         return new Complex(Math.sin(re) * Math.cosh(im),
102                 Math.cos(re) * Math.sinh(im));
103     }
104
105     // devuelve el coseno compljeo de este
106     public Complex cos() {
107         return new Complex(Math.cos(re) * Math.cosh(im),
108                 -Math.sin(re) * Math.sinh(im));
```

```
109     }
110
111     // devuelve la tangente compljeo de este
112     public Complex tan() {
113         return sin().divides(cos());
114     }
115
116
117
118     // versión de la suma para dos valore de entrada
119     // (función estática para la clase):
120     public static Complex plus(Complex a, Complex b) {
121         double real = a.re + b.re;
122         double imag = a.im + b.im;
123         Complex sum = new Complex(real, imag);
124         return sum;
125     }
126
127     // compara si dos complejos son iguales
128     public boolean equals(Object x) {
129         if (x == null) return false;
130         if (this.getClass() != x.getClass()) return false;
131         Complex that = (Complex) x;
132         return (this.re == that.re) && (this.im == that.im);
133     }
134
135     // hascodeo de clase:
136     public int hashCode() {
137         return Objects.hash(re, im);
138     }
139 }
140
```

```java
package com.example.jalab.pruebaaccelgraphview;
// Objeto para almacenar UN registro de aceleración
public class AccelDataSimple {

    private double time;
    private float x, y, z;
    // Instancio a cero las variables en el constructor:
    public AccelDataSimple(){
        time=0;
        x=0;
        y=0;
        z=0;
    }
    // Para crear  a partir de datos de entrada:
    public AccelDataSimple create(double t, float x, float y,
                                  float z){
        AccelDataSimple acc=new AccelDataSimple();
        acc.setData(t,x,y,z);
        return acc;
    }
    // Setter de datos:
    public void setData (double time, float x, float y, float z){
        this.time=time;
        this.x=x;
        this.y=y;
        this.z=z;
    }
    // Getters:
    public double getTime(){
        return time;
    }
    public float getX(){
        return x;
    }
    public float getY(){
        return y;
    }
    public float getZ(){
        return z;
    }


}

```

```java
package com.example.jalab.pruebaaccelgraphview;

// Clase dodne almacenaré los datos recogídos por el accelerómetro.
// Alternatva simplificada para revisiones futuras:

public class AccelData {

    private int length;
    private double [][] accel;
    // Columnas: 0- tiemo; 1- accel x; 2- accel y; 3- accel z;
    int cont=0;
    // COsntrucctor: inicializa todo a cero:
    public AccelData(int length){
        this.length=length;
        accel= new double[length][4];
        accel[0][0]=0;
        accel[0][1]=0;
        accel[0][2]=0;
        accel[0][3]=0;
    }
    // Setter de datos
    public void setData (int contAccelPos, double time, float [] acc){

            accel[contAccelPos][0]=time;
            accel[contAccelPos][1]=acc[0];
            accel[contAccelPos][2]=acc[1];
            accel[contAccelPos][3]=acc[2];

    }
    // Getters
    public int getLength() {
        return length;
    }
    public double getTime(int i){
        return accel[i][0];
    }
    public double getX(int i){
        return accel[i][1];
    }
    public double getY(int i){
        return accel[i][2];
    }
    public double getZ(int i){
        return accel[i][3];
    }
}

```

```
1  package com.example.jalab.pruebaaccelgraphview;
2
3  /* Collección de objetos para apilar los AccelDataSimple
4   * -->Funciona como una coleccion de objetos estandar.
5   * Ademas de apilarlos objetos realiza algunas operaciones
6   * como:
7   *      -Identifica los máximos y localiza su posición,
8   *          para el rms_trend
9   *      -Construye un string con los datos almacenados
10  *          en su interior para guardarlos en un txt
11  *          más rápidamente*/
12 public class AccelColection {
13     // Instancio a cero every variable
14     private AccelDataSimple data[]=null;
15     private String name=" ";
16     private String stringData=" ";
17     private int length=0;
18     private double MTVVX=0, MTVVX_T=0, MTVVY_T=0, MTVVY=0, MTVVZ_T=0,
19             MTVVZ=0;
20     private float xSum=0, ySum=0, zSum=0;
21     // Constructor a partir de tamaño inicial dado. Se incrementa
22     // dinámicamente en caso de superar este valor
23     public AccelColection(int initialLength){
24         data= new AccelDataSimple[initialLength];
25         // Inicializo el recorde de datos en string con el encavezado
26         stringData="\t"+"Time"+"\t"+"X"+"\t"+"Y"+"\t"+"Z"+"\n";
27     }
28     // Función para generar los datos necesarios para el RMS_Trend:
29     public void generateMTVV(){
30         double max, maxT, maxy, maxyT, maxz, maxzT;
31         max=0; maxT=0; maxy=0; maxyT=0; maxz=0; maxzT=0;
32         for(int i=0; i<length; i++){
33             if(max< data[i].getX()){ max= data[i].getX(); maxT=
34                     data[i].getTime();}
35             if(maxy< data[i].getY()){ maxy= data[i].getY(); maxyT=
36                     data[i].getTime();}
37             if(maxz< data[i].getZ()){ maxz= data[i].getZ(); maxzT=
38                     data[i].getTime();}
39         }
40         MTVVX=max; MTVVY=maxy; MTVVZ=maxz;
41         MTVVX_T=maxT; MTVVY_T=maxyT; MTVVZ_T=maxzT;
42     }
43     // Función para obtener el tiempo medio del registro
44     public double getTimeMean(){
45         double timeMean=data[length-1].getTime()/length;
46         return timeMean;
47     }
48     // De aquí en adelante, funciones estandar de una colección
49     // y getters/setters
50     public AccelDataSimple get(int i){
51         return data[i];
52     }
53
54     public int length(){
```

```
55          return length;
56      }
57
58      public void set (AccelDataSimple elm, int i)
59      {
60          if( length==data.length )
61          {
62              AccelDataSimple aux[] = data;
63              data = new AccelDataSimple [data.length*2];
64              for(int j=0; j< length; j++)
65              {
66                  data[j]=aux[j];
67              }
68              aux=null;
69          }
70          for( int j=length-1; j>=i; j-- )
71          {
72              data[j+1]=data[j];
73          }
74          data[i]=elm;
75          length++;
76          xSum+=elm.getX();
77          ySum+=elm.getY();
78          zSum+=elm.getZ();
79      }
80
81      public void add(AccelDataSimple acc){
82          set(acc, length);
83          stringData=stringData+"\t"+
84                  String.valueOf(acc.getTime())+"\t"+
85                  String.valueOf(acc.getX())+"\t"+
86                  String.valueOf(acc.getY())+"\t"+
87                  String.valueOf(acc.getZ())+"\n";
88
89      }
90
91      public float getXMean(){
92          return xSum/length;
93      }
94      public float getYMean(){
95          return ySum/length;
96      }
97      public float getZMean(){
98          return zSum/length;
99      }
100
101     public String getName() {
102         return name;
103     }
104
105     public void setName(String name) {
106         this.name = name;
107     }
108
```

AccelColection

```
109     public String getStringAccel(){
110         return stringData;
111     }
112 
113     public double getMTVVX() {
114         return MTVVX;
115     }
116 
117     public double getMTVVY() {
118         return MTVVY;
119     }
120 
121     public double getMTVVZ() {
122         return MTVVZ;
123     }
124 
125     public double getMTVVX_T() {
126         return MTVVX_T;
127     }
128 
129     public void setMTVVX_T(double MTVVX_T) {
130         this.MTVVX_T = MTVVX_T;
131     }
132 
133     public double getMTVVY_T() {
134         return MTVVY_T;
135     }
136 
137     public void setMTVVY_T(double MYVVY_T) {
138         this.MTVVY_T = MYVVY_T;
139     }
140 
141     public double getMTVVZ_T() {
142         return MTVVZ_T;
143     }
144 
145     public void setMTVVZ_T(double MTVVZ_T) {
146         this.MTVVZ_T = MTVVZ_T;
147     }
148 
149     public void setMTVVX(double MTVVX) {
150         this.MTVVX = MTVVX;
151     }
152 
153     public void setMTVVY(double MTVVY) {
154         this.MTVVY = MTVVY;
155     }
156 
157     public void setMTVVZ(double MTVVZ) {
158         this.MTVVZ = MTVVZ;
159     }
160 
161 
162 }
```

XML

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.jalab.pruebaaccelgraphview.MainActivity">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="horizontal"
        tools:ignore="MissingConstraints"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="0dp">

        <LinearLayout
            android:id="@+id/loGraphs"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="3"
            android:orientation="vertical"
            tools:layout_editor_absoluteX="8dp"
            tools:layout_editor_absoluteY="8dp">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/textViewSamplingGraph"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:layout_weight="20"
                    android:text="Sampling:" />

            </LinearLayout>

            <com.jjoe64.graphview.GraphView
                android:id="@+id/graph2"
                android:layout_width="match_parent"
                android:layout_height="0dp"
                android:layout_marginBottom="10dp"
                android:layout_marginLeft="10dp"
                android:layout_weight="1" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal">

                <TextView
                    android:id="@+id/textView2"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="Loaded Data:" />

```

activity_main

```
61              </LinearLayout>
62
63              <com.jjoe64.graphview.GraphView
64                  android:id="@+id/graph"
65                  android:layout_width="match_parent"
66                  android:layout_height="0dp"
67                  android:layout_marginBottom="10dp"
68                  android:layout_marginLeft="-15dp"
69                  android:layout_weight="2"></com.jjoe64.graphview.GraphView>
70
71          </LinearLayout>
72
73          <LinearLayout
74              android:id="@+id/loCom"
75              android:layout_width="0dp"
76              android:layout_height="wrap_content"
77              android:layout_weight="2"
78              android:orientation="vertical">
79
80              <LinearLayout
81                  android:layout_width="match_parent"
82                  android:layout_height="match_parent"
83                  android:layout_weight="1"
84                  android:orientation="horizontal">
85
86                  <LinearLayout
87                      android:layout_width="wrap_content"
88                      android:layout_height="wrap_content"
89                      android:layout_weight="1"
90                      android:orientation="vertical">
91
92                      <LinearLayout
93                          android:layout_width="match_parent"
94                          android:layout_height="wrap_content"
95                          android:layout_weight="1"
96                          android:orientation="horizontal">
97
98                          <LinearLayout
99                              android:layout_width="0dp"
100                             android:layout_height="wrap_content"
101                             android:layout_weight="2"
102                             android:orientation="vertical">
103
104                             <TextView
105                                 android:id="@+id/contpos"
106                                 android:layout_width="match_parent"
107                                 android:layout_height="wrap_content"
108                                 android:layout_marginLeft="10dp"
109                                 android:layout_weight="1"
110                                 android:text="Ejes" />
111
112                             <LinearLayout
113                                 android:layout_width="match_parent"
114                                 android:layout_height="match_parent"
115                                 android:layout_weight="1"
116                                 android:orientation="horizontal">
117
118                                 <CheckBox
119                                     android:id="@+id/xCheckBox"
120                                     android:layout_width="match_parent"
```

activity_main

```
121                                     android:layout_height="wrap_content"
122                                     android:layout_weight="1"
123                                     android:checked="true"
124                                     android:text="x"
125                                     android:theme="@style/BlueCheck" />
126
127                                 <CheckBox
128                                     android:id="@+id/yCheckBox"
129                                     android:layout_width="match_parent"
130                                     android:layout_height="wrap_content"
131                                     android:layout_weight="1"
132                                     android:checked="true"
133                                     android:text="y"
134                                     android:theme="@style/RedCheck" />
135
136                                 <CheckBox
137                                     android:id="@+id/zCheckBox"
138                                     android:layout_width="match_parent"
139                                     android:layout_height="match_parent"
140                                     android:layout_weight="1"
141                                     android:checked="true"
142                                     android:text="z"
143                                     android:theme="@style/BlackCheck" />
144
145                                 <CheckBox
146                                     android:id="@+id/butterFilterCheckButton"
147                                     android:layout_width="match_parent"
148                                     android:layout_height="wrap_content"
149                                     android:layout_weight="1"
150                                     android:text="bF" />
151
152                                 <CheckBox
153                                     android:id="@+id/gCheckBox"
154                                     android:layout_width="match_parent"
155                                     android:layout_height="wrap_content"
156                                     android:layout_weight="1"
157                                     android:text="g" />
158
159                             </LinearLayout>
160
161                         </LinearLayout>
162
163                     </LinearLayout>
164
165                 </LinearLayout>
166
167             </LinearLayout>
168
169             <LinearLayout
170                 android:layout_width="match_parent"
171                 android:layout_height="wrap_content"
172                 android:layout_weight="1"
173                 android:orientation="horizontal">
174
175                 <TextView
176                     android:id="@+id/textViewFrecc"
177                     android:layout_width="wrap_content"
178                     android:layout_height="wrap_content"
179                     android:layout_marginLeft="10dp"
180                     android:text="Frecuencia " />
```

activity_main

```
181
182                 <Spinner
183                     android:id="@+id/frecuenciesSelector"
184                     android:layout_width="match_parent"
185                     android:layout_height="match_parent" />
186
187             </LinearLayout>
188
189             <Space
190                 android:layout_width="match_parent"
191                 android:layout_height="15dp"
192                 android:layout_weight="0.5" />
193
194             <LinearLayout
195                 android:layout_width="match_parent"
196                 android:layout_height="match_parent"
197                 android:layout_weight="1"
198                 android:orientation="horizontal">
199
200                 <Button
201                     android:id="@+id/arSellector"
202                     android:layout_width="match_parent"
203                     android:layout_height="wrap_content"
204                     android:layout_weight="1"
205                     android:text="AR" />
206
207                 <Button
208                     android:id="@+id/sfSelector"
209                     android:layout_width="match_parent"
210                     android:layout_height="wrap_content"
211                     android:layout_weight="1"
212                     android:text="SF" />
213
214                 <ToggleButton
215                     android:id="@+id/startToggleButton"
216                     android:layout_width="match_parent"
217                     android:layout_height="wrap_content"
218                     android:layout_weight="1"
219                     android:text="ToggleButton"
220                     android:textSize="8dp" />
221
222                 <ToggleButton
223                     android:id="@+id/saveToggleButton"
224                     android:layout_width="match_parent"
225                     android:layout_height="wrap_content"
226                     android:layout_weight="1"
227                     android:text="Save"
228                     android:textSize="8dp" />
229             </LinearLayout>
230
231             <LinearLayout
232                 android:layout_width="match_parent"
233                 android:layout_height="match_parent"
234                 android:layout_weight="50"
235                 android:orientation="horizontal">
236
237                 <ListView
238                     android:id="@+id/savedListView"
239                     android:layout_width="match_parent"
240                     android:layout_height="match_parent"
```

activity_main

```
                    android:layout_weight="2" />

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:layout_weight="3"
                    android:orientation="vertical">

                    <LinearLayout
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_weight="1"
                        android:orientation="horizontal">

                        <Button
                            android:id="@+id/loadButton"
                            android:layout_width="0dp"
                            android:layout_height="wrap_content"
                            android:layout_weight="0.5"
                            android:text="L"
                            android:textSize="10dp" />

                        <Button
                            android:id="@+id/saveButton"
                            android:layout_width="0dp"
                            android:layout_height="wrap_content"
                            android:layout_weight="0.5"
                            android:text="S"
                            android:textSize="10dp" />

                    </LinearLayout>

                    <LinearLayout
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_weight="1"
                        android:orientation="horizontal">

                        <Button
                            android:id="@+id/detrendButton"
                            android:layout_width="0dp"
                            android:layout_height="wrap_content"
                            android:layout_weight="0.5"
                            android:text="D"
                            android:textSize="10dp" />

                        <Button
                            android:id="@+id/detrendSaveButton"
                            android:layout_width="0dp"
                            android:layout_height="wrap_content"
                            android:layout_weight="0.5"
                            android:text="S"
                            android:textSize="10dp" />

                    </LinearLayout>

                    <LinearLayout
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_weight="1"
```

activity_main

```
                android:orientation="horizontal">

                <Button
                    android:id="@+id/fftButton"
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="0.5"
                    android:text="F"
                    android:textSize="10dp" />

                <Button
                    android:id="@+id/saveFFTButton"
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="0.5"
                    android:text="S"
                    android:textSize="10dp" />

            </LinearLayout>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:orientation="horizontal">

                <Button
                    android:id="@+id/butterButton"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="B"
                    android:textSize="10dp" />

                <Button
                    android:id="@+id/butterSetButton"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="u"
                    android:textSize="10dp" />

                <Button
                    android:id="@+id/butterSaveButton"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="S"
                    android:textSize="10dp" />
            </LinearLayout>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:orientation="horizontal">

                <Button
                    android:id="@+id/rsmButton"
                    android:layout_width="wrap_content"
```

activity_main

```
361                             android:layout_height="wrap_content"
362                             android:layout_weight="1"
363                             android:text="R"
364                             android:textSize="10dp" />
365
366                         <Button
367                             android:id="@+id/rsmSetButton"
368                             android:layout_width="wrap_content"
369                             android:layout_height="wrap_content"
370                             android:layout_weight="1"
371                             android:text="U"
372                             android:textSize="10dp" />
373
374                         <Button
375                             android:id="@+id/rsmSaveButton"
376                             android:layout_width="wrap_content"
377                             android:layout_height="wrap_content"
378                             android:layout_weight="1"
379                             android:text="S"
380                             android:textSize="10dp" />
381                     </LinearLayout>
382
383                 </LinearLayout>
384
385             </LinearLayout>
386
387         </LinearLayout>
388
389     </LinearLayout>
390
391 </android.support.constraint.ConstraintLayout>
392
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <TextView
            android:id="@+id/textView4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.2"
            android:text="MAC Address :" />

        <EditText
            android:id="@+id/macAddressEt"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.7"
            android:ems="10"
            android:inputType="textPersonName"
             />

        <Button
            android:id="@+id/saveAddress"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.2"
            android:text="SAVE" />

    </LinearLayout>

    <TextView
        android:id="@+id/currentAddress"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Current Address :"
        android:gravity="center"/>

</LinearLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:padding="10dp">


    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_weight="0.7">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="0.4"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/textView"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.5"
                android:gravity="center_vertical"
                android:paddingLeft="10dp"
                android:text="OREDER: (0-9)" />

            <Spinner
                android:id="@+id/orderSpinner"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.5" />

        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="0.3"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/textView3"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.5"
                android:gravity="center_vertical"
                android:paddingLeft="10dp"
                android:text="CUTTOF FREQ (hz):"
                android:textSize="12dp"/>

            <EditText
                android:id="@+id/cutoffEditText"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.5"
                android:ems="10"
                android:inputType="numberDecimal"
```

input_butter_dialog

```
                android:textSize="10dp" />
        </LinearLayout>

    </LinearLayout>

    <Button
        android:id="@+id/applyButter"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.3"
        android:text="APPLY" />

</LinearLayout>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:padding="10dp">

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.7"
        android:orientation="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/wintext"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.2"
                android:text="Win :" />

            <EditText
                android:id="@+id/winEditText"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.8"
                android:ems="10"
                android:inputType="numberDecimal"
                />

        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/textView5"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.2"
                android:text="T2 :" />

            <EditText
                android:id="@+id/t2EditText"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.8"
                android:ems="10"
                android:inputType="numberDecimal"
                 />
        </LinearLayout>
    </LinearLayout>

```

rms_trend_data_dialog

```
61      <Button
62          android:id="@+id/rsmtrednApplyButton"
63          android:layout_width="0dp"
64          android:layout_height="match_parent"
65          android:layout_weight="0.3"
66          android:text="APPLY" />
67
68  </LinearLayout>
```

```xml
<resources>
    <string name="app_name">Prueba AccelGraphView</string>

    <string-array name="frecuencies">
        <item>Max</item>
        <item>5 ms</item>
        <item>10 ms</item>
        <item>50 ms</item>
        <item>75 ms</item>
        <item>100 ms</item>
        <item>150 ms</item>
    </string-array>
    <string-array name="order">
        <item>1</item>
        <item>2</item>
        <item>3</item>
        <item>4</item>
        <item>5</item>
        <item>6</item>
        <item>7</item>
        <item>8</item>
        <item>9</item>
    </string-array>

</resources>

```

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

    <style name="BlueCheck">
        <item name="colorAccent">#182ce6</item>
    </style>

    <style name="RedCheck">
        <item name="colorAccent">#de1a13</item>
    </style>

    <style name="BlackCheck">
        <item name="colorAccent">#000000</item>
    </style>


</resources>

```

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.jalab.pruebaaccelgraphview">

    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:screenOrientation="landscape"
            android:configChanges="keyboardHidden|orientation|screenSize">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

CÓDIGO ARDUINO

```
     ude <SoftwareSerial.h>

     areSerial BTserial(10, 11); // RX | TX

     ensorPin0 = A0;  // Eje Z
     ensorPin1 = A1;  // Eje y
int sensorPin2 = A2;  // Eje x
int sensorValue0 = 0;
int sensorValue1 = 0;
int sensorValue2 = 0;
/* Parámetros para calcular la media móvil desde el
int zmean[3]={0,0,0};
int ymean[3]={0,0,0};
int xmean[3]={0,0,0};
int sumZ, sumY, sumX;*/

void setup() {
  BTserial.begin(9600);
}

void loop() {

  sensorValue0 = analogRead(sensorPin0); // Lectura analógica del eje Z
  sensorValue1 = analogRead(sensorPin1); // Lectura analógica del eje y
  sensorValue2 = analogRead(sensorPin2); // Lectura analógica del eje x
  /* Inicializo a cero los valores de sumatorio de los ejes, para la media movil:
   * Pra calcular la media movil de los valores en el arduino directamente
   *  Decido enviar el valores de los sensores directamente porque los cálculos
   *  se hacen más rápido en los otros softwares, con lo que se gana tiempo.
  sumZ=0;
  sumY=0;
  sumX=0;
  for (int i=0; i<2; i++){
    sumZ+=zmean[i];
    zmean[i+1]=zmean[i];
    sumY+=ymean[i];
    ymean[i+1]=ymean[i];
    sumX+=xmean[i];
    xmean[i+1]=xmean[i];
  }
  zmean[0]=sensorValue0;
  sumZ+=zmean[0];
  ymean[0]=sensorValue1;
  sumY+=ymean[0];
  xmean[0]=sensorValue2;
  sumX+=xmean[0];*/


  BTserial.print("#");
  BTserial.print(sensorValue2);
  //BTserial.print(sumX/3.0);
  BTserial.print(" ");
  BTserial.print(sensorValue1);
  //BTserial.print(sumY/3.0);
  BTserial.print(" ");
  BTserial.print(sensorValue0);
 //BTserial.print(sumZ/3.0);
  BTserial.println(";");

}
```

CÓDIGO MATLAB

```
function varargout = gui_app(varargin)
% GUI_APP MATLAB code for gui_app.fig
%      GUI_APP, by itself, creates a new GUI_APP or raises the existing
%      singleton*.
%
%      H = GUI_APP returns the handle to a new GUI_APP or the handle to
%      the existing singleton*.
%
%      GUI_APP('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in GUI_APP.M with the given input arguments.
%
%      GUI_APP('Property','Value',...) creates a new GUI_APP or raises the
%      existing singleton*.  Starting from the left, property value pairs
%      are applied to the GUI before gui_app_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
%      application stop.  All inputs are passed to gui_app_OpeningFcn via
%      varargin.
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help gui_app

% Last Modified by GUIDE v2.5 12-May-2018 11:37:34

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @gui_app_OpeningFcn, ...
                   'gui_OutputFcn',  @gui_app_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before gui_app is made visible.
function gui_app_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to gui_app (see VARARGIN)

% Carpeta por defecto donde se almacenan los archivos--> se puede cambiar
% una vez se accede al programa
```

```
filepath='C:\Users\patri\Desktop\TFM Juan\Excels_app';
% Se almacena las variables correspondientes en el objeto handles
% para enviarlo a las diferentes partes del programa
handles.filepath=filepath;
filename=' ';
handles.filename=filename;
handles.loadedfile=' ';
% Envio estos datos a la función LoadListBox, que carga en la Listview
% loas archivos .txt encontrados
LoadListBox(handles,filepath);

handles.output = hObject;
% Establezco los parametros de inicialización de los elementos de la
% interfáz gráfica

set(handles.xradiobutton, 'Value', 1);
set(handles.yradiobutton, 'Value', 1);
set(handles.zradiobutton, 'Value', 1);
set(handles.xDumpingRB, 'Value', 1);
set(handles.yDumpingRB, 'Value', 1);
set(handles.zDumpingRB, 'Value', 1);
set(handles.rsmbutton, 'Enable', 'off');
set(handles.resampbutton, 'Enable', 'off');
set(handles.fftbutton, 'Enable', 'off');
set(handles.detrendbutton, 'Enable', 'off');
set(handles.butterbutton, 'Enable', 'off');

set(handles.startsfbutton, 'Enable', 'off');
set(handles.stopsfbutton, 'Enable', 'off');
set(handles.savebutton, 'Enable', 'off');

set(handles.startardbutton, 'Enable', 'off');
set(handles.stopardbutton, 'Enable', 'off');
set(handles.saveardbutton, 'Enable', 'off');
if get(handles.smfradiobutton,'Value')==1
    set(handles.cntardtogglebutton, 'Enable', 'off');
else
     set(handles.cntsftogglebutton, 'Enable', 'off');
end
set(handles.orderbuttertext, 'String', ['Order: ',...
    get(handles.orderbutteredit,'String')]);
handles.orderbutter=5;
set(handles.t2edit, 'String', num2str(0.25));

set(handles.tminEdit, 'String', num2str(0));
set(handles.tmaxEdit, 'String', num2str(0));

set(handles.t2text, 'String', ['T2: ',get(handles.t2edit,'String')]);
handles.T2=str2double(get(handles.t2edit,'String'));
set(handles.winedit, 'String', num2str(0.5));
set(handles.wintext, 'String', ['Win: ',get(handles.winedit,'String')]);
handles.win=str2double(get(handles.winedit,'String'));
set(handles.rstedit, 'String', num2str(10));
set(handles.rstimetext, 'String', ['Tiempo de remuestreo (ms): '...
    ,get(handles.rstedit,'String'),' ms']);
```

```
handles.rst=str2double(get(handles.rstedit,'String'));
handles.Trsm=0;
handles.Xrsm=0;
handles.Yrsm=0;
handles.Zrsm=0;
handles.tminCut=0;
handles.tmaxCut=0;
% Variables globales para controlar los bucles que permiten el muestreo en
% tiempo real y su visualización en pantalla:
global KeepRunning;
global a;
a=0;
KeepRunning=1;
% Se guarda y envía toda la información determinada hasta ahorá. EL
% comportamiento de esto es análogo a un Intent en Java, y permite
% enviar/recivir información de un lugar a otro sin que esta sea global.
guidata(hObject, handles);

% UIWAIT makes gui_app wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = gui_app_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on selection change in listbox_excel.
function listbox_excel_Callback(hObject, eventdata, handles)
% hObject    handle to listbox_excel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
contents = cellstr(get(hObject,'String'));
data.filename=contents{get(hObject,'Value')};

set(handles.loaded, 'String', data.filename);
guidata(hObject,data);
% Hints: contents = cellstr(get(hObject,'String')) returns listbox_excel
%        contents as cell array contents{get(hObject,'Value')} returns
%        selected item from listbox_excel
% --- Executes during object creation, after setting all properties.
function listbox_excel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox_excel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

% --- Para cargar los excels de la carpeta al arrancar el programa
function handles = LoadListBox(handles, filepath)
try
    yourFolder = filepath;
    % Load up the listbox.
    ListOfFilenames = {};
    dirListing = dir([yourFolder'/*.txt*']);
    for Index = 1:length(dirListing)
        baseFileName = dirListing(Index).name;
        ListOfFilenames = [ListOfFilenames baseFileName];
    end
    set(handles.listbox_excel,'string', ListOfFilenames);
catch ME
    errorMessage = sprintf(...
        'Error in LoadListBox().\nThe error reported by MATLAB is:\n\n%s'...
        , ME.message);
    uiwait(warndlg(errorMessage));
end
return;

% --- Función para seleccionar las carpetas donde esten nuestros txt.
function folderButton_Callback(hObject, eventdata, handles)
% hObject    handle to folderButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filepath = uigetdir('C:\');
if filepath~=0
    data.filepath=filepath;
end
LoadListBox(handles, data.filepath);
guidata(hObject,data);

% --- Botton para cargar los archivos
function loadExcelButton_Callback(hObject, eventdata, handles)
% hObject    handle to loadExcelButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
if handles.filename~=' '
%     En caso de querer modificar esto y hacerlo con archivos excel (mucho
%     más lento el abrirlos)
%     set(handles.loadingText, 'String', 'Loading...');
%     data.T=xlsread([data.filepath,'\',data.filename],'B:B')
%     data.X=xlsread([data.filepath,'\',data.filename],'C:C')
%     data.Y=xlsread([data.filepath,'\',data.filename],'D:D')
%     data.Z=xlsread([data.filepath,'\',data.filename],'E:E')
%     set(handles.loadingText, 'String', 'Loaded!');
data.loadedfile=handles.filename;
% Se extraen los datos del txt y se almacenan en su vector correspondiente
[T, X, Y, Z] = textread([data.filepath,'\',data.filename],...
    '%f %f %f %f', 'headerlines', 1);
```

```
data.T=T;
data.X=X;
data.Y=Y;
data.Z=Z;
% Comprobaciones para ver si los registros de medición estan o no
% equiespaciado. Enf uncion de si lo estan o no se habilitarán unos
% comandos u otros:

if T(2)-T(1)==T(3)-T(2) && T(2)-T(1)==T(4)-T(3) && T(2)-T(1)==T(5)-T(4)...
        && T(2)-T(1)==T(6)-T(5) && T(2)-T(1)==T(7)-T(6)...
        && T(2)-T(1)==T(9)-T(8) && T(2)-T(1)==T(10)-T(9)
    set(handles.rsmbutton,'Enable', 'on');
    set(handles.fftbutton,'Enable', 'on');
    set(handles.butterbutton,'Enable', 'on');
    set(handles.detrendbutton,'Enable', 'on');
    set(handles.resampbutton,'Enable', 'off');
    data.ts=T(2)-T(1);
    set(handles.tstext,'String',['Tiempo de muestreo :  ',...
        num2str(data.ts),' ms' ]);
    set(handles.tsmean,'String', 'Tiempo de muestreo (media) : ');
    guidata(hObject,data);

else
    set(handles.resampbutton,'Enable', 'on');
    set(handles.rsmbutton,'Enable', 'off');
    set(handles.fftbutton,'Enable', 'off');
    set(handles.detrendbutton,'Enable', 'off');
    set(handles.butterbutton,'Enable', 'off');
    guidata(hObject,data);
    set(handles.tsmean,'String', ['Tiempo de muestreo (media) : '...
        ,num2str(tsmeanFnc(hObject)),' ms']);
    set(handles.tstext,'String','Tiempo de muestreo :  ');
end

plotFnc(hObject,handles);

end

% Función para representar en la gráfica los datos almacenados hasta ahora:
function plotFnc(hObject,handles)
data=guidata(hObject);
if data.Xrsm==0
    if get(handles.xradiobutton,'Value')==1 && ...
            get(handles.yradiobutton,'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.T,data.Y,'r'...
            ,data.T,data.Z,'k');
    elseif get(handles.yradiobutton,'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.Y,'r',data.T,data.Z,'k');
    elseif get(handles.xradiobutton,'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.T,data.Z,'k');
    elseif get(handles.xradiobutton,'Value')==1 && ...
            get(handles.yradiobutton,'Value')==1
```

```
        plot(handles.axes1,data.T,data.X,'b',data.T,data.Y,'r');
    elseif get(handles.xradiobutton, 'Value')==1
        plot(handles.axes1,data.T,data.X,'b');
    elseif get(handles.yradiobutton, 'Value')==1
        plot(handles.axes1,data.T,data.Y,'r');
    else
        plot(handles.axes1,data.T,data.Z,'k');
    end
%     Si se trata del Rms:
else
    if get(handles.xradiobutton, 'Value')==1 && ...
            get(handles.yradiobutton,'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.Trsm,data.Xrsm,'b:',...
            data.TMTVVX,data.MTVVX,'bo'...
            ,data.T,data.Y,'r',data.Trsm,data.Yrsm,'r:',...
            data.TMTVVY,data.MTVVY,'ro'...
            ,data.T,data.Z,'k',data.Trsm,data.Zrsm,'k:',...
            data.TMTVVZ,data.MTVVZ,'ko');
    elseif get(handles.yradiobutton, 'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,...
            data.T,data.Y,'r',data.Trsm,data.Yrsm,'r:',...
            data.TMTVVY,data.MTVVY,'bo'...
            ,data.T,data.Z,'k',data.Trsm,data.Zrsm,'k:',...
            data.TMTVVZ,data.MTVVZ,'ko');
    elseif get(handles.xradiobutton, 'Value')==1 && ...
            get(handles.zradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.Trsm,data.Xrsm,'b:',...
            data.TMTVVX,data.MTVVX,'bo'...
            ,data.T,data.Z,'k',data.Trsm,data.Zrsm,'k:',...
            data.TMTVVZ,data.MTVVZ,'ko');
    elseif get(handles.xradiobutton, 'Value')==1 && ...
            get(handles.yradiobutton,'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.Trsm,data.Xrsm,'b:',...
            data.TMTVVX,data.MTVVX,'bo'...
            ,data.T,data.Y,'r',data.Trsm,data.Yrsm,'r:',...
            data.TMTVVY,data.MTVVY,'ro');
    elseif get(handles.xradiobutton, 'Value')==1
        plot(handles.axes1,data.T,data.X,'b',data.Trsm,data.Xrsm,'b:',...
            data.TMTVVX,data.MTVVX,'bo');
    elseif get(handles.yradiobutton, 'Value')==1
        plot(handles.axes1,data.T,data.Y,'b',data.Trsm,data.Yrsm,'r:',...
            data.TMTVVY,data.MTVVY,'ro');
    else
        plot(handles.axes1,data.T,data.Z,'b',data.Trsm,data.Zrsm,'k:',...
            data.TMTVVZ,data.MTVVZ,'ko');

    end
    data.Trsm=0;
    data.Xrsm=0;
    data.Yrsm=0;
    data.Zrsm=0;
    guidata(hObject,data);
end
```

```
% --- Radio button para eje X.
function xradiobutton_Callback(hObject, eventdata, handles)
% hObject    handle to xradiobutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
if data.loadedfile~=' '
    plotFnc(hObject,handles);
end
% Hint: get(hObject,'Value') returns toggle state of xradiobutton

% --- Radio button para eje Y.
function yradiobutton_Callback(hObject, eventdata, handles)
% hObject    handle to yradiobutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
if data.loadedfile~=' '
    plotFnc(hObject,handles);
end
% Hint: get(hObject,'Value') returns toggle state of yradiobutton

% --- ERadio button para eje Z.
function zradiobutton_Callback(hObject, eventdata, handles)
% hObject    handle to zradiobutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
if data.loadedfile~=' '
    plotFnc(hObject,handles);
end
% Hint: get(hObject,'Value') returns toggle state of zradiobutton

% --- Calculo de la FFT.
function fftbutton_Callback(hObject, eventdata, handles)
% hObject    handle to fftbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filename=strrep(data.filename,'.txt','.FFT.txt');
data.filename=filename;
set(handles.rsmbutton,'Enable','off');

if data.loadedfile~=' '
    n = length(data.X);
    Ts = data.T(2)-data.T(1);   % Tiempo de muestreo en ms
    Fs = 1000/Ts;     % Frecuencia en hz
    NFFT = 2^nextpow2(n);  % Siguiente multiplo de 2 (respecto a n)
    xfft = fft(data.X,NFFT)/n;
    yfft = fft(data.Y,NFFT)/n;
    zfft = fft(data.Z,NFFT)/n;
    f = Fs/2*linspace(0,1,NFFT/2+1);
    Iv = 1:length(f);
    Xfft = 2*abs(xfft(Iv));
```

```
    Yfft = 2*abs(yfft(Iv));
    Zfft = 2*abs(zfft(Iv));
    % Almaceno
    data.X=Xfft;
    data.Y=Yfft;
    data.Z=Zfft;
    data.T=f;

    guidata(hObject,data);
    plotFnc(hObject,handles);
    set(handles.fftbutton, 'Enable', 'off');
end

% --- Resampleo de los datos guardadods a una nueva frecuencia.
function resampbutton_Callback(hObject, eventdata, handles)
% hObject    handle to resampbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
if data.loadedfile~=' '
    disp(data.rst);
    filename=strrep(data.filename,'.txt','.RESAMPLED.txt');
    data.filename=filename;
    tsin = timeseries([data.X data.Y data.Z],data.T');
    timeXresampled=(data.T(1):data.rst:data.T(size(data.T)))';
    tsout = resample(tsin,timeXresampled');
%       Almaceno los datos resampleados:
    data.T=timeXresampled;
    data.X=tsout.Data(:,1);
    data.Y=tsout.Data(:,2);
    data.Z=tsout.Data(:,3);
%     Habilito/desabilito lo que corresponda del programa
    set(handles.rsmbutton, 'Enable', 'on');
    set(handles.fftbutton, 'Enable', 'on');
    set(handles.detrendbutton, 'Enable', 'on');
    set(handles.butterbutton, 'Enable', 'on');
    set(handles.resampbutton, 'Enable', 'off');
    data.ts=data.T(2)-data.T(1);
    set(handles.tstext,'String',['Tiempo de muestreo :  ',...
        num2str(data.ts), ' ms' ]);
    set(handles.tsmean,'String', 'Tiempo de muestreo (media) : ');
    guidata(hObject,data);
    % Represento:
    plotFnc(hObject,handles);
end

% --- Cálculo de la RMS Trend.
function rsmbutton_Callback(hObject, eventdata, handles)
% hObject    handle to rsmbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filename=strrep(data.filename,'.txt','.RMSTrend.txt');
data.filename=filename;
set(handles.fftbutton,'Enable','off');
```

```
set(handles.rsmbutton, 'Enable','off');
fs = 1000/data.ts;
Ndat = round(data.win*fs); % Número de datos para la RMS_trend

Npts = round(data.T2*fs); % Número de puntos en cada cálculo.
% Vector posición de los tiempos donde finaliza la ventana de cálculo:
posi = Ndat:Npts:length(data.X);
% Vector con los tiempos de cada valor de la media
RMS_T = data.T(posi)-data.win/2;
RMSX = zeros(length(RMS_T),1);
RMSY = zeros(length(RMS_T),1);
RMSZ = zeros(length(RMS_T),1);
% Calculo cada una de las medias
for i = 1:length(RMSX)
    RMSX(i) = sqrt(mean(data.X(posi(i)-Ndat+1:posi(i)).^2));
    RMSY(i) = sqrt(mean(data.Y(posi(i)-Ndat+1:posi(i)).^2));
    RMSZ(i) = sqrt(mean(data.Z(posi(i)-Ndat+1:posi(i)).^2));
end
% Los datos correspodnietes a los valores máximos y al tiempo en el que se
% produce:
data.MTVVX = max(RMSX);
Ix=find(RMSX==data.MTVVX);
data.TMTVVX = RMS_T(Ix(1));

data.MTVVY = max(RMSY);
Iy=find(RMSY==data.MTVVY);
data.TMTVVY = RMS_T(Iy(1));

data.MTVVZ = max(RMSZ);
Iz=find(RMSZ==data.MTVVZ);
data.TMTVVZ = RMS_T(Iz(1));
% Almeceno y represento lo calculado
data.Trsm=data.T;
data.Xrsm=data.X;
data.Yrsm=data.Y;
data.Zrsm=data.Z;
data.T=RMS_T;
data.X=RMSX;
data.Y=RMSY;
data.Z=RMSZ;
set(handles.mtvvxtext,'String', ['X : ',num2str(data.MTVVX)]);
set(handles.mtvvytext,'String', ['Y : ',num2str(data.MTVVY)]);
set(handles.mtvvztext,'String', ['Z : ',num2str(data.MTVVZ)]);
guidata(hObject,data);
plotFnc(hObject,handles);

function t2edit_Callback(hObject, eventdata, handles)
% hObject    handle to t2edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.t2edit,'String');
if isnan(str2double(str))
    set(handles.t2edit,'string','0.25');
    warndlg('Input must be numerical','Error');
```

```
else
    data.T2=str2double(str);
end
set(handles.t2text, 'String', ['T2: ',num2str(data.T2)]);
guidata(hObject,data);
% Hints: get(hObject,'String') returns contents of t2edit as text
%        str2double(get(hObject,'String')) returns contents of t2edit as a double

% --- Executes during object creation, after setting all properties.
function t2edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to t2edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function winedit_Callback(hObject, eventdata, handles)
% hObject    handle to winedit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.winedit,'String');
if isnan(str2double(str))
    set(handles.winedit,'string','0.25');
    warndlg('Input must be numerical','Error');
else
    data.win=str2double(str);
end
set(handles.wintext, 'String', ['Win: ',num2str(data.win)]);
guidata(hObject,data);
% Hints: get(hObject,'String') returns contents of winedit as text
%        str2double(get(hObject,'String')) returns contents of winedit
%        as a double
% --- Executes during object creation, after setting all properties.
function winedit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to winedit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% Cálculo de la media de tiempo de muestreo.
function Tsm= tsmeanFnc(hObject)
data=guidata(hObject);
T=data.T;
```

```
Tm=zeros(length(data.T)-1,1);
for i=2:length(data.T)
    Tm(i-1)=T(i)-T(i-1);
end
Tsm=mean(Tm);

function rstedit_Callback(hObject, eventdata, handles)
% hObject    handle to rstedit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rstedit as text
%        str2double(get(hObject,'String')) returns contents of rstedit
%        as a double
data=guidata(hObject);
str=get(handles.rstedit,'String');
if isnan(str2double(str))
    set(handles.rstedit,'string','10');
    warndlg('Input must be numerical','Error');
else
    data.rst=str2double(str);
end
set(handles.rstimetext, 'String', ['Tiempo de remuestreo (ms): '...
    ,num2str(str),' ms']);
guidata(hObject,data);

% --- Executes during object creation, after setting all properties.
function rstedit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rstedit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Detrend, realizado mediante el propio comando de matlab.
function detrendbutton_Callback(hObject, eventdata, handles)
% hObject    handle to detrendbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filename=strrep(data.filename,'.txt','.DETREND.txt');
data.filename=filename;
set(handles.detrendbutton, 'Enable','off');
X=detrend(data.X);
Y=detrend(data.Y);
Z=detrend(data.Z);
data.X=X;
data.Y=Y;
data.Z=Z;
```

```
guidata(hObject,data);
plotFnc(hObject,handles);


% --- Función no habilitada.
function smoothbutton_Callback(hObject, eventdata, handles)
% hObject    handle to smoothbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
X=smooth(data.X);
Y=smooth(data.Y);
Z=smooth(data.Z);
data.X=X;
data.Y=Y;
data.Z=Z;
guidata(hObject,data);
plotFnc(hObject,handles);


% --- Funcion para conectar el smartphone.
function cntsftogglebutton_Callback(hObject, eventdata, handles)
% hObject    handle to cntsftogglebutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
if get(handles.cntsftogglebutton,'Value')==1

    set(handles.startsfbutton,'Enable', 'on');
%     La primera vez que se ejecuta este comando debe introducirse una
%     contraseña, detrás del on, que será la misma que se introduzca en el
%     movil. Por lo tanto, la primera vez que se inicie deverá hacerse "a
%     mano". una vez hecho ya funcionara con normalidad.
    connector on

else
    set(handles.startsfbutton,'Enable', 'off');
    set(handles.stopsfbutton,'Enable', 'off');
    connector off
    disp('Connector off');
end

% Hint: get(hObject,'Value') returns toggle state of cntsftogglebutton


% --- Función para detener el bucle de muestreo en tiempo real.
function stopsfbutton_Callback(hObject, eventdata, handles)
% hObject    handle to stopsfbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.startsfbutton,'Enable','on');
set(handles.stopsfbutton,'Enable','off');
set(handles.savebutton, 'Enable', 'on');
global KeepRunning;

KeepRunning=0;
```

```
% Hint: get(hObject,'Value') returns toggle state of stopsfbutton



% --- Función de toma de datos en tiempo real mediante el Smartphone.
function startsfbutton_Callback(hObject, eventdata, handles)
% hObject    handle to startsfbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Inicializo lo que corresponda:
global KeepRunning;
data=guidata(hObject);
data.T=0;
data.X=0;
data.Y=0;
data.Z=0;
set(handles.rsmbutton, 'Enable', 'off');
set(handles.resampbutton, 'Enable', 'off');
set(handles.fftbutton, 'Enable', 'off');
set(handles.detrendbutton, 'Enable', 'off');
set(handles.butterbutton, 'Enable', 'off');
set(handles.startsfbutton, 'Enable', 'off');
set(handles.stopsfbutton, 'Enable', 'off');
set(handles.savebutton, 'Enable', 'off');

cla
KeepRunning=1;

try
    %     Declaro e inicilo las animatedline donde se apilarán los puntos
    %     de cada eje
    hx = animatedline('Color','b', 'Parent', handles.axes1);
    hy = animatedline('Color','r', 'Parent', handles.axes1);
    hz = animatedline('Color','k', 'Parent', handles.axes1);
    set(handles.axes1,'XLim',[0 5],'YGrid','on');
    %    Inicio el objeto spamrtphone, con sus propiedades:
    m=mobiledev; % Creo el objeto que representa el móvil conectado
    m.SampleRate = 100; % Determino la máxima frecuencia
    m.AccelerationSensorEnabled = 1;% Activo el acelerómetro
    m.Logging = 1; % Establezco la conexión
    pause(0.2);

    t1=0;
    t2=0;
    tm=zeros(1,10);

    set(handles.startsfbutton,'Enable','off');
    set(handles.stopsfbutton,'Enable','on');
    set(handles.savebutton, 'Enable', 'off');
    % Inicializo los registros:
    try
        [a, t] = accellog(m);% Leo el registro cada vez que instancio esto
        l=length(a); % Inicializo el tamaño del registro
    %     Comienzo a ejecutar el bucle de registro
```

```
        tic % Inicio el registro temporal
        while (KeepRunning)
    %       Límites variables del gráfico:
            if toc>5
            set(handles.axes1,'XLim',[toc-5,toc]);
            end
    %       Se almacena y representan los puntos:
            [a, t] = accellog(m);% Leo
            try
                if length(a)> l
                    t2=toc;% Tiempo final de los registros
%                     tm=t1:(t2-t1)/(length(a)-l):t2-(t2-t1)/(length(a)-l);
%                     Construyo el vector con los tiempos entre envíos,
%                     equiespaciandolo con el n° de registroe entre envíos
                    tm=linspace(t1,t2,1+(length(a)-l))';
                    % Alaceno los datos
                    addpoints(hx,tm(2:end),...
                        a(l+1:length(a),1))
                    addpoints(hy,tm(2:end),...
                        a(l+1:length(a),2))
                    addpoints(hz,tm(2:end),...
                        a(l+1:length(a),3))
                    % Declaro el tiempo inicial del siguiente registro
                    t1=toc;
                    l=length(a);
                    data.hz=hz;
                    data.hy=hy;
                    data.hx=hx;
                    guidata(hObject,data);
                    drawnow limitrate
                end
            catch
            end
            pause(0.00005);
        end
    catch
    end

catch
    set(handles.startsfbutton,'Enable','on');
    set(handles.stopsfbutton,'Enable','off');
    set(handles.savebutton, 'Enable', 'on');
end


% Finalización de registro en tiempo real:
function figure1_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global KeepRunning;
KeepRunning=0;
pause(1);
```

```
% Hint: delete(hObject) closes the figure
delete(hObject);


% --- Guardado del registro del smartphone.
function savebutton_Callback(hObject, eventdata, handles)
% hObject    handle to savebutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.savebutton, 'Enable', 'off');
data=guidata(hObject);
% Por defecto el nombre del archivo será la fecha y hora del guardado
c=clock;
namefile=[num2str(c(1)),'-',num2str(c(2)),'-',...
    num2str(c(3)),'-',num2str(c(4)),'.',num2str(c(5)),'.',...
    num2str(round(c(6))),'.txt'];
data.filename=namefile;
[t1,z]=getpoints(data.hz);
[t2,y]=getpoints(data.hy);
[t,x]=getpoints(data.hx);
data.T=t';
data.X=x';
data.Y=y';
data.Z=z';
% Por como se reciven los datos, el primer registro suele estar
% incompleto, con lo que para que no haya error al indexar la matriz que
% introduciremos en el txt, ajusto los vectores.
L=[length(data.T) , length(data.X),length(data.Y),length(data.Z)];
if L(1)~=L(2) || L(1)~=L(3) || L(1)~=L(4)
    data.T=data.T(1:min(L));
    data.X=data.X(1:min(L));
    data.Y=data.Y(1:min(L));
    data.Z=data.Z(1:min(L));
end
A=[data.T'.*1000; data.X'; data.Y'; data.Z'];
fileID = fopen([data.filepath,'\',data.filename],'w');
fprintf(fileID,'\t %s \t %s \t %s \t %s \n','T','X','Y','Z');
fprintf(fileID,'\t %f \t %f \t %f \t %f \n',A);
fclose(fileID);
% Guardo y refresco la lista con los archivos de texto
guidata(hObject,data);
LoadListBox(handles,data.filepath);


% Edit Text correspondiente al orden del filtro butterworth
function orderbutteredit_Callback(hObject, eventdata, handles)
% hObject    handle to orderbutteredit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.orderbutteredit,'String');
if isnan(str2double(str)) || str2double(str)~=floor(str2double(str))..
        || str2double(str)<0
    set(handles.orderbutteredit,'string','5');
    warndlg('Input must be numerical and integer','Error');
```

```
else
    data.orderbutter=str2double(str);
    set(handles.orderbuttertext,'String', ['Order: '...
    ,num2str(data.orderbutter)]);
end
% set(handles.orderbuttertext, 'String', ['Order: '...
%     ,num2str(str)]);
guidata(hObject,data);
% Hints: get(hObject,'String') returns contents of orderbutteredit as text
%        str2double(get(hObject,'String')) returns contents of
%        orderbutteredit as a double

% --- Executes during object creation, after setting all properties.
function orderbutteredit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to orderbutteredit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function fcbutteredit_Callback(hObject, eventdata, handles)
% hObject    handle to fcbutteredit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.fcbutteredit,'String');
if isnan(str2double(str)) || str2double(str)<0
    set(handles.fcbutteredit,'string',' ');
    warndlg('Input must be numerical ','Error');
else
    data.fcbutter=str2double(str);
    set(handles.fcbuttertext,'String', ['Cutoff frec.: '...
    ,num2str(data.fcbutter),' hz']);
end
% set(handles.orderbuttertext, 'String', ['Order: '...
%     ,num2str(str)]);
guidata(hObject,data);
% Hints: get(hObject,'String') returns contents of fcbutteredit as text
%        str2double(get(hObject,'String')) returns contents of fcbutteredit
%        as a double


% --- Executes during object creation, after setting all properties.
function fcbutteredit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to fcbutteredit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Ejecución del filtro butterworth mediante la funcion de matlab.
function butterbutton_Callback(hObject, eventdata, handles)
% hObject    handle to butterbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filename=strrep(data.filename,'.txt','.BUTTER.txt');
data.filename=filename;
set(handles.butterbutton,'Enable','off');
fm=1000/data.ts;
[but_b, but_a] = butter(data.orderbutter, data.fcbutter/(fm/2));
data.X=filter(but_b, but_a, data.X);
data.Y=filter(but_b, but_a, data.Y);
data.Z=filter(but_b, but_a, data.Z);
guidata(hObject,data);
plotFnc(hObject,handles);


% --- Guardar arhivo modificado.
function savefilebutton_Callback(hObject, eventdata, handles)
% hObject    handle to savefilebutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.savebutton, 'Enable', 'off');
data=guidata(hObject);
A=[data.T'; data.X'; data.Y'; data.Z'];
fileID = fopen([data.filepath,'\',data.filename],'w');
fprintf(fileID,'\t %s \t %s \t %s \t %s \n','T','X','Y','Z');
fprintf(fileID,'\t %f \t %f \t %f \t %f \n',A);
fclose(fileID);

guidata(hObject,data);
LoadListBox(handles,data.filepath);

% --- Executes during object creation, after setting all properties.
function detrendbutton_CreateFcn(hObject, eventdata, handles)
% hObject    handle to detrendbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- Radio button para seleccionar el smartphone.
function smfradiobutton_Callback(hObject, eventdata, handles)
% hObject    handle to smfradiobutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hint: get(hObject,'Value') returns toggle state of smfradiobutton
if get(handles.smfradiobutton,'Value')==1
    set(handles.cntsftogglebutton,'Enable','on');
    if get(handles.cntsftogglebutton,'Value')==1
        set(handles.startsfbutton,'Enable','on');
    end
    set(handles.cntardtogglebutton,'Enable','off');
    set(handles.startardbutton,'Enable','off');
    set(handles.stopardbutton,'Enable','off');
    set(handles.saveardbutton,'Enable','off')
else
    set(handles.cntsftogglebutton,'Enable','off');
    set(handles.startsfbutton,'Enable','off');
    set(handles.stopsfbutton,'Enable','off');
    set(handles.savebutton,'Enable','off');

end

% --- Radiobutton para seleccionar el arduino.
function ardradiobutton_Callback(hObject, eventdata, handles)
% hObject    handle to ardradiobutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of ardradiobutton
if get(handles.ardradiobutton,'Value')==1
    set(handles.cntardtogglebutton,'Enable','on');
    if get(handles.cntardtogglebutton,'Value')==1
        set(handles.startardbutton,'Enable','on');
    end
    set(handles.cntsftogglebutton,'Enable','off');
    set(handles.startsfbutton,'Enable','off');
    set(handles.stopsfbutton,'Enable','off');
    set(handles.savebutton,'Enable','off');
else
    set(handles.cntardtogglebutton,'Enable','off');
    set(handles.startardbutton,'Enable','off');
    set(handles.stopardbutton,'Enable','off');
    set(handles.saveardbutton,'Enable','off');

end


% --- Ejecución de la conexión con el arduino.
function cntardtogglebutton_Callback(hObject, eventdata, handles)
% hObject    handle to cntardtogglebutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of cntardtogglebutton
data=guidata(hObject);
% Almacena en una variable global el objeto bluetooth
global a;
if get(handles.cntardtogglebutton,'Value')==1
```

```
    a=Bluetooth(get(handles.ardport,'String'),1)

    set(handles.startardbutton,'Enable','on');

else
end
guidata(hObject,data);

% --- Detiene el registro en tiempo real del arduino.
function stopardbutton_Callback(hObject, eventdata, handles)
% hObject    handle to stopardbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
% Importante!! Como hemos usado un bluetooth HC-06, hay que asegurarse de
% cerrar la conexxión, puesto que al ser esclavo sólo puede haber una
% conexión activa, con lo que si no se cierra apropiadamentes no se podra
% volver a conectar de manera normal, habrá que reiniciarlo.
fclose(a);
set(handles.startardbutton,'Enable','on');
set(handles.stopardbutton,'Enable','off');
set(handles.saveardbutton, 'Enable', 'on');
global KeepRunning;

KeepRunning=0;

% --- Guardo los registros del arduino.
function saveardbutton_Callback(hObject, eventdata, handles)
% hObject    handle to saveardbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.saveardbutton, 'Enable', 'off');
data=guidata(hObject);
% De nuevo, el nombre se corresponde con la fecha y hora de guardado del
% archivo
c=clock;
namefile=[num2str(c(1)),'-',num2str(c(2)),'-',...
    num2str(c(3)),'-',num2str(c(4)),'.',num2str(c(5)),'.',...
    num2str(round(c(6))),'.txt'];
data.filename=namefile;
[t1,z]=getpoints(data.hz);
[t2,y]=getpoints(data.hy);
[t,x]=getpoints(data.hx);
data.T=t';
data.X=x';
data.Y=y';
data.Z=z';
% Por como se reciven los datos, el primer registro suele estar
% incompleto, con lo que para que no haya error al indexar la matriz que
% introduciremos en el txt, ajusto los vectores.
L=[length(data.T) , length(data.X),length(data.Y),length(data.Z)];
if L(1)~=L(2) || L(1)~=L(3) || L(1)~=L(4)
    data.T=data.T(1:min(L));
    data.X=data.X(1:min(L));
    data.Y=data.Y(1:min(L));
```

```
    data.Z=data.Z(1:min(L));
end
%  Y con esto ya guardo:
A=[data.T'.*1000; data.X'; data.Y'; data.Z'];
fileID = fopen([data.filepath,'\',data.filename],'w');
fprintf(fileID,'\t %s \t %s \t %s \t %s \n','T','X','Y','Z');
fprintf(fileID,'\t %f \t %f \t %f \t %f \n',A);
fclose(fileID);

guidata(hObject,data);
LoadListBox(handles,data.filepath);


function ardport_Callback(hObject, eventdata, handles)
% hObject    handle to ardport (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ardport as text
%        str2double(get(hObject,'String')) returns contents of
%        ardport as a double


% --- Executes during object creation, after setting all properties.
function ardport_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ardport (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get↙
(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Entrada en tiempo real de los datos del arduino.
function startardbutton_Callback(hObject, eventdata, handles)
% hObject    handle to startardbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Inicializo los datos:
data=guidata(hObject);
global KeepRunning;
global a;
data=guidata(hObject);
data.T=0;
data.X=0;
data.Y=0;
data.Z=0;
initialTime=0;
%  Variables para la media móvil:
xprev=zeros(1,3);
```

```
yprev=zeros(1,3);
zprev=zeros(1,3);
% Habilito/deshabilito lo que corresponda al funcionamiento del programa:
set(handles.rsmbutton, 'Enable', 'off');
set(handles.resampbutton, 'Enable', 'off');
set(handles.fftbutton, 'Enable', 'off');
set(handles.detrendbutton, 'Enable', 'off');
set(handles.butterbutton, 'Enable', 'off');
set(handles.startardbutton, 'Enable', 'off');
set(handles.stopardbutton, 'Enable', 'off');
set(handles.saveardbutton, 'Enable', 'off');
% Abro el objeto bluetooth, para poder leerolo despues
fopen(a);
% Para cerciorarme de que se ha abierto:
disp('a connected');
% pause(2);
cla
KeepRunning=1;

%   Declaro las animated line donde se apilaran los puntos registrados
    hz = animatedline('Color','k', 'Parent', handles.axes1);
    hy = animatedline('Color','r', 'Parent', handles.axes1);
    hx = animatedline('Color','b', 'Parent', handles.axes1);
    set(handles.axes1,'XLim',[0 5],'YGrid','on');

    set(handles.startardbutton,'Enable','off');
    set(handles.stopardbutton,'Enable','on');
    set(handles.saveardbutton, 'Enable', 'off');

tic
%  Bucle de registro:
while (KeepRunning)
    % Reajuste del tamaño de la ventana
    if toc>5
        set(handles.axes1,'XLim',[toc-5,toc]);
    end
    % Leo el objeto bluetooth: Me envia los datos tal que:
    % "#XXXX YYYY ZZZZ", que corresponden con la salida digital entre 0 y
    % 1023 de cada eje.
    p=fgets(a);
    % No leo el primer elemento de la cadena de caracteres, ya que es #,
    % que sirve para detectar el inicio del registro.
    A= sscanf(p(2:end),'%f %f %f')
    try

%         Versión que calcula aquí la media movil:
%           Me quedo con esta porque va más rápido.

%       Roto los valores almacenados en un vector de tres posiciones para
%       calcular la media movil
        xprev=circshift(xprev,[0,1]);
        yprev=circshift(yprev,[0,1]);
        zprev=circshift(zprev,[0,1]);
%       Versión precaria de la rotación, mediante bucle:
%         for n=1:2
```

```
%             xprev(n+1)=xprev(n);
%             yprev(n+1)=yprev(n);
%             zprev(n+1)=zprev(n);
%         end
        % Almaceno los nuevos:
        xprev(1)=A(1);
        yprev(1)=A(2);
        zprev(1)=A(3);
%         Le doi medio segundo para que no empiece con los valores de
%         tiempo a almacenar algunos valores para la media movil
        if toc>0.5
%             Inicializo el tiempo de registro
            if initialTime==0
                initialTime=toc;
            end
%               Comienzo el registro. Para ello paso el valor que viene del
%               arduino a m/s^2, como se explica en la memoria
            addpoints(hx,toc-initialTime,(mean(xprev)*..
                (3.3/1023)-1.65)*(9.806/0.8))
            addpoints(hy,toc-initialTime,(mean(yprev)*..
                (3.3/1023)-1.65)*(9.806/0.8))
            addpoints(hz,toc-initialTime,(mean(zprev)*..
                (3.3/1023)-1.65)*(9.806/0.8))
        end

%  Versión con la media movil ya calculada por el arduino:
%         addpoints(hx,toc,(A(1)*(3.3/1023)-1.65)*(10/0.8))
%         addpoints(hy,toc,(A(2)*(3.3/1023)-1.65)*(10/0.8))
%         addpoints(hz,toc,(A(3)*(3.3/1023)-1.65)*(10/0.8))
%

        drawnow limitrate

    catch
    end
    data.hz=hz;
    data.hy=hy;
    data.hx=hx;

end

guidata(hObject,data);



function tminCutEdit_Callback(hObject, eventdata, handles)
% hObject    handle to tminCutEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.tminCutEdit,'String');
if isnan(str2double(str)) || str2double(str)<0tminCutEdit
    set(handles.fcbutteredit,'string',' ');
    warndlg('Input must be numerical ','Error');
else
```

```
    data.fcbutter=str2double(str);
    set(handles.fcbuttertext,'String', ['Cutoff frec.: '...
    ,num2str(data.fcbutter),' hz']);
end
% set(handles.orderbuttertext, 'String', ['Order: '...
%     ,num2str(str)]);
guidata(hObject,data);

% Hints: get(hObject,'String') returns contents of tminCutEdit as text
%        str2double(get(hObject,'String')) returns contents of tminCutEdit
%        as a double

% --- Executes during object creation, after setting all properties.
function tminCutEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tminCutEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function tmaxCutEdit_Callback(hObject, eventdata, handles)
% hObject    handle to tmaxCutEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tmaxCutEdit as text
%        str2double(get(hObject,'String')) returns contents of tmaxCutEdit
%        as a double


% --- Executes during object creation, after setting all properties.
function tmaxCutEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tmaxCutEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% Función para recortar el registro.
function cutButton_Callback(hObject, eventdata, handles)
% hObject    handle to cutButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
filename=strrep(data.filename,'.txt','.CUT.txt');
data.filename=filename;
MinCut=1;
MaxCut=1;

% Si no introduczo tmin, inicia en cero:
if data.tminCut==0
    MinCut=1;
else
% Si sí, buesco el valor más próximo sin pasarme:
    while data.tminCut>data.T(MinCut)
        MinCut=MinCut+1;
    end
    MinCut=MinCut-1;
end
%  Análogo a tmin pero con el máximo:
if data.tmaxCut==0 || data.tmaxCut<data.tminCut
    MaxCut=length(data.T);
else
    while data.tmaxCut>data.T(MaxCut) &&...
            MaxCut<length(data.T)-1
        MaxCut=MaxCut+1;
    end
    MaxCut=MaxCut-1;
end
% El nuevo tiempo:
newTime=data.T(MinCut:MaxCut)-data.T(MinCut);
newX=data.X(MinCut:MaxCut);
newY=data.Y(MinCut:MaxCut);
newZ=data.Z(MinCut:MaxCut);
data.T=newTime;
data.X=newX;
data.Y=newY;
data.Z=newZ;
guidata(hObject,data);
plotFnc(hObject,handles);

guidata(hObject,data);



function tminEdit_Callback(hObject, eventdata, handles)
% hObject    handle to tminEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.tminEdit,'String');
if isnan(str2double(str)) || str2double(str)~=floor(str2double(str))..
        || str2double(str)<0
    set(handles.tminEdit,'string','0');
    warndlg('Input must be numerical and integer','Error');
else
    data.tminCut=str2double(str);
```

```
    set(handles.tminText, 'String', ['t min:  '...
    ,num2str(data.tminCut),' ms']);
end

guidata(hObject,data);

% Hints: get(hObject,'String') returns contents of tminEdit as text
%        str2double(get(hObject,'String')) returns contents of tminEdit
%        as a double


% --- Executes during object creation, after setting all properties.
function tminEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tminEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function tmaxEdit_Callback(hObject, eventdata, handles)
% hObject    handle to tmaxEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
str=get(handles.tmaxEdit,'String');
if isnan(str2double(str)) || str2double(str)~=floor(str2double(str))..
        || str2double(str)<0
    set(handles.tmaxEdit,'string','0');
    warndlg('Input must be numerical and integer','Error');
else
    data.tmaxCut=str2double(str);
    set(handles.tmaxText, 'String', ['t max:  '...
    ,num2str(data.tmaxCut),' ms']);
end

guidata(hObject,data);
% Hints: get(hObject,'String') returns contents of tmaxEdit as text
%        str2double(get(hObject,'String')) returns contents of tmaxEdit
%        as a double


% --- Executes during object creation, after setting all properties.
function tmaxEdit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tmaxEdit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in xDumpingRB.
function xDumpingRB_Callback(hObject, eventdata, handles)
% hObject    handle to xDumpingRB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of xDumpingRB


% --- Executes on button press in yDumpingRB.
function yDumpingRB_Callback(hObject, eventdata, handles)
% hObject    handle to yDumpingRB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of yDumpingRB


% --- Executes on button press in zDumpingRB.
function zDumpingRB_Callback(hObject, eventdata, handles)
% hObject    handle to zDumpingRB (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of zDumpingRB


% Ejecuta la función de Dumping:
function dumpingButton_Callback(hObject, eventdata, handles)
% hObject    handle to dumpingButton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
data=guidata(hObject);
t=data.T.*0.001;
% Lo único que hace antes de lanzar la funcion es seleccionar los datos a
% mandar, en funciónd e los radiobutton seleccionados.
if get(handles.xDumpingRB, 'Value')==1 && ...
        get(handles.yDumpingRB, 'Value')==1 && ...
        get(handles.zDumpingRB, 'Value')==1
    a=[data.X data.Y data.Z];
elseif get(handles.xDumpingRB, 'Value')==1 && ...
        get(handles.yDumpingRB, 'Value')==1
    a=[data.X data.Y];
elseif get(handles.yDumpingRB, 'Value')==1 && ...
        get(handles.zDumpingRB, 'Value')==1
    a=[data.Y data.Z];
elseif get(handles.xDumpingRB, 'Value')==1 && ...
        get(handles.zDumpingRB, 'Value')==1
```

```
    a=[data.X data.Z];
elseif get(handles.xDumpingRB, 'Value')==1
    a=[data.X];
elseif get(handles.yDumpingRB, 'Value')==1
    a=[data.Y];
else
    a=[data.Z];
end
getDamping3(t,a);
```