

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE CARTAGENA



Trabajo Fin de Grado

Desarrollo e Implementación en MATLAB de una herramienta para la realización de medidas del canal de radiocomunicaciones mediante el analizador de espectro portátil MS2090A



AUTOR: David Párraga Riquelme
DIRECTORES: Leandro Juan Llácer
José María Molina García-Pardo

MARZO / 2021

Autor:	David Párraga Riquelme
E-mail del Autor:	davidparraga@hotmail.com
Directores:	Leandro Juan Llácer José María Molina García-Pardo
E-mail de los Directores:	leandro.juan@upct.es josemaria.molina@upct.es
Título del TFG:	Desarrollo e Implementación en MATLAB de una herramienta para la realización de medidas del canal de radiocomunicaciones mediante el analizador de espectro portátil MS2090A
Resumen:	<p>La planificación de los sistemas de radiocomunicaciones se realiza con herramientas informáticas que calculan la cobertura radioeléctrica a partir de los parámetros de transmisión (potencia transmitida, ganancia de antena, pérdidas en cables y conectores), los parámetros en recepción (ganancia de antena, pérdidas y sensibilidad) y las pérdidas de propagación. Para estimar las pérdidas de propagación, dichas herramientas incorporan modelos electromagnéticos teóricos, empíricos o semiempíricos que pueden aplicarse en una banda de frecuencia y en un entorno (rural, urbano, suburbano o interiores). Estos modelos necesitan ser validados mediante la realización de campañas de medidas en los entornos y en las bandas de frecuencia correspondientes. En este sentido, se hace necesario disponer de sistemas de medidas que faciliten la realización de estas campañas de medidas.</p> <p>Existen numerosos sistemas de medida: basados en analizadores de redes, de espectro, en generadores de señal, oscilloscopios, etc. La gran mayoría de ellos incorporan un PC con programas informáticos que ayudan a controlar el equipamiento del sistema de medidas.</p> <p>El objetivo principal del proyecto es desarrollar una herramienta informática cuyas funcionalidades ayuden a la realización de campañas de medidas en un sistema basado en el analizador de espectros portátil MS2090A de Anritsu.</p>
Titulación:	Grado en Ingeniería en Telemática
Departamento:	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación:	Marzo 2021

“Si caminas solo, irás más rápido.

Si caminas acompañado, llegarás más lejos”

Proverbio chino

Agradecimientos

Antes de empezar la redacción de esta memoria, me gustaría agradecer a todos quienes han hecho posible este proyecto, de forma directa o indirecta. También a los que me han acompañado y ayudado durante mis años de universidad, y por supuesto, a los que han estado a mi lado desde el inicio mi vida hasta este momento, en el que me veo convertido en la persona que libremente elegí ser y en un profesional completo.

A Leandro, como amigo y director de este proyecto. Desarrolló la idea y me propuso llevarla a cabo para después guiarme adecuadamente durante todo el proceso.

A José María, que como subdirector ha acelerado el proyecto con su conocimiento en la materia.

A Luis, Jaime y Andrés como primeros compañeros de piso durante mi estancia en Cartagena; y a Ignacio y José como segundos. Con nuestras conversaciones ambiciosas y soñadoras logramos graduarnos.

A mis amigos de siempre, Javier, José María, Juan, Juan Carlos, Pepe, Gonzalo, Álvaro, Pablo, Antonio y Otto. Juntos hemos sabido exprimir nuestra juventud más temprana.

A Fini, Roberto, Irene y Roberto, por su apoyo y motivación en los últimos esfuerzos de mi grado.

A mis tíos Javier, Maite, Inma y Enrique; a quienes les debo útiles consejos y muchas invitaciones a buenas comidas y mejores ratos.

A mis abuelos Enrique, María, Amparo y Jesús, por su enorme ternura y por aportarme sin saberlo un filtro retrospectivo con el que entender y prever mis vivencias.

A mi padre, Manuel, por demostrarme como hacer uso de la inteligencia y la tenacidad para alcanzar metas, como aquella de dar una buena vida a sus hijos.

A mis hermanos, Rubén y Manuel, por todas nuestras risas compartidas y buenos momentos en la huerta y, por formar (junto a nuestros padres), una familia compacta de buenos valores que ha sabido y sabe empujar fuerte cada proyecto del camino, bajo cualquier circunstancia.

A Fina, por sus meriendas, por aconsejarme desde su sobresaliente pensamiento crítico y por criarme (algo que sigue haciendo) con mucho afecto.

A mi madre, Concepción, que, con su fuerte ética de trabajo, su amor incondicional y todos sus logros me ha inculcado que si puedes pensar lo puedes hacerlo.

Finalmente, pero no menos importante, a mi novia Ana, por decidir formar parte de mi vida, por su oído incansable, por valorarme y hacerme de valer y, por encima de todo eso, por darme la oportunidad de verme a través de sus Ojos.

Contenido

Capítulo 1 Introducción y objetivos	1
1.1. Introducción	1
1.2. Objetivos	2
1.3. Fases del proyecto.....	2
1.4. Estructura	2
Capítulo 2. El canal de radiocomunicaciones y sistemas de medidas.....	4
2.1. Introducción	4
2.2. Radiación electromagnética y espectro electromagnético.....	4
2.3. Radiocomunicación	6
2.4. El espectro radioeléctrico.....	7
2.5. Caracterización de propagación del canal radio	9
2.6. Sistemas de medidas	12
2.6.1. Introducción	12
2.6.2. Sistemas de medida en banda estrecha.....	12
2.6.3. Sistema de medidas en banda ancha.....	14
Capítulo 3. Diseño e integración en un sistema de medidas	16
3.1. Introducción	16
3.2. Implementación de los componentes.....	16
Capítulo 4. El analizador de espectro Field Master Pro MS2090A.....	18
4.1. Introducción	18
4.2. Descripción y aspecto físico	18
4.3. Modo analizador de espectro.....	20
4.3.1. Configuración de traza (Trace Menu).....	21
4.3.2. Tipos de mediciones.....	23
4.3.3. GPS para referenciar medidas.....	26
4.4. Configuración para la programación y operatividad remota.....	26

4.4.1. Conexión y configuración de la interfaz de red.....	26
4.4.2. Estándar SCPI.....	29
4.5. Parámetros específicos a tener en cuenta: CF, SPAN, RBW, VBW, SWEEP POINTS	31
4.6. Configuración seleccionada para la medición.....	32
Capítulo 5. Desarrollo de la aplicación en MATLAB©	33
5.1. Introducción	33
5.2. Funcionalidades de la aplicación.....	33
5.3. Verificación de la comunicación remota.....	34
5.4. <i>Guide</i> de MATLAB.....	35
5.5. Código de la aplicación.....	39
5.5.1. Abrir conexión con analizador y configurar sus parámetros más relevantes	39
5.5.2. Toma de medidas manual y botón para guardar campaña	42
5.5.3. Solución para la excepción de falta de cobertura GPS.....	44
5.5.4. Toma de medidas automáticas para <i>Dirve testing</i>	45
5.5.5. Gráfica de la potencia en tiempo real.....	46
5.5.6. Generación de un .exe	47
5.6. Formato en el que se guarda la campaña	47
5.7. Manual de usuario de la aplicación.....	48
Capítulo 6. Aplicación del sistema y resultados	51
6.1. Introducción	51
6.2. Montaje de los equipos.....	51
6.3. Campañas de medidas	53
6.3.1. Antiguo cuartel de Antiguones.....	53
6.3.2. Cultivo de cítricos en Sierra Espuña	53
6.4. Resultados	55
6.5. Scripts utilizados para el procesado de las medidas	59
6.5.1. Script para convertir de coordenadas geográficas decimales a UTM	60

6.5.2. Script para representar gráficas y calcular parámetros	62
6.6. Metodología para la representación de las rutas	64
Capítulo 7. Conclusiones	65
Bibliografía.....	66
Anexo. Código de MATLAB.....	68

Índice de figuras

Figura 1. Propagación de una onda [3]	4
Figura 2. Espectro electromagnético [3]	5
Figura 3. Propagación de las ondas de radio [7]	7
Figura 4. Refracción y reflexión de una onda electromagnética al cambiar de medio [11]	10
Figura 5. Propagación multicamino [12]	11
Figura 6. Principales mecanismos de transmisión multicamino [12].....	11
Figura 7. Esquema general de un sistema de medidas [10].....	12
Figura 8. Medidor de campo Televés [14].....	13
Figura 9. Dispositivo TEMS pocket [16].....	13
Figura 10. Analizador de espectros Rodhe&Schwarz[17]	14
Figura 11. Rélicas recibidas cuando se transmite un pulso de corta duración [10].....	14
Figura 12. Esquemático de sistema de medidas utilizado	17
Figura 13. Panel frontal del instrumento Field Master Pro [18]	18
Figura 14. Panel de conectores superior [18]	19
Figura 15. Panel de conectores lateral [18].....	19
Figura 16. Vista previa menú de selección modos de funcionamiento [18].....	20
Figura 17. Vista predeterminada analizador de espectro [18].....	20
Figura 18. Vista previa menú de traza [18]	21
Figura 19. Vista de pantalla con 6 trazas con diferentes configuraciones [18].....	22
Figura 20. Vista analizador operando en "Channel Power" [18].....	24
Figura 21. Vista analizador operando en “Occupied Bandwith” [18]	24
Figura 22. Vista analizador operando en "Adjacent Channel Power" [18]	25
Figura 23. Vista analizador operando en “Spectrum emision Mask” [18]	25
Figura 24. Funcionamiento del sistema de posicionamiento global [19]	26
Figura 25. Menú configuración de red Anritsu [18]	29
Figura 26. Verificación comunicación remota Entorno MATLAB PC-Analizador Anritsu.....	35

Figura 27. Ejemplo de GUI muy sencilla en Matlab Guide	36
Figura 28. Primera versión de la aplicación.....	40
Figura 29. Mensaje de error “Fill in the blanks”.....	41
Figura 30. Segunda versión de la aplicación	42
Figura 31. Cuadro de diálogo que se muestra al fin de la campaña	44
Figura 32. Tercera versión de la aplicación	45
Figura 33. Última versión de la aplicación.....	46
Figura 34. Archivos creados por compilador MATLAB	47
Figura 35. Aspecto del fichero “.mat” que devuelve la aplicación.....	48
Figura 36. Montaje estación transmisora	51
Figura 37. Montaje estación transmisora	52
Figura 38. Montaje estación receptora: Vista vehículo con antena receptora.....	52
Figura 39. Montaje estación receptora: Interior de vehículo con ordenador y ANRITSU	53
Figura 40. Corte vertical con árboles equiespaciados.....	54
Figura 41. Ruta campaña de medidas ETSIT UPCT	55
Figura 42. Plantación de limoneros, situación del transmisor y recorridos con las medidas realizadas sin limón	56
Figura 43. Plantación de limoneros, situación del transmisor y recorridos con las medidas realizadas con limón.....	57
Figura 44. Distancia (m) – Potencia (dBm) en Antiguones.....	58
Figura 45. Gráfica Distancia (m) – Potencia recibida (dBm) sin limón y recta de regresión	58
Figura 46. Gráfica Distancia (m) – Potencia recibida (dBm) con limón y recta de regresión.....	59
Figura 47. Mapa global de zonas UTM [23].....	60

Índice de tablas

Tabla 1. Uso de bandas y frecuencias del espectro radioeléctrico [9].....	8
Tabla 2. Ejemplo de uso de las bandas del espectro radioeléctrico [9]	9
Tabla 3. Componentes que ofrece MATLAB Guide para disponer en el área de diseño (.fig) [21]	
.....	36
Tabla 4. Pendiente y desviación típica rectas de regresión de las campañas en cultivo de cítricos	
.....	59
Tabla 5. Ejemplo simplificado medidas en hoja de cálculo previa importación a Google Earth	64

Capítulo 1 Introducción y objetivos

1.1. Introducción

Las tecnologías inalámbricas han experimentado un gran desarrollo en los últimos años para las comunicaciones a corta distancia, como la tecnología Bluetooth; a media distancia, como ZigBee; y a larga distancia, como WiFi, GSM / GPRS (2G), UMTS (3G) o LTE (4G) [1].

Para realizar una planificación eficiente de los sistemas que proveen de estas tecnologías de telecomunicación es preciso caracterizar el entorno sobre el que van a instalarse, entendiendo la forma en que se propagará la señal transmitida a medida que se difunde por el terreno. La técnica más efectiva para extraer información del entorno de propagación es la realización de campañas de medidas. Este proyecto tiene por objetivo principal el desarrollo y la implementación en MATLAB de una herramienta de *drive-testing* para la realización de medidas del canal de radiocomunicaciones mediante el analizador de espectro portátil Field Master Pro MS2090A de Anritsu. Dicha herramienta tomará medidas referenciadas geográficamente del nivel de potencia del radiocanal. Los resultados de la campaña serán procesados para determinar cómo varía el valor de potencia recibida con respecto a la distancia entre transmisor y receptor en el momento en que se va tomando cada medida. Una vez acometido el objetivo principal se valoró en qué tipo de entorno sería interesante su aplicación. En agricultura, las tecnologías inalámbricas a media y larga distancia se han combinado para enviar remotamente parámetros capturados sobre el terreno en áreas localizadas. La agricultura 4.0 [2] va a suponer un aumento considerable del número de sensores, así como la aparición de nuevas tecnologías inalámbricas que van a facilitar, por ejemplo, el guiado automático de tractores, el seguimiento del estado del fruto a lo largo del tiempo antes de su cosecha, la automatización del abonado o la recolección de frutas, etc. Por estos motivos se concluyó que los datos que la herramienta podría arrojar para un entorno agrícola podrían ser de interés para el Departamento de Tecnologías de la Información y las Comunicaciones de la Universidad, por lo que se realizaron dos campañas de medidas en un entorno de cultivo de cítricos (con y sin limón en el árbol), no sin antes hacer una prueba de funcionamiento en las inmediaciones de la Escuela Técnica Superior de Ingeniería de Telecomunicaciones de la Universidad Politécnica de Cartagena.

La frecuencia a la que se midió la campaña en la plantación de cítricos fue de 3.5 GHz, una de las frecuencias del 5G, mientras que en la campaña de prueba en la ETSIT de la UPCT se midió a una frecuencia central de 1 GHz.

1.2. Objetivos

El objetivo principal del proyecto es desarrollar una herramienta informática cuyas funcionalidades ayuden a la realización de campañas de medidas en un sistema basado en el analizador de espectros portátil MS2090A de Anritsu.

Después se utilizará esta herramienta para caracterizar el canal de radiofrecuencia en un entorno de cultivo de cítricos, lo que arrojará información valiosa sobre la forma en que se propaga una señal radio en este tipo de entornos.

1.3. Fases del proyecto

Las fases seguidas en este trabajo han sido:

1. Estado del arte de sistemas de medidas basados en analizadores de espectro (septiembre 2020).
2. Estudio del analizador de espectros portátil MS2090A (octubre 2020).
3. Desarrollo de las funcionalidades que va a incorporar el programa (octubre 2020).
4. Implementación en MATLAB del programa (octubre 2020 – enero 2020).
5. Pruebas de funcionamiento mediante la realización de campañas de medidas (febrero 2020 – febrero 2021).
6. Redacción de la memoria (febrero 2021).

1.4. Estructura

En el capítulo 2 se describe qué es la radiación electromagnética y cómo es desglosada y clasificada para conformar el espectro radioeléctrico. También se expondrá cómo la radiación electromagnética es utilizada y regulada en el campo de las telecomunicaciones, y del uso que se hace de ella en la disciplina de la radiocomunicación. Después, se muestra qué es el radiocanal y las técnicas y sistemas que permiten modelarlo y caracterizarlo.

En el capítulo 3 se muestra el esquema del sistema de medidas que se ha utilizado en las campañas junto a la descripción de las partes que lo constituyen.

En el capítulo 4 se describe, desde lo más general a lo más particular, el analizador de espectros utilizado para la campaña de medidas. Su aspecto físico, sus diferentes modos de funcionamiento, sus parámetros de configuración y las características que ofrece.

En el capítulo 5 se presenta el diseño de la herramienta de medidas, y como se va desarrollando paso a paso.

El capítulo 6 describe las campañas que se realizaron con la herramienta de medidas diseñada, los resultados obtenidos de estas campañas y los scripts que han sido diseñados y usados para el procesado de las medidas.

Al final del documento se añade una bibliografía con todas las referencias de las que se ha valido esta redacción y un anexo con el código en MATLAB de la herramienta desarrollada.

Capítulo 2. El canal de radiocomunicaciones y sistemas de medidas

2.1. Introducción

En este capítulo se describirá qué es la radiación electromagnética y cómo es desglosada y clasificada para conformar el espectro radioeléctrico. También se expondrá cómo la radiación electromagnética es utilizada y regulada en el campo de las telecomunicaciones, y del uso que se hace de ella en la disciplina de la radiocomunicación. Después, se muestra qué es el radiocanal y las técnicas y sistemas que permiten modelarlo y caracterizarlo.

2.2. Radiación electromagnética y espectro electromagnético

En nuestras vidas hacemos uso de muchas tecnologías basadas en radiación electromagnética, una combinación de campos eléctricos y magnéticos oscilantes que se propagan a través del espacio transportando energía, como lo hace la luz [3]. La radiación electromagnética se enmarca en un rango ordenado según la energía que transporta. Este rango se conoce como el espectro electromagnético (Fig. 2). Para caracterizar una onda atendemos a su longitud, o lo que es lo mismo, a la longitud de onda (λ) que se define como la distancia entre dos crestas consecutivas.

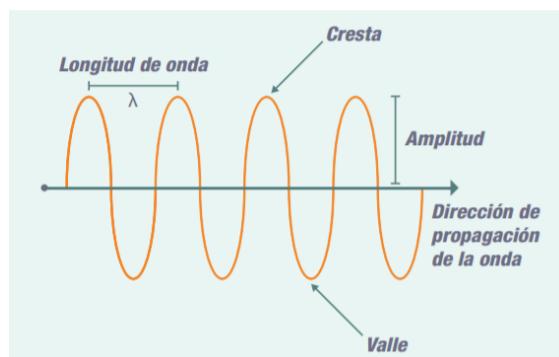


Figura 1. Propagación de una onda [3]

El periodo (T) es el tiempo que le toma a la onda completar un ciclo. La frecuencia (f) es la cantidad de ciclos que completa una onda en un segundo, se mide en Hertz (Hz) y se calcula como la inversa del periodo: $f = 1/T$, por lo que a mayor frecuencia menor es su longitud de onda. Por el contrario, a mayor frecuencia de onda, mayor energía transporta la onda.

Extrapolando, una onda que vibra rápido tiene menor longitud de onda y más energía que otra que vibra más lentamente [3].

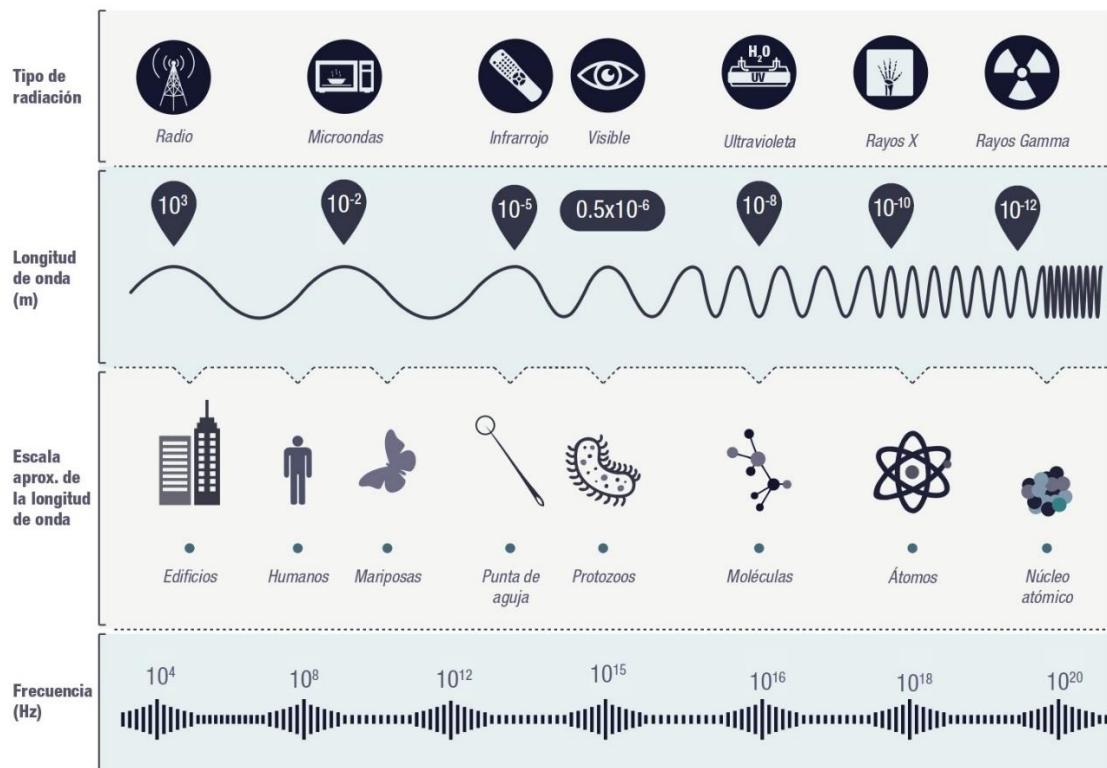


Figura 2. Espectro electromagnético [3]

Atendiendo a la figura 2 vemos que las ondas de radio se corresponden con la radiación de mayor longitud de onda y por tanto menor energía.

Las microondas, que se usan en telecomunicación celular, tienen longitudes de onda del orden de un milímetro hasta un metro. A continuación de las microondas ubicamos la radiación infrarroja que utilizamos para observar el cosmos o para el control remoto de aparatos electrónicos. Sus longitudes de onda van de los 700 nm. hasta 1 mm. Inmediatamente después tenemos el intervalo de la luz visible. Aunque el espectro de la luz visible es un continuo de infinitos colores, a modo de ejemplo podemos ver las longitudes de onda aproximada de los más tradicionales: rojo (700 nm), naranja (650 nm), amarillo (600 nm), verde (570 nm), azul (450 nm) y violeta (400 nm).

La luz blanca- que emite nuestro Sol por ser una estrella blanca- es la combinación de todos los colores superpuestos y si se hace pasar por un prisma se descompone en todo el espectro visible. Después, con más energía, tenemos los rayos ultravioleta UV (400 nm a 10 nm). Con más energía todavía tenemos los rayos-x (10^{-10} m). Finalmente, en el extremo superior podemos ver la radiación con mayor energía del universo y menor longitud de onda, los rayos gamma (con longitud de onda del orden de 10^{-14} m).

Independientemente de cuál sea su lugar en el espectro, toda radiación electromagnética viaja siempre a la velocidad de la luz, que en el vacío es equivalente a 300.000 km/s [3].

2.3. Radiocomunicación

Las ondas de radio son un tipo de radiación electromagnética [4] de menor longitud de onda, y por tanto con mayor energía y mayor frecuencia que la luz infrarroja. Aunque sean creadas de forma natural por fenómenos como objetos astronómicos o relámpagos, también pueden ser generadas de manera artificial por transmisores radio para diversas aplicaciones, como radio móvil y fija, radar, satélites de comunicaciones, radiodifusión, redes telemáticas [5], etc.

La forma en que las ondas de radio interactúan con un obstáculo variará dependiendo del tamaño del obstáculo y de la frecuencia a la que se radie. Las que cuentan con una frecuencia baja pueden atravesar obstáculos tan voluminosos como montañas, sufriendo una refracción que le facilite propagarse siguiendo el contorno de la Tierra. A las ondas que de esta forma son propagadas más allá de la línea de visión se les denomina ondas de superficie. Las que tienen una frecuencia más alta, pueden también refractarse en la ionosfera y alcanzar puntos que de otra forma no podría; puesto que se encuentran más allá del horizonte de visión (ondas ionosféricas). Para terminar, las ondas de radio con longitudes de onda más estrecha apenas sufren difracción [6], pudiendo ser usadas únicamente para comunicarse con estaciones que se encuentren en la línea de visión (LOS). Esto se conoce como propagación en la línea de visión, así que sus distancias de propagación están limitadas al horizonte visual.

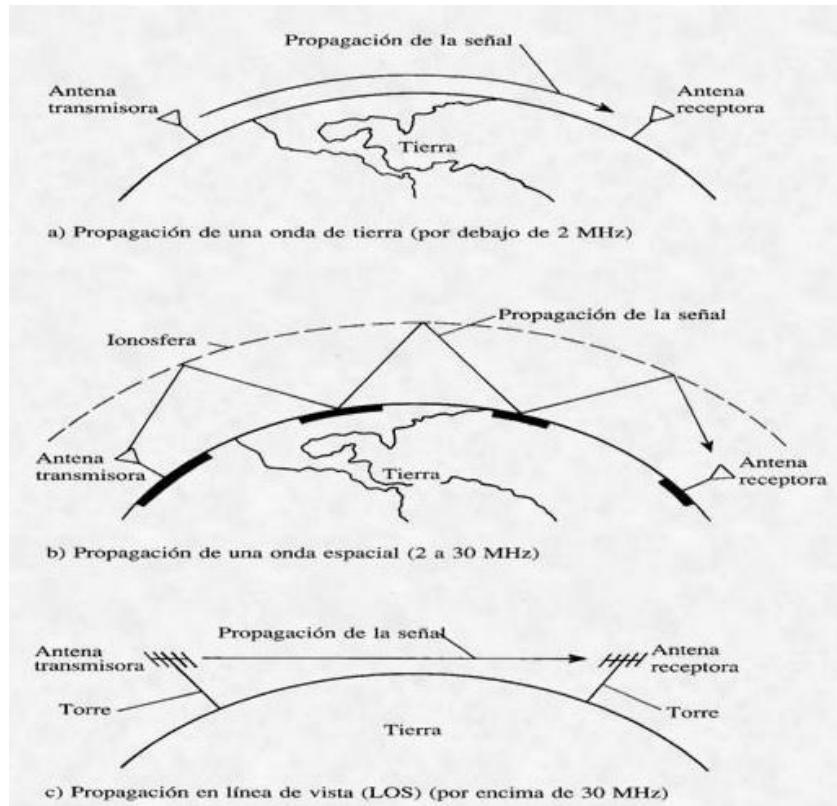


Figura 3. Propagación de las ondas de radio [7]

La radiocomunicación, teniendo en cuenta lo anterior, es una forma de telecomunicación que se realiza a través de ondas de radio, es decir, a través de la emisión de ondas cuya frecuencia cae dentro del rango que delimita el espectro radioeléctrico, lo que no quita que éstas tengan propiedades diferentes dependiendo de su frecuencia exacta [8]. Podemos encontrar bandas de baja frecuencia, media frecuencia, alta frecuencia, muy alta frecuencia, ultra alta frecuencia, etc. Además de la propia radio, también hacen uso del espectro radioeléctrico los transmisores de telefonía móvil, televisión, radar, etc.

2.4. El espectro radioeléctrico

Formalmente y según la unión internacional de telecomunicaciones o ITU por sus siglas en inglés, el rango de frecuencias que comprende el espectro radioeléctrico es de 0 Hz a 3000 GHz. El Reglamento de Radiocomunicaciones de la UI establece las llamadas bandas de frecuencia [9], y divide el espectro radioeléctrico en nueve bandas de frecuencia, las que en la siguiente tabla aparecen numeradas de la 4 a las 12.

Núm. de la banda	Nombre	Acrónimo	Rango de frecuencias	Subdivisión métrica	Acrónimo
	Tremendously low frequency	TLF	0 a 3 Hz		
	Extremely low frequency	ELF	3 a 30 Hz		
	Super low frequency	SLF	30 a 300 Hz		
	Ultra low frequency	ULF	300 Hz a 3 KHz		
4	Very low frequency	VLF	3 a 30 kHz	Ondas miriamétricas	B.Man
5	Low frequency	LF	30 a 300 kHz	Ondas kilométricas	B.km
6	Medium frequency	MF	300 a 3000 kHz	Ondas hectométricas	B.hm
7	High frequency	HF	3 a 30 MHz	Ondas decamétricas	B.dam
8	Very high frequency	VHF	30 a 300 MHz	Ondas métricas	B.m
9	Very high frequency	UHF	300 a 3000 MHz	Ondas decimétricas	B.dm
10	Super high frequency	SHF	3 a 30 GHz	Ondas centimétricas	B.cm
11	Extremely high frequency	EHF	30 a 300 GHz	Ondas milimétricas	B.mm
12	Terahertz or tremendously high frequency	THF	300 a 3000 GHz	Ondas decimilimétricas	B.dmm

Tabla 1. Uso de bandas y frecuencias del espectro radioeléctrico [9].

A continuación, se muestra una tabla más simple con los usos típicos que se hacen de esas frecuencias.

Núm. de la banda	Nombre	Rango de frecuencias	Ejemplos de uso
	TLF	0 a 3 Hz	Ruido natural o provocado por el ser humano
	ELF	3 a 30 Hz	Comunicación con submarinos
	SLF	30 a 300 Hz	Radar. Enlace de radio
	ULF	300 Hz a 3 KHz	Enlaces de Radio. Ayuda a navegación aérea. Comunicaciones en minas a través de la tierra
4	VLF	3 a 30 kHz	Radioayuda, señales de tiempo, comunicación submarina, pulsómetros inalámbricos, geofísica
5	LF	30 a 300 kHz	Radiodifusión en AM (onda larga, en Europa y partes de Asia), RFID ¹ , radioafición
6	MF	300 a 3000 kHz	Radiodifusión en AM (onda media), radioafición, balizamiento de aludes
7	HF	3 a 30 MHz	RFID, radar, comunicaciones ALE ² , comunicación cuasi-vertical (NVIS) ³ , telefonía móvil y marina.
8	VHF	30 a 300 MHz	FM, televisión, comunicaciones con aviones a la vista entre tierra-avión y avión-avión, telefonía móvil marítima y terrestre
9	UHF	300 a 3000 MHz	Comunicaciones por microondas, radioastronomía, redes inalámbricas, bluetooth, GPS
10	SHF	3 a 30 GHz	Comunicaciones por satélite, televisión por satélite, DBS ⁴
11	EHF	30 a 300 GHz	Radioastronomía, transmisión por microondas de alta frecuencia, teledetección
12	THF	300 a 3000 GHz	Radiografía, espectroscopia, comunicaciones/computación mediante terahercios

Tabla 2. Ejemplo de uso de las bandas del espectro radioeléctrico [9]

2.5. Caracterización de propagación del canal radio

El canal radio está constituido por el espacio y todos los elementos que se encuentran entre el transmisor y el receptor. Los canales cableados son predecibles y constantes [10], pero el canal móvil es totalmente aleatorio, sufriendo variaciones en los dominios frecuencial y temporal. Con un canal tan inestable es oportuno caracterizarlo lo más fielmente posible, con el objeto de aprovechar al máximo los recursos de una instalación de radiocomunicación y evitar incapacidades en el diseño de la misma.

¹ RFID: sistema de identificación de objetos por radiofrecuencia.

² ALE: estándar mundial para iniciar y mantener conversaciones usando radio de alta frecuencia.

³ NVIS: comunicación a larga distancia.

⁴ DBS: satélite de comunicación directa

Una señal puede encontrar en su trayectoria desde el emisor hasta el receptor una gran variedad de obstáculos que producirán diferentes efectos sobre la señal de radio. Los tres fenómenos más comunes son la reflexión, la refracción y el *scattering*. Se producen cuando una onda electromagnética pasa de un medio en el cual la luz se propaga con una velocidad a otro en el que la luz se propaga con otra velocidad [11]. Podemos cuantificar este efecto de un medio con el índice de refracción (n) que nos informa de la velocidad de propagación de la luz en ese medio con respecto al vacío. Estos principios se basan en las leyes de la óptica, porque recordemos que la luz no deja de ser un tipo de radiación electromagnética

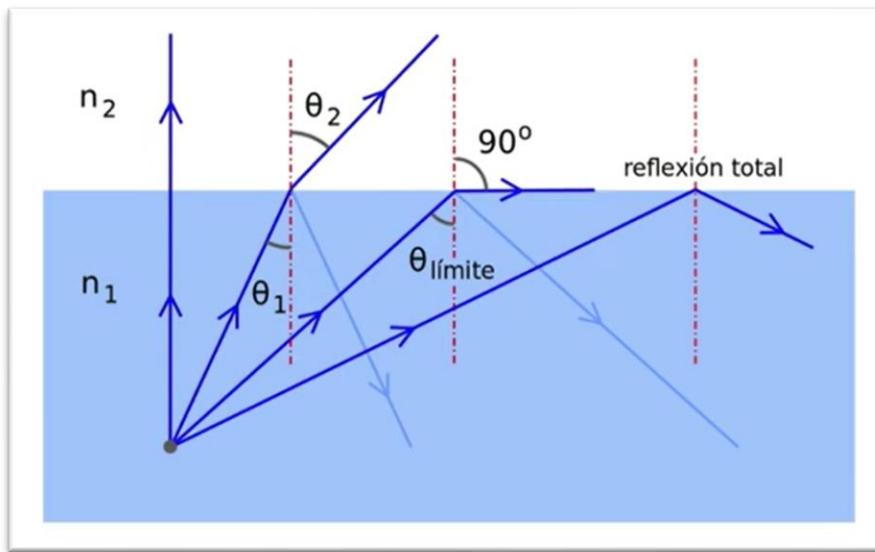


Figura 4. Refracción y reflexión de una onda electromagnética al cambiar de medio [11]

En la figura 4 podemos observar como la onda al cambiar de medio, puede verse inalterada porque tengan el mismo coeficiente de reflexión que el medio del que venía, también puede que se refracte, cambiando su trayectoria y reflejándose parte de ella de nuevo al primer medio o; a la derecha del todo se nota como puede reflejarse por completo al medio del que venía.

Además de los fenómenos de la refracción y la reflexión, también tenemos el fenómeno de la difusión o *scattering*, que tiene lugar cuando la radiación electromagnética choca con un objeto cuyas dimensiones son similares a su longitud de onda [12]. El efecto subyacente de este fenómeno es que la potencia de señal recibida en el receptor será mayor que la esperada.

La refracción, la difracción, la reflexión y la difusión terminan con múltiples réplicas o contribuciones de la señal original transmitida, de las cuales cada una han podido sufrir estos efectos. Las distintas réplicas han podido recorrer trayectorias distintas de diferente longitud hasta llegar al receptor, alcanzándole con diferentes retardos de propagación, ángulos de

incidencia, desfases o atenuaciones. A este complejo conjunto se le conoce como efecto multicamino.

Como consecuencia de estos efectos, el receptor será alcanzado por múltiples contribuciones o réplicas de la señal original transmitida, pudiendo sufrir cada réplica diversos procesos de reflexión, difracción y difusión. Las distintas réplicas recorrerán diferentes trayectorias de distinta longitud hasta su destino. Por tanto, alcanzarán el receptor con diferentes ángulos de incidencia, atenuaciones, desfases y retardos de propagación. Este efecto recibe el nombre de efecto multicamino. Finalmente, las distintas contribuciones se sumarán por superposición en el receptor tal que se puede producir una interferencia constructiva o destructiva.

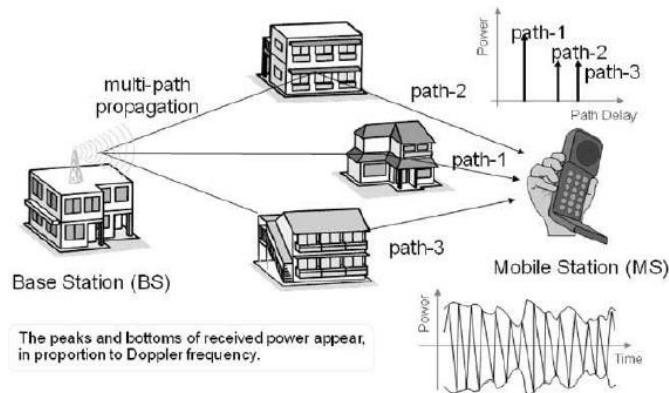


Figura 5. Propagación multicamino [12]

El efecto multicamino [10] es capaz de modificar en mayor o menor medida la amplitud, frecuencia y fase de la señal en recepción. Dichos cambios varían con respecto al tiempo, por lo que se dice que el canal de comunicaciones es cronovariante.

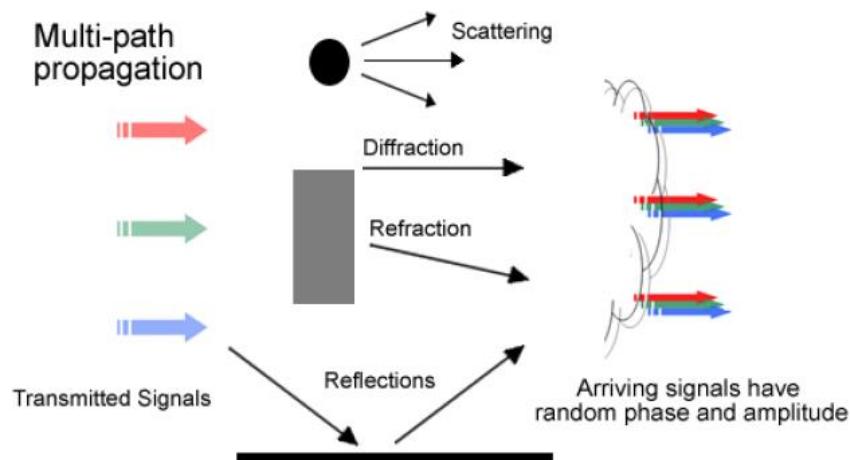


Figura 6. Principales mecanismos de transmisión multicamino [12]

2.6. Sistemas de medidas

2.6.1. Introducción

Previamente en este capítulo, se ha informado de la importancia y complejidad que supone hacer una buena caracterización del canal móvil. Las campañas de medidas son muy útiles para esa caracterización, al permitirnos conocer las características del entorno con la finalidad de efectuar la adecuada planificación de un modelo de propagación que sirva para cada situación y nos permita diseñar el sistema en función de las coberturas radio predichas. Es por eso que se necesitan instrumentos y métodos para medir el canal móvil.

Se puede modelar al canal como un filtro que modifica una señal entrante, así en un sistema de medida encontraremos un generador fijo de señales que reciben los equipos de medida tras su propagación a través del canal. Al conocer la señal emitida y la medición de la señal en recepción será posible determinar las características del canal.

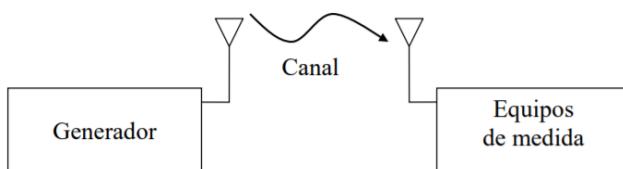


Figura 7. Esquema general de un sistema de medidas [10]

2.6.2. Sistemas de medida en banda estrecha

Los sistemas de medida en banda estrecha sólo son capaces de medir el nivel de potencia recibida, sin ofrecer información de la fase de la señal recibida en la mayoría de los casos. Son llamados sistema de banda estrecha porque el canal no va a ser medido a lo largo de un gran ancho de banda. También son de utilidad para estudiar la compatibilidad electromagnética [13]. A continuación, se van a presentar tres dispositivos que se utilizan en esta clase de sistemas: el medidor de campo, el dispositivo TEMS y el analizador de espectro.

Medidor de campo

Por medio de una antena que lleva conectada permite medir el nivel de campo eléctrico a su entrada. Algo que hace muy interesante a este dispositivo es el hecho de que pueda demodular la señal recibida con diversas técnicas: OFDM, QAM, QPSK, etc. Por esta característica desempeña un buen papel para orientar antenas receptoras de televisión. Mas allá de ello también puede medir potencia y voltaje expresadas en diferentes unidades: dBmV, dB μ V, dBm.



Figura 8. Medidor de campo Televés [14]

Mide canales analógicos y digitales. Se usa más comúnmente en sistemas vía satélite pero también se le requiere en sistemas GSM [15].

Dispositivo TEMS

Usado años atrás, este dispositivo era un receptor móvil que fue creado por Ericsson con el propósito de estudiar y planificar el sistema GSM [15]. Permitía la medición de diversa información acerca del tráfico recibido y enviado por el móvil, información de los canales de control o potencia de señales enviadas y recibidas.



Figura 9. Dispositivo TEMS pocket [16]

Actualmente TEMS ha sido progresivamente sustituido por aplicaciones móviles para *smart phones* que incorporan las mismas funcionalidades.

Analizador de espectros

Un analizador de espectros calcula el espectro de una señal de radiofrecuencia. También es capaz de evaluar la potencia instantánea de la señal en el dominio del tiempo. Sometiéndolo a las diferentes configuraciones que ofrece permite medir atributos de la potencia tanto de la estación móvil (MS) en del receptor móvil como de la estación base (BS).

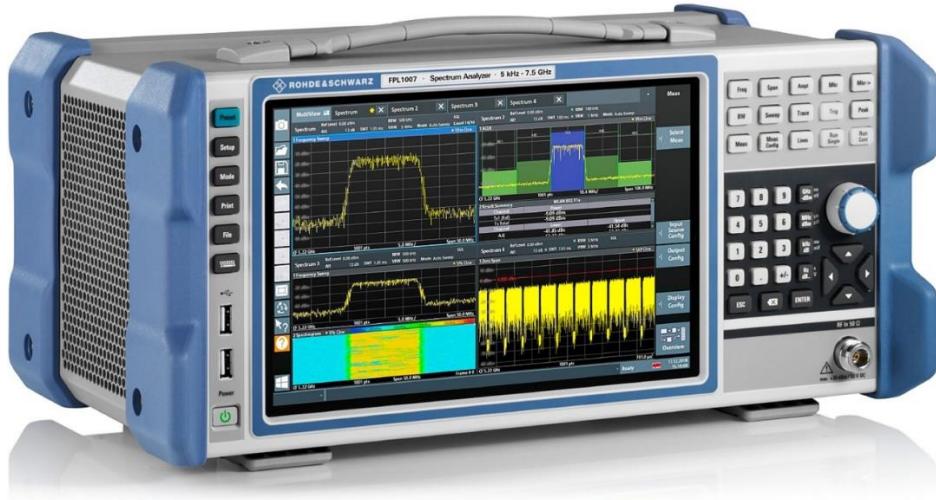


Figura 10. Analizador de espectros Rodhe&Schwarz[17]

2.6.3. Sistema de medidas en banda ancha

Con una reproducción de la estructura multicamino del canal radio se pueden medir todas las particularidades de la dispersión en frecuencia y en el tiempo. Para llegar a esa reproducción es necesario hacer la medición del perfil de retardo normalizado (PDP), que se extrae de la respuesta al impulso cronovariable $h(t,\tau)$ y es la respuesta del canal en un instante t a un impulso τ generado justo antes [15]. Aunque la transmisión de un impulso no es posible, sí lo es transmitir un pulso de muy corta duración (TBB).

Los sistemas de medida del PDP tradicionales se basan en la transmisión de pulsos de corta duración.

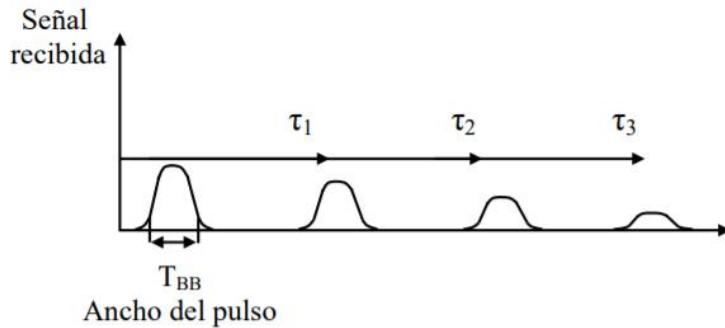


Figura 11. Rélicas recibidas cuando se transmite un pulso de corta duración [10]

Estos pulsos presentan un ancho de banda notable, muchas veces mayor que el ancho de banda de coherencia del canal, hecho que les hace a estos sistemas acuñar el nombre sistemas de medida de banda ancha.

Por lo general se debe medir el canal a lo largo de un gran ancho de banda para caracterizarlo por completo y así medir todas sus propiedades en los dominios de tiempo y frecuencia. A

continuación, se enumeran algunos equipos utilizados para la medición del radiocanal en banda ancha:

- “*Direct RF pulse system*”: Sistema de medida PDP tradicional basado en la transmisión de pulsos de corta duración.
- “*Spread Spectrum sliding correlator system*”: Sistema de medida PDP tradicional basado en la transmisión de pulsos de corta duración.
- “*Frequency domain channel sounding*”: No mide de forma directa la función $h(t,\tau)$, en lugar de ello mide su respuesta en frecuencia.

Capítulo 3. Diseño e integración en un sistema de medidas

3.1. Introducción

Anteriormente se ha remarcado lo importante y complejo que resulta caracterizar el canal móvil, lo que convierte a las campañas de medidas en un trabajo primordial para analizar cómo influye realmente un determinado terreno en la señal transmitida y así planificar correctamente el modelo de propagación que más encaje. Para poder llevar a cabo una campaña es necesario contar con instrumentos adecuados y ser conocedor de la metodología a seguir.

Puesto que el canal se modela como un sistema que altera una determinada señal entrante, el sistema de medidas se compone de un generador fijo de señales recibidas por los aparatos de medida [15]. Si se conocen los atributos de la señal emitida y la señal recibida se podrán calcular las características de canal.

3.2. Implementación de los componentes

En nuestro caso, el sistema de medidas utilizado se basa en:

- ✓ El analizador de redes VNA ZVK (10MHz-40GHz) de Rhode&Schwarz.

El análisis vectorial de redes (VNA) es uno de los principales métodos de medida de RF/microondas. Los analizadores de redes ofrecen unas excelentes características de RF y una extensa gama de funciones de análisis que ayudan al usuario a evaluar parámetros importantes a simple vista [17]. En nuestro caso utilizaremos el analizador de redes para generar la señal que nos permitirá caracterizar el radiocanal.

- ✓ El analizador de espectros portátil MS2090A (9kHz-32GHz) de Anritsu.

Se ocupará de recibir la señal emitida por el analizador de redes y proporcionar medidas de potencia del radiocanal. En el siguiente punto de esta redacción se analiza en profundidad sus características y funcionalidad en el proyecto.

- ✓ El amplificador ZVE-8G+ (2-8GHz).

Se encargará de amplificar la señal generada por el analizador de redes para que el analizador de espectros pueda recibirla a distancias significativas.

- ✓ Fuente de alimentación de banco RS pro

Para proveer de energía al amplificador

- ✓ 2 antenas ultra-wideband (0.8-40GHz) STEATITE Q-PAR.

Las dos frecuencias con las que trabajaremos serán de 1 GHz para la prueba en las inmediaciones del cuartel de Antiguones y 3.5 GHz en el cultivo de limoneros.

- ✓ Antena GPS con conector SMA para usar el sistema GPS que integra el Anritsu.

En el punto 4.3.3 se trata el sistema GPS y cómo está implementado en el analizador de espectros de Anritsu.

- ✓ Ordenador portátil Asus TUF Gaming FX505GT-BQ025

Este ordenador correrá el software desarrollado en MATLAB que automatiza el proceso de medidas a través de una conexión TCP/IP

- ✓ Teléfono móvil Samsung Galaxy S8

Se activará su función de *hotspot* simulando a un router con la función de servidor DHCP.

Mediará la comunicación entre el ordenador que corre el software de medidas y el MS2090A

- ✓ Cables, conectores y transiciones

Para conectar las antenas al analizador de espectros y al amplificador, y para conectar el amplificador al analizador de redes se utilizaron cables coaxiales con conectores SMA (de color azul en la figura 12). Para conectar la fuente de alimentación al amplificador se utilizaron dos cables banana-cocodrilo (uno para el terminal positivo y otro para el negativo, de color rojo y negro respectivamente en la figura 12).

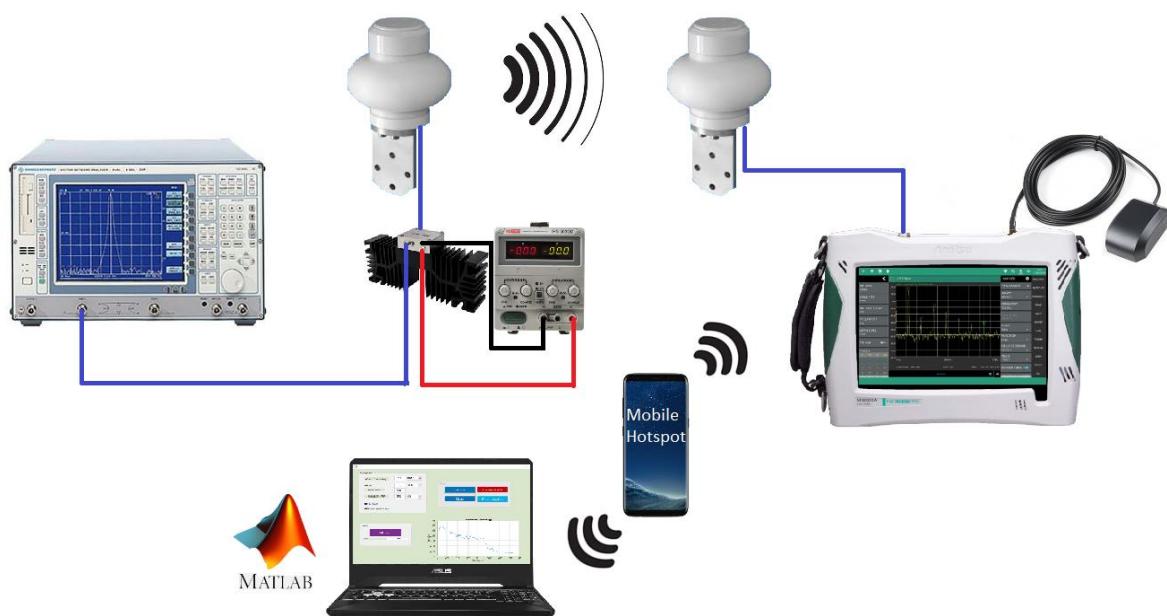


Figura 12. Esquemático de sistema de medidas utilizado

Capítulo 4. El analizador de espectro Field Master Pro MS2090A

4.1. Introducción

A lo largo de este capítulo se describe, desde lo más general a lo más particular, el analizador de espectros utilizado para la campaña de medidas. Su aspecto físico, sus diferentes modos de funcionamiento, sus parámetros de configuración y demás características que ofrece.

4.2. Descripción y aspecto físico

El MS2090A Field Master Pro de Anritsu es un analizador de señal portátil basado en sintetizador que proporciona resultados de medición precisos, con una cobertura de frecuencia continua que va de 9 KHz a 54 GHz. El instrumento está diseñado para monitorear, medir y analizar señales ambientes. Las mediciones se pueden realizar fácilmente utilizando las funciones principales del instrumento: frecuencia, SPAN, amplitud y ancho de banda [18]. Una pantalla táctil capacitiva permite una lectura y entrada de datos rápida y fácil.

Las mediciones típicas incluyen interferencia en banda y análisis de espectro de transmisión. Hay opciones disponibles para RF, demodulación avanzada y medición *over-the-air* (OTA). Proporciona una gama completa de capacidades de marcador (como funciones de pico, centro y delta) para análisis más rápido y completo de las señales mostradas.

Las medidas y configuraciones se pueden almacenar internamente en el instrumento o en una unidad flash USB para su posterior recuperación.

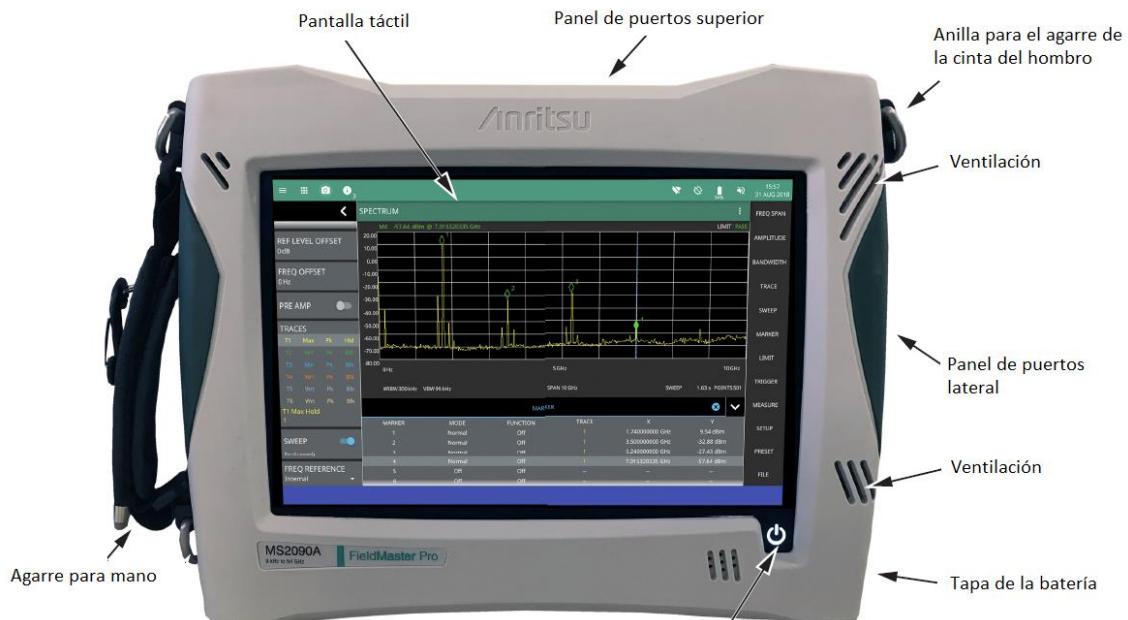


Figura 13. Panel frontal del instrumento Field Master Pro [18]

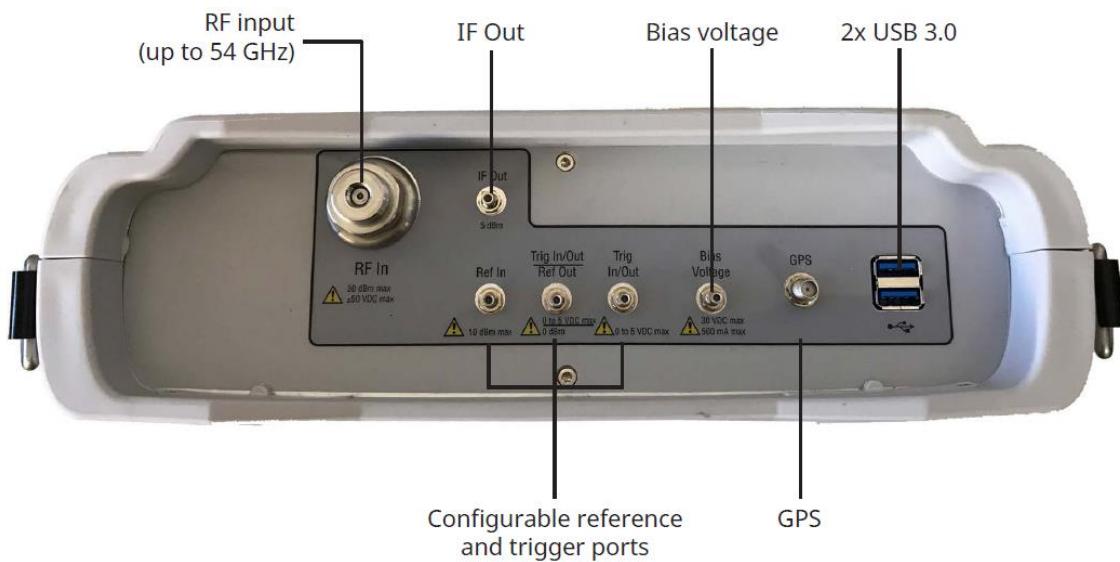


Figura 14. Panel de conectores superior [18]

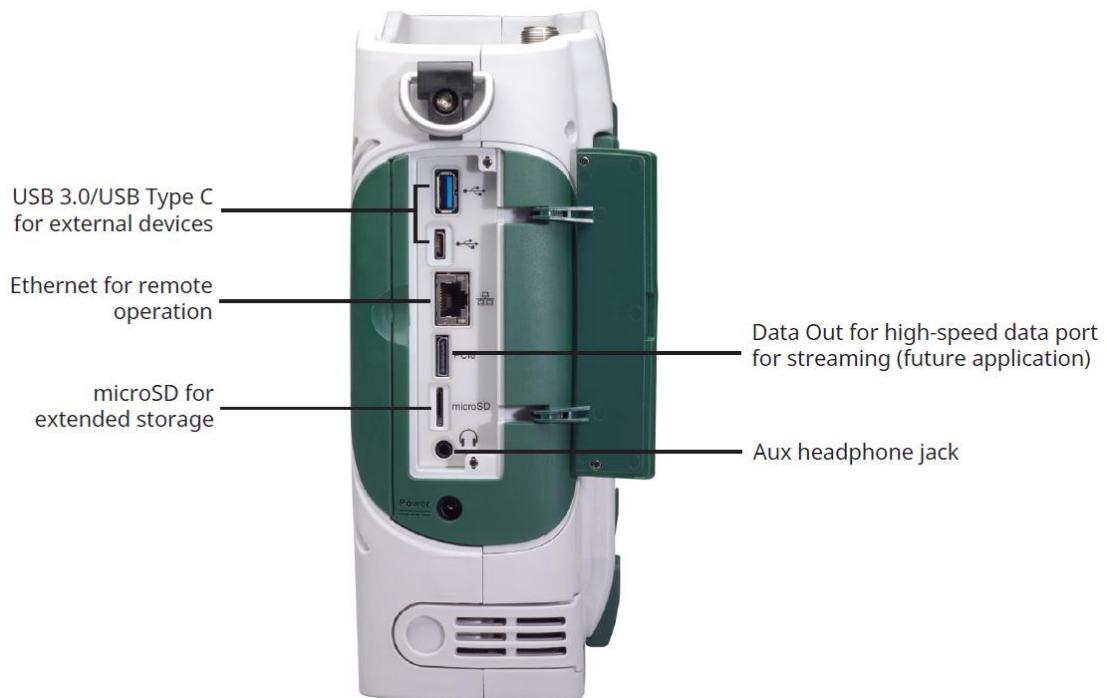


Figura 15. Panel de conectores lateral [18]

4.3. Modo analizador de espectro

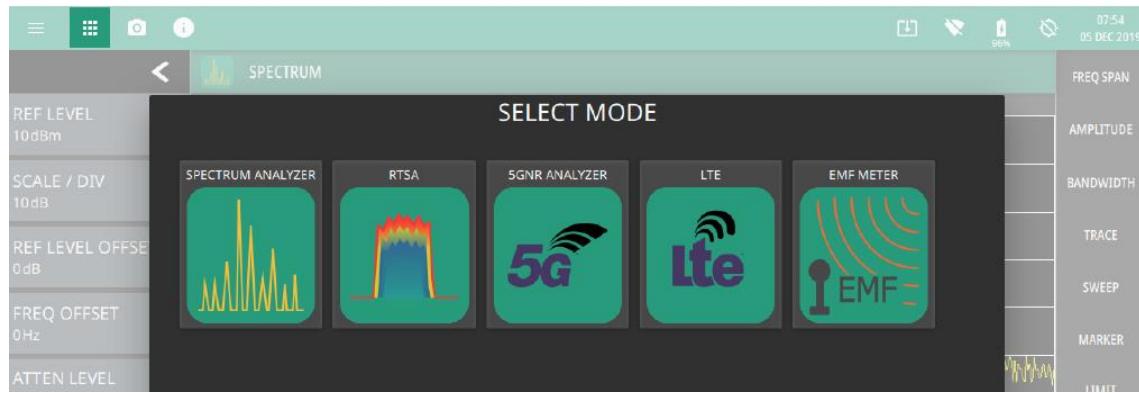


Figura 16. Vista previa menú de selección modos de funcionamiento [18]

Como vemos en la figura 16 este aparato tiene cinco modos distintos de funcionamiento. En nuestro caso sólo tenemos activo el modo *Spectrum analyzer* puesto que es el único que vamos a necesitar. Para trabajar con los demás se deberá abonar un determinado importe a la marca para que lo active.

El Anritsu en modo analizador de espectro tiene por defecto la vista normal de espectro (spectrum), que es la adecuada para ver señales en el dominio de la frecuencia y así, poder estudiar sus amplitudes, anchos de banda y cualidades armónicas.

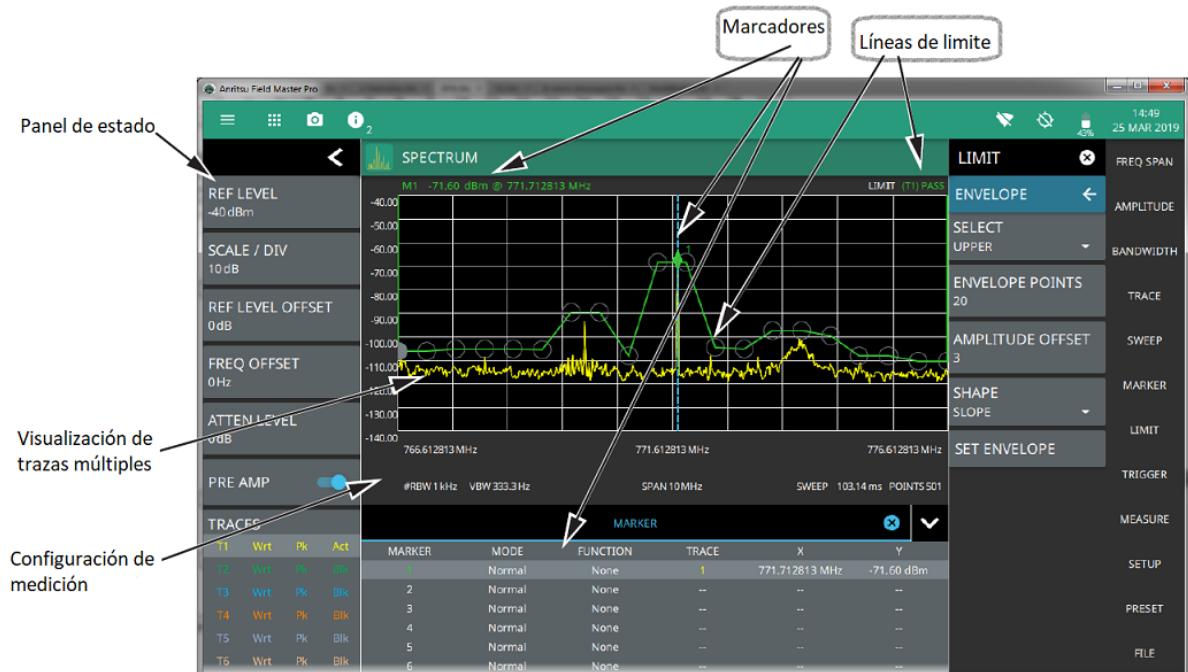


Figura 17. Vista predeterminada analizador de espectro [18]

El panel de estado nos muestra los datos e información relevantes de la configuración actual de medición y visualización.

Los marcadores se muestran como diamantes verdes en la traza a la que están asignados. El marcador que se está usando en un determinado momento se muestra como un diamante verde relleno con color y lo atraviesa una línea vertical discontinua. La amplitud y frecuencia del marcador aparecen en texto verde en el borde superior izquierdo de la pantalla. En el ejemplo de la figura 17 se muestra un marcador en el pico de la señal. Puedes colocar un marcador en el pico de forma rápida si tocas dos veces cerca de la ubicación del pico.

En la pantalla se pueden visualizar hasta seis trazas, de ahí que pueda haber varios marcadores al mismo tiempo y se tenga que seleccionar el deseado. Este aparecerá coloreado; como se ha especificado anteriormente. Cada traza tendrá un color diferente y cada una de ellas se podrá configurar independientemente.

4.3.1. Configuración de traza (Trace Menu)

Traza hace referencia a la trayectoria que describe la señal que dibuja el analizador de espectro.

Cuando seguimos la ruta (MEASURE > VIEW > Combined) los menús “TRACE” y “SET UP” se hacen visibles. En el menú Trace es de vital importancia entender el funcionamiento de los tipos de traza que aparecen en el desplegable de la pestaña TYPE que vemos en la figura siguiente.

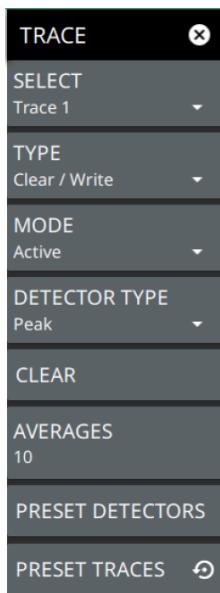


Figura 18. Vista previa menú de traza [18]

SELECT: selecciona una de las seis posibles trazas. Hay que tener en cuenta que todos los datos del espectrograma se crean a partir de una sola traza.

TYPE: selecciona uno de los siguientes tipos de trazas:

- *Clear/Write*: borra la traza después de completar cada barrido y actualiza a los nuevos valores.
- *AVERAGE*: promedio de los N barridos anteriores. El número de barridos se muestra en la tabla *TRACES* del panel de estado.
- *MAX HOLD*: representa el valor máximo desde que comenzó el barrido. El número de barridos se muestra en la tabla *TRACES* del panel de estado.
- *MIN HOLD*: representa el valor mínimo desde que comenzó el barrido. El número de barridos se muestra en la tabla *TRACES* del panel de estado.
- *ROLLING AVERAGE*: es el promedio de las últimas N trazas, donde N es el valor establecido en “*AVERAGES*”. El número de barridos se muestra en la tabla *TRACES* del panel de estado.
- *ROLLING MAX HOLD*: es el valor promedio de las últimas N trazas, donde N, es el valor establecido en “*AVERAGES*”. El número de barridos se muestra en la tabla *TRACES* del panel de estado.
- *ROLLING MIN HOLD*: es el valor mínimo de las últimas N trazas, donde N es el valor establecido en “*AVERAGES*”. El número de barridos se muestra en la tabla *TRACES* del panel de estado.

Una vez comprendido bien cómo se obtiene la traza en función de los parámetros de configuración del menú “*TRACE*”, se presentan en la siguiente captura seis trazas cada una de ellas con una selección diferente de “*TYPE*” combinadas con diferentes tipos de detección (*DETECTOR TYPE*):

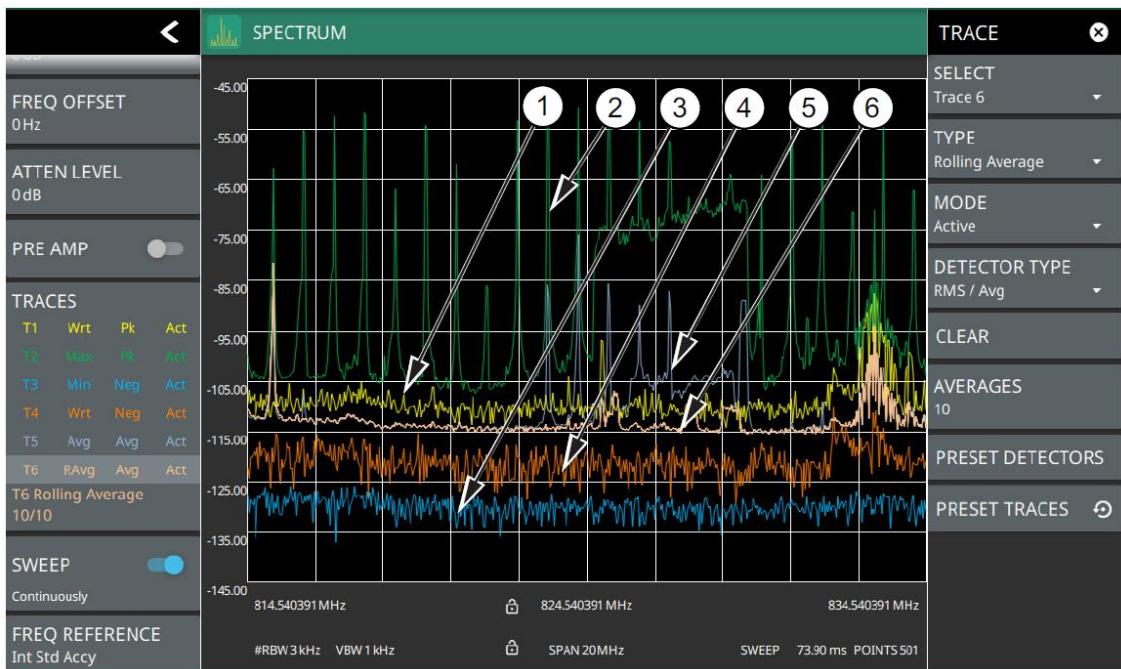


Figura 19. Vista de pantalla con 6 trazas con diferentes configuraciones [18]

1. *Clear/Write y Peak Detection*: esta es la configuración por defecto. La traza se limpia en cada barrido y se usa el mayor valor medido para cada uno de los puntos visualizados.
2. *Max Hold y Peak Detection*: cada punto de rastreo conserva su valor máximo y se utiliza el mayor valor obtenido durante la visualización.
3. *Min Hold y Negative Detection*: cada punto de rastreo conserva su valor mínimo y se utiliza menor valor obtenido en cada uno de los puntos a lo largo de la visualización.
4. *Clear/Write y Negative Detection*: los puntos de rastreo se borran durante cada barrido y se utiliza el punto de medición más pequeño para cada punto de visualización.
5. *Average y RMS/Average Detection*: cada punto de la traza es solución promedio de los N barridos anteriores. RMS/Average Detection depende de la configuración del tipo de ancho de banda de video (Bandwith > VBW). Cuando el tipo de VBW/Average se establece en “Lineal”, se detecta la potencia promedio de cada punto de medición. Por otro lado, si “VBW/Average” se fija a “Logarithmic” se muestra el promedio logarítmico clásico (potencia).
6. *Rolling Average*: da a cada punto de la traza el valor promedio de las N últimas muestras, donde N es la configuración “AVERAGES”.

4.3.2. Tipos de mediciones

- La medición de potencia del canal (*Channel power*) es una de las mediciones más comunes realizadas a un transmisor radio. Mide la potencia de salida o potencia del canal de un transmisor en un rango de frecuencias. Con ella pueden detectarse fallas del sistema, que pueden estar en los amplificadores de potencia o en los circuitos de filtro. Se puede usar para validar el rendimiento del transmisor, cumplir con las regulaciones gubernamentales o para mantener la interferencia general del sistema al mínimo. Para acceder a este pulsamos “*Measures*” y después “*Channel Power*” en el Anritsu.

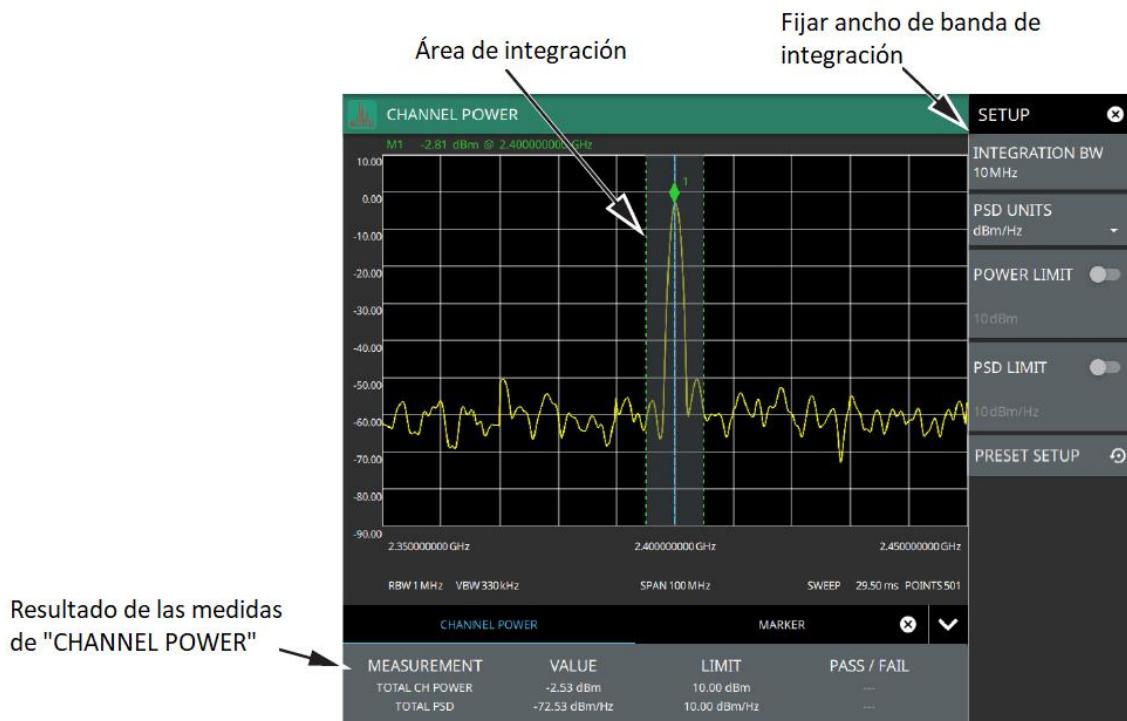


Figura 20. Vista analizador operando en "Channel Power" [18]

En la figura 20 podemos ver un ejemplo de medida utilizando “Channel Power” tal y como el que nosotros hemos usado en las medidas. En la parte de la derecha introducimos el ancho de banda de integración, que delimita el ancho de banda de la señal representada del cuál queremos conocer su potencia. En la parte de debajo de la pantalla, el display arroja el resultado de esa potencia en dbm.

- *Occupied Bandwith* (Ancho de banda ocupado): es una medida común realizada en transmisores radio. Calcula el ancho de banda de frecuencia ocupado.

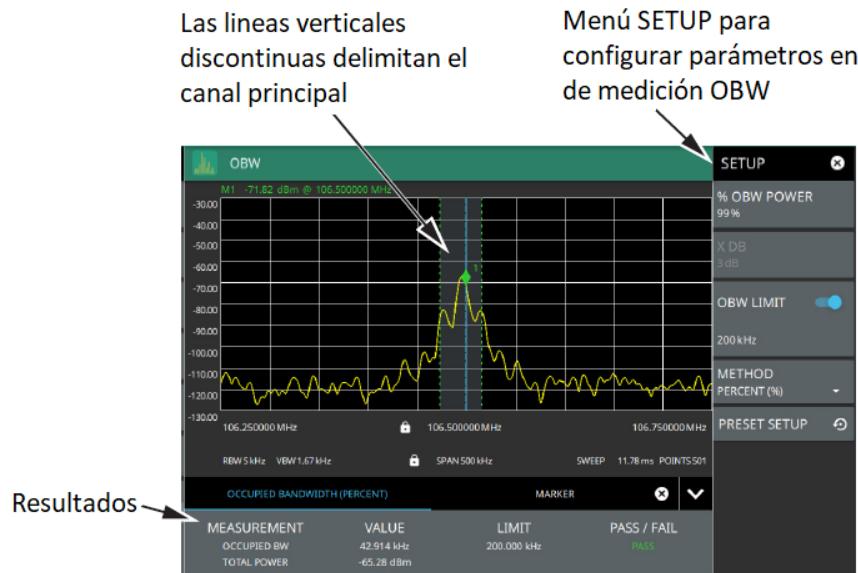


Figura 21. Vista analizador operando en “Occupied Bandwith” [18]

- *Adjacent Channel Power* (potencia del canal adyacente): mide la potencia que se filtra a los canales de transmisión adyacentes. Los espacios que delimitan las diferentes líneas discontinuas es el ancho de banda de integración de diferentes canales. Su funcionamiento se basa en “*channel power*”.

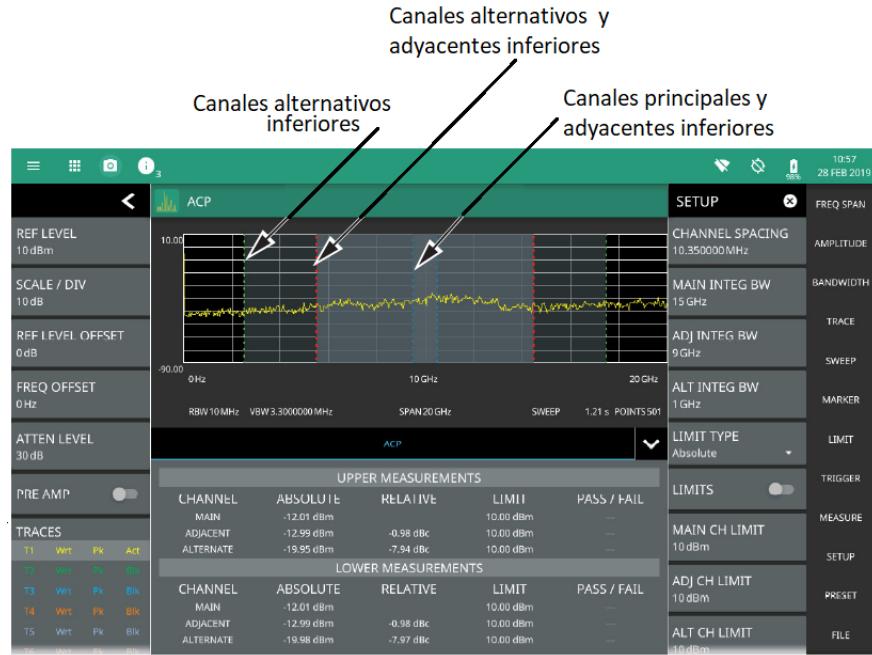


Figura 22. Vista analizador operando en "Adjacent Channel Power" [18]

- *Spectrum emission mask* (Medición de máscara de emisión de espectro): identifica el nivel de potencia de las emisiones transitorias fuera de banda que se encuentran fuera del ancho de banda del canal de la señal. Por tanto, la medición SEM muestra las emisiones que interfieren con otros canales.



Figura 23. Vista analizador operando en “Spectrum emission Mask” [18]

4.3.3. GPS para referenciar medidas

El sistema de posicionamiento global o Global Positioning system (GPS) se trata de un sistema de navegación basado en satélites (GNSS) que fue desarrollado por el Departamento de Defensa de los Estados Unidos a principios de los años 70. Nos permite saber la localización, velocidad y otros datos, como la altura en la que nos encontramos, mediante el receptor adecuado. Funciona a través de una red de alrededor de 24 satélites que se encuentran en órbita a unos 20.000 km de altura, con órbitas distribuidas para que en todo momento haya al menos cuatro satélites visibles en cualquier punto de la Tierra.

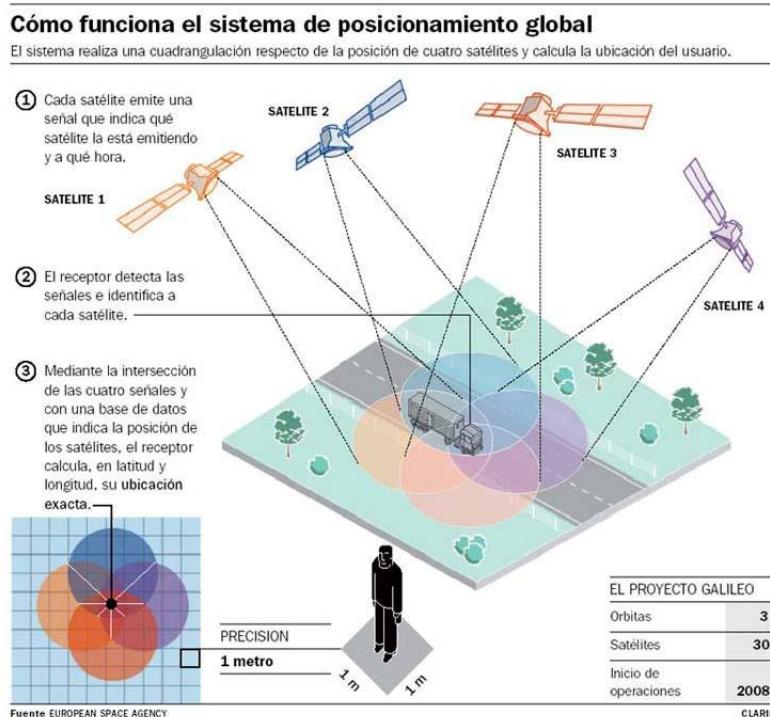


Figura 24. Funcionamiento del sistema de posicionamiento global [19]

El Master Field Pro de Anritsu lleva un receptor de posicionamiento global incorporado que puede proporcionar información de latitud, longitud, altura y tiempo UTC. Sólo será necesario conectar una antena GPS al conector destinado para ello en la parte superior del instrumento. Devuelve la latitud y longitud en formato de coordenadas decimales (DEG).

4.4. Configuración para la programación y operatividad remota

4.4.1. Conexión y configuración de la interfaz de red

El MS2090A utiliza Ethernet o WLAN (Wi-Fi) para comunicarse de forma remota con un controlador. La mayoría de las funciones del instrumento (excepto encendido/apagado) se pueden controlar mediante una conexión de red a un PC conectado directamente (con un cable

cruzado Ethernet o Wi-Fi peer-to-peer/ad hoc) o a través de conexión red de área local. El software del instrumento es compatible con el protocolo de red TCP/IP raw socket.

La red Ethernet utiliza una topología de bus o estrella en la que todos los dispositivos de interfaz están o bien conectados a un cable central llamado bus, o bien a un hub. Ethernet utiliza el método de acceso *Carrier Sense Multiple Access/Collision Detection (CSMA/CD)* para manejar transmisiones simultáneas a través del bus.

Este estándar permite que los dispositivos de red detecten el uso simultáneo del canal de datos, lo que se le denomina colisión, y proporciona un protocolo de contención. Cuando un dispositivo de red detecta una colisión, el estándar CSMA/CD establece que los datos sean retransmitidos después de esperar una cantidad de tiempo aleatoria. Si se detecta una segunda colisión, los datos vuelven a ser retransmitidos tras esperar el doble de tiempo. Esto se conoce como “*exponential back off*”.

Wi-Fi utiliza una topología en estrella similar en la que todos los dispositivos están conectados a un punto de acceso. Utiliza el método de acceso “*Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA)*” para manejar transmisiones. CSMA/CA no detecta colisiones, sino que las evita mediante el uso de mensajes de control. Si el mensaje de control choca con otro mensaje de control de otro nodo, significa que el medio no está disponible para la transmisión y el algoritmo de retroceso se aplica antes de intentar otra transmisión.

La configuración de TCP/IP requiere lo siguiente:

- Dirección IP: cada computadora y dispositivo electrónico en una red TCP/IP requiere una dirección IP, que está formada por cuatro números (cada uno entre 0 y 255) separados por puntos. Por ejemplo: 128.111.122.42 es una dirección IP válida.
- Máscara de subred: la máscara de subred distingue la parte de la dirección IP que es el ID de red de la parte que es el ID de la estación. La máscara de subred 255.255.0.0, cuando se aplica a la dirección IP proporcionada anterior, identificaría el ID de la red como 128.111 y el ID de la estación como 122.42. Todas las estaciones de la misma red de área local deben tener el mismo ID de red, pero diferentes ID de estación.
- Máscara de subred: la máscara de subred distingue la parte de la dirección IP que es el ID de red de la parte que es el ID de la estación. La máscara de subred 255.255.0.0, cuando se aplica a la dirección IP proporcionada anterior, identificaría el ID de la red como 128.111 y el ID del dispositivo como 122.42. Todos los equipos de la misma red de área local deben tener el mismo ID de red, pero diferentes ID de estación.

- Puerta de enlace predeterminada (*Default Gateway*): sirve para comunicarse con redes externas a la red de área local (*LAN*) identificada por el ID de red. Una puerta de enlace es una computadora o dispositivo electrónico que está conectado a dos redes diferentes y puede mover datos TCP/IP de una red a otra. Una sola LAN que no está conectada a otra LAN requiere una configuración de puerta de enlace predeterminada de 0.0.0.0.
- Dirección Ethernet: una dirección Ethernet, o *Media Access Control* (MAC), es un valor único de 48 bits que identifica una tarjeta de interfaz de red con el resto de la red. Cada tarjeta de red tiene una dirección Ethernet única almacenada permanentemente en su memoria.

DHCP

El MS2090A se puede configurar para el Protocolo de configuración dinámica de host (DHCP), un protocolo de Internet que automatiza el proceso de configuración de direcciones IP para dispositivos que usan TCP/IP, y es el método más común para configurar un dispositivo para uso en red.

Para determinar si una red está configurada para DHCP, conecte el instrumento a la red y seleccione el protocolo DHCP. Apague y encienda el instrumento. Si la red está configurada para DHCP, la dirección IP asignada debe ser la que se muestra en la configuración de red.

Socket

Los sockets son mecanismos de comunicación entre procesos que permiten que un proceso emita o reciba información con otro proceso incluso estando en distintas máquinas. Desde el punto de vista de programación, un socket no es más que un "fichero" que se abre de una manera especial. Una vez abierto se pueden escribir y leer datos de él con las funciones de *read()* y *write()*.

Un socket queda definido por una dirección IP, un protocolo y un número de puerto. Para el caso concreto de TCP/IP, que es el precisa nuestro script, un *socket* se define por una dupla Origen – Destino. Tanto el origen como el destino vienen indicados por un par (ip, puerto).

Para operar remotamente con el MS2090A, se debe establecer una conexión de *socket raw* TCP/IP sin procesar por el puerto 9001 [18]. La aplicación remota puede conectarse a la dirección IP del instrumento o a su *HOSTNAME*. Si usa DHCP en lugar de una IP estática, usar *HOSTNAME* puede ser más fiable para encontrar un instrumento en una red.

Conexión física de red mediante cable

La comunicación entre el instrumento y otros dispositivos en la red se realiza a través de un cable de interfaz de categoría cinco (CAT-5) conectado a una red. Este cable utiliza cuatro pares trenzados de hilos de cobre aislados terminados en un conector RJ45. El cableado CAT-5 es capaz de admitir frecuencias de hasta 100 MHz y su velocidad de transferencia de datos puede alcanzar hasta 1 Gbps.

En la siguiente figura vemos un ejemplo de configuración Ethernet (y la ruta para acceder a esa configuración) en la que se ha seleccionado DHCP para que asigne de forma automática los valores de red y la ruta necesaria para acceder a ella. También podríamos introducir de forma manual los valores TCP/IP necesarios. Como ya se ha mencionado anteriormente también podríamos realizar la comunicación mediante Wi-Fi.

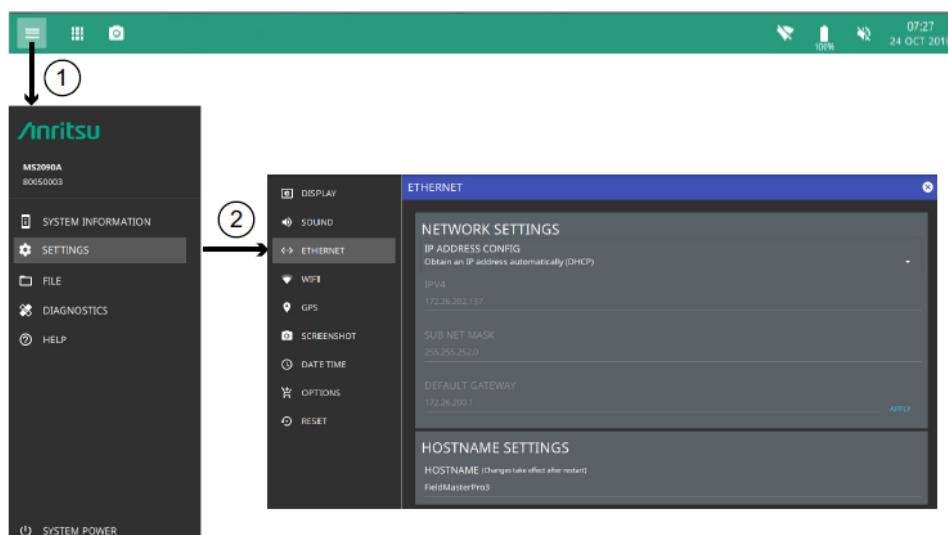


Figura 25. Menú configuración de red Anritsu [18]

4.4.2. Estándar SCPI

El estándar SCPI define un conjunto de comandos de programación y estándares del que pueden hacer uso todos los instrumentos compatibles con SCPI. Este estándar fue desarrollado para proporcionar un entorno práctico enfocado al desarrollo de programas.

Para ello define mensajes de control estandarizados, respuestas de los instrumentos, y el formato de los mensajes de todos los instrumentos compatibles.

Un comando típico SCPI consta de una o más palabras clave y de parámetros. Las palabras clave pueden contener caracteres en mayúscula y en minúscula. Al igual que en cualquier lenguaje de programación, las palabras clave y la sintaxis de los comandos tienen que ser exactas.

A excepción de los comandos comunes, cada palabra clave tiene una forma larga y una corta. La forma corta es una abreviación de la forma larga del comando. Por ejemplo, la forma larga de la palabra clave de comando para controlar la pantalla del instrumento es :DISPlay. La palabra clave de forma corta son generalmente los primeros cuatro caracteres de la forma larga (:DISP para :DISPlay). Sin embargo cuando la forma larga tiene más de cuatro caracteres y el cuarto carácter es una vocal, esta vocal se elimina y la forma corta pasa a ser los primeros tres caracteres de la palabra clave. Por ejemplo, la forma abreviada de la palabra clave :POWER es :POW.

En un mismo comando se pueden combinar palabras clave en formato corto con palabras clave en formato largo. Por ejemplo :SENS:FREQuency:STAR es una forma correcta de comando. Sin embargo, :SENS:FREQuen:STAR no es una forma correcta de comando porque :FREQuen no es ni la forma corta ni larga de la palabra clave del comando.

Un comando SCPI comienza generalmente con dos puntos ":" e incluye varias palabras clave y parámetros.

Los dos puntos iniciales son opcionales, pero es obligatorio que cada uno de los comandos siguientes esté separado del anterior con ":". El final de la cadena de comando y el primer parámetro debe estar separado por un espacio. Por ejemplo:

```
:DISPlay:POINTcount 201
```

También se utiliza una coma "," para separar los parámetros de comando. Por ejemplo:

```
:SISTEMA:FECHA 2018,10,31
```

Se pueden combinar comandos completos en una sola línea y separarlos con un punto y coma ";" como sigue:

```
:SENSe:FREQuency:STARt 1000; :SENSe:FREQuency:STOP 5000
```

Algunas palabras pueden tener un sufijo correspondiente a un parámetro para diferenciar entre múltiples características del instrumento. Cuando estas funciones están disponibles, el parámetro de palabra clave se identifica y se encierra entre llaves. Por ejemplo, :TRACe <n> se ingresa como: TRACe1 o :TRACe3.

Los parámetros podrán ser de tipo entero, numérico, carácter (char) o cadena (string), entre otros.

4.5. Parámetros específicos a tener en cuenta: CF, SPAN, RBW, VBW, SWEEP POINTS

- Se denomina frecuencia central (CF) del analizador a la que corresponde con la frecuencia en el punto medio de la pantalla.
- El SPAN varía la anchura del espectro de frecuencia a visualizar.
- La selectividad en frecuencia es una medida del rendimiento de un receptor para responder sólo a la señal que está sintonizado y rechazar otras señales cercanas en frecuencia.
- Ancho de banda de resolución o *Resolution Bandwidth* (RBW). En función de su valor habrá una mejor o peor selectividad en frecuencia. La elección del ancho de banda de resolución depende de diferentes factores.
 - Los filtros necesitan tiempo en arrojar un valor de salida estable, y esa cantidad de tiempo debe conocerse. Cuanto más estrecho sea el ancho de banda del filtro (o ancho de banda de resolución), más tiempo requiere para estabilizarse, por lo tanto, más lenta será la velocidad de barrido y menos medidas por unidad de tiempo podrán ser tomadas.
 - La señal que está siendo medida. Si se tienen que medir por separado una señal poco espaciada en frecuencia de otra que la que también se está recibiendo una potencia considerable, se requiere un ancho de banda estrecho para no incluir energía procedente de la señal que no queremos cuantificar. Por otra parte, una medición de banda estrecha produce una separación de los componentes en frecuencia de la señal recibida, lo que dará lugar a una medición que incluye picos separados. Como vemos cada uno tiene sus ventajas e inconvenientes, de ahí que la decisión final dependa del tipo de medición que se requiera.
 - El ruido, que está presente en un amplio rango de frecuencias siempre está incluido en mayor o menor cantidad en una medición, y por supuesto, es un factor a tener en cuenta en la elección del RBW, porque una medición que contiene demasiado ruido no será representativa ni válida. A medida que se aumenta el ancho de banda, mayor ruido se incluye en una medición y menor precisión tendrá, o lo que es lo mismo, tendrá una menor selectividad en frecuencia.
- Ancho de banda de video o *Video Bandwidth* (VBW). Se trata de otro tipo de filtro que se usa normalmente tras el detector de un analizador de espectro. Como el RBW también afecta al ruido que se muestra en pantalla, pero de una forma diferente. En el filtrado de video el nivel de ruido medio sigue siendo el mismo, pero se reduce su variación,

como si realizara un suavizado del ruido. No mejora la sensibilidad, pero si consigue que se pueda discernir mejor la traza de la señal. Como regla general el ancho de banda de video se suele fijar en un valor que sea factor de 10 o 100 del ancho de banda de resolución.

- Tiempo de barrido o *Sweep time*. Tiempo que toma el analizador en actualizar todos los valores de la traza que está escudriñando. Que ese tiempo sea mayor o menor depende por un lado del RBW y del VBW, pues la velocidad de barrido (*Sweep speed*) proviene del tiempo de respuesta de los filtros que los forman; y del rango de frecuencias con el que se está trabajando.

El Anritsu Field Master Pro calcula y configura sin intervención del usuario la velocidad más rápida de barrido que dará resultados precisos, tardando así el menor tiempo posible en realizar un nuevo barrido. Para valores bajos de RBW o VBW la velocidad de barrido será más lenta, mientras que para un mayor RBW o VBW será más rápida.

- Número de puntos de barrido o *Sweep points*. Establece el número de puntos por barrido que se muestra en cada trazo. El uso de más puntos proporciona mayor resolución, mientras que el uso de menos reduce el tiempo necesario para acceder a una traza.

4.6. Configuración seleccionada para la medición

Teniendo en cuenta la información detallada en los puntos anteriores sobre el analizador de espectro detallaremos ahora los aspectos más importantes de la configuración elegida:

- Se ha usado el modo analizador de espectros del Anritsu, que como se había dicho al comienzo de este capítulo, es el único modo que se había adquirido a la marca.
- Se utiliza el modo de medida *Channel Power*, representando una sola traza AVG, frecuencia central 3.5 GHz, SPAN 50 kHz, nivel de referencia 15 db, *Integration BW* 6 kHz y Auto VBW. *Sweep points* = 300 e *Integration BW* = 100 Hz.

Capítulo 5. Desarrollo de la aplicación en MATLAB®

5.1. Introducción

Con el fin de caracterizar el radiocanal por medio del analizador de espectros MS2090A se ha recurrido al desarrollo de una aplicación en MATLAB que corriendo desde un ordenador establezca con él una comunicación remota.

Teniendo en cuenta lo que se ha expuesto en el capítulo 4, a lo largo de este capítulo se va a explicar las funciones que desempeña la aplicación y cómo antes de empezar a escribir código, se comprobó que se daba la correcta comunicación entre los dos equipos (ordenador y analizador). Se explica cómo se fueron añadiendo paso por paso nuevas funciones a la aplicación tras asegurar que las ya incluidas funcionaban de la forma esperada. También se explica el formato en que la aplicación guarda las medidas para poder trabajar con ellas y, al final se añade un manual de usuario que clarifica su uso.

5.2. Funcionalidades de la aplicación

El código programado y la interfaz dispuesta (parte de la aplicación que ve el usuario) deben ser capaces operando conjuntamente de:

- Establecer una conexión socket TCP/IP con el dispositivo de medidas y mantenerla hasta el fin de la campaña.
- Leer desde teclado los parámetros más comunes de configuración del analizador de espectros (frecuencia central, SPAN, número de puntos de barrido y ancho de banda de integración) para establecerlos en su configuración.
- Fijar a su valor de activada o no activada las funciones de preamplificador y de barrido continuo del analizador.
- Extraer del analizador cuantas veces se quiera la medida de potencia del radiocanal junto a las coordenadas GPS donde ha sido tomada dicha medida y junto al instante de tiempo “t” en el que ha sido tomada.
- Extraer del analizador de forma repetida y periódica, y durante el tiempo que se deseé, la medida de potencia del radiocanal junto a las coordenadas GPS donde ha sido tomada dicha medida y junto al instante de tiempo “t” en el que ha sido tomada.
- Cerrar la conexión socket TCP/IP y guardar la campaña de medidas en un archivo “.mat”.

5.3. Verificación de la comunicación remota

En un primer momento para comprobar que la comunicación se realizaba correctamente se recurrió al software que ofrece Anritsu en su web oficial para interactuar con él desde el PC, “*Field Masters Pro PC Software*”. Al ver la que sincronización con el Analizador era la apropiada, tal y como se muestra en la figura 27, se determinó que la comunicación era correcta.



Figura 27. Verificación configuración remota PC-Analizador Anritsu

Lo siguiente es comprobar que desde el entorno de MATLAB se pueden intercambiar los comandos SCPI sin problema. La forma más adecuada para hacerlo es utilizar la aplicación que ofrece este entorno, la cual se titula “*Instrument Control Toolbox*” que permite conectarse directamente con instrumentos tales como osciloscopios, generadores de funciones, fuentes de alimentación, instrumentos analíticos y, como en este caso, analizadores de señales [20]. Desde esta aplicación se puede generar un nuevo objeto de interfaz TCP/IP con el par IP-puerto requerido. Su funcionamiento es simple, en el campo “Data to write” se escribe el comando SCPI que se quiere transmitir al equipo. Si el comando es de tipo consulta y se espera una respuesta tras clicar en el botón “Query” aparece la respuesta del equipo en el campo “Response”.

En la imagen podemos ver que se realiza la consulta “*IDN?” y se devuelven los datos de fabricación de la unidad, por lo que se verifica que la comunicación es correcta.

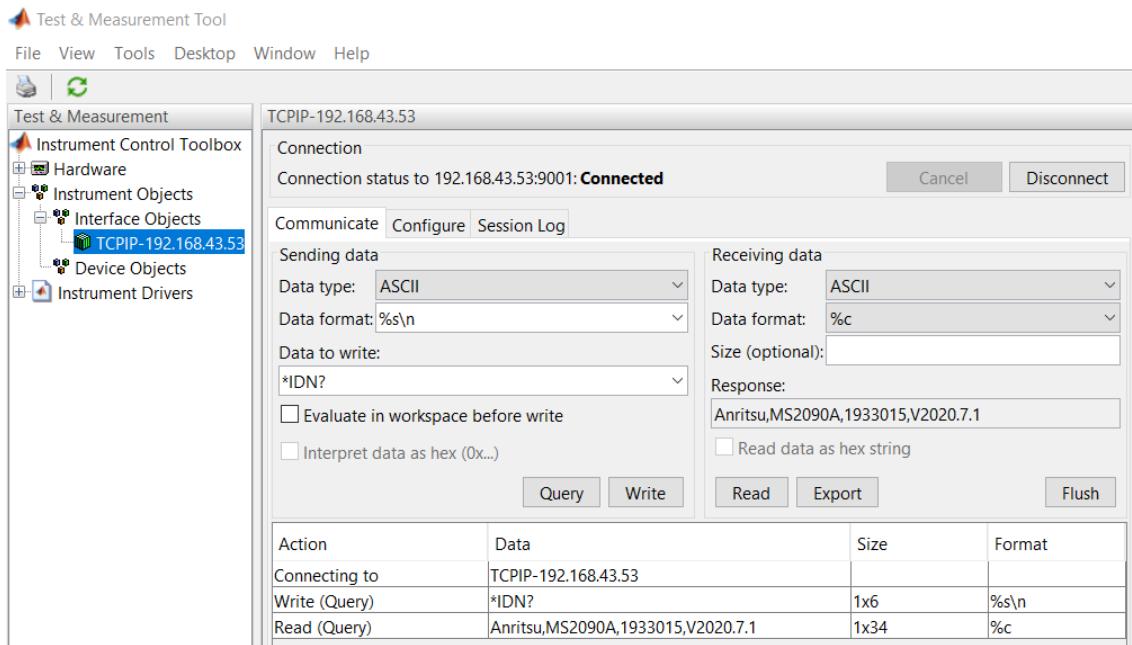


Figura 26. Verificación comunicación remota Entorno MATLAB PC-Analizador Anritsu

5.4. Guide de MATLAB

Puesto que nuestra herramienta de medidas precisa una comunicación continua con el analizador necesitamos un entorno en el que se pueda programar una aplicación.

Guide es un entorno de programación visual disponible en MATLAB para realizar y ejecutar programas que necesiten ingreso continuo de datos. Una aplicación GUIDE consta de dos archivos, el .m que contiene el código con las correspondencias de los botones de control de la interfaz y el archivo ".fig" que es el área de diseño gráfico de la aplicación [21]. Cada vez que se añade un elemento en la interfaz gráfica, se genera automáticamente código en el archivo .m. Los componentes que ofrece GUIDE para disponer en el ".fig" se muestran en la siguiente tabla:

Control	Valor de estilo	Descripción	Aspecto
Check box	'checkbox'	Indicar el estado de una opción o atributo	<input type="checkbox"/> Check Box
Editable Text	'edit'	Caja para editar texto	<input type="text"/> Edit Text
Pop-up menú	'popupmenu'	Provee de una lista de opciones	Pop-up Menu
List Box	'listbox'	Muestra una lista deslizable	<input type="button" value="Listbox"/>
Push Button	'pushbutton'	Invoca un evento inmediatamente	Push Button
Radio Button	'radio'	Indica una opción que puede ser seleccionada	<input checked="" type="radio"/> Radio Button
Toggle Button	'toglebutton'	Solo dos estados "1" o "0"	Toggle Button
Slider	'slider'	Usado para representar un rango de valores	<input type="range"/>
Static Text	'text'	Muestra un string de texto en una caja	Static Text
Panel button		Agrupa botones como un grupo	Panel Push Button Push Button
Button Group		Permite exclusividad de selección con los radio button	Button Group <input type="radio"/> Radio Button <input checked="" type="radio"/> Radio Button

Tabla 3. Componentes que ofrece MATLAB Guide para disponer en el área de diseño (.fig) [21]

Para entender el funcionamiento se ha realizado una GUI (Interfaz gráfica de usuario) que solicita la introducción de dos operandos que sumará o restará según escoja el usuario.

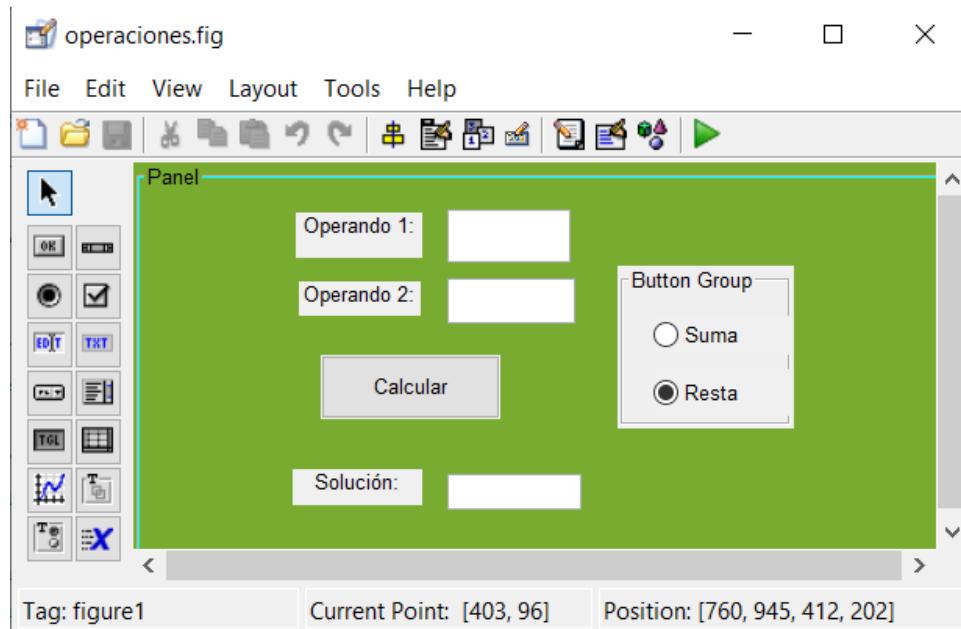


Figura 27. Ejemplo de GUI muy sencilla en Matlab Guide

Al añadir por ejemplo el EditText donde se introducirá operando 1 se genera automáticamente su función CreateFcn y Callback. Se puede ver directamente haciendo clic derecho en el componente, entonces se abre un menú desplegable en el que se pulsa "View Callbacks".

```

function editOperando1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editOperando1 (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 editOperando2_Callback(hObject, eventdata, handles)
% hObject    handle to editOperando2 (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 editOperando2 as
% text
% str2double(get(hObject,'String')) returns contents of
% editOperando2 as a double

% --- Executes during object creation, after setting all properties.

```

Fundamentos de Matlab GUIDE: guidata, handles, get, set

Antes de explicarlos conceptualmente, es una buena didáctica describir la función Callback de *pushbuttonCalcular* que tiene esta forma:

```

% --- Executes on button press in pushbuttonCalcular.

function pushbuttonCalcular_Callback(hObject, eventdata, handles)
% hObject    handle to pushbuttonCalcular (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
operando1 = str2num(get(handles.editOperando1,'String'));
operando2 = str2num(get(handles.editOperando2,'String'));
if (get(handles.radioButtonSuma,'Value') == 1)
    resultado = operando1 + operando2;
elseif (get(handles.radioButtonResta,'Value') == 1)
    resultado = operando1 - operando2;
end
set(handles.editSolucion,'String',resultado);
guidata(hObject,handles);

```

- 1) En las variables “operando1” y “operando2” se guardan los valores que introduce el usuario en los edit dispuestos en el Guide “operando1” y “operando2”, haciendo uso de un *get*.
- 2) Con *if* y *elseif* se comprueba cuál de los dos *radiobuttons* se han marcado haciendo uso de un *get*, para guardar en la variable *resultado* la suma o la resta según el caso,

- 3) Para mostrar la solución del resultado en la aplicación se hace un *set* del resultado en el *editSolucion* de la interfaz de la aplicación.
- 4) Se actualizan los nuevos valores del *guidata*.

Para programar una interfaz en *GUIDE* se deben almacenar las variables en algún lugar para que otros *callbacks* de la misma interfaz tengan acceso a esa información, por eso se hace uso de la estructura *handles* para el cual todos los *callback* dentro de la misma GUI tienen acceso, porque cada callback obtiene como tercer argumento una copia idéntica al *guidata* de manera automática. Todos los valores de los elementos (color, valor, posición, string...) y los valores de las variables transitorias del programa se almacenan en dicha estructura, y se accede a todos ellos mediante el mismo identificador [21].

Estas son las dos sentencias que hay que conocer bien para el buen uso del *GUIhandles*:

- La primera:

```
handles.output = hObject;
```

Con ella se almacena el identificador del GUI para que pueda ser usado posteriormente por la función de salida (*output function*). Resultará muy útil si queremos devolver el identificador del GUI a la línea de órdenes de MATLAB.

- La segunda:

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

Con esta sentencia la función *guidata* guarda las variables y propiedades de los elementos en la estructura de datos de la aplicación (estructura *handles*), por lo tanto, como regla general, en cada subrutina se escribe en la última línea. Nos garantiza que cualquier cambio o asignación de propiedades o variables quede almacenado.

Por ejemplo, si dentro de una subrutina una operación dio como resultado una variable denominada *resultado* se debe guardar de la siguiente manera:

```
handles.resultado = resultado;
guidata(hObject,handles);
```

La igualación crea la variable *resultado* a la estructura de datos de la aplicación apuntada por *handles* y la función *guidata* guarda el valor en la estructura *handles*.

Como se ha podido ver en el *Callback* de “*pushbuttonCalcular*”, para obtener o fijar el valor de un elemento del *GUIData* se utiliza *get* y *set*. Recordamos:

```
get(handles.editOperando1,'String'));  
set(handles.editSolucion,'String',resultado);
```

5.5. Código de la aplicación

5.5.1. Abrir conexión con analizador y configurar sus parámetros más relevantes

Cuando se estuvo comprobando la correcta conexión con “Instrument Control Toolbox” se descubrió que las instrucciones necesarias para abrir una comunicación socket TCP/IP con el *Master field* son las siguientes:

```
Anritsu = instrfind('Type', 'tcpip', 'RemoteHost', '192.168.43.53',  
'RemotePort', 9001, 'Tag', '');  
%Crear el objeto TCP/IP si no existe. En caso contrario utilizar el  
%objeto encontrado  
if isempty(Anritsu)  
    Anritsu = tcpip('192.168.43.53', 9001);  
else  
    fclose(Anritsu);  
    Anritsu = Anritsu(1);  
end
```

Si observamos el código vemos que se especifica la dirección IP asignada al Master Field y el puerto que se debe utilizar (9001).

El primer punto de control de la aplicación será introducir los parámetros relevantes del analizador que se han seleccionado, con *edits* para las cifras y *pop-up menus* (menús desplegables) para las unidades de medida. Esos parámetros son frecuencia central, SPAN, número de puntos y ancho de banda de integración. También se han añadido *radiobuttons* para activar o no preferencias de configuración, concretamente el preamplificador y el barrido continuo. Junto a ese panel de configuraciones se ubica el panel de acciones que para esta versión de la aplicación sólo tiene el *pushbutton* “Conect” que abre la configuración y gracias al *guidata* la deja abierta con el fin de usarla para funciones que se van a añadir posteriormente.

En la figura 28 localizada a continuación, puede verse el aspecto de esta primera versión:

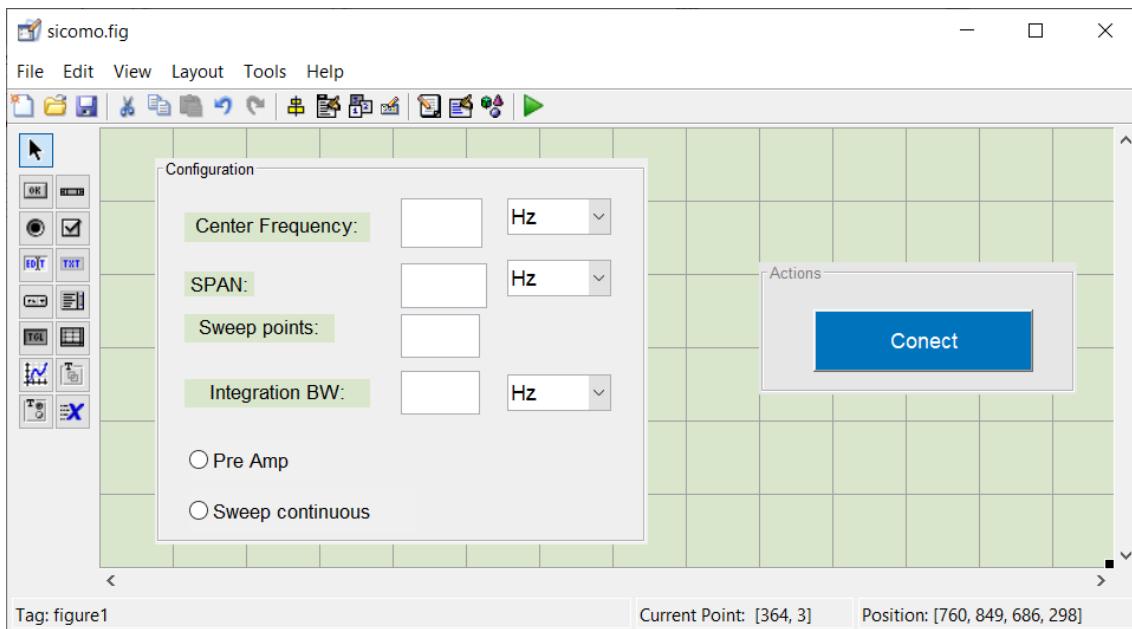


Figura 28. Primera versión de la aplicación

Como se ha dicho una vez el usuario selecciona y completa los campos de la configuración, puede pulsar el botón conectar, cuya función *Callback* simplificada para facilitar su entendimiento es la siguiente:

```
function Conectar_Callback(hObject, eventdata, handles)
%% Instrument Connection
% Find a tcpip object.
Anritsu = instrfind('Type', 'tcpip', 'RemoteHost', '192.168.43.53',
'RemotePort', 9001, 'Tag', '');

% Create the tcpip object if it does not exist
% otherwise use the object that was found.
if isempty(Anritsu)
    Anritsu = tcpip('192.168.43.53', 9001);
else
    fclose(Anritsu);
    Anritsu = Anritsu(1);
end
% Connect to instrument object, obj1.
handles.Anritsu = Anritsu;
fopen(Anritsu);

%% Configuración y control del instrumento
% Se configura SPAN:
fprintf(Anritsu, [:FREQuency:SPAN
',get(handles.SPAN_edit_text,'String'),',handles.unidades]);
% Frecuencia central:
fprintf(Anritsu, [:FREQuency:CENTER
',get(handles.Center_Frequency_edit_text,'String'),'
',handles.unidades2]);
% Se activa o desactiva preamplificador según elección de usuario:
fprintf(Anritsu, [:POWER:RF:GAIN:STATE',',handles.preamp]);
% Ancho de banda de integración:
fprintf(Anritsu, [:CHPower:BWIDth:INTegration
',get(handles.Integration_BW,'String'),',', handles.unidades3]);
% Se activa o desactiva barrido continuo según elección de usuario:
```

```
fprintf(Anritsu, ['INIT:CONT', ' ',handles.sweep]);
fprintf(Anritsu, [:DISPLAY:POINTcount ,
get(handles.Sweep_point_edit_text,'String')]);
guidata(hObject,handles);
```

Primero se abre la conexión tipo socket TCP como se ha explicado anteriormente y después haciendo uso de la función “fprintf()” —que sería la función “write()” en este tipo de socket y recibe como parámetros el nombre del socket (Anritsu) y el comando SCPI que se quiere transmitir al Anritsu— se envían los comandos SCPI necesarios para configurar los parámetros que ha introducido y seleccionado el usuario en el panel de configuración de la GUI. En este *Callback* también se puede destacar el uso de handles y *get* (para acceder a variables del GUIhandles) y de guidata (para guardar nuevos posibles valores necesarios para la aplicación en el GUIhandles) tan recurridas en GUIDE y que ya se han expuesto anteriormente.

Mas tarde al principio de la rutina “Conectar_Callback” se incluyeron las siguientes líneas de código que comprueban si el usuario ha dejado en blanco alguno de los campos de configuración a llenar:

```
%Checking if the user has filled in all the fields
if isempty (get(handles.Center_Frequency_edit_text,'String')) ||
isempty(get(handles.SPAN_edit_text,'String')) ||
isempty(handles.unidades) ||
isempty(get(handles.Sweep_point_edit_text,'String')) ||
isempty(get(handles.Sweep_point_edit_text,'String')) ||
isempty(get(handles.Integration_BW,'String'))
    errordlg('Fill in the blanks','Warning')
```

En caso afirmativo abre una ventana de diálogo con un mensaje de error que informa de la necesidad de completar todos los campos para poder abrir la conexión con el dispositivo:

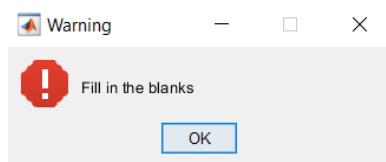


Figura 29. Mensaje de error “Fill in the blanks”

5.5.2. Toma de medidas manual y botón para guardar campaña

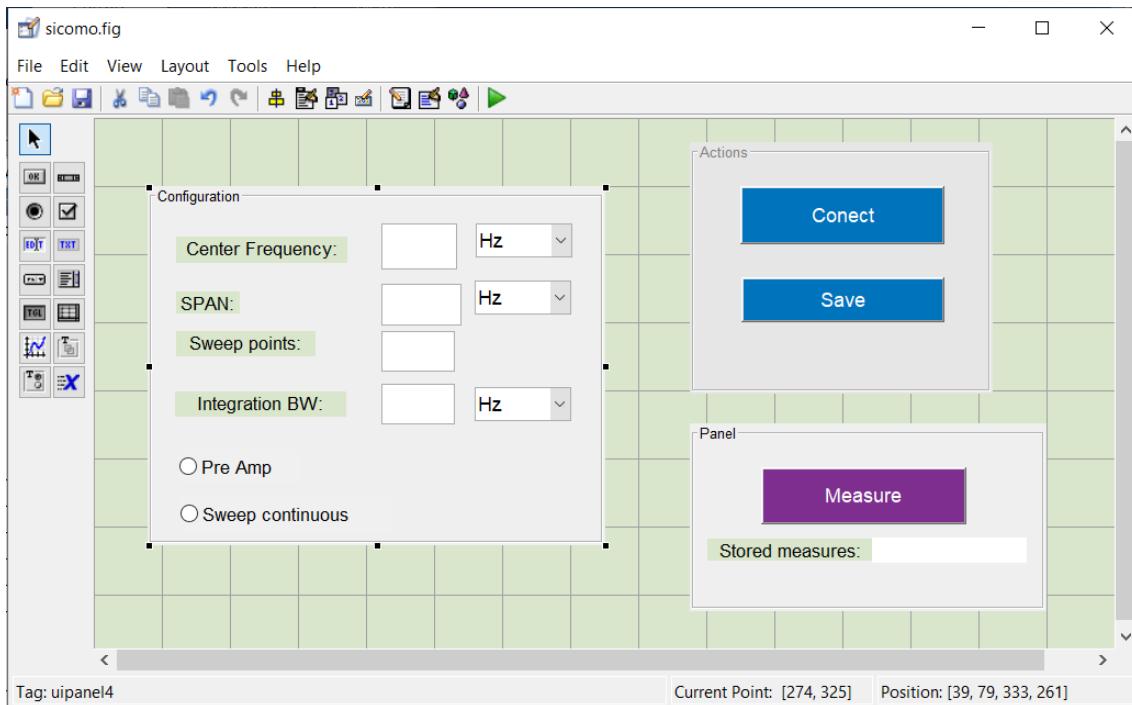


Figura 30. Segunda versión de la aplicación

Para esta segunda versión de la aplicación ya tenemos el botón “*Measure*” que permite tomar medidas de potencia asociada a sus datos GPS una a una cada vez que se hace click, y con un botón “*Save*” que guarda esas medidas en un archivo “.mat.” La función *callback* que se esconde tras el botón “*Measure*” es la siguiente:

```
function Measure_Callback(hObject, eventdata, handles)
global medidas i gpstotal chpower chpower2;
i = i + 1;
Anritsu=handles.Anritsu;
fopen(Anritsu);
%Este comando devuelve la potencia total del canal en dbm:
fprintf(Anritsu,:FETCh:CHPower:CHPower?');
chpower{i,} = fscanf(Anritsu);
%Devuelve marca de tiempo, latitud, longitud, altura, nºsatélites gps:
fprintf(Anritsu,:FETCh:GPS:FULL?');
gps = fscanf(Anritsu);
%Lineas para dar el formato deseado a los datos GPS:
gps2=split(gps,[ " ",",","] );
gps3 = gps2.';
for j=1:7
gpstotal{i,j} = gps3{1,j};
end
%Concatenar la potencia y datos GPS y lo guardamos en una celda:
medidas = horzcat(chpower, gpstotal);
%Actualiza Stored measures (Número de medidas que se han tomado)
set(handles.Stored_measures,'String',i);
fclose(Anritsu);
guidata(hObject,handles);
```

- Las variables que se declaran globales son accesibles y modificables desde cualquier *callback*, aunque otra alternativa como ya sabemos hubiera sido anexarlas a la estructura *handles*.
- La variable “*i*” se incrementa en cada medida, y tiene dos funciones:
 - o Indicar al usuario cuántas medidas se han tomado.
 - o Indicar al código que columna de la celda de medidas toca rellenar.
- Usamos la función “*fprintf*” con el comando SCPI apropiado de la guía del Anritsu para consultar la potencia total del canal, y con la función “*fscanf*” —que sería la función “*read()*” en este tipo de socket— guardamos en la variable “*chpower*” el dato de respuesta que el analizador ha vertido en el socket por el puerto 9001.
- La metodología del guión anterior se aplica para consultar y guardar los datos completos GPS que incluyen si la conexión es buena (GOOD FIX o NO FIX), fecha, hora, latitud, longitud, atura, y número de satélites GPS de los que se recibe cobertura. Todos esos datos son devueltos en un *string* separado por comas, por lo que es necesario convertirlo a una celda 1x7 en el que cada columna se corresponde con uno de los campos.
- En cada iteración se concatena horizontalmente la celda “*chpower*” con la celda “*gpstotal*” y se guarda en la celda “*medidas*”.
- Finalmente, con un *set* se establece en el *editable text* del panel de medidas el valor de *i*, que informa al de usuario cuantas medidas se han tomado.

Vista la función para tomar medidas, es momento de mostrar la función *callback* simplificada de “Save”.

```
function Save_Callback(hObject, eventdata, handles)
prompt = 'Name campaing: ';
campaing_name = input(prompt,'s');
if isempty(campaing_name)
    campaing_name = 'no_name';
end
save(campaing_name,'chpower','gpstotal','medidas');
fclose(Anritsu);
msgbox('Closed connection and saved campaign','OK')
guidata(hObject,handles);
```

- Cuando termina la campaña y se quieren guardar las medidas, pulsar “Save” hará que se le muestre al usuario por el *prompt* de MATLAB el texto “Name campaing:”, entonces el usuario deberá escribir el nombre que desee y pulsar Intro. Si no introduce ningún nombre y se pulsa Intro, la campaña se llamará “no_name”.

- Además, al pulsar Intro aparecerá este cuadro de diálogo:

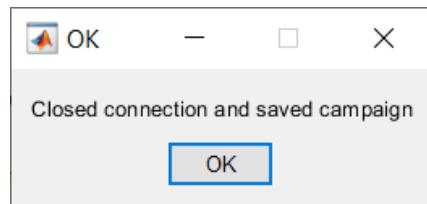


Figura 31. Cuadro de diálogo que se muestra al fin de la campaña

5.5.3. Solución para la excepción de falta de cobertura GPS

Se pudo observar en las pruebas de medición que si en algún momento no se recibía conexión GPS correcta se interrumpía la ejecución del código por desbordamiento en el bucle que guarda los valores en la variable “gpstotal”, porque sin conexión se obtiene un único campo que indica ‘NO FIX’. Aunque fuera poco probable en campo abierto, podría ocurrir que, aunque por un periodo de tiempo corto o en una porción de terreno muy concreta no se recibiera la conexión esperada, interrumpiéndose la campaña de medidas. Para solventar el problema se incluyó esta variación en los callbacks de “Measure” y “Start drive test”:

```
C = 'NO FIX';
C = [C newline ''];
tf = strcmp(C,gps);
if tf == 1
    for j=1:7
        gpstotal{i,j} = 'NF';
    end
else
gps2=split(gps,[ " ",",",""]);
gps3 = gps2.';
for j=1:7
gpstotal{i,j} = gps3{1,j};
end
end
```

Con la función “strcmp” se comprueba si los datos GPS que devuelve el analizador son los correspondientes a que no hay cobertura (“NO FIX”) con la ayuda de la función de comparación “strcmp()”. Si hay coincidencia se rellena las siete columnas de “gpstotal” con los caracteres “NF” para que no se dé el desbordamiento antes citado; en caso contrario se rellena normalmente.

5.5.4. Toma de medidas automáticas para *Dirve testing*

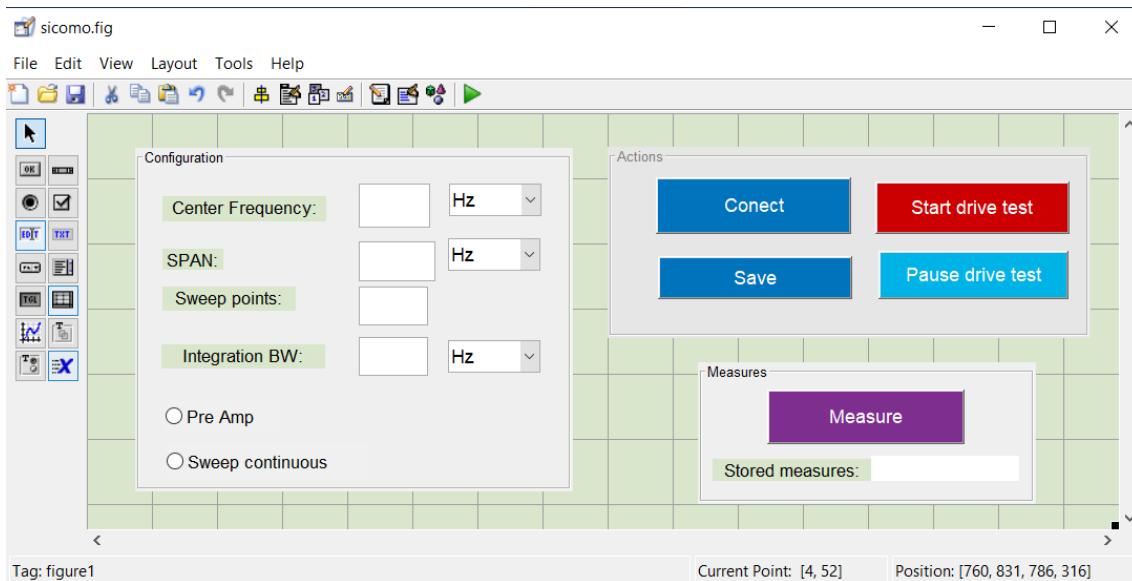


Figura 32. Tercera versión de la aplicación

En esta tercera versión se buscaba que el programa ya fuera capaz de tomar las medidas de forma automática y periódica para lo que se han añadieron los botones “Start drive test” y “Pause drive test”. La rutina que se ejecuta al presionar el botón “Start drive test” es la misma que la del botón “Measure”, pero en esta ocasión encapsulada en un bucle que hace posible la medición periódica ininterrumpida:

```
function Start_drive_test_Callback(hObject, eventdata, handles)
global medidas i chpower gpstotal chpower2 stop_drive_test;
stop_drive_test = 0;
while (stop_drive_test==0)
%-----
%Misma rutina para tomar medida de la función "Measure"
%-----
pause(0.5);
end
guidata(hObject,handles);
```

- Mientras que la variable “stop_drive_test” valga “0” se seguirá ejecutando la rutina de tomar una nueva medida. “stop_drive_test” deja de valer “0” cuando se clica en el botón “pause_drive_test” cuya única labor es la de poner esta variable a “1”. Una vez pausada bastará con volver a clicar en “start_drive_testing” para continuar con la campaña.
- Con la función “pause()” al final del bucle se busca detener la ejecución del código para que se tome una nueva medida cada medio segundo aproximadamente.

5.5.5. Gráfica de la potencia en tiempo real

Aquí tenemos la última versión de nuestra aplicación ejecutándose correctamente:

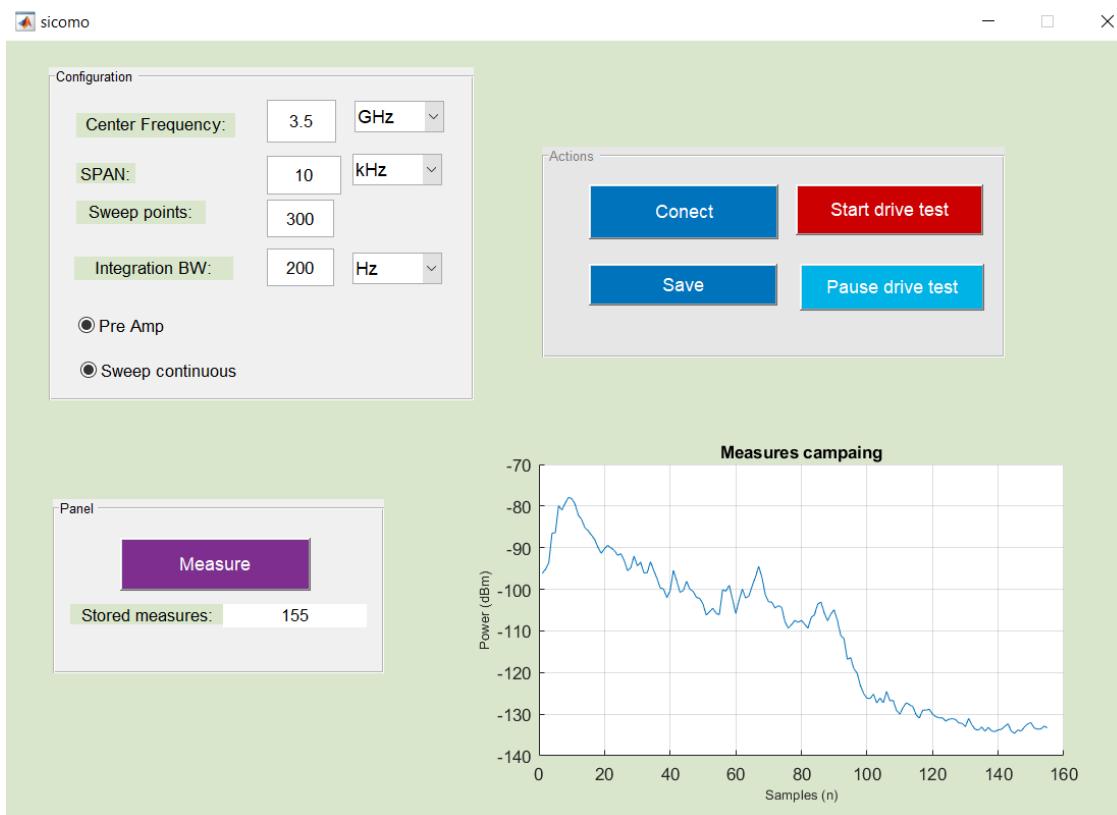


Figura 33. Última versión de la aplicación

Para comprobar de un vistazo que los valores de potencia que va tomando la herramienta mantienen una cierta coherencia, se estimó oportuno añadir una gráfica que vaya reflejando los valores de potencia que se van obteniendo en función del tiempo. Para graficar los valores de potencia en tiempo real fue necesario incluir a los callbacks de “Measure” y “Start_drive_test” las siguientes líneas:

```
cla(handles.axes1,'reset');
hold on;
title(handles.axes1, plotTitle, 'FontSize', 10);
xlabel(handles.axes1, xlabel, 'FontSize', 8);
ylabel(handles.axes1, ylabel, 'FontSize', 8);
grid(plotGrid);
plot(handles.axes1, 1:1:length(chpower), chpower);
hold off;
```

- La función “cla()” vacía la gráfica en cada iteración.
- Después se vuelve a dibujar de nuevo la gráfica con los valores de la celda “chpower” que contiene todos los valores tomados hasta el momento de potencia de canal.

5.5.6. Generación de un .exe

Es posible hacer uso del compilador de MATLAB para convertir nuestras GUIs en archivos .exe que pueden ejecutarse directamente sin la necesidad de abrir MATLAB. Para ello sólo se debe ejecutar el siguiente código desde la ventana de comandos, teniendo en cuenta que mi GUI se denomina “sicomo”.

```
mcc -m sicomo.m sicomo.fig
```

El compilador entonces creará los siguientes cuatro archivos en el “current folder”:

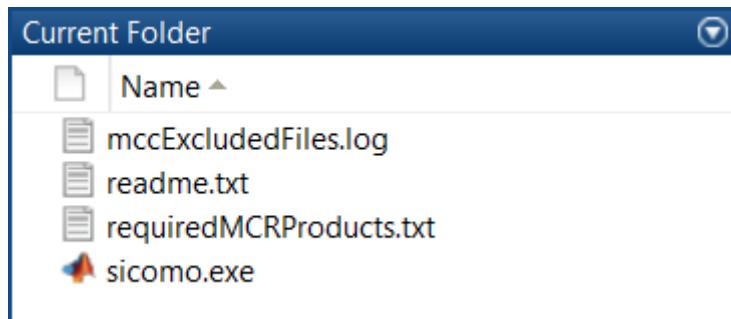


Figura 34. Archivos creados por compilador MATLAB

El archivo “.exe”, ultimo de la lista, es el ejecutable de la aplicación desarrollada desde MATLAB GUIDE. Con tan sólo un doble clic desde el explorador de archivos se estará ejecutando.

5.6. Formato en el que se guarda la campaña

Cuando se concluye la campaña de medidas y se salva como se ha expuesto anteriormente en el punto 5.5.2, se genera un fichero “.mat” con el nombre introducido por el usuario; éste queda guardado en la carpeta actual (*current folder*) que coincide con el directorio de memoria desde donde se ejecuta la aplicación. El fichero “.mat” contendrá una celda llamada medidas formada por ocho columnas cuyo aspecto al abrirla desde Matlab se muestra en la siguiente captura:

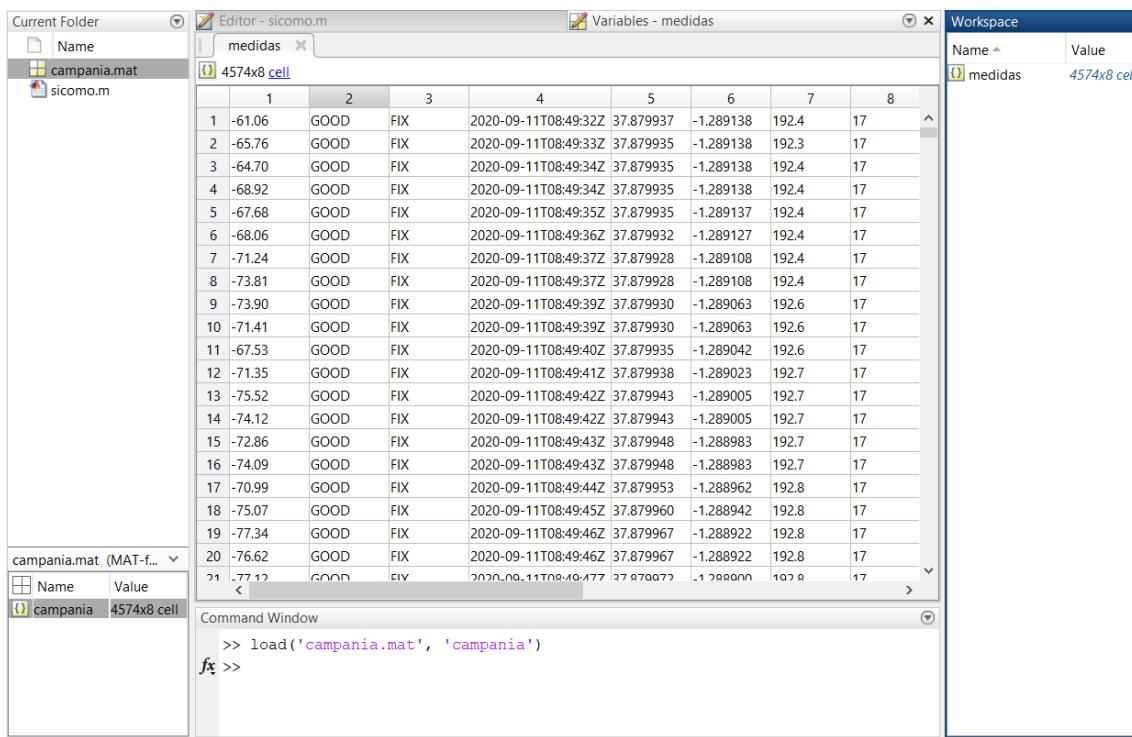


Figura 35. Aspecto del fichero “.mat” que devuelve la aplicación

- Columna 1: Potencia
- Columnas 2 y 3: Si estas dos columnas muestran “GOOD FIX”, al momento de tomar la medida se tenía buena conexión GPS, por el contrario, si la conexión no era buena mostrarán el mensaje “NO FIX”.
- Columna 4: Fecha y hora al momento de tomar la muestra correspondiente a esa fila.
- Columna 5: Latitud al momento de tomar la muestra.
- Columna 6: Longitud al momento de tomar la muestra
- Columna 7: Altitud con respecto al nivel del mar al momento de tomar la muestra.
- Columna 8: Número de satélites GPS de los que se valía el analizador para triangular las coordenadas GPS al momento de tomar la muestra.

5.7. Manual de usuario de la aplicación

Usar la GUI para tomar medidas de potencia del radiocanal de forma automática es muy sencillo, siga los siguientes pasos:

1. Instale Matlab R2017B.
2. Conecte el Analizador de espectro Anritsu MS209A y el ordenador desde el que se esté corriendo la aplicación a la misma red. Puede ser a la de un *router* (por *WIFI* o cable *ethernet* directo) o a la de un terminal móvil haciendo la función de *hotspot*.

3. En lugar de ello se podría conectar el ordenador directamente al Anritsu utilizando un cable ethernet cruzado y configurando desde Windows un área de red local, pero siempre acaba siendo lento y problemático.

Tras la conexión del Anritsu a la red, puede verse en su menú de *WIFI* cómo se le ha asignado una IP de forma automática por medio del protocolo DHCP.

Abra Matlab, localice los archivos de la aplicación en su “Current Folder” y en “sicomo.m” diríjase a la rutina de código que se ejecuta al pulsar el botón “Conect”, o lo que es lo mismo, a la función *callback* de ese botón. Una vez ahí, localice las líneas de código relacionadas con el establecimiento de la conexión TCP/IP que se pueden ver abajo. Se deben sustituir las dos IPs que aparecen subrayadas (192.168.43.53 en mi caso) por las IPs que se le han asignado al Anritsu de forma automática. (paso 2).

```
% Instrument Connection. Find a tcpip object
Anritsu = instrfind('Type', 'tcpip', 'RemoteHost', '192.168.43.53',
'RemotePort', 9001, 'Tag', '');
% Create the tcpip object if it does not exist
% otherwise use the object that was found.
if isempty(Anritsu)
    Anritsu = tcpip('192.168.43.53', 9001);
else
    fclose(Anritsu);
    Anritsu = Anritsu(1);
end
```

Si se ha optado por una conexión con cable ethernet cruzado (ver paso 1) se tendrán que sustituir las IPs por la que se le ha asignado manualmente al Anritsu.

4. Ejecute la GUI “sicomo.m” clicando en el botón “Run” de MATLAB. Entonces se mostrará la interfaz gráfica, introduzca los parámetros solicitados en función de la configuración sus requerimientos
5. Pulse el botón “Conect”.
6. Pulse “Start drive test” para iniciar la toma de medidas automática.
7. Si quiere pausar su campaña pulse “Pause drive test”. Para reanudarla pulse “Start drive test” y seguirá añadiendo tomas a la campaña.
8. Cuando haya terminado su campaña clique en “Pause drive test” y a continuación en “Save drive test”. Dese cuenta que la ventana de comandos de Matlab le solicitará que introduzca un nombre para su recién tomada campaña. Introdúzcalo y asegúrese de presionar *Intro* en su teclado. Entonces su campaña será guardada en su “Current Matlab folder” en la forma de un “.mat”.

9. Si quiere iniciar otra campaña pulse “Conect” de nuevo, aunque mantenga los mismos parámetros de configuración. Dicho de otra forma, si quiere iniciar otra campaña, regrese al paso 6.

Si lo que quiere es usar la GUI de forma manual, cuando se encuentre en el paso 6, en lugar de clicar en “Pause drive test” clique en “Measure” cada vez que desee añadir una toma a la campaña. Cuando termine de añadir tomas continúe al paso 8.

Capítulo 6. Aplicación del sistema y resultados

6.1. Introducción

Este capítulo da testimonio de las campañas de medidas que se realizaron con el fin de aplicar el sistema de medidas integrado y la herramienta desarrollada. Para ello también se muestran los resultados que verifican la correcta realización de dichas campañas y, los scripts diseñados y utilizados para el procesado del fichero de medidas que proporciona la GUI de MATLAB.

6.2. Montaje de los equipos

En todas las campañas efectuadas, el montaje tenía las mismas características. En transmisión, el analizador de redes genera un tono de 3dBm a la frecuencia de 3.5 GHz que es amplificado y transmitido con polarización vertical por una antena situada en un mástil a una altura de 4.2m.

En las figuras 36 y 37 se observa cómo se ingenió el montaje de la estación transmisora en el área rural.



Figura 36. Montaje estación transmisora



Figura 37. Montaje estación transmisora

El sistema receptor se monta en un vehículo con la antena receptora a una altura de 1.7m.



Figura 38. Montaje estación receptora: Vista vehículo con antena receptora



Figura 39. Montaje estación receptora: Interior de vehículo con ordenador y ANRITSU

Entonces la estación receptora es transportada por el vehículo mientras va recogiendo medidas en diferentes localizaciones de forma periódica, de ahí el porqué de que al tipo de herramienta desarrollada se le llame “herramienta de *drive testing*”

6.3. Campañas de medidas

6.3.1. Antiguo cuartel de Antiguones

La primera prueba a la que se sometió la aplicación fue a la de realizar una campaña de medidas en las proximidades del antiguo cuartel de Antiguones, actual sede de la Escuela Técnica Superior de Ingeniería de Telecomunicación (ETSIT) de la Universidad Politécnica de Cartagena (UPCT).

Se siguió la ruta delimitada por la carretera que rodea al edificio. Ese recorrido puede verse delimitado por puntos en la figura 41 del apartado Resultados (6.4), cada uno de los cuales se corresponden con cada una de las 146 medidas tomadas durante la campaña.

6.3.2. Cultivo de cítricos en Sierra Espuña

En el sector agrícola, las plantaciones de cítricos tienen una especial relevancia. A nivel mundial se estiman 6.9 millones de hectáreas de plantaciones de cítricos, ocupando China el primer lugar con 1.3 millones de hectáreas. En cuanto a España, la Comunidad Valenciana ocupa el primer lugar con unas 175.000 hectáreas, en segundo lugar, Andalucía con unas 80.000, en tercer lugar, la Región de Murcia con unas 30.000, seguidas del resto de comunidades autónomas que entre todas suman unas 5000 hectáreas. En cuanto a la producción mundial, Brasil y Estados Unidos

son los líderes con el 37 por ciento de la cosecha de cítricos del mundo, con 14,8 millones de toneladas y 14,3 millones respectivamente. España se encuentra entre los primeros 10 países productores en el mundo, siendo el principal exportador con unos 3 millones de toneladas. Con estos números, el aumento significativo de sistemas de radiocomunicaciones en estas plantaciones para estar al nivel de lo que supone la agricultura 4.0, requiere cada vez más de una planificación eficiente de estos sistemas en este tipo de plantaciones.

Por todo ello, se vio interesante aplicar el sistema de medidas que se ha moldeado en un entorno de este tipo, concretamente en una plantación de limoneros de la empresa FRUCA que se sitúa en el Valle de Carrascoy, en la Región de Murcia.

Las plantaciones de cítricos (limoneros, naranjos, mandarinos y pomelos) siguen un marco de plantación que viene definido por la distancia entre filas y la distancia entre los árboles en una misma fila (ver Fig. 40). Este marco de plantación suele realizarse de tal forma que la distancia entre filas es mayor que la distancia entre árboles, dando lugar habitualmente a que las ramas de árboles consecutivos en una misma fila se toquen, dejando un camino (calle) entre filas que es utilizado principalmente para fumigar, podar y recolectar el fruto, y en nuestro caso serán recorridas por el vehículo que porta nuestra estación receptora. El marco de plantación es 7x5m, es decir, 7m. entre filas y 5 m entre limoneros de una misma fila.

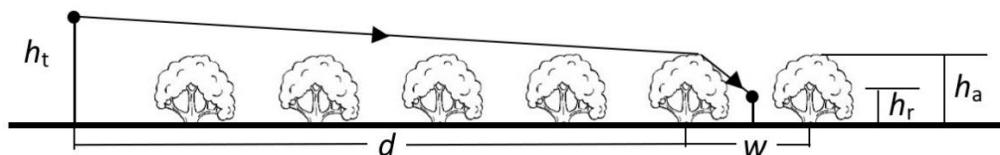


Figura 40. Corte vertical con árboles equiespaciados

Para caracterizar bien el entorno era preciso tomar las medidas de potencia del radio canal sin fruta en el árbol y con fruta en el árbol. Para la campaña sin limón se llevaron a cabo 19 recorridos, cada uno de ellos en un camino (calle) de longitud de 200m. El número de muestras por recorrido fue de 109 en media, siendo el número de muestras totales de 2086. Para la campaña con limón se llevaron a cabo 23 recorridos. El número de muestras por recorrido fue de 114 en media, siendo el número de muestras totales de 2638. En el siguiente punto (6.4) se muestran la localización vía satélite de cada una de las medidas, representando la ruta seguida por el analizador. Para tener en cuenta el efecto que tiene todo el equipamiento sobre la medida se llevó a cabo un proceso de calibración con un conjunto de medidas en caminos de visión directa entre transmisor y receptor.

6.4. Resultados



Figura 41. Ruta campaña de medidas ETSIT UPCT

Ruta sin limón en el árbol



Figura 42. Plantación de limoneros, situación del transmisor y recorridos con las medidas realizadas sin limón

Ruta con limón en el árbol



Figura 43. Plantación de limoneros, situación del transmisor y recorridos con las medidas realizadas con limón

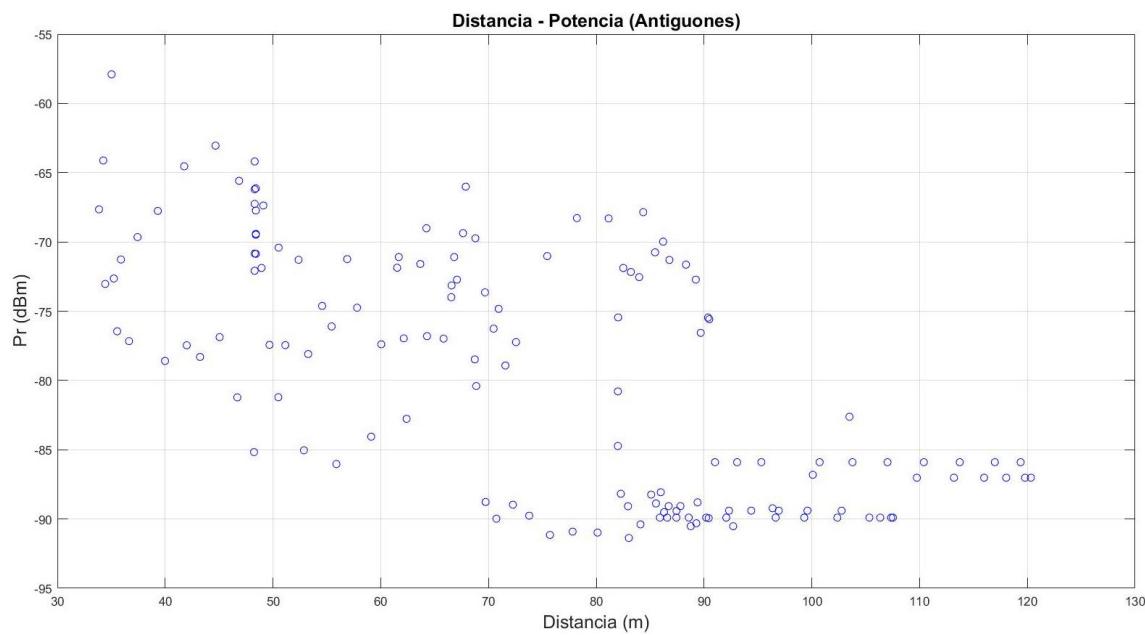


Figura 44. Distancia (m) – Potencia (dBm) en Antiguones

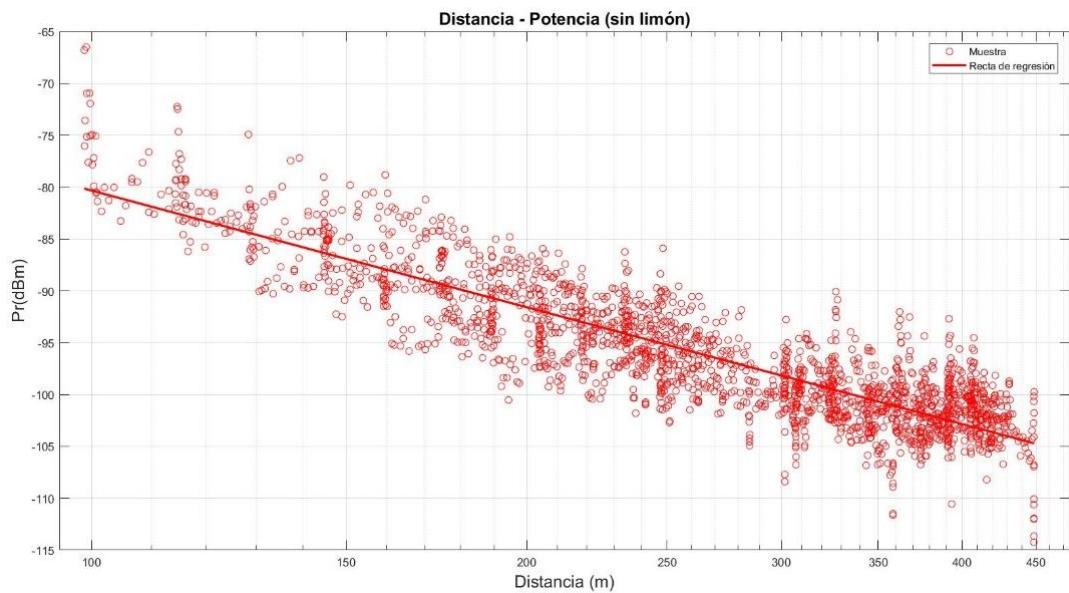


Figura 45. Gráfica Distancia (m) – Potencia recibida (dBm) sin limón y recta de regresión

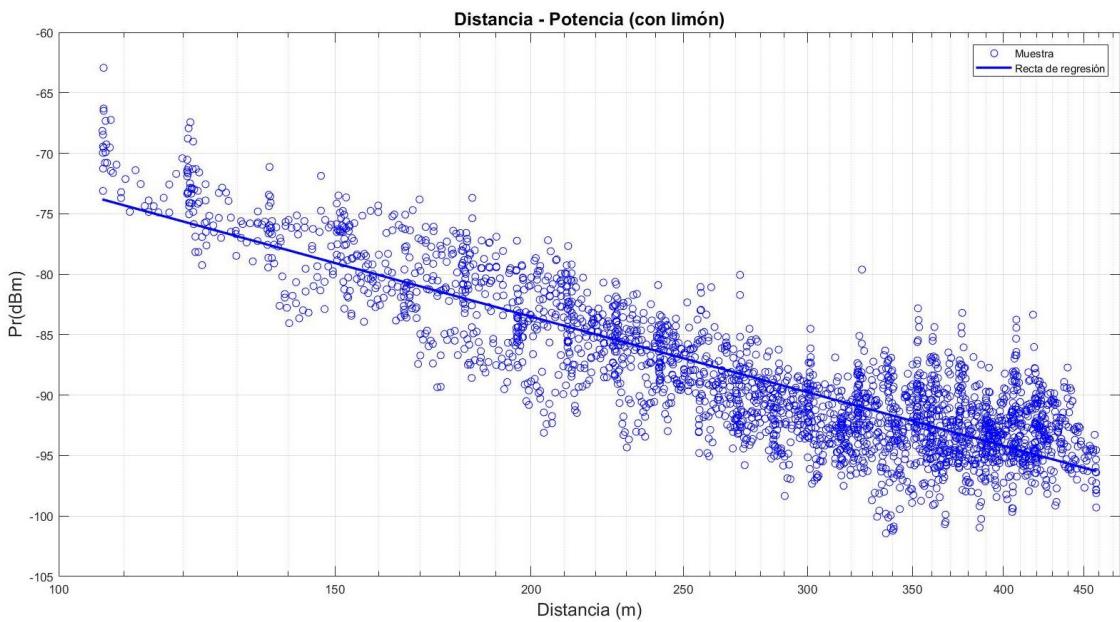


Figura 46. Gráfica Distancia (m) – Potencia recibida (dBm) con limón y recta de regresión

Como se puede ver en las figuras 45 y 46, sólo se calcula la recta de regresión para las campañas en entorno agrícola, pues recordamos que las medidas en los alrededores de la universidad consistían en una prueba sin mayor importancia. Los parámetros de las rectas de regresión de la nube de puntos correspondientes a las medidas en cultivo de cítricos se muestran en la tabla mostrada seguidamente:

	Pendiente	Desviación típica
Sin fruta	-3.55	3.16
Con fruta	-3.74	3.18

Tabla 4. Pendiente y desviación típica rectas de regresión de las campañas en cultivo de cítricos

Los valores expuestos en la tabla 4 junto al resto de resultados fueron enviados al Departamento de Tecnologías de la Información y las Comunicaciones de la UPCT para que fueran interpretados, pareciéndoles de gran interés al presentar las pendientes un valor cercano al que predicen modelos teóricos de propagación en entorno urbano como el de Walfish-Bertoni [22].

6.5. Scripts utilizados para el procesado de las medidas

El tipo de programa más simple de MATLAB se conoce como *script*. Un script es un archivo que contiene varias líneas secuenciales de comandos y llamadas a funciones de MATLAB [20].

Para poder trazar las gráficas de distancia y potencia que se muestran el apartado 6.4, se tienen que realizar una serie de pasos que se describen a continuación.

6.5.1. Script para convertir de coordenadas geográficas decimales a UTM

Como se indicó en el punto 4.3.3 el analizador de espectros con el que trabajamos devuelve los datos de latitud y longitud en el formato de coordenadas decimales, sin embargo, este formato no es el más adecuado para calcular la distancia entre dos puntos, propósito que es necesario para saber la distancia entre cada una de las medidas tomadas y el emisor que siempre está fijo en la misma localización. Así es como surge la necesidad de tener que convertir al formato de coordenadas geográficas UTM.

El sistema de coordenadas UTM (Universal Transverse Mercator) se fundamenta en la proyección cartográfica transversal de Mercator, que se basa en la de Mercator normal con la diferencia de que se es secante a un meridiano en vez de ser tangente al Ecuador [15].

En lugar de expresar las coordenadas geográficas con latitud y longitud, como lo hace el sistema tradicional, el sistema UTM lo expresa en metros sobre el nivel del mar, siendo esta la base del elipsoide de referencia. Los valores de las coordenadas UTM (X e Y) son siempre positivos; los ejes cartesianos X e Y se establecen sobre el huso, siendo el eje X el ecuador y el eje Y el meridiano. En esta proyección, la Tierra queda dividida en 60 husos de 6° de longitud, numerándose cada huso con un dígito entre 1 y 60.

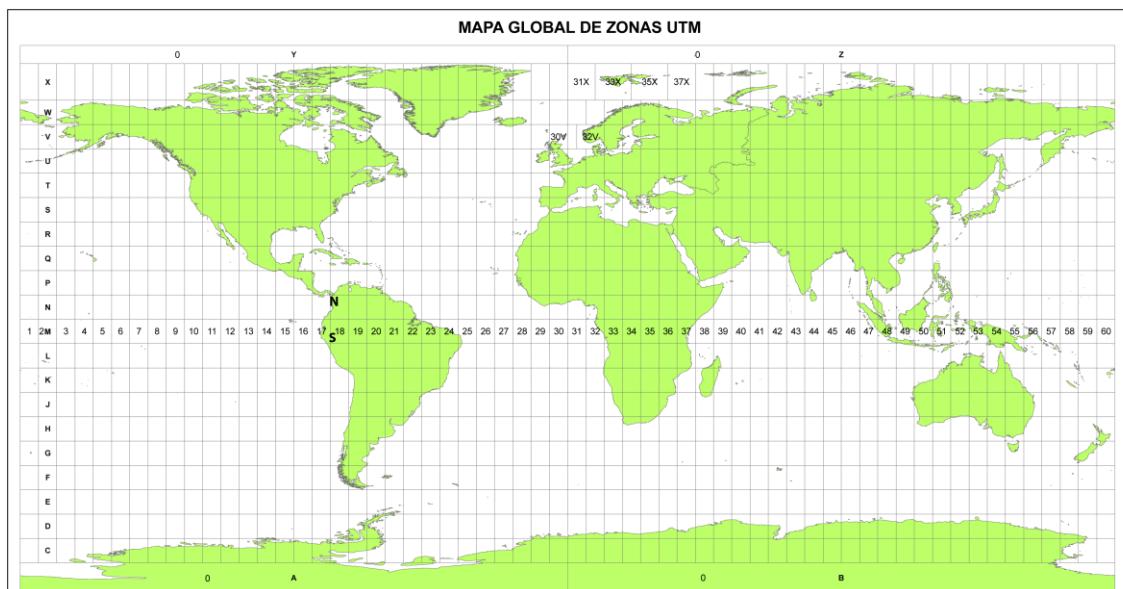


Figura 47. Mapa global de zonas UTM [23]

Por ejemplo, encontramos la Península Ibérica está situada en los husos 29, 30 y 31. La Tierra queda dividida en 20 bandas de 8° de latitud, que son denominadas con letras desde la C hasta

la X, excluyendo las letras “I” y “O”, para evitar entrar en conflicto con los números “1” y “0”. De modo que, una banda pertenecerá al hemisferio norte si su letra identificadora es igual o mayor que “N”, mientras que pertenecerá al hemisferio sur si su letra es menor que “N”. La zona UTM en la que se encuentra la ciudad de Cartagena corresponde a la 30S, como se puede comprobar en la figura 47.

A modo de ejemplo se muestran las coordenadas UTM del Cuartel de Antiguones, ubicación en la que ejecutó la campaña de medidas de prueba: 30S 678415 4163494, donde 30 indica la zona UTM, S la banda UTM, la primera ristra de números (678415) es la distancia en metros al Este y la segunda (4163494) la distancia en metros al Norte.

La función encargada de realizar la conversión puede encontrarse compartida por la comunidad en la web oficial de MATLAB. Se denomina “deg2utm” y tiene esta pinta:

```

function [x,y,utmzone] = deg2utm(Lat,Lon)
error(nargchk(2, 2, nargin)); %2 arguments required
n1=length(Lat);
n2=length(Lon);
if (n1~=n2)
    error('Lat and Lon vectors should have the same length');
end
% Memory pre-allocation
%
x=zeros(n1,1);
y=zeros(n1,1);
utmzone(n1,:)=['60 X'];
% Main Loop
%
for i=1:n1
    la=Lat(i);
    lo=Lon(i);
    sa = 6378137.000000 ; sb = 6356752.314245;

    %e = ( ( ( sa ^ 2 ) - ( sb ^ 2 ) ) ^ 0.5 ) / sa;
    e2 = ( ( ( sa ^ 2 ) - ( sb ^ 2 ) ) ^ 0.5 ) / sb;
    e2cuadrada = e2 ^ 2;
    c = ( sa ^ 2 ) / sb;
    %alpha = ( sa - sb ) / sa; %f
    %ablandamiento = 1 / alpha; % 1/f
    lat = la * ( pi / 180 );
    lon = lo * ( pi / 180 );
    Huso = fix( ( lo / 6 ) + 31 );
    S = ( ( Huso * 6 ) - 183 );
    deltaS = lon - ( S * ( pi / 180 ) );
    if (la<-72), Letra='C';
    elseif (la<-64), Letra='D';
    elseif (la<-56), Letra='E';
    elseif (la<-48), Letra='F';
    elseif (la<-40), Letra='G';
    elseif (la<-32), Letra='H';
    else
        Letra='A';
    end
    utmzone(i)=Letra;
end

```

```

elseif (la<-24), Letra='J';
elseif (la<-16), Letra='K';
elseif (la<-8), Letra='L';
elseif (la<0), Letra='M';
elseif (la<8), Letra='N';
elseif (la<16), Letra='P';
elseif (la<24), Letra='Q';
elseif (la<32), Letra='R';
elseif (la<40), Letra='S';
elseif (la<48), Letra='T';
elseif (la<56), Letra='U';
elseif (la<64), Letra='V';
elseif (la<72), Letra='W';
else Letra='X';
end
a = cos(lat) * sin(deltaS);
epsilon = 0.5 * log( ( 1 + a ) / ( 1 - a ) );
nu = atan( tan(lat) / cos(deltaS) ) - lat;
v = ( c / ( ( 1 + ( e2cuadrada * ( cos(lat) ) ^ 2 ) ) ) ^ 0.5
) * 0.9996;
ta = ( e2cuadrada / 2 ) * epsilon ^ 2 * ( cos(lat) ) ^ 2;
a1 = sin( 2 * lat );
a2 = a1 * ( cos(lat) ) ^ 2;
j2 = lat + ( a1 / 2 );
j4 = ( ( 3 * j2 ) + a2 ) / 4;
j6 = ( ( 5 * j4 ) + ( a2 * ( cos(lat) ) ^ 2 ) ) / 3;
alfa = ( 3 / 4 ) * e2cuadrada;
beta = ( 5 / 3 ) * alfa ^ 2;
gama = ( 35 / 27 ) * alfa ^ 3;
Bm = 0.9996 * c * ( lat - alfa * j2 + beta * j4 - gama * j6
);
xx = epsilon * v * ( 1 + ( ta / 3 ) ) + 500000;
yy = nu * v * ( 1 + ta ) + Bm;
if (yy<0)
    yy=9999999+yy;
end
x(i)=xx;
y(i)=yy;
utmzone(i,:)=sprintf('%02d %c',Huso,Letra);
end

```

El anterior código recibe un array de latitudes y otro de longitudes como parámetros de entrada, donde cada pareja latitud-longitud se corresponde con la posición geográfica donde cada medición ha sido realizada. La función devolverá dos arrays “X” e “Y”, que se corresponden a las mismas coordenadas del array de entrada, pero esta vez en su formato UTM.

6.5.2. Script para representar gráficas y calcular parámetros

Este script carga el fichero de medidas, convierte los datos de coordenadas GPS a coordenadas UTM mediante la implementación del script anterior, calcula la matriz de distancias entre el transmisor y cada uno de los puntos, genera una matriz de potencias recibidas, y después, por medio de las dos matrices representa la potencia de señal recibida por cada muestra en función

de la distancia sobre un eje de abscisas logarítmico. A continuación, calcula la recta de regresión lineal de la nube de puntos que constituyen cada una de las medidas para, acto seguido, representarla sobre esa misma nube de puntos. Podemos tratar de entender este procedimiento analizando directamente el script, que además se encuentra bien comentado:

```
%Carga el fichero de medidas:
load('nombre_fichero.mat');
posicion_tx = [37.880904,-1.289244]; %Ejemplo de posición transmisor

%Paso 1: extraer la matriz distancia que contiene cada una de las
distancias existentes entre emisor y receptor.
lat = str2double(campania(:,5));
long = str2double(campania(:,6));
%Conversión de coordenadas decimales a UTM
[x,y,utmzone1] = deg2utm(lat,long);
posicion_tx = [37.880904,-1.289244];
[z,t,zone2] = deg2utm (posicion_tx(1),posicion_tx(2));
z(1:length(x),:) = z;
t(1:length(x),:) = t;
%Por trigonometría se calcula la distancia entre cada medida y el tx:
dist = (sqrt((x-z).^2+(y-t).^2)).';
distancia=dist;
%La potencia de cada muestra define al vector potencias:
Pr_con=str2double((campania(:,1))).';

%Paso 2: calcular la recta de regresión de la nube de muestras:
N1=length(distancia);
Ldc_con=Pr_con;
%Método FI
Dlog=10*log10(distancia);
Pend_FI=polyfit(Dlog,Ldc_con,1);
LO_FI_con=Pend_FI(2)
np_FI_con=Pend_FI(1)
Lp_FI_con=LO_FI_con+np_FI_con*Dlog;
% Varianza FI
NMSE_con=(sum((Ldc_con-Lp_FI_con).^2))/N1;
std_FI_con=sqrt(NMSE_con);

%Paso 3: representar de cada una de las muestras:
figure(1)
semilogx(distancia,Pr_con,'bo')
xlim([90 500])
x=[100 200 300 400 500];
set(gca,'XTick',x)
grid on
hold on
title('Distancia - Potencia (con limón)')
xlabel('Distancia (m)')
ylabel('Pr(dBm)')

%Paso 4: Representar la recta de regresión de la nube de muestras:
semilogx(distancia,Lp_FI_con,'b','linewidth',2);
legend('Muestra', 'Recta de regresión');
```

6.6. Metodología para la representación de las rutas

Para obtener las imágenes vía satélite con la representación muy aproximada de la posición en que fueron tomadas cada una de las muestras se recurrió a Google Earth. Para conseguirlo, lo primero es exportar nuestro fichero de medidas “.mat” de la campaña desde MATLAB a un archivo de hoja de cálculo Ecel “.xml” por medio del siguiente comando:

```
xlswrite('nombre_campaña.mat', nombre que deseas)
```

Una vez en la hoja de cálculo hay que asegurarse de que Excel esté configurado para que “.” sea el separador decimal. Se debe añadir una cabecera a la hoja de cálculo, como en el siguiente ejemplo simplificado compuesto por 7 medidas:

Potencia	Latitud	Longitud	Altura
-81.3	37.880472	-1.286933	191
-80.34	37.880468	-1.286948	191
-78.77	37.880468	-1.286948	191
-78.1	37.880465	-1.28697	190.9
-80.52	37.88046	-1.28699	190.8
-82.11	37.880455	-1.287012	190.7

Tabla 5. Ejemplo simplificado medidas en hoja de cálculo previa importación a Google Earth

Después se guarda el documento con el formato CSV-UTF8 (delimitado por comas) (*.csv) para que pueda ser importado en Google Earth para formar un archivo “.kmz” dentro del mismo programa. El formato KMZ se utiliza para guardar datos geográficos dentro de un navegador terrestre [24], como es el caso. Para que no haya problemas al momento de importarlo se deberán seleccionar todos los campos de la tabla como tipo “cadena”.

Una vez importado Google Earth permite editar el aspecto de las localizaciones seleccionadas, como su tamaño o color.

Capítulo 7. Conclusiones

Como ya se ha mencionado a lo largo de esta redacción, el objetivo principal del proyecto era el desarrollo en MATLAB de una herramienta informática cuyas funcionalidades ayuden a la realización de campañas de medidas en un sistema basado en el analizador de espectros portátil MS2090A de Anritsu, objetivo que se ha completado con éxito. Se han efectuado campañas de medidas del canal de radiocomunicación en un entorno de cultivo de cítricos cuyos datos han sido procesados y cuyos resultados, que arrojaban información de interés de la propagación de la señal en este tipo de entornos, han sido facilitados al Departamento de Tecnologías de la Información y las Comunicaciones de la UPCT para que, desde ahí, lleven a cabo su correcta interpretación.

Junto al objetivo principal se han cumplido otros hitos no contemplados al arranque del proyecto, como el diseño de una interfaz amena que permita una cómoda interacción del usuario, siendo esta además capaz de graficar en tiempo real la potencia de cada muestra, lo que permite asegurar que la campaña se está desarrollando adecuadamente, evitando imprevistos indeseados.

A lo largo del proyecto se ha estudiado el estado del arte, se ha podido aprender acerca del entorno de MATLAB y su lenguaje de programación, y de cómo hacer una interfaz de usuario amigable con MATLAB GUIDE. Se ha adquirido el conocimiento necesario para establecer una conexión TCP/IP con un instrumento y de cómo comunicarse remotamente con él mediante el uso del lenguaje de programación SCPI, específico para ello. Se ha visto el conjunto de instrumentos que forman un sistema de medidas y la manera en que se monta y manipula para realizar campañas de medidas.

No obstante, existen mejoras que podrían investigarse en futuras líneas. Por ejemplo, la aplicación podría mostrar en tiempo real una gráfica con la potencia en función de la distancia. Otra mejora de la aplicación sería que, al guardar la campaña, se generara de forma automática un fichero que diera cuenta de todos los parámetros de configuración sobre los que estaba operando el analizador de espectros a lo largo de esa campaña, para así no tener que apuntarlos cada vez que se lleva a cabo una campaña de medidas.

Bibliografía

- [1] H. Mahmood Jawad, R. Nordin, S. Kamel Gharghan, A. Mahmood Jawad, and M. Ismail, (agosto de 2015), “Energy-Efficient Wireless Sensor Networks for Precision Agriculture: A Review”, Sensors, vol. 17, 1-45.
- [2] “Digital Farming: What does it really mean?” CEMA. Recuperado de:
<http://www.cema-agri.org/page/digital-farming-what-does-it-really-mean>.
- [3] Centro Mario Molina para Estudios Estratégicos sobre Energía y Medio Ambiente. (agosto de 2016). Programa de educación en cambio climático, Cuajimalpa, México, D.F.
- [4] Atkins, Peter; Jones, Loretta (2006). Principios de química: los caminos del descubrimiento. Ed. Médica Panamericana. ISBN 9789500600804.
- [5] Ellingson, Steven W. (2016). Radio Systems Engineering. Cambridge University Press. ISBN 9781316785164.
- [6] Servicio de Biblioteca y Archivos de la Unión Internacional de Telecomunicaciones (UIT). (2016). Radio Regulations.
- [7] Centro Andaluz de Estudios y Entrenamientos. (2018). Clasificación de las ondas según su propagación y alcances y propagación por bandas. Recuperado de:
<https://centroandaluz.net/blog-academia/clasificacion-ondas-segun-propagacion-alcances-propagacion-bandas/>
- [8] Rudolf F. Graf (1975). Dictionary of Electronics, USA.
- [9] Arellano Hernández, Job. (septiembre de 2019). Blockchain en la optimización de la comprobación del gasto en actividades de vigilancia del espectro radioeléctrico. Ciudad de México.
- [10] Juan Llácer, L, Molina García-Pardo, J. M. & Pascual García, J. (2010). Caracterización del Canal Móvil. Universidad Politécnica de Cartagena.
- [11] Despujol, Z.I. (2019). Propagación atmosférica. Reflexión, refracción y scattering [Archivo de video]. Recuperado de <http://www.upv.es/visor/media/e721de30-15d3-11e9-8f65-1183f370292d/c>
- [12] Rappaport, Theodore S. (2002). *Wireless Communications principles and practice*, Second Edition, Prentice Hall, ISBN 0130422320. New York.

- [13] C. A. Balanis, *Advanced Engineering Electromagnetics*, John Wiley & Sons, New York, May 1989.
- [14] Web oficial de Televés: <https://www.televes.com/es/g-192-medidor-de-campo-h30flex.html>
- [15] Noguera Gil, A. J. (2018). (Trabajo fin de estudios). Desarrollo e implementación de un sistema de medidas de coberturas radio basado en un analizador de espectros y un GPS.
- [16] W. Stallings, *Wireless Communications and Networks*, 2nd Ed., Prentice Hall, 2005.
- [17] Web oficial de Rohde & Schwarz. (2011). Recuperado de https://www.rohdeschwarz.com/es/producto/fpl1000-pagina-de-inicio-producto_63493-465280.html
- [18] Anritsu Company. (Sep. 2019). Field Master Pro™ MS2090A Spectrum Analyzer Programming Manual. Morgan Hill, USA.
- [19] Rappaport, Theodore S. (2002). *Wireless Communications principles and practice*, Second Edition, Prentice Hall, ISBN 0130422320. New York.
- [20] Web oficial de MATLAB. (1994-2021). Recuperado de: <https://es.mathworks.com/>
- [21] Barragán Guerrero, D. O. (agosto de 2007). Manual de Interfaz Gráfica de Usuario en Matlab, Guayaquil, Ecuador.
- [22] J. Walfisch, and H.L. Bertoni. (diciembre de 1988) "A Theoretical Model of UHF Propagation in Urban Environments", IEEE Transactions on Antennas and Propagation, Vol. 36, No. 12, pp. 1788-1796.
- [23] Franzpc. Mapa global de Zonas UTM. (septiembre de 2011). [Blog] Recuperado de: <https://acolita.com/mapa-global-mundi-todas-las-zonas-utm/>
- [24] Albornoz, F. "Extensión de archivos .KMZ ¿Qué son y cómo abrir este tipo de archivos?". (Blog) Recuperado de: <https://internetcasoapaso.com/archivos-kmz/>

Anexo. Código de MATLAB

```

function varargout = sicomo(varargin)
% SICOMO MATLAB code for sirecomo.fig
%   fi SICOMO, by itself, creates a new SICOMO or raises the
existing
%   singleton*.
%
%   H = SICOMO returns the handle to a new SICOMO or the handle to
the existing
%   singleton*.
%
%   SICOMO('CALLBACK', hObject, eventData, handles,...) calls the
local
%   function named CALLBACK in SICOMO.M with the given input
arguments.
%
%   SICOMO('Property','Value',...) creates a new SICOMO or raises
the
%   existing singleton*. Starting from the left, property value
pairs are
%   applied to the GUI before sicomo_OpeningFcn gets called. An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to sicomo_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 sicomo

% Last Modified by GUIDE v2.5 04-Mar-2020 16:06:34

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',         mfilename, ...
                   'gui_Singleton',    gui_Singleton, ...
                   'gui_OpeningFcn',   @sicomo_OpeningFcn, ...
                   'gui_OutputFcn',    @sicomo_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 sicomo is made visible.
function sicomo_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 sicomo (see VARARGIN)

% Choose default command line output for sicomo
handles.output = hObject;
handles.unidades = 'HZ';
handles.unidades2 = 'HZ';
handles.unidades3 = 'HZ';
handles.preamp = '0';
handles.sweep = '0';

% Update handles structure
guidata(hObject, handles);

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

% --- Outputs from this function are returned to the command line.
function varargout = sicomo_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 button press in Medir.
function Medir_Callback(hObject, eventdata, handles)
% hObject     handle to Medir (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
global medidas i gpstotal chpower chpower2;
i = i + 1;
Anritsu=handles.Anritsu;
fclose(Anritsu);
fopen(Anritsu);
%Este comando devuelve TOTAL CHPOWER en dbm
fprintf(Anritsu,:FETCh:CHPower:CHPower?');
chpower1 = fscanf(Anritsu);
chpower2{i,1} = chpower1;
chpower(i,1) = str2double(chpower1);

% % % Representar la potencia % % %
cla(handles.axes1,'reset');
plotTitle = 'Measures campaing'; % plot title
xLabel = 'Samples (n)'; % x-axis label
yLabel = 'Power (dBm)'; % y-axis label
plotGrid = 'on'; % 'off' to turn off grid
hold on;
title(handles.axes1, plotTitle,'FontSize',10);
xlabel(handles.axes1, xlabel,'FontSize',8);
ylabel(handles.axes1, ylabel,'FontSize',8);
grid(plotGrid);
plot(handles.axes1,1:1:length(chpower),chpower);
hold off;
%Devuelve marca de tiempo, latitud, longitud y altura
fprintf(Anritsu,:FETCh:GPS:FULL?');
% fprintf(Anritsu,:FETCh:GPS:LAST?');
```

```

gps = fscanf(Anritsu);
% Comprueba si se recibe que no hay conexión y se rellenan todos los
% campos de los datos GPS con NF
C = 'NO FIX';
C = [C newline ''];
tf = strcmp(C,gps);
if tf == 1
    for j=1:7
        gpstotal{i,j} = 'NF';
    end
else
% Si no la hay guarda los 7 campos de los datos gps normalmente
gps2=split(gps,[ " ",",",""]);
gps3 = gps2.';
for j=1:7
gpstotal{i,j} = gps3{1,j};
end
end
medidas = horzcat(chpower2, gpstotal);
set(handles.Tomas,'String',i);
fclose(Anritsu);
guidata(hObject,handles);

% --- Executes on button press in Conectar.
function Conectar_Callback(hObject, eventdata, handles)
% hObject      handle to Conectar (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles       structure with handles and user data (see GUIDATA)
global medidas gpstotal chpower chpower2 i;
medidas = {};
gpstotal = {};
chpower = zeros ();
chpower2 = {};
i = 0;

%Checking if the user has filled in all the fields
if isempty (get(handles.Center_Frequency_edit_text,'String')) ||
isempty(get(handles.SPAN_edit_text,'String')) ||
isempty(handles.unidades) ||
isempty(get(handles.Sweep_point_edit_text,'String')) ||
isempty(get(handles.Sweep_point_edit_text,'String')) ||
isempty(get(handles.Integration_BW,'String'))
    errordlg('Fill in the blanks','Warning')
else
%% Instrument Connection
% Find a tcpip object.
% 192.168.1.100 (Para router de Leandro)
% 192.168.0.197 (Para router Sicomo)
Anritsu = instrfind('Type', 'tcpip', 'RemoteHost', '192.168.43.53',
'RemotePort', 9001, 'Tag', '');
% Create the tcpip object if it does not exist
% otherwise use the object that was found.

if isempty(Anritsu)
    Anritsu = tcpip('192.168.43.53', 9001);
else
    fclose(Anritsu);
    Anritsu = Anritsu(1);
end

```

```
% Connect to instrument object, obj1.
handles.Anritsu = Anritsu;
fopen(Anritsu);
%% Instrument Configuration and Control
% Fija el span bajo elección del usuario:
fprintf(Anritsu, [':FREQuency:SPAN
',get(handles.SPAN_edit_text,'String'),',',handles.unidades]);
% Fija la frecuencia central bajo elección del usuario:
fprintf(Anritsu, [':FREQuency:CENTER
',get(handles.Center_Frequency_edit_text,'String'),',
',handles.unidades2]);
% Fija el nivel de refetencia (eje x) a -20 dbm bajo elección del
usuario:
fprintf(Anritsu, 'DISP:WIND:TRAC:Y:SCAL:RLEV -20');
% Modo de medida Channel power:
fprintf(Anritsu,'CONFigure:CHPower');
%Fija el type del trace en AVG:
fprintf(Anritsu,:'TRACel:TYPE AVER');
% Activa el preamplificador o no según elección de usuario:
fprintf(Anritsu,[':POWER:RF:GAIN:STATE',' ',handles.preamp]);
%Fija la anchura de integración según elección del usuario:
fprintf(Anritsu,[':CHPower:BWIDth:INTegration
',get(handles.Integration_BW,'String'),', ', handles.unidades3]);
% Se Configura el barrido continuo o no según elección del usuario:
fprintf(Anritsu, [':INIT:CONT', ' ',handles.sweep]);
fprintf(Anritsu, [':DISPlay:POINTcount ',
get(handles.Sweep_point_edit_text,'String')]);
guidata(hObject,handles);
end

% medidas(i)=strcat(psd,gps);
% i+1;
% medidas= zeros(1000,5);
% i=1;
%Este comando devuelve TOTAL CHPOWER en dbm/HZ
% fprintf(Anritsu,:'FETCH:CHPOWER:DENSITY?');
% chpower = fscanf(Anritsu);
% fprintf(Anritsu, '*WAI');
%Devuelve marca de tiempo, latitud, longitud y altura
% fprintf(Anritsu,:'FETCh:GPS?');
% gps = str2double(fscanf(Anritsu));
% fprintf(Anritsu, '*WAI');

function Center_Frequency_edit_text_Callback(hObject, eventdata,
handles)
% hObject    handle to Center_Frequency_edit_text (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
Center_Frequency_edit_text as text
%       str2double(get(hObject,'String')) returns contents of
Center_Frequency_edit_text as a double

% --- Executes during object creation, after setting all properties.
function Center_Frequency_edit_text_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to Center_Frequency_edit_text (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 SPAN_edit_text_Callback(hObject, eventdata, handles)
% hObject handle to SPAN_edit_text (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 SPAN_edit_text as
text
%       str2double(get(hObject, 'String')) returns contents of
SPAN_edit_text as a double


% --- Executes during object creation, after setting all properties.
function SPAN_edit_text_CreateFcn(hObject, eventdata, handles)
% hObject handle to SPAN_edit_text (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 Start_drive_testing.
function Start_drive_testing_Callback(hObject, eventdata, handles)

% hObject handle to Start_drive_testing (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
global medidas i chpower gpstotal chpower2 end_drive_test;
end_drive_test = 0;
Anritsu=handles.Anritsu;
fclose(Anritsu);
fopen(Anritsu);
while (end_drive_test==0)
i = i + 1;
%Este comando devuelve TOTAL CHPOWER en dbm
fprintf(Anritsu, ':FETCh:CHPower?');
chpower1 = fscanf(Anritsu);
chpower2{i,1} = chpower1;
chpower(i,1) = str2double(chpower1);

% % % Representar la potencia % % %
cla(handles.axes1, 'reset');
plotTitle = 'Measures campaing'; % plot title
```

```

xLabel = 'Samples (n)'; % x-axis label
yLabel = 'Power (dBm)'; % y-axis label
plotGrid = 'on'; % 'off' to turn off grid
hold on;
title(handles.axes1, plotTitle,'FontSize',10);
xlabel(handles.axes1, xlabel,'FontSize',8);
ylabel(handles.axes1, ylabel,'FontSize',8);
grid(plotGrid);
plot(handles.axes1,1:1:length(chpower),chpower);
hold off;

%Devuelve marca de tiempo, latitud, longitud y altura
fprintf(Anritsu,:FETCh:GPS:FULL?');
% fprintf(Anritsu,:FETCh:GPS:LAST?');
gps = fscanf(Anritsu);
C = 'NO FIX';
C = [C newline ''];
tf = strcmp(C,gps);
if tf == 1
    for j=1:7
        gpstotal{i,j} = 'NF';
    end
else
gps2=split(gps,[ " ",",",""]);
gps3 = gps2.';
% [m,n]=size(gps3);
for j=1:7
gpstotal{i,j} = gps3{1,j};
end
end
medidas = horzcat(chpower2, gpstotal);
save('medidas','medidas');
set(handles.Tomas,'String',i);
pause(0.5);
end
guidata(hObject,handles);

% --- Executes on button press in pause_drive_test.
function pause_drive_test_Callback(hObject, eventdata, handles)
% hObject    handle to pause_drive_test (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global end_drive_test
end_drive_test=1;
guidata(hObject,handles);

% --- Executes on button press in Save.
function Save_Callback(hObject, eventdata, handles)
% hObject    handle to Save (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global medidas chpower gpstotal chpower2;
prompt = 'Name campaing: ';
campaing_name = input(prompt,'s');
if isempty(campaing_name)
    campaing_name = 'no_name';
end
save(campaing_name,'chpower','gpstotal','medidas');
Anritsu=handles.Anritsu;
fclose(Anritsu);

```

```

msgbox('Closed connection and saved campaign','OK')
guidata(hObject,handles);

function Save_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Integration_BW (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
% called

% Hint: popupmenu 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 selection change in unidades.
function unidades_Callback(hObject, eventdata, handles)
% hObject    handle to unidades (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns unidades
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
unidades
v=get(hObject,'Value');
switch v
    case 1
handles.unidades = 'HZ';
    case 2
handles.unidades = 'KHZ';
    case 3
handles.unidades = 'MHZ';
    case 4
handles.unidades = 'GHZ';
    otherwise
handles.unidades = 'HZ';
end
guidata(hObject,handles);

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

% Hint: popupmenu 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 selection change in unidades_2.
function unidades_2_Callback(hObject, eventdata, handles)
% hObject    handle to unidades_2 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```
% Hints: contents = cellstr(get(hObject,'String')) returns unidades_2
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
unidades_2
v=get(hObject,'Value');
switch v
    case 1
handles.unidades2 = 'HZ';
    case 2
handles.unidades2 = 'KHZ';
    case 3
handles.unidades2 = 'MHZ';
    case 4
handles.unidades2 = 'GHZ';
    otherwise
handles.unidades2 = 'HZ';
end
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function unidades_2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to unidades_2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu 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 Pream_edit_text_Callback(hObject, eventdata, handles)
% hObject    handle to Pream_edit_text (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 Pream_edit_text as
text
%        str2double(get(hObject,'String')) returns contents of
Pream_edit_text as a double

% --- Executes during object creation, after setting all properties.
function Pream_edit_text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Pream_edit_text (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 Sweep_point_edit_text_Callback(hObject, eventdata, handles)
% hObject    handle to Sweep_point_edit_text (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
% Sweep_point_edit_text as text
%         str2double(get(hObject,'String')) returns contents of
% Sweep_point_edit_text as a double

% --- Executes during object creation, after setting all properties.
function Sweep_point_edit_text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Sweep_point_edit_text (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 radiobutton1.
function radiobutton1_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton1 (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.radiobutton1, 'Value') == 0)
    handles.preamp = 'OFF';
else
    handles.preamp = 'ON';
end
guidata(hObject, handles);

% Hint: get(hObject, 'Value') returns toggle state of radiobutton1

% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (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.radiobutton2, 'Value') == 0)
    handles.sweep = 'OFF';
else
    handles.sweep = 'ON';
end
guidata(hObject, handles);
% Hint: get(hObject, 'Value') returns toggle state of radiobutton2

% --- Executes on selection change in popupmenu5.
function popupmenu5_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu5 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB

```

```
% handles      structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu5
contents as cell array
%       contents{get(hObject,'Value')} returns selected item from
popupmenu5

% --- Executes during object creation, after setting all properties.
function popupmenu5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu5 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu 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 Integration_BW_Callback(hObject, eventdata, handles)
% hObject    handle to Integration_BW (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 Integration_BW as
text
%       str2double(get(hObject,'String')) returns contents of
Integration_BW as a double

% --- Executes during object creation, after setting all properties.

% --- Executes on selection change in Integration_BW.

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

% Hint: popupmenu 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 selection change in unidades3.
function unidades3_Callback(hObject, eventdata, handles)
% hObject    handle to unidades3 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```

v=get(hObject,'Value');
switch v
    case 1
handles.unidades3 = 'HZ';
    case 2
handles.unidades3 = 'KHZ';
    case 3
handles.unidades3 = 'MHZ';
    case 4
handles.unidades3 = 'GHZ';
    otherwise
handles.unidades3 = 'HZ';
end
guidata(hObject,handles);

% Hints: contents = cellstr(get(hObject,'String')) returns unidades3
contents as cell array
%       contents{get(hObject,'Value')} returns selected item from
unidades3

% --- Executes during object creation, after setting all properties.
function unidades3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to unidades3 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu 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 during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes1 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: place code in OpeningFcn to populate axes1

```