

UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL
CUSCO

FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA, INFORMÁTICA Y
MECÁNICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA



“CARACTERIZACIÓN EXPERIMENTAL DEL CANAL DE
COMUNICACIÓN INALÁMBRICO DE LA ZONA
ARQUEOLÓGICA DE RUMIWASI ”

TESIS

PARA OBTENER EL TÍTULO DE:

INGENIERO ELECTRÓNICO

PRESENTADO POR:

HENRY LOIS DAVILA ANDRADE

ASESOR: ING. JORGE LUIS ARIZACA CUSICUNA

TESIS FINANCIADA POR:



Cusco- 2022

Resumen

En los últimos el sitio arqueológico de Rumiwasi ubicado en el Cusco, distrito de San Sebastian; ha sufrido varios atentados que han dejado como consecuencias varios muros dañados y pintados con aerosol, por lo que un manejo preventivo de este sitio permitiría a sus administradores conocer la ubicación, tiempo de estadía, número y recorrido de los visitantes para así tener un mejor control y cuidado de este, actualmente herramientas como una red de sensores inalámbricos (WSN) nos ofrecen una solución eficiente a este tipo de problemas pero esta requiere de una etapa previa llamada modelamiento de canal que ofrece los parámetros suficientes para obtener un diseño óptimo de esta.

En el presente trabajo tiene como objetivo general caracterizar el canal de comunicación de la zona arqueológica de Rumiwasi en el que se considero dos aspectos principales: Modelado de la propagación a gran escala y modelado de la propagación en pequeña es escala.

La caracterización de la propagación a gran escala de las ondas electromagnéticas se realizó usando modelos aplicables a las redes de sensores inalámbricos WSN usando como variable dependiente el indicador de potencia recibida (RSSI) debido a que estos modelos proveen información utilizada para la geolocalización en tiempo real sin el uso de GPS. Esto se realizó para las trayectorias LOS (línea de vista), NLOS (sin línea de vista) y LOBS (trayectoria con obstrucción). La adquisición de datos se realizó usando dos tarjetas LoRa TTGO T-BEAM (una configurada como nodo sensor y la otra como gateway), dos trípodes y una laptop. Los parámetros fueron calculados usando modelos estadísticos de regresión simple, comparando cada modelo con los datos medidos de RSSI y luego se procedió elegir el modelo más adecuado para cada trayectoria usando como parámetro de comparación la desviación estándar. A partir de esto un análisis del exponente de pérdida de trayecto en la trayectoria NLOS de $n = 15,76$, mostró que los restos arqueológicos ubicados en esta zona poseen una atenuación muy grande, por lo que si se diseña una red de sensores se recomienda usar un radio máximo de cobertura óptima de $d = 140m$.

La caracterización de la propagación en pequeña escala se realizó usando una sonda de canal en el dominio de la frecuencia. Esta consta de dos antenas tipo parche HG908LP-NF colocadas en trípodes y el Fieldfox Handheld Analyzer N9914A configurado en el modo analizador de redes vectorial con el cual se midió el parámetro S_{21} para dos de los muros más representativos de esta zona arqueológica. Este parámetro sirvió para caracterizar el tiempo de retardo RMS el cual comprobó que la atenuación en la trayectoria NLOS se debe a que consta en su mayor parte de muros con la estructura del muro de prueba 2 debido a que estos tienen el retardo RMS 2 veces que mayor que el muro de prueba 1 ($\tau_{RMS} = 11,56ns$). También se obtuvo los modelos estadísticos o procesos autoregresivos con media móvil (ARMA) de los principales muros de esta zona arqueológica cuya importancia radica en su aplicación en la simulación de datos, destacando en cada uno de ellos que para las misma distancia de prueba $d = 60cm$, el orden del proceso ARMA es de (4,2).

DEDICATORIA

A mi padre en el cielo Abraham Davila Davalos, un destacado profesor y tallador en madera, que nos dejó a mi y a mis hermanos el mejor ejemplo de que el arte y la ciencia, pueden convivir en armonía en una persona.

A mi madre Lourdes Hilaria Cama Andrade, una gran profesora que siempre vio en la educación, el futuro de sus hijos.

A Sammy Maxi Cama, una maravillosa persona que me enseñó el significado de la amistad y que por muchos años fue un gran apoyo incondicional.

A mi mejor amiga Leonela Zambrano Chata por su gran apoyo incondicional.

A mi gran amiga Alessandra Sota Gonzales por su gran apoyo, en especial su destacado trabajo en la traducción del segundo artículo científico que resultó de este trabajo de tesis.

A mis hermanos que de alguna manera influyeron en mi formación académica.

Y por último a todos mis alumnos y demás estudiantes universitarios que incluso en las peores condiciones demuestran con su esfuerzo que se puede lograr la excelencia académica.

Agradecimientos

Al CONCYTEC, PRO CIENCIA y el Banco Mundial por el financiamiento al proyecto “Análisis y caracterización experimental de un modelo de canal inalámbrico y su uso en áreas de conservación”, proyecto en el que se realizó este trabajo.

Al Ministerio de Cultura por los permisos proporcionados para realizar las pruebas necesarias en esta tesis en la zona arqueológica de Rumiwasi.

A mi asesor por la mentoría y el gran apoyo proporcionado en todo el tiempo que se realizó el presente trabajo.

Índice general

1. Generalidades	1
1.1. Planteamiento del problema	1
1.1.1. Situación problemática	1
1.1.2. Formulación del problema	2
1.2. Justificación de la investigación	2
1.3. Objetivos de la investigación	3
1.3.1. Objetivo general	3
1.3.2. Objetivos específicos	3
1.4. Alcances y limitaciones	3
1.4.1. Alcances	3
1.4.2. Limitaciones	4
2. Marco teórico	5
2.1. Concepto de un sistema de comunicación	5
2.2. Componentes de ruido en un canal de comunicación inalámbrico	6
2.2.1. Ruido aditivo	6
2.2.2. Ruido multiplicativo	6
2.3. Clasificación de modelos de canal	7
2.3.1. De acuerdo a la naturaleza matemática del modelo	7
2.3.2. De acuerdo al entorno el que se modela	8
2.3.3. De acuerdo al tipo de banda (estrecha o ancha)	8
2.3.4. De acuerdo al número de antenas en el transmisor y receptor utilizados en la sonda de canal	8
2.4. Modelos de propagación a gran escala	9
2.4.1. Mecanismos básicos de propagación	9
2.4.2. Modelo de propagación en el espacio libre	9
2.4.3. Modelo de pérdida de trayecto log-distancia	10
2.4.4. Modelo de pérdida de trayecto log-normal con ensombrecimiento o Modelo de una sola pendiente	10
2.4.5. Modelos de RSSI para Redes de Sensores Inalámbricos (WSN)	10
2.5. Desvanecimiento en pequeña escala y multitrayecto	11
2.5.1. Propagación multitrayecto	12
2.5.2. Respuesta al impulso de un canal multitrayecto [1]	12

2.5.3.	Perfil de retardo de potencia (PDP)	13
2.5.4.	Retardo de tiempo de propagación	13
2.6.	Modelamiento autoregresivo con media móvil (ARMA) de un canal de comunicación	14
2.6.1.	Selección del orden del modelo	16
2.7.	Métodos de sondeo de canal	17
2.7.1.	Técnicas de medida en el dominio del tiempo	17
2.7.2.	Técnicas de medida en el dominio de la frecuencia [3]	17
2.8.	Redes de Sensores Inalámbricos (WSN) y la Tecnología LoRa	20
2.8.1.	Características de LoRa	20
2.8.2.	Principales propiedades de la modulación LoRa [6]	21
2.8.3.	Comparación de LoRa con varias tecnologías existentes [7]	22
3.	Metodología	24
3.1.	Caracterización de la propagación a gran escala del canal de comunicación inalámbrico	24
3.1.1.	Pruebas preliminares y elección de equipos utilizados en la sonda de canal	24
3.1.2.	Características de las antenas utilizadas en las sondas de canal	31
3.1.3.	Características y comparación de todas las tarjetas utilizadas en las sondas de canal	31
3.1.4.	Configuración de las tarjetas TTGO	34
3.2.	Metodología aplicada para caracterizar el RSSI	39
3.3.	Metodología para la caracterización del tiempo de retardo RMS y modelamiento ARMA de la propagación en muros inca	43
4.	Resultados y discusión	46
4.1.	Caracterización del RSSI	46
4.1.1.	Modelos del RSSI	46
4.1.2.	Primera campaña de medidas	46
4.2.	Tiempo de retardo RMS y modelos ARMA de los muros inca principales de Rumiwasi	51
4.2.1.	Tiempo de retardo RMS	51
4.3.	Conclusiones y Recomendaciones	63
4.3.1.	Conclusiones	63
4.3.2.	Recomendaciones	64
	Bibliografía	65
	Apéndices	67
	A. Parámetros S de una red	68

B. Resumen de la función de autocorrelación simple (fas) y la función de autocorrelación parcial (fap) de los procesos ARMA [28]	70
C. Codigos utilizados en el presente trabajo	71
C.1. Codigos en Arduino para el transmisor y receptor LoRa	71
C.1.1. Transmisor	71
C.1.2. Receptor	74
C.2. Códigos en Matlab para el cálculo de parámetros del modelo para la caracterización de RSSI	76
C.2.1. Programa LOS para la primera campaña de medidas	76
C.2.2. Programa NLOS para la primera campaña de medidas	77
C.2.3. Programa NLOS.m	78
C.2.4. Programa LOBS.m	80
C.2.5. Programa retardo2.m	83
C.3. Notebooks de Python	85

Índice de figuras

2.1. Diagrama esquemático de un sistema de comunicación generalizado	5
2.2. Componentes de ruido	6
2.3. Ruido multiplicativo	7
2.4. Típico perfil de retardo de potencia Fuente: Adaptado de [21]	13
2.5. Modelo ARMA (generador del proceso) de orden (M, K) , asumiendo $M > K$ Fuente: Adaptado de [23]	15
2.6. Sistema de medida en el dominio de la frecuencia. Fuente: Adaptado de [1, 3]	18
2.7. (a) Diagrama de bloques de la medición de la respuesta en frecuencia usando un analizador de redes como sistema de medidas (b) Postprocesado de los datos para el cálculos de la respuesta al impulso periódica Fuente: Adaptado de [18]	19
3.1. Primera propuesta de la sonda de canal	25
3.2. Pulso recibido usando en el Field Fox Handheld Analyzer	26
3.3. Diagrama de bloques de la variante de la primera sonda de canal	26
3.4. Variante de la primera sonda de canal	27
3.5. Diagrama de bloques de la variante de la segunda propuesta de la sonda de canal	27
3.6. Implementación de la segunda propuesta de la sonda de canal	28
3.7. Primera opción de la tercera propuesta de la sonda de canal	30
3.8. Diagrama de bloques del sistema de medidas de la segunda opción de la Tercera propuesta de la sonda de canal.	30
3.9. Pulso transmitido por el TTGO t-beam medido usando el Fieldfox Handheld Analyzer N9914A	31
3.10. Pruebas preliminares	32
3.11. Primer recorrido Fuente: Google Earth Pro	32
3.12. Segundo recorrido Fuente: Google Earth Pro	33
3.13. Tercer recorrido Fuente: Google Earth Pro	33
3.14. Cuarto recorrido Fuente: Google Earth Pro	34
3.15. Sistema de adquisición de datos o sonda de canal	40
3.16. Algunas posiciones de prueba	41
3.17. Trayectoria LOS y NLOS (Posiciones amarillas) y trayectoria LOBS (Posi- ciones rojas) Fuente: Google Earth Pro	42
3.18. Diagrama de bloques del sistema de medida Fuente: elaboración propia	43

3.19. Muro de prueba 1	44
3.20. Muro de prueba 2	44
4.1. Primera campaña de medidas	47
4.2. Modelo log-normal de una sola pendiente para la trayectoria LOS de la primera campaña de medidas	47
4.3. Modelo log-normal de una sola pendiente para la trayectoria NLOS de la primera campaña de medidas	48
4.4. Comparación de modelos para la trayectoria NLOS	49
4.5. Comparación de modelos para la trayectoria LOBS	50
4.6. Perfil retardo de potencia muro de prueba 1 a una distancia de 30cm	51
4.7. Perfil retardo de potencia muro de prueba 1 a una distancia de 60cm	52
4.8. Perfil retardo de potencia muro de prueba 2 a una distancia de 60cm	53
4.9. Señal medida normalizada a una distancia de 30cm	54
4.10. Señal medida normalizada a una distancia de 30cm	54
4.11. Señal medida normalizada a una distancia de 30cm	55
4.12. Comparación de la predicción del modelo con los datos obtenidos	55
4.13. Grafica de los residuos del modelo	56
4.14. Gráfico de densidad de los valores de error residual	56
4.15. Señal medida normalizada a una distancia de 60cm	57
4.16. Señal medida normalizada a una distancia de 60cm	57
4.17. Señal medida normalizada a una distancia de 60cm	58
4.18. Comparación de la predicción del modelo con los datos obtenidos	58
4.19. Grafica de los residuos del modelo	59
4.20. Gráfico de densidad de los valores de error residual	59
4.21. Señal medida normalizada a una distancia de 60cm	60
4.22. Señal medida normalizada a una distancia de 60cm	61
4.23. Señal medida normalizada a una distancia de 60cm	61
4.24. Comparación de la predicción del modelo con los datos obtenidos	62
4.25. Grafica de los residuos del modelo	62
4.26. Gráfico de densidad de los valores de error residual	63
A.1. Parámetros S de un dispositivo bajo prueba (DUT) Fuente: Adaptado de [35]	69
A.2. Medición de los parámetros S Fuente: Adaptado de [35]	69

Índice de cuadros

3.1. Ventajas y desventajas de la primera sonda de canal	26
3.2. Ventajas y desventajas de la variante de la sonda de canal	28
3.3. Ventajas y desventajas de la segunda propuesta de la sonda de canal	29
3.4. Ventajas y desventajas de la primera opción de la tercera propuesta de la sonda de canal	29
3.5. Ventajas y desventajas de la segunda opción de la tercera propuesta de la sonda de canal	34
3.6. Características de las antenas utilizadas	35
3.7. Comparación de tarjetas LoRa	36
3.8. Resumen de las ventajas y desventajas de las tarjetas LoRa comúnmente utilizadas	37
4.1. Parámetros del modelo log-normal de una sola pendiente para las trayectorias LOS y NLOS de la primera campaña de medidas	48
4.2. Parámetros de los modelos para NLOS	49
4.3. Parámetros de los modelos para LOBS	50
4.4. Valores estadísticos del tiempo de retardo para ambos muros de prueba	52
B.1. Resumen de las <i>fas</i> y <i>fap</i> de procesos ARMA	70

Capítulo 1

Generalidades

1.1. Planteamiento del problema

1.1.1. Situación problemática

El modelamiento de un canal de comunicación es de importancia para el análisis y diseño de sistemas de comunicación [1, 2, 4], es por esto que numerosos investigadores han desarrollado estudios extensos sobre los mecanismos y parámetros que caracterizan la propagación de ondas electromagnéticas siendo la mayoría de estos realizados en principales ciudades ubicadas fuera de latinoamerica [12, 3, 19, 20].

Cusco es uno de los principales destinos turísticos del Perú, cuyo impacto es de vital importancia para el desarrollo de actividades productivas. Dicho impacto es en su mayoría positivo, sin embargo, no se puede ocultar que este trae un lado que repercute negativamente en el patrimonio; principalmente por el alto flujo de visitantes por lo que en los últimos años ha presentado sobreexplotación en sus distintas zonas arqueológicas, razón por la cual las autoridades han elaborado una serie de normas y mecanismos que permiten gestionar mejor el número de visitantes, tiempo de visita y recorrido de estos para evitar de la congestión del área de conservación como ejemplo de esto encontramos el “Reglamento de uso sostenible y visita turística para la conservación de la Llaqta o ciudad inca de Machupicchu”¹. Aunque en el Perú se está tratando de empezar a utilizar las herramientas tecnológicas para la conservación del patrimonio, esto solo se limita al uso de sistemas informáticos para la disminución de trámites administrativos.

Las tecnologías actuales ofrecen distintas soluciones para el monitoreo en tiempo real de visitantes en una zona arqueológica; una de estas es el uso de una red de sensores inalámbricos o elementos de geolocalización conectados a un nodo central, otra es el sistema de posición geográfica (GPS). Diversos investigadores proponen el uso del primero por sus ventajas comparado con el segundo [12, 13, 14], entre las cuales podemos destacar las siguientes:

¹<http://transparencia.cultura.gob.pe/sites/default/files/transparencia/2017/02/resoluciones-ministeriales/rm070.pdf>

bajo consumo de energía y bajo costo. Estos métodos de geolocalización usando una red de sensores inalámbricos dependen principalmente de los parámetros de propagación de un canal inalámbrico tales como: intensidad de la señal recibida (RSSI), tiempo de salida (ToA) o también la diferencia de tiempo de salida (TDoA) de los cuales depende de manera crítica su precisión [8]. Por tanto el modelamiento de canal de una zona arqueológica no solo facilitaría el diseño de una red de sensores inalámbricos con el que se podría monitorear distintas variables del entorno, sino que permitiría un futuro proyecto de geolocalización de visitantes en esta.

1.1.2. Formulación del problema

Problema general

Existe una gestión básica en tiempo real de visitantes, tiempo de visita y recorrido en la zonas arqueológicas de Cusco. Una propuesta para mejorar la gestión de zonas arqueológicas comprendería el estudio de la propagación de ondas electromagnéticas o modelamiento de canal de estas zonas arqueológicas porque no sólo proporcionaría los parámetros para un proyecto futuro de diseño de una red de sensores inalámbricos, sino que también permitiría la geolocalización de visitantes sin el uso de GPS.

Problemas específicos

- ¿Qué modelos y características tiene un canal inalámbrico?
- ¿Qué modelo o modelos son aplicables al entorno geográfico de la zona arqueológica de Rumiwasi?
- ¿Cómo obtener y analizar los parámetros del canal inalámbrico de la zona arqueológica de Rumiwasi?

1.2. Justificación de la investigación

En los últimos el sitio arqueológico de Rumiwasi ubicado en el Cusco, distrito de San Sebastian; ha sufrido varios atentados que han dejado como consecuencias varios muros dañados y pintados con aerosol ², por lo que un manejo preventivo de este sitio permitiría a sus administradores conocer la ubicación, tiempo de estadía, número y recorrido de los visitantes para así tener un mejor control y cuidado de este, actualmente herramientas como una red de sensores inalámbricos (WSN) nos ofrecen una solución eficiente a este tipo de problemas pero esta requiere de una etapa previa llamada modelamiento de canal que ofrece los parámetros suficientes para obtener un diseño óptimo de esta [4].

²<https://andina.pe/agencia/noticia-restauran-piedras-muro-inca-se-desprendieron-sitio-arqueologico-cusco-804626.aspx>

1.3. Objetivos de la investigación

1.3.1. Objetivo general

Caracterizar experimentalmente el canal inalámbrico de la zona arqueológica de Rumiwasi.

1.3.2. Objetivos específicos

- Identificar distintos modelos de propagación de ondas electromagnéticas y establecer las características del canal inalámbrico.
- Elegir el modelo o modelos experimentales más adecuados al entorno geográfico de la zona arqueológica de Rumiwasi.
- Calcular y analizar los parámetros del modelo en Matlab y Python, luego de realizar una campaña experimental de medidas.

1.4. Alcances y limitaciones

1.4.1. Alcances

- El estudio se realizará en la banda *ISM* (banda de radio industrial, científica y médica) del Perú.
- Se implementará una equipo de transmisión y recepción (sonda de canal); para caracterizar el RSSI en tres trayectorias distintas LOS (Linea de vista), NLOS (no linea de vista) y OBS (Trayectoria con obstrucción) ubicados en la zona arqueológica de Rumiwasi, de tal manera que los datos obtenidos sean representativos y caractericen mejor el comportamiento del canal a modelar. El equipamiento será proporcionado por el Centro de Investigación y Servicios Tecnológicos ubicado en el 5^o piso de la Escuela Profesional de Ingeniería Electrónica. Una vez realizada la adquisición de datos se procederá al cálculo y análisis gráfico de los parámetros del modelo de pérdida de trayecto (pérdida de trayecto inicial, exponente de perdida de trayecto, etc) con el uso de un programa implementado en Matlab.
- Se implementará un sonda de canal en dominio de la frecuencia usando el Fieldfox Handheld analyzer configurado en modo VNA (analizador vectorial de redes), cables de conexión y dos antenas receptoras tipo parche con el que se obtendra el parámetro S_{21} que viene a ser la respuesta en frecuencia del canal inalámbrico. Una vez realizado esto se procederá a implementar un programa para calcular los retardos de propagación a partir del PDP (perfil de retardo de potencia); también se implementara otro programa en Python para estimar los parámetros de la técnica de estimación espectral de modelamiento paramétrico ARMA (modelo autoregresivo con media móvil) o también

conocido como modelado de super-resolución utilizado para reducir la interferencia en los datos obtenidos [22].

1.4.2. Limitaciones

- En este trabajo de investigación solo se usará modelos SISO (entrada simple-salida simple) de canal inalámbrico, esto debido a dos razones:
 1. La red de sensores que se podría diseñar usaría las tecnologías 3G y 3.5G.
 2. La instrumentación a utilizar en la sonda de canal es del tipo SISO.
- No será posible caracterizar el tiempo de salida (TOA) del modelo, utilizando la sonda en el dominio de la frecuencia.
- Debido a que las posiciones de las antenas serán fijas, el efecto Doppler será despreciable por lo que no se considerara como un parámetro en el modelamiento de canal.

Capítulo 2

Marco teórico

2.1. Concepto de un sistema de comunicación

En 1948 Claude Shannon en su artículo “*A mathematical theory of communication*” [15], plantea un diagrama esquemático de un sistema de comunicación generalizado como se muestra en la figura 2.1.

Este trabajo refiere que todo sistema de comunicación es aquel que consta de por lo menos

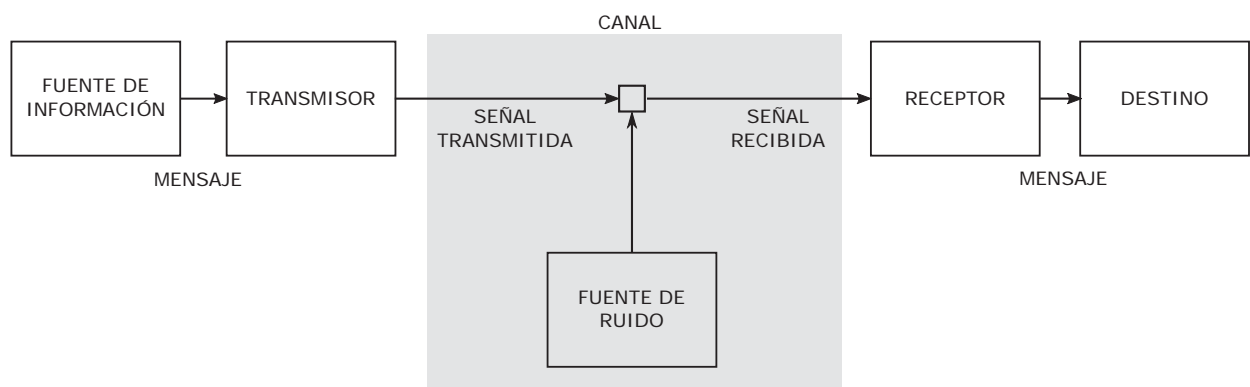


Figura 2.1: Diagrama esquemático de un sistema de comunicación generalizado, Fuente: Adaptado de [15]

5 partes:

1. *Una fuente de información* que produce el mensaje o secuencia de mensajes para ser comunicados a la terminal del receptor.
2. *Un transmisor* que acondiciona el mensaje para producir una señal adecuada para transmitirse en el canal.

3. *El canal*, es simplemente el medio usado para transmitir la señal del transmisor al receptor, esta puede ser alámbrico o inalámbrico.
4. *El receptor*, ordinariamente realiza la operación inversa que realiza el transmisor, reconstruye el mensaje de la señal.
5. *El destino*, es la persona o objeto para quién esta destinado el mensaje.

2.2. Componentes de ruido en un canal de comunicación inalámbrico

S. Saunders y A. Aragón Zavala refieren que el ruido en un sistema de comunicación puede descomponerse en dos partes como muestra la figura 2.2 [16]:

- Ruido aditivo
- Ruido multiplicativo

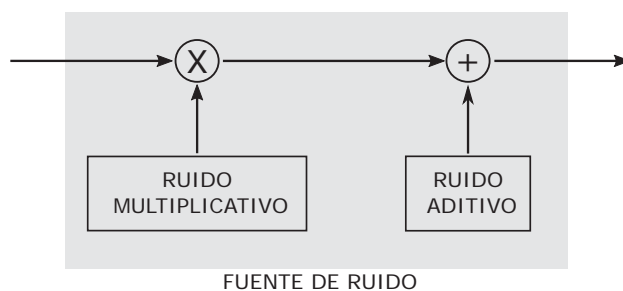


Figura 2.2: Componentes de ruido. Fuente: Adaptado de [16]

2.2.1. Ruido aditivo

Surge del ruido generado dentro del receptor tales como el ruido termal y el ruido de disparo en componentes activos y pasivos, también de fuentes externas tales como efectos atmosféricos, radiación cósmica y interferencia de otros transmisores y accesorios eléctricos.

2.2.2. Ruido multiplicativo

Surge de varios procesos encontrados por las ondas transmitidas en el trayecto del transmisor al receptor. Algunos de estos procesos son los siguientes:

- Las características direccionales de las antenas del transmisor y receptor.

- La reflexión.
- Absorción.
- Dispersión.
- Difracción.
- Refracción.

A su vez podemos subdividir el ruido multiplicativo en tres tipos de desvanecimiento: *pérdida de trayecto*, *ensombrecimiento* (ó *desvanecimiento lento*) y *desvanecimiento rápido* ó (*desvanecimiento multitrayecto*) como muestra la figura 2.3.

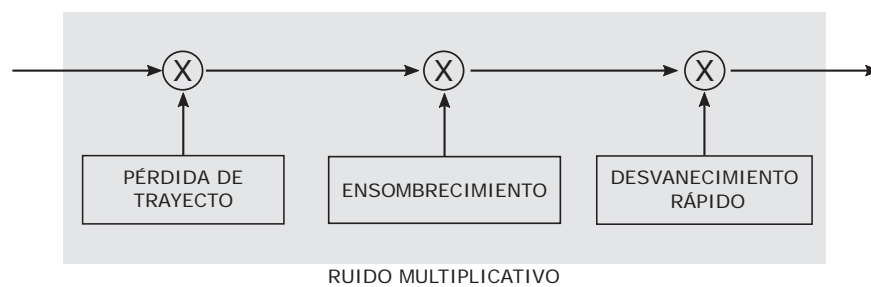


Figura 2.3: Ruido multiplicativo, Fuente: Adaptado de [16]

2.3. Clasificación de modelos de canal

Los canales inalámbricos se clasifican según sus características, y pueden clasificarse como:

- De acuerdo a la naturaleza matemática del modelo.
- De acuerdo al entorno en el que se modela
- De acuerdo al tipo de banda (estrecha o ancha).
- De acuerdo al número de antenas en el transmisor y receptor utilizados en la sonda de canal.

2.3.1. De acuerdo a la naturaleza matemática del modelo

- Modelos determinísticos. Son aquellos en los que todas las características del canal son conocidas, razón por la cual podemos estimar los parámetros de la señal recibida como es: amplitud, fase y frecuencia, como ejemplos tenemos: los modelos basados en el trazado de rayos y los modelos basados en medidas (empíricas).

- Modelos estocásticos. Estos modelos están basados en la función de densidad de probabilidad (PDF) de la respuesta al impulso de canal o sus funciones equivalentes, consideran los efectos aleatorios generados en la señal recibida originados por el entorno (ensombrecimiento).

2.3.2. De acuerdo al entorno el que se modela

- Modelos para entornos exteriores. La transmisión de radio en un sistema de comunicaciones móvil a menudo toma lugar sobre terrenos irregulares, la presencia de árboles, construcciones y otros obstáculos deben ser tomados en cuenta, además el perfil del terreno influye de manera preponderante en la estimación de la pérdida de trayecto.
- Modelos para entornos interiores. Difiere del canal de radio móvil tradicional en dos aspectos: las distancias cubiertas son mucho más pequeñas y la variabilidad del entorno es mucho más grande para un rango más pequeño de las distancias de separación entre transmisor y receptor.

2.3.3. De acuerdo al tipo de banda (estrecha o ancha)

- Modelos de banda angosta. También llamados modelos *selectivos en frecuencia*, se dan cuando el retardo de propagación es mucho menor que la inversa del ancho de banda de la señal transmitida, lo que quiere decir que el retardo de las componentes multitrayecto de la señal transmitida son despreciables.
- Modelos de banda ancha o modelos *no selectivos en frecuencia*. Se dan cuando el retardo de propagación de las componentes multitrayecto no son despreciables. Esto influye si se usa una modulación lineal para la transmisión en un canal con banda ancha puede generarse interferencia intersímbolo (ISI), dependiendo del tiempo de símbolo es mayor o menor que el retardo de propagación.

2.3.4. De acuerdo al número de antenas en el transmisor y receptor utilizados en la sonda de canal

- Modelos de canal SISO. Estos modelos se caracterizan por solo una antena usada en el transmisor y una antena en el receptor.
- Modelos de canal SIMO. Estos se caracterizan por el uso de un transmisor y múltiples antenas de recepción.
- Modelos de canal MISO. Estos se caracterizan por usar múltiples antenas de transmisión y solo una de recepción.
- Modelos de canal MIMO. Estos se caracterizan por el uso de múltiples antenas de transmisión y múltiples antenas de recepción.

2.4. Modelos de propagación a gran escala

2.4.1. Mecanismos básicos de propagación

Existen tres mecanismos de propagación:

- La *reflexión*, ocurre cuando una onda plana electromagnética encuentra un obstáculo cuyo tamaño es mayor que la longitud de onda de la señal incidente.
- La *difracción*, ocurre cuando una onda plana electromagnética encuentra un obstáculo cuyo tamaño es del mismo orden de la longitud de onda de la señal incidente.
- La *dispersión*, ocurre cuando una onda plana electromagnética encuentra un obstáculo cuyo tamaño es mucho menor que la longitud de onda de la señal incidente.

2.4.2. Modelo de propagación en el espacio libre

Este modelo expresa la relación entre la potencia recibida y la distancia entre el transmisor y receptor de tal manera que no haya obstrucción en la línea de vista, viene dado por la ecuación de Friis mostrado a continuación:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2} \quad (2.1)$$

donde P_t es la potencia transmitida, $P_r(d)$ es la potencia recibida, G_t y G_r son las ganancias del transmisor y receptor respectivamente, d es la distancia en metros entre transmisor y receptor y λ es la longitud de onda de la señal transmitida en metros.

El modelo de espacio libre de Friis solo es valido para valores de P_r medidos en distancias ubicadas en el campo lejano del transmisor. El campo lejano o región de Fraunhofer de una antena transmisora esta definida como la región ubicada mas alla de la distancia de campo lejano d_f , dada por:

$$d_f = \frac{2D^2}{\lambda} \quad (2.2)$$

donde D es la distancia fisica lineal mas larga de la antena. La *pérdida de trayecto* relaciona la potencia transmitida y la potencia recibida, como se muestra a continuación:

$$L(dB) = 10 \log \frac{P_t}{P_r} \quad (2.3)$$

Tambien puede ser expresada como la perdida de espacio libre y la pérdida por exceso,

$$L(dB) = L_0 + L_{exceso} \quad (2.4)$$

Donde L_0 es la pérdida en el espacio libre expresado en unidades prácticas entre antenas isotrópicas con ganancias unitarias,

$$L_0(dB) = 32,44 + 20 \log f_{MHz} + 20 \log d_{Km} \quad (2.5)$$

2.4.3. Modelo de pérdida de trayecto log-distancia

Si analizamos (2.1) no es posible calcular la potencia recibida para $d = 0$, por tanto en la práctica es necesario tomar un punto de referencia desde el cual realizar las mediciones, el modelo que se muestra a continuación refleja esta características y además considera el decremento logaritmico de la potencia recibida con la distancia.

$$L(dB) = L(d_0) + 10n \log \frac{d}{d_0} \quad (2.6)$$

Donde n es el exponente de pérdida de trayecto que indica la razón de incremento de la pérdida de trayecto, d_0 es l distancia de referencia en el que se realizan las medidas y d es la distancia entre el transmisor y receptor.

2.4.4. Modelo de pérdida de trayecto log-normal con ensombrecimiento o Modelo de una sola pendiente

Este modelo considera las variaciones aleatorias generadas por el entorno a diferencia de (2.6), por lo que es una aproximación más real de la pérdida de trayecto.

$$L(dB) = L(d_0) + 10n \log \frac{d}{d_0} + X_\sigma \quad (2.7)$$

Donde X_σ es una variable aleatoria con distribución Gaussiana con media cero y desviación estándar σ (también en dB).

2.4.5. Modelos de RSSI para Redes de Sensores Inalámbricos (WSN)

Estos modelos están basados en los efectos de la propagación a gran escala pero con las restricciones de WSN, es este trabajo sólo consideramos los modelos con mayor precisión [4].

Restricciones de WSN

Debido a las aplicaciones y los entornos en los que se aplica tendremos las siguientes restricciones [4]:

- Alturas bajas de las antenas.
- Baja potencia de transmisión.
- Topología de red estacionaria.
- Directividad de las antenas. Usualmente se consideran omnidireccionales.

Modelo log-normal de una sola pendiente

Este modelo es el mismo que describimos en (2.7) pero usamos como parámetro el Received Signal Strength Indicator (RSSI) o indicador de intensidad potencia recibida en lugar de la pérdida de trayecto L , cuya ecuación se muestra a continuación:

$$RSSI(d) = RSSI(d_0) - 10n \log\left(\frac{d}{d_0}\right) + X_\sigma \quad (2.8)$$

Donde X_σ es una variable aleatoria con distribución gaussiana con desviación estándar σ , n es el exponente de pérdida de trayecto, d_0 la distancia inicial en la que se realizan las pruebas. El $RSSI$ se mide en dBm y d, d_0 en metros. Los parámetros de este modelo pueden ser fácilmente calculados usando un modelo de regresión simple.

Modelo log-normal de doble pendiente

Este modelo es una variante de la ecuación (2.8) pero esta es aplicada cuando en el entorno en el que se realiza las pruebas presenta dos regiones con características diferentes cuya ecuación se presenta a continuación:

$$RSSI(d) = \begin{cases} RSSI(d_{01}) - 10n_1 \log(d) + X_{\sigma 01} & \text{si } d \leq d_b \\ RSSI(d_{02}) - 10n_2 \log(d) + X_{\sigma 02} & \text{si } d > d_b \end{cases} \quad (2.9)$$

Donde d_b es el punto de ruptura o punto en el que hay el cambio de las características del medio, n_1 y n_2 son los exponentes de pérdida de trayecto de cada región, $X_{\sigma 01}$ y $X_{\sigma 02}$ son variables aleatorias con distribución gaussiana con media cero y desviaciones estándar σ_{01} y σ_{02} respectivamente. El $RSSI$ se mide en dBm y d, d_{01}, d_{02} en metros. Los parámetros de este modelo pueden ser fácilmente calculados usando un modelo de regresión simple para la recta de cada región.

2.5. Desvanecimiento en pequeña escala y multitrayecto

Este describe pequeñas fluctuaciones de las amplitudes, fases o retrasos multitrayecto de una señal en un corto periodo de tiempo o distancia de vuelo de la señal [1].

2.5.1. Propagación multitrayecto

Este efecto se produce por la presencia de objetos reflectantes y disipadores en el canal que cambian frecuentemente el entorno que disipa la señal de energía en amplitud, fase, tiempo. Estos efectos resultan en múltiples versiones de la señal transmitida que llega a la antena receptora, desplazados respecto el uno del otro en tiempo y orientación espacial. La fase y las amplitudes aleatorias de los diferentes componentes multitrayecto causan fluctuaciones en intensidad de la señal de este modo induce desvanecimiento a pequeña escala, distorsión de la señal o ambos. La propagación multitrayecto a menudo alarga el tiempo para que la porción de la señal en banda base para alcanzar el receptor que puede causar mancha en la señal debido a la interferencia intersimbolo (ISI) [1].

2.5.2. Respuesta al impulso de un canal multitrayecto [1]

Si la respuesta al impulso se asume invariante en el tiempo o es al menos estacionario en el sentido amplio (wide sense stationary) sobre una pequeña escala de tiempo o intervalo de distancia, entonces la respuesta al impulso del canal puede ser simplificada como:

$$h_b(\tau) = \sum_{i=0}^{N-1} a_i e^{j\theta_i} \delta(\tau - \tau_i) \quad (2.10)$$

La suposición de la invariancia de tiempo es valido cuando la resolución del tiempo de retardo del modelo de la respuesta al impulso del canal es preciso y resuelve de manera única cada componente multitrayecto sobre el área local.

Cuando medimos o predecimos $h_b(\tau)$, un pulso de prueba $p(t)$ que se aproxima a una función delta es usada en el transmisor. Esto es,

$$p(t) = \delta(t - \tau) \quad (2.11)$$

que es usada para sondear el canal para determinar $h_b(\tau)$.

Para el modelamiento de canal en pequeña escala, el perfil de retardo de potencia o power delay profile (PDP) se encuentra tomando el promedio espacial de $|h_b(t; \tau)|^2$ sobre el área local. Haciendo varias medidas en el área local de $|h_b(t; \tau)|^2$ en diferentes ubicaciones, es posible construir un conjunto de power delay profiles, cada uno representando un posible estado del canal multitrayecto en pequeña escala.

Si $p(t)$ tiene un tiempo de duración mucho menor que la respuesta al impulso del canal multitrayecto, $p(t)$ no necesita ser deconvolucionado de la señal recibida $r(t)$ en razón a determinar las intensidad relativa de las señales multitrayecto. El perfil de retardo de potencia es un área local esta dado por:

$$P(\tau) = \overline{|h_b(t; \tau)|^2} \quad (2.12)$$

donde la barra representa el valor promedio sobre el área local de las capturas de $|h_b(t; \tau)|^2$, que son típicamente promediados sobre el área local (pequeña escala) para proveer un simple

perfil de retraso de potencia multitrayecto invariante en el tiempo $P(\tau)$. La ganancia k en (2.12) relaciona la potencia transmitida en el pulso de prueba $p(t)$ para la total potencia recibida en un perfil de retardo de potencia multitrayecto.

2.5.3. Perfil de retardo de potencia (PDP)

Los canales aleatorios y de radio-propagación complicada pueden ser caracterizados usando el enfoque de la respuesta al impulso. Para cada punto en el entorno tridimensional, el canal es un filtro lineal con respuesta al impulso $h(t)$. La respuesta al impulso provee una caracterización de banda ancha de la propagación del canal y contiene toda la información necesaria para simular o analizar cualquier tipo de radio-transmisión mediante el canal [21]. Un canal móvil exhibe una estructura continua multitrayecto, por consiguiente el perfil de retardo de potencia puede ser expresado mediante una función de densidad de la forma [21]:

$$P(\tau) = \frac{|h(t)|^2}{\int_{-\infty}^{\infty} |h(t)|^2 dt} \quad (2.13)$$

Una típica grafica se muestra en la figura 2.4.

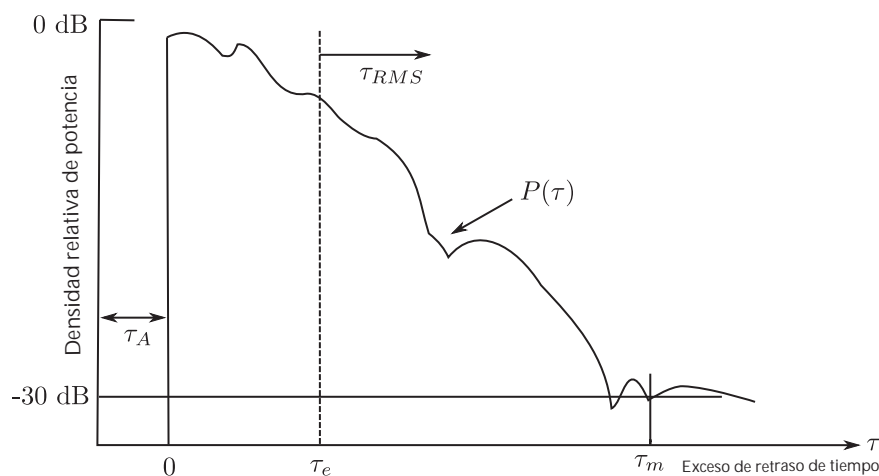


Figura 2.4: Típico perfil de retardo de potencia Fuente: Adaptado de [21]

2.5.4. Retardo de tiempo de propagación

La dispersión de tiempo varía ampliamente en un canal de radio móvil, debido al hecho de que las reflexiones y dispersiones ocurren aparentemente en posiciones aleatorias y la resultante respuesta del canal multitrayecto parece también aleatoria, porque la dispersión de tiempo depende de las relaciones de las posiciones geométricas entre el transmisor, el receptor y el entorno físico circundante, algunos parámetros que pueden cuantificar de manera grosera el canal multitrayecto se describen a continuación [21].

Retardo de primera llegada (τ_A)

Este es el tiempo correspondiente a la llegada de la primera señal transmitida en el receptor. Este es usualmente medida en el receptor. Este retardo esta establecido por el mínimo retardo de propagación posible del transmisor al receptor. Esta sirve como referencia, y todos los demás retardos son medidos respecto a este. Este es el origen como se muestra en la Figura 2.4. Cualquier medida de retardo mayor que esta referencia es considerada un retardo excesivo [21].

Retardo excesivo medio (τ_e)

Es el primer momento del perfil de retardo de potencia como se muestra en la Figura 2.4 con respecto al primer retardo [21]. Este esta expresado como:

$$\tau_e = \int (\tau - \tau_A) P(\tau) d\tau \quad (2.14)$$

Retardo RMS (τ_{RMS})

Esta es la raíz cuadrada del segundo momento central de un perfil de retardo de potencia como se puede ver en la Figura 2.4. Esta es la desviación estándar sobre el retardo excesivo medio y puede ser expresado como [21]:

$$\tau_{RMS} = \sqrt{\int (\tau - \tau_e - \tau_A)^2 P(\tau) d\tau} \quad (2.15)$$

El retardo RMS es una buena medida de la propagación multitrayecto. Este da una indicación de la naturaleza de la interferencia intersimbolo (ISI) [21].

Retardo excesivo máximo (τ_m)

Este valor es medido con respecto a un nivel específico de potencia, el cual es caracterizado como el umbral de la señal. Cuando el nivel de señal es mas pequeño que el umbral, esta se procesa como ruido. Por ejemplo el retardo excesivo máximo puede ser especificado como el retardo excesivo para el cual $P(\tau)$ cae debajo de los -30 dB con respecto a este valor pico como se muestra en la Figura 2.4 [21].

2.6. Modelamiento autoregresivo con media móvil (AR-MA) de un canal de comunicación

La respuesta en frecuencia de un canal puede ser interpretada como la salida de un proceso autoregresivo con media móvil (ARMA). Asumiendo este proceso de orden (M, K) ,

2.6.1. Selección del orden del modelo

Existen distintos criterios pero en este trabajo utilizaremos el ampliamente utilizado en modelamiento de canales de comunicación Criterio de información de Akaike (AIC) [3, 18, 22].

Criterio de información de Akaike (AIC)

Sea $u_i = u(i), i = 1, 2, \dots, N$, denota los datos obtenidos por N observaciones independientes de un proceso estocástico estacionario en tiempo discreto y $g(u_i)$ denota la función de densidad de probabilidad de u_i . Sea $f_U(u_i|\hat{\theta}_m)$ denota la función de densidad de probabilidad condicional de u_i dado $\hat{\theta}_m$ que es el vector estimado de los parámetros del modelo del proceso. Sea m el orden del modelo, de tal manera que podemos escribir :

$$\hat{\theta}_m = [\hat{\theta}_{m1}, \hat{\theta}_{m2}, \dots, \hat{\theta}_{mm}]^T \quad (2.18)$$

Así tenemos varios modelos que compiten el uno con el otro para representar el proceso de interés. El criterio de información (AIC) [23] propuesto por Akaike selecciona el modelo para el cual la cantidad:

$$AIC(m) = -2L(\hat{\theta}_m) + 2m \quad (2.19)$$

es mínima y la función $L(\hat{\theta}_m)$ viene dada por

$$L(\hat{\theta}_m) = \max \sum_{i=1}^N \ln f_U(u_i|\hat{\theta}_m), \quad (2.20)$$

Donde \ln denota el logaritmo natural. El criterio expresado en (2.20) se deriva minimizando la divergencia de Kullback-Leibler que es usado para proveer una medida de divergencia entre la función de densidad de probabilidad verdadera “desconocida” $g(u)$ y la función de densidad de probabilidad condicional $f_U(u_i|\hat{\theta}_m)$ dado por el modelo a luz de los datos observados.

La función $L(\hat{\theta}_m)$ en (2.19) excepto por un escalar es reconocido como el logaritmo de las estimaciones de máxima verosimilitud de los parámetros en el modelo. El segundo termino, $2m$ representa las consecuencias de la complejidad del modelo que hace que el $AIC(m)$ una estimación de la divergencia de Kullback-Leibler.

El primer termino de la ecuación (2.19) tiende a decrecer rápidamente con el orden del modelo m . Por otra parte el segundo termino incrementa linealmente con m . El resultado es que si graficáramos el $AIC(m)$ versus m la grafica en general mostraría una valor mínimo definido y el orden optimo del modelo esta definido por el cual el valor de m para el cual $AIC(m)$ alcanza el valor mínimo.

Para procesos autoregresivos con media móvil (ARMA) de orden (p, q) , el criterio de

Akaike normalizado [30] tiene la siguiente forma:

$$AIC(p, q) = \ln(\hat{\sigma}_v^2) + \frac{2(p + q)}{N} \quad (2.21)$$

Donde $\hat{\sigma}_v^2$ es la estimación de máxima verosimilitud de σ_v^2 .

2.7. Métodos de sondeo de canal

2.7.1. Técnicas de medida en el dominio del tiempo

El objetivo de este tipo de sistemas es obtener la respuesta al impulso del canal,

$$h(\tau, t) = \sum_{i=1}^L \beta_i(t) \delta(t - \tau_i(t)) e^{\phi_i(t)} \quad (2.22)$$

por medidas directas de β_i , τ_i y ϕ que representan la magnitud, tiempo y fase de salida, caracterizando trayectorias individuales entre transmisor y receptor. Para un canal lentamente variante en el tiempo, si transmitimos un pulso RF angosto con envolvente $p(t)$, visto como un impulso en el tiempo $t = 0$ podemos capturar la señal recibida,

$$h(\tau) = h(\tau, 0) = \sum_{i=1}^L \beta_i p(\tau - \tau_1 - \tau_i) e^{\phi_i} \quad (2.23)$$

en la que τ_1 es el TOA de la trayectoria directa entre el transmisor y receptor. Para obtener estos datos existen distintas formas de implementar una sonda en el dominio del tiempo, podemos citar dos de las más conocidas:

- Medidas usando transmisión de pulso directo.
- Medidas usando señales de espectro ensanchado.

2.7.2. Técnicas de medida en el dominio de la frecuencia [3]

En este tipo de técnica de medidas podemos obtener la respuesta en frecuencia del canal de manera directa. Si consideramos la ecuación (2.22) como el objetivo de las medidas en el dominio del tiempo, en el dominio de la frecuencia podemos medir la transformada de Fourier de esta función sobre la variable retrasada, dada por,

$$H(f, t) = \int_{-\infty}^{+\infty} h(\tau, t) e^{-j\omega\tau} d\tau = \sum_{i=1}^L \beta_i(t) e^{-j\omega\tau_i(t)} e^{-j\phi_i(t)} \quad (2.24)$$

Si el canal es lentamente variante en el tiempo, los parámetros multitrayecto del canal siguen siendo constantes durante fracciones del tiempo de coherencia del canal, y durante ese periodo podemos medir,

$$H(f) = H(f, 0) = \sum_{i=1}^L \beta_i e^{-j\omega\tau_i} e^{-j\phi_i} \quad (2.25)$$

el cual es la transformada de Fourier de la envolvente compleja de la respuesta al impulso del canal. En la práctica sin embargo los sistemas de medida son de banda limitada y lo que medimos es actualmente son las características en frecuencia enventanada del canal, definida como

$$H(f) = H(f, 0) = W(f) \sum_{i=1}^L \beta_i e^{-j\omega\tau_i} e^{-j\phi_i} \quad (2.26)$$

donde $W(f)$ representa las características en el dominio de la frecuencia del filtro RF usado en el sistema de medidas, esto puede ser realizado usando un analizador de redes como se muestra en la Figura. 2.6.

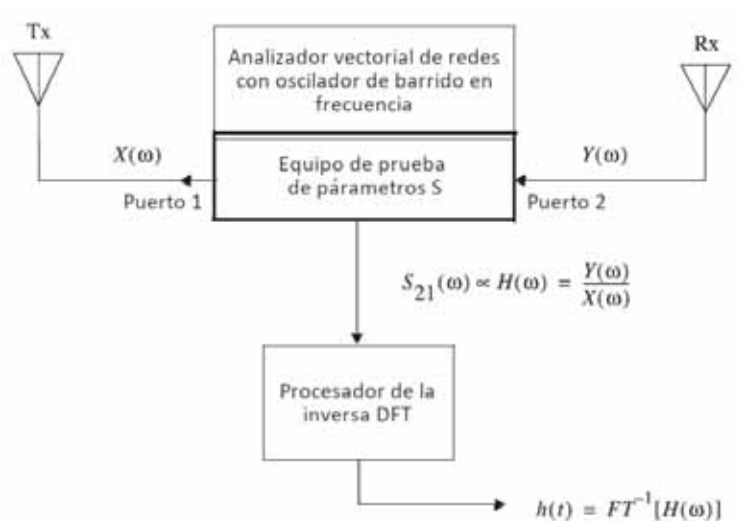


Figura 2.6: Sistema de medida en el dominio de la frecuencia. Fuente: Adaptado de [1, 3]

Medición usando un analizador de redes [18]

La Figura 2.7 muestra los principios básicos de operación de un sistema de medición en el dominio de la frecuencia usando un analizador de redes y los detalles de como la señal medida es trasladada a los perfiles del canal en el dominio del tiempo. El analizador de redes hace un barrido del canal en frecuencias discretas $f_k = f_0 + k\Delta f$, $0 \leq k < N$, con incrementos iguales de Δf , en la antena del transmisor y mide N muestras complejas de la respuesta en frecuencia $H(k)$, $0 \leq k < N$. Si asumimos que $f_0 = 0$, las muestras son

la respuesta en frecuencia compleja en banda base del canal dado por la ecuación (2.25) y tendremos:

$$H(k) = \sum_{i=1}^L \beta_i e^{-j2\pi k \Delta f \tau_i} e^{-j\phi_i}, \quad 0 \leq k < N \quad (2.27)$$

Por lo tanto, la inversa de la transformada discreta de Fourier de estas muestras es la

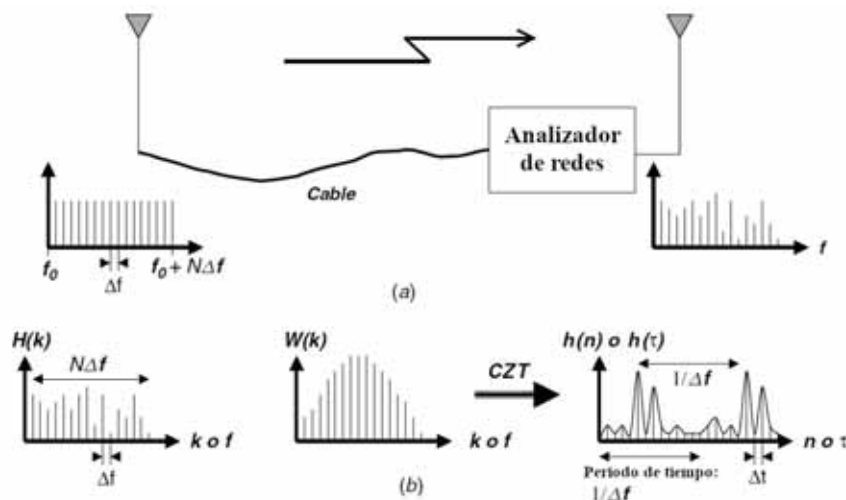


Figura 2.7: (a) Diagrama de bloques de la medición de la respuesta en frecuencia usando un analizador de redes como sistema de medidas (b) Postprocesado de los datos para el cálculos de la respuesta al impulso periódica Fuente: Adaptado de [18]

respuesta al impulso coherente en banda base o la envolvente compleja del canal de la que podemos medir los parámetros multitrayecto. La respuesta al impulso del canal determinada de la inversa de la transformada discreta de fourier (DTFT) de la secuencia $H(k)$, $0 \leq k < N$, incluyendo los efectos del filtrado en el dominio de la frecuencia $W(k)$, esta dado por:

$$h(\tau) = \frac{1}{N} \sum_{k=0}^{N-1} W(k) H(k) e^{j2\pi \tau k \Delta f} \quad (2.28)$$

Que es una función periódica de t con un periodo de $T_m = \frac{1}{\Delta f}$, que es el lapso de tiempo que uno puede medir con el sistema de medición en el dominio de la frecuencia. En la práctica se tiene que seleccionar Δf de tal manera que T_m es mucho mayor que la propagación multitrayecto del canal, $\tau_1 + \tau_L$, más otros retardos introducidos por los cables conectados a las antenas. Si Δf es demasiado pequeño, no obstante para el calculo de las muestras de la ecuación (2.28) podemos calcular solo para la región de interés. En la práctica, como se muestra en la Figura 2.7b, podemos calcular la respuesta al impulso en tiempo discreto del

canal, dado por:

$$h(n) = h(\tau) |_{\tau=n\Delta t} = \frac{1}{N} \sum_{k=0}^{N-1} W(k)H(k)e^{j2\pi n\Delta t k \Delta f}, \quad T_{inicio} \leq \tau < T_{inicio} + M\Delta\tau \quad (2.29)$$

En intervalos arbitrarios de muestreo de Δt y M muestras.

2.8. Redes de Sensores Inalámbricos (WSN) y la Tecnología LoRa

Las Redes de Sensores Inalámbricos o simplemente WSN por sus siglas en ingles, son un termino colectivo para especificar un conjunto independiente de pequeñas computadoras con el objetivo principal de medir alguna propiedad física del entorno tales como vibración, humedad, temperatura, etc. Este consiste de pocos o cientos de nodos sensores que están conectados el uno al otro vía comunicación inalámbrica. Usualmente esta tiene un nodo especial llamado gateway o estación base [5] y utiliza distintos tipos de tecnologías siendo una de las más importantes la Tecnología LoRa [7].

La Tecnología LoRa o sólo LoRa es un esquema de modulación de espectro ensanchado patentado derivado del Chirp Spread Spectrum modulation (CCS) y que intercambia la velocidad de datos por la sensibilidad dentro de un ancho de banda de canal fijo. Eso implementa una tasa de datos variable, utilizando factores de dispersión ortogonales, lo que permite que el diseñador del sistema intercambiar la velocidad de datos por el rango o la potencia, a fin de optimizar el rendimiento de la red en un ancho de banda constante [6].

2.8.1. Características de LoRa

En la modulación LoRa, la expansión del espectro se logra generando una señal chirp que varía continuamente en frecuencia. Una ventaja de este método es que las compensaciones de tiempo y frecuencia entre transmisor y receptor son equivalentes, lo que reduce en gran medida la complejidad del diseño del receptor [6].

El ancho de banda de frecuencia de este chirp es equivalente al ancho de banda espectral de la señal lo que reduce el ruido y la interferencia [7].

La tasa de bit de la modulación LoRa puede ser expresada como:

$$R_b = SF * \frac{1}{\left[\frac{2^{SF}}{BW}\right]} \text{ bit/seg} \quad (2.30)$$

Donde R_b es la tasa de bit, SF es el spreading factor o factor de ensanchamiento y BW el ancho de banda.

Ahora definimos el periodo de símbolo T_s como:

$$T_s = \frac{2^{SF}}{BW} \text{ segs} \quad (2.31)$$

Por lo tanto la tasa de símbolo R_s es el reciproco de T_s :

$$R_s = \frac{BW}{2^{SF}} \text{ simbolos/seg} \quad (2.32)$$

Finalmente la tasa de chip R_c será:

$$R_c = BW \text{ chips/seg} \quad (2.33)$$

También podemos definir la tasa nominal de bit de la señal de datos como:

$$R_b = SF * \frac{\left[\frac{4}{4+CR}\right]}{\left[\frac{2^{SF}}{BW}\right]} \quad (2.34)$$

2.8.2. Principales propiedades de la modulación LoRa [6]

Ancho de banda escalable

La modulación LoRa es escalable tanto en ancho de banda como en frecuencia. Se puede utilizar tanto para banda estrecha aplicaciones de salto de frecuencia y secuencia directa de banda ancha. A diferencia de los esquemas de modulación existentes de banda ancha o angosta, LoRa se puede adaptar fácilmente para cualquier modo de operación con sólo unos pocos cambios en el registro de configuración.

Envolvente constante / bajo consumo

Similar a FSK, LoRa es un esquema de modulación de envolvente constante, lo que significa que el mismo bajo costo y las etapas de amplificación de alta eficiencia y baja potencia se pueden reutilizar sin modificaciones. Además, debido a la ganancia de procesamiento asociada con LoRa, la potencia de salida del transmisor se puede reducir en comparación con un enlace FSK convencional manteniendo el mismo o mejor presupuesto de enlace.

Alta robustez

Debido al alto producto de tiempo de ancho de banda (BT) ($BT > 1$) y su naturaleza asincrónica, una señal LoRa es muy resistente a ambos mecanismos de interferencia dentro y fuera de banda. Dado que el período del símbolo LoRa puede ser más largo que el ráfaga típica de corta duración de los sistemas de salto rápido (FHSS), proporciona una excelente inmunidad a los mecanismos de pulsos de interferencia AM; cifras típicas de selectividad fuera de canal del receptor de 90 dB y co-canal se puede obtener un rechazo superior a 20

dB. Esto se compara con típicamente 50 dB para adyacentes y rechazo de canal alternativo y rechazo de co-canal de -6 dB para la modulación FSK.

Resistente al desvanecimiento y al efecto multitrayecto

El pulso de chirrido (chirp) es de banda relativamente ancha y, por lo tanto, LoRa ofrece inmunidad a múltiples trayectos y desvanecimiento, ideal para uso en entornos urbanos y suburbanos, donde dominan ambos mecanismos.

Resistente al Doppler

El desfase Doppler provoca un pequeño cambio de frecuencia en el pulso LoRa que introduce un cambio relativamente insignificante en el eje de tiempo de la señal en banda base. Esta tolerancia de desplazamiento de frecuencia mitiga el requisito para fuentes de reloj de referencia de tolerancia estricta. LoRa es ideal para enlaces de comunicaciones de datos móviles como sistemas inalámbricos de control de la presión de los neumáticos, aplicaciones de conducción como cabina de peaje y lectores de etiqueta móvil y comunicaciones en tierra para infraestructura ferroviaria.

Capacidad de largo alcance

Para una potencia de salida y un rendimiento fijos, el balance de enlace de LoRa excede al de FSK convencional. Cuando se toma en conjunto con la robustez probada a los mecanismos de interferencia y desvanecimiento, esta mejora en el presupuesto de enlace se puede traducir fácilmente a x4 y más allá de la mejora en el rango.

Alcance / Localización

Una propiedad inherente de LoRa es la capacidad de discriminar linealmente entre errores de frecuencia y tiempo. LoRa es la modulación ideal para aplicaciones de radar y, por lo tanto, es ideal para el alcance y la localización, aplicaciones tales como servicios de ubicación en tiempo real.

2.8.3. Comparación de LoRa con varias tecnologías existentes [7]

LoRa versus Red Celular

Las redes celulares tradicionales consistentes de GSM, 2G, 3G, 4G consumen demasiada potencia y no son una buena opción cuando una pequeña cantidad de datos que son transmitidos con menor frecuencia. La tecnología LoRa por otra parte tiene una tasa de consumo muy bajo y es muy adecuada cuando una pequeña cantidad de datos es transmitida sobre largas distancias.

LoRa versus LAN

LAN o Local Area Network es un estándar ampliamente utilizado, este es usado dentro de un área limitado tales como edificios, escuelas, oficinas, etc. LAN puede ser cableado o inalámbrico. Ethernet y Wi-Fi son las tecnologías más usadas en LAN. El Wi-Fi es la tecnología inalámbrica usada en LAN, esta usualmente esta confinada dentro de un pequeño área y puede extenderse sobre un rango limitado de 1 Km de radio. Por otra parte LoRa proporciona un largo alcance. Un simple gateway LoRa puede cubrir un área de $100Km^2$. La calidad de servicio de Wi-Fi es muy pobre mientras que LoRa tiene una calidad de servicio razonable. Wi-Fi es inalámbrico y así la seguridad es baja debido a una encriptado muy pobre mientras que LoRa ofrece doble encriptado AES por lo que es altamente seguro. LoRa esta basado en CCS por lo que es altamente resistente a multitrayecto y desvanecimiento.

LoRa versus ZigBee

ZigBee esta basado en un protocolo de comunicación de alto nivel usado para crear redes de area personal. Este consiste de pequeños radios digitales de bajo consumo de potencia. Estos son más adecuados para proyectos a pequeña escala que necesitan transferir datos en distancias pequeñas en el rango de 10-100 metros. Estos están basados en redes malla que transmiten datos sobre largas distancias por la trasmisión de datos a muchos dispositivos intermedios por otra parte LoRa esta basado en topologia estrella que elimina la transmisión de datos a dispositivos intermedios que reduce la potencia utilizada.

Capítulo 3

Metodología

En esta capítulo se describirá la metodología utilizada para el modelamiento de canal de comunicación inalámbrico de la zona Arqueológica de Rumiwasi para el cual dividiremos esta en dos secciones:

- Caracterización de la propagación a gran escala.
- Cálculo del tiempo de retardo y modelamiento ARMA de los muros incas ubicados en esta zona

3.1. Caracterización de la propagación a gran escala del canal de comunicación inalámbrico

En esta sección caracterizaremos la intensidad de potencia recibida (RSSI) en función de la distancia utilizando tecnología LoRa, en la trayectorias: con línea de vista (LOS), sin línea de vista (NLOS) y línea de vista con obstrucción (LOBS) diferenciándose esta última de la anterior por la cantidad de obstrucciones que se encuentran en la trayectoria. El RSSI se utilizó como variable dependiente en vez de la pérdida de trayecto en razón de que esta provee la información utilizada para la cobertura tanto como la localización sin el uso del GPS [8, 9, 10].

Las pruebas se realizaron en la zona arqueológica de Rumiwasi de las que detallaremos los procesos que se realizaron con mayor detalle las siguientes secciones para obtener los modelos de RSSI descritos en las Ecuaciones 2.8 y 2.9.

3.1.1. Pruebas preliminares y elección de equipos utilizados en la sonda de canal

Para caracterizar el canal fue necesario hacer distintas pruebas con distintos equipos, a continuación se muestran un breve resumen de las pruebas preliminares y variantes de la sonda de canal utilizados.

Primera propuesta de la sonda de canal

La primera sonda de canal esta conformada en el transmisor por una laptop conectada al USRP N210 que transmitía usando una antena omnidireccional para transmitir un pulso continuo a 915 MHz usando el software GNU Radio Companion. El receptor estaba conformado por la antena direccional de tipo panel PA-727-8D8NH sujetado a un tripode al que se conecto el equipo multifunción Field Fox Handheld Analyzer configurado como analizador de espectro en el que se recibió el pulso transmitido como se muestra en la Figura 3.2 y se anoto manualmente el valor pico de cada pulso para luego obtener los modelos matemáticos de los entornos en lo que se realizo las pruebas y hacer la validación de cada modelo. La implementación de la sonda de canal se muestra en la Figura 3.1. A continuación se muestra el Cuadro 3.1 donde se muestra un resumen de las ventajas y desventajas de esta primera propuesta.



Figura 3.1: Primera propuesta de la sonda de canal

Variante de la primera propuesta de la primera sonda de canal

Uno de las propuestas para solucionar algunas de las desventajas fue cambiar la antena transmisora por la antena logarítmica periódica HyperLOG 6080 como se muestra en la Figura 3.3. En el cuadro 3.2 se muestra las ventajas y desventajas para esta variante. La implementación se muestra en la Figura 3.4.

Segunda propuesta de la sonda de canal

Debido a la poca portabilidad de la sonda anterior y la necesidad de una fuente de alimentación, una mejor opción fue usar el nodo sensor LoRa Mote RN2903 como transmisor conectado a una laptop y configurado como un generador de pulso a 915MHz utilizando el

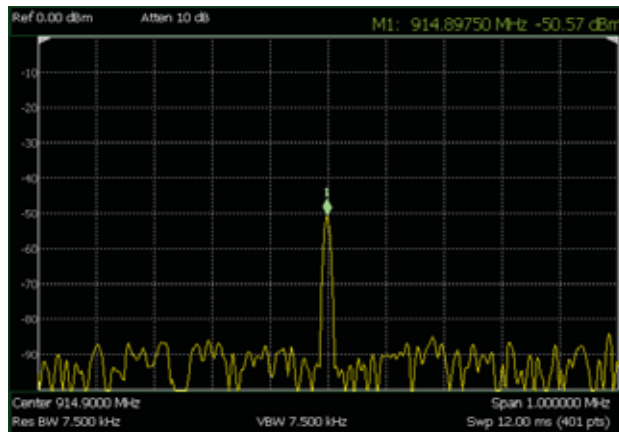


Figura 3.2: Pulso recibido usando en el Field Fox Handheld Analyzer

Ventajas	<ul style="list-style-type: none"> La configuración del USRP N210 es relativamente simple usando el software GNU Radio Companion.
Desventajas	<ul style="list-style-type: none"> La potencia transmitida por el USRP no era constante, cada prueba realizada en la mismas condiciones arrojaba distintos valores de potencias recibidas. Los valores medidos tienen que adquirirse de manera manual. Muy poco portable y con dificultad para poner en operación (necesidad de tener alimentación de energía externa). Al validar los parámetros de los modelos estos eran erróneos para los valores ya conocidos del exponente de pérdida de trayecto para entornos comúnmente encontrados [1].

Cuadro 3.1: Ventajas y desventajas de la primera sonda de canal

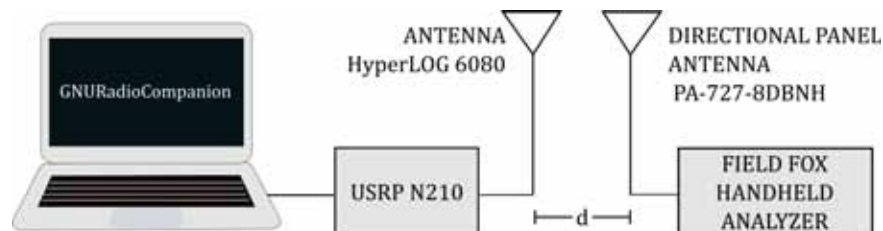


Figura 3.3: Diagrama de bloques de la variante de la primera sonda de canal



Figura 3.4: Variante de la primera sonda de canal

software del fabricante LoRaDevUtility. El diagrama de bloques se muestra en la Figura 3.5. La implementación de la sonda de canal se muestra en la Figura 3.6. Las ventajas y desventajas de esta sonda se muestra en el Cuadro 3.3.

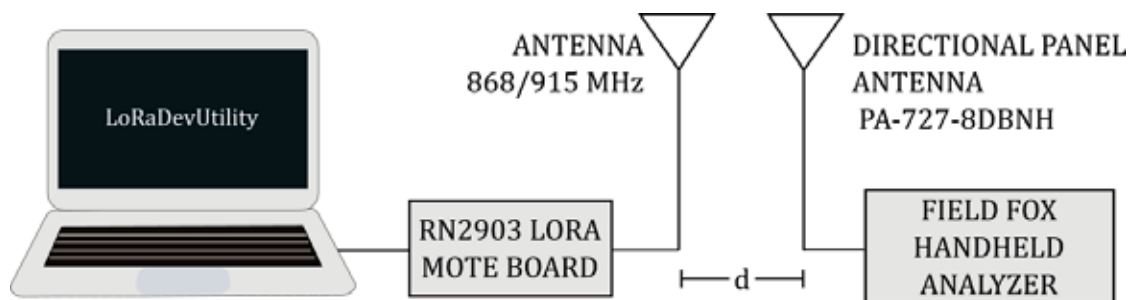


Figura 3.5: Diagrama de bloques de la variante de la segunda propuesta de la sonda de canal

Ventajas	<ul style="list-style-type: none"> ■ La configuración del USRP N210 es relativamente simple usando el software GNU Radio Companion. ■ Se soluciono el problema de los errores en la validación de datos ya que se obtuvieron los parámetros de los modelos muy cercanos a los valores ya calculados para entornos comúnmente encontrados [1].
Desventajas	<ul style="list-style-type: none"> ■ La potencia transmitida por el USRP no es constante, cada prueba realizada en la mismas condiciones arrojaba distintos valores de potencias recibidas. ■ Los valores medidos tienen que adquirirse de manera manual. ■ Muy poco portable y con dificultad para poner en operación (necesidad de tener alimentación de energía externa).

Cuadro 3.2: Ventajas y desventajas de la variante de la sonda de canal



Figura 3.6: Implementación de la segunda propuesta de la sonda de canal

Tercera propuesta de la sonda de canal

La tercera propuesta surgió debido a la necesidad de almacenar los datos de manera mas simple y optima, ademas también era necesario transmitir a la frecuencia de 920MHz

Ventajas	<ul style="list-style-type: none"> ■ El transmisor ahora es portable y no necesita una alimentación de corriente alterna sino simplemente se conecta a la laptop. ■ El transmisor al usar tecnología LoRa posee un gran alcance o mayor cobertura (teóricamente 15km). ■ Se encontró mejor aproximación de los parámetros de los modelos ya calculados para entornos comúnmente encontrados [1].
Desventajas	<ul style="list-style-type: none"> ■ Los valores medidos tienen que adquirirse de manera manual. ■ La configuración del hardware no es flexible, además un cambio en el firmware no tiene efecto en el funcionamiento del dispositivo.

Cuadro 3.3: Ventajas y desventajas de la segunda propuesta de la sonda de canal

debido a que esta frecuencia pertenece a la banda ISM ¹. Esta propuesta tuvo dos opciones de implementación. La implementación de la primera opción de esta sonda de canal contó con el uso de la tarjeta LoPy4 con su respectivo circuito de alimentación tanto para el nodo sensor y gateway como se muestra en la Figura 3.7. Un breve resumen de sus ventajas y desventajas se muestran en el Cuadro 3.4.

Ventajas	<ul style="list-style-type: none"> ■ El transmisor y receptor son portables. ■ Se encontró mejor aproximación de los parámetros de los modelos ya calculados para entornos comúnmente encontrados [1].
Desventajas	<ul style="list-style-type: none"> ■ Es necesario tener un circuito de alimentación para cada tarjeta. ■ El transmisor a pesar de tener tecnología LoRa tiene una rápida atenuación en entornos con obstrucciones.

Cuadro 3.4: Ventajas y desventajas de la primera opción de la tercera propuesta de la sonda de canal

¹<https://www.osiptel.gob.pe/media/g4zh4gcn/dt-43-estado-espectro-radioelectrico-peru.pdf>

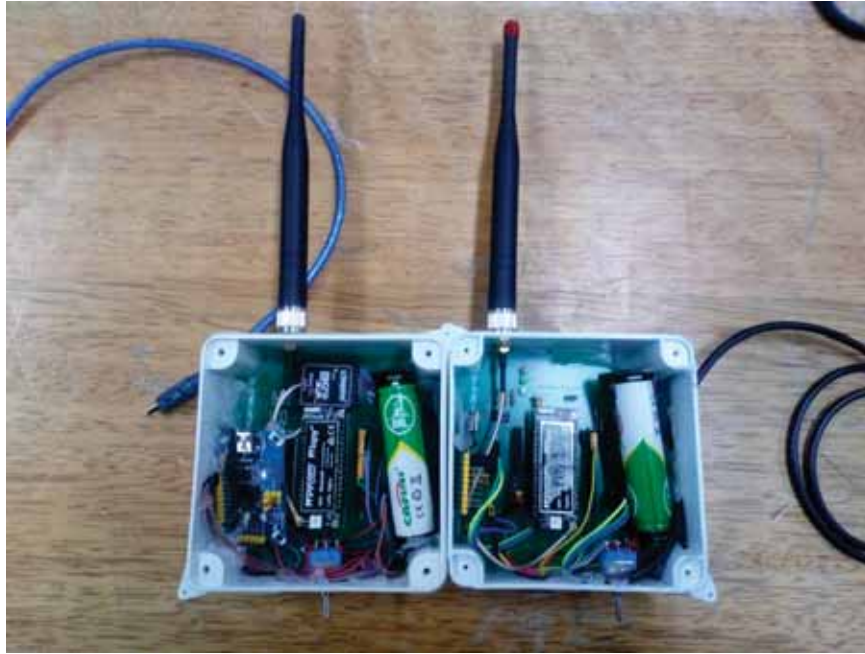


Figura 3.7: Primera opción de la tercera propuesta de la sonda de canal

Debido a que la anterior opción tenía desventajas una segunda opción de esta sonda de canal fue necesaria y constó de dos tarjetas TTGO T-beam en lugar del LoPy4 como se muestra en el diagrama de bloques Figura 3.8. El pulso transmitido por la tarjeta configurada como nodo sensor se muestra en la Figura 3.9. Las ventajas y desventajas de esta implementación se muestran en el Cuadro 3.5.

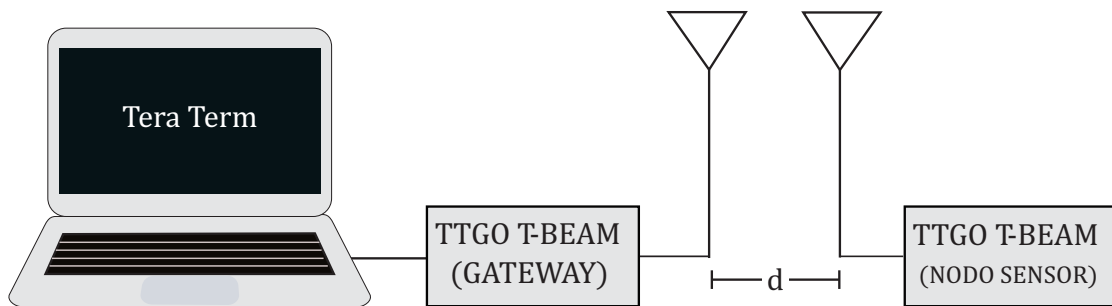


Figura 3.8: Diagrama de bloques del sistema de medidas de la segunda opción de la Tercera propuesta de la sonda de canal.

Las pruebas preliminares para la segunda opción de la tercera propuesta de sonda de canal se realizaron con el nodo transmisor sin estar en un trípode como muestra la Figura 3.10 y se usara para ver la cobertura de la tarjeta ttgo t-beam y ver la precisión relativa del modulo GPS de la tarjeta. Los datos almacenados en Tera Term se guardaron en archivos con formato csv que luego fueron importados en Google Earth Pro donde se les asigno un

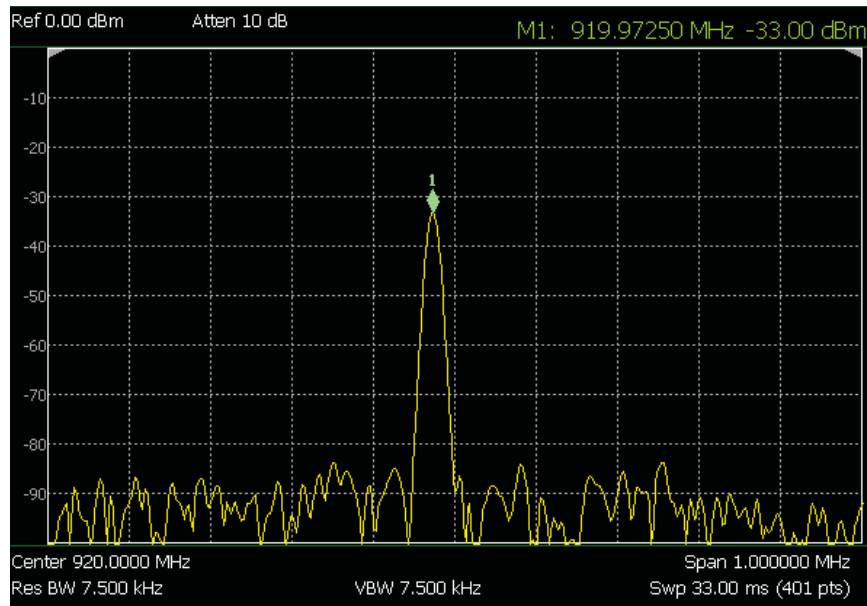


Figura 3.9: Pulso transmitido por el TTGO t-beam medido usando el Fieldfox Handheld Analyzer N9914A

patrón de colores al RSSI (Verde: mayor intensidad y Rojo: menor intensidad) y se grafico las posiciones con sus respectivo valor de RSSI.

Estas pruebas preliminares se hicieron para cuatro recorridos los cuales se muestran en las Figuras 3.11, 3.12, 3.13 y 3.14. Podemos concluir respecto a estas pruebas preliminares con referencia las posiciones reales que el GPS tiene muy buena precisión esto en razón de la comparativo del camino real recorrido y el mostrado en el programa Google Earth Pro, también que el nodo tiene la intensidad suficiente para modelar la parte principal de los restos arqueológicos de Rumiwasi (aprox. 250m de radio).

3.1.2. Características de las antenas utilizadas en las sondas de canal

Las características de las antenas utilizadas se muestran en el Cuadro 3.6.

3.1.3. Características y comparación de todas las tarjetas utilizadas en las sondas de canal

Las características de las tres tarjetas utilizadas en las sondas se muestran en el Cuadro 3.7. Un breve resumen de las ventajas y desventajas de estas, se muestran en el Cuadro 3.8.



Figura 3.12: Segundo recorrido Fuente: Google Earth Pro



Figura 3.13: Tercer recorrido Fuente: Google Earth Pro






Figura 3.14: Cuarto recorrido Fuente: Google Earth Pro

Ventajas	<ul style="list-style-type: none"> ■ El transmisor y receptor son portables. ■ Se encontró mejor aproximación de los parámetros de los modelos ya calculados para entornos comúnmente encontrados [1]. ■ Buena cobertura en entornos con obstrucciones. ■ No es necesario tener un circuito de alimentación para cada tarjeta ya que esta tarjeta posee un portapilas. ■ Posee un modulo GPS que nos servirá para tener la posición de referencia.
Desventajas	<ul style="list-style-type: none"> ■ Ninguna para este trabajo específico.

Cuadro 3.5: Ventajas y desventajas de la segunda opción de la tercera propuesta de la sonda de canal

3.1.4. Configuración de las tarjetas TTGO

La tarjeta TTGO T-beam permite usar el IDE de Arduino para realizar la codificación con la que configura el equipo por lo que para este trabajo se utilizó la versión 1.8.13 en el

Modelo de antena	HyperLOG 6080	L-com HG908P-NF	CDSNET TX915-JK-11
Imagen de la antena			
Características	<ul style="list-style-type: none"> ▪ Diseño:logarítmica-periódica. ▪ Ganancia: 5dBi. ▪ Rango de frecuencia: 680MHz - 8GHz. ▪ Conexión RF: puerto SMA. ▪ Impedancia nominal: 50 Ohms. ▪ Polarización: lineal vertical/horizontal. ▪ Return loss: mejor que -10dB 	<ul style="list-style-type: none"> ▪ Diseño: plana tipo parche. ▪ Ganancia: 8dBi. ▪ Rango de frecuencia: 902-928MHz ▪ Conexión RF: Conector N-Hembra. ▪ Impedancia nominal: 50 Ohms. ▪ Polarización: vertical/horizontal. 	<ul style="list-style-type: none"> ▪ Diseño: Omnidireccional- ▪ Ganancia: 2.5dBi. ▪ Frecuencia de trabajo: 915MHz ▪ Conexión RF: SMA-J.




Cuadro 3.6: Características de las antenas utilizadas

que se instalo previamente todas las librerías necesarias para trabajar con las tarjetas que usen como procesador el dispositivo ESP32.

Configuración del transmisor (Nodo Sensor)

Este código esta basado en el código del transmisor LoRa proporcionado en GitHub por los desarrolladores del MIT para distintos radios LoRa. ², al cual se le realizó distintas modificaciones que se describen a continuación:

²https://github.com/LilyGO/TTGO-T-Beam/blob/master/OLED_LoRa_Sender/OLED_LoRa_Sender.ino

Nombre de la tarjeta	TTGO T-Beam	LoRa Mote RN2903	LoPy4
Imagen del modulo			
Características principales	<ul style="list-style-type: none"> ■ Procesador: ESP32 dual con Wifi y bluetooth. ■ Voltaje de funcionamiento modulo LoRa: 1.8-3.7 V. ■ Frecuencia de funcionamiento: 915MHz ■ Sensibilidad de recepcion: -136dBm ■ Corriente de transmisión: 120mA @ +20dBm ■ Temperatura de funcionamiento:- 40 °C- +85°C ■ Modulo GPS NEO-6M ■ Modulo LoRa: SemTech SX1276 	<ul style="list-style-type: none"> ■ Procesador: PIC18LF45K50 8-bit MCU ■ Voltaje de funcionamiento modulo LoRa: 2.1-3.6 V. ■ Frecuencia de funcionamiento: 868MHz ■ Sensibilidad: -146dBm ■ Potencia de transmisión: 18.5dBm ■ Backlight LCD Display; 1.2 Inch 128 X 32 Dot Matrix ■ Temperatura de funcionamiento:- 40 °C- +85°C ■ Modulo LoRa: Microchip RN2903 	<ul style="list-style-type: none"> ■ Procesador: ESP32 dual con Wifi ■ Voltaje de funcionamiento modulo LoRa: 1.8-3.7 V. ■ Frecuencia de funcionamiento: 915MHz ■ Sensibilidad de recepcion: -136dBm ■ Potencia de transmisión: +20dBm ■ Temperatura de funcionamiento:- 40 °C- +85°C ■ Modulo LoRa: SemTech SX1276

Cuadro 3.7: Comparación de tarjetas LoRa

Nombre de la tarjeta	TTGO T-Beam	LoRa Mote RN2903	LoPy4
Ventajas	<ul style="list-style-type: none"> ▪ Buen procesador y modulos GPS,Wifi y bluetooth. ▪ Fácil programación y configuración de hardware gracias al IDE de arduino. ▪ Portapilas lo que lo hace util para aplicaciones remotas. 	<ul style="list-style-type: none"> ▪ Fácil uso. ▪ Viene con software que facilita las tareas simples. ▪ Portapilas lo que lo hace util para aplicaciones remotas. 	<ul style="list-style-type: none"> ▪ Programación relativamente fácil usando Atom o Visual Studio Code. ▪ Capacidad para manejar mayor cantidad de tecnologías como SigFox.
Desventajas	<ul style="list-style-type: none"> ▪ Menor capacidad de tecnologías de transmisión de datos. 	<ul style="list-style-type: none"> ▪ Poca flexibilidad en el cambio de configuración. ▪ Un cambio de firmware no cambia la configuración del hardware. 	<ul style="list-style-type: none"> ▪ No posee portapilas y necesita un circuito adicional para su alimentación. ▪ No posee modulo GPS.

Cuadro 3.8: Resumen de las ventajas y desventajas de las tarjetas LoRa comúnmente utilizadas

- Declaración de librerías. En esta sección del código utilizamos las librerías por defecto para la configuración de la tarjeta añadiéndole la librería **TinyGPS++.h** que nos provee de métodos de fácil uso para la extracción de datos tales posición, fecha, tiempo, altitud y velocidad de dispositivos de consumo GPS como se muestra en el Listado de Códigos 3.1. Esto se realizó debido a que para tener las referencias de distancia en la zona arqueológica de Rumiwasi es necesario usar el modulo GPS de la tarjeta.

```

1 #include <SPI.h> //Libreria que permite la comunicacion con
   la interfaz de perifericos en serie
2 #include <LoRa.h> //Libreria LoRa para enviar y recibir datos
   usando radios LoRa
3 #include <Wire.h> //Libreria que permite comunicarse con I2C
4 #include <TinyGPS++.h> //Libreria para uso del modulo GPS y sus
   valores

```

```

5 TinyGPSPlus gps;          // parte del codigo para usar tiny GPS++
   creando un objeto gps

```

Listado de codigos 3.1: Declaración de librerías

- Configuración de spreading factor y la frecuencia de transmisión. Este tipo de métodos se encuentran en la librería **LoRa.h** configuramos el spreading factor a 7 por ser un valor además de ser utilizado en trabajos simulares nos permite obtener una muestra por segundo luego configuramos la frecuencia de transmisión a 920MHz (Figura 3.9) ya que es parte de la banda ISM de acuerdo a la asignación de frecuencias realizado por el Estado Peruano. Esto se muestra en el Listado de Códigos 3.2.

```

1 LoRa.setSpreadingFactor(7); //Configuracion de valor de
   Spreading factor
2 LoRa.setFrequency(920E6);   //Frecuencia a la cual se
   transmite el pulso modulado

```

Listado de codigos 3.2: Configuración del Spreading factor y frecuencia de transmisión

- Declaración de variables auxiliares y creación del paquete enviado por el transmisor. En esta sección se creó variables auxiliares para obtener los datos del GPS y luego los enviamos creando un paquete que inicia con la línea de código **LoRa.beginPacket()**: y termina con la línea de código **LoRa.endPacket()**:. Esto se muestra en el Listado de Códigos 3.3.

```

1 LoRa.setSpreadingFactor(7); // configuracion del valor del
   spreading factor
2 LoRa.setFrequency(920E6);   // Frecuencia a la cual se recibe
   el pulso modulado
3 LoRa.receive();

```

Listado de codigos 3.3: Configuración del Spreading factor y frecuencia de transmisión

Configuración del receptor (Gateway)

Este código está basado en el código del receptor LoRa proporcionado en GitHub por los desarrolladores del MIT para distintos radios LoRa ³, al cual se le realizó distintas modificaciones que se describen a continuación:

- Impresión del paquete enviado y la recepción del RSSI. Por defecto podemos imprimir el RSSI usando el comando **Serial.println(rssi)** que es necesario para el modelamiento del canal y el comando **Serial.println(packet)** para mostrar el paquete enviado. Esto se muestra en el Listado de Códigos 3.5.

³<https://github.com/sandeepmistry/arduino-LoRa/blob/master/examples/LoRaReceiver/LoRaReceiver.ino>


```

1 Serial.println(rssi); //impresion del valor de RSSI recibido
2 Serial.println(packet); //impresion del paquete enviado

```

Listado de codigos 3.4: Impresión del paquete enviado y la recepción del RSSI

- Configuración de spreading factor y la frecuencia de transmisión. Para que pueda existir comunicación entre ambas tarjetas estas deben de tener la misma configuración por lo que tendrán los mismos parámetros. Esto se muestra en el Listado de Códigos ??.

```

1 Serial.println(rssi); //impresion del valor de RSSI recibido
2 Serial.println(packet); //impresion del paquete enviado

```

Listado de codigos 3.5: Impresión del paquete enviado y la recepción del RSSI

3.2. Metodología aplicada para caracterizar el RSSI

El diagrama de bloque del sistema de medidas se muestra en la Figura 3.8. Este consiste de dos tarjetas TTGO T-beam colocados en trípodes a una altura de $h = 1,55m$, el primero fue configurado como nodo sensor el cual enviaba la información del modulo GPS de la tarjeta y el valor del RSSI en un paquete de datos para las trayectorias NLOS y LOBS; la segunda tarjeta fue configurada como un nodo gateway ubicado en una posición fija (posición 1 Figura 3.17) conectada a una laptop la cual almacenaba el paquete recibido (aproximadamente 100 por posición) usando el software Tera Term. Ambos nodos fueron configurados a una frecuencia de 920 MHz (Figura 3.9), obteniendo un total de 2091 muestras.

Los modelos usados fueron el modelo log-normal de una sola pendiente y el modelo log-normal de doble pendiente (Ecuaciones (2.8) y (2.9)). El primero se eligió debido es el modelo más simple más comúnmente usado en este tipo de modelamiento y el segundo debido a que en este caso específico de la zona arqueológica de Rumiwasi hay espacios en los que distinguen dos regiones con obstrucciones completamente diferenciadas: espacio abierto sin restos arqueológicos y espacio abierto con restos arqueológicos. El sistema implementado se muestra en la Figura 3.2 y algunas posiciones de prueba en la Figura 3.16.

Podemos distinguir dos trayectorias principales NLOS (posiciones de color amarillo) y LOBS (posiciones color rojo) como se muestran en la Figura 3.17. Dentro de la trayectoria NLOS podríamos aproximar parte de estas como LOS (posiciones 1-6) debido a las condiciones del entorno (poca vegetación y menor cantidad de obstrucciones).

Los códigos para configurar el modulo TTGO T-beam se muestran en el Apéndice B.1., estos fueron implementados en el IDE de Arduino versión 1.8.13.

Los parámetros de los modelos se calcularon usando un modelo de regresión simple descrita en [31, 32], se implementaron los programas usando Matlab versión R2018b para cálculo de los parámetros para las trayectorias principales NLOS y LOBS, el código de los programas se muestran en el Apéndice B.2.1 y B.2.2.



Figura 3.15: Sistema de adquisición de datos o sonda de canal



Figura 3.16: Algunas posiciones de prueba

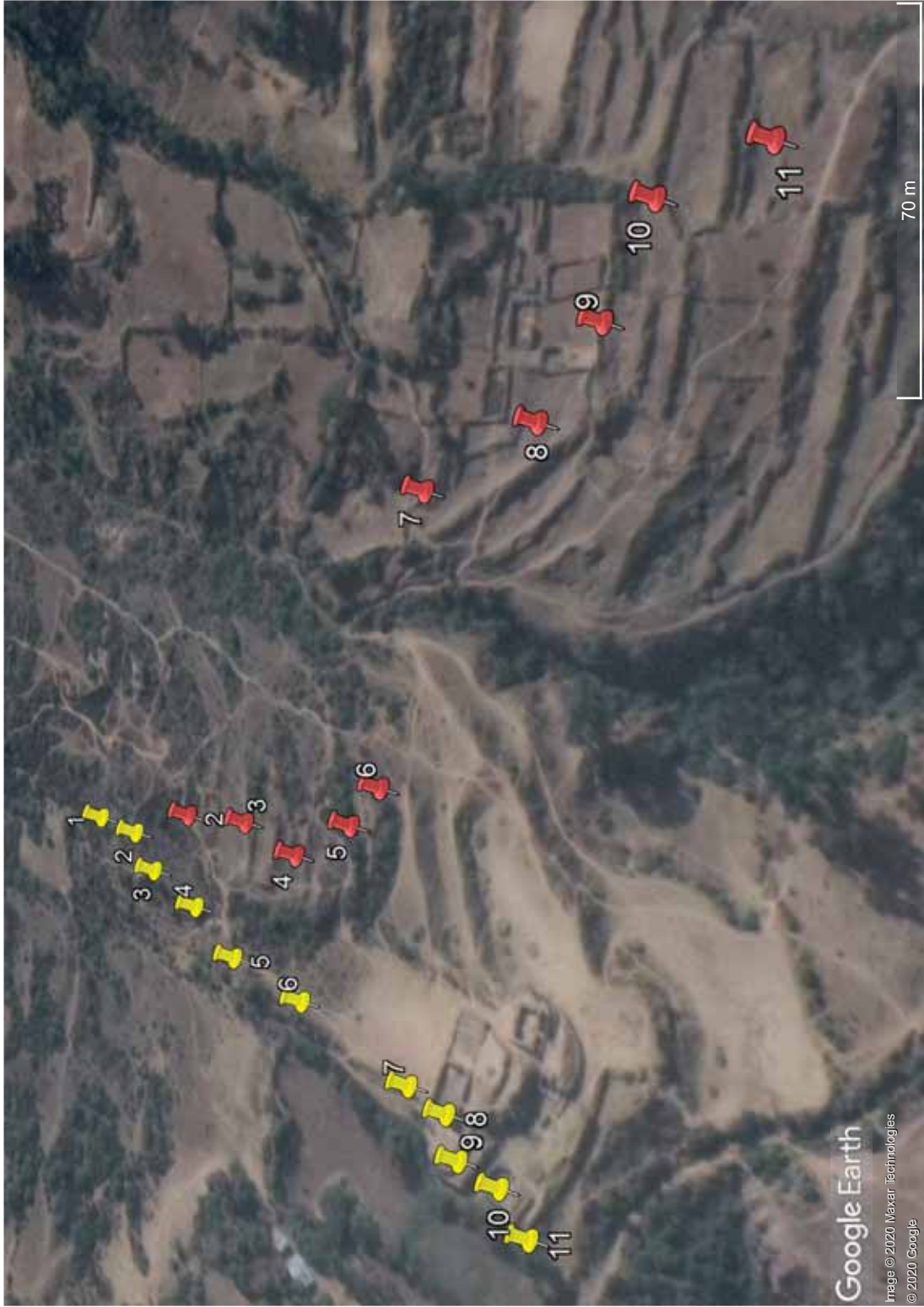


Figura 3.17: Trayectoria LOS y NLOS (Posiciones amarillas) y trayectoria LOBS (Posiciones rojas) Fuente: Google Earth Pro

3.3. Metodología para la caracterización del tiempo de retardo RMS y modelamiento ARMA de la propagación en muros inca

Para caracterizar el tiempo de retardo utilizamos la metodología propuesta en [12], esto se realizó usando dos antenas tipo parche HG908LP-NF y el Fieldfox Handheld Analyzer N9914A configurado en el modo analizador de redes vectorial (VNA) para almacenar el parámetro S_{21} en el rango de frecuencia de 800MHz a 1GHz, el diagrama de bloques se muestra en la Figura 3.18. Estas medidas se realizaron para los muros inca más represen-

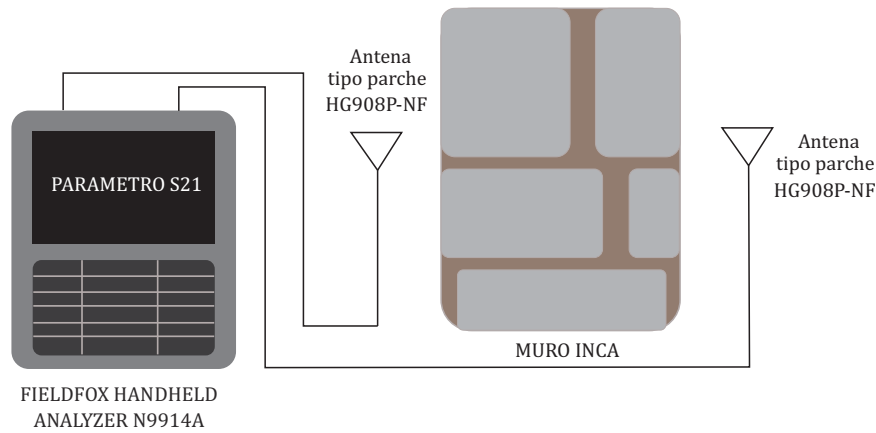


Figura 3.18: Diagrama de bloques del sistema de medida Fuente: elaboración propia

tativos. Encontrando que en la zona arqueológica de Rumiwasi existen dos tipos, ambos de 1m de espesor, diferente material y diferente geometría como se muestra en la Figuras 3.3 y 3.3. Para cada uno de los muros se almaceno 4 perfiles del parámetro S_{21} a una altura de 99cm, para el primer muro a una distancia de 30 y 60cm del muro a la antena y para el segundo a solo 60cm del muro a la antena debido a la falta de accesibilidad del trípode, teniendo estas medidas almacenadas se promediaron para realizar los respectivos análisis.

Calculo del tiempo de retardo RMS

Para calcular el tiempo de retardo RMS usamos las relaciones descritas [12]: considerando $H(k) = S_{21}$ debido a que usamos la tecnica en dominio de la frecuencia descrita en la sección 2.7.2, la respuesta en el dominio del tiempo estará dado por:

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} H(k)W(k)e^{-\frac{j2\pi kn}{N}} \quad (3.1)$$



Figura 3.19: Muro de prueba 1



Figura 3.20: Muro de prueba 2

es una función periódica de periodo $T = \frac{1}{\Delta f}$, $W(k)$ es un filtro de enventanado Hamming, N es el número de muestras.

Para un canal con dispersión no correlacionada estacionaria de sentido amplio o por sus siglas en inglés (WSSUS), El perfil de retardo de potencia (PDP) puede ser calculado usando las siguientes relaciones:

$$PDP(t) = P(t) = |h(t)|^2 = \left| \sum_{n=0}^{N-1} h(n)\delta(t - n\Delta t) \right|^2 \quad (3.2)$$

El tiempo de retardo RMS puede ser calculado usando las siguiente relaciones:

$$\tau = \sqrt{\langle \tau^2 \rangle - \langle \tau \rangle^2} \quad (3.3)$$

Donde:

$$\langle \tau \rangle = \frac{\sum P(\tau)\tau}{\sum P(\tau)} \quad (3.4)$$

conocido como el exceso medio de retardo y

$$\langle \tau^2 \rangle = \frac{\sum P(\tau)\tau^2}{\sum P(\tau)} \quad (3.5)$$

conocido como exceso de retardo medio cuadrático, estas ecuaciones son aproximaciones en tiempo discreto de las ecuaciones (2.14) y (2.15).

Esto se implementó en Matlab versión R2018b en el programa retardo2.m que se describe en el Apéndice B.2.3. teniendo en cuenta un umbral de -20dBm para analizar los datos.

Modelamiento del proceso ARMA y el cálculo de sus parámetros

Para modelar el proceso ARMA de cada muro se utilizo el promedio de los parámetros S_{21} y utilizando la libreria statsmodels de Python [33], esto debido a que Python ofrece mucha flexibilidad y rapidez para hacer cálculos ademas statsmodels es una librería validada con paquetes estadísticos como R, Stata o SAS y ampliamente usada para modelos estadísticos. En esta librería el proceso ARMA de orden (p,q) se describe de la siguiente manera:

$$H(z) = \frac{G(1 + MA_1 * z + MA_2 * z^2 + \dots + MA_q * z^q)}{1 - AR_1 * z - AR_2 * z^2 + \dots + AR_p * z^p} \quad (3.6)$$

Donde MA_1, \dots, MA_q son los parámetros del proceso de media móvil, AR_1, \dots, AR_p son los parámetros del proceso auto-regresivo, ambos calculados usando el estimador de máxima verosimilitud a través del filtro de Kalman.

Todo el código se implementó usando el entorno Jupyter Notebook del IDE Anaconda Navigator Version 1.10.0 que es utiliza como motor de procesamiento python version 3, los notebooks se describen en el Apendice B.2.4.

Cabe destacar que el modelamiento ARMA es utilizado ampliamente en la simulación lo que provee un gran valor para trabajos futuros,

Capítulo 4

Resultados y discusión

4.1. Caracterización del RSSI

4.1.1. Modelos del RSSI

4.1.2. Primera campaña de medidas

En la Figura 4.1 se muestra la primera campaña de medidas usando la sonda de canal en la que se propuso dos tramos el primero como línea de vista (LOS) de punto inicial hasta el punto intermedio (ver Fig. 4.1) y el segundo tramo como la trayectoria sin línea de vista (NLOS) desde el punto intermedio hasta el punto final (ver Fig. 4.1), estas pruebas se realizaron cada 3m para la trayectoria LOS teniendo un promedio de 150 muestras por punto y para 5 puntos a diferentes distancias en el tramo NLOS, distancias que fueron medidas usando la herramienta regla del Google Earth Pro. Esta prueba se realizó para encontrar dos mejores trayectos que sean los más representativos de esta zona con el modelo log-normal de una sola pendiente (Ecuación (2.8)) como primera aproximación. Los gráficos de la comparación de los modelos con los datos medidos experimentalmente se muestran en las Figuras 4.2 y 4.3; los parámetros del modelo de una sola pendiente para ambas trayectorias se muestran en el Cuadro 4.1, un pequeño análisis de este cuadro nos revela que para la trayectoria LOS la pérdida de trayecto es muy pequeña $n = 0,0477$ esto debido a la gran cobertura de LoRa en terrenos planos. Para la trayectoria LOBS tenemos un $n = 10,9106$ que refleja una alta atenuación de la señal transmitida además de una menor aproximación del modelo ajustado por lo que será necesario hacer un reajuste en las pruebas para tener mejores resultados.

Trayectoria NLOS

Para esta trayectoria podemos ver que en la Figura 4.4 el modelo log-normal de doble pendiente (Ecuación (2.9)) se aproxima mejor a los valores medidos esto debido a que en esta trayectoria tenemos dos regiones completamente diferenciadas:



Figura 4.1: Primera campaña de medidas

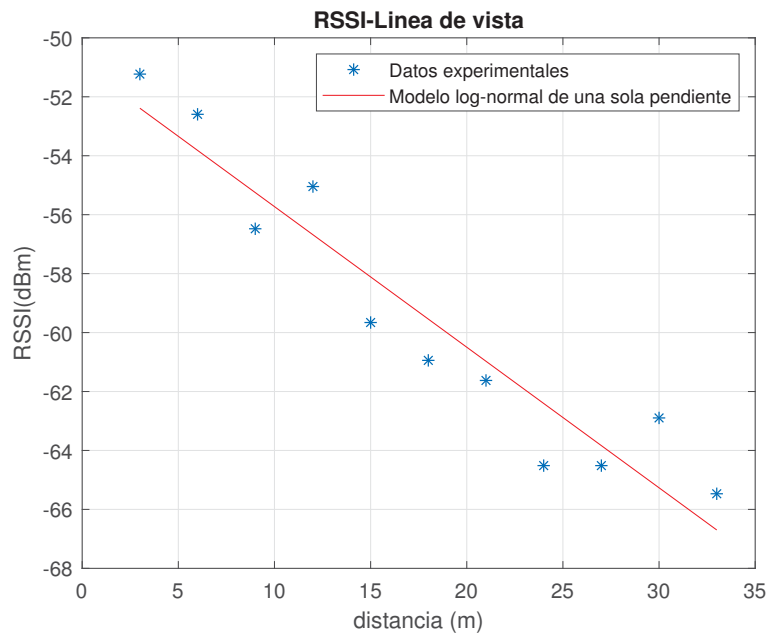


Figura 4.2: Modelo log-normal de una sola pendiente para la trayectoria LOS de la primera campaña de medidas

- Espacio abierto sin restos arqueológicos (posiciones de color amarillo del 1-7, ver Figura 3.17) siendo esta aproximadamente la trayectoria LOS debido a la mínima cantidad de obstrucciones que posee.

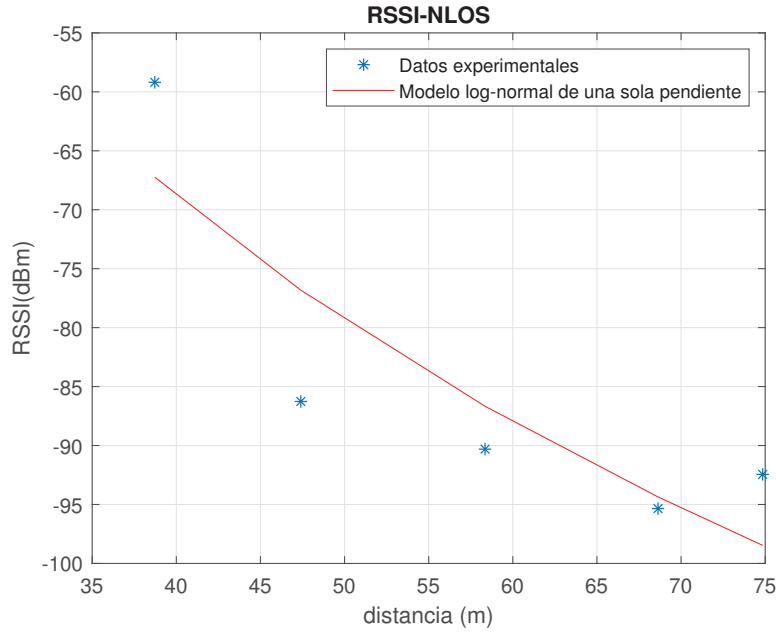


Figura 4.3: Modelo log-normal de una sola pendiente para la trayectoria NLOS de la primera campaña de medidas

Trayectoria	Parámetros		σ (dBm)
	$RSSI(d_0)$	n	σ
LOS	-50.9567	0.0477	1.6274
NLOS	-67.2370	10.9106	8.2533

Cuadro 4.1: Parámetros del modelo log-normal de una sola pendiente para las trayectorias LOS y NLOS de la primera campaña de medidas

- Espacio abierto con restos arqueológicos (posiciones de color amarillo del 8-11, ver Figura 3.17).

Los parámetros de los modelos para esta trayectoria se muestran en el Cuadro 4.2. Analizando estos parámetros podemos ver que el modelo log-normal de doble pendiente tiene una menor desviación estándar, lo que se refleja en la cercanía de los valores estimados con los valores reales por lo tanto este modelo caracterizará mejor este trayecto, también podemos notar que el exponente de pérdida de trayecto de la primera región $n_1 = 1,14$ es relativamente pequeño lo que refleja la buena cobertura de LoRa y la cantidad mínima de obstrucciones en esta región (sólo arbustos nativos), por otra parte podemos ver lo opuesto en la segunda región donde el exponente de pérdida de trayecto tiene un valor de $n_2 = 15,76$ que refleja una alta atenuación debido a los restos arqueológicos (muros de tipo inca).

El modelo más aproximado para este trayecto se describe en la siguiente ecuación:

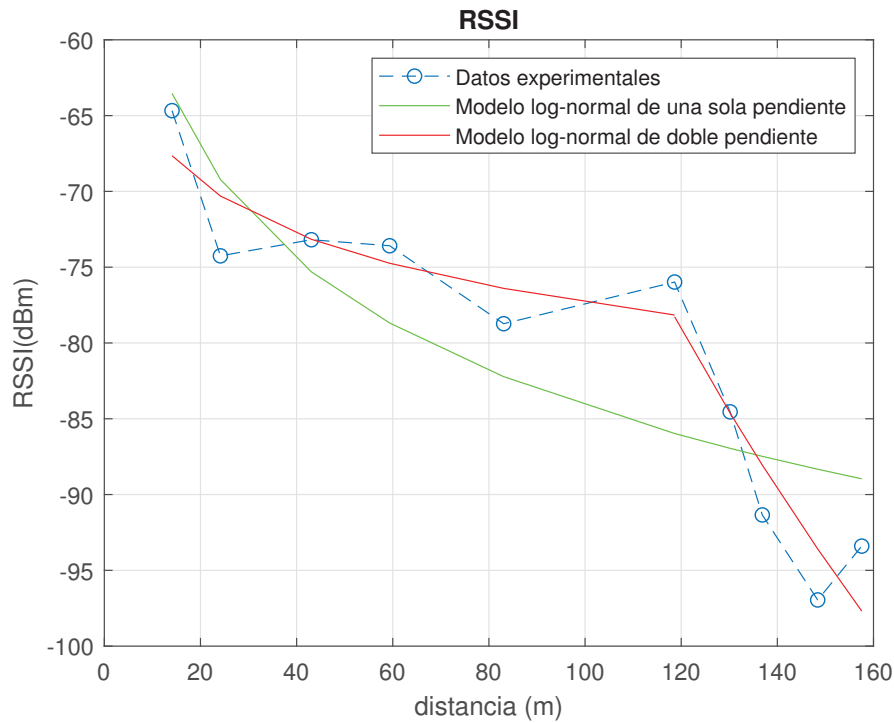


Figura 4.4: Comparación de modelos para la trayectoria NLOS

Modelo	Parámetros				σ (dBm)	
	$RSSI(d_{01})$	$RSSI(d_{02})$	n_1	n_2	σ_1	σ_2
Log-normal de una sola pendiente	-63.54	-	2.42	-	5.96	-
Log-normal de doble pendiente	-54.61	248.46	1.14	15.76	3	3.9

Cuadro 4.2: Parámetros de los modelos para NLOS

$$RSSI_{NLOS}(d) = \begin{cases} -54,61 - 11,4 \log(d) & \text{si } d \leq 120 \\ 248,46 - 157,6 \log(d) & \text{si } d > 120 \end{cases} \quad (4.1)$$

Trayectoria LOBS

Para esta trayectoria podemos considerar dos regiones:

- La loma (posiciones de color rojo del 2-5, ver Figura 3.17), esta obstrucción genera una gran atenuación en la cual no hay línea de vista.
- Espacio abierto con restos arqueológicos (posiciones de color rojo del 8-11, ver Figura 3.17).

Podemos ver en la Figura 4.5 que el modelo log-normal de doble pendiente no tiene coincidencia entre las rectas que caracterizan el RSSI para cada región, lo cual se debe a que la loma tiene mayor obstrucción que los restos arqueológicos además de introducirle mayor

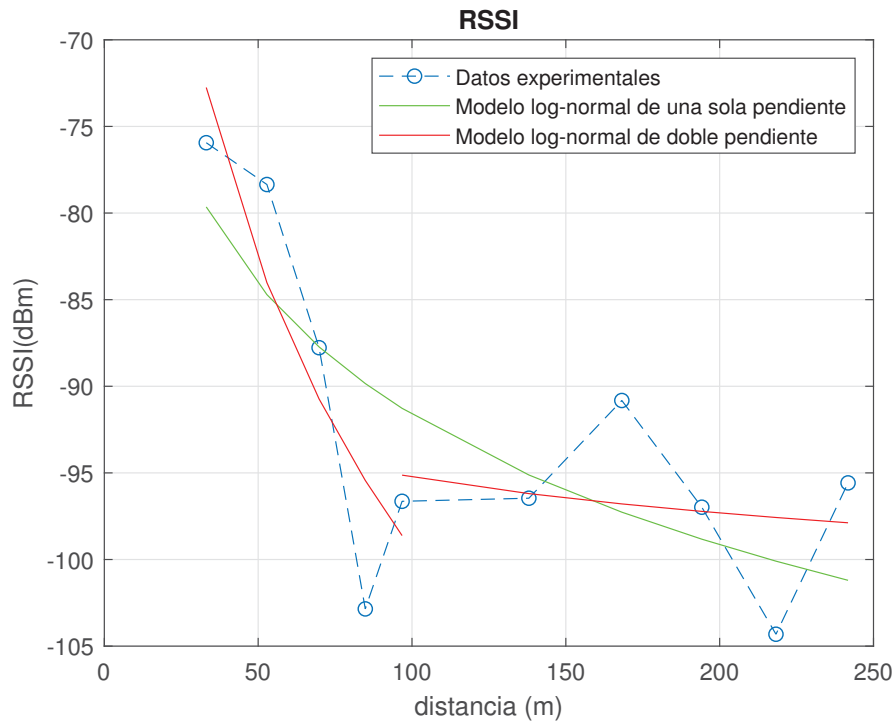


Figura 4.5: Comparación de modelos para la trayectoria LOBS

aleatoriedad a esa región, los parámetros de los modelos se muestran en el Cuadro 4.3. Un análisis a estos parámetros muestra que las desviación estándar de ambos modelos no son muy lejanos, para este trayecto debido a las fluctuaciones provocadas por la loma en el RSSI el modelo más acorde será el de una pendiente debido a que refleja mejor el comportamiento del RSSI para este trayecto.

Modelo	Parámetros				$\sigma(dBm)$	
	$RSSI(d_{01})$	$RSSI(d_{02})$	n_1	n_2	σ_1	σ_2
Log-normal de una sola pendiente	-79.65	-	2.5	-	6.6	-
Log-normal de doble pendiente	11.71	-81.39	5.56	0.69	6.05	4.71

Cuadro 4.3: Parámetros de los modelos para LOBS

El modelo más aproximado para este trayecto se describe en la siguiente ecuación:

$$RSSI_{LOBS}(d) = -79,65 - 2,5 \log(d) \tag{4.2}$$

4.2. Tiempo de retardo RMS y modelos ARMA de los muros inca principales de Rumiwasi

4.2.1. Tiempo de retardo RMS

Los perfiles de retardo de potencia promedio para el muro de prueba 1 a las distancias de 30cm y 60cm se muestran en las Figuras 4.6 y 4.7 respectivamente, el perfil de retardo de potencia promedio del muro de prueba 2 se muestra en la Figura 4.8. Los valores estadísticos del tiempo de retardo se muestran en el Cuadro 4.4, de ahí podemos comparar los valores del tiempo de retardo RMS de ambos muros a una distancia de 60cm y concluir que el muro de prueba 2 posee materiales que tienen una atenuación mayor al muro de prueba 1, cabe resaltar que si hacemos pruebas con una estructura conformada por estos muros equivaldría a hacer las pruebas con las antenas apuntando en direcciones opuestas.

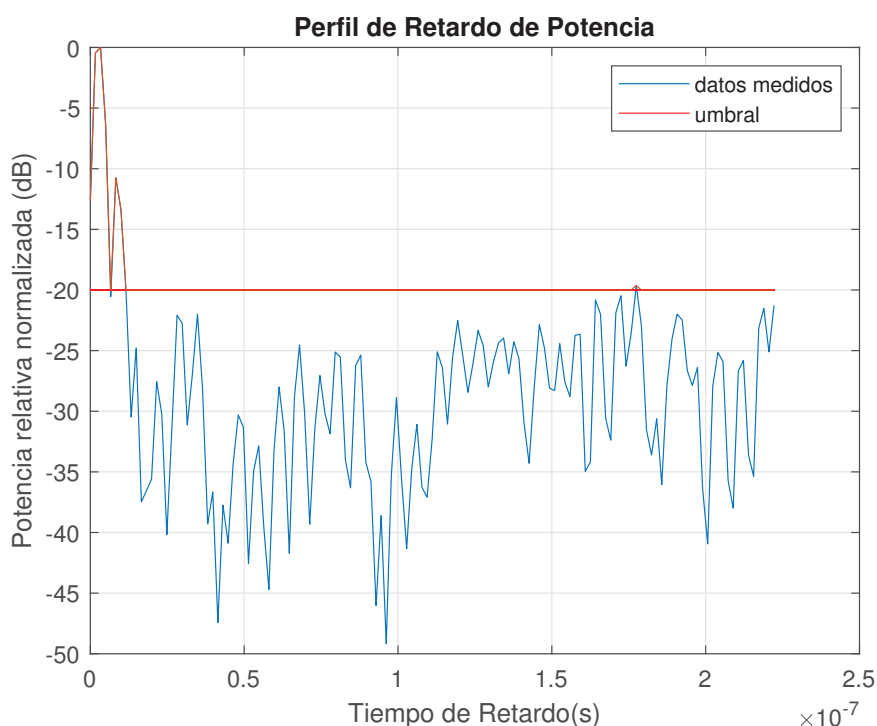


Figura 4.6: Perfil retardo de potencia muro de prueba 1 a una distancia de 30cm

Modelo ARMA del muro de prueba 1 a una distancia de 30cm

La metodología aplicada se describe en la Sección 3.3., la grafica de la señal medida normalizada se muestra en la Figura 4.9, esto se debido a que las señal original estaba em dBm pero al convertirla a milivattios esta señal era demasiado pequeña se procedió a normalizarla.

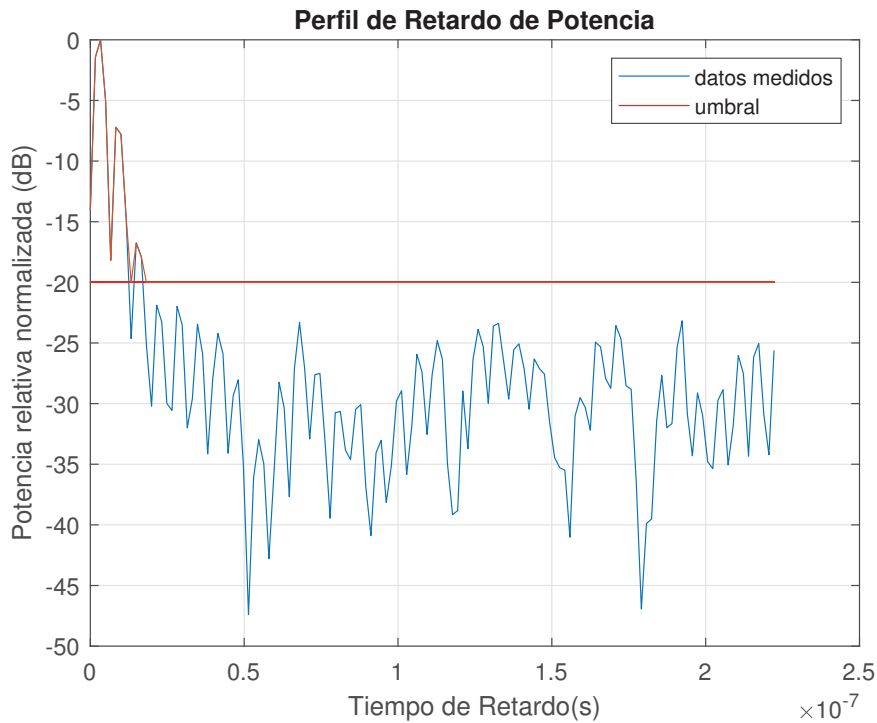


Figura 4.7: Perfil retardo de potencia muro de prueba 1 a una distancia de 60cm

Distancia (cm)	Valores estadísticos de τ_{RMS} en ns			
	Muro de prueba 1		Muro de prueba 2	
	μ	σ	μ	σ
30	14.68	7.65	-	-
60	11.56	0.85	23.22	10.89

Cuadro 4.4: Valores estadísticos del tiempo de retardo para ambos muros de prueba

Los gráficos de autocorrelación simple y autocorrelación parcial se muestran en las Figuras 4.10 y 4.11. Estos gráficos nos sirven para analizar cuando el proceso puede ser solo media móvil, autoregresivo o autoregresivo con media móvil, para este último caso ambas gráficas tienen que tender a un decaimiento hacia cero [28]. Para este modelo vemos que ambas gráficas tienden a cero por lo que el proceso debería salirnos un proceso ARMA. Para calcular el orden del modelo se usó el comando *arma_order_select_ic* que utiliza el criterio de Akaike para calcular el orden óptimo que en este caso sería de (4,1). Una vez calculado esto el comando *model_fit.predict* calcula los parámetros para el orden asignado, la comparación de la predicción del modelo con la señal original se muestra en la Figura 4.12. La función de transferencia obtenida viene dado por:

$$H_{30cm} = \frac{0,1567(1 - 0,9097 * z)}{1 - 1,2587 * z - 0,0627 * z^2 - 0,0047 * z^3 + 0,3321 * z^4} \quad (4.3)$$

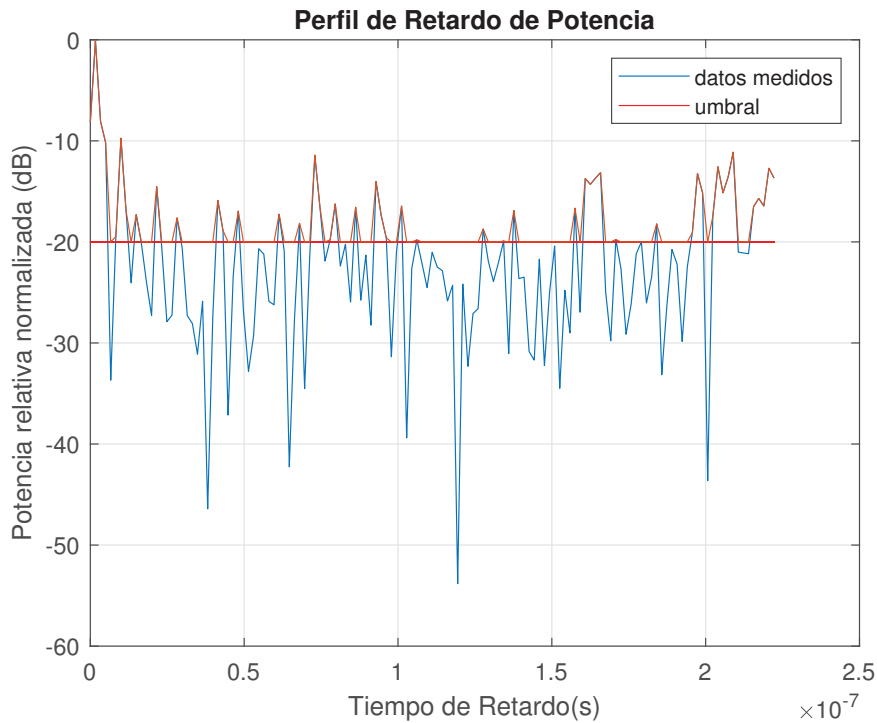


Figura 4.8: Perfil retardo de potencia muro de prueba 2 a una distancia de 60cm

El gráfico de los residuos del modelo y de los valores de error residual se muestran en las Figuras 4.13 y 4.14 respectivamente. Este último nos indica que los residuos siguen una distribución gaussiana no centrada en cero. El código completo es el notebook 1 que se muestra en el Apéndice B.2.4..

Modelo ARMA del muro de prueba 1 a una distancia de 60cm

La gráfica de señal medida normalizada se muestra en la Figura 4.15, las gráficas de autocorrelación simple y autocorrelación parcial se muestran en las Figuras 4.16 y 4.17 respectivamente los cuales podemos interpretar siguiendo el Cuadro B.1 del Apéndice B, la autocorrelación simple tiene un decaimiento hacia cero, mientras la autocorrelación parcial tiene una mezcla de exponenciales y senoidales por lo que será un proceso arma pero con un orden mayor a 1 para el proceso de media móvil. Una vez calculado el orden bajo el criterio de Akaike este nos da el valor de (4,2) lo que contrasta el análisis de las autocorrelaciones. Con ese orden se calcularon los parámetros, la gráfica de comparación de la señal original y la predicción del modelo se muestra en la Figura 4.18, de ahí tendremos la siguiente función de transferencia:

$$H_{60cm} = \frac{0,0797 * (1 - 0,4799 * z + 0,7109 * z^2)}{1 - 1,2626 * z + 0,8448 * z^2 - 0,7324 * z^3 + 0,1915 * z^4} \quad (4.4)$$

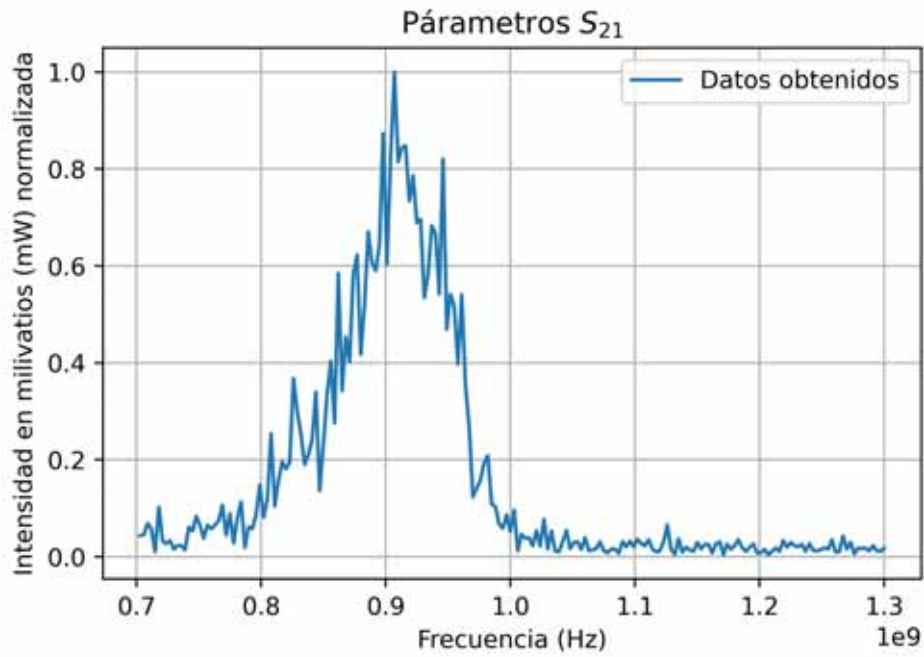


Figura 4.9: Señal medida normalizada a una distancia de 30cm

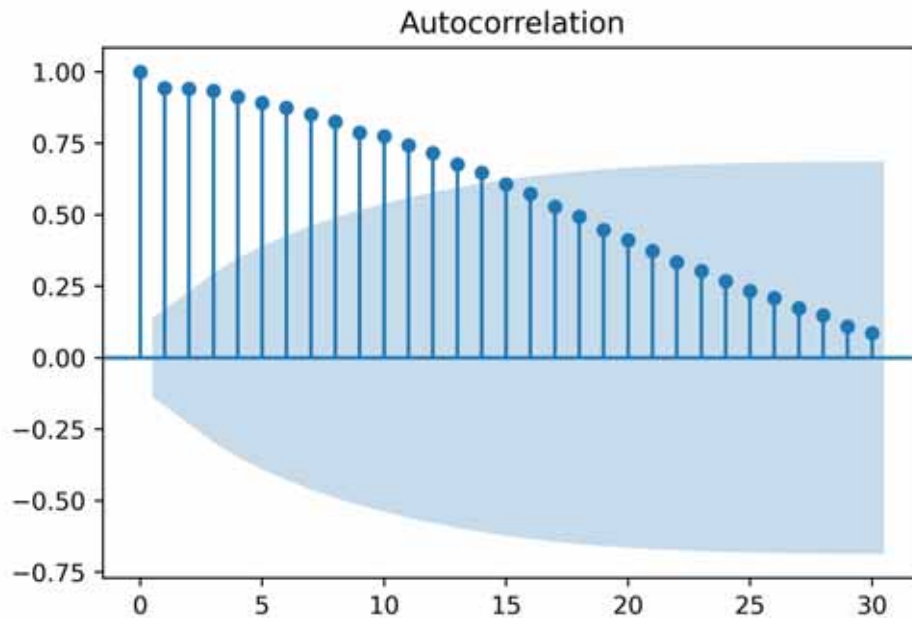


Figura 4.10: Señal medida normalizada a una distancia de 30cm

El gráfico de los residuos del modelo y de los valores de error residual se muestran en las Figuras 4.19 y 4.20 respectivamente. Este último nos indica que los residuos siguen una

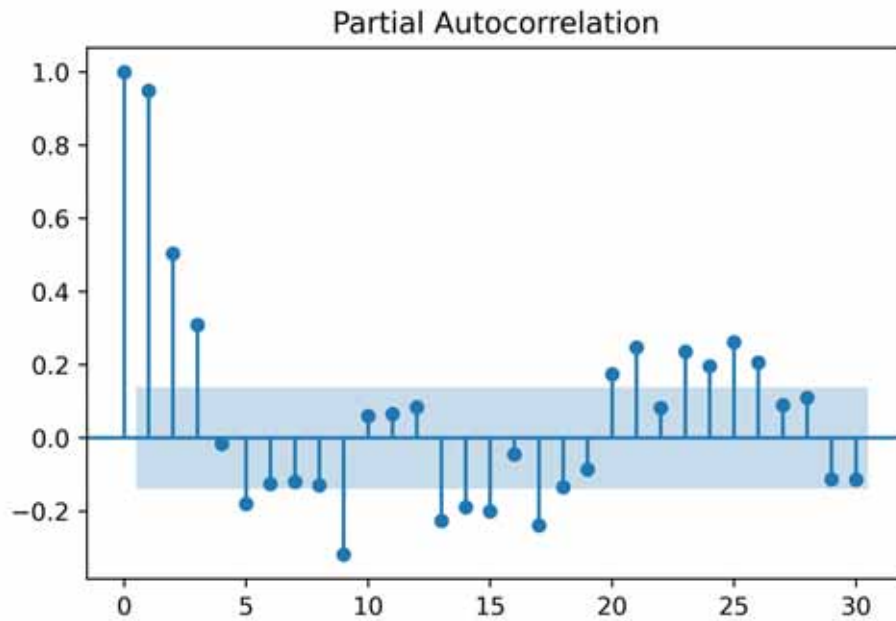


Figura 4.11: Señal medida normalizada a una distancia de 30cm

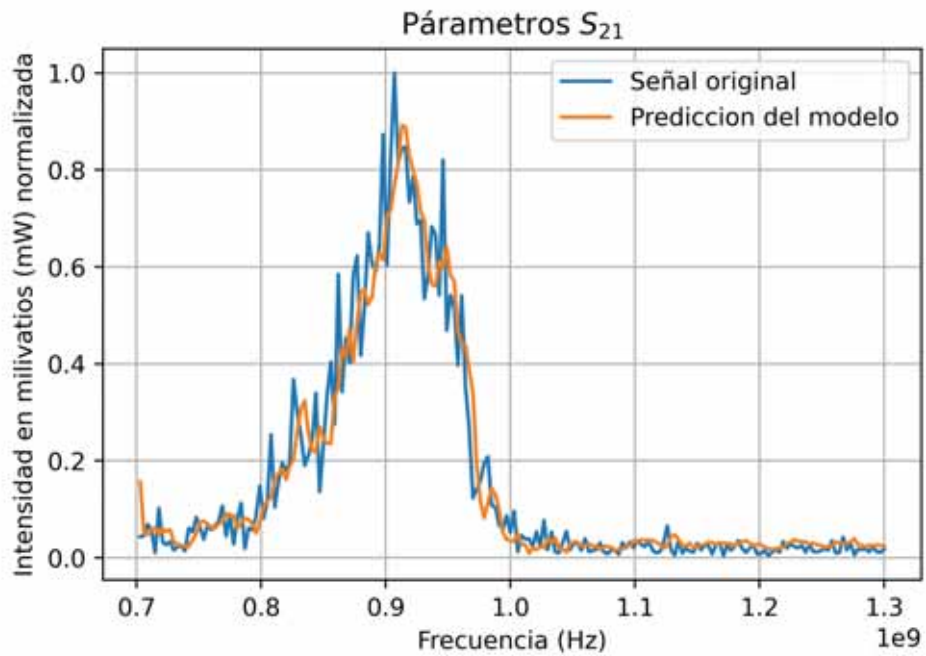


Figura 4.12: Comparación de la predicción del modelo con los datos obtenidos

distribución gaussiana no centrada en cero. El código completo es el notebook 2 que se

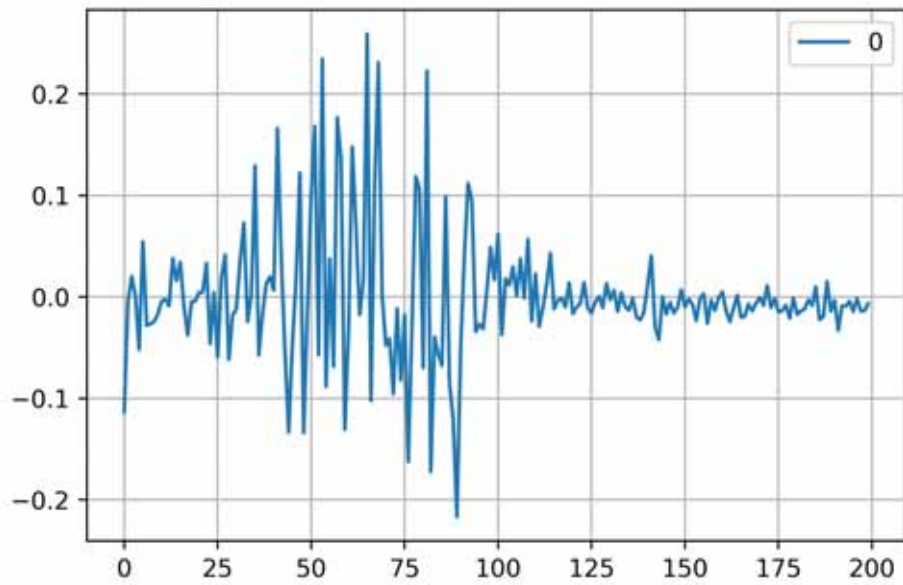


Figura 4.13: Grafica de los residuos del modelo

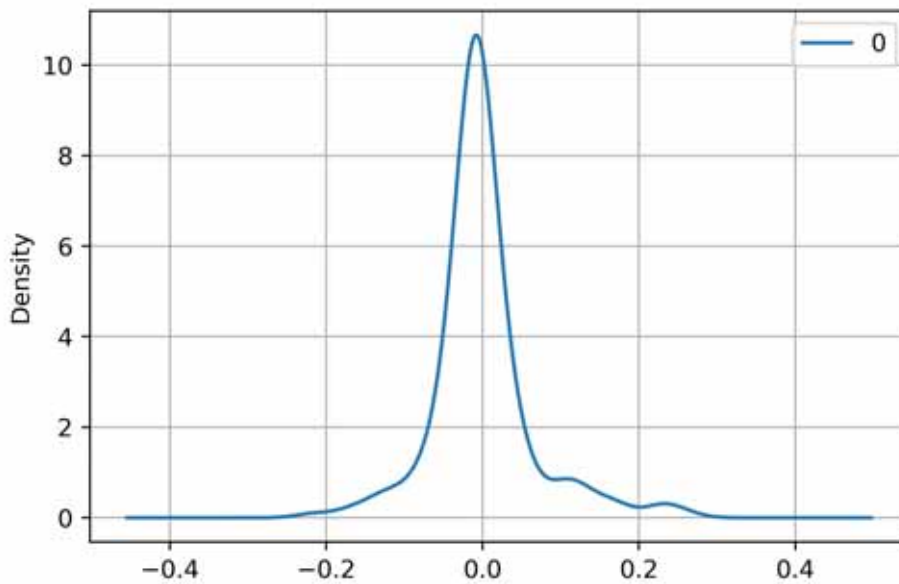


Figura 4.14: Gráfico de densidad de los valores de error residual

muestra en el Apéndice B.2.4..

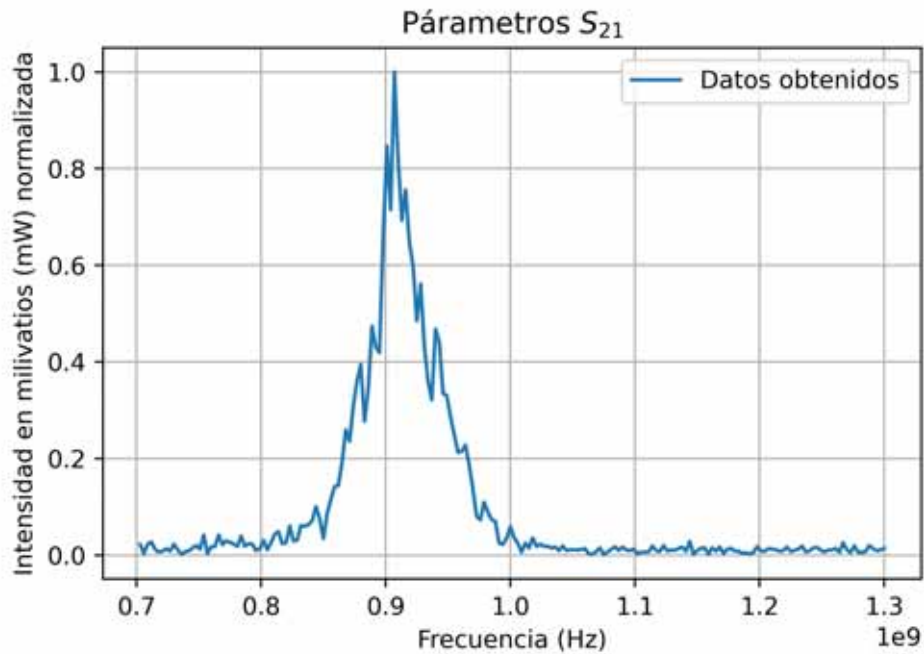


Figura 4.15: Señal medida normalizada a una distancia de 60cm

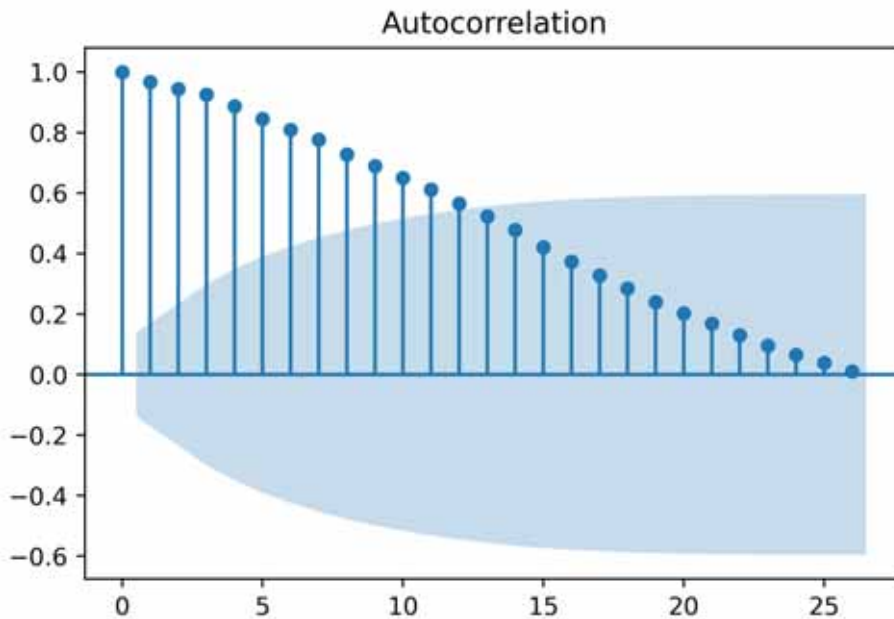


Figura 4.16: Señal medida normalizada a una distancia de 60cm

Modelo ARMA del muro de prueba 2 a una distancia de 60cm

La grafica de señal medida normalizada se muestra en la Figura 4.21, las graficas de autocorrelación simple y autocorrelación parcial se muestran en las Figuras 4.22 y 4.23

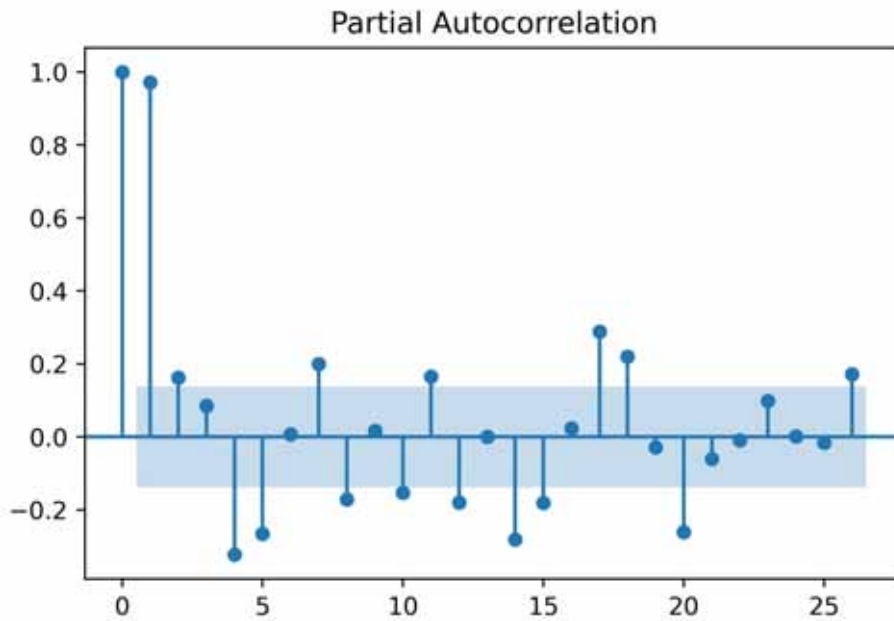


Figura 4.17: Señal medida normalizada a una distancia de 60cm

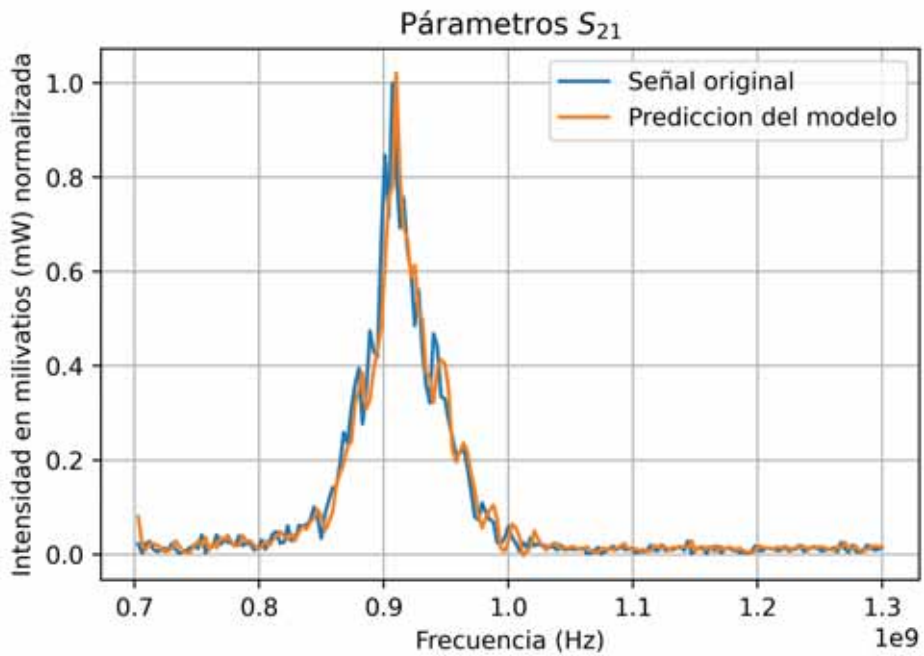


Figura 4.18: Comparación de la predicción del modelo con los datos obtenidos

respectivamente los cuales podemos interpretar de la siguiente manera la autocorrelación

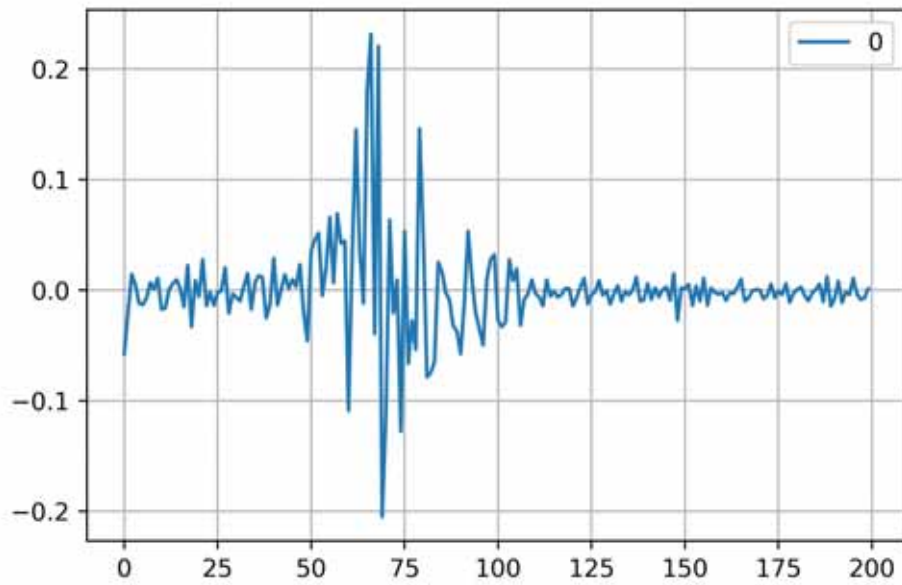


Figura 4.19: Grafica de los residuos del modelo

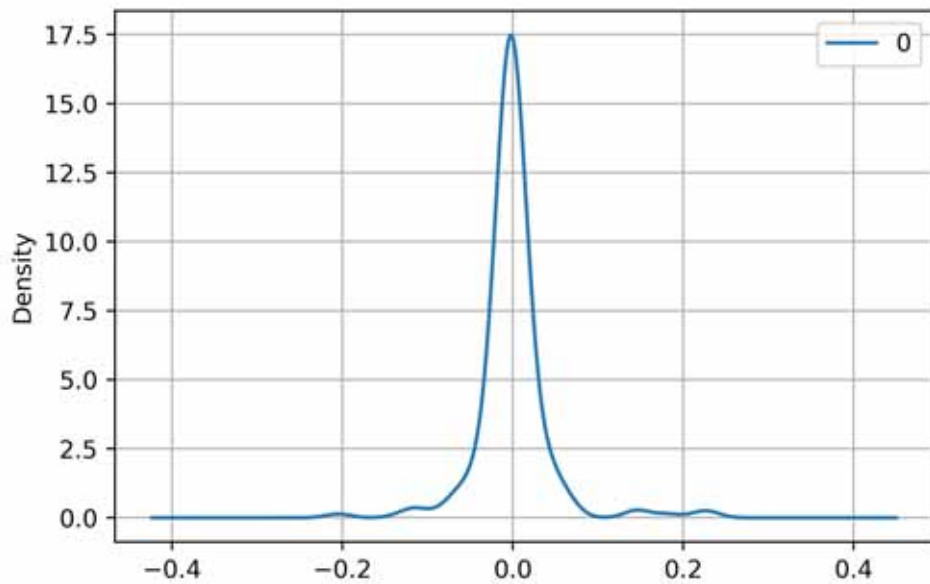


Figura 4.20: Gráfico de densidad de los valores de error residual

simple tiene un decaimiento hacia cero, mientras la autocorrelación parcial tiene una mezcla

de exponenciales y senoidales por lo que sera un proceso arma pero con un orden mayor a 1 para el proceso de media móvil. Una vez calculado el orden bajo el criterio de Akaike este nos da el valor de (4,2) lo que contrasta el análisis de las autocorrelaciones. Con ese orden se calcularon los parámetros, la grafica de comparación de la señal original y la predicción del modelo se muestra en la Figura 4.24, de ahí tendremos la siguiente función de transferencia:

$$H_{60cm} = \frac{0,2318 * (1 - 0,9161 * z + 0,4280 * z^2)}{1 - 1,1497 * z + 0,3219 * z^2 + 0,3233 * z^3 - 0,4158 * z^4} \quad (4.5)$$

El grafico de los residuos del modelo y de los valores de error residual se muestran en las Figuras 4.25 y 4.26 respectivamente. Este ultimo nos indica que los residuos siguen una distribución gaussiana no centrada en cero. El código completo es el notebook 3 que se muestra en el Apéndice B.2.4..

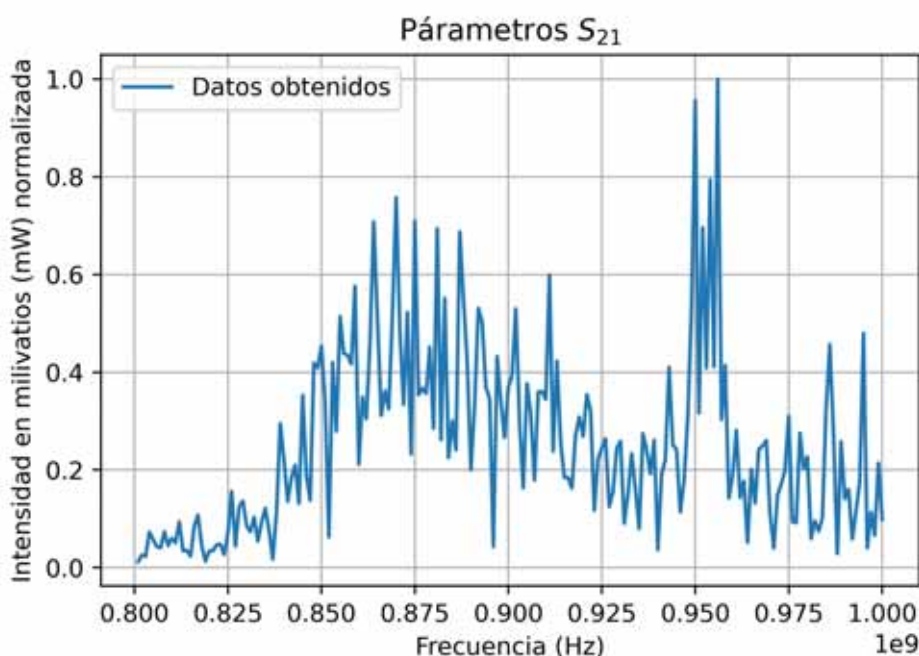


Figura 4.21: Señal medida normalizada a una distancia de 60cm

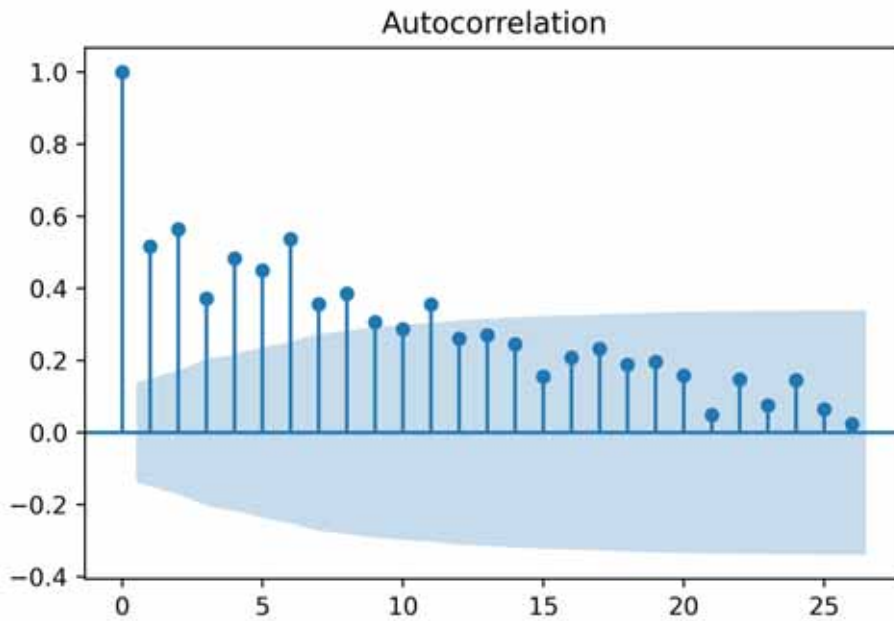


Figura 4.22: Señal medida normalizada a una distancia de 60cm

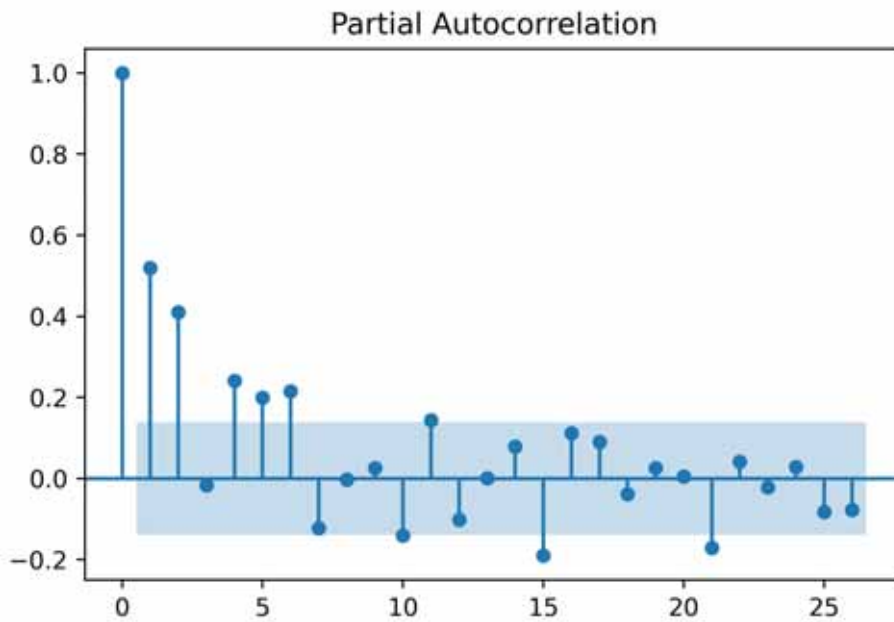


Figura 4.23: Señal medida normalizada a una distancia de 60cm

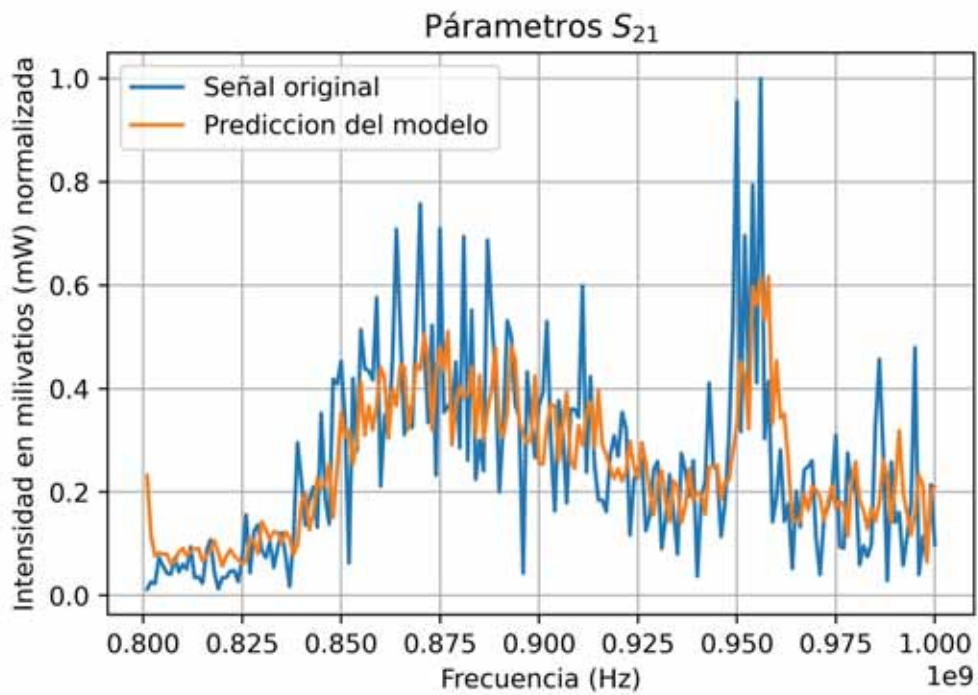


Figura 4.24: Comparación de la predicción del modelo con los datos obtenidos

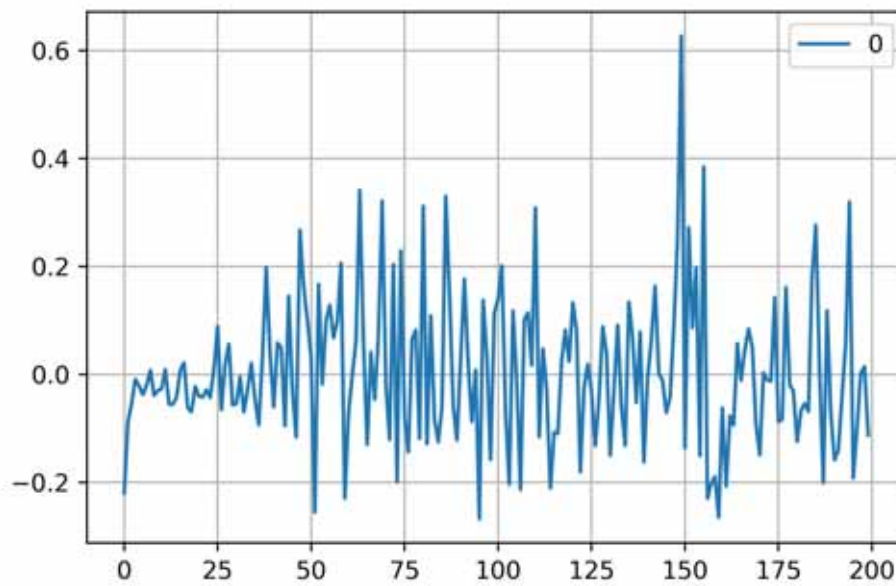


Figura 4.25: Grafica de los residuos del modelo

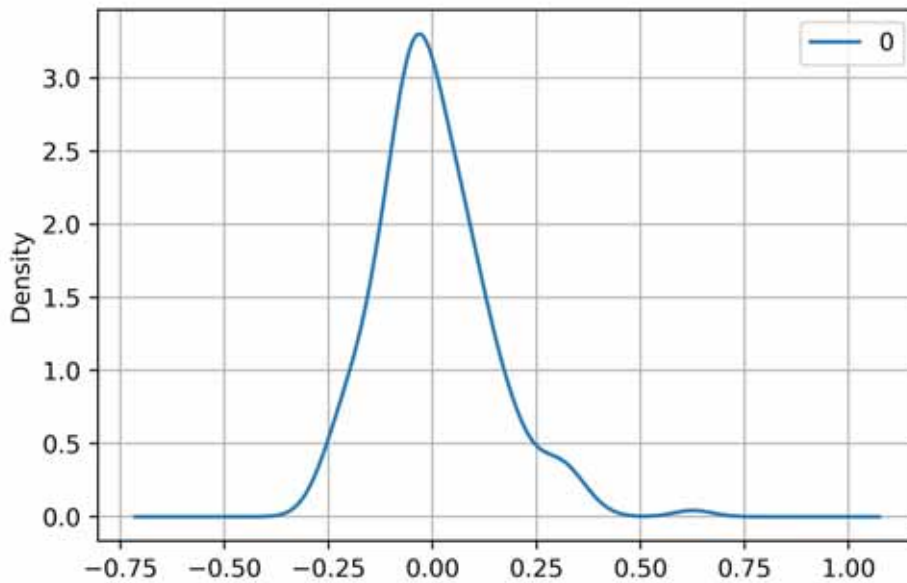


Figura 4.26: Gráfico de densidad de los valores de error residual

4.3. Conclusiones y Recomendaciones

4.3.1. Conclusiones

- Se determinó que para efectos de gran escala, los restos arqueológicos (muros de prueba 1 y 2) poseen una gran atenuación lo que delimitaría, para una futura red de sensores, un radio máximo de $d = 140m$ para una recepción óptima de cada nodo sensor y por lo menos 3 nodos sensores para cubrir el área principal de esta zona arqueológica. Además para efectos en pequeña escala el parámetro S_{21} nos proporcionó dos parámetros principales en el modelamiento de canal: tiempo de retardo RMS del canal que confirmo que el muro de prueba 2 de la zona arqueológica posee mayor atenuación y los modelos ARMA de los muros en los que se obtuvo una aproximación cercana a los datos medidos.
- Se identificó que los modelos de una sola y doble pendiente de RSSI aplicados a WSN caracterizan con mayor aproximación la propagación en gran escala, además de modelos autoregresivos con media móvil (ARMA) para caracterizar la respuesta en frecuencia del canal de comunicación inalámbrico. Esto en base a las características de la zona arqueológica de Rumiwasi.
- Se obtuvo los parámetros de los modelos experimentales más adecuados a la zona arqueológica de Rumiwasi. También se caracterizó el tiempo de retardo RMS del canal y el modelo ARMA de los muros, ya que proveen información útil para simulaciones

de la propagación de ondas electromagnéticas en muros con las mismas características de esta zona arqueológica.

- Se logró calcular exitosamente los parámetros para los modelos propuestos usando Matlab debido a que esta es una herramienta potente para manejar una gran cantidad de datos y Python que posee una librería llamada Statmodels que nos provee herramientas potentes para obtener modelos estadísticos de los datos medidos, usando algoritmos eficientes como la estimación de máxima verosimilitud usando filtro de Kalman.

4.3.2. Recomendaciones

- Se recomienda utilizar tarjetas que ofrezcan mayor flexibilidad tanto para modificar su firmware o su hardware ya que permite implementar con facilidad cualquier modificación para implementar una sonda de canal o implementar una red de sensores inalámbricos para la geolocalización.
- Se recomienda usar Python como una herramienta con mucha más versatilidad que Matlab debido a la cantidad de herramientas útiles con resultados validados con software propietario.
- Si se desea implementar un sistema de geolocalización se recomienda implementar un sistema con tres nodos sensores en un radio menor a 150m para garantizar la cobertura y la menor pérdida de paquetes enviados.
- Si se desea simular los modelos obtenidos en este trabajo se recomienda usar Unity para su implementación ya que este provee herramientas como physics que nos permite desarrollar simulaciones mucho más reales.
- Los modelos ARMA son ampliamente usados para la simulación de datos por lo que los modelos obtenidos en este trabajo serían de mucha utilidad para futuras investigaciones de la propagación de ondas electromagnéticas en zonas arqueológicas con características similares a la zona arqueológica de Rumiwasi.

Bibliografía

- [1] T. S. Rappaport, *Wireless communications: principles and practice*, 2nd ed., Prentice Hall PTR, 2002.
- [2] X. Yin y X. Cheng, *Propagation channel characterization, parameter estimation and modeling for wireless communications*, Wiley-IEEE, 2016.
- [3] S. J. Howard y K. Pahlavan, “Measurement and analysis of the indoor radio channel in the frequency domain”, *IEEE Transactions on Instrumentation and Measurement*, IEEE, 1990.
- [4] S. Kurt y B. Tavli, “Path-Loss Modeling for Wireless Sensor Networks”, *IEEE Antennas & Propagation Magazine*, IEEE, 2017.
- [5] A. Förster, *Introduction to wireless sensor networks*, IEEE Press-Wiley, 2016.
- [6] Semtech Corporation, *AN1200.22 LoRa Modulation Basics*, Application Note, 2015, URL: <https://www.frugalprototype.com/wp-content/uploads/2016/08/an1200.22.pdf>
- [7] Devalal S, Karthikeyan A, “LoRa Technology - An Overview”, In: 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 284-290, Coimbatore, 2018. doi:10.1109/ICECA.2018.8474715
- [8] Han G, Jiang J, Zhang C, Duong T, Guizani M, Karagiannidis GK (2018) Survey on Mobile Anchor Node Assisted Localization in Wireless Sensor Networks. In: *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2220-2243,, IEEE. doi:10.1109/COMST.2016.2544751
- [9] Gharghan SK, Nordin R, Ismail M., Ali J A (2016) Accurate Wireless Sensor Localization Technique Based on Hybrid PSO-ANN Algorithm for Indoor and Outdoor Track Cycling. In: *IEEE Sensors Journal*, vol. 16, no. 2, pp. 529-541, Jan.15, IEEE. doi:10.1109/JSEN.2015.2483745
- [10] P. Gogendeau, N. M. Murad, S. Bernard, V. Kerzerho, L. Deknyff y S. Bonhommeau, “Oversea Radio Measurements and Channel Characterization with LoRa Technology”, En: 2018 IEEE Radio and Antenna Days of the Indian Ocean (RADIO), Grand Port, pp. 1-2, 2018.

- [11] Y. Cheng and H. Kobayashi, “Signal strength based indoor geolocation”, 2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333), IEEE, 2002.
- [12] L. Cheng, T. Ostertag, M. Loschonsky y L. M. Reindl, “Measurement of Mobile Radio Propagation Channel in Ruins”, IEEE International Conference on Wireless Communications, Networking and Information Security, IEEE, 2010.
- [13] B. C. Fargas y M. N. Petersen, “GPS-free Geolocation using LoRa in Low-Power WANs’, Global Internet of Things Summit (GloTS), IEEE, 2017.
- [14] J. Danebjer y V. Halldórsson, “A Hybrid Approach to GPS-Free Geolocation over LoRa’, tesis de maestria, Lund University, 2018.
- [15] N. J. A. Sloane y A. D. Wyner, *Claude Elwood Shannon Collected Papers*, IEEE PRESS, WILEY-INTERSCIENCE, 1993.
- [16] S. R. Saunders y A. Aragón-Zavala , *Antennas and Propagation for wireless communications systems*, John Wiley & sons, 2007.
- [17] C. Phillips, D. Sicker and D. Grunwald, “A survey of Wireless path loss prediction and coverage mapping methods”, IEEE Communications Surveys & Tutorials, IEEE, 2013.
- [18] K. Pahlavan y A. H. Levesque, *Wireless Information Networks*, 2nd ed., Wiley-Interscience, 2005.
- [19] J. Miranda, R. Abrishambaf, T. Gomes, P. Golçalves, J. Cabral, A. Tavares y J. Monteiro, “Path loss exponent analysis in wireless sensor networks: experimental evaluation”, 11th IEEE International Conference on Industrial Informatics (INDIN), IEEE, 2013.
- [20] S. Y. Seidel and T. S. Rappaport, “914 MHz path loss prediction models for indoor wireless communications in multifloored buildings”, IEEE Transactions on Antennas and Propagation, Vol. 40, IEEE, 1992.
- [21] T. K. Sarkar, Z. Ji, K. Kim, A. Medouri, y M. Salazar-Palma, “A survey of various propagation models for mobile communication”, IEEE Antennas and Propagation Magazine, Vol. 45, No. 3, IEEE, 2003.
- [22] G. Morrison y G. Fattouche, “Super-Resolution Modeling of the Indoor Radio Propagation Channel”, IEEE Transactions on Vehicular Technology, IEEE, 1998.
- [23] S. Haykin, *Adaptive filter theory*, 5th ed., Pearson, 2014.
- [24] M. Hayes, *Statistical digital signal processing and modeling*, John Wiley & Sons, 1996.
- [25] C. W. Therrien, *Discrete random signals and statistical signal modeling*, Prentice Hall, 1992.

- [26] M. D. Srinath, P. K. Rajasekaran y R. Viswanathan, *Statistical signal processing with applications*, Prentice Hall, 1996.
- [27] D. Peña, *Análisis de series temporales*, Alianza Editorial, 2010.
- [28] D. Peña, *Estadística: modelos y métodos*, Vol. 2, Alianza Editorial, 1995.
- [29] G. Box, G. Jenkins, G. Reinsel y G. Ljung, *Time series analysis-forecasting and control*, 5th Ed., Wiley, 2016.
- [30] H. Akaike, “A new look at the statistical model identification”, en *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716-723, December 1974, doi: 10.1109/TAC.1974.1100705.
- [31] T. Söderström y P. Stoica, *System identification*, Prentice Hall, 1989.
- [32] D. C. Montgomery y G. C Runger, *Applied statistics and probability for engineers*, 7th Ed., Wiley, 2018.
- [33] Seabold S. y Perktold J., *statsmodels: Econometric and statistical modeling with python*, Proceedings of the 9th Python in Science Conference, 2010. URL: <https://www.statsmodels.org/stable/index.html>
- [34] C. W. Davidson, *Transmission Lines for Communications with CAD programs*, 2nd ed., Macmillan, 1989.
- [35] Keysight Technologies, *Understanding the Fundamental Principles of Vector Network Analysis*, Application Note, URL: <https://www.keysight.com/zz/en/assets/7018-06841/application-notes/5965-7707.pdf>

Apéndice A

Parámetros S de una red

Los parámetros de propagación describen la relación entre varias ondas viajeras que pueden ser excitadas en todos los puertos de una línea de transmisión o red [34]. En resumen [35]:

- Son relativos a medidas familiares (ganancia, pérdida, coeficiente de reflexión, etc.).
- Relativamente fáciles de medir.
- Pueden conectarse en cascada los parámetros S de múltiples dispositivos para predecir el desempeño del sistema.
- Analíticamente conveniente
 - Programas CAD.
 - Análisis en diagramas de flujo.
- Si se desean pueden calcularse los parámetros H, Y, o Z de los parámetros S.

La relación entre los parámetros viene dado por:

$$\begin{aligned}b_1 &= S_{11}a_1 + S_{12}a_2 \\b_2 &= S_{21}a_1 + S_{22}a_2\end{aligned}$$

de tal manera que cada parámetro se puede calcular usando las siguientes relaciones:

$$\begin{aligned}S_{11} &= \frac{\textit{Reflejada}}{\textit{Incidente}} = \frac{b_1}{a_1} \Big|_{a_2=0} \\S_{21} &= \frac{\textit{Transmitida}}{\textit{Incidente}} = \frac{b_2}{a_1} \Big|_{a_2=0}\end{aligned}$$

$$S_{22} = \frac{\text{Reflejada}}{\text{Incidente}} = \frac{b_2}{a_2} \Big|_{a_1=0}$$

$$S_{12} = \frac{\text{Transmitida}}{\text{Incidente}} = \frac{b_1}{a_2} \Big|_{a_1=0}$$



Figura A.1: Parámetros S de un dispositivo bajo prueba (DUT) Fuente: Adaptado de [35]

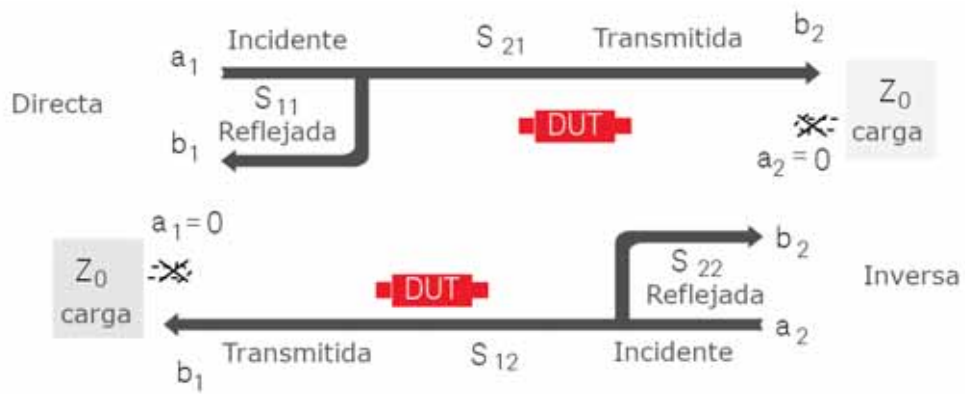


Figura A.2: Medición de los parámetros S Fuente: Adaptado de [35]

Apéndice B

Resumen de la función de autocorrelación simple (fas) y la función de autocorrelación parcial (fap) de los procesos ARMA [28]

Los fas y fap de los procesos ARMA es el resultado de la superposición de sus propiedades AR (autorregresivo) y MA (media móvil): en la *fas* ciertos coeficientes iniciales que dependen del orden de la parte del MA y después de un decrecimiento por la parte AR. En la *fap* valores iniciales dependientes del orden AR y seguidos del decrecimiento debido a la parte MA. Esta estructura compleja hace que el orden de un proceso ARMA sea difícil de identificar en la practica.

En el siguiente tabla podemos ver un resumen de las *fas* y *fap* de los procesos arma.

	f.a. simple (<i>fas</i>)	f.a. parcial (<i>fap</i>)
AR(p)	Muchos coeficiente no nulos que decrecen con el retardo como mezcla de exponenciales y senoides	p primeros coeficientes no nulos y el resto cero
MA(q)	q primeros coeficientes no nulos y el resto cero	Muchos coeficientes no nulos que decrecen con el retardo como mezcla de exponenciales y senoides
ARMA(p,q)	Decrecimiento hacia cero.	Decrecimiento hacia cero.

Cuadro B.1: Resumen de las *fas* y *fap* de procesos ARMA

Apéndice C

Codigos utilizados en el presente trabajo

C.1. Codigos en Arduino para el transmisor y receptor LoRa

C.1.1. Transmisor

```
1 #include <SPI.h> //Libreria que permite la comunicacion con la
    interfaz de perifericos en serie
2 #include <LoRa.h> //Libreria LoRa para enviar y recibir datos
    usando radios LoRa
3 #include <Wire.h> //Libreria que permite comunicarse con I2C
4 #include <TinyGPS++.h> //Libreria para uso del modulo GPS y sus
    valores
5 TinyGPSPlus gps; // parte del codigo para usar tiny GPS++
    creando un objeto gps
6 #define SCK 5 // GPIO5 — SX1278's SCK
7 #define MISO 19 // GPIO19 — SX1278's MISO
8 #define MOSI 27 // GPIO27 — SX1278's MOSI
9 #define SS 18 // GPIO18 — SX1278's CS
10 #define RST 14 // GPIO14 — SX1278's RESET
11 #define DIO 26 // GPIO26 — SX1278's IRQ(Interrupt Request)
12 #define BAND 915E6 //Banda de transmision LoRa
13
14 unsigned int counter = 0;
15
16 String rssi = "RSSI --";
```

```

17 String packSize = "--";
18 String packet ;
19
20
21 void setup() {
22
23     Serial.begin(115200);    // Inicializacion del puerto serial y
        sus parametros
24     Serial1.begin(9600, SERIAL_8N1, 12, 15);
25     while (!Serial);
26     Serial.println();
27     Serial.println("LoRa Sender Test");
28
29     SPI.begin(SCK,MISO,MOSI,SS);
30     LoRa.setPins(SS,RST,DI0);
31     if (!LoRa.begin(915E6)) {
32         Serial.println("Starting LoRa failed!");
33         while (1);
34     }
35     //LoRa.onReceive(cbk);
36 // LoRa.receive();
37 LoRa.setSpreadingFactor(7); //Configuracion de valor de Spreading
        factor
38 LoRa.setFrequency(920E6); //Frecuencia a la cual se transmite
        el pulso modulado
39     Serial.println("init ok");
40
41 }
42
43 void loop() {
44
45     delay(1000);
46
47     Serial.println(String(counter)); // imprime el valor del contador
48     Serial.print("Latitude : "); //codigo para mostrar los valores
        del GPS
49     Serial.println(gps.location.lat(), 5);
50     Serial.print("Longitude : ");
51     Serial.println(gps.location.lng(), 4);
52     Serial.print("Satellites: ");
53     Serial.println(gps.satellites.value());
54 //Serial.print("Altitude : ");
55 //Serial.print(gps.altitude.feet() / 3.2808);

```

```

56 //Serial.println("M");
57 Serial.print("Time      : ");
58 Serial.print(gps.time.hour());
59 Serial.print(":");
60 Serial.print(gps.time.minute());
61 Serial.print(":");
62 Serial.println(gps.time.second());
63 Serial.println("*****");
64
65 smartDelay(1000);
66
67 if (millis() > 5000 && gps.charsProcessed() < 10)
68     Serial.println(F("No GPS data received: check wiring"));
69
70 float lon=gps.location.lng(); //variables auxiliares para imprimir
    los datos
71 float lati=gps.location.lat();
72 int hora=gps.time.hour();
73 int minu=gps.time.minute();
74 int seco=gps.time.second();
75 int alt=gps.altitude.feet()/3.2808;
76 String con="(hora, lati)";
77 // send packet
78 LoRa.beginPacket();
79 LoRa.println(counter);
80 LoRa.println("latitud");
81 LoRa.println(lati,5);
82 LoRa.println("longitud");
83 LoRa.println(lon,5);
84 LoRa.println("Hora");
85 LoRa.print(hora);
86 LoRa.print(":");
87 LoRa.print(minu);
88 LoRa.print(":");
89 LoRa.print(seco);
90 LoRa.println("altitud");
91 LoRa.println(alt,5);
92 LoRa.endPacket();
93
94 counter++;
95 //digitalWrite(2, HIGH); // turn the LED on (HIGH is the
    voltage level)
96 //delay(1000); // wait for a second

```

```

97 // digitalWrite(2, LOW); // turn the LED off by making the
    voltage LOW
98 //delay(1000); // wait for a second
99 }
100 static void smartDelay(unsigned long ms)
101 {
102     unsigned long start = millis();
103     do
104     {
105         while (Serial1.available())
106             gps.encode(Serial1.read());
107     } while (millis() - start < ms);
108 }

```

Listado de codigos C.1: Transmisor LoRa

C.1.2. Receptor

```

1 #include <SPI.h> //Libreria que permite la comunicacion con la
    interfaz de perifericos en serie
2 #include <LoRa.h> //Libreria LoRa para enviar y recibir datos
    usando radios LoRa
3 #include <Wire.h> //Libreria que permite comunicarse con I2C
4
5 #define SCK      5 // GPIO5 — SX1278's SCK
6 #define MISO     19 // GPIO19 — SX1278's MISO
7 #define MOSI    27 // GPIO27 — SX1278's MOSI
8 #define SS      18 // GPIO18 — SX1278's CS
9 #define RST     14 // GPIO14 — SX1278's RESET
10 #define DIO     26 // GPIO26 — SX1278's IRQ(Interrupt Request)
11 #define BAND    915E6 //Banda de transmision LoRa
12
13
14 String rssi = "RSSI --";
15 String packSize = "--";
16 String packet ;
17
18 void loraData(){
19     Serial.println(rssi); //impresion del valor de RSSI recibido
20     Serial.println(packet); //impresion del paquete enviado
21 }
22
23 void cbk(int packetSize) {

```

```

24  packet ="";
25  packetSize = String(packetSize,DEC);
26  for (int i = 0; i < packetSize; i++) { packet += (char) LoRa.read
    (); }
27  rssi = "RSSI " + String(LoRa.packetRssi(), DEC) ;
28  loraData();
29  }
30
31  void setup() {
32    pinMode(16,OUTPUT);
33    digitalWrite(16, LOW); // set GPIO16 low to reset OLED
34    delay(50);
35    digitalWrite(16, HIGH);
36
37    Serial.begin(115200); //configuracion del puerto serial
38    while (!Serial);
39    Serial.println();
40    Serial.println("LoRa Receiver Callback");
41    SPI.begin(SCK,MISO,MOSI,SS);
42    LoRa.setPins(SS,RST,DI0);
43    if (!LoRa.begin(915E6)) {
44      Serial.println("Starting LoRa failed!");
45      while (1);
46    }
47    //LoRa.onReceive(cbk);
48    LoRa.setSpreadingFactor(7); // configuracion del valor del
    spreading factor
49    LoRa.setFrequency(920E6); // Frecuencia a la cual se recibe el
    pulso modulado
50    LoRa.receive();
51    Serial.println("init ok");
52    // display.init();
53    // display.flipScreenVertically();
54    // display.setFont(ArialMT_Plain_10);
55
56    delay(1500);
57  }
58
59  void loop() {
60    int packetSize = LoRa.parsePacket();
61    if (packetSize) { cbk(packetSize); }
62    delay(10);
63  }

```

C.2. Códigos en Matlab para el cálculo de parámetros del modelo para la caracterización de RSSI

C.2.1. Programa LOS para la primera campaña de medidas

```
1 clear all
2 close all
3 %format long
4 M1=csvread("lv1.csv");
5 M2=csvread("lv2.csv");
6 M3=xlsread("lv3.xlsx");
7 M4=xlsread("lv4.xlsx");
8 M5=xlsread("lv5.xlsx");
9 M6=xlsread("lv6.xlsx");
10 M7=xlsread("lv7.xlsx");
11 M8=xlsread("lv8.xlsx");
12 M9=xlsread("lv9.xlsx");
13 M10=xlsread("lv10.xlsx");
14 M11=xlsread("lv11.xlsx");
15 m1=mean(M1);
16 m2=mean(M2);
17 m3=mean(M3);
18 m4=mean(M4);
19 m5=mean(M5);
20 m6=mean(M6);
21 m7=mean(M7);
22 m8=mean(M8);
23 m9=mean(M9);
24 m10=mean(M10);
25 m11=mean(M11);
26 Y=[m1; m2; m3; m4; m5; m6; m7; m8; m9; m10; m11];
27 x=3:3:33;
28 X=[x.',ones(11,1)];
29 P=inv(X.'*X)*X.'*Y;
30 n1=-P(1,1)/10
31 b1=P(2,1)
32 Ys=X*P;
33 E=Y-Ys;
```

```

34 E1=Ys-mean(Y);
35 SSE=E.'*E;
36 SSR=E1.'*E1;
37 SST=SSR+SSE;
38 R2=SSR/SST
39 u=mean(Y)
40 sig2=SSE/(length(Y)-2);
41 desv=sqrt(sig2)
42 figure
43 plot(x,Y,'*')
44 grid on
45 title("RSSI-Linea de vista")
46 xlabel("distancia (m)")
47 ylabel("RSSI(dBm)")
48 hold on
49 plot(x,Ys,'r')
50 legend('Datos experimentales','Modelo log-normal de una sola
pendiente')

```

C.2.2. Programa NLOS para la primera campaña de medidas

```

1 clc
2 clear all
3 close all
4 format long
5 M1=csvread("n11.csv",1); %lectura de datos
6 M2=csvread("n12.csv",1);
7 M3=csvread("n13.csv",1);
8 M4=csvread("n14.csv",1);
9 M5=csvread("n15.csv",1);
10 m1=mean(M1);
11 m2=mean(M2);
12 m3=mean(M3);
13 m4=mean(M4);
14 m5=mean(M5);
15 x=[38.72,47.40,58.35,68.63,74.85];
16 x1=log10(x/38.72);
17 X=[x1.',ones(length(x),1)];
18 Y=[m1; m2; m3; m4; m5];
19 P=inv(X.'*X)*X.'*Y;
20 n1=-P(1,1)/10
21 b1=P(2,1)
22 Ys=X*P;
23 E=Y-Ys;

```

```

24 E1=Ys - mean(Y);
25 SSE=E.'*E;
26 SSR=E1.'*E1;
27 SST=SSR+SSE;
28 R2=SSR/SST
29 u=mean(Y)
30 sig2=SSE/(length(Y)-2);
31 desv=sqrt(sig2)
32
33 figure
34 plot(x,Y,'*')
35 grid on
36 title("RSSI-NLOS")
37 xlabel("distancia (m)")
38 ylabel("RSSI (dBm)")
39 hold on
40 plot(x,Ys,'r')
41 legend('Datos experimentales','Modelo log-normal de una sola
pendiente')

```

C.2.3. Programa NLOS.m

```

1 clc
2 clear all
3 close all
4 format long
5 M1=csvread("2.csv",1);
6 M2=csvread("3.csv",1);
7 M3=csvread("4.csv",1);
8 M4=csvread("5.csv",1);
9 M5=csvread("6.csv",1);
10 M6=csvread("7.csv",1);
11 M7=csvread("8.csv",1);
12 M8=csvread("9.csv",1);
13 M9=csvread("10.csv",1);
14 M10=csvread("11.csv",1);
15 m1=mean(M1);
16 m2=mean(M2);
17 m3=mean(M3);
18 m4=mean(M4);
19 m5=mean(M5);
20 m6=mean(M6);
21 m7=mean(M7);
22 m8=mean(M8);

```



```

23 m9=mean(M9);
24 m10=mean(M10);
25 %modelo simple
26 X
    = [14.08, 24.15, 43.06, 59.35, 83.07, 118.63, 130.18, 136.87, 148.39, 157.56];

27 d0=14.08;
28 d01=1;
29 x1=log10(x/d0);
30 X=[x1.', ones(length(x),1)];
31 Y=[m1;m2; m3;; m4; m5; m6;m7;m8;m9;m10];
32 P=inv(X.'*X)*X.'*Y
33 n1=-P(1,1)/10
34 b1=P(2,1)
35 Ys=X*P;
36 E=Y-Ys;
37 E1=Ys-mean(Y);
38 SSE=E.'*E;
39 SSR=E1.'*E1;
40 SST=SSR+SSE;
41 R2=SSR/SST
42 u=mean(Y)
43 sig2=SSE/(length(Y)-2);
44 desv=sqrt(sig2)
45 %-----
46 %doble pendiente
47 %-----
48 %primera recta
49 x2=x(1:6);
50 xn1=log10(x2/d01);
51 X1=[xn1.', ones(length(x2),1)];
52 Y2=Y(1:6);
53 P2=inv(X1.'*X1)*X1.'*Y2
54 n11=-P2(1,1)/10
55 b11=P2(2,1)
56 Ys1=X1*P2;
57 E1=Y2-Ys1;
58 E11=Ys1-mean(Y2);
59 SSE1=E1.'*E1;
60 SSR1=E11.'*E11;
61 SST1=SSR1+SSE1;
62 R21=SSR1/SST1
63 u1=mean(Y2)

```

```

64 sig21=SSE1/(length(Y2)-2);
65 desv1=sqrt(sig21)
66 %segunda recta
67 x3=x(6:10);
68 xn2=log10(x3/d01);
69 X2=[xn2.',ones(length(x3),1)];
70 Y3=Y(6:10);
71 P3=inv(X2.'*X2)*X2.'*Y3
72 n12=-P3(1,1)/10
73 b12=P3(2,1)
74 Ys2=X2*P3;
75 E2=Y3-Ys2;
76 E12=Ys2-mean(Y3);
77 SSE2=E2.'*E2;
78 SSR2=E12.'*E12;
79 SST2=SSR2+SSE2;
80 R22=SSR2/SST2
81 u2=mean(Y3)
82 sig22=SSE2/(length(Y3)-2);
83 desv2=sqrt(sig22)
84 %GRAFICOS
85 %EXP=[M1 M2 M3 M4 M5 M6 M7 M8 M9 M10];
86 figure
87 plot(x,Y,'o--')
88 grid on
89 title("RSSI")
90 xlabel("distance (m)")
91 ylabel("RSSI (dBm)")
92 hold on
93 plot(x,Ys,'g')
94 plot(x2,Ys1,'r')
95 plot(x3,Ys2,'r')
96 legend('Experimental Data','One-slope log-normal model',
97 , 'Two-slope log-normal model')

```

C.2.4. Programa LOBS.m

```

1 clc
2 clear all
3 close all
4 format long
5 M1=csvread("1b.csv",1);
6 M2=csvread("2b.csv",1);
7 M3=csvread("3b.csv",1);

```

```

8 M4=csvread("4b.csv",1);
9 M5=csvread("5b.csv",1);
10 M6=csvread("6b.csv",1);
11 M7=csvread("7b.csv",1);
12 M8=csvread("8b.csv",1);
13 M9=csvread("9b.csv",1);
14 M10=csvread("10b.csv",1);
15 m1=mean(M1);
16 m2=mean(M2);
17 m3=mean(M3);
18 m4=mean(M4);
19 m5=mean(M5);
20 m6=mean(M6);
21 m7=mean(M7);
22 m8=mean(M8);
23 m9=mean(M9);
24 m10=mean(M10);
25
26 x
    = [33.13, 52.84, 69.80, 84.82, 96.78, 137.99, 168.20, 194.20, 218.30, 241.73];

27 d0=33.13;
28 d01=1;
29 x1=log10(x/d0);
30 X=[x1.', ones(length(x),1)];
31 Y=[m1;m2; m3;; m4; m5; m6;m7;m8;m9;m10];
32 P=inv(X.'*X)*X.'*Y
33 n1=-P(1,1)/10
34 b1=P(2,1)
35 Ys=X*P;
36 E=Y-Ys;
37 E1=Ys-mean(Y);
38 SSE=E.'*E;
39 SSR=E1.'*E1;
40 SST=SSR+SSE;
41 R2=SSR/SST
42 u=mean(Y)
43 sig2=SSE/(length(Y)-2);
44 desv=sqrt(sig2)
45 %-----
46 %doble pendiente
47 %-----
48 %primera recta

```

```

49 x2=x(1:5);
50 xn1=log10(x2/d01);
51 X1=[xn1.',ones(length(x2),1)];
52 Y2=Y(1:5);
53 P2=inv(X1.'*X1)*X1.'*Y2
54 n11=-P2(1,1)/10
55 b11=P2(2,1)
56 Ys1=X1*P2;
57 E1=Y2-Ys1;
58 E11=Ys1-mean(Y2);
59 SSE1=E1.'*E1;
60 SSR1=E11.'*E11;
61 SST1=SSR1+SSE1;
62 R21=SSR1/SST1
63 u1=mean(Y2)
64 sig21=SSE1/(length(Y2)-2);
65 desv1=sqrt(sig21)
66 %segunda recta
67 x3=x(5:10);
68 xn2=log10(x3/d01);
69 X2=[xn2.',ones(length(x3),1)];
70 Y3=Y(5:10);
71 P3=inv(X2.'*X2)*X2.'*Y3
72 n12=-P3(1,1)/10
73 b12=P3(2,1)
74 Ys2=X2*P3;
75 E2=Y3-Ys2;
76 E12=Ys2-mean(Y3);
77 SSE2=E2.'*E2;
78 SSR2=E12.'*E12;
79 SST2=SSR2+SSE2;
80 R22=SSR2/SST2
81 u2=mean(Y3)
82 sig22=SSE2/(length(Y3)-2);
83 desv2=sqrt(sig22)
84
85 figure
86 plot(x,Y,'o--')
87 grid on
88 title("RSSI")
89 xlabel("distance (m)")
90 ylabel("RSSI (dBm)")
91 hold on

```

```

92 plot(x, Ys, 'g')
93 plot(x2, Ys1, 'r')
94 plot(x3, Ys2, 'r')
95 legend('Experimental Data', 'One-slope log-normal model', 'Two-slope
    log-normal model')

```

C.2.5. Programa retardo2.m

```

1 close all;
2 clear all;
3 format long;
4 m1=csvread('M60.csv'); %lectura de files
5 m2=csvread('M601.csv'); %entrada de datos
6 m3=csvread('M602.csv');
7 m4=csvread('M603.csv');
8 m5=csvread('M301.csv'); %lectura de files
9 m6=csvread('M302.csv'); %entrada de datos
10 m7=csvread('M303.csv');
11 m8=csvread('M304.csv');
12 m9=csvread('MUR0.csv'); %lectura de files
13 m10=csvread('MUR02.csv'); %entrada de datos
14 m11=csvread('MUR03.csv');
15 m12=csvread('MUR04.csv');
16 x=m1(:,1);
17 y=(m9(:,2)+m10(:,2)+m11(:,2)+m12(:,2))/4; %promedio de datos
18 %y=m12(:,2);
19 y1=10.^(y/10); %conversion a mw
20 ts=1/(x(2)-x(1)); %tiempo entre muestras
21 fs=1/(ts); %frecuencia de muestreo
22 w=hamming(70); %filtro de hamming en el tiempo discreto
23 W=fft(w).'; %filtro de hamming en la frecuencia
24 H=conv(W,y1); % H(w)
25 %H=y1;
26 h=ifft(H); %respuesta al impulso
27 h2=10.^(h/10); %conversion a db
28 h3=abs(h); %pasos opcionales
29 pd=(h3).^2;
30 pdpn=pd/max(pd);
31 pdp=10.*log10(pdpn);
32 n1=floor(length(pdp)/2);
33 n=n1 % valor del numero de muestra del umbral
34
35 %x1=(0:348)*(ts)/length(y);
36 x1=(0:1:(length(pdp)-1))*(ts)/length(y);

```

```

37 pdpu=[];
38 %-----PDP CON EL UMBRAL .....
39 %UMBRALE PDP ADIMENSIONAL
40 for i=1:1:n1
41     if pdpn(i)>0.01
42         pdput(i)=pdpn(i);
43     else
44         pdput(i)=0.01;
45     end
46 end
47 %UMBRALE PDP en dB
48 for i=1:1:n1
49     if pdp(i)>-20
50         pdpu(i)=pdp(i);
51     else
52         pdpu(i)=-20;
53     end
54 end
55 %-----
56 pdpt=sum(pdput(1:n));
57 tau=x1;
58 %--- CALCULO DEL TIEMPO DE RETARDO ---
59 for i=1:n
60     tfm=0; %first moment of tau
61     tfmi=(tau(i)*pdput(i))/(pdpt);
62     tfm=tfm+tfmi;
63 end
64 for i=1:n
65     tsm=0; %second moment of tau
66     tsmi=((tau(i)^2)*pdput(i))/(pdpt);
67     tsm=tsm+tsmi;
68 end
69 pdpt
70 a=tfm
71 b=tsm
72 trms=sqrt(tsm-tfm^2)
73 %-----
74 x2=(0:1:(n1-1))*ts/length(y);
75 u=(-20)*ones(length(x2));
76 pdpumbrale=pdpu.';
77 %---GRAFICOS DEL PDP-----
78 figure
79 plot(x2,pdp(1:n1))

```

```

80 hold on
81 plot(x2,u,'r')
82 plot(x2,pdpumbral)
83 grid
84 title('Perfil de Retardo de Potencia')
85 ylabel('Potencia relativa normalizada (dB)')
86 xlabel('Tiempo de Retardo(s)')
87 legend('datos medidos','umbral')
88 figure
89 plot(x2,pdpu(1:n1))
90 hold on
91 plot(x2,pdput)
92 title('Perfil de Retardo de Potencia')
93 ylabel('Potencia relativa normalizada')
94 xlabel('Tiempo de Retardo(s)')
95 grid on
96 %-----

```

C.3. Notebooks de Python

Modelado arma

June 6, 2021

0.1 NOTEBOOK 1

0.1.1 MODELADO ARMA DEL PÁRAMETRO S_{21} DEL MURO DE PRUEBA 1, $d = 30\text{cm}$

Primera parte, declaracion de librerias e importacion de datos

```
[1]: import pandas as pd
import numpy as np
import statsmodels as st #libreria encargada de todos los metodos estadisticos
import matplotlib.pyplot as plt #libreria para graficos
from statsmodels.tsa.arima_model import ARMA # libreria para los modelos
    ↪ arma, ar, ma
from statsmodels.tsa.stattools import adfuller #criterio de estacionariedad de
    ↪ una serie de tiempo
from statsmodels.tsa.stattools import arma_order_select_ic #criterio de akaike
    ↪ para la seleccion del mejor modelo
from statsmodels.graphics.tsaplots import plot_pacf #sample partial
    ↪ autocorrelation function
from statsmodels.graphics.tsaplots import plot_acf
import warnings
warnings.filterwarnings(action='ignore', category=FutureWarning)
```

```
[17]: datacsv1=pd.read_excel('M301.xlsx') # lectura del archivo
datacsv2=pd.read_excel('M302.xlsx') # lectura del archivo
datacsv3=pd.read_excel('M303.xlsx') # lectura del archivo
datacsv4=pd.read_excel('M304.xlsx') # lectura del archivo
data1=np.array(datacsv1)
data2=np.array(datacsv2)
data3=np.array(datacsv3)
data4=np.array(datacsv4)
#print(data)
freq=data1[:,0] #asignacion de valores a los ejes
y=(data1[:,1]+data2[:,1]+data3[:,1]+data4[:,1])/4
y3=np.power(10,y/10) #convirtiendo a milivatios
y1=y3/np.max(y3) #normalizacion de la señal
plt.plot(freq,y1) #grafico de la señal adquirida
plt.xlabel('Frecuencia (Hz)')
plt.ylabel('Intensidad en milivatios (mW) normalizada')
```



```
plt.title('Párametros  $S_{21}$ ')
plt.legend(['Datos obtenidos'])
plt.grid()
plt.savefig("muro130.jpg",dpi=500)
```

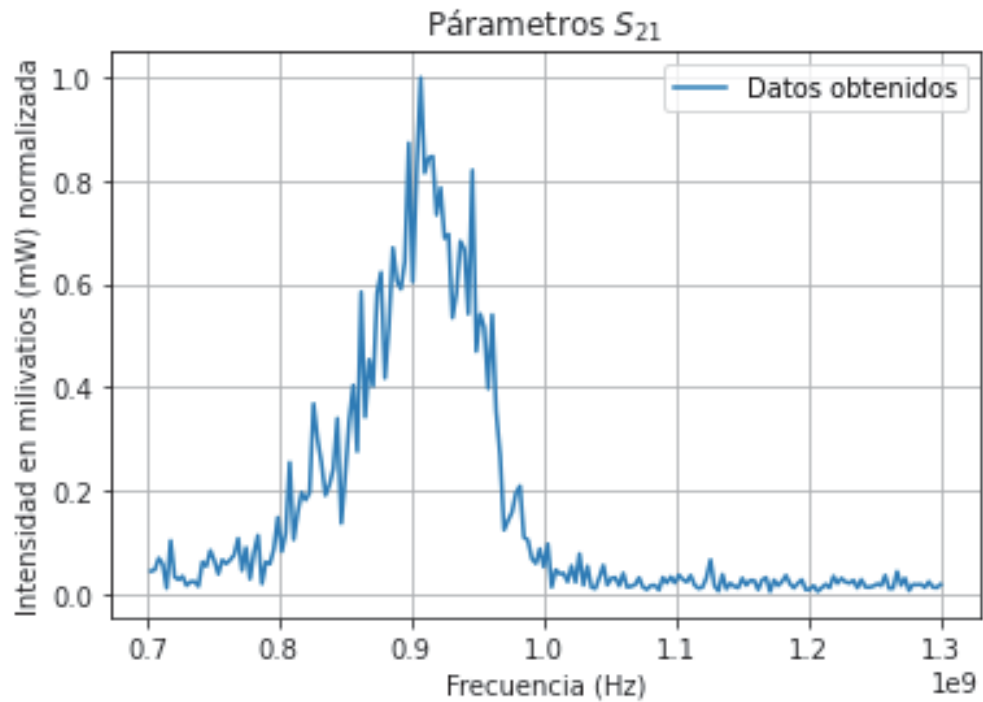
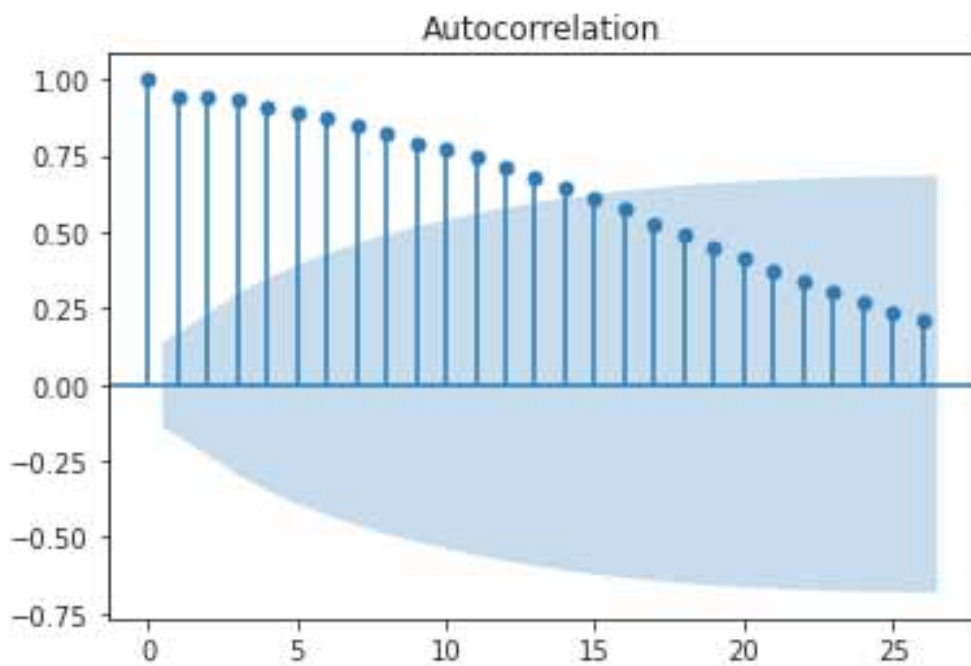
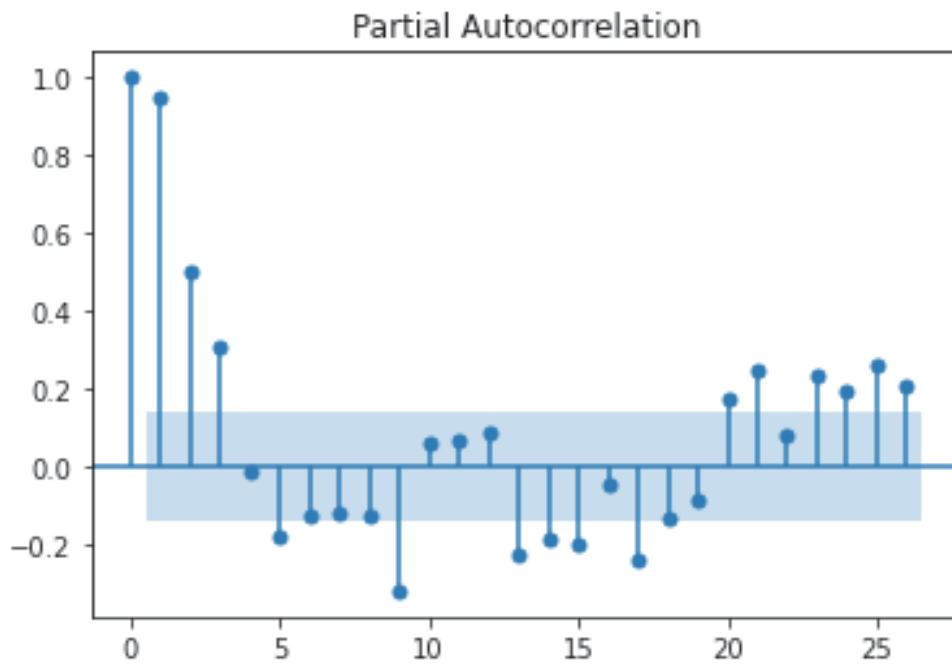


Grafico de la sample partial autocorrelation function (PACF)

```
[18]: plot_pacf(y1, lags=26);
#plt.savefig("autocorrelacion_parcial30.jpg",dpi=500)
plot_acf(y1, lags=26);
#plt.savefig("autocorrelacion30.jpg",dpi=500)
```



Criterio de Akaike para seleccionar el orden del proceso ARMA

```
[19]: res=arma_order_select_ic(y1, ic=["aic", "bic"], trend="nc")
print('el orden del modelo es',res.aic_min_order)
```

el orden del modelo es (4, 1)

Cálculo de los parametros del modelo ARMA con el orden anterior

```
[20]: model=ARMA(y1,order=(4,1))
model_fit=model.fit() #este comando calcula los parametros del modelo usando
↳ filtro de kalman
print(model_fit.summary())
y2=model_fit.predict()
```

ARMA Model Results

```
=====
Dep. Variable:          y      No. Observations:          200
Model:                 ARMA(4, 1)  Log Likelihood          267.056
Method:                css-mle    S.D. of innovations      0.063
Date:                  Sun, 06 Jun 2021  AIC                    -520.112
Time:                  21:36:05      BIC                      -497.024
Sample:                0           HQIC                     -510.769
=====
```

```
=====
              coef      std err          z      P>|z|      [0.025      0.975]
-----
const          0.1567      0.068          2.301      0.021      0.023      0.290
ar.L1.y         1.2587      0.078         16.052      0.000      1.105      1.412
ar.L2.y          0.0627      0.114          0.550      0.582     -0.161      0.286
ar.L3.y          0.0047      0.113          0.041      0.967     -0.218      0.227
ar.L4.y        -0.3321      0.067         -4.971      0.000     -0.463     -0.201
ma.L1.y        -0.9097      0.051        -17.779      0.000     -1.010     -0.809
=====
```

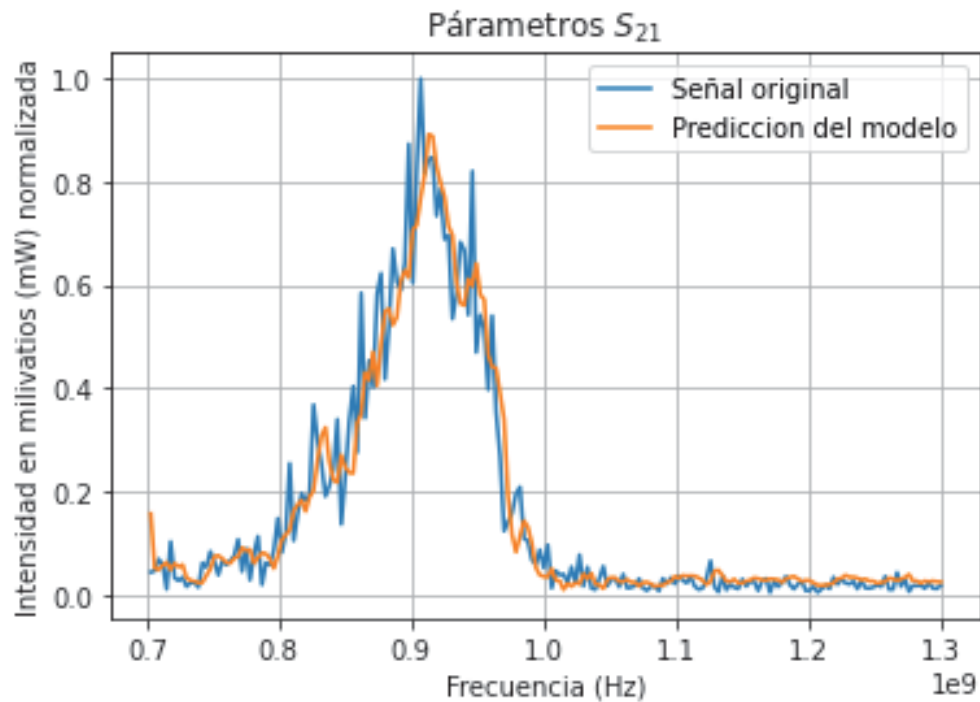
Roots

```
=====
              Real          Imaginary      Modulus      Frequency
-----
AR.1          1.0188          -0.0519j          1.0201          -0.0081
AR.2          1.0188           +0.0519j          1.0201           0.0081
AR.3         -1.0117          -1.3674j          1.7010          -0.3514
AR.4         -1.0117           +1.3674j          1.7010           0.3514
MA.1          1.0992           +0.0000j          1.0992           0.0000
=====
```

Grafico de comparacion de los datos obtenidos y la prediccion del modelo

```
[21]: plt.plot(freq,y1)
plt.plot(freq,y2)
plt.xlabel('Frecuencia (Hz)')
plt.ylabel('Intensidad en milivatios (mW) normalizada')
plt.grid()
```

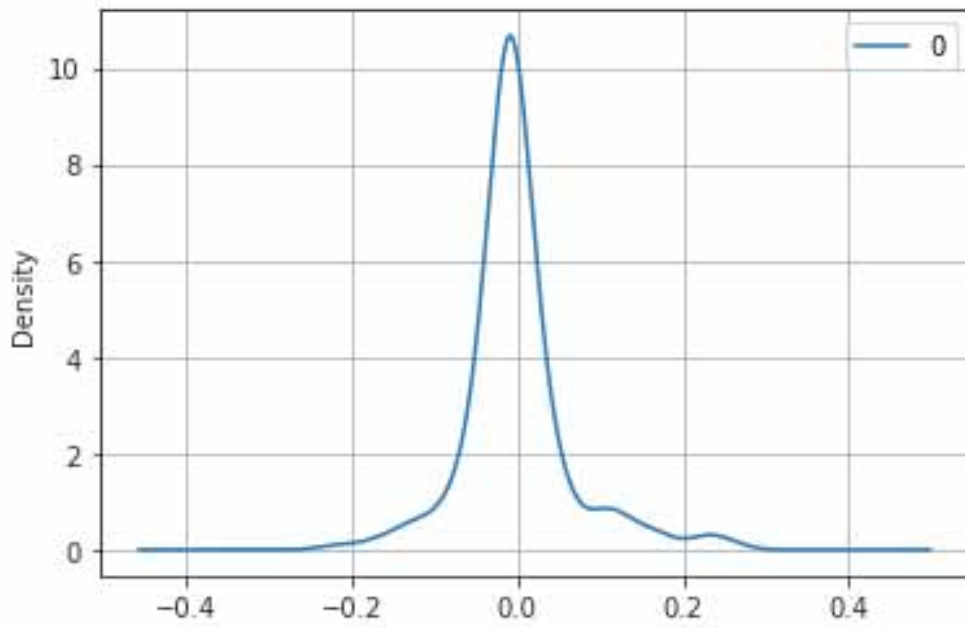
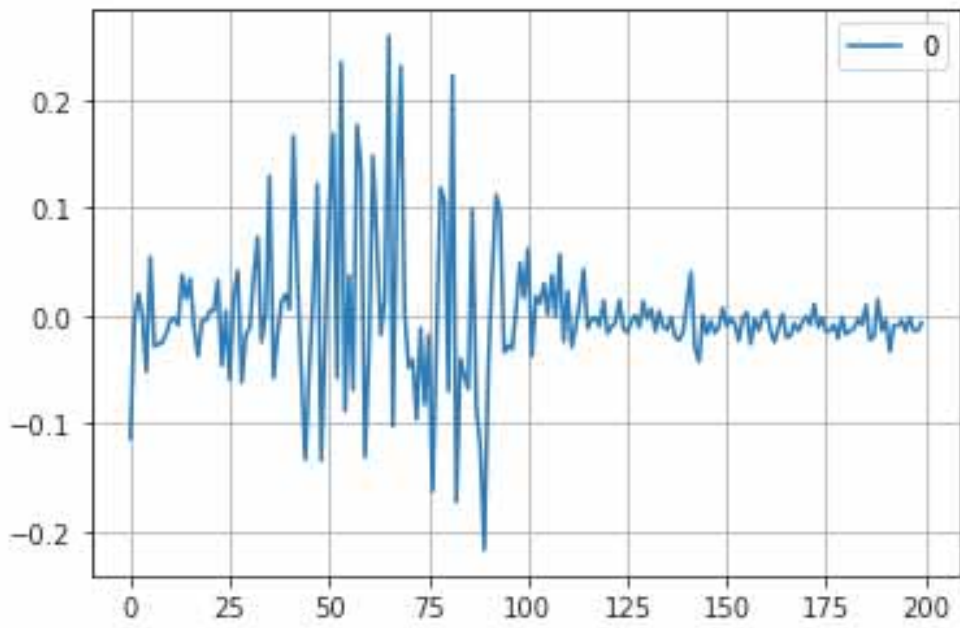
```
plt.legend(['Señal original', 'Predicción del modelo'])
plt.title('Parámetros $S_{21}$')
plt.savefig("prediccion30.jpg",dpi=500)
```



Análisis de los residuos de la estimación del modelo

```
[22]: # plot residual errors
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.grid()
plt.savefig("residuos30.jpg",dpi=500)
residuals.plot(kind='kde')
plt.grid()
plt.savefig("residuos301.jpg",dpi=500)
plt.show()

print(residuals.describe())
```



```
0
count 200.000000
mean   0.000049
std    0.063780
min    -0.216674
```

25%	-0.021314
50%	-0.007634
75%	0.012387
max	0.259192

[]:

Modelado arma

June 6, 2021

0.1 NOTEBOOK 2

0.1.1 MODELADO ARMA DEL PÁRAMETRO S_{21} DEL MURO DE PRUEBA 1, $d = 60\text{cm}$

Primera parte, declaracion de librerias e importacion de datos

```
[23]: import pandas as pd
import numpy as np
import statsmodels as st #libreria encargada de todos los metodos estadisticos
import matplotlib.pyplot as plt #libreria para graficos
from statsmodels.tsa.arima_model import ARMA # libreria para los modelos
    ↪ arma, ar, ma
from statsmodels.tsa.stattools import adfuller #criterio de estacionariedad de
    ↪ una serie de tiempo
from statsmodels.tsa.stattools import arma_order_select_ic #criterio de akaike
    ↪ para la seleccion del mejor modelo
from statsmodels.graphics.tsaplots import plot_pacf #sample partial
    ↪ autocorrelation function
from statsmodels.graphics.tsaplots import plot_acf
import warnings
warnings.filterwarnings(action='ignore', category=FutureWarning)
```

```
[25]: datacsv1=pd.read_excel('M601.xlsx') # lectura del archivo
datacsv2=pd.read_excel('M602.xlsx') # lectura del archivo
datacsv3=pd.read_excel('M603.xlsx') # lectura del archivo
datacsv4=pd.read_excel('M604.xlsx') # lectura del archivo
data1=np.array(datacsv1)
data2=np.array(datacsv2)
data3=np.array(datacsv3)
data4=np.array(datacsv4)
#print(data)
freq=data1[:,0] #asignacion de valores a los ejes
y=(data1[:,1]+data2[:,1]+data3[:,1]+data4[:,1])/4
y3=np.power(10,y/10) #convirtiendo a milivatios
y1=y3/np.max(y3) #normalizacion de la señal
plt.plot(freq,y1) #grafico de la señal adquirida
plt.xlabel('Frecuencia (Hz)')
plt.ylabel('Intensidad en milivatios (mW) normalizada')
```

```
plt.title('Párametros  $S_{21}$ ')
plt.legend(['Datos obtenidos'])
plt.grid()
plt.savefig("muro160.jpg",dpi=500)
```

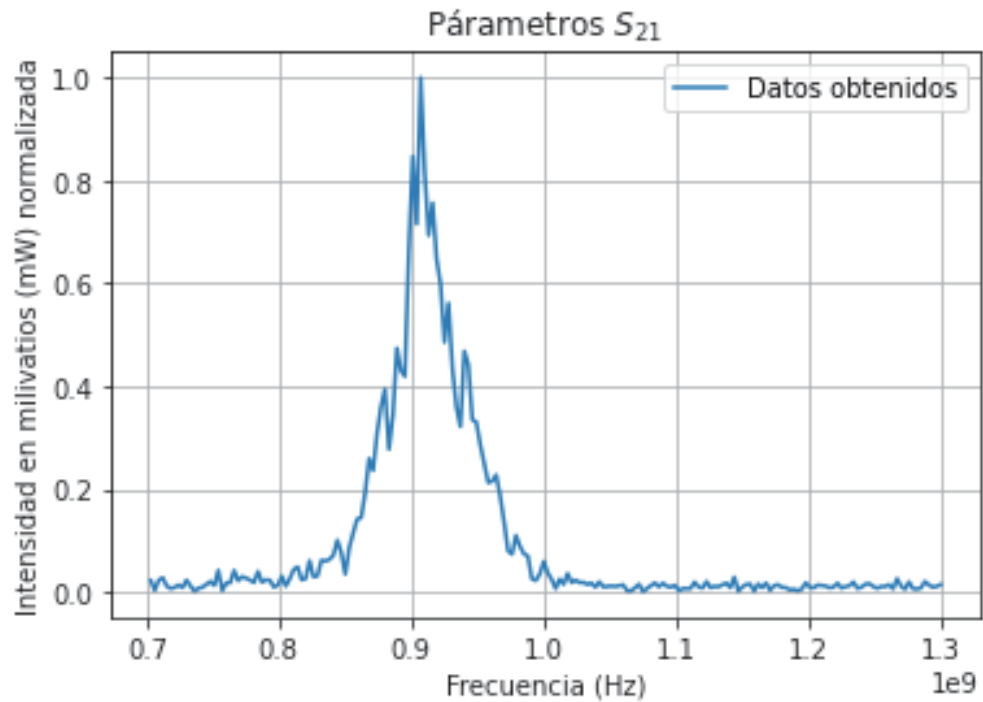
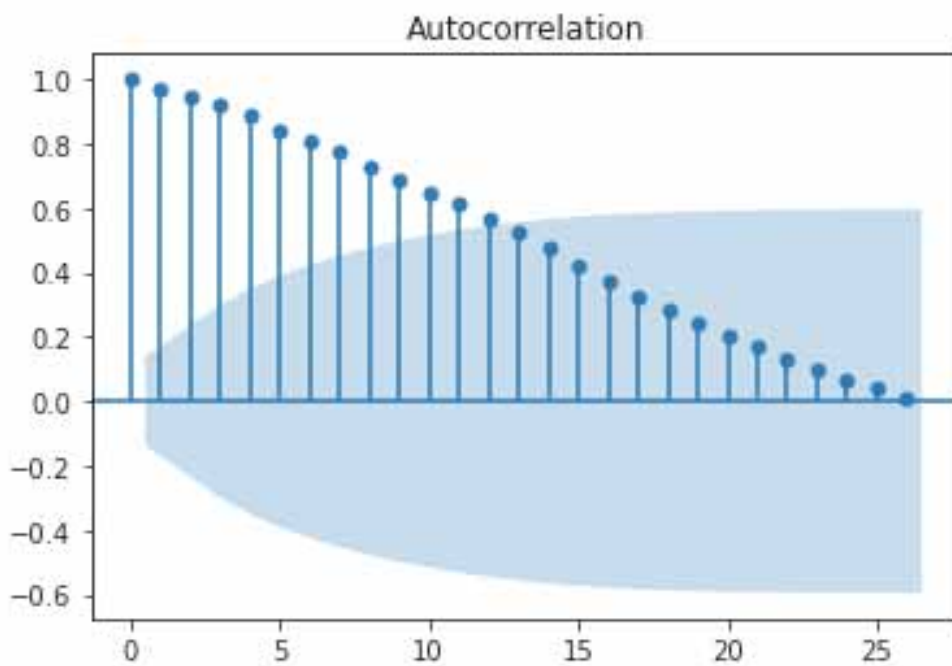
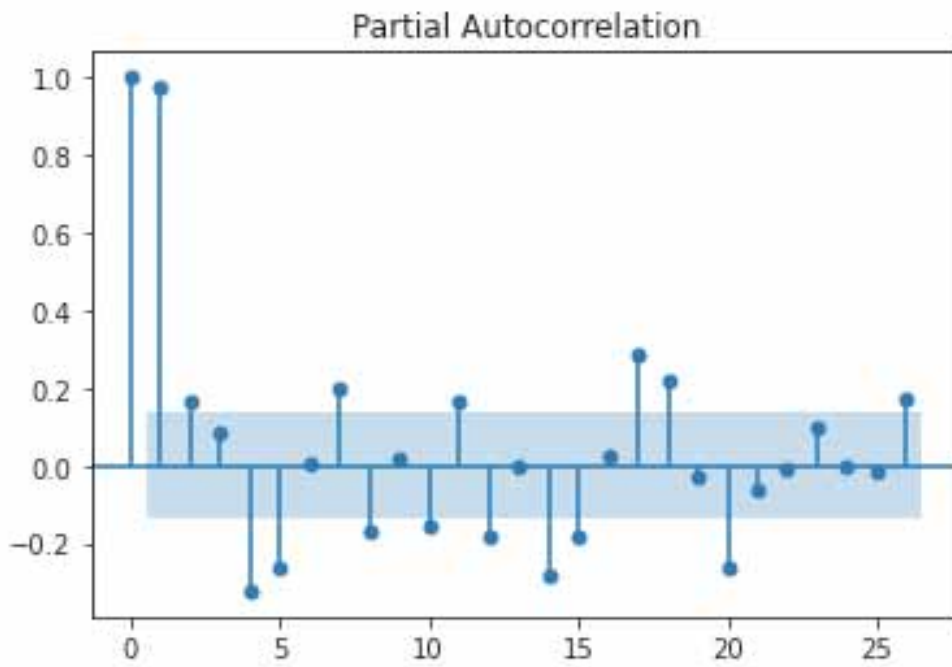


Gráfico de la sample partial autocorrelation function (PACF)

```
[26]: plot_pacf(y1, lags=26);
plt.savefig("autocorrelacion_parcial60.jpg",dpi=500)
plot_acf(y1, lags=26);
plt.savefig("autocorrelacion60.jpg",dpi=500)
```

Criterio de Akaike para seleccionar el orden del proceso ARMA

```
[27]: res=arma_order_select_ic(y1, ic=["aic", "bic"], trend="nc")
print('el orden del modelo es',res.aic_min_order)
```

el orden del modelo es (4, 2)

Cálculo de los parametros del modelo ARMA con el orden anterior

```
[28]: model=ARMA(y1,order=(4,2))
model_fit=model.fit() #este comando calcula los parametros del modelo usando
↳ filtro de kalman
print(model_fit.summary())
y2=model_fit.predict()
```

ARMA Model Results

```
=====
Dep. Variable:          y      No. Observations:          200
Model:                 ARMA(4, 2)  Log Likelihood          347.632
Method:                css-mle    S.D. of innovations      0.042
Date:                  Sun, 06 Jun 2021  AIC                    -679.263
Time:                  21:54:29      BIC                     -652.877
Sample:                0           HQIC                    -668.585
=====
```

```
=====
              coef      std err          z      P>|z|      [0.025      0.975]
-----
const          0.0797      0.080          1.000      0.318      -0.077      0.236
ar.L1.y        1.2626      0.136          9.275      0.000      0.996      1.529
ar.L2.y       -0.8448      0.233         -3.626      0.000     -1.301     -0.388
ar.L3.y        0.7324      0.112          6.533      0.000      0.513      0.952
ar.L4.y       -0.1915      0.103         -1.865      0.062     -0.393      0.010
ma.L1.y       -0.4799      0.130         -3.694      0.000     -0.735     -0.225
ma.L2.y        0.7109      0.176          4.030      0.000      0.365      1.057
=====
```

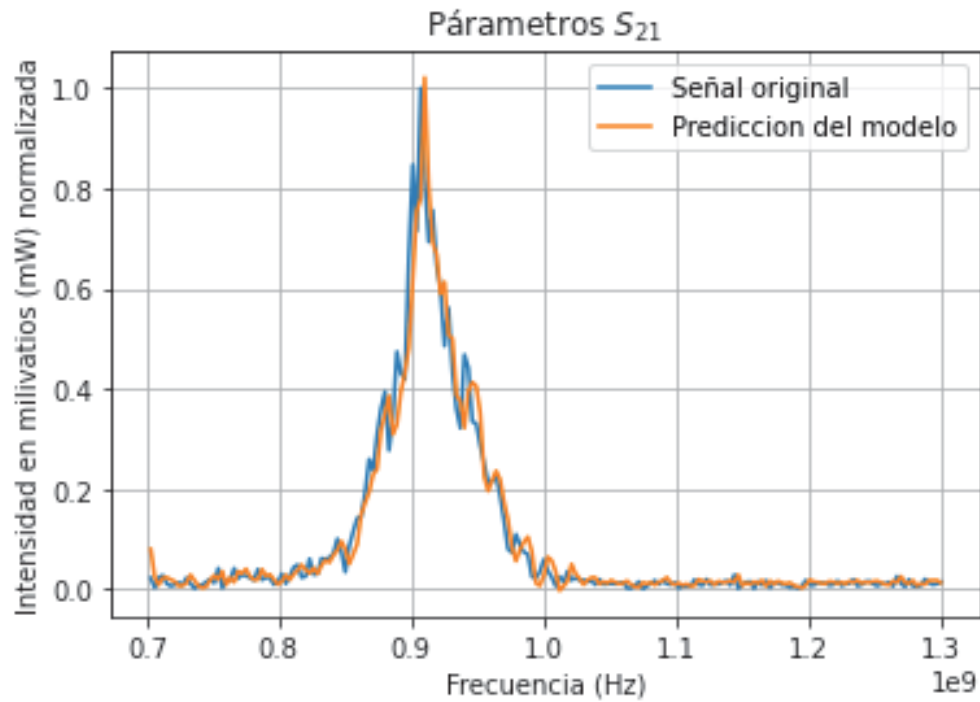
Roots

```
=====
              Real          Imaginary      Modulus      Frequency
-----
AR.1          1.0408          -0.0000j          1.0408          -0.0000
AR.2         -0.0412          -1.3224j          1.3230          -0.2550
AR.3         -0.0412           +1.3224j          1.3230           0.2550
AR.4          2.8660          -0.0000j          2.8660          -0.0000
MA.1          0.3376          -1.1370j          1.1861          -0.2041
MA.2          0.3376           +1.1370j          1.1861           0.2041
=====
```

Grafico de comparacion de los datos obtenidos y la prediccion del modelo

```
[29]: plt.plot(freq,y1)
plt.plot(freq,y2)
plt.xlabel('Frecuencia (Hz)')
```

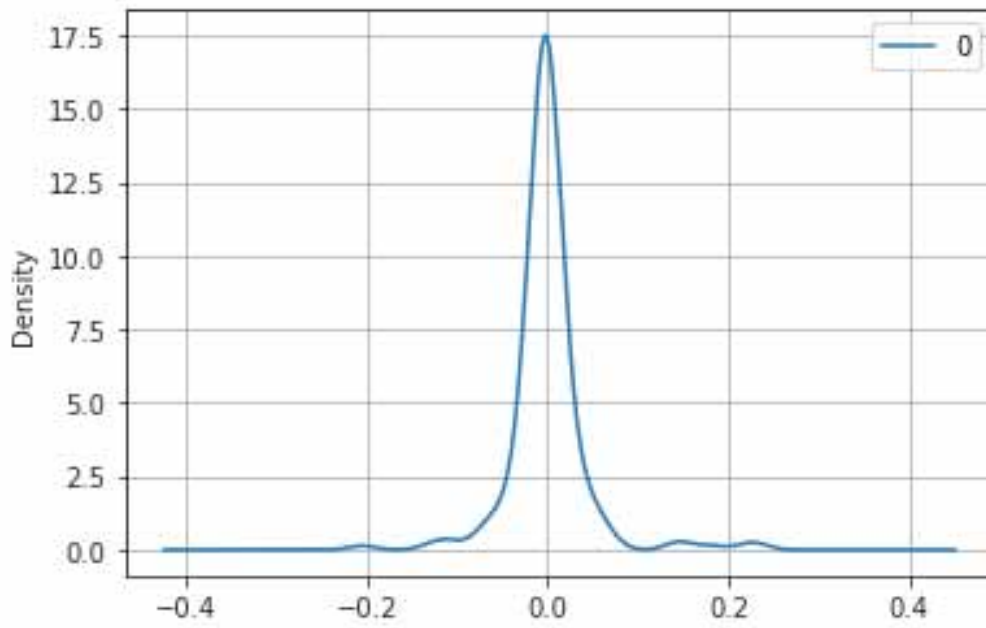
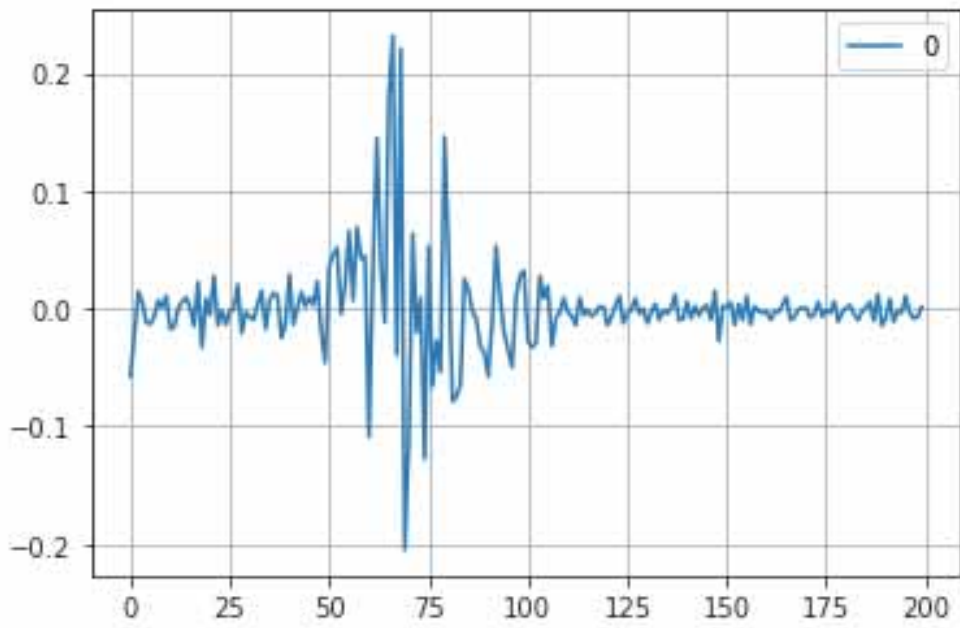
```
plt.ylabel('Intensidad en milivatios (mW) normalizada')
plt.grid()
plt.legend(['Señal original', 'Prediccion del modelo'])
plt.title('Párametros $S_{21}$')
plt.savefig("prediccion60.jpg", dpi=500)
```



Análisis de los residuos de la estimación del modelo

```
[30]: # plot residual erros
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.grid()
plt.savefig("residuos60.jpg", dpi=500)
residuals.plot(kind='kde')
plt.grid()
plt.savefig("residuos601.jpg", dpi=500)
plt.show()

print(residuals.describe())
```



```
0
count 200.000000
mean  0.000242
std   0.042467
min   -0.204924
```

25%	-0.010755
50%	-0.002077
75%	0.008858
max	0.231747

[]:

Modelado arma

June 6, 2021

0.1 NOTEBOOK 3

0.1.1 MODELADO ARMA DEL PÁRAMETRO S_{21} DEL MURO DE PRUEBA 2, $d = 60\text{cm}$

Primera parte, declaracion de librerias e importacion de datos

```
[23]: import pandas as pd
import numpy as np
import statsmodels as st #libreria encargada de todos los metodos estadisticos
import matplotlib.pyplot as plt #libreria para graficos
from statsmodels.tsa.arima_model import ARMA # libreria para los modelos
    ↪ arma, ar, ma
from statsmodels.tsa.stattools import adfuller #criterio de estacionariedad de
    ↪ una serie de tiempo
from statsmodels.tsa.stattools import arma_order_select_ic #criterio de akaike
    ↪ para la seleccion del mejor modelo
from statsmodels.graphics.tsaplots import plot_pacf #sample partial
    ↪ autocorrelation function
from statsmodels.graphics.tsaplots import plot_acf
import warnings
warnings.filterwarnings(action='ignore', category=FutureWarning)
```

```
[32]: datacsv1=pd.read_excel('MURO.xlsx') # lectura del archivo
datacsv2=pd.read_excel('MURO2.xlsx') # lectura del archivo
datacsv3=pd.read_excel('MURO3.xlsx') # lectura del archivo
datacsv4=pd.read_excel('MURO4.xlsx') # lectura del archivo
data1=np.array(datacsv1)
data2=np.array(datacsv2)
data3=np.array(datacsv3)
data4=np.array(datacsv4)
#print(data)
freq=data1[:,0] #asignacion de valores a los ejes
y=(data1[:,1]+data2[:,1]+data3[:,1]+data4[:,1])/4
y3=np.power(10,y/10) #convirtiendo a milivatios
y1=y3/np.max(y3) #normalizacion de la señal
plt.plot(freq,y1) #grafico de la señal adquirida
plt.xlabel('Frecuencia (Hz)')
plt.ylabel('Intensidad en milivatios (mW) normalizada')
```

```
plt.title('Párametros $S_{21}$')
plt.legend(['Datos obtenidos'])
plt.grid()
plt.savefig("muro260.jpg",dpi=500)
```

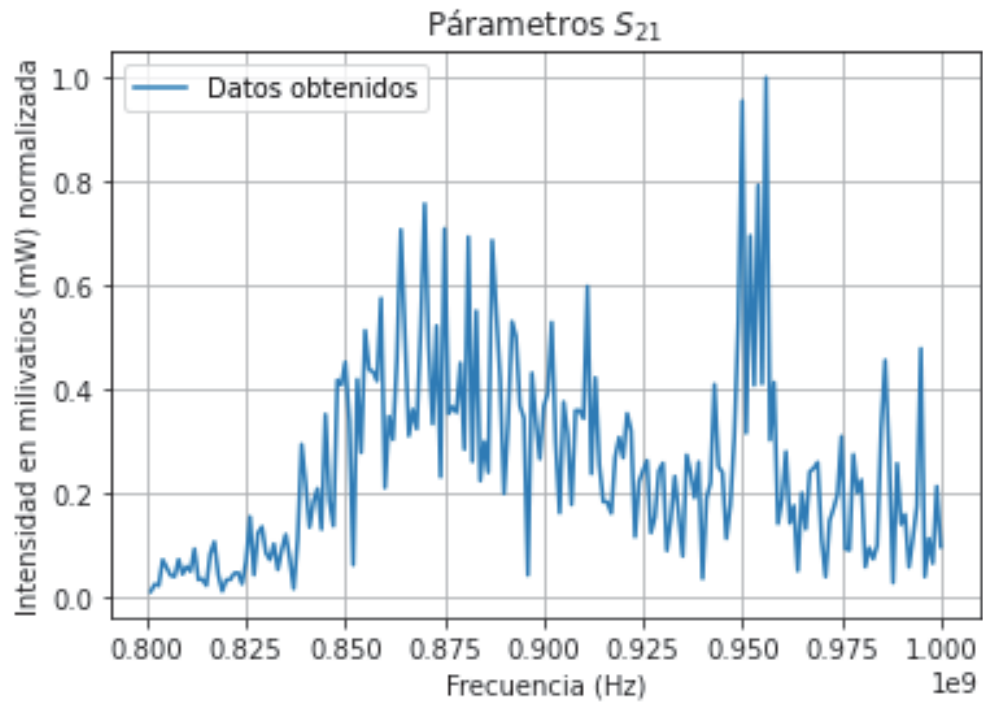
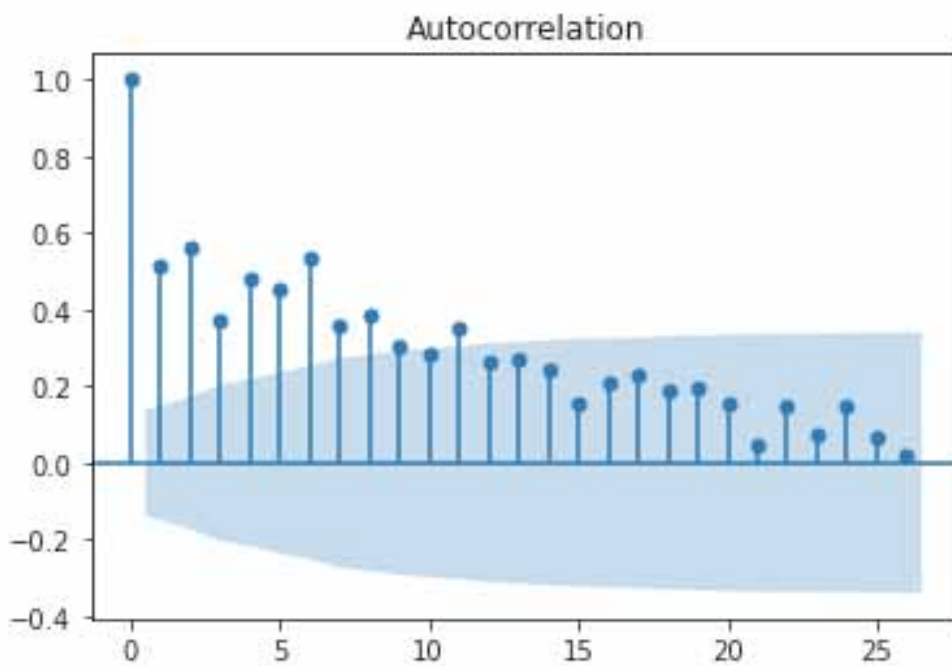
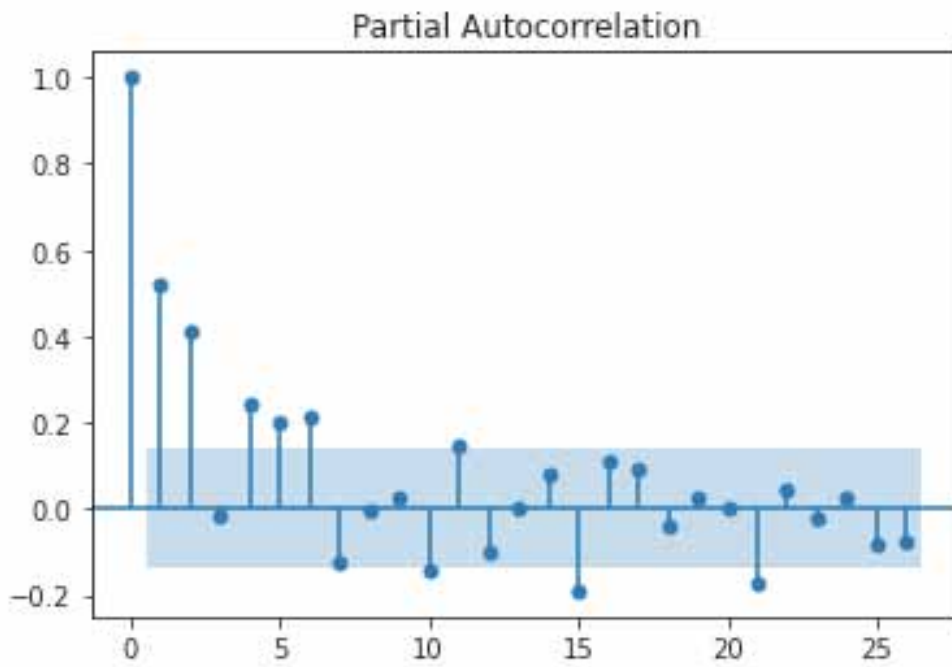


Grafico de la sample partial autocorrelation function (PACF)

```
[33]: plot_pacf(y1, lags=26);
plt.savefig("autocorrelacion_parcial260.jpg",dpi=500)
plot_acf(y1, lags=26);
plt.savefig("autocorrelacion260.jpg",dpi=500)
```



Criterio de Akaike para seleccionar el orden del proceso ARMA


```
[34]: res=arma_order_select_ic(y1, ic=["aic", "bic"], trend="nc")
print('el orden del modelo es',res.aic_min_order)
```

el orden del modelo es (4, 2)

Cálculo de los parametros del modelo ARMA con el orden anterior

```
[36]: model=ARMA(y1,order=(4,2))
model_fit=model.fit() #este comando calcula los parametros del modelo usando
↳ filtro de kalman
print(model_fit.summary())
y2=model_fit.predict()
```

ARMA Model Results

```
=====
Dep. Variable:          y      No. Observations:          200
Model:                 ARMA(4, 2)  Log Likelihood          116.577
Method:                css-mle    S.D. of innovations      0.135
Date:                  Sun, 06 Jun 2021  AIC                    -217.154
Time:                  22:04:29    BIC                      -190.768
Sample:                0          HQIC                     -206.476
=====
```

```
=====
              coef      std err          z      P>|z|      [0.025      0.975]
-----
const          0.2318      0.058      3.989      0.000      0.118      0.346
ar.L1.y        1.1497      0.158      7.271      0.000      0.840      1.460
ar.L2.y       -0.3219      0.197     -1.630      0.103     -0.709      0.065
ar.L3.y       -0.3233      0.102     -3.170      0.002     -0.523     -0.123
ar.L4.y        0.4158      0.081      5.104      0.000      0.256      0.575
ma.L1.y       -0.9161      0.166     -5.506      0.000     -1.242     -0.590
ma.L2.y        0.4280      0.165      2.600      0.009      0.105      0.751
=====
```

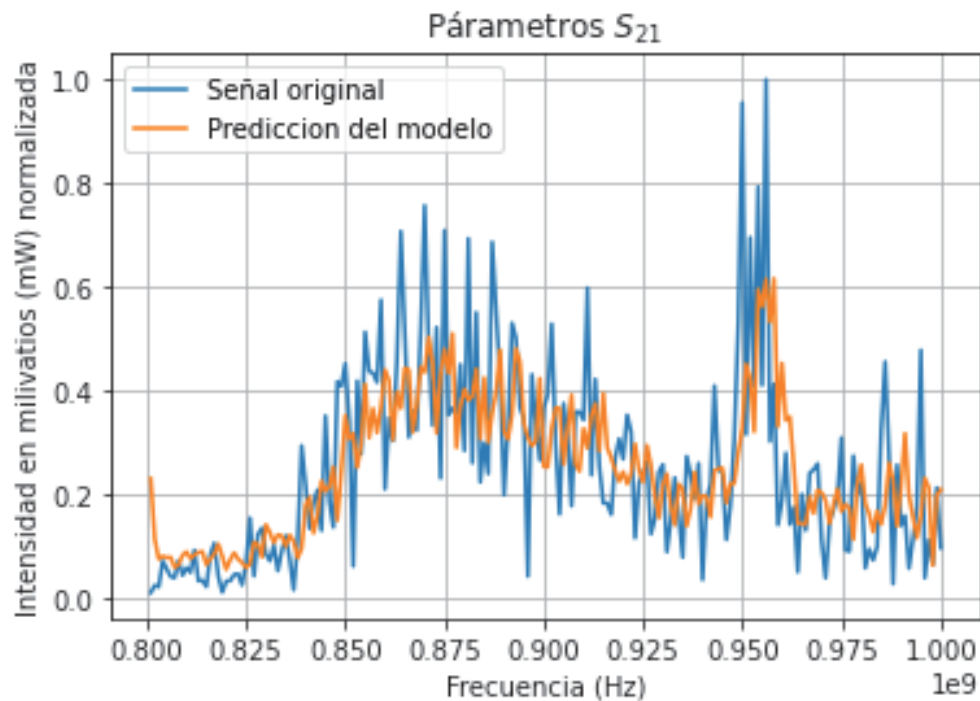
Roots

```
=====
              Real          Imaginary      Modulus      Frequency
-----
AR.1         -1.5423          -0.0000j          1.5423          -0.5000
AR.2          0.6288          -1.0356j          1.2116          -0.1632
AR.3          0.6288           +1.0356j          1.2116           0.1632
AR.4          1.0623          -0.0000j          1.0623          -0.0000
MA.1          1.0702          -1.0914j          1.5286          -0.1266
MA.2          1.0702           +1.0914j          1.5286           0.1266
=====
```

Grafico de comparacion de los datos obtenidos y la prediccion del modelo

```
[37]: plt.plot(freq,y1)
plt.plot(freq,y2)
plt.xlabel('Frecuencia (Hz)')
```

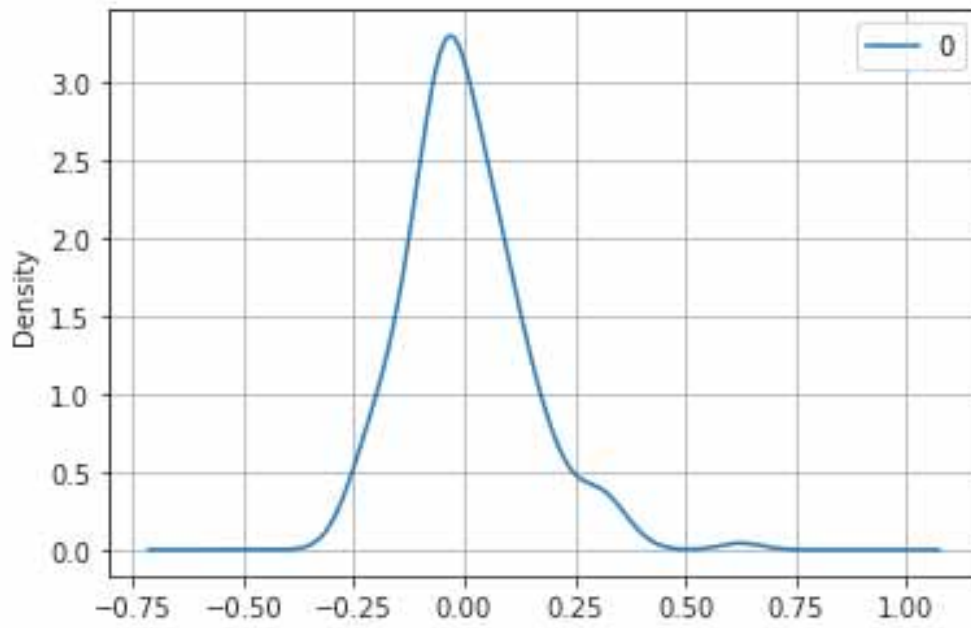
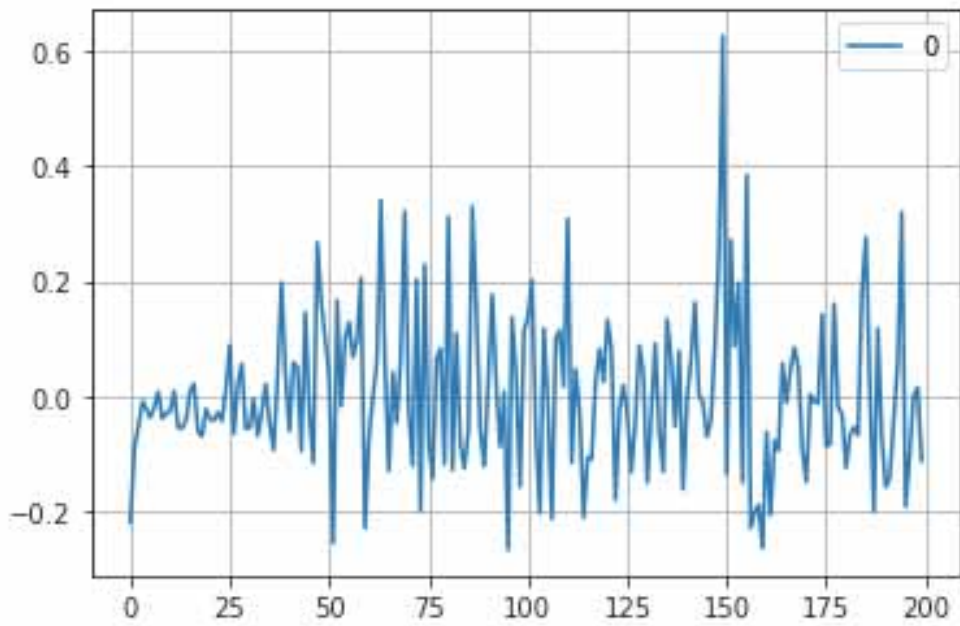
```
plt.ylabel('Intensidad en milivatios (mW) normalizada')
plt.grid()
plt.legend(['Señal original', 'Prediccion del modelo'])
plt.title('Párametros $S_{21}$')
plt.savefig("prediccion260.jpg",dpi=500)
```



Análisis de los residuos de la estimación del modelo

```
[38]: # plot residual erros
residuals = pd.DataFrame(model_fit.resid)
residuals.plot()
plt.grid()
plt.savefig("residuos260.jpg",dpi=500)
residuals.plot(kind='kde')
plt.grid()
plt.savefig("residuos2601.jpg",dpi=500)
plt.show()

print(residuals.describe())
```



```
0
count 200.000000
mean  0.003280
std   0.135413
min   -0.267328
```

25%	-0.077903
50%	-0.015542
75%	0.078871
max	0.626730

[]: