CENTRO POLITÉCNICO SUPERIOR
UNIVERSIDAD DE ZARAGOZA

Chaotic receivers for optical communication systems

Ingeniería de Telecomunicación
Proyecto de Fin de Carrera

Realizado en:
Technical University of Denmark

Javier Franco Biurrun
Julio 2011

Director:
Idelfonso Tafur Monroy
Dept. Metro-Access & Short Range Systems
Technical University of Denmark

Ponente:
Juan Ignacio Garcés Gregorio
Dpto. de Teoría de la Señal y Comunicaciones
C.P.S., Universidad de Zaragoza

Centro Politécnico Superior
Dpto. de Ingeniería Electrónica y Comunicaciones
C/ Maria de Luna 3, 50018 Zaragoza
Edificio Ada Byron
http://www.cps.unizar.es

Chaotic receivers for optical communication systems

Resumen

Este proyecto concierne al estudio de la detección de señales enmascaradas en entornos de ruido. Para los escenarios en los que la señal recibida es débil, se presenta una alternativa a los métodos de detección determinista más comúnmente empleados. Esta alternativa consiste en la implementación de un sistema caótico basado en un receptor Duffing de segundo orden. Dicho sistema de recepción caótico Duffing esta compuesto principalmente por un oscilador no lineal. Su comportamiento variará en función de la señal de entrada y transitará principalmente entre los estados de orden y caos. Esta característica nos servirá como punto de partida para la detección de la señal.

En esta tesis se ha implementado y evaluado en primer lugar, un modelo de oscilador Duffing para la detección de señales binarias. Además, el autor ha propuesto y testado una modificación del diseño convencional del sistema de recepción caótico Duffing con el propósito de mejorar el rendimiento en términos de Bit Error Rate (BER) frente a la Relación Señal a Ruido (SNR).

Posteriormente y a través de simulaciones, se han evaluado diversos métodos para la detección de símbolos. Con ello se ha pretendido estudiar el rendimiento de cada método analizando la sensibilidad del BER con respecto al SNR recibido.

Más tarde, utilizando el método de detección de símbolos óptimo, se han establecido comparaciones en términos del BER entre los métodos de demodulación estándar y los métodos propuestos. Diversos escenarios han sido analizados para diferentes anchos de banda del Filtro Paso Banda (BPF) en recepción.

Finalmente, se ha llevado a cabo una demostración experimental para escenarios de Radio sobre Fibra (RoF). El formato de modulación considerado ha sido Amplitude-Shift Keying (ASK) para tasas de datos superiores a $1Gbps$, con una frecuencia de portadora de $13GHz$ y todo ello integrado en un montaje de laboratorio sobre fibra óptica. Se ha podido observar que los resultados experimentales y las simulaciones han resultado en consonancia.

Tabla de contenidos

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

Este proyecto se ha desarrollado en el marco de una Master Thesis de 35 ECTS en el departamento 'Photonics Engineering Metro-Access and Short Range Systems' de la Technical University of Denmark (DTU), Lyngby (Copenhagen). Dicho proyecto, previamente a su depósito en el Centro Politécnico Superior (C.P.S) de Zaragoza, ha sido entregado, presentado y evaluado en la DTU obteniendo la nota de 12, máxima calificación en la escala académica danesa y equivalente a un excelente o 'A' en el estándar ECTS.

El término telecomunicaciones hace referencia un amplio grupo de tecnologías para la transmisión de información sobre cierta distancia. En todas sus formas, el propósito fundamental que se persigue es la transmisión de la mayor cantidad de datos posible sin ningún error. En todo sistema existen siempre una serie de limitaciones para alcanzar una comunicación robusta que vienen descritas por el teorema de Shannon-Hartley. En él se establece una relación entre las potencias de señal y ruido, la capacidad del canal y el ancho del mismo [2].

Teóricamente, una cantidad ilimitada de datos podría ser transmitida libre de errores sobre un ancho de banda infinito y para un canal de comunicaciones libre de ruido. Sin embargo, una comunicación real se ve afectada por estas restricciones de ancho de banda y limitaciones debidas al ruido. En esta tesis considera el caso de detección de una señal de comunicación débil en entornos con una fuerte presencia de ruido.

La detección de señales débiles es un concepto ampliamente utilizado en ámbitos como la navegación por sonido (sonar), las comunicaciones radar, prevención de terremotos, así como el estudio de las radiaciones emitidas y otra serie de aplicaciones de medición industrial [3]. Recientemente la detección de señales débiles ha adquirido cierta relevancia en temas de comunicaciones seguras [4]. Este proyecto esta enfocado para el caso de

las comunicaciones ópticas y concretamente para RoF.

Se considera un escenario híbrido, en el que la comunicaciones wireless está integrada en una red óptica, como puede verse en la Figura 1.1. La señal wireless es generada por medios ópticos y transmitida sobre un enlace wireless de Radiofrecuencia (RF). Después de la transmisión wireless, la señal recibida junto con el ruido son detectados y transportados sobre una fibra óptica a la oficina central para su posterior procesado. Se ha considerado la transmisión de señales en presencia de ruido, debido al canal de transmisión y a la transformación de la onda en una señal digital.

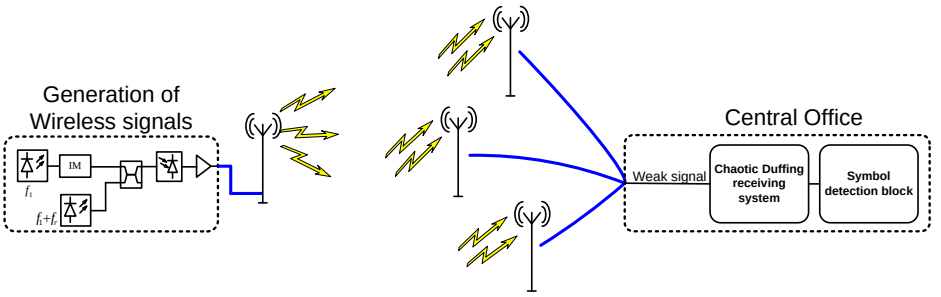


Figura 1.1: Visión general del escenario híbrido, fibra óptica - comunicación wireless. El modelo implementa la generación de una señal wireless por métodos ópticos heterodinos, transmisión wireless y posteriormente, transporte sobre una fibra óptica para el procesado en la oficina central.

1.1 Estado del Arte de los sistemas de recepción

La mayoría de los métodos usados para la recuperación de señales en presencia de fuerte ruido alcanzando una correcta demodulación, están basado en métodos probabilísticos. Sin embargo, cuando la señal recibida es débil se requiere el desarrollo de un modelo de distribución probabilístico. Esto conlleva dificultades para aplicaciones prácticas como el alto coste computacional [5].

Otro método empleado en la recuperación de señales basado en la Densidad Espectral de Potencia (PSD) donde se analizan las componentes frecuenciales de la señal. Este método resulta útil para identificar las periodicidades de la señal aunque su rendimiento esta limitado cuando la señal se encuentra inmersa en entornos ruidosos [6]. Aún así, puede ser utilizado en combinación con otras técnicas de demodulación para la detección de señales. Profundizaremos en dicho planteamiento en la sección 5.1.1.

Existen otras alternativas relacionadas con las distribuciones tiempo-frecuencia como es el caso de la Transformada Rápida de Fourier (FFT). La limitación de este método viene impuesta debido a su baja resolución frecuencial, lo que puede conducir a decisiones imprecisas. A lo que hay que añadir que estos planteamientos no resultan los más apropiados para el análisis de señales no estacionarias ya que no son capaces de revelar la información inherente almacenada en ellas [7]. Otra alternativa sería la Transformada de Fourier de Tiempo Reducido (STFT). Como la señal a detectar variará a lo largo del tiempo, se utiliza la STFT para determinar la frecuencia sinusoidal y la fase contenida en secciones locales de dicha señal. La STFT junto con la Wavelet Transform (WT) son unas transformadas lineales de representación tiempo-frecuencia. Como transformadas lineales trabajan directamente en el dominio del tiempo y no resultarán ventajosas para la reducción de fuertes ruidos [8].

Las dificultades encontradas para aplicaciones prácticas se deben principalmente a las limitaciones de la señal, a la baja resolución frecuencial y a las decisiones imprecisas. Los métodos probabilísticos, los métodos basados en PSD y los métodos relacionados con las distribuciones tiempo-frecuencia también fallan en la detección de señales ruidosas, debido a dichas dificultades. De ahí la necesidad de implementar un nuevo esquema para la detección de señales basado en los sistemas dinámicos no lineales. La propiedad fundamental de este tipo de sistemas se debe a su elevada Sensibilidad Dependiente de las Condiciones Iniciales (SDIC). De la referencia [9], se muestra como se puede interpretar esta propiedad como una sensibilidad dependiente de los parámetros y por consiguiente, una pequeña perturbación de los parámetros de entrada conllevaría un cambio significativo en el estado de todo el sistema. Este complejo comportamiento de los sistemas deterministas no lineales viene descrito por la teoría del caos [10].

1.2 Objetivo de esta tesis

El propósito de esta tesis es analizar el enlace de comunicaciones, centrándose en la estructura del receptor para señales débiles enmascaradas en un fuerte ruido. En la Figura 1.1 se puede ver el escenario de comunicaciones. Se observa que el receptor esta directamente conectado con la oficina central. En este caso, en el receptor implementa un detector caótico a través de la ecuación Duffing. Se ha elegido esta ecuación entre los sistemas clásicos no lineales debido a la gran cantidad de documentación previa existente [11].

La naturaleza no lineal de la comunicación objeto de estudio afectará a la estructura del receptor, por ello un sistema que resolviese la ecuación Duffing con métodos lineales nunca sería una solución óptima. Consecuentemente, el modelo de simulación computacional implementado se basa en un análisis numérico.

Se conocen técnicas contrastadas de detección que trabajan con niveles negativos de SNR con una elevada precisión, [12] y [13]. Este bajo SNR ha sido conseguido como resultado del uso de una elevada frecuencia de muestreo. Este trabajo se centrará en la Eb/No, que esta relacionada con la SNR como $Eb/No = SNR \cdot \frac{F_s}{2F_b}$, donde F_b es la tasa de bits y F_s es la frecuencia de muestreo de la señal. La principal diferencia entre Eb/No y SNR es que Eb/No tiene en cuenta el ancho de banda del ruido (F_s) y el ancho de banda de la señal (F_b).

En esta tesis se pretende evaluar teórica y experimentalmente el rendimiento del receptor Duffing. Además, se requiere analizar la viabilidad del sistema de recepción caótica Duffing utilizado como alternativa a la detección de envolvente u otros métodos de demodulación. Las simulaciones computacionales y los resultados experimentales son presentados para escenarios de RoF.

Una vez se haya implementado el sistema de recepción caótico Duffing, se caracterizará dicho sistema para cada una de las técnicas de modulación ASK, Phase-Shift Keying (PSK) y Frequency-Shift Keying (FSK). Posteriormente, se llevará a cabo un estudio profundizando en la modulación ASK y asemejando el escenario a una comunicación real. El sistema de

recepción caótico Duffing y el método de detección coherente serán analizados y se evaluará su rendimiento para diferentes anchos de banda entre $1GHz$ y $8GHz$ del filtro de entrada, siendo la tasa de la señal definida a $1GHz$ para todos los escenarios.

El objetivo final es la comparación del rendimiento en términos de BER requerido para diferentes Eb/No entre la detección de envolvente y una modificación del diseño convencional del Duffing. La ventaja de este diseño es su mayor simplicidad ya que no se necesita ninguna señal de referencia. Las comparaciones son llevadas en las mismas condiciones para ambos sistemas, y para diferentes anchos de banda del BPF. Este BPF será implementado como el primer bloque en el esquema de recepción. Finalmente, se pretenden validar los resultados de simulación analizando los correspondientes datos del montaje óptico de laboratorio. La demostración experimental esta implementada para escenarios donde la velocidad de modulación es de $13GHz$ de frecuencia de portadora y $1Gpbs$ de tasa de datos.

1.3 Estructura de la tesis

La estructura de este trabajo se ilustra en la Figura 1.2. Esta compuesto por tres bloques principales; antecedentes y planteamiento del problema a resolver, desarrollo de la solución y conclusiones.

- **Antecedentes y Problema** expone una introducción básica a la detección de señales débiles enmascaradas en entornos ruidos, se presenta el estado del arte y se analiza la aportación de este trabajo en los sistemas de recepción.
- **Implementación y Resultados** engloba cuatro capítulos. En el primero se muestran los pasos a seguir para alcanzar una implementación precisa del modelo de Matlab. En el segundo capítulo se estudian las tres técnicas de modulación y como se ha de caracterizar al oscilador Duffing para demodular cada una de ellas. En el tercer capítulo se estudia en profundidad la modulación ASK. En el último de estos capítulos, se realizan diversos montajes de

laboratorio con los escenarios que han resultado más interesantes en los análisis previos.

- **Conclusiones** de esta tesis agrupadas y resumidas. También se propone una linea futura de estudio.

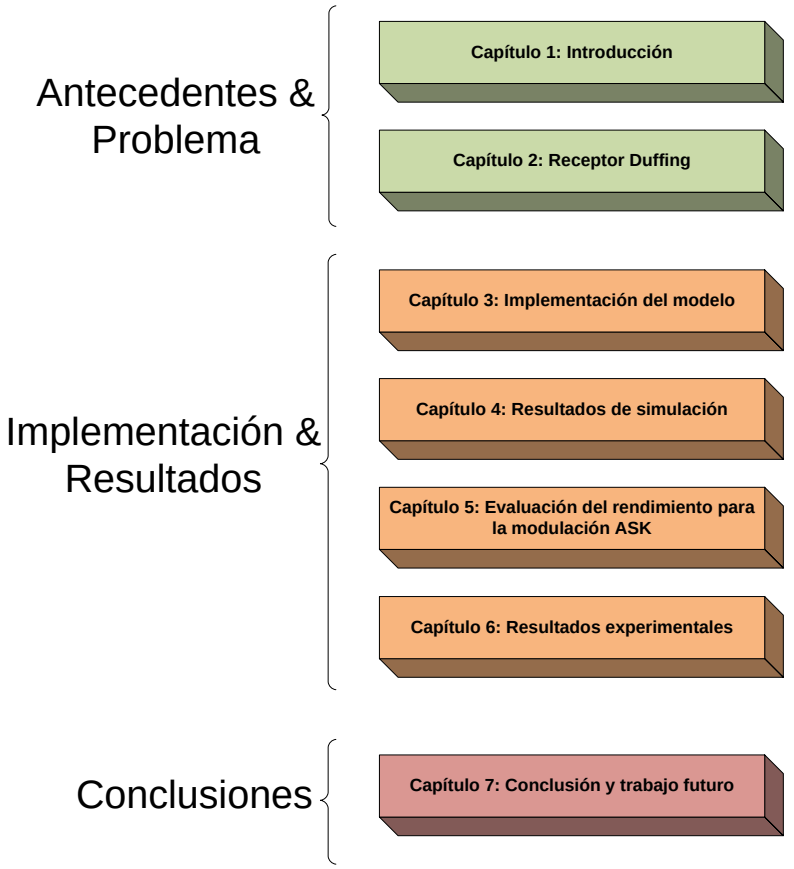


Figura 1.2: Estructura de la tesis.

Capítulo 2

Receptor Duffing

El propósito de este capítulo es presentar de una manera simple una visión general sobre el oscilador Duffing. Para un análisis más profundo se puede consultar el Apéndice A, donde se detalla paso a paso el desarrollo matemático seguido para alcanzar las soluciones aquí presentadas.

2.1 Principios fundamentales

Para la implementación de un receptor caótico se hace uso de una Ecuación Diferencial Ordinaria (ODE) de segundo orden. Concretamente, se implementa la ecuación Duffing. Su formula viene expresada a continuación:

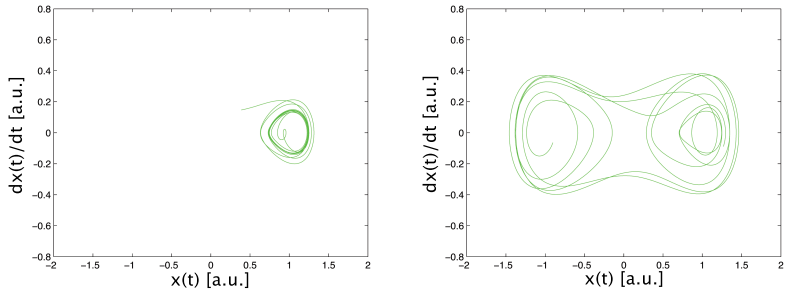
$$\ddot{x} + \delta\dot{x} - x + x^3 = f_r \cdot \cos(t + \theta) \tag{2.1}$$

en donde destaca el parámetro $f_r \cdot \cos(t + \theta)$ que representa la señal de referencia del oscilador. Está centrada a la frecuencia de portadora y su fase es desconocida.

Para simplificar futuras implementaciones, se convierte la ecuación (2.1) en un sistema equivalente de ecuaciones de primer orden:

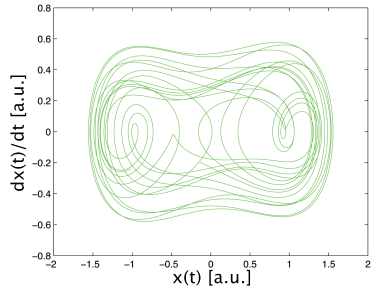
$$\begin{cases} \dot{x} = y \\ \dot{y} = x - x^3 + \varepsilon(f_r \cdot \cos(t + \theta) - \delta y) \end{cases} \tag{2.2}$$

En esta ecuación destacan dos parámetros que resultarán determinantes para la caracterización del sistema. El ratio de amortiguamiento (*damping ratio*) δ y la amplitud de la señal de referencia f_r. De los distintos valores que adquiera su cociente δ/f_r saldrán los cuatro estados del oscilador Duffing. También resulta de especial interés determinar los valores frontera entre unos estados y otros. Para un análisis más detallado,

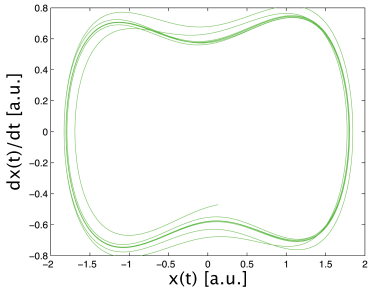


(a) Orbita simple, $f_r/\delta < R^0(\omega)$

(b) Bifurcación, $f_r/\delta > R^0(\omega)$



(c) Caos, $f_r/\delta > R^0(\omega)$ & $f_r < F_d$



(d) Movimiento periódico, $f_r > F_d$

Figura 2.1: Diferentes diagramas de fase para el sistema de recepción caótico Duffing representado sus cuatro estados. El parámetro δ está definido a 0.5, el umbral de bifurcación $R^0(\omega)$ a 0.3765 y $F_d = 0.753$.

dirigirse al apartado A.2. En la figura 2.1, podemos observar los cuatro estados por los que transmita el sistema. Los dos más interesantes son:

- El estado **caótico**, figura 2.1(c), donde la trayectoria de las órbitas tiende a rellenar la zona interior del diagrama de fase de forma aleatoria.
- El estado de **movimiento periódico**, figura 2.1(d), donde las órbitas siguen una trayectoria periódica trazando una curva cerrada.

Esta característica de los sistemas no lineales junto con la alta sensibilidad

a una variación en las condiciones iniciales, convierten a estos sistemas en apropiados para la detección de señales débiles. Para ello, tendremos que asociar cada uno de los dos estados (caótico y periódico) con cada uno de los valores lógicos que se pueden transmitir, unos y ceros lógicos.

2.2 Modelo para la detección de señales

Una vez se ha caracterizado el sistema Duffing, se añade la señal ruidosa que ha de ser detectada $s(t) = a\cos((1 + \Delta\omega)t + \varphi) + zs$. La nueva ecuación queda descrita como:

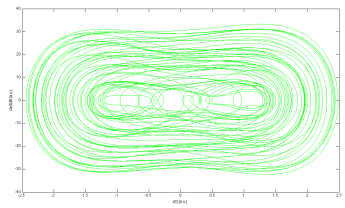
$$\begin{cases} \dot{x} = y \\ \dot{y} = x - x^3 - \delta y + f_r \cos t + s(t) \end{cases} \tag{2.3}$$

zs representa el ruido blanco Gaussiano con media cero y varianza z. $\Delta\omega$ es la diferencia de la frecuencia angular entre la señal de referencia y la señal a ser detectada.

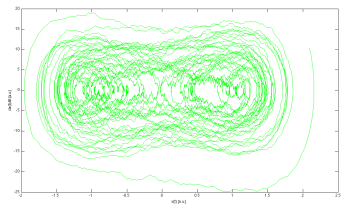
El ruido afectará únicamente a la trayectoria de la señal. Los trazos de las órbitas dejarán de ser firmes. Pero en ningún momento, el ruido causará una transición entre los estados del sistema. La diferencia en el diagrama de fase entre una señal ruidosa y otra libre de ruido puede verse en la figura 2.2.

Para simplificar la transición entre estados, se define el parámetro $\delta = 0.5$. Entonces, en ausencia de ruido, la transición solo dependería del parámetro f_r. Ahora se incorpora la señal a detectar $s(t)$, por lo que también se ha de tener en cuenta su amplitud en la transición de estados. Al estar trabajando con señales débiles, la amplitud a va a ser un valor mucho más pequeño que f_r. De ahí se utilice un oscilador no lineal, ya que como se ha mencionado, una pequeña perturbación en sus condiciones iniciales provocará un cambio en el estado de todo el sistema.

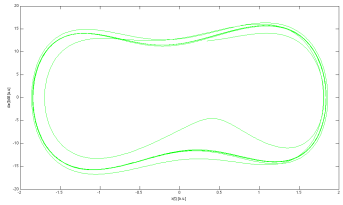
Lo que se pretende es asociar la recepción de cada uno de los bits lógicos con un estado distinto. Se va a trabajar con transmisiones binarias y ya se conocen los dos estados más importantes de nuestro sistema. Este planteamiento esta ilustrado en la figura 2.2. En la primera imagen,



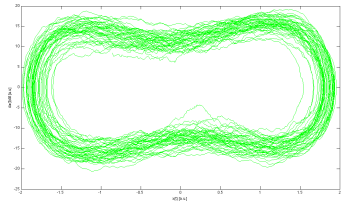
(a) Estado caótico sin ruido



(b) Estado caótico, señal ruidosa



(c) Estado periódico sin ruido



(d) Estado periódico, señal ruidosa

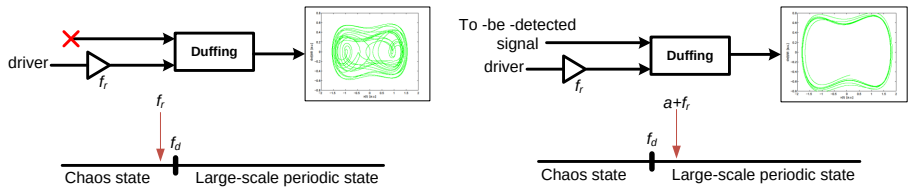
Figura 2.2: Diagrama de fase de los distintos estados para situaciones con fuerte ruido o ausencia del mismo.

figura 2.3(a), se presenta la transmisión de un '0' lógico. El objetivo es definir el parámetro f_r lo más próximo al umbral de transición entre estados pero manteniendo el sistema en el estado caótico. Cuando el bit recibido sea un '1' lógico, como se presenta en la figura 2.3(b), la amplitud de f_r se verá incrementada debido a la aportación de la señal recibida y se excederá el umbral F_d; pasando el sistema al estado periódico.

2.2.1 Sincronización de fase

La fase de la señal a detectar es el segundo factor a tener en cuenta para hacer posible la transición entre los estados del sistema. Esta fase ha de estar definida dentro del rango indicado en la ecuación (2.4). En caso de no cumplir esta condición, el sistema permanecería en el estado caótico todo el tiempo.

$$\pi - \cos^{-1}\frac{a}{2f_r} \leq \varphi \leq \pi + \cos^{-1}\frac{a}{2f_r} \tag{2.4}$$



(a) 0 lógico transmitido (b) 1 lógico transmitido

Figura 2.3: Diagrama de fase en función de las entradas del oscilador Duffing.

Para facilitar la interpretación de la ecuación (2.4) se aplican una serie de aproximaciones y el resultado se ilustra en la figura 2.4. En ella están claramente indicados los rangos en los que ha de estar definida la fase de la señal.

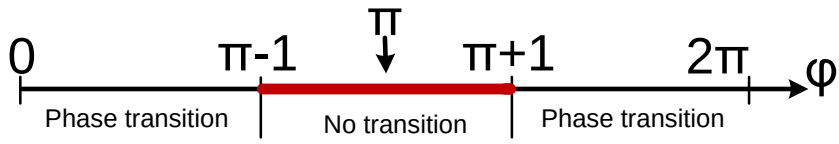


Figura 2.4: Comportamiento del oscilador Duffing en función del parámetro φ.

2.2.2 Extensión del modelo a cualquier frecuencia de portadora

Hasta ahora el sistema Duffing solo estaba caracterizado para la detección de señales sinusoidales a baja frecuencia. Con objeto de hacer extensible dicho sistema para frecuencias más altas, modificamos la ecuación (2.2).

El resultado se muestra a continuación:

$$\begin{cases} \dot{x}_1 = \omega x_2 \\ \dot{x}_2 = \omega(x_1 - x_1^3 - bx_2 + c\cos\omega t) \end{cases} \tag{2.5}$$

Esta nueva ecuación hará el sistema adecuado para futuras simulaciones en escenarios de RoF.

Capítulo 3

Implementación del modelo

Una vez se ha explicado teóricamente el oscilador Duffing en el capítulo anterior, se pretende construir el modelo de simulación. En una primera aproximación, se implementa dicho modelo a través de Simulink. Más tarde y a fin de tener un control total del sistema, se define un modelo similar mediante Matlab. Se entrará en detalle en el bloque de integración para resolver la ecuación Duffing, así como en las soluciones propuestas.

Este capítulo se explica más en profundidad en el Apéndice B. Se presenta un análisis exhaustivo de los métodos y procedimientos utilizados, así como una serie de gráficas y tablas comparativas justificando todas las decisiones adoptadas.

3.1 Modelo mediante Simulink

El uso de Simulink como herramienta de simulación no es algo casual. Es el método más extendido para la implementación del oscilador Duffing como se puede apreciar en la gran mayoría de las publicaciones consultadas, como por ejemplo [14] y [15].

El resultado de dicha implementación puede verse en la figura 3.1. Este sistema trabaja muestra a muestra a partir de la señal de entrada. En el primer bloque (*ADD1*), la señal de referencia (*DRIVER*) junto con la señal a detectar (*TO-BE-DETECTED SIGNAL*) son sumadas a las muestras de salida del ciclo anterior del sistema Duffing. Todo ello es posteriormente amplificado y previa integración, se le resta a la señal de salida de primer orden del sistema. Tras esto, la señal es integrada dos veces (bloques *INT*) a fin de resolver la ODE de segundo orden de la ecuación. Las salidas *OUTPUT_X* y *OUTPUT_Y*, serán utilizadas para

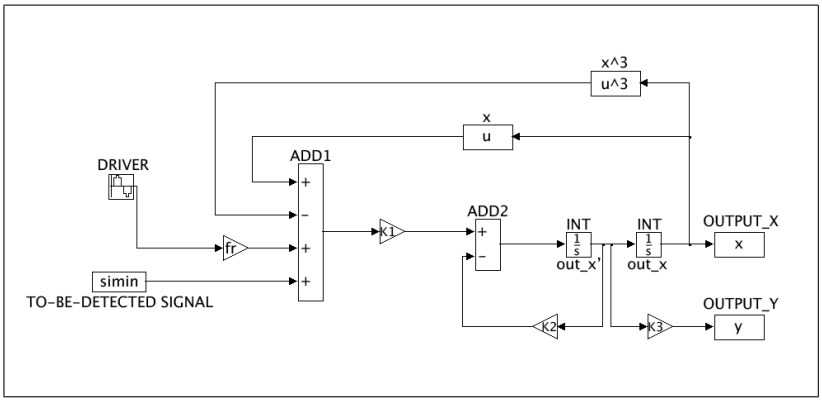


Figura 3.1: Implementación del oscilador Duffing en Simulink.

representar los diagramas de fase. Gracias a estos diagramas es posible conocer el estado en el que se encuentra el sistema en todo momento.

Después de una serie de simulaciones y a pesar de que la mayoría de nuestras referencias han obtenido sus resultados a través de la herramienta Simulink, se observa que este método presenta una serie de desventajas. Al tener sus funciones previamente predefinidas, perdemos gran parte del control sobre la información con la que se está tratando. Además, se necesita una interacción continua con Matlab para el procesado de la señal. Es por ello que decidimos construir el modelo haciendo uso únicamente de Matlab.

3.2 Modelo mediante Matlab

El procedimiento para la implementación del oscilador Duffing en Matlab consiste en descomponer el modelo de Simulink presentado en la figura 3.1, analizando las entradas y salidas de cada bloque. De esta forma es posible calcular los retardos y otras relaciones entre bloques. Una vez se hayan caracterizado todos los bloques, se conectarán unos con otros.

A lo largo de este proceso aparece un bloque crítico, el bloque de integración. En él se resuelve la ODE de primer orden. El objetivo es definir el algoritmo más sencillo posible sin perder precisión. También se tendrán en cuenta otros factores como el rendimiento y el coste computacional; ya que para cada iteración la operación se ejecuta varias veces.

Para ello han sido testados tres métodos diferentes. El primero utiliza el método integral de Euler. Además se analizan tres posibles variantes de este método (Apéndice B.2.1). El segundo método implementado es la regla trapezoidal (Apéndice B.2.3). La tercera alternativa es el método de Runge-Kutta de cuarto orden (Apéndice B.2.3).

Una vez se han implementado los tres métodos, definimos unas serie de criterios para elegir el método más adecuado en cada situación. Estos factores son: rendimiento, estabilidad y precisión. Los resultados se indican de forma simplificada en la tabla 3.1.

Metodo	Estabilidad	Precisión	Rendimiento
Euler Forward	baja	baja	alto
Trapezoidal	media	media	medio
Runge-Kutta	alta	alta	bajo

Tabla 3.1: Comparativa entre el método de Euler Forward, la regla trapezoidal de integración y la implementación del método Runge-Kutta para los términos de estabilidad, precisión y rendimiento.

Como conclusión del análisis de estos tres métodos podemos afirmar que el método de Euler (*Forward Euler*) es el que menor coste computacional requiere y a su vez, es el más simple de implementar. Sin embargo, es el método menos preciso y estable de los tres. La regla trapezoidal es bastante más precisa y estable que el método de Euler. Este método lo utilizaremos para escenarios donde existe una gran cantidad de datos a procesar y el tiempo de procesado es un factor a tener en cuenta. El método de Runge-Kutta tiene como desventaja un mayor requerimiento en el tiempo de procesado que otros métodos multi-incremento de similares prestaciones. Sin embargo, entre los sistemas analizados en este proyecto, la relativa simplicidad y facilidad de uso compensa su elevado coste computacional. Por lo tanto, este tercer método será utilizado en

las principales simulaciones, así como en las caracterizaciones de todo el sistema.

Capítulo 4

Resultados de simulación

La mayor parte de la documentación sobre osciladores Duffing presenta este sistema como una potente herramienta para la detección de señales débiles. Sin embargo, es muy difícil encontrar referencias específicas en la literatura en las que se trabaje con un formato concreto de modulación. Por este motivo se explica en este proyecto como debería estar definido y caracterizado el oscilador Duffing para poder utilizarlo con los tres formatos básicos de modulación.

Este capítulo se encuentra detallado en el Apéndice C de este proyecto.

4.1 Detección de señales 2ASK mediante el oscilador Duffing

Se ha utilizado OOK, la forma más simple de la modulación ASK. Tiene la particularidad de que la portadora es nula para la transmisión de un cero lógico, como se puede observar en la figura 4.1.

Como anteriormente se presento en el capítulo 2, tratamos de asociar el 0 lógico con el estado caótico del sistema Duffing. Para ello nos aseguramos de que el parámetro f_r sea ligeramente inferior al valor del umbral entre ambos estados. Esta explicación está ilustrada en la figura 2.3(a). Para el caso de un 1 lógico a la entrada del oscilador, la amplitud resultante será la suma de las amplitudes de la señal de referencia con la señal a ser detectada. Esta amplitud total hará que el oscilador Duffing se desplace desde el estado caótico hasta el estado periódico, se aprecia en la figura 2.3(b). Esto sucederá siempre y cuando la fase de la señal este definida fuera del rango indicado en la ecuación (2.4).

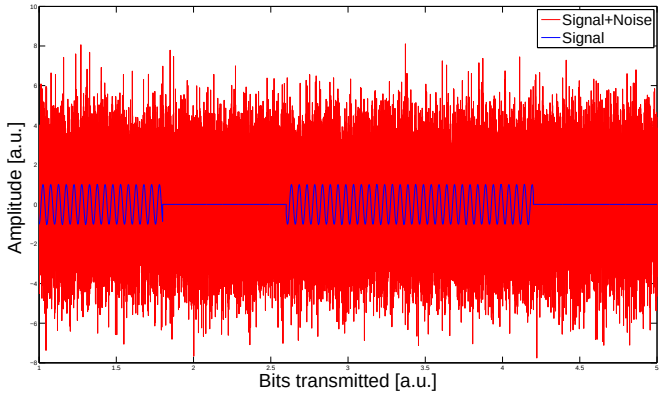


Figura 4.1: Señal OOK modulada. En color azul aparece la señal transmitida y por tanto, libre de ruido. En color rojo observamos la señal recibida tras el paso por el canal ruidoso. Los parámetros de la simulación son $f_c = \frac{50.000}{\pi}$ como frecuencia de portadora, amplitud $a = 0.2$, potencia de ruido 40 y tasa de datos $F_B = 100Hz$.

La salida del sistema para la detección de la señal OOK se muestra en la figura 4.2. En la primera imagen se presenta el diagrama de fase. Se aprecia como el sistema ha transitado por los estados caótico y periódico durante toda la transmisión. En la segunda imagen, se distingue en color azul la salida de nuestro sistema Duffing. Sobre esta señal, se presentan en color rojo la secuencia de 1's y 0's transmitidos. Así se puede ver que para la transmisión de un 1 lógico, el oscilador se encuentra en el estado periódico y por lo tanto su salida respecto al tiempo es una señal periódica. Para el caso de un 0 transmitido, la señal resultante a la salida del sistema carece de cualquier tipo de periodicidad. Resulta sencillo identificar a simple vista cuando se ha transmitido un 1 o un 0.

El código de Matlab para la detección una señal modulada OOK haciendo uso de un oscilador Duffing se muestra en el Apéndice G.1.

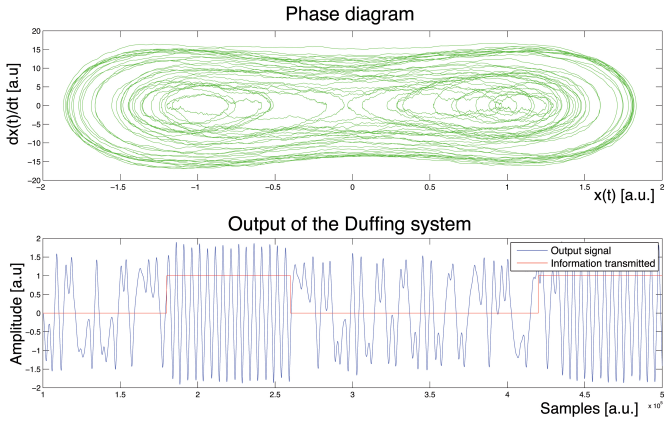


Figura 4.2: Diagrama de fase y salida del Duffing para el caso de detección OOK. En la primera imagen se presenta el diagrama de fase de toda la comunicación. Las órbitas exteriores son algo más gruesas, ya que han sido recorridas varias veces estando el sistema en el estado periódico. Las otras órbitas son aleatorias como consecuencia del estado caótico. En la segunda imagen, vemos en azul la salida del oscilador Duffing y sobre esa imagen en rojo, una señal que representa la secuencia binaria transmitida.

4.2 Detección de señales BPSK mediante el oscilador Duffing

Ahora se va a analizar la modulación PSK. Si se utiliza la configuración anterior para este escenario, el oscilador va a permanecer en el mismo estado durante toda la comunicación. Esto se debe a que la amplitud de la señal de entrada es constante tanto para los 1's lógicos como para los 0's. Para poder asociar los estados caótico y periódico con estos 1's y 0's lógicos, se habrá de redefinir la fase de nuestra señal.

Se fija la amplitud de la señal de entrada por encima del valor umbral F_d, así nos aseguraremos de que el sistema este en el estado periódico. Al transmitir un 1 lógico, la señal tendrá la fase φ_1. Esta fase ha de estar definida en el rango de fases donde se produce transición, por lo

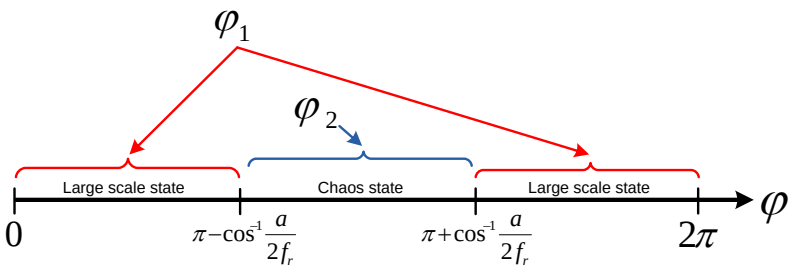


Figura 4.3: Definición de fase para BPSK. Para el escenario en el que los requerimientos de amplitud se cumplen y una señal con fase φ_1 es recibida en el oscilador, el sistema estará en el estado periódico. Por le contrario, cuando es una señal con fase φ_2 la que se introduce en el oscilador Duffing, el sistema estará en el estado caótico.

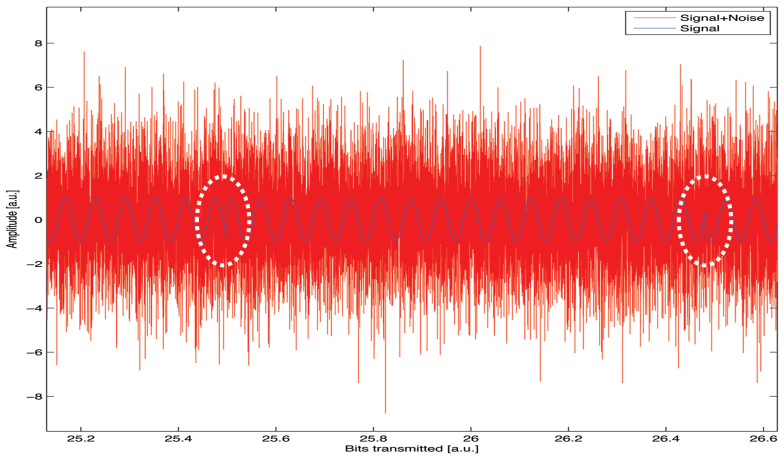


Figura 4.4: Señal BPSK modulada. En color azul se representa la señal transmitida con sus transiciones de fase. En color rojo, la señal recibida en el oscilador Duffing. Los parámetros de la simulación son la frecuencia de portadora $f_c = \frac{50.000}{\pi}$, amplitud $a = 0.2$, potencia de ruido 40 y tasa de bit $F_B = 100Hz$.

tanto el sistema permanecerá en el estado periódico. Cumpliendo estas

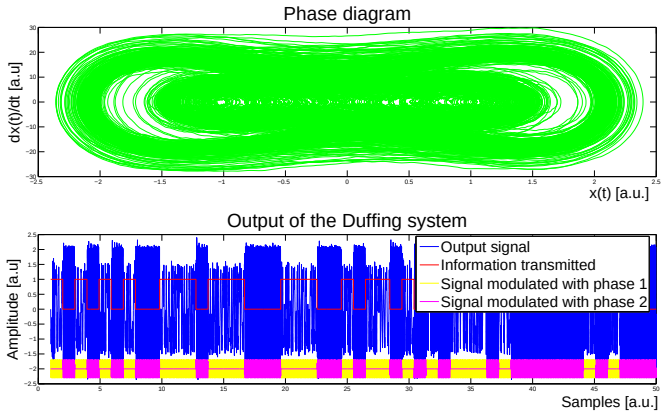


Figura 4.5: Diagrama de fase y salida del oscilador Duffing para la modulación BPSK. En la primera imagen se aprecia el diagrama de fase. En la segunda imagen podemos ver la salida del oscilador Duffing en color azul y sobre esta, una señal que simula la información transmitida. Debajo de ambas señales, la señal transmitida se presenta en dos colores diferentes. Así puede distinguirse entre las dos fases de la señal.

dos condiciones se ha asociado el estado periódico al 1 lógico. La fase φ_2 estará definida en el rango en el que no se produce transición. De esta forma, las dos condiciones para alcanzar el estado periódico no se cumplen y el sistema volverá al estado caótico para el caso del 0 lógico. Ambas fases, φ_1 y φ_2, se encuentran definidas en la figura 4.3.

En la figura 4.4, se muestra la modulación BPSK antes y después de atravesar el canal ruidoso, colores azul y rojo respectivamente. Los cambios de fase se acentúan con un circulo blanco a su alrededor.

En la figura 4.5, se presenta el diagrama de fase de toda la comunicación. Bajo esto, se ilustra la señal modulada, representada en amarillo y rosa para diferenciar cuando se ha transmitido un 0 o un 1. En color azul esta representada la salida del oscilador Duffing y sobre dicha señal, en color rojo, la secuencia de 1's y 0's transmitidos, antes de ser modulada.

El código de Matlab para la detección una señal modulada BPSK usando

un oscilador Duffing se encuentra en el Apéndice G.2.

4.3 Detección de señales BFSK mediante el oscilador Duffing

El tercer escenario se centra en el análisis para FSK. Hasta ahora la frecuencia de la señal de referencia ha sido la misma que la de la señal a demodular. Pero no sucede lo mismo en este caso, ya que tenemos dos frecuencias diferentes. Se define la señal de referencia a la frecuencia de portadora f_1. Frecuencia con la que transmitiremos un 1 lógico. También hay que asegurarse de definir f_r ligeramente menor que el valor umbral F_d.

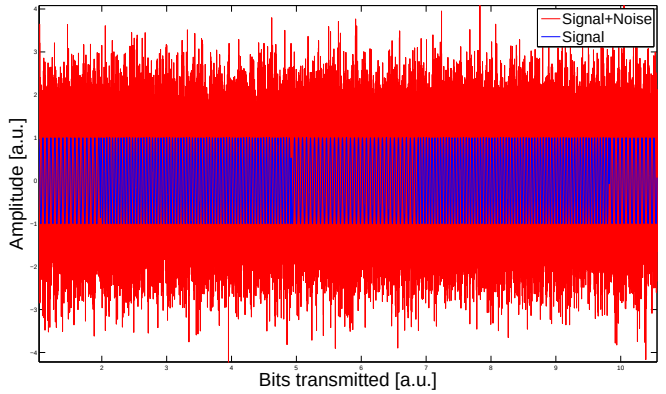


Figura 4.6: Señal BFSK modulada. En azul aparece la señal transmitida, se observa como esta compuesta por dos frecuencias distintas. En color rojo, la señal recibida tras pasar por el canal. Las frecuencias de portadora son $f_1 = 59.000Hz$ y $f_2 = 60.000Hz$. Frecuencia de sampleo $f_s = 24.000.000Hz$ y tasa de datos $F_B = 600$ bits.

Cuando se recibe un 1 lógico, la amplitud de la señal recibida se sumará al valor de f_r, superándose así el umbral F_d. Si la fase de la señal está bien definida, ecuación (2.4), el sistema se situará en el estado periódico.

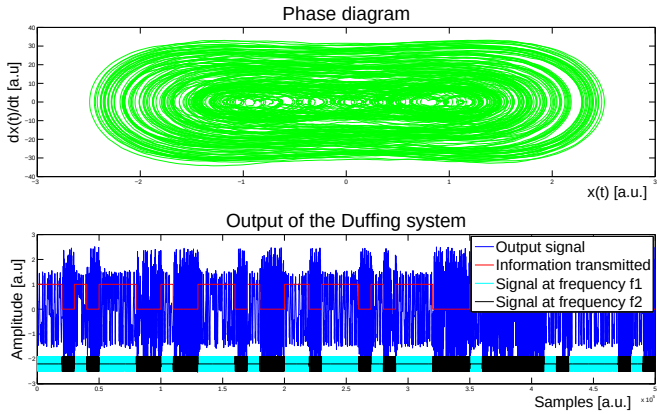


Figura 4.7: Diagrama de fase y salida del Duffing para la detección de una señal BFSK. En la primera imagen se muestra el diagrama de fase. En la segunda imagen se muestra en color azul la salida del oscilador Duffing y sobre ella, en color rojo, aparece la señal de información. Debajo de estas señales se presenta la señal transmitida en dos colores diferentes, representando las distintas frecuencias que la componen.

Para el caso de recibir un 0 lógico, la señal a detectar estará definida a la frecuencia f_2 y su amplitud no contribuirá en el incremento del el valor de f_r por lo que el sistema ha de permanecer en el estado caótico.

En la figura 4.6, se muestra la modulación BFSK antes y después de atravesar el canal ruidoso, colores azul y rojo respectivamente. Se puede apreciar como la señal modulada está compuesta por dos frecuencias diferentes.

En la figura 4.7, se presenta el diagrama de fase de la comunicación completa. Más abajo se presenta la señal modulada, representada en azul claro y negro a fin de diferenciar cuando se transmite un 0 o un 1 lógico. En color azul se representa la salida del oscilador Duffing y sobre dicha señal, en color rojo, la secuencia de 1's y 0's transmitidos.

El código de Matlab para la detección una señal modulada BFSK usando un oscilador Duffing se encuentra en el Apéndice G.3.

Capítulo 5

Evaluación del rendimiento para la modulación ASK

El objetivo que se persigue en este capítulo es la evaluación del rendimiento del sistema de recepción caótico Duffing comparado con otros métodos. Para ello se implementa una comunicación completa y los más parecida posible a una situación real.

5.1 Descripción del sistema completo

El transmisor y el canal ruidoso se implementan conjuntamente. El resultado es una señal OOK modulada donde la portadora esta definida entorno a los $15GHz$ y la tasa de datos es $1Gbps$. El ruido que se ha añadido a la señal modulada es Additive White Gaussian (AWG). Además se tiene en cuenta el ruido de cuantización, puesto que el sistema se probará posteriormente a partir de los datos obtenidos del montaje de laboratorio, donde la señal será digitalizada. La información referente al transmisor y el canal se presenta con más detalle en los Apéndices D.1 y D.2, respectivamente. Se adjunta además la figura 5.1, donde se puede

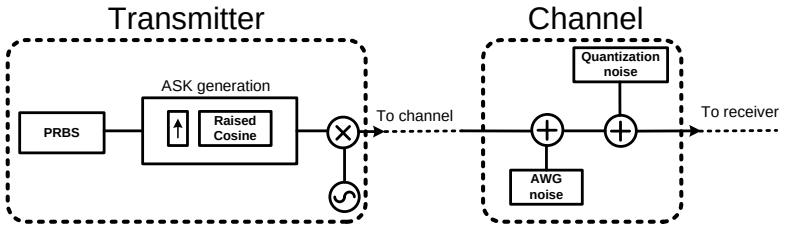


Figura 5.1: Diagrama de bloques del transmisor y el canal.

ver un diagrama de bloque con los elementos que componen esta parte inicial de la comunicación.

El receptor se compone en primera instancia de un BPF y a continuación por el oscilador Duffing. Tras esto, se añade un bloque para la detección de símbolos. Su finalidad es la de interpretar la salida del Duffing y transformarla en una señal binaria, calculando así el BER del sistema. El diagrama de bloques del receptor se presenta en la figura 5.2.

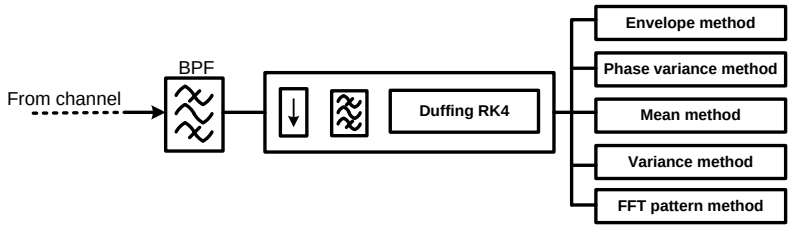
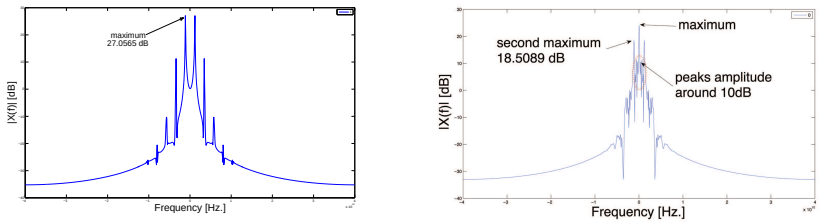


Figura 5.2: Diagrama de bloques del receptor.

Para la interpretación de la salida del Duffing, se han desarrollado y testado cinco métodos diferentes. En el primero, se calcula la envolvente de la salida del Duffing (Apéndice D.3.1). En la segunda propuesta, se analiza la variación de fase de los diferentes estados del sistema en relación con la señal detectada (Apéndice D.3.2). En tercer lugar, se modifica la configuración de nuestro sistema a fin de poder calcular la media de la señal de salida (Apéndice D.3.3). La cuarta alternativa estudia la varianza de la salida del sistema (Apéndice D.3.4). Por último, se trabaja con la señal caracterizada en el dominio frecuencial resultando ser el método más preciso de los cinco, el *método del patrón FFT*. Por tanto, se utiliza este último método para llevar a cabo las simulaciones (Apéndice D.3.5).

5.1.1 Método del patrón FFT

Como sabemos, la señal en el estado periódico tiene cierta periodicidad y esto se ha de ver reflejado en su FFT. El estado caótico también ha de es-



(a) Espectro de un 1 lógico (periódico) (b) Espectro de un 0 lógico (caótico)

Figura 5.3: Espectros de los posibles bits recibidos.

tar caracterizado en frecuencia de algún modo. Con estos planteamientos iniciales, se divide la salida del Duffing en trozos de longitud equivalente a un bit con el fin de caracterizarla.

Evaluando la figura 5.3(a) se observa un máximo en el espectro. Desde la frecuencia del máximo hacia banda base no se encuentra ningún otro máximo. En direccion a frecuencias más altas, los únicos valores significativos que aparecen son los armónicos de la señal. Para el caso de la figura 5.3(b), existe un mayor número de componentes frecuenciales con una amplitud significativa. Hay un máximo en banda base y un submáximo a la frecuencia de referencia, máximo en la figura 5.3(a). Por lo tanto, este último valor frecuencial también tiene gran importancia en el estado caótico. Entre ambos máximos se pueden apreciar numerosas componentes frecuenciales con una amplitud no despreciable.

Una vez conocidos los dos posibles espectros, se implementa un algoritmo que analiza la frecuencia donde existe un máximo para el 1 lógico transmitido, almacenando dicho valor en un vector. Desde esa frecuencia hacia banda base, se busca la componente frecuencial con mayor amplitud, que llamaremos sub-máximo. Este sub-máximo se almacena en otro vector. El principal motivo de utilizar dos muestras por símbolo en la decisión es el gran aumento de exactitud en los resultados.

Una vez se haya almacenado toda la información en los dos vectores, se calculan los umbrales de decisión mediante el uso de histogramas. En la figura 5.4 se ilustra como se lleva a cabo el proceso de decisión a partir

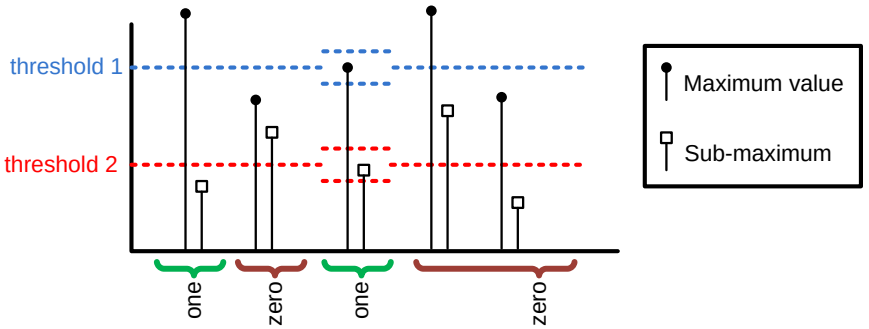


Figura 5.4: Umbrales para la decisión de símbolo. Un símbolo es interpretado como un 1 lógico si su máximo se encuentra próximo a *threshold 1* y su sub-máximo próximo a *threshold 2*. En cualquier otro caso, el símbolo se interpretará como un 0.

de esos dos valores. Este proceso está dividido en cuatro pasos:

- Primero, los símbolos cuyo máximo excede el primer umbral pero su sub-máximo está por debajo del segundo umbral se consideran 1's lógicos. Esto se puede interpretar como una señal con una única componente frecuencial. El comportamiento corresponde al espectro de la figura 5.3(a).

- Se consideran como 0's lógicos aquellos bits que no cumplen el primer paso. Y además, su máximo ha de estar por debajo del primer umbral, así como su sub-máximo ha de ser superior al segundo umbral. Se puede ver este caso como una señal que no tiene una componente frecuencial concreta y suficientemente fuerte, pero que a su vez, está compuesta por varias frecuencias de no despreciable amplitud. Esto sucede en el espectro de la figura 5.3(b).

- En este punto ya se han clasificado los bits más evidentes. Aquellos que no son tan claros, porque sus máximos y sub-máximos no cumplen ambas condiciones, se considerarán como 1's lógicos si sus valores están localizados en un rango en torno al $\pm 11\%$ de cada valor umbral. Principalmente, en este paso se corrigen errores debidos a una decisión de umbrales no óptima.

- Cualquier otro símbolo que no haya sido clasificado se considera como un 0 lógico.

Después del bloque de decisión, la señal se compara con el mensaje original transmitido y se calcula el BER correspondiente. El código de Matlab desarrollado se puede encontrar adjunto en el Apéndice H.7.

5.2 Resultados de la comparación entre el sistema Duffing y la detección coherente

La comparación entre el sistema de recepción caótico Duffing y la detección coherente se lleva a cabo utilizando una frecuencia de muestreo de $80GHz$, una frecuencia de portadora de $13.8GHz$ y una tasa de datos de $1Gbps$. Se analizan ambos sistemas para el rango de valores de Eb/No que va desde los $0dB$ hasta los $16dB$. La implementación del detector coherente se ilustra mediante bloques en la figura 5.5.

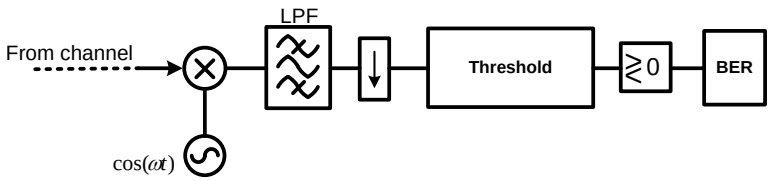
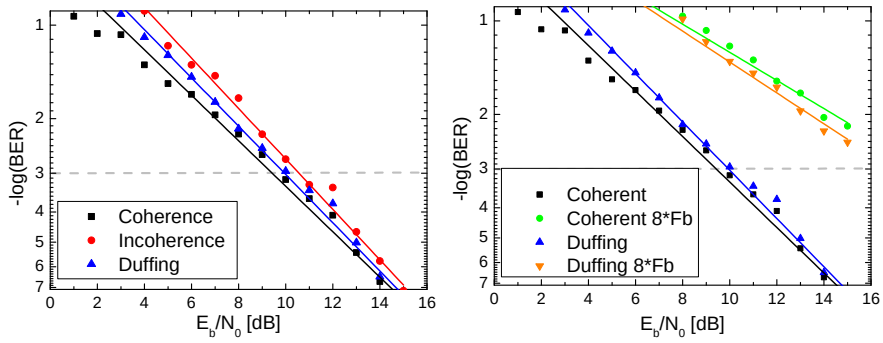


Figura 5.5: Diagrama de bloques de la demodulación coherente.

Los resultados más significativos se encuentran en la figura 5.2. Más gráficas comparativas aparecen explicadas en el Apéndice D.4. En la figura 5.6(a) se presenta la comparación directa entre el sistema Duffing, el método coherente y la detección de envolvente. Se observa como la detección coherente resulta ser el método más efectivo. Por lo tanto nuestro sistema no presenta ninguna mejora para este escenario. Aunque se comporta mejor que el método de detección de envolvente, no se puede tener en cuenta este resultado ya que no se lleva a cabo en las mismas condiciones. El sistema Duffing, así como el detector coherente, requieren una señal de referencia que el detector de envolvente no utiliza.

En la literatura se ha observado como el oscilador Duffing ha sido utilizado en entornos con tasas elevadas de muestreo y sin ningún filtrado de banda previo. Eso nos lleva a comparar directamente el sistema de recepción caótico Duffing con la detección coherente cuando el BPF a la entrada del receptor está definido con ocho veces más ancho que el ancho de banda de la señal. Los resultados se puede ver en la figura 5.6(b). De ahí concluimos que el sistema Duffing tiene un mejor rendimiento para este escenario pero aún así no es suficiente para alcanzar un BER de 10^{-3} con $16dB$ de Eb/No.



(a) BPF definido a $1GHz$ (b) BPF con $8GHz$ de ancho de banda

Figura 5.6: Comparación entre el método coherente, incoherente y el sistema de recepción caótico Duffing. Frecuencia de portadora definida a $13.8GHz$, frecuencia de muestreo de $80GHz$ y tasa de datos a $1Gbps$.

Resumiendo, se observa como el sistema Duffing resulta más efectivo en términos de Eb/No que el detector coherente cuando el filtro de entrada del receptor empeora. Se podría asemejar el comportamiento del oscilador Duffing al de un BPF. Esto puede resultar ventajoso en escenarios donde la frecuencia de portadora no se conoce con precisión o en caso de que un Phase-Locked Loop (PLL) no pueda ser implementado.

5.3 Implementación alternativa del receptor Duffing

En la sección anterior se ha visto como el sistema Duffing rendía mejor que el método de detección de envolvente. Esto es algo lógico debido a la mayor complejidad del sistema Duffing. Lo que se pretende ahora es simplificar este sistema para que no requiera ninguna señal de referencia a la entrada del mismo. Así el sistema de recepción caótico Duffing podrá ser comparado con la detección de envolvente en igualdad de condiciones.

Por ello, eliminamos la entrada de la señal de referencia. Entonces, toda la responsabilidad para controlar el sistema se transfiere a la señal a detectar. Por lo tanto, la señal de entrada se encargará de hacer transitar al sistema de un estado a otro utilizando únicamente su amplitud.

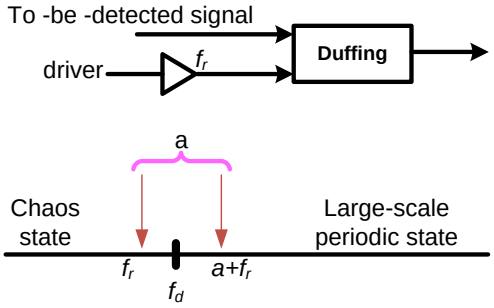


Figura 5.7: Implementación clásica del oscilador Duffing. Se encuentra alimentado por una señal de referencia y la señal a ser detectada. La distancia entre el estado caótico y el estado periódico es mínima.

Observando la figura 5.7, se aprecia la influencia de cada parámetro en la transición. Se ha predefinido el valor del parámetro f_r para que se sitúe el sistema en el umbral de transición entre el caos y la periodicidad. A partir de esto, la amplitud de la señal a ser detectada inducirá al sistema al estado periódico o permanecerá en el estado caótico en función de su valor.

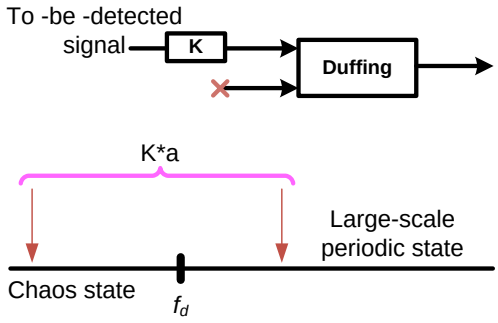


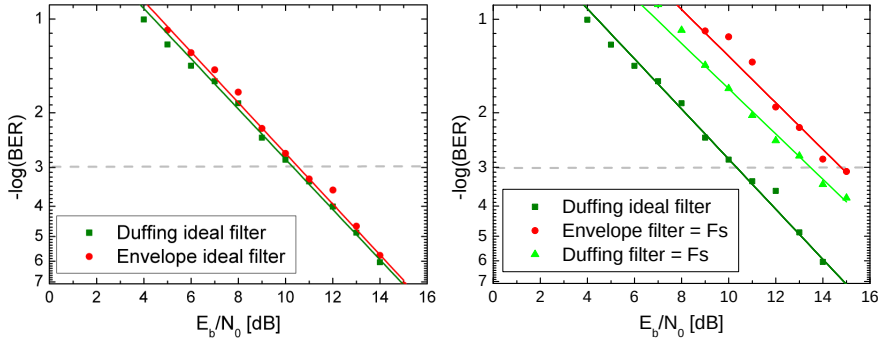
Figura 5.8: Modificación propuesta sobre el diseño convencional del oscilador Duffing. El sistema esta alimentado únicamente por la señal a ser detectada. La distancia entre el estado caótico y el estado periódico es superior al caso clásico.

Una vez que hemos eliminado la señal de referencia, insertamos un nuevo bloque de amplificación. Se sitúa justo antes de alimentar el Duffing (bloque **K** en la figura 5.8). A partir de aquí existen dos posibles casos de estudio:

- Cuando se recibe un 0, no aparece ninguna señal fuerte a la entrada. Solo habrá interferencias de símbolos anteriores o ruido del canal. Aunque todo ello sea amplificado no se alcanza un valor considerable para superar el umbral f_d. Por lo que el sistema no alcanzará el estado periódico.

- La otra situación se da cuando se recibe un 1 lógico. La señal no es muy fuerte por sí sola, pero una vez se amplifique, su amplitud superará el umbral f_d obteniendo una salida periódica de nuestro sistema.

Los resultados más significativos se encuentran en la figura 6.2. Las gráficas restantes en este escenario aparecen explicadas en el Apéndice D.6. Para las simulaciones se utiliza una frecuencia de muestreo de $80GHz$, una frecuencia de portadora de $13.8GHz$ y una tasa de datos de $1Gbps$. En la figura 5.9(a) se presenta la comparación directa entre la



(a) Comparación para un filtro ideal, BPF definido a $1GHz$

(b) Comparación cuando eliminamos el BPF, ancho de banda equivalente de $80GHz$

Figura 5.9: Comparación entre el sistema de recepción caótico Duffing y el método de detección de envolvente con diferentes anchos de banda del BPF. Frecuencia de portadora definida a $13.8GHz$, frecuencia de muestreo de $80GHz$ y tasa de datos a $1Gbps$.

modificación del sistema Duffing tradicional y la detección de envolvente. El FWHM del BPF y la señal, tienen el mismo ancho de banda. Se puede ver que aunque el sistema Duffing tenga un mejor comportamiento, las diferencias entre ambos métodos son prácticamente nulas para un rango de valores de Eb/No, que va entre 0 y $16dB$. En la figura 5.9(b), se elimina el filtro BPF y la señal queda limitada por la frecuencia de muestreo. En este caso obtenemos la máxima ventaja haciendo uso de la modificación del oscilador Duffing. Se requieren $1.5dB$ menos para alcanzar un BER de 10^{-3} que en el caso de la detección de envolvente. Cabe destacar también que existe una penalización de aproximadamente $3dB$ entre el sistema con un BPF ideal y aquel sin ningún tipo de BPF.

Para concluir, se ha visto cómo este diseño rinde mejor en términos de Eb/No que la detección de envolvente. Cada vez que se ha incrementado el FWHM del BPF, las diferencias en cuanto a rendimiento entre ambos métodos han aumentado.

Capítulo 6

Resultados experimentales

En este capítulo se presenta el montaje de laboratorio implementado para verificar si los métodos descritos anteriormente pueden ser utilizados para mejorar los resultados ofrecidos por los métodos tradicionales

Se llevan a cabo dos experimentos. En el primero se pretende caracterizar eléctricamente el sistema Duffing. Sirve como primera aproximación a un escenario real. Permite anticipar futuros problemas que puedan surgir al trabajar con señales reales. En este caso se conecta el Vector Signal Analizer (VSA) directamente al Digital Storage Oscilloscope (DSO), por lo que apenas existe ruido en la comunicación. Para más información sobre la caracterización del sistema de recepción caótico Duffing y los procedimientos seguidos, referirse al Apéndice E.1. En el segundo experimento se implementa un montaje óptico y más tarde, se procesan los datos. Dicho procesado utiliza la modificación del oscilador Duffing convencional.

6.1 Sistema de recepción caótico Duffing para una transmisión de radio sobre fibra

El esquema del montaje de laboratorio detallado se muestra en la figura 6.1. El transmisor es un Teraxion Distributed Feedback Laser (DFB) Narrow Linewidth con un *linewidth* inferior a $50kHz$. La señal PRBS eléctrica es generada a diferentes tasas con el Pulse Pattern Generator (PPG) Agilent HP 70843A. La señal es amplificada a fin de alimentar el Modulador Mach-Zehnder (MZM). El voltaje de *driving* del MZM se ajusta para obtener el máximo ratio de extensión, $17dB$.

La señal que se obtiene ha sido transmitida a lo largo de la fibra para

posteriormente añadir ruido. La fuente de ruido blanco es un Erbium-Doped Fiber Amplifier (EDFA). Tras esto, aparece un filtro de $0.8nm$ de ancho de banda óptico para limitar únicamente el ruido a banda. A continuación, la señal se amplifica de nuevo y se conecta a otro filtro con un ancho de banda óptico de $0.5nm$.

La salida del filtro se une directamente al Atenuador Óptico Variable (VOA) para mantener el nivel de potencia de la señal constante. El láser utilizado como Oscilador Local (LO) es un External Cavity Laser (ECL) con un *linewidth* inferior a $300kHz$. Después se mide la señal con un Power Meter (PM), siendo la potencia del LO $-3.2dBm$ y la potencia de señal $-0.9dBm$. Finalmente, se conecta la señal a un Fotodiodo (PD). Para este experimento han sido usados dos tipos de PD diferentes. Uno para baja frecuencia con $10GHz$ de ancho de banda y amplificador interno. El otro PD para alta frecuencia, $40GHz$ de ancho de banda y amplificación eléctrica auxiliar. La salida del PD está conectada a dos DSO, capaces de capturar 80GSa/s con un ancho de banda de entrada de $30GHz$.

Las características de la señal obtenida son las siguiente: $1Gbps$ de tasa de datos, frecuencia de modulación a $13.8GHz$ para un OSNR entre 18 y $1dB$. La frecuencia de muestreo es $80GHz$.

Con la señal del laboratorio se alimenta el esquemático de la figura 6.2. Este diagrama de bloques viene precedido por un algoritmo de sincronismo de bit (Apéndice E.2.2). Tras esto, la salida sincronizada se divide en tres ramas:

- La primera rama simula un detector coherente con elementos ideales. Su función es la de generar una réplica de la señal transmitida. Para ello, se demodulan los primeros 15 bits y se implementa un algoritmo que dará una réplica de la señal PRBS.
- La segunda rama implementa el sistema de recepción caótico Duffing haciendo uso de la ecuación RK4. Como primer bloque, esta implementado un BPF de ancho de banda variable.
- La última rama es la detección de envolvente. En este caso también lleva un BPF variable para estudiar como afecta a su rendimiento

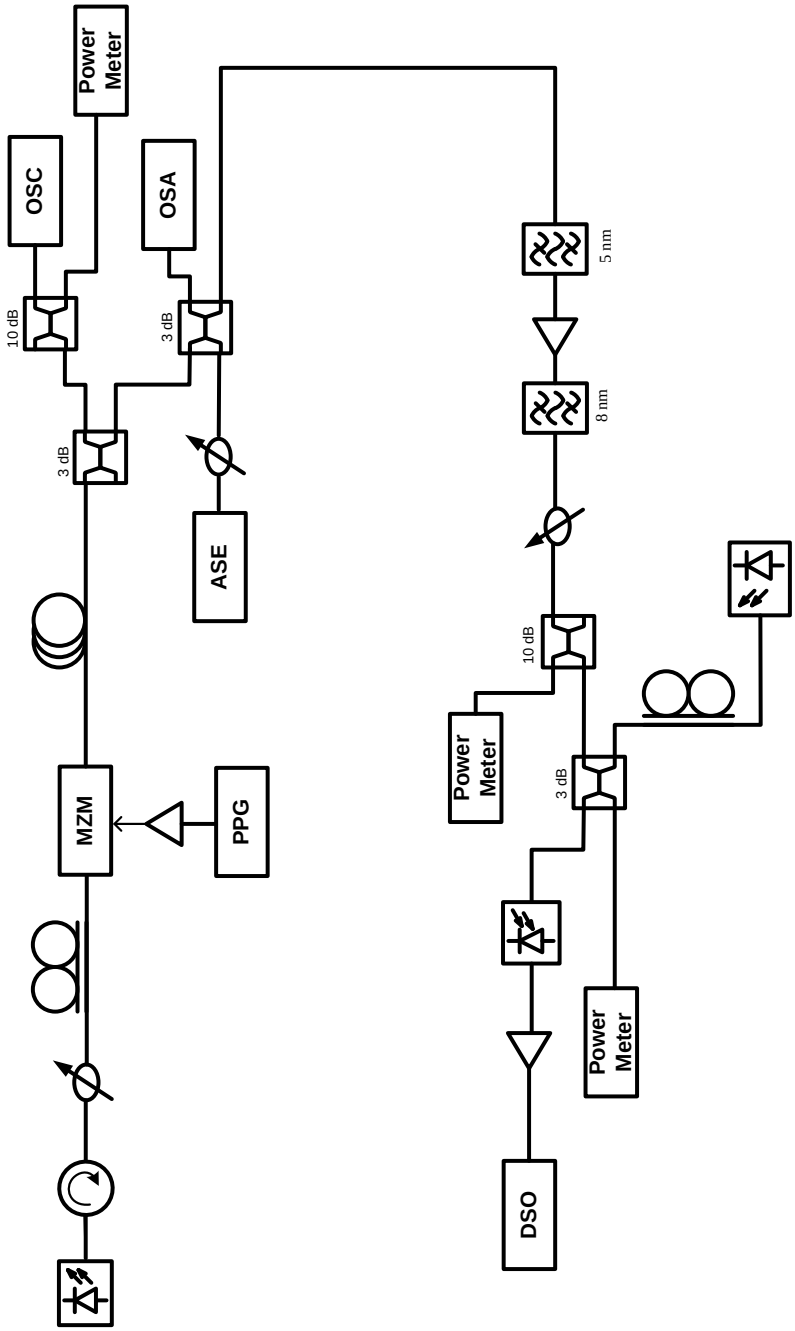


Figura 6.1: Esquemático para una transmisión de RoF. Los parámetros de la comunicación son $1Gbps$ de tasa de datos, $13.8GHz$ de frecuencia de portadora y un OSNR que va de 18 a $1dB$. La frecuencia de muestreo está definida a $80GHz$.

para los diversos escenarios.

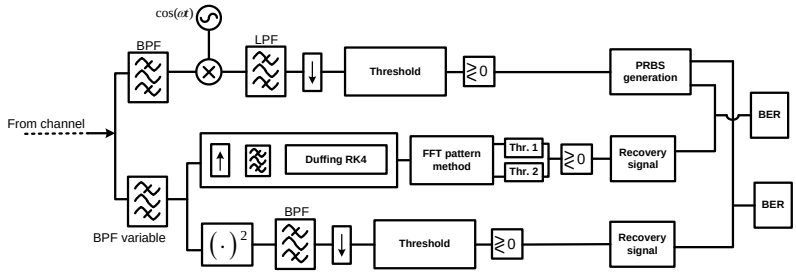


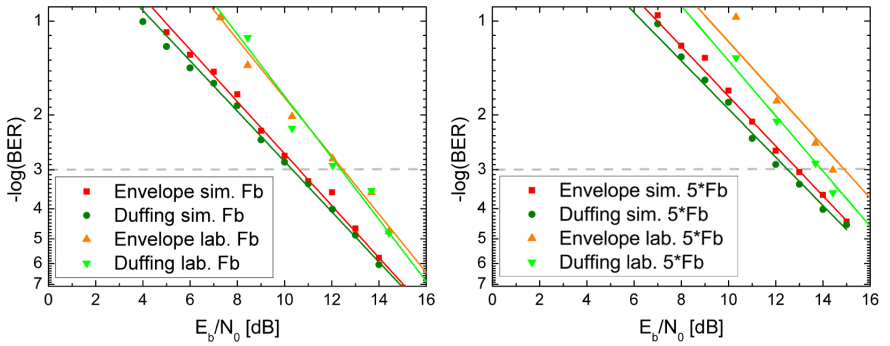
Figura 6.2: Esquemático del demodulador implementado. Está compuesto por tres ramas. La primera simula un receptor coherente. La segunda rama es el sistema Duffing. La tercera rama es el demodulador de envolvente.

6.2 Resultado del montaje de laboratorio

El primer problema aparece al tratar de establecer una relación entre el OSNR medido, que hace referencia al dominio óptico, y el valor Eb/No utilizado en las simulaciones anteriores que es una medida utilizada en las comunicaciones digitales. Para hacer ambos sistemas compatibles se opta por un método empírico (Apéndice E.2.3 y tabla E.1).

Los resultados más significativos se encuentran en la figura 6.2. Las gráficas restantes de este escenario aparecen explicadas en el Apéndice E. De las comparaciones entre la modificación del sistema Duffing convencional con el detector de envolvente, se observa en la figura 6.3(a) el resultado usando un filtro adaptado al ancho de banda de la señal. Ambos métodos tienen un rendimiento similar para los diferentes valores de Eb/No, como fue previsto en las simulaciones explicadas en capítulos anteriores. En el segundo escenario, presentado en la figura 6.3(b), se obtienen mejoras significativas. Para este caso se ha definido un ancho de banda del filtro cinco veces superior al ancho de la señal. La mejoría de $1dB$ entre el detector de envolvente y el sistema Duffing en términos

de Eb/No. Se considera el mejor resultado obtenido y supera las expectativas obtenidas en las simulaciones previas.



(a) Filtro de recepción ideal, $1GHz$. (b) Ancho de banda del BPF a $5GHz$.

Figura 6.3: Comparación entre los resultados experimentales y de simulación para detección de envolvente frente al sistema de recepción caótico Duffing. Los parámetros son $1Gbps$ de tasa de datos, frecuencia de portadora a $13.8GHz$ y frecuencia de muestreo a $80GHz$.

Se puede apreciar una penalización entre los resultados simulados y aquellos experimentales, ver figura 6.2. Esto se debe a diversos factores. Uno de ellos es el hecho de que las simulaciones se hayan llevado a cabo teniendo en cuenta principalmente el ruido Gaussiano y no otros factores que también afectan a las comunicaciones en entornos ópticos. Otro factor es el desconocimiento de forma precisa de la frecuencia entre láseres en el momento de modular la señal. Además, la señal se ha podido ver empeorada a causa de la variación de frecuencia producida en el *beating* del ruido de fase de los láseres.

Para concluir, se ha definido e implementado una variación del sistema Duffing. Su rendimiento se ha comprado en términos de Eb/No con la detección de envolvente, resultando en un mejor comportamiento. Esta mejora lograda se produce cuando el FWHM del BPF en recepción resulta más ancho que la señal de información. El escenario donde las diferencias han sido máximas ha sido para el caso de un BPF de $5GHz$, necesitando $1dB$ menos de Eb/No para alcanzar un BER de 10^{-3}.

Capítulo 7

Conclusión y trabajo futuro

La linea de trabajo de esta tesis se ha basado en el análisis de un enlace de comunicaciones. Con especial interés en la estructura del receptor requerida para la demodulación de señales de naturaleza débil y enmascaradas a su vez por un fuerte ruido. Para ello, hemos hecho uso de un sistema de recepción caótico Duffing. Este sistema ha sido seleccionado frente a otras alternativas debido a la gran cantidad de información existente en la literatura que se valía de dicho método para la detección de señales con niveles negativos de SNR. Sin embargo, nuestro estudio se ha centrado en el análisis del parámetro Eb/No en vez del ya conocido SNR.

La primera de las simulaciones ha sido realizada para un formato de modulación ASK con una frecuencia de portadora de $13.8GHz$ y una tasa de datos de $1Gbps$. Como primer bloque de nuestro sistema de recepción, hemos implementado un BPF con un ancho de banda de $1GHz$. Comparando el sistema Duffing con el método de detección coherente, hemos observado como los resultados no han sido tan buenos como se preveía en un principio. El método de detección coherente ha rendido $0.5dB$ por encima del sistema de recepción caótico Duffing para el caso de un BER de 10^{-3}. Esto es en parte debido al uso de un BPF en recepción. Se preveía un incremento en el rendimiento de este nuevo sistema con respecto al método de detección coherente. La consecuencia de esta disconformidad viene por la alta tasa de muestreo utilizada en la literatura de referencia. Ello permitía la detección de señales débiles para escenarios con bajo SNR debido a la gran diferencia entre los anchos de banda del ruido y la señal. En nuestras simulaciones se ha utilizado el mismo ancho de banda tanto para la señal como para el ruido. Por ello un SNR negativo puede resultar en valores positivos de Eb/No. Dado que en los escenarios de las referencias citadas en el estado del arte el ancho de banda del ruido es mayor, decidimos incrementar dicho ancho en nuestras simulaciones. De esta forma permitiremos comparaciones más

adecuadas entre ambos métodos. Por lo tanto definiremos distintos valores para este BPF que van desde los $2GHz$, en donde las diferencias en cuanto al rendimiento de ambos métodos son prácticamente nulas; hasta los $8GHz$, donde el sistema de recepción caótico Duffing ha alcanzado un mejor rendimiento en términos de BER. En este último escenario necesitamos $1dB$ menos de Eb/No que en detección coherente para alcanzar un BER de 10^{-3}. Como conclusión de las comparaciones entre el sistema de recepción caótica Duffing y el método de detección coherente, podemos observar como el sistema Duffing rinde mejor que el detector coherente cuando se incrementa el FWHM del BPF. Esta característica puede ser aprovechada en escenarios donde portadora no es conocida con precisión o escenarios en los que que un PLL no pudiese ser implementado o integrado en el sistema de recepción.

En las comparaciones llevadas a cabo entre el método de detección de envolvente y el sistema Duffing, este segundo sistema ha rendido mejor en términos de sensibilidad del BER. Esto puede resultar lógico ya que el sistema Duffing requiere una entrada extra como señal de referencia a la frecuencia de portadora, lo que resulta un sistema más complejo de implementar. Por su parte, el método de detección de envolvente no hace uso de dicha señal. Con objeto de simplificar nuestro sistema y compararlo con la detección de envolvente en las mismas condiciones, modificamos el diseño del oscilador Duffing convencional eliminando la señal de referencia. Volvemos a utilizar para las simulaciones el formato de modulación ASK. La frecuencia de portadora y la tasa de datos han sido definidas como $13.8GHz$ y $1Gbps$ respectivamente. Se han realizado diversas simulaciones analizando diferentes anchos de banda del BPF, primer bloque del receptor. Desde valores de $1GHz$ de ancho de banda, en donde la nueva aproximación del oscilador Duffing y el método de detección de envolvente han mostrado un rendimiento similar en términos de BER para valores de Eb/No que van de $0dB$ a $16dB$. Hasta un ancho de banda de $80GHz$, eliminado el BPF en la recepción. Por lo tanto la señal venía solo limitada por la frecuencia de muestreo. Bajo estas condiciones, el sistema de recepción caótico Duffing ha mostrado en un mayor rendimiento en comparación con el método de detección de envolvente. Se han requerido $1.5dB$ menos de Eb/No para alcanzar un BER de 10^{-3}.

Por último, se ha implementado un montaje óptico en el laboratorio con el fin de validar los resultados de las simulaciones. La señal transmitida ha sido modulada a $13.8GHz$ de frecuencia de portadora y $1Gbps$ de tasa de datos. Las diferencias de rendimiento entre la modificación respecto al sistema Duffing tradicional y el método de detección de envolvente han resultado en consonancia con los resultados obtenidos en simulaciones previas. En el último de los escenarios objeto de estudio, el ancho de banda del BPF esta fijado a $5GHz$ de FWHM, manteniendo la frecuencia de portadora y tasa de bit con los mismos valores. El sistema de recepción caótico Duffing ha necesitado $1dB$ menos que el método de detección de envolvente para alcanzar el valor de 10^{-3} de BER. Los resultados experimentales y de simulación han resultado en concordancia.

Esta modificación en el diseño convencional del oscilador Duffing ha presentado un mayor rendimiento en términos de BER para escenarios donde el FWHM del BPF excedía el ancho de la señal. Esta modificación del oscilador Duffing convencional puede ser utilizada como una alternativa válida a los métodos de detección de envolvente cuando la frecuencia de portadora no es conocida con exactitud, o en situaciones donde el ancho de banda del BPF necesitase ser algo relajado debido a factores de coste/diseño, siempre y cuando el incremento en el coste computacional no se presente como un inconveniente.

El presente proyecto aporta nuevos conocimientos al novedoso campo de los receptores caóticos, y abre el camino a nuevos trabajos de investigación, entre los que se puede incluir:

- Caracterización del nuevo diseño del oscilador Duffing para hacerlo extensible a las otras dos modulaciones básicas FSK y PSK.
- Extensión de la modificación del sistema Duffing convencional hacia formatos de modulación más eficientes como podría ser el caso de la modulación Quadrature Amplitude Modulation (QAM).
- Un estudio profundo para la implementación de varios sistemas caóticos de recepción Duffing en paralelo. El objetivo en ese caso sería extender la modulación FSK a ordenes superiores, definiendo en cada uno de los sistemas una frecuencia de portadora diferente.

- En el campo de las comunicaciones seguras se podría tener en cuenta el algoritmo ya desarrollado. Su aplicación más interesante esta en su uso en el transmisor de un sistema para la codificación de la señal. Una vez esta señal fuese transmitida, se utilizaría el mismo algoritmo en el receptor a fin de decodificar la señal detectada.

- Otro punto de estudio interesante es el uso de nuestro sistema en demodulaciones en las que la señal trasmitida haya sido modulada a través de un modulador no lineal. Podría analizarse si este sistema resulta una alternativa válida con un rendimiento mejorado frente a métodos existentes.

- Implementación en tiempo real para Field-programmable Gate Array (FPGA).

Apéndice A

Duffing receiver

The aim of this chapter is to present a mathematical overview about the chaotic Duffing receiving system. We start by defining the basic principles of our second order differential Duffing equation. After this, we describe the procedure to find the bifurcation value which will make our system transit from different states of chaos and stability. Later on in this chapter, we will explain how to characterize this model and which requirements we should fulfil in order to be able to detect weak signals. Finally, we will find the parameter settings required for detecting signals with any frequency.

A.1 Fundamental principle

Theoretically, chaos can be generated through a first-order differential equation [16]. Nevertheless, the regularity of its output would not result easy to study. This is a risk we cannot afford because the output is a determinant factor to distinguish between noise and signal. For that case, signal and noise would play the same role and would have the same weight in affecting the initial conditions of the chaotic time series [17]. This is the main reason why a second-order ODE is employed to implement the chaotic detector:

$$\ddot{x} + \delta\dot{x} + f(x) = f_r \cdot s(t) \tag{A.1}$$

where $f(x)$ is a nonlinear restoring force, and $f_r \cdot s(t)$ is the driving force.

As we mention in the previous chapter; Duffing equation, among all the ODEs, was a good choice for our chaotic detector due to the intensively documentation available. Its associated bifurcation process has already been studied extensively ([18], [19] and [20]).

Thus, we formulate the Duffing equation as follows:

$$\ddot{x} + \delta\dot{x} - x + x^3 = f_r \cdot \cos(t + \theta) \tag{A.2}$$

where the driving force is now defined as a reference signal $f_r \cdot s(t) = f_r \cdot \cos(t + \theta)$ with unknown phase θ.

So as to simplify future implementations, second order equation above is converted to an equivalent system of first order equations and formula (A.2) can be also written as:

$$\begin{cases} \dot{x} = y \\ \dot{y} = x - x^3 + \varepsilon(f_r \cdot \cos(t + \theta) - \delta y) \end{cases} \tag{A.3}$$

where δ is the damping ratio, f_r is the force amplitude and ε is a small scaling parameter.

A.2 Bifurcation value calculated via Melnikov method

The aim of this section is to find numerically the bifurcation threshold. This value causes the transition between states. To achieve that, we start by forcing the scaling parameter to be zero ($\varepsilon = 0$) in equation (A.3)and we obtain an unperturbed system:

$$\begin{cases} \dot{x} = y \\ \dot{y} = x - x^3 \end{cases} \tag{A.4}$$

Equation (A.4) yields a Hamilton system [21]. Its Hamiltonian function is defined as:

$$H(x, y) = \frac{y^2}{2} - \frac{x^4}{4} + \frac{x^6}{6} = h \tag{A.5}$$

Let $h = 0$, the system is composed of two homoclinic [1] orbits, Γ^0_+, Γ^0_-

[1] In mathematics, a homoclinic orbit is a trajectory of a flow of a dynamical system which joins a saddle equilibrium point to itself. More precisely, a homoclinic orbit lies in the intersection of the stable manifold and the unstable manifold of an equilibrium.

and a point $p_0 = (0, 0)$. Or we can also see it, as one center $(0, 0)$ and two hyperbolic saddle points $(1, 0)$ and $(-1, 0)$.

So, the two homoclinic orbits are defined as:

$$\begin{aligned} q_+^0(t) = (\sqrt{2}\text{sech } t, -\sqrt{2}\text{sech } t \cdot \tanh t) \\ q_-^0(t) = -q_+^0(t) \end{aligned} \tag{A.6}$$

From the perspective of $q_\pm(0) = (\pm\sqrt{2}, 0)$ the unperturbative homoclinic orbits can be given as follows:

$$\begin{cases} x_0(t) = \pm\sqrt{2}\text{sech } t \\ y_0(t) = \mp\sqrt{2}\text{sech } t \cdot tht \end{cases} \tag{A.7}$$

Melnikov function is used to compute the parameter values for which a transverse crossing occurs [22]. This function is related to the separation between the stable and unstable manifolds:

$$\begin{aligned} M(t_0) &= \int_{-\infty}^{+\infty} y^0(t)[F\cos\omega(t+t_0) - \delta y^0(t)]dt \\ &= -\sqrt{2}F\int_{-\infty}^{+\infty} \text{sech}t\tanh t\cos\omega(t+t_0)dt - 2\delta\int_{-\infty}^{+\infty} \text{sech}^2 t\tanh^2 tdt \end{aligned} \tag{A.8}$$

With reference to the direct integration and residue theory [23], we can get:

$$M(t_0; F, \delta, \omega) = -\frac{4\delta}{3} + \sqrt{2}F\text{sech}(\frac{\pi\omega}{2})\sin\omega t_0 \tag{A.9}$$

Letting: $I_1 = -\frac{4}{3}, I_2 = \sqrt{2}\pi\omega\text{sech}(\frac{\pi\omega}{2})$.

If a chaotic state occurs, $M(t_0, F, \delta, \omega) = 0$ must have solutions, so we can obtain: $I_1\delta + I_2 f\sin\omega t_0 = 0$.

Therefore: $\left|\frac{I_1\delta}{I_2 f}\right| = |\sin\omega t_0| \leq 1$.

And because of:

$$\frac{dy(t_0)}{dt_0} = \sqrt{2}F\frac{\cos(\omega t_0)}{\cosh(\pi\omega/2)} \tag{A.10}$$

If $\frac{dy(t_0)}{dt_0} \neq 0, |\sin(\omega t_0)| \neq 1$ can be obtained. So we can get: $\frac{F}{\delta} > \frac{I_1}{I_2}$ and we can obtain the following bifurcation threshold for $q_+^0(t)$ or $q_-^0(t)$:

$$R^0(\omega) = \frac{F}{\delta} = \frac{4\cosh(\pi\omega/2)}{3\sqrt{2}\pi\omega} \tag{A.11}$$

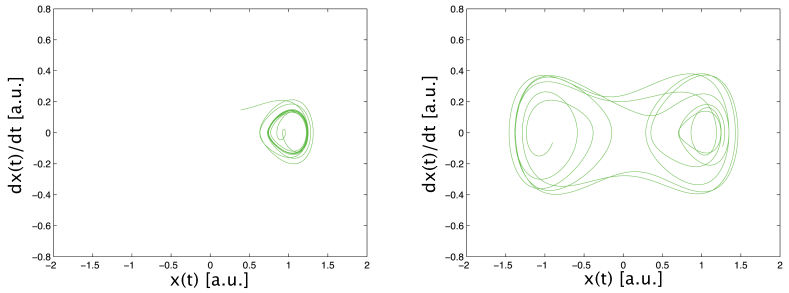
Solving numerically equation A.11, we can know the threshold values for which the system enters a state of chaos, as well into large-scale state when δ established.

For further information in mathematical procedures, refer to [11] and [24]. Above calculations have been done following the these two references.

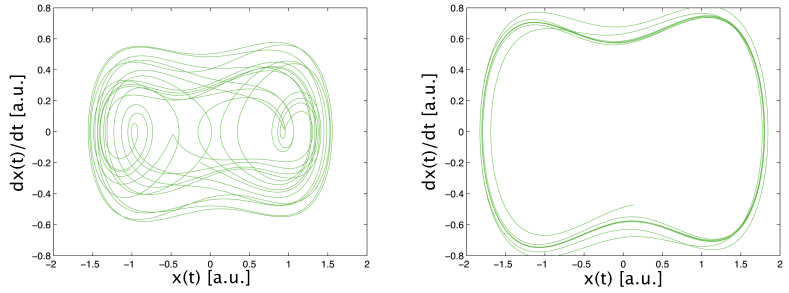
Once the parameter δ is fixed, the system will change regularly when varying the parameter F. Using equation (A.11) we can get that the threshold value is 0.7530 , which means that chaos will appear in the system when $F > 0.3765$ [25].

Depending on the relation between F, δ and the threshold value we identify the following states.

- When $F/\delta < R^0(\omega)$, in the parameter region stable manifold is disjoint with unstable mainfold in the Poincaré mapping of the system [25]. **Period 1 interior trajectory** system in reached, as we can see in the Figure A.1(a).
- When $F/\delta > R^0(\omega)$, stable manifold must be joint with unstable manifold and traverse homoclinic point occurs, reciprocal odd-step bifurcation occurs as shown in Figure A.1(b). We refer to this state as **bifurcation**.
- With the increasing of F, system turns into the **chaotic motion**, as we can see in Figure A.1(c) where the trajectory of the orbits in the phase will tend to fill up a section of the phase space.



(a) Homoclinic Orbit, $F/\delta < R^0(\omega)$ (b) Bifurcation, $F/\delta > R^0(\omega)$



(c) Chaos, $F/\delta > R^0(\omega)$ & $F < F_d$ (d) Great Scale Periodic, $F > F_d$

Figura A.1: Different phase plane diagrams of the chaotic Duffing receiving system representing its four different states. The parameter δ is fixed at 0.5 and the bifurcation threshold $R^0(\omega)$ at 0.3765.

- When F increases, the oscillator turns from the chaotic motion into the non-equilibrium phase state and making F increasing further, $F > F_d$, the **periodic motion** takes place, where the phase plane orbit traces out a closed curve, shown in Figure A.1(d).

From equation (A.3), analytically predicted threshold values are $R^0(\omega) = 0.41$ and $F_d = 0.829$ through numerically research. F_d will be a key factor in future implementations. It will define the border between chaos and periodic state and this fact will be our most important tool in order to differentiate between transmitted logical 1's and logical 0's. In *Table 1*, from the literature [3], we can see some experimental results as reference, where it is illustrated the relation between the bifurcation value

F_d and the step size h, γ_c and h respectively. The main conclusion obtained is that the smallest the step size, the largest the bifurcation value. This property keeps constant until a minimum step size for which the bifurcation value remain constant and equal to 0.816.

A chaotic phase change predicts that weak signal excitation can induce chaotic response. That is why F should be set a little smaller than the critical value F_d. This way, the system is put on the verge of changing to the periodic motion but in a critical state.

A.3 Signal detection model

So far, Duffing has been theoretically characterized. We are going to add the noisy to-be-detected signal $s(t)$ into our system. It represents a small amplitude signal having a little angular frequency difference with respect to the inner driving force. The new Holmes-Duffing equation can be written as:

$$\begin{cases} \dot{x} = y \\ \dot{y} = x - x^3 - ky + f_r \cos t + s(t) \end{cases} \tag{A.12}$$

where $s(t) = a\cos((1+\Delta\omega)t+\varphi)+zs$. zs represents Gaussian white noise with zero mean and variance z. $\Delta\omega$ is the angular frequency difference between the inside driving force and the periodic disturbing signal.

To configure a Duffing oscillator for weak signal detection, firstly we have to assure the oscillator is on the chaotic motion state. It means to adopt f_r a bit smaller than the threshold f_d, but in the edge transforming to large periodic state. Once the to-be-detected signal is included, its amplitude 'a' will contribute to the value of f_r becoming $f_r + a$ a bit larger than f_d. That will change the Duffing oscillator from the previous chaotic state to the stable one.

Numerically, what has happened with the total drive force is the following:

$$f_r \cos t + a\cos((1 + \Delta\omega)t + \varphi)$$

re-written as

$$\begin{aligned} f_r \cos t + a\cos((1+\Delta\omega)t+\varphi) &= \\ = (f_r + a\cos(\Delta\omega t+\varphi))\cos t - a\sin t\sin(\Delta\omega t+\varphi) &= F(t)\cos(t+\theta(t)) \end{aligned}$$

So we can get:

$$\begin{cases} F(t) = \sqrt{f_r^2 + 2f_r a\cos(\Delta\omega t+\varphi) + a^2} \\ \theta(t) = arctan\left[\dfrac{a\sin(\Delta\omega t+\varphi)}{f_r + A\cos(\Delta\omega t+\varphi)}\right] \end{cases} \tag{A.13}$$

Because of $a \ll f$, we can neglect $\theta(t) \approx 0$, and only $F(t)$ affects the system state. It can also be seen from equation (A.13) that the orbit shift is related to the difference of the phase between the external signal and the reference signal [1].

At this point, we can define two cases to study, depending on the phase difference. First, when $\Delta\omega$ is null, valid only for simulation experiments and theoretical explanations. Second, and closer to realistic environments, when there is a difference of frequency between reference and to-be-detected signals. Even using a frequency synchronization scheme based on PLL, it would not be possible to avoid it.

A.3.1 Phase synchronization

There is a second factor to take into account and also a second parameter to calculate in order not to keep the system in the chaotic motion all the time. There is a range of values:

$$\pi - \cos^{-1}\frac{a}{2f_r} \leq \varphi \leq \pi + \cos^{-1}\frac{a}{2f_r} \tag{A.14}$$

where phase variation does not occur, and only if φ is not in this regime, the phase transition will take place. To have an approximate idea of this range, we simplify equation (A.14) using $a \ll f$ and plot a simple graph in Figure A.2.

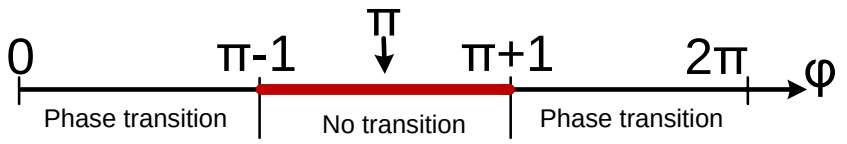


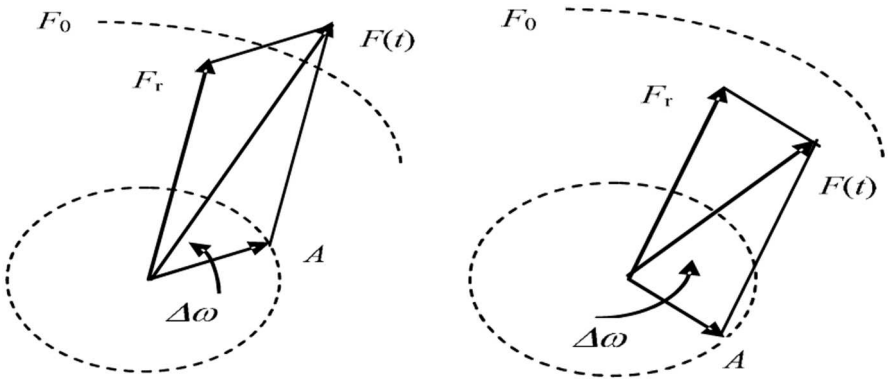
Figura A.2: Duffing oscillator behavior depending on φ.

A.3.2 Small frequency difference

Considering equation (A.13) again, if $\Delta\omega \neq 0$, $F(t)$ will be periodically less than or more than the bifurcation value f_d. Figure A.3 represents the vector chart of the total drive force $F(t)$ in both cases. Supposing the inside driving force vector f_r fixed, the disturbing signal vector will circumrotate very slowly around the former using the angular frequency.

- When driver and to-be-detected signal frequencies are close and their directions tend to consistency, the resultant vector becomes the sum of both forces and makes the total driving force amplitude larger than in some time range $a + f_r > f_d$. The consequence is that the oscillator transition occurs from chaos state to the large periodic state [26], see graphical description in Figure A.3(a).
- The other situation is shown in Figure A.3(b), where force directions tend to deviate from each other and the resultant vector makes the total drive force amplitude smaller than in some time range $a + f_r < f_d$. The oscillator will remain in the chaotic motion state.

In conclusion, since the frequency of $s(t)$ is the same as that of the referenced signal, the system described by equation (A.12) will cause a phase shift, even if its amplitude is weak. This means that the system will convert the chaotic motion into a large periodic motion. When the frequency of weak signal $s(t)$ is different and far from that of referenced signal, the system will not induce phase shift. Otherwise, phase transition could be done. This property of bifurcation can be used as an indicator for detecting weak signals. When the system is fixed in a critical chaotic state, the signal $a\cos\omega t$ is null and only white noise zs exists, since the



(a) $F(t)$ larger than the bifurcation value (b) $F(t)$ smaller than the bifurcation value

Figura A.3: The vector relation of the chief instigating force. *This figure has been taken from [1] and slightly modified in order to be used for the above explanation.*

system is immune to a wide range of additive white noise, it remains in the chaotic state. When the signal is detected, even though it is weak, the system will go into periodic state, the phase transition state from chaotic to periodic state is the criterion whether the signal exists.

A.4 Detection model for any frequency signal

Duffing system in equation (A.3) can only detect low-frequency sinusoidal signals. In order to make the equation (A.3) suitable for detecting high-frequency sine signal and keep the state of chaos [27], it can be modified as follows:

$$\ddot{x}(t) + \delta\dot{x}(t) - x(t) + x^3(t) = \gamma\cos(t + \theta) \tag{A.15}$$

Let $t = \omega\xi$, then

$$x(t) = x(\omega\xi) = y(\xi) \tag{A.16}$$

$$\dot{x}(t) = \frac{dx(t)}{dt} = \frac{1}{\omega}\frac{dy(\xi)}{d\xi} = \frac{1}{\omega}\dot{y}(\xi) \tag{A.17}$$

$$\ddot{x}(t) = \frac{1}{\omega^2}\frac{d^2y(\xi)}{d\xi^2} = \frac{1}{\omega^2}\ddot{y}(\xi) \tag{A.18}$$

By substituting (A.16), (A.17), (A.18) into equation (A.15), we obtain

$$\frac{1}{\omega^2} + \frac{b}{\omega}\dot{y}(\xi) - y(\xi) + y^3(\xi) = c\cos(\omega\xi) \tag{A.19}$$

Let $y(\xi) = x_1$, $x_2 = \frac{1}{\omega}\dot{x}_1$, $\xi = t$, formula (A.19) can be written in the form of a state equation

$$\begin{cases} \dot{x}_1 = \omega x_2 \\ \dot{x}_2 = \omega(x_1 - x_1^3 - bx_2 + c\cos\omega t) \end{cases} \tag{A.20}$$

From equation (A.20), we can see that the parameter ω does not have any influence on Hamilton equation and Melnikov judging method. From reference [28], the change in frequency has no influence on the threshold value of Duffing oscillator, just the speed of the system is different from each other. Therefore, our system will be able to accept signals whose frequency is not necessary equal to 1. This is a key step in order to simulate scenarios closer to a realistic RoF communication (e.g. 1 or $5GHz$ of carrier frequency).

Apéndice B

Model Implementation

Once the mathematical and more theoretical part of the Duffing oscillator has been explained, we move to the implementing a simulation model. Duffing oscillator will become the first block in the receiver scheme, hence the importance of an accurate design. The target is to solve firstly equation A.12 and subsequently improve the model in order be able to introduce high-frequency signal carriers to solve also equation A.20.

In this chapter, we will develop a Simulink implementation of the Duffing, as a first approach. After that, we will solve the model using Matlab in order to obtain a high control level of every block. It will be also shown an evolution in the implementation. Starting with one of the simplest integral method (Euler), going through second-order methods and wrapping up with more complex solutions. Later on, those methods will be compared and their behavior characterized.

B.1 Simulink Model

From publications [14] and [15], we decide to use Matlab Simulink to implement the Duffing oscillator. This software was chosen because it is one of the most popular tools to solve differential-equations due to its extensive library which allows easy implementation of any control block.

From Figure B.1 we can see all the blocks which compose the Duffing oscillator. The result is a system that works sample by sample with the incoming signal. Main blocks and functions are commented below.

Firstly, to-be-detected signal together with the reference signal are added in the block *ADD1* with the previous output sample of the Duffing. All

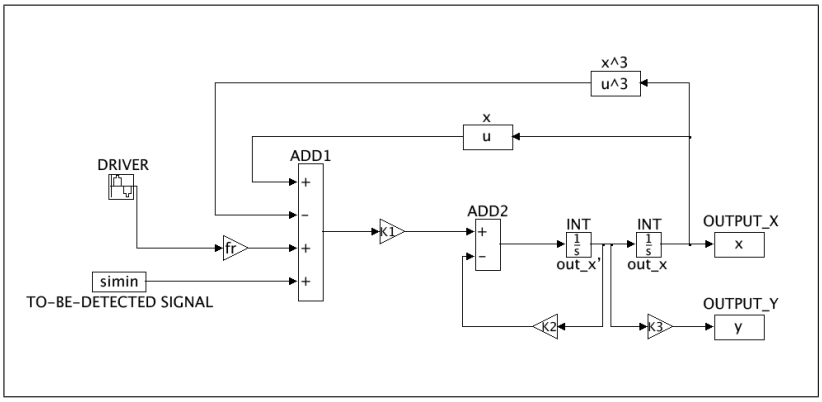


Figura B.1: Simulink implementation of Duffing oscillator.

together are amplified. Before the integration block, first-order output is subtracted from the incoming signal, in the block *ADD2*. Subsequently the signal is integrated twice (in blocks named as *INT*) in order to solve the second-order ODE.

Both outputs, referred to as $OUTPUT\_X$ and $OUTPUT\_Y$, are used to plot the phase diagram. This plot will be used to experimentally define in which state, between chaos and large periodic, Duffing oscillator is working depending on the different values of the constant f_r. Doing that, we will be able to define the maximum value of the reference signal slightly lower than the bifurcation value for signal detection.

Output data from $OUTPUT\_X$ are also useful to study and analyze the Duffing behavior to the incoming signal. From this block, we will take the data to carry out the demodulation.

Finally, signal output from the second integral block is non-linearly modified and set ready for being added again to a new incoming input in the next iteration.

The main objective of this implementation is to get a first accurate approach to this kind of systems. A big effort has been put in the characterization of the two states. They will be the main tools to detect the signal. As we can see from the plots below (Fig. B.2(b) and Fig. B.3(b)),

obtained results are quite similar to [29].

In our case, factor f_d is defined experimentally to be in the verge of bifurcation with the value of 0.692.

Thus, $f_r = 0.71$ is used in the fist image (Fig. B.1) to assure Duffing to be in the large scale periodic state ($f_r > f_d$). From the phase diagram (Fig. B.2(a)) we can see how periodicity is kept since the first iteration. Its corresponding output signal is shown in Figure B.2(c) where also periodicity can be appreciated.

The only point which could disturb the periodicity in the phase diagram is the beginning of the iteration. It is due to the simulation starting point which is defined at $(0, 0)$. So initial conditions of the INT blocks are defined as zero.

Figure B.2(b) uses the same simulation parameters as Figure B.2(a) unlike in this scenario, signal is submerged in the noise. Figure B.2(d) is the corresponding output of the system when noise is included. We observe that signal is mostly periodic. In future chapters it will be explained how to avoid these small irregularities in order to get an accurate detection.

The other state that has to be studied in the chaotic one. In the previous chapter it was explained how to go from one through the other defining a limit value. For this scenario, that value was experientially found. In order to get chaos, we define $f_r = 0.68$ ($f_r < f_d$).

Figures B.3(a) and B.3(b) show the phase diagram. These plots are really useful and almost essential when you are characterizing your system because calculations are not needed to identify in which state the system is. Thus system can be defined without further information. Later on, when chaotic and periodic states are mixed in a real transmission, other methods will be required to distinguish the current state.

From figures B.3(b) and B.3(d), hardly any information can be obtained. No periodicity in the output of the system is the only characteristic which will help us in a future demodulation process.

After some simulations were carried out and even knowing that most of the published experiments were solved with Simulink, we decided to go further in the Duffing implementation. The problem we found using Simulink lies in not having the control of the data since they go through blocks. Until you recover the information from the output, all blocks in between are connected and everything is pre-defined. A time vector could be defined in the entrance but you will not be able to control the integration step size as integrals can be solved with variable-steps. Furthermore, simulations in this environment are also remarkably slow.

These difficulties added to the point that iteration with Matlab was always necessary, encouraged us to define a new entire Duffing model in Matlab.

B.2 Matlab Model

The difficulties of building a digital system with feedback are more than compensated with the high degree of control obtained in your own system.

The first approach consists in decomposing the Simulink model shown above (section B.1), studying the input and the output. In such case, we will be able to calculate delays and other relations among blocks. Once all the blocks were defined we can get ones and others in touch.

To-be-detected signal block is replaced by an OOK signal with AWG noise. Driver is now, a sinusoidal signal with the same frequency as the to-be-detected signal. The critical block in our system will be the integral, where a first ODE has to be solved twice. The goal is to define the easiest algorithm possible, without neglecting the accuracy of our solution. Performance is also a point to be taken into account due to the high number of integrals needed for each iteration.

Many differential equations cannot be solved analytically, thus in these cases we must come up with an approximation to the solutions. The algorithms studied below can be used to compute such approximation. An alternative method would be the use of techniques from *calculus* to

obtain a series expansion of the solution.

B.2.1 Euler Integral Method

The first proposal to solve ODE is the Euler integral method, using the formula $y_{n+1} = y_n + h * f(x_n, y_n)$ which advances a solution from x_n to $x_{n+1} = x_n + h$. Where h is the interval step between samples and $f(x_n, y_n)$ the nth value of the function to be integrated.

Note that the method increments a solution through an interval while using derivative information only at the beginning of the interval (explicit method). As a result, the step error is $O(h^2)$ [30]. This method is called simply "the Euler method", although it is actually the forward version of the analogous Euler backward method.

From [31], the method is described as neither very accurate nor very stable when compared to other methods using the same step size, the accuracy is actually not too bad and the stability turns out to be reasonable as long as the so-called Courant-Friedrichs-Lewy condition is fulfilled. This condition states that, given a space discretization, a time step bigger than some computable quantity should not be taken. In situations where this limitation is acceptable, Euler's forward method becomes quite attractive because of its simplicity of implementation [32].

Euler backward is also tested in our Duffing (implicit method). The implemented formula is $y_{n+1} = y_n + h * f(x_{n+1}, y_{n+1})$. For this case $f(x_{n+1}, y_{n+1})$ the difference is that the derivative is evaluated at point $n+1$th instead of at point nth.

A third method is also implemented, in that case we do not take the information neither from the beginning nor the end on the interval, evaluated at point $n+\frac{1}{2}$th. It is taken from the value in the middle of the interval and its formula is $y_{n+1} = y_n + h * f(x_{n+\frac{1}{2}}, y_{n+\frac{1}{2}})$.

From simulations we can conclude that Forward Euler integral is the most accurate method among the three Euler integral methods implemented. Results can be seen in Figure B.4 in comparison with Figure B.5. The

first plot represents the result of the first integral in the system. In blue, it will always be the output of Simulink and over that, it is plotted the output of the integral implemented in Matlab. The second plot shows the second integral consecutive in the system. It is fed by the output of the first integral, so errors could be caused by the previous integral block. Second signals (red or green color) show the evolution of the Euler integrals.

It is seen how, in the case of Forward Euler, Matlab output is still converging. By contrast, it does not happen the same for the other two integral approximation methods. They have similar values for the first integral. But in the second plot, there is not any kind of convergence between both signals. This situation is illustrated in the second plot of Figure B.5 for the case of Backward Euler, where signals follow different paths.

It is obvious that Euler Forward is the closer method to the Simulink case because it follows the same trajectory. But anyway, for large vector the integral is not accurate enough and in order to prevent future errors from that case when simulations are quite long, we decide to implement a better approximation.

Another factors of this method are analyzed in section B.2.4. Matlab code for the implementation can be found in the Appendix F.1.

B.2.2 Trapezoidal Rule for Integration

Large amount of data will be simulated and errors will appear. In order to solve ODEs more accurately and limit errors as much as possible, we replace Euler method for the Trapezoidal rule. It means a second-order method instead of a first-order method. From an integral which approximates its value by the area of a rectangle to a method which makes use of the approximation by the area of a trapezoid. Trapezoidal rule has a local error $O(h^3)$.

The choice of the previous function can be refined to obtain more accuracy with fewer evaluations of the function $f(x)$. We have seen that

using a constant value for the approximation to the function $f(x)$ on each interval leads to a roughly triangular area that either over or under estimates the area beneath $f(x)$. This roughly triangular characteristic suggests the simple refinement of using a straight line to approximate $f(x)$ on each interval [33]. This straight line is the chord joining the points a and b on the curve, as we can see in Figure B.6. The equation for this method can be written as: $y_{n+1} = y_n + h * \frac{f(x_n)+f(x_{n+1})}{2}$

Once the method is already implemented and tested, we can see how much precise and stable than Euler Forward is, comparisons are shown in Figure B.8 and table B.1.

Trapezoidal Rule is also a more stable method. However, comparing Forward Euler and Trapezoidal Rule, the second one takes much more time to converge [34]. It is shown in Figure B.7 where Duffing oscillator was defined in the large scale periodic. So, the output of the system should be a sine periodic signal form the first sample but it takes around 5000 samples to achieve the convergence where less than 100 were needed in the Forward Euler method implementation. This fact could entail the lost of the first data in the communication using the current method.

There is another method that can be implemented in order to improve Trapezoidal Rule method and go a little bit further in the goal of implementing the most accurate system as possible, we can use Runge-Kutta methods. These methods are still one step methods, but they depend on estimations of the solution at different points.

Other factors of Trapezoidal Rule method are analyzed and compared in B.2.4. Matlab code for the implementation can be found in F.2.

B.2.3 Runge-Kutta Method

The main advantages of Runge-Kutta methods are that they are still easy to be implemented, they are very stable, and they are “self-starting” (i.e., unlike multi-step methods, we do not have to treat the first few steps taken by a single-step integration method as special cases).

Among all the others, we use the RK4 algorithm to solve the Duffing equation [35]. Four evaluations of $f(x)$ at every timestep will be required. Therefore, the system is a discrete dynamic system by nature. Different step sizes obtain different maps from the same differential equation. The dynamics of all these derived discrete systems are similar, but slightly different from the original continuous system.

The common RK4 equation implemented in matlab is the next:

$$\begin{aligned} y(t_0) &= y_0 \\ \\ k_1 &= f(t_n, y_n) \\ k_2 &= f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1) \\ k_3 &= f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2) \\ k_4 &= f(t_n + h, y_n + hk_3) \end{aligned}$$

$$\begin{aligned} t_{n+1} &= t_n + h \\ y_{n+1} &= y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

where y_0 defines the initial value.

Thus, the next value y_{n+1} in the output will be determined by the present value y_n added to the step size of the interval h and an estimated slope. The slope is a weighted average of slopes:

- k_1 is the slope at the beginning of the interval.
- k_2 is the slope at the midpoint of the interval, using slope k_1 to determine the value of y at the point $t_n + \frac{1}{2}h$ using Euler's method.
- k_3 is again the slope at the midpoint, but now using the slope k_2.
- k_4 is the slope at the end of the interval, using in this case k_3.

A weighted average of the slopes is performed, giving more importance to the slopes at the midpoint:

$$slope = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

As we know, there is truncation error (also named as discretization error) linked to a Runge-Kutta algorithm. It exists even with infinite precision arithmetic, because it is caused by truncation of the infinite Taylor series to form the algorithm. Discretization error depends on the step size used, and the dependence is especially distinct when the system is strongly nonlinear.

As far as our system is concerned, if the step size used is different, the truncation error will bring a distinguishable discrepancy of the critical value f_r. Whatever the value of the step size is, the phase transition itself is clear and distinct; what makes the difference is just the value of f_r. The truncation error does not mean that the step size is required to be very small to detect chaos onset accurately.

The most difficult part of the implementation was the concatenation of the two integrals in our system. Output data from one block were needed as an input for the second one. How it was solved and implemented is shown in Appendix F.3. Other factors of this method are analyzed and compared in section B.2.4.

B.2.4 Comparative of integral methods

To chose the integral method that will match better in our system, we define as the most important criteria the three following factors: performance, stability, and accuracy.

Performance refers to the computational cost of simulations for a given timestep. Stability refers to how well the integrator copes with stiff constraints such as high spring constants before errors become unacceptably large. Accuracy refers to how well the integrator matches the expected result.

For stability, systems are tested when the damping ratio δ is defined as 0.5. This will be the value used in all the simulations. This choice is supported by the fact that in most of the papers same value is used (e.g., [27] where is named as b or [24], etc).

In Figure B.8, it can be observed the behavior of the Forward Euler method, the Trapezoidal Rule method and Runge-Kutta implementation. The one which tends to zero is defined as the most stable. Ordered from least to most stable: Euler, Trapezoidal and RK4.

Regarding accuracy, there are also some simulations which show the error with respect to the exact value. Table B.1 illustrates the errors of each case for a simulation. We can state that higher order integrators are most accurate. So, RK4 is the most accurate. Then, it is Trapezoidal Rule method and the least accurate, is the Forward Euler integral method.

Time	Euler Error	Trapezoidal Error	RK4 Error
0,4	0,00651	1,79E-05	8,85E-10
0,8	0,03939	1,44E-04	1,98E-09
1,2	0,21938	5,59E-04	2,57E-08
1,6	0,902	4,85E-04	1,12E-07
2	2,8816	3,36E-03	3,57E-07

Tabla B.1: Numerical results for analyzing the accuracy between Forward Euler integral method, Trapezoidal Rule method and the Runge-Kutta implementation.

Performance is the last criterion to evaluate. We run the same simulation for the three methods and therefore, we can classify them from slow to faster: RK4, Trapezoidal and Euler. RK4 method resulted four times slower than Euler method.

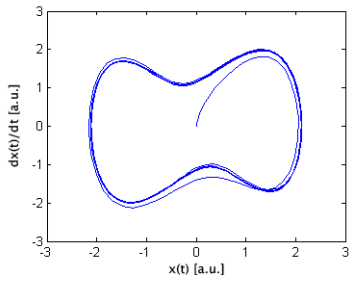
To summarize, Table B.2 indicates the ranking of each integrator in terms of stability, accuracy and performance.

As conclusion, Forward Euler method requires the lowest computational power and it is the easiest to implement. However, it is the least accurate and stable. Trapezoidal method is much more accurate than Euler method and also more stable. This system will be considered for scenarios

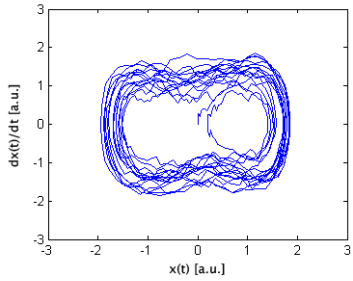
Method	Stability	Accuracy	Performance
Euler Forward	low	low	high
Trapezoidal	medium	medium	medium
Runge-Kutta	high	high	low

Tabla B.2: Comparative between Forward Euler integral method, Trapezoidal Rule method and the Runge-Kutta implementation in terms of stability, accuracy and performance.

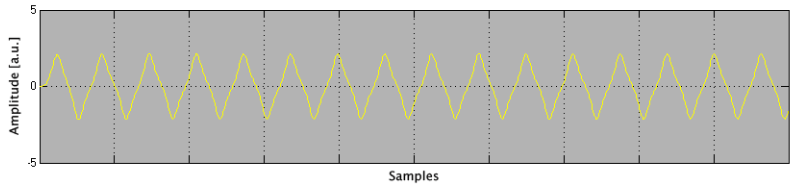
where a high amount of data is processed and the computational time becomes important. In the last case we analyze the Runge-Kutta method, its primary disadvantage is that requires significantly more computation time than multi-step methods of comparable accuracy and mainly compared to the developed methods above, and it does not easily yield good global estimations of the truncation error. However, for the systems under investigation in this thesis, the advantage of the relative simplicity and ease of use of RK4 method far outweighs the disadvantage of their relatively high computational cost. It will be used in most of the simulations as well as in the main Duffing characterization.



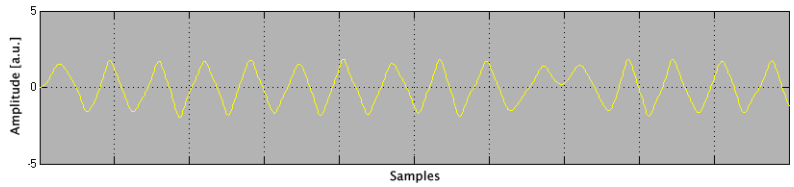
(a) Phase diagram in periodic state, no input noise



(b) Phase diagram in periodic state, noisy input

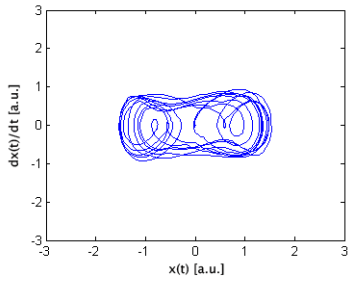


(c) System output in the periodic state free of noise

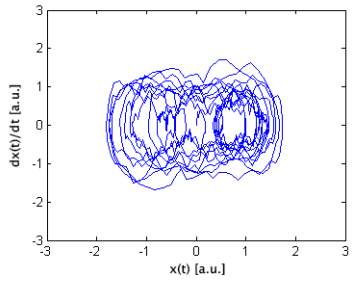


(d) System output in the periodic state with noisy input

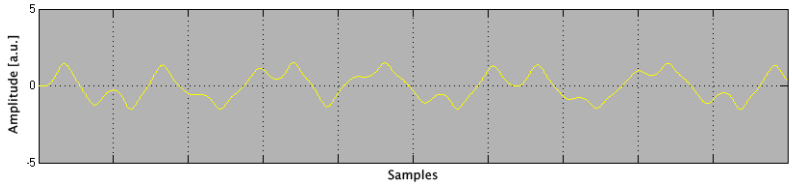
Figura B.2: Large scale periodic state - Matlab Simulink results.



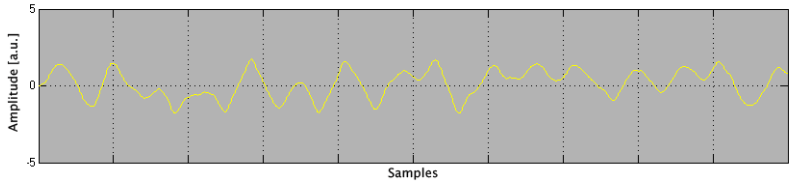
(a) Phase diagram in chaos, no input noise



(b) Phase diagram in chaos, noisy input



(c) System output in chaos free of noise



(d) System output in chaos with noisy input

Figura B.3: Chaotic state - Matlab Simulink results.

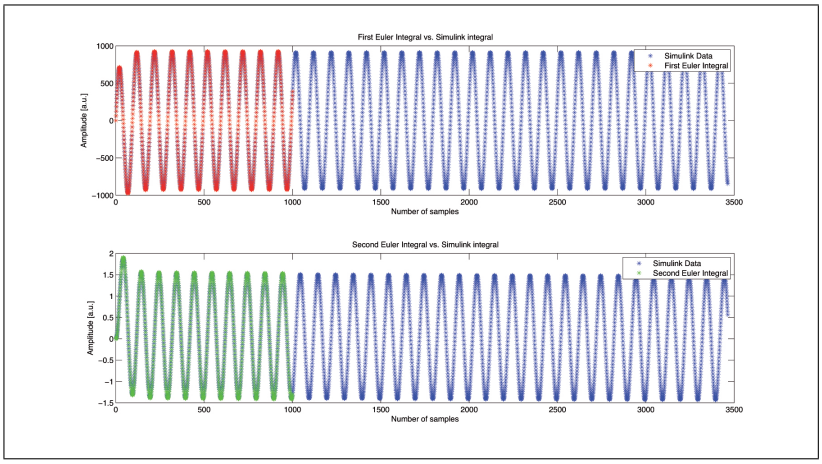


Figura B.4: Forward Euler Integration method. The first plot represents the result of the first integral in the system using Forward Euler. The second plot is the output of a second Forward Euler integral, fed by the output of the first Forward Euler integral block. Simulink outputs are plotted in color blue, they represent the most accurate solution. Red color signal represents the output of the first integral and green color signal shows the second integral, both obtained through Matlab simulation.

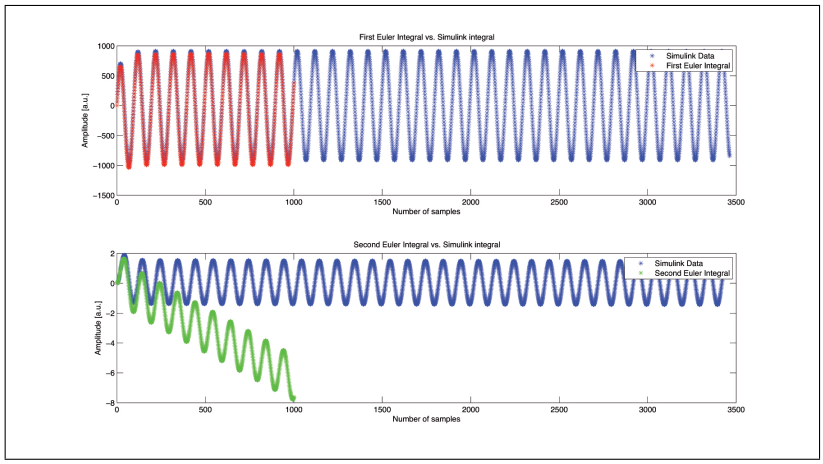


Figura B.5: Backward Euler Integration. The first plot represents the result of the first integral in the system using Backward Euler. The second plot is the output of a second Backward Euler integral, fed by the output of the first Backward Euler integral block. Simulink outputs are plotted in color blue, they represent the most accurate solution. Red color signal represents the output of the first integral and green color signal shows the second integral, both obtained through Matlab simulation.

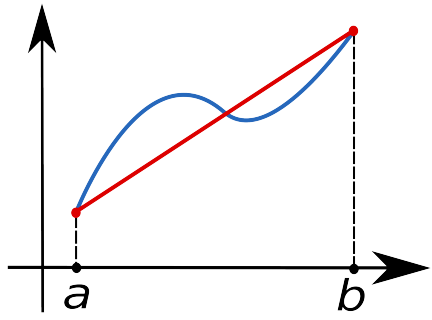


Figura B.6: Trapezoidal Rule, $f(x)$ (blue) is approximated by a linear function (red). Image obtained from http://wikipedia.org

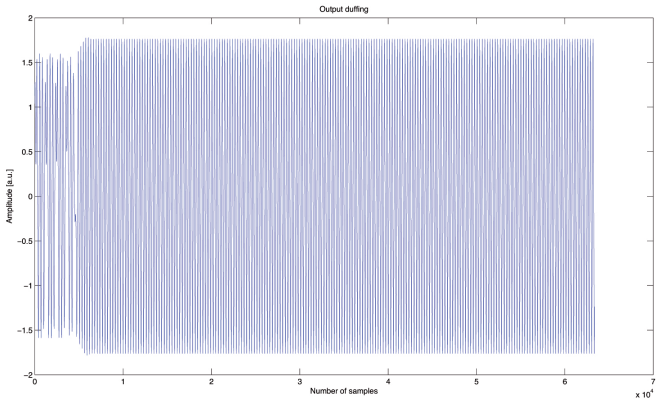


Figura B.7: Convergence of Trapezoidal Rule. Duffing oscillator set in the large scale periodic and its output convergence is achieved after 5000 samples. The consequence is the lost of the first data in the communication.

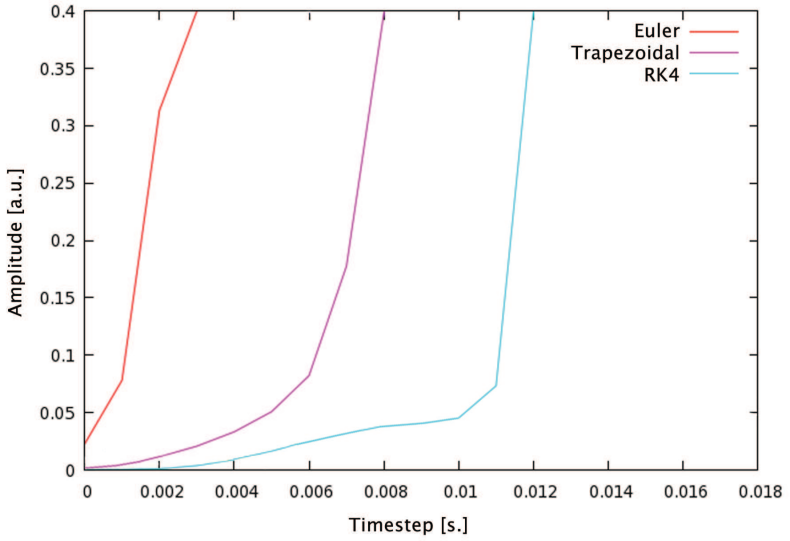


Figura B.8: Underdamped System Stability.

Apéndice C

Simulation results

We have already characterized the theoretical behavior of the Duffing in chapter A and we have also defined the states which are interesting from the weak signal detection point of view. Later on, we have implemented different simulation models and we have briefly defined in which cases will be better than others (chapter B).

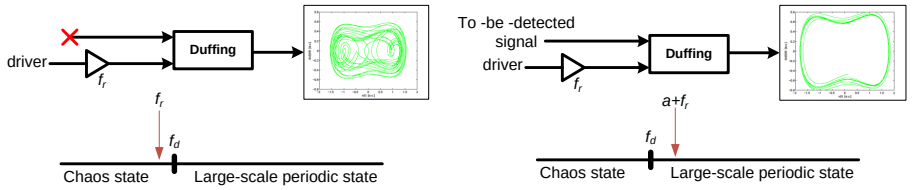
Most of the information in the field of Duffing oscillator is presenting the system as a method to detect weak signals. But it is difficult to find an example of where they go trough a specific modulation. This thesis wants to focus in how the system should be defined and characterized in order to be used for the most basic modulation techniques. For that, we have defined three section. The first is center on the most common case, the detection of ASK. The second section is dedicated to FSK detection and the third section is focused in the PSK modulation.

C.1 A method for 2ASK signal detection using Duffing Oscillator

For the purpose of our simulation, we are going to use OOK modulation which is the special case of ASK, where no carrier is present during the transmission of a zero. OOK is chosen in order to be closer to ideal definition because, as we will see later, periodic signal is responsible for the increase of the value of the parameter f_r which will be so relevant in the state transition. So, we have tried to define a signal which is only periodic when '1' is transmitted and that does not affect our system in any other case.

The two most important parameters are f_r, the strength of the reference

signal going into the Duffing oscillator and f_d, the value that defines the threshold of the Duffing oscillator. That is the exact point from where system enters the state of large scale periodicity from the state of chaos or vice versa. This value is theoretically calculated in advance, but it would be better to fix it experimentally later by means of numerical approximations. Thus, it is convenient to test the system when only reference signal is used and any other incoming signal feeds the Duffing in order to be as much accurate as possible defining f_r close to f_d.



(a) 0 is transmitted

(b) 1 is transmitted

Figura C.1: Phase plane diagram.

In order the Duffing oscillator to be able to transit from one state to the other, there is a second condition to fulfil, the phase of the reference signal (φ) has to be out of the next regime: $\pi - \cos^{-1}\frac{a}{2f_r} \leq \varphi \leq \pi + \cos^{-1}\frac{a}{2f_r}$. More details about the phase can be found in subsection A.3.1.

As we can see in Figure C.1(a), only the driver feeds the system. Its amplitude value defined by f_r is situated close to f_d. However the output remains in the chaotic state because it value is still small to exceed the bifurcation value. Then we feed the system with the modulated signal. The periodicity of this new signal will be added to the driver and they will exceed the bifurcation value setting the system into the large periodic state (see Figure C.1(b)).

Duffing behavior to distinguish among transmitted data:

- if a logical '0' is transmitted, system will be in the situation of Figure C.1(a) in which there is not any periodicity coming through

the signal input. There is only AWG noise and even if it is intense, it can not cause any phase transition.

- if a logical '1' is transmitted, there will be a periodic signal in the entrance. Its amplitude (even a tiny amplitude) will be added to the driver and it will lead to phase transition to great periodic motion as long as it results in $f_r + a > f_d$ (Figure C.1(b)).

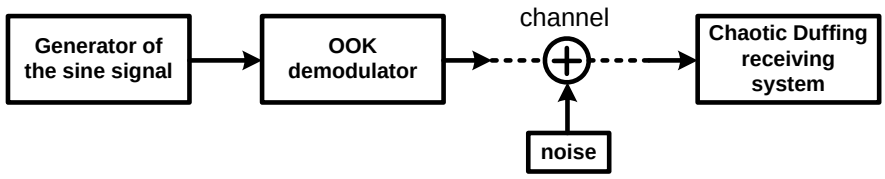


Figura C.2: OOK set-up. It is composed of a block for the generation of the signal and subsequent OOK modulation. After transmission, the received signal and noise feed the chaotic Duffing receiving for offline processing.

For the simulation, we have followed [27] and therefore the basic set-up shown in Figure C.2 has been implemented: PRBS signal is generated and amplitude modulated. Then, it goes to the channel where noise is added and finally is fed to the Duffing block when detection is carried.

Carrier frequency of OOK digital signal is $f_c = \frac{50.000}{\pi}$, amplitude $a = 0.2$, noise power of 40 and bit rate $F_B = 100Hz$.

Signal after the channel is plotted in Figure C.3. It is easy to appreciate how signal is embedded in intense noise. This signal will be introduced directly into the Duffing.

In Figure C.4 there are two plots. In the first one, the phase diagram is shown. It can be highlighted how the external orbit gets slightly darker, it is due to the path followed in the periodic state. It is going to be the same all the time. The other random orbits are consequence of the chaotic state. Second plot in Figure C.4 shows in blue the output of the system and over that, in red color, it is a line which simulates the

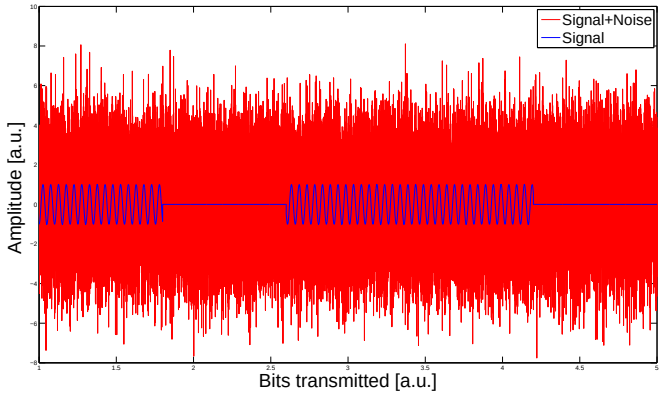


Figura C.3: OOK modulated signal. In blue color, the transmitted signal and in red color, the received signal. Parameters of the carried simulation are the carrier frequency at $f_c = \frac{50.000}{\pi}$, amplitude $a = 0.2$, noise power of 40 and bit rate $F_B = 100Hz$.

correspondent input value of the system at every moment. It can be clearly pointed when a binary '1' or a binary '0' is transmitted.

The method of observing the phase plane can not meet the practical engineering appliance, so we need to use a method to identify each code automatically. In the subsection D.3 we will discuss further all the implemented methods to transform the output of the Duffing into a sequence of logical ones and zeros.

Matlab code for 2ASK signal detection using Duffing oscillator can be found in Appendix G.1.

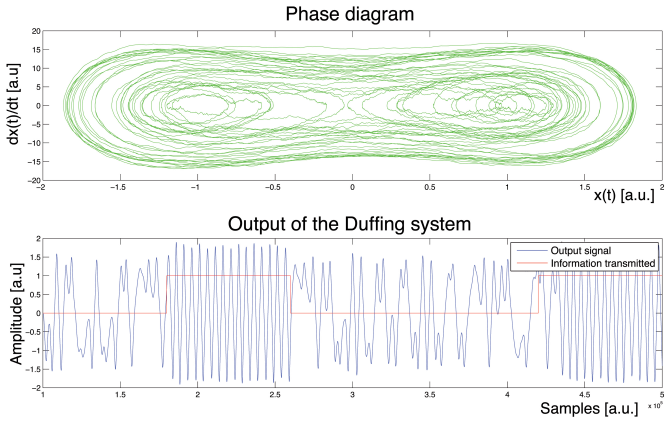


Figura C.4: Phase diagram and output of the Duffing for 2ASK detection. In the first one, the phase diagram is shown. The external orbit gets slightly darker due to the path followed in the periodic state. The other random orbits are consequence of the chaotic state. The second plot shows in blue color the output of the Duffing oscillator and over that, in red color, it is a line which simulates the transmitted binary information.

C.2 A method for BPSK signal detection using Duffing Oscillator

In the previous section, we took advantage of the importance of the driver amplitude combined with the amplitude of the signal. This characteristic allows us to detect the ASK signal. There was also another condition to fulfil, the phase of the signal should be in a specific regime, see Figure A.2. In case of any of those conditions were not satisfied, detection could not be carried out.

Now, we are working with PSK. It is a digital modulation that conveys data by changing the phase of a signal. For this scenario and with the previous configuration, to-be-detected signal would keep the Duffing oscillator in the periodic state all the time, no matter which symbol was being transmitted. Signal frequency would be the same for the whole transmission and only phase would change. Transition between chaos

and large scale periodic would not be possible. Due to this fact, we have to redefine the Duffing in order to reach both states making the system able to differentiate between logical '1' and logical '0'.

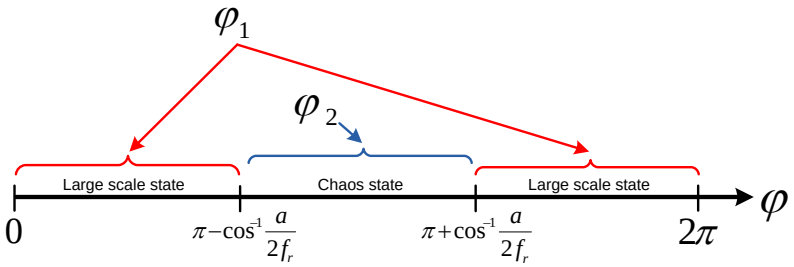


Figura C.5: Phase definition for BPSK. For the scenario in which amplitude requirements were fulfilled, if a signal with phase φ_1 feeds the Duffing oscillator, the system will keep in the large scale periodic state. By contrast, when the signal with phase φ_2 goes into the Duffing oscillator, the system will remain in the chaotic state.

The new idea lies in the smart definition of two different phases. One of these should be in the regimen that fulfil Duffing oscillator requirements (φ_1) and the other should be out (φ_2). When the signal with phase φ_1 feeds the Duffing oscillator, the system will reply as usual. It will keep in the large scale periodic state because phase and amplitude requirements are fulfilled. By contrast, when the signal with phase φ_2 goes into the Duffing oscillator, only the amplitude requirement will be satisfied and consequently, system will remain in the chaotic state. This concept is explained graphically in Figure C.5.

The general form for BPSK follows the equation: $s_b(t) = \sqrt{\frac{2E_b}{T_b}}\cos(2\pi f_c t + \pi(1-n)), n = 0, 1$. This yields two phases and for our case, binary data propagates with the following signals:

$$\phi_1(t) = \sqrt{\frac{2}{T_s}}\cos(2\pi f_c t + \varphi_1)$$
$$\phi_2(t) = \sqrt{\frac{2}{T_s}}\cos(2\pi f_c t + \varphi_2)$$

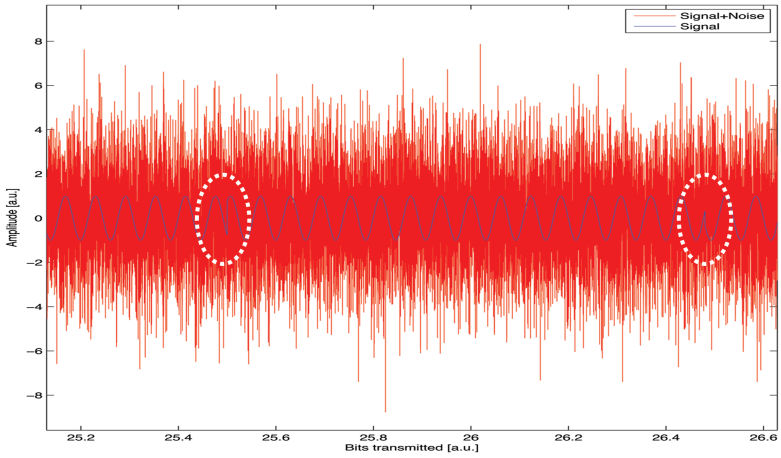


Figura C.6: BPSK modulated signal. In blue color is the transmitted signal where it can be observed the phase transitions. In red color, the received signal after the channel. Parameters of the carried simulation are the carrier frequency at $f_c = \frac{50.000}{\pi}$, amplitude $a = 0.2$, noise power of 40 and bit rate $F_B = 100Hz$.

For the simulation we have implemented a basic set-up similar to the one in Figure C.2, but in this case signal it is modulated with the equations above. Then, strong noise involves the transmitted signal.

Transmission parameters are also the same that in previous section: $f_c = \frac{50.000}{\pi}$, amplitude $a = 0.2$, noise power of 40 and bit rate $F_B = 100Hz$.

In Figure C.6, it is depicted the signal before going though the channel (in blue) and the same signal after the channel (in red). Phase changes in the signal are highlighted with a white circle around them.

In Figure C.7, it is plotted the phase diagram in the first image. In the second, it is shown the output of the system. In blue the output of the Duffing, in red transmitted logical 1's and logical 0's. Just below that, it is the transmitted signal in two different colors. Each one represents a different phase in the signal.

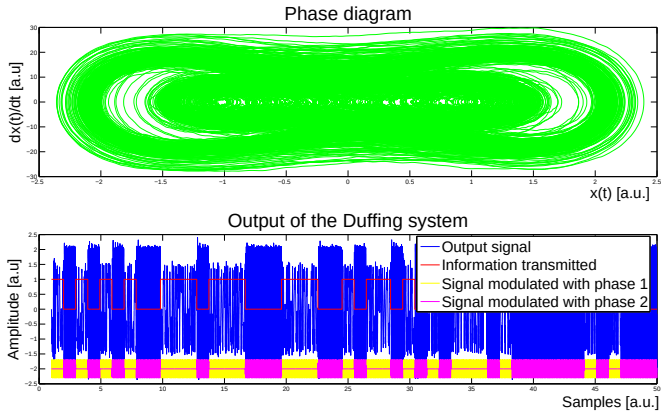


Figura C.7: Phase diagram and output of the Duffing for BPSK detection. In the first one, the phase diagram is shown. The second plot shows in blue color the output of the Duffing oscillator and over that, in red color, it is a line which simulates the transmitted binary information. Below these signals, it is the transmitted signal in two different colors. Each color represents a different phase in the signal.

By the Figure C.7, we can intuitionally see that the envelope waves of phase shift are smooth, hence we can say the system is in the periodic state; and if the envelope waves of phase shift have a great fluctuation, we can say that the system is in the chaotic state.

Matlab code for the BPSK signal detection using Duffing oscillator can be found in Appendix G.2.

C.3 A method for BFSK signal detection using Duffing Oscillator

Last but not least, the modulation to analyze is FSK. This technology has many merits such high power efficiency, convenient realization, reliable control and low cost. That is why is widely applied in digital communication fields [36].

In our experiment we will use a pair of discrete frequencies to transmit binary information (f_1 and f_2). We are going to fulfil all the specifications for f_1, so driver frequency will be really close and phase will be also fixed in the right regime (see Figure A.2). All this will make Duffing oscillator goes into the periodic state. The other frequency f_2 will not match with the reference signal, so periodic state could not be reached, keeping the Duffing oscillator in the chaotic state.

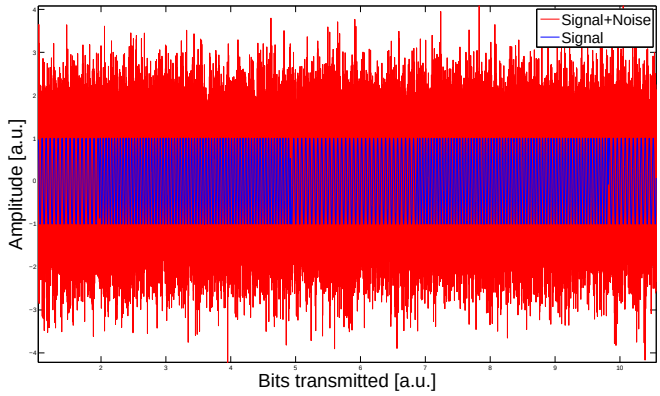


Figura C.8: BFSK modulated signal. In blue color is the transmitted signal where it can be observed the two different frequencies. In red color, the received signal after the channel. The two carrier frequencies are $f_1 = 59.000Hz$ and $f_2 = 60.000Hz$. Sampling frequency is $f_s = 24.000.000Hz$ and code rate 600 bits.

Also for this scenario, the values f_r and f_d are really determinant. As in the of OOK modulation, they have to be close to each other (see

Figure C.1).

For the simulation we have slightly followed [13] and we defined the same parameters. The two carrier frequencies are $f_1 = 59.000Hz$ (corresponding to 1) and $f_2 = 60.000Hz$ (corresponding to 0). Sampling frequency is $f_s = 24.000.000Hz$ and code rate 600 bits, having the next relation between frequencies $f_s/f_b = 40.000$.

Figure C.8 follows the structure than the previous plots when noisy signal was shown. We can see how the two different frequencies compose the transmitted signal. In the plot below (Figure C.9) is illustrated how Duffing behaves with this kind of modulation.

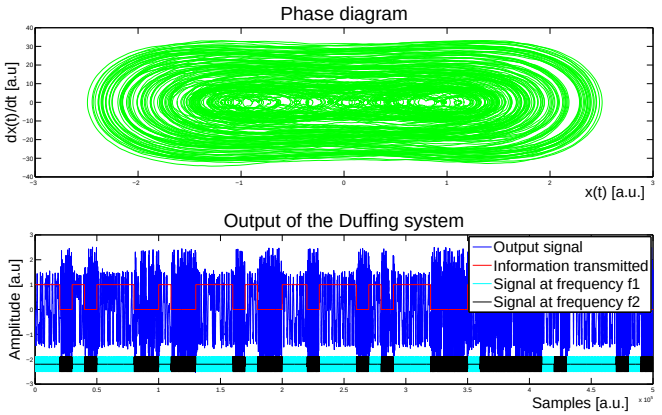


Figura C.9: Phase diagram and output of the Duffing for BFSK detection. In the first one, the phase diagram is shown. The second plot shows in blue color the output of the Duffing oscillator and over that, in red color, it is a line which simulates the transmitted binary information. Below these signals, it is the transmitted signal in two different colors. Each color represents a different frequency in the signal.

The method here explained is able to distinguish between two frequencies. However, there are some proposals to implement a system which is able to distinguish an unknown frequency signal, [25] and [29]. The idea is to design a Duffing oscillator array to scan the incoming signal. Whenever the Duffing oscillator reaches the intermittent chaos, the frequency of the

detected signal is close to the frequency of the system. This should be the basic theory to detect weak signals using Duffing oscillator arrays.

Matlab code for BFSK signal detection using Duffing oscillator can be found in Appendix G.3.

Apéndice D

Performance evaluation for ASK

The main goal in this chapter is to establish in which cases a chaotic Duffing receiving system can be applied, in which cases it is worse or better than the conventional methods. As a Duffing oscillator is one part of the demodulation set-up, we also test which option for subsequent symbol decision match better just after the Duffing oscillator in order to get the lowest BER possible. To accomplish this assessment, five different methods for symbol decision are implemented and tested in the following sections.

We developed two scripts. The first one is in charge of the transmitter implementation and noise loading, the other is in charge of the reception and subsequent demodulation. The two scripts are linked by a file. This file is created by the transmitter and it is similar to those obtained from a DSO. It contains its sampled output bits, which are fed as input for the receiver model. This configuration will become useful for future experimental applications where we will have only the file obtained from the DSO and our receiver.

D.1 Transmitter

The first block in the transmitter is the PRBS signal generator. It is a random sequence, that is, the value of the first elements is independent of the values of any of the other elements, similar to real random sequences. But it is 'pseudo' because it is deterministic and after N elements it starts to repeat itself. N in our simulations will be $2^{15} - 1$ or $2^7 - 1$.

PRBS signal generator block feeds the ASK modulator block. There,

PRBS signal is upsampled and filtered. Upsample is carried in order to increase the Número de muestras por símbolo (Nss) in the demodulation. Filter can be a raised-cosine, which is a filter frequently used for pulse-shaping [1] in digital modulation due to its ability to minimize Interferencia Intersimbólica (ISI).

Before transmission, baseband signal is modulated in to a carrier frequency; as shown in Figure D.1. Carrier frequencies used in our study are around 15 and $25GHz$.

The scheme for all this explained above can be seen in Figure D.1.

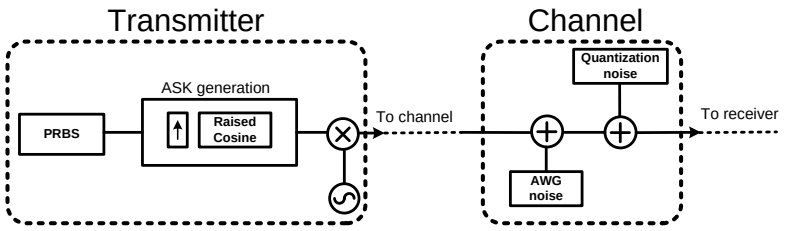


Figura D.1: Blocks diagram of transmitter and channel model.

D.2 Channel

The noisy channel is implemented in the same script as the transmitter but it takes up a different part in the communication. They can be found in Appendix H.1.

Two noises affects the channel:

[1] Pulse shaping is the process of changing the waveform of transmitted pulses. Its purpose is to make the transmitted signal better suited to the communication channel by limiting the effective bandwidth of the transmission. By filtering the transmitted pulses this way, the intersymbol interference caused by the channel can be kept in control. In RF communication, pulse shaping is essential for making the signal fit in its frequency band.

- **AWG noise** is the first of them. It is a random signal with a flat power spectral density and Gaussian amplitude distribution that will be added to the modulated signal [37].

  We are going to study all values within a range from 15 to $-5dB$ of Eb/No.

- **Quantization Noise** is a type of distortion that occurs when an analog waveform is encoded into a digital signal [38]. It will appear in the laboratory when the signal was acquired from the channel by the DSO. It has been included in the simulation in order to be the most accurate possible.

D.3 Receiver

The first stage in the receiver is a BPF. Its main function is to remove the out-of-band noise. It is a fifth order butterworth filter. This option was chosen among all other filters because it has the flattest passband meaning which is very good at simulating the passband of an ideal filter.

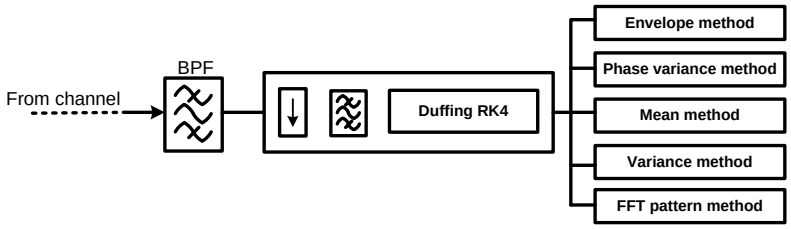


Figura D.2: Blocks diagram of receiver.

After this, all the chaotic Duffing receiving set-up is implemented. First, we use an interpolation block [39] made up by an upsample plus a BPF. The purpose of that is to reduce the stepsize, getting more samples per symbol. It will make Duffing oscillator output more accurate. This block could be dispensable. In a simulation environment, increasing the sampling frequency and fixing the bit rate should be enough; that would increase the number of samples per symbol. In a real environment it is not an easy task due to the limitation of the devices, the most we will

be able to obtain is about 100 samples per symbol. That is why we are using this block also in simulation, in order to keep it as real as possible.

Furthermore, we find the Duffing oscillator. For this scenario, we are going to use RK4 implementation (see subsection **??**) to delimit errors at maximum. All those blocks defined above are shown in Figure D.2 and implemented in Appendix H.2.

After Duffing, five methods will work in symbol detection with the output of the Duffing oscillator. The goal is to get the best estimation of the signal.

D.3.1 Envelope detection

The envelope detector is composed of a square block, LPF, downsampling , threshold block, symbol decision and BER calculation. This method has a first block which works by squaring the output of the Duffing oscillator and sending it through a LPF. Squaring the signal effectively demodulates the input by using itself as the carrier wave, keeping low varying envelope while filtering out high speed variation. The envelope can then be extracted by keeping all the Direct Current (DC) low-frequency energy and eliminating the high-frequency energy. LPF is a fifth order butterworth filter.

After the LPF, we calculate the position of the most significant sample for every symbol. Later on, the process of reducing the sampling rate of a signal is applied in the downsampling block. There, the position value will be used in order to select that sample as the representation of the whole symbol.

Since we have one sample per symbol, we have to define a threshold that will decide if the value of the signal correspond to a logical '0' or a logical '1'. We start calculating the histogram of the downsampled signal. There, we will observe two distributions with their correspondent peaks. Each distribution represents demodulated values for the case of either transmitted '1' or '0'. Using the implemented code, we can locate those two maximums. We know the threshold is in between. So, we search

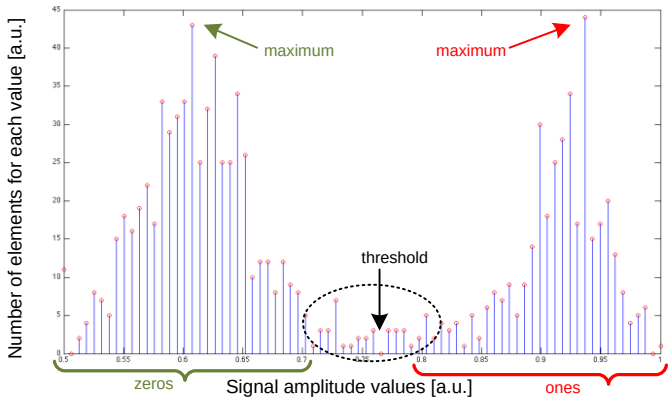


Figura D.3: Histogram for the threshold decision in envelope detection. Maximums represent the peak value of the distribution of logical 1's and the distribution of logical 0's. Threshold is defined as the minimum value between both distributions.

for the minimum close to the middle point between maximums and that will be defined as our threshold value. This procedure is illustrated in Figure D.3.

After the threshold, a decision block is implemented which decides if the input value should be '0' or '1' depending on its value compared to the threshold. Just after that, it is implemented another function which regenerates transmitted PRBS signal. It will be compared with the demodulated signal in order to determine the transmission BER. The scheme for this method is shown in Figure D.4.

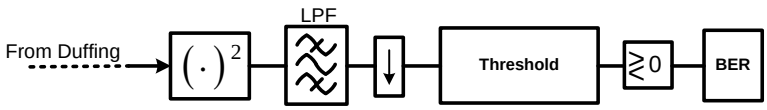


Figura D.4: Blocks diagram envelope detector.

Figure D.5 depicts an example of simulated results. Green color repre-

sents the output of RK4 Duffing oscillator. It feeds a LPF, whose output is in blue. Red signal is the PRBS transmitted signal, it is used to quickly evaluate how the demodulation goes.

This method is the least accurate of the five implemented. Its response is acceptable for high values of Eb/No, but around $9dB$ it starts having problems. Duffing transits quickly for sequences as '101010101...' ('1' and '0' alternatively) and it is not so well interpreted by the square block. The problem increases, when LPF is applied because not all the transitions are cached. Its behaves better when detecting a consecutive sequence of 1's or 0's (e.g. '1111000111000'). In that case we can increase the number of samples and therefore, its accuracy will be improved but the final performance will be worse than other methods commented below.

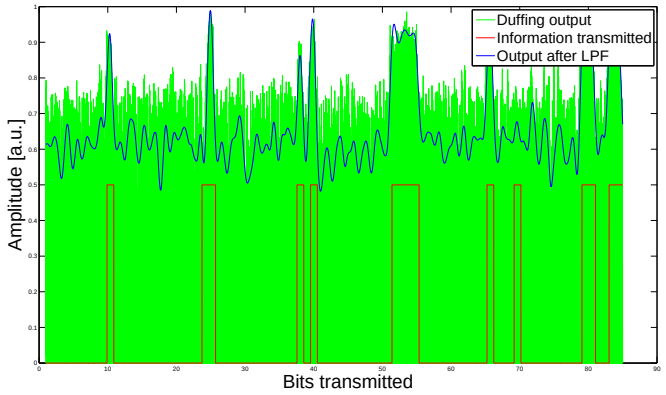


Figura D.5: Demodulation using envelope detection method. Green color represents the output of RK4 Duffing oscillator. In blue, it is the output of the LPF, whose input is the green signal. Red signal is the PRBS transmitted signal.

Matlab code can be found in Appendix H.3.

D.3.2 Phase variance

The phase variance is composed of a detector of phase changes block, accumulator, threshold block, symbol decision and BER calculation. As we mentioned in previous sections, large scale periodic state carries a phase change. Nothing is mentioned about what happens in the chaotic state. Focusing on those points, we have studied the phase change of the Duffing output and we can conclude that:

- In the large scale periodic state, there are constant and quickly phase changes.
- In the chaotic state, there are not as many phase changes as in the periodic state and they do not occur neither periodically.

In Figure D.6, we can see how the above explanation matches with the image. Pink signal shows the output of the detector of phase changes block where every vertical line means a change in the phase. It is easy to compare among the first group of transmitted logical 0's and the second one where logical 1's are transmitted. There is a considerable difference in the amount of lines between both parts.

Due to this property in the output signal of the Duffing oscillator, we have implemented an algorithm that detects the number of phase transitions for every transmitted symbol. All these numbers are stored in a vector which feeds an histogram. Threshold decision is then calculated based on that histogram. As in case of envelope detection (subsection D.3.1), samples are classified according to their values respect to the threshold. Finally, BER is quantified. Scheme in shown in Figure D.7.

An advantage with respect to the previous method, phase variance is independent of the incoming sequence, there is not any problem if the signal is '101010101...'. However, system does not increase so much the accuracy and the optimum performance is more complicated to achieve than in the methods below. This difficulty is because the phase of the reference signal has to be carefully defined in order to get the behavior observed in Figure D.6.

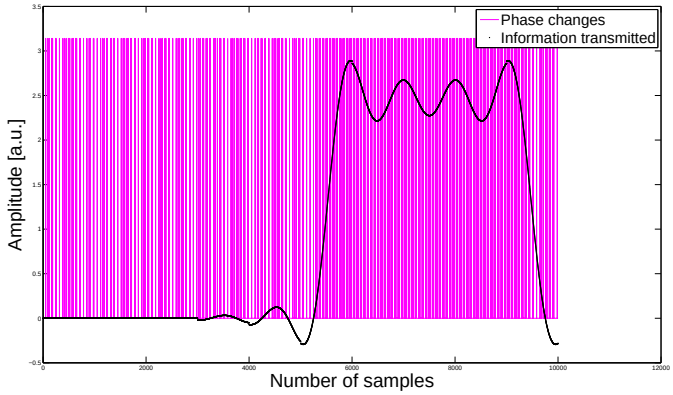


Figura D.6: Demodulation using phase change. The signal in pink color shows the output of the detector of phase changes block where every vertical line means a change in the phase. Back color signal in the transmitted signal before the noisy channel.

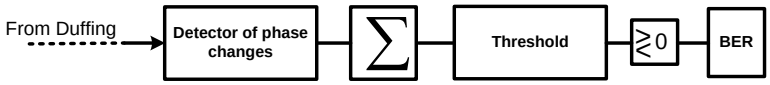


Figura D.7: Blocks diagram phase detector.

Matlab code can be found in Appendix H.4.

D.3.3 Mean method

While configuring the Duffing oscillator, another characteristic in the output was discovered. To achieve it, it is necessary to fulfil some requirements:

- Phase of the reference signal has to be defined close to the verge between transition and no transition but still in the transition state (see Figure A.2).

- f_r should be defined in the one period interior trajectory state, it means smaller than we have been setting in the previous methods.
- Reference signal has to be multiplied by a factor (e.g. 3) in order to skip the chaotic state and move directly from one period to large period state.

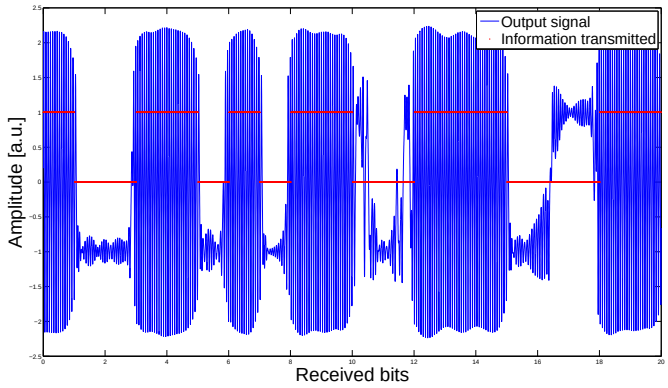


Figura D.8: Duffing oscillator response for the mean method in symbol decision. Signal in blue color is the output of the Duffing oscillator when all the requirements for the mean method are fulfilled. In red color is plotted the information signal.

The output is presented in Figure D.7, similar results were obtained in [40]. As indicated in the figure, logical 1's continue being the periodic signal with maximal amplitude but logic 0's are moved to the 1 period interior trajectory state, see Figure A.1(a). Due to this fact, amplitude of the output for that case is smaller than the amplitude achieved in all the previous method configurations.

So far, we split the output Duffing signal. Every part of that signal corresponds to the number of samples for one symbol. For calculations, we skip the first and last 5% of the samples for every symbol. Those removed samples were involved in the transition between systems and after being eliminated, data will become more stable. Now, the idea is to calculate the absolute value of the mean for every symbol. In the output,

we will have approximately null mean for the case of transmitted logical 1's. And for logical 0's, the arithmetic mean will be a large value.

After that, threshold decision is carried through the histogram and finally BER is calculated. All the set-up is shown in Figure D.9. The mean method is composed of a block where the mean of each bit is calculated, threshold block, symbol decision and BER calculation.

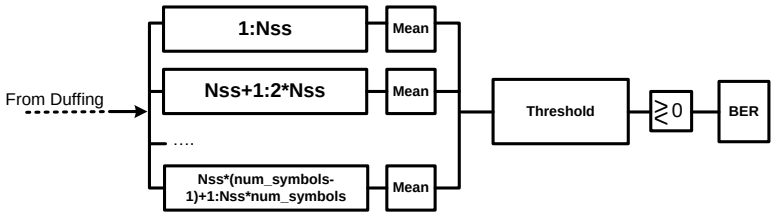


Figura D.9: Blocks diagram mean method.

The main advantage of this method is its simplicity. Duffing oscillator response has an acceptable accuracy compared with the current methods. The worst point is the difficulty to fulfil all the requirements above in order to have similar outputs with Figure D.7.

Matlab code can be found in Appendix H.5.

D.3.4 Variance method

We would like to extent the previous method of symbol decision for all kind of Duffing oscillator outputs. Signals with small amplitude in the chaotic state, as the one shown in Figure C.4, are more common in detection than the conventional Duffing oscillator output, e.g. Figure D.8. In this new proposal, we will continue splitting the Duffing oscillator output by symbols. It will be eliminated about 7% of the first and last samples of every symbol to assure stability. However, in this case we will apply the variance to every symbol instead of the mean. Variance will be used as a measurement to quantify how far a set of samples are spread out from each other.

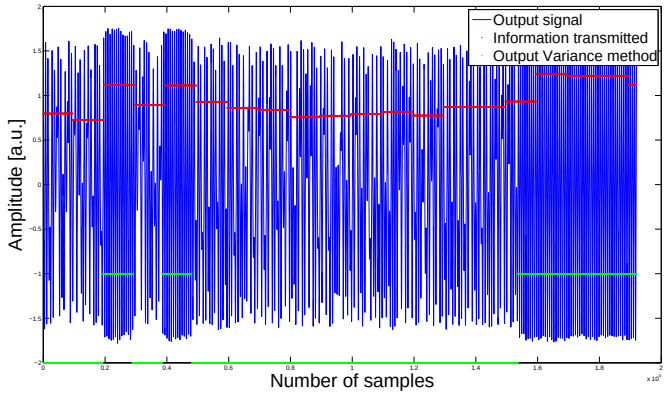


Figura D.10: Demodulation using variance method. The signal in blue represents the output of our Duffing oscillator. The signal in green is the transmitted PRBS signal and the signal in red color shows the variance value for every symbol.

Figure D.10 shows in blue color the output of the Duffing oscillator and how it is the variance for each case. When the Duffing oscillator is in the large periodic state, variance value is higher than it is in the chaotic state. Signal in blue is, as usual, the output of our Duffing system. Green signal is the transmitted PRBS signal. It represents transmitted logical 1's or 0's. The signal we will use for symbol decision is the one in red color, it shows the variance value for every symbol.

As we can see in the Figure D.10, results for ones and zeros do not have a complicated fixed value decision process. It is worthy to say that in this case we do not feed the threshold decision block with those values and we do not compare them with the threshold. The most interesting issue is to study the relation of one value with the previous one. In such case, we can really appreciate differences among the two kinds of transmitted symbols. Thus, we create a new vector which stores the difference between values, related to that it will be easier to establish a threshold value.

This method is the most accurate until now, it takes long time to process

a signal in comparison with those methods before but it is an accurate method to work with the output of the Duffing oscillator.

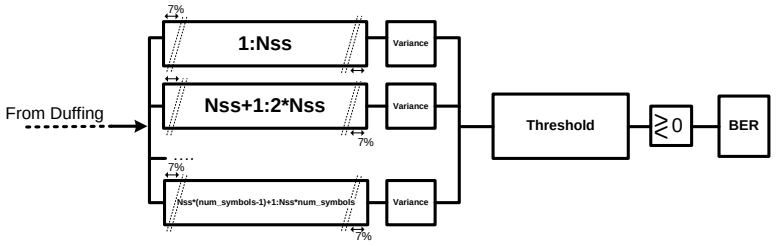


Figura D.11: Blocks diagram variance method.

The set-up for this method is depicted in Figure D.11. The variance method is composed of a block where the variance of each bit is calculated, threshold block, symbol decision and BER calculation. The main difference between the plot in Figure D.9 and the current set-up, is the replacement of the mean block for a different one where the variance is calculated. By contrast, Matlab code differs substantively and it is attached at the Appendix H.6.

D.3.5 FFT pattern method

By now, the four developed and tested methods explained above, have worked with the properties of the Duffing oscillator output in the time domain. Now, we are going to characterize the signal in the frequency domain. As we know, signal in the large periodic state has some kind of periodicity and this fact should be reflected on its FFT. Chaos state has to be characterized at some point, perhaps signal is composed by the mixture of different frequencies or it has a constant spectrum without any frequency standing out.

To clear up these question, we split the Duffing output signal in symbol and we apply FFT to every part. FFT of transmitted logical 1's and 0's are shown in Figure D.12 and Figure D.13. From there, we are going to extract some conclusions.

Evaluating Figure D.12, it can be seen how a maximum peak appears. From that frequency to baseband there is no any other peak in the spectrum. In the other direction, the only significative frequencies are the integer multiples of the signal, called harmonics.

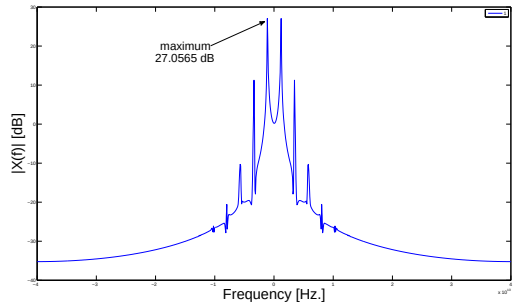


Figura D.12: Spectrum of transmitted logical 1.

In the case of Figure D.13, signal has more frequencies than Figure D.12. We can find a maximum in baseband. The second maximum is located at the same frequency at which the transmitted logical '1' had its peak. So, this frequency is also strong in the chaos signal but lower than for the first case. There are also a lot of frequencies, between the two maximums, present in the spectrum with a significative amplitude.

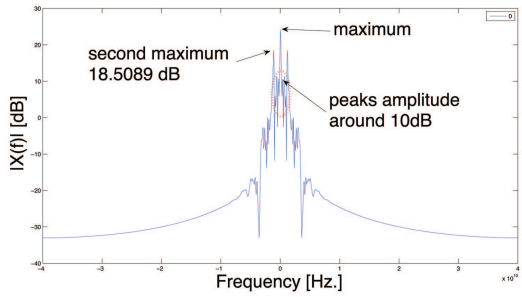


Figura D.13: Spectrum of transmitted logical 0.

What we are going to implement is an algorithm which studies the frequency where logical '1's have a maximum (reference peak) and stores

that value in a vector. From that frequency, we will scan all the frequencies down to baseband. The goal is to find a second frequency peak or pattern which proves the existence of more strong frequencies in that range (we will name the new peak as sub-maximum). Amplitude value of these sub-maximums will be also stored in another vector.

Theoretically, it should be enough defining only one maximum, one threshold and studying only one point per symbol to decide between '1' or '0'. But experimentally, we have noticed a high improvement in the detection accuracy when two different reference points per symbol are taken. Therefore two conditions, two thresholds, have to be analyzed instead of only one.

So far, two histograms should be created from the two vectors. One with the stored maximum and the other with the stored sub-maximum peaks. Subsequently threshold values for each case can be calculated. Once we have the two thresholds, one for maximums and other for sub-maximums, we implement a decision block.

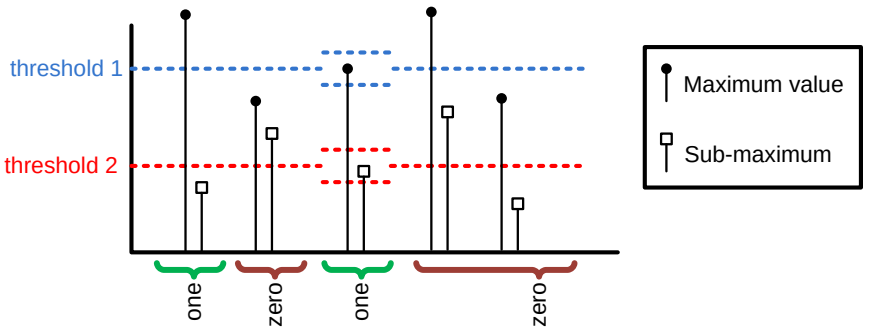


Figura D.14: Threshold system for symbol decision. Symbol is interpreted as logical '1' if its maximum is around the *threshold 1* and its sub-maximum around the *threshold 2*. Symbol is interpreted as logical '0' in any other case.

How decision is taken is illustrated in Figure D.14. This block consists in four steps:

1. In the first one, symbols which its maximum exceed the first thresh-

old and at the same time, its sub-maximum is below the second threshold are considered as logical '1'. This could be interpret as a signal with only one strong frequency component and nothing else around. Figure D.12 has a similar behavior.

2. In the case that first condition is not fulfilled. They will be consider as logical '0', symbols with maximum smaller than the first threshold and sub-maximum larger than the second threshold. That is interpreted as symbols which does not have an specified strong periodicity signal but still have important signals in other frequencies. As happens in Figure D.13.

3. We have already classified most evident symbols. For the third case we analyze those in doubt. Their values do not fulfil any of the conditions above but are close to do it. Those symbols which are located in a range around the $\pm 11\%$ of each threshold value, will be considered as logical '1'. It includes those symbols which only fulfilled one of the thresholds and were close to the other. This case corrects errors produced due to a non-optimal definition of threshold values.

4. In the last case, we take as logical '0' any other value that were not included in the previous conditions: when sub-maximum is extremely high being the maximum larger than its value, when maximum is not large enough but sub-maximum is correct and so on.

After the decision block, signal is compared with transmitted message and BER is calculated. All the set-up can be seen in Figure D.15. The FFT pattern method is composed of a block which calculates the FFT, two different threshold blocks, symbol decision for each threshold and BER calculation.

After numerous simulations, this FFT pattern method is the one selected among methods (coherent and incoherent detection) due to its accurately. Variance method and FFT pattern method have a similar performance until $4dB$. From this value to $15dB$, FFT pattern has achieved in a better performance of around $0.7dB$.

Matlab code of the current method can be found in Appendix H.7.

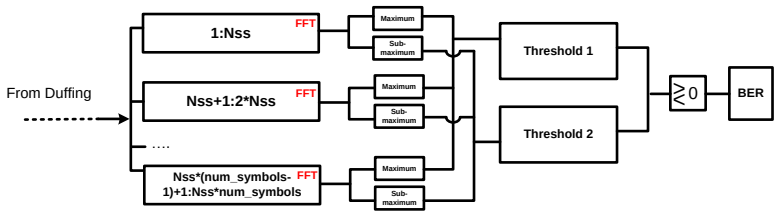


Figura D.15: Blocks diagram FFT pattern method.

D.4 Comparative and results: Duffing vs. Coherent detection

In order to be able to compare our demodulator with the coherent and envelope detection methods, we are going to implement the coherent and the incoherent demodulators. They have to match with the transmitter code (Appendix H.1), so there will be two extra blocks in our code.

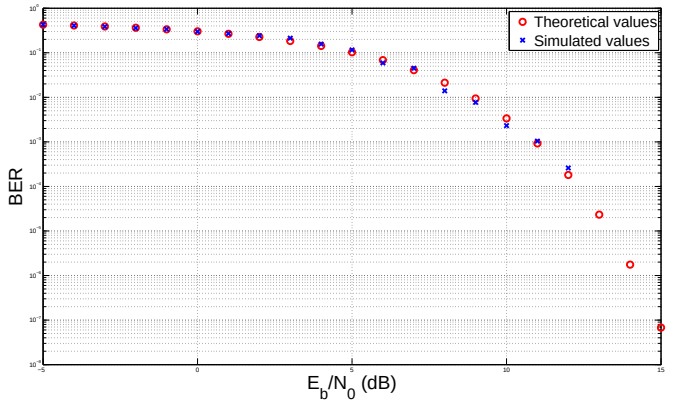


Figura D.16: Envelope detection. Comparative between theoretical results and our simulation results.

Envelope detector, as incoherent method, will follow a similar scheme such at that shown in Figure D.4. The only different is from where the signal comes, this time it will come directly from the channel, instead of

coming from the Duffing output.

To guarantee a properly implementated method, we compare simulated BER in our system with theoretical curves. Comparison is plotted in Figure D.16 and it demonstrates how similar results are obtained. It is also important to realize than simulation only goes until $12dB$ of Eb/No, after that value there are no errors. It is due to the high number of simulated bits needed to find an error in the receiver. For example, in case of $15dB$ of Eb/No, it would be needed to process 10 million of bits and computer resources are limited for such a case. Matlab code for the whole communication can be found in Appendix H.8.

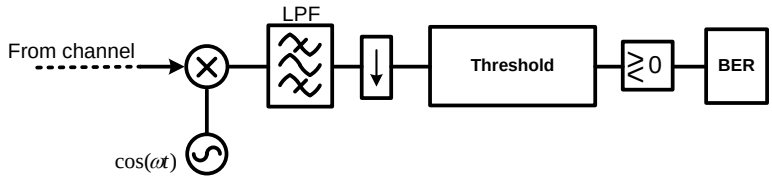


Figura D.17: Set-up of coherent demodulation.

It is also implemented a coherent demodulator [41]. It is composed of mixer, a LPF, a downsampling block, a threshold block and BER calculation. The set-up is shown in Figure D.17. In this case we multiply signal from the channel by a sine signal at the same frequency. Phases of both signals have to be matched as well. Output is low pass filtered.

We have also implemented a script to compare simulated and theoretical results. It happens the same as in the envelope detection where simulation is limited by the number of simulated symbols, as we can see in Figure D.18.

Code implemented for all this above can be located in Appendix H.9.

After coherent and incoherent models have been implemented, we are going to compare their performance with our chaotic Duffing receiving system. Communication will be configured in the next way:

- Frequency sampling: $80GHz$.

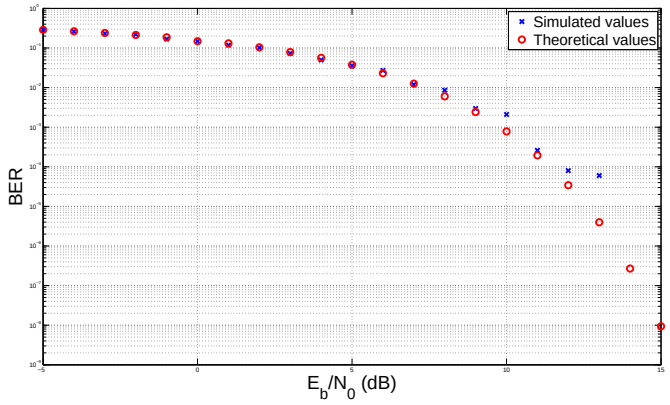


Figura D.18: Coherence detection. Comparative between theoretical results and our simulation results.

- Carrier frequency: $13.8GHz$.
- Bit rate: $1Gbps$.

As we said in section D.3, we try to get a high number of samples per symbol. That will be advantageous for the Duffing oscillator in the way the more samples, the more immunity against noise will be acquired. It is not the only reason, also reference papers define this relation between carrier and sampling frequency as large as we have done, [12] and [15].

In Figure D.19, it is depicted the result of the first comparison. Filters in the reception are implemented as ideal filters, their bandwidth is set as the signal width.

Coherent detection results in the most effective method. Duffing oscillator response is just located in the middle of both classical methods. For this first test, we do not observe any advantage in our system. Chaotic Duffing receiving system has a better performance in terms of Eb/No than envelope detection. However, this comparison is not carried at the same conditions. The Duffing oscillator and the coherent method are using a reference signal in the same frequency range as the transmitted. For

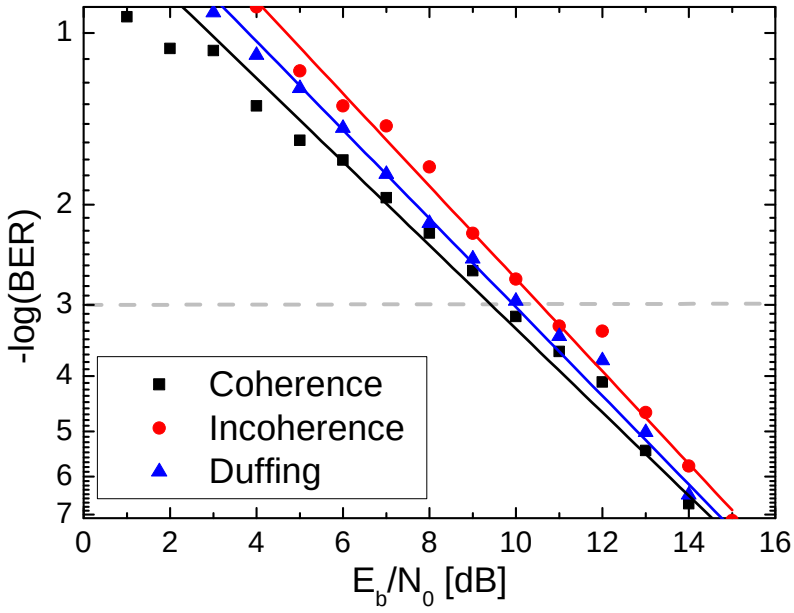


Figura D.19: Comparison between coherence detection method, incoherence detection method and chaotic Duffing receiving system. BPF set at $1GHz$, carrier frequency of $13.8GHz$, sampling frequency of $80GHz$ and $1Gbps$ of bit rate.

this reason, both are more complicated to implement than the incoherent method and thus, they should achieve better results.

In the next simulation, we increase the bandwidth of the BPF in the reception until to $2GHz$ FWHM. This wide of the filter could be closer to a real environment where filters are wider than theoretically needed. The idea to proceed this way came from the fact that Duffing oscillator has been used with high oversampling rates and no filter in reception while detecting weak signals [15], as we mentioned in the introduction of this Thesis. For that reason, we are going to study along some simulations its behavior for different bandwidths of the receiver filter. New bandwidth will be two times the signal bandwidth, $2GHz$ FWHM.

Results are depicted in Figure D.20. For this situation we skip envelope

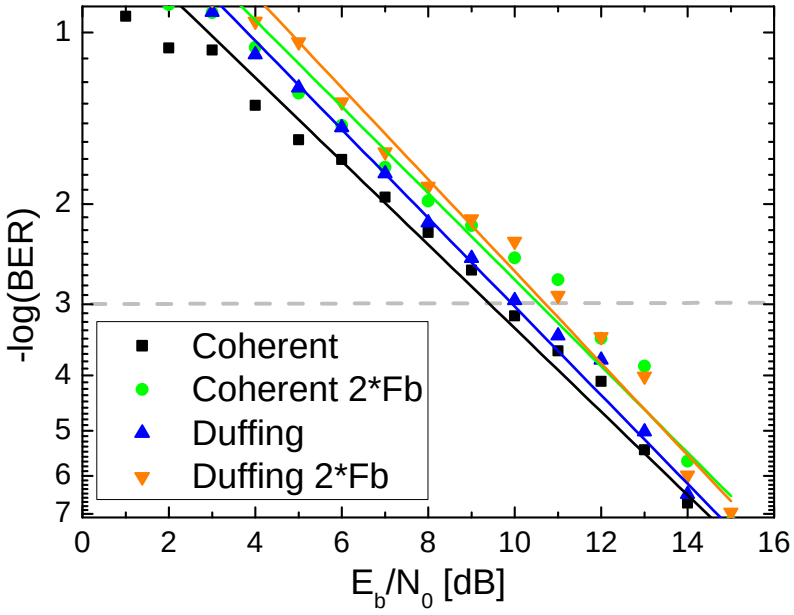


Figura D.20: Comparison between coherence detection method, incoherence detection method and chaotic Duffing receiving system. BPF set at $2GHz$ FWHM, carrier frequency of $13.8GHz$, sampling frequency of $80GHz$ and $1Gbps$ of bit rate.

detection and compare directly with the coherent method. Its performance continues being still better for this case compared with the chaotic Duffing receiving system in terms of Eb/No. However, it can be seen how the worsening for this new filter width is larger for the coherent method than for the case of the Duffing oscillator. A better performance of chaotic Duffing receiving system in the case of BPF with $2GHz$ FWHM suggests us to continue analyzing what would happen if the BPF was even larger.

To solve this question we increase the bandwidth of the filter in the receiver. In Figure D.21, it is shown the result of increasing the FWHM of the BPF 8 times the signal bandwidth. In this new scenario, Duffing has better performance than the coherence method in terms of Eb/No but the performance is not enough to achieve 10^{-3} of BER with $16dB$ of Eb/No.

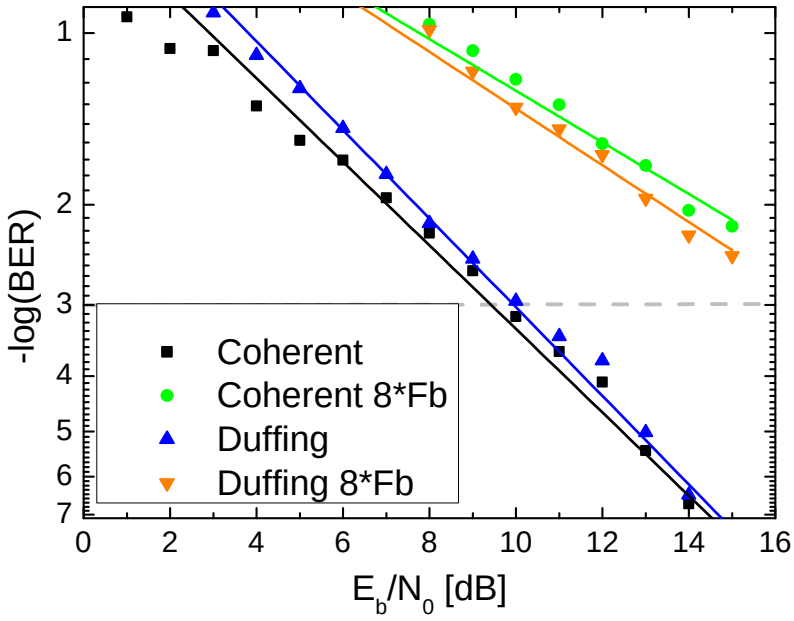


Figura D.21: Comparison between coherence detection method, incoherence detection method and chaotic Duffing receiving system. BPF set at $8GHz$, carrier frequency of $13.8GHz$, sampling frequency of $80GHz$ and $1Gbps$ of bit rate.

To sum up, we have seen how chaotic Duffing receiving system becomes more effective in terms of Eb/No than coherent detector as filters in reception get worse. Duffing system behaves like a BPF. This could be used as an advantage for scenarios where radio frequency is not known accurately or in case PLL could not be implemented.

D.5 Alternative Duffing receiver implementation

In the previous section we proved that in some cases chaotic Duffing receiving system achieved better performance in terms of Eb/No than coherent detector. However, this performance was not good enough due to low BER. So, any advantage could be taken from there, but we draw

one conclusion: Duffing has better response than current systems when filter in reception becomes poor.

We also saw how chaotic Duffing receiving system had better performance in terms of Eb/No whatever the width of the filters was for incoherent detection. That was not a so impressive piece of novelty because Duffing oscillator was fed by a reference signal and the envelope detector did not have any similar help. The next question would be, what will happen if we get the reference signal out in the Duffing oscillator implementation?

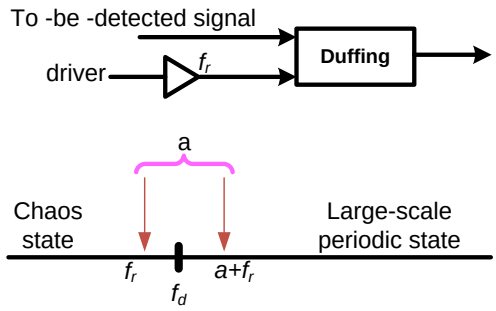


Figura D.22: Classical Duffing oscillator implementation. It is fed by the driver signal and the to-be-detected signal. Distance from chaos state to large periodic state is small.

In order to solve the previous question, we redesign the chaotic Duffing receiving system. Previously there were two inputs, the information signal from the channel and the reference signal or driver, see Figure D.22. Driver was fixed at the same frequency as the carrier signal. Now, we remove this entrance (Figure D.23) and all the responsibility for controlling the system is transferred to the input signal. It would take care of changes from one state to the other by using its amplitude.

Having a look at Figure D.22 again, we can see the influence of every parameter in the state transition. We have a pre-defined value of f_r which locates Duffing oscillator in the verge of bifurcation. Then, amplitude of the to-be-detected signal will induce the Duffing oscillator to large scale periodic state or it will not induce to that state, depending of which symbol is transmitted. Thus, this 'a' amplitude value will be ten times

smaller than f_r but extremely determinant at the same time.

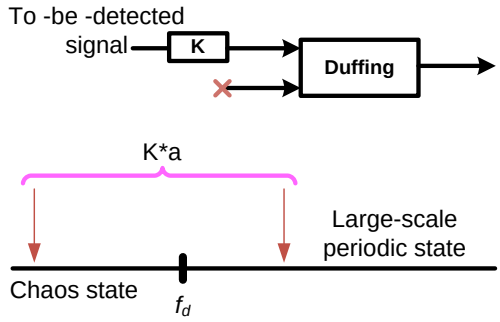


Figura D.23: Modification of the conventional Duffing oscillator implementation. The system is fed by the to-be-detected signal. The driver signal is not needed. Distance from chaos state to large periodic state is larger than in the classical implementation of the Duffing oscillator.

Once the branch of the driver is eliminated, the next idea will be implemented: to insert a new gain block just before feeding the Duffing (block referred as **K** in Figure D.23). Its function will be to amplify the to-be-detected signal. There are again two cases of study:

- When a binary zero is transmitted, there will not be any strong signal amplitude. We could find interferences from previous symbols and noise from the channel. All those will be amplified but in any case, large-scale periodic state will be reached.
- The other situation occurs when a binary one is transmitted. Signal is not so strong by itself, but once it is amplified, it will go beyond f_d and the output of our system will be periodic.

D.6 Comparative and results: Duffing vs. Envelope detection

Before analyzing the behavior of the current system and the envelope detection, we are going to compare the chaotic Duffing receiving sys-

tem with and without reference signal. The result of that comparison is depicted in Figure D.24 and it can be observed how this new implementation performs $0.5dB$ worse in terms of Eb/No than the model used in the previous section with driver signal, as we would have expected.

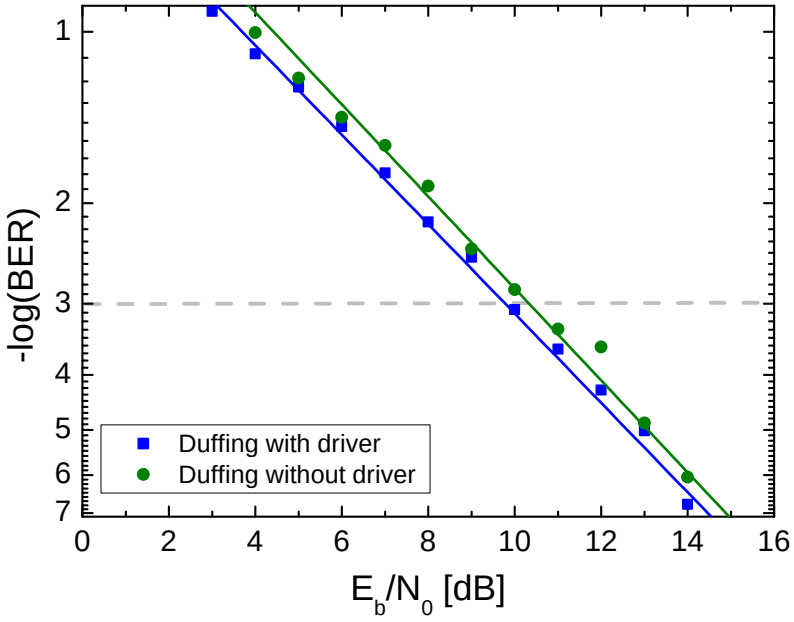


Figura D.24: Comparison between chaotic Duffing receiving system fed with a reference signal and the modification of the conventional Duffing oscillator implementation. BPF set at $1GHz$, carrier frequency of $13.8GHz$, sampling frequency of $80GHz$ and $1Gbps$ of bit rate.

The next step is the comparison between chaotic Duffing receiving system and envelope detection receiver. First, in Figure D.25(a) we can see the results of comparing the new design of the Duffing oscillator with the envelope detection for the case of an ideal filter, the FWHM of the BPF and the signal have the same bandwidth. It is observed how there is a negligible difference between both methods, being the values pretty similar until $10dB$ of Eb/No and since then, chaotic Duffing receiving system performs slightly better in terms of Eb/No.

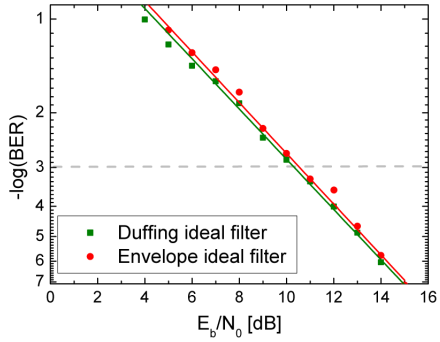
The BPF in reception is set wider for this simulation, we are going to

define the FWHM of the BPF two times the signal bandwidth, $2GHz$. In Figure D.25(b) there are depicted three lines. Two correspond to this scenario with a wider reception filter and the third line will be used as a reference. This line is in dark green color and represents the behavior of the Duffing system for the case studied before when the BPF and the signal bandwidth matched. The output of the Duffing has a penalty of $0.5dB$ compared with the ideal scenario and the envelope detection has a penalty of about $1.3dB$ over the same reference. All these penalties are measured at the value of $10^{-3}dB$ of BER (grey dashed line) which could be seen as the value free of error if we were using Forward Error Correction (FEC) methods. For this case, the Duffing receiving system fed with a reference signal and the modification of the conventional Duffing oscillator implementation starts to become better than the envelope detection method.

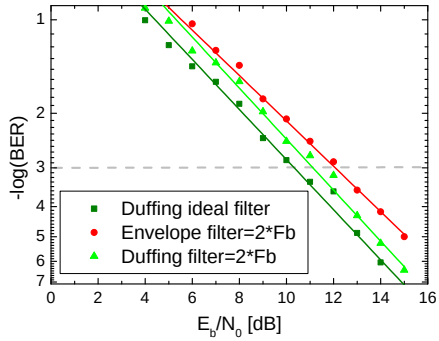
In the next scenario we define the FWHM of the BPF, three times the signal bandwidth, $3GHz$. The result is shown in Figure D.25(c). The Duffing line has a penalty about $1dB$ over the reference signal. The envelope detection has $2dB$ of penalty. This advantageous position is kept until $5dB$ of Eb/No. For example, this can be seen as a tolerance of $3GHz$ of uncertainty in the RF frequency.

The last simulation in this scenario is done with the goal to test the modification of the conventional Duffing oscillator implementation and the envelope detection method in a extreme situation. We suppose there is not any filter at reception. Signal is limited by the frequency sample, the equivalent bandwidth will be $80GHz$. Output is plot in Figure D.25(d). In this case we get the maximum advantage of implementing a chaotic Duffing receiving system. Gain remains constant for the whole simulation and it is about $1.5dB$ for the whole range. The Duffing ideal filter line has a penalty about $3.5dB$ over the reference signal and the envelope detection has about $5dB$ of penalty.

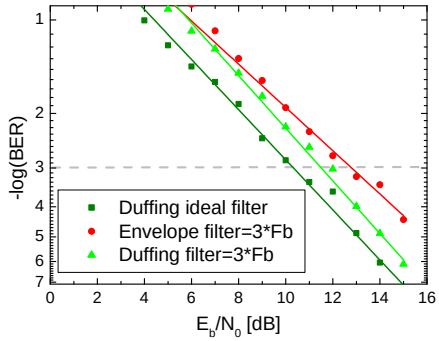
As a conclusion, we have seen how this new design of Duffing performs better in terms of Eb/No than the envelope detection method. If we increase the width of the filter , we can observe that envelope is a worse solution. Every time we increase the FWHM of the BPF, differences in the performance between Duffing and envelope output increase.



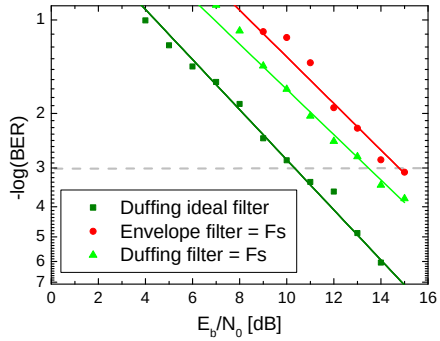
(a) Comparison for an ideal filter. BPF set at $1GHz$.



(b) Comparison with filter = 2*Fb. BPF set at $2GHz$.



(c) Comparison with filter = 3*Fb. BPF set at $3GHz$.



(d) Comparison when there is no reception system. Equivalent BPF $80GHz$.

Figura D.25: Comparison between chaotic Duffing receiving system and envelope detection for different filter widths. Carrier frequency is set at $13.8GHz$, sampling frequency at $80GHz$ and $1Gbps$ of bit rate.

Apéndice E

Experimental results

Experimental set-up and results will be presented in this chapter to verify whether methods described previously can be used to obtain improved results.

In the first section, there is a first experiment with the aim to characterize electrically our Duffing system for future experiments. In the second section we analyze experimentally our simulation results. For that, we implement a set-up in the laboratory and we process offline the captured data. Demodulation will be carried with the new design of Duffing system.

E.1 Electrical characterization of the system

This simulation has the purpose to get a first approximation about chaotic Duffing receiving system performance with real data, test and decide which PRBS signal would be more convenient or which output power will influence the system positively. That is why the implemented set-up is really simple, as we can see in Figure E.1. It is composed by a VSA directly connected to a DSO. There is not any external noise source added to the system, the very little amount of noise comes from the output of the VSA.

As we said, there is not any interest in a correct demodulation at this point, we will focus on the phase diagram of our system and how difficult distinguishing between logical 0's and logical 1's from the output signal is.

Configuration parameters in the set-up are defined as follows:

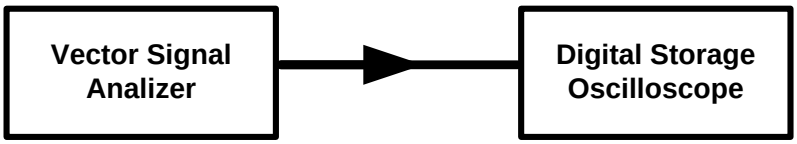


Figura E.1: Set-up for the electrical characterization of the system.

- Bit rate of the signal will be fixed as $400Msps$ for the entire experiment.
- Sampling frequency will be defined either $20GHz$ or $40GHz$. It will lead in 500 or 1000 samples per symbol, respectively. The idea continue being to define a system as simple as possible and with this sampling rate, we would not need any upsampling block before Duffing.
- Carrier frequency is set at $600MHz$.
- Different PRBS signals will be tested. PRBS pattern: $2^n - 1$ (n : $9, 11$ and 15).
- Different VSA output power will be also tried (0 dBm, -20 dBm, -30 dBm and -40 dBm).

After that, there are also some variables in the script that will remain fixed for the entire experiment. To assure phase transition, the phase of the driver will be defined as $\varphi = 0$. As in previous scripts, damping ratio will be set at 0.5 and the limit value between chaos and periodicity will be $f_d = 0.829$, so depending on the input power we will define the force amplitude f_r in the range $(0.7852, 0.8137)$.

In Figure E.2 there are plotted 20 symbols of the signal acquired by the DSO after being filtered by a fourth order butterworth BPF. It becomes apparent the clarity of the signal.

Signal from Figure E.2 is then introduced into the Duffing oscillator. Output can be seen at Figure E.3(b) and phase diagram at Figure E.3(a). Signals with different powers have been tested and no problems turned up.

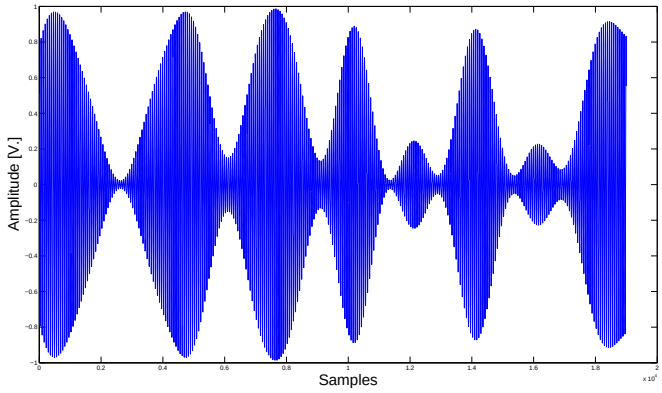
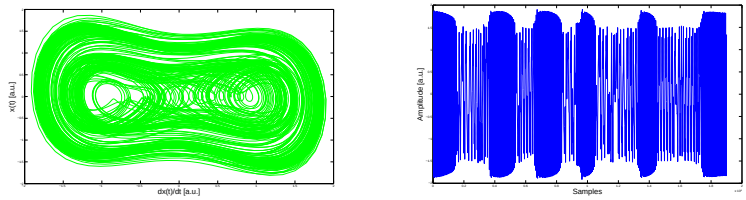


Figura E.2: Band-pass filtered signal from the output of Digital Storage Oscilloscope. Bit rate of $400Msps$, sampling frequency $20GHz$, carrier frequency $600MHz$ and Vector Signal Analizer output power around $-30dBm$.



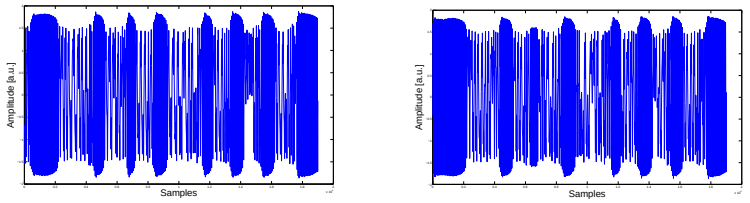
(a) Phase diagram (b) Duffing output

Figura E.3: Outputs of the Duffing for 30 dBm of signal power, 40 GHz of sampling frequency, 40 Msps of bit rate and force driver $f_r = 0.8085$.

For cases where frequency sampling was $20GHz$ and $40GHz$, it can be easily noticed when a binary zero or a binary one was transmitted in both cases. From previous simulations we discovered a better behavior in Duffing response when the number of samples per symbol was above 400. But testing two different values that would not present any problem (500 and 1000 samples per symbol), we wanted to quantize if there was any significative difference between those two different values or on the

contrary, accuracy was guaranteed above 400 samples in any case. And we conclude that the second affirmation was fulfilled and above a certain number of samples, there is not any significative improvement.

Another conclusion for this first experiment is that Duffing setting is required for any variation of power in the incoming signal. The value f_r can be calculated automatically in the code and it will be valid, but it is always advisable to test nearby values in order to specify the best one. It also means for future experiment that different SNR would need a different configuration of the parameter f_r.



(a) $\varphi = 0.007 * \pi$ (b) $\varphi = 0.012 * \pi$

Figura E.4: Outputs of the Duffing for 40 dBm of signal power, 40 GHz of sampling frequency, 40 Msps of bit rate and force driver $f_r = 0.7816$.

Same problem as above happens to drive phase φ. It is also connected to f_r, as we can see in A.3.1. If f_r is modified, φ should be also adjusted. In the case of not doing it we could incurrent in a loss of power for transmitted ones and they could be interpreted as zeros. This lost of power (lost of periodicity) is shown in Figure E.4(b) comparing with Figure E.4(a).

In the previous chapter it was mentioned a problem in the detection of signal when it transits quickly and continuously from binary one to binary zero or viceversa. We have used different PRBS signals in order to check this fact. When $n = 15$ is supposed than there will be more consecutive ones or zeros than for the case of $n = 7$, obtaining less transitions. For all those cases, Duffing reacts in the same way and there is not any PRBS signal which produces a better performance in our system.

E.2 Chaotic Duffing receiving system implementation for Radio-over-Fiber transmission

This second experiment is much more complex than our first attempt. It has been built in order to simulate an entire optical communication where its output will be demodulated and offline processed.

E.2.1 Set-up implementation

The detailed schematic showing the experimental setup used is presented in Figure E.5. The transmitter is a Teraxion DFB Narrow Linewidth Laser with a linewidth of less than $50kHz$. Electrical PRBS signal at different rates are generated with Agilent HP 70843A PPG. Signals are subsequently amplified to feed the MZM. The driving voltage of the MZM has been adjusted to obtain the highest extension ratio, $17dB$.

The output signal is transmitted through the fiber. Noise is added to the received signal. EDFA as a white noise source and filtered with $0.8nm$ bandwidth optical filter with which we will keep only in-band noise, whereas out-of-band noise will be removed. Signal is amplified again and connected to a new filter having $0.5nm$ bandwidth.

The output of the filter is attached to a VOA in order to keep the power of the optical signal in a constant level. LO laser is ECL with a linewidth less than $300kHz$. Signal is measured with a PM at the output, being LO power $-3.2dBm$ and signal power $-0.9dBm$. Finally, signal is fed to a PD. Two different PDs were used in the experiment. One for low frequency with a $10GHz$ bandwidth and internal amplifier and a second one for high frequency, $40GHz$ and external electrical amplifier. The output of the PD are attached to two LeCroy WaveMaster 830Zi-A DSO capable of capturing 80GSa/s with a $30GHz$ bandwidth input.

The measurements gathered with the previous set-up are:

- $1Gbps$ of bit rate modulated at $13.8GHz$ for OSNR among 18 and

$1dB$ in the cases of having or not a high linewidth. Sampling frequency at $80GHz$.

- $1Gbps$ of bit rate modulated at $26.4GHz$ for OSNR among 18 and $1dB$. Sampling frequency at $80GHz$.
- $2Gbps$ of bit rate modulated at $26.4GHz$ for OSNR among 18 and $1dB$. Sampling frequency at $80GHz$.
- $5Gbps$ of bit rate modulated at $26.4GHz$ for OSNR among 18 and $1dB$. Sampling frequency at $80GHz$.
- $10Gbps$ of bit rate modulated at $24.8GHz$ for OSNR among 18 and $1dB$. Sampling frequency at $80GHz$.

Captured data are later on transferred to a computer and offline processed on Matlab. The results obtained are outlined in the following sections.

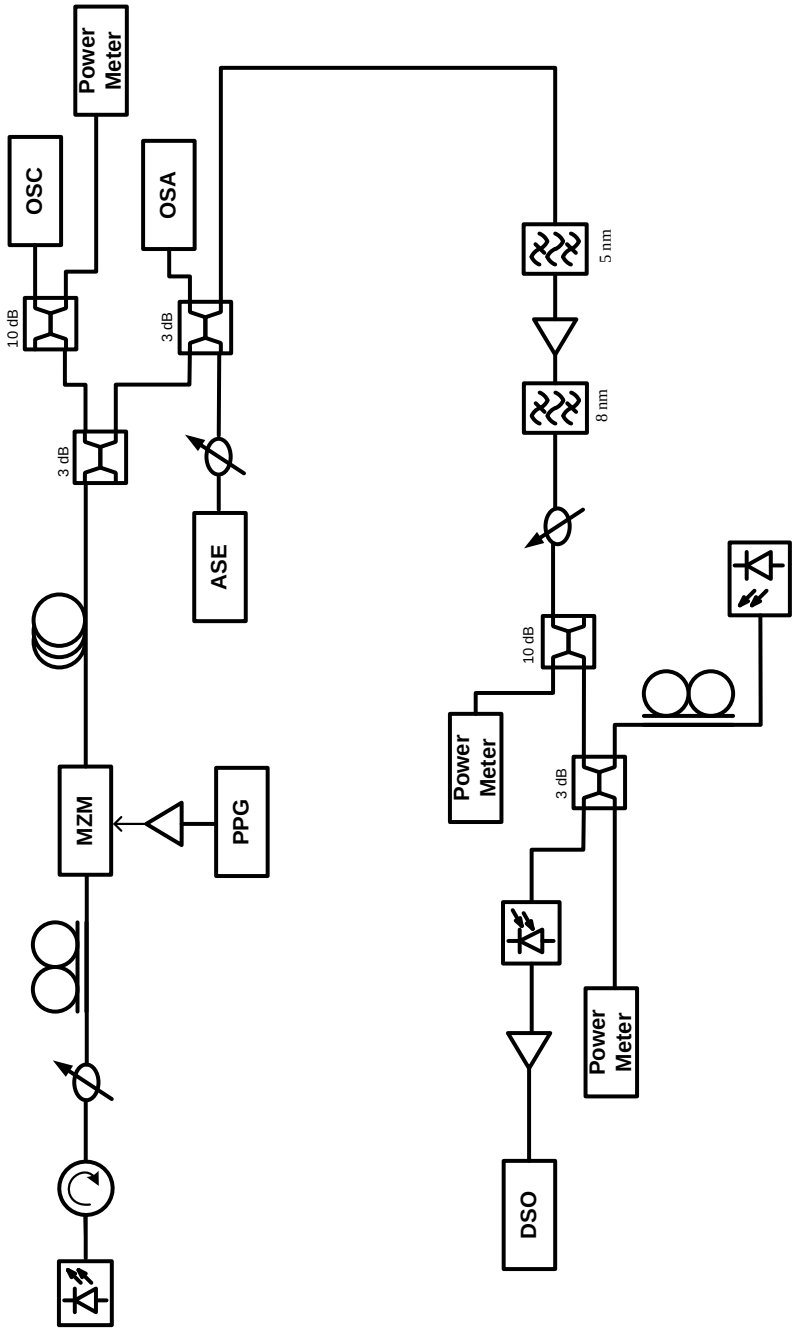


Figura E.5: Schematic for a Radio sobre Fibra transmission. Set-up parameters are $1Gbps$ of bit rate, signal modulated at $13.8GHz$ for OSNR among 18 and $1dB$. Sampling frequency at $80GHz$.

E.2.2 Offline demodulation

Unlike simulations in the previous chapter, we have to put an extra effort to determine which sample is the first for every symbol. We did not have this problem before in the symbol synchronization because transmission was also under our control. Until now, we have just taken a part of the generated PRBS signal, no matter if it was the beginning of the communication or not.

For every captured signal, their symbols have a fixed Nss. Our goal is to find the location of that first sample in one of the first symbols. From that, we can demodulate the signal as we were doing previously.

To achieve this synchronization we implement one block previous to the BPF. It squares the input signal and from its output we search for a maximum in the first transmitted symbols. From all the samples which compose a symbol, that maximum will be in $\frac{Nss}{2}$, so that position means that we are located in the middle of a transmitted one. We reject the next $\frac{Nss}{2}$ samples and since then, synchronization for next symbols is assured.

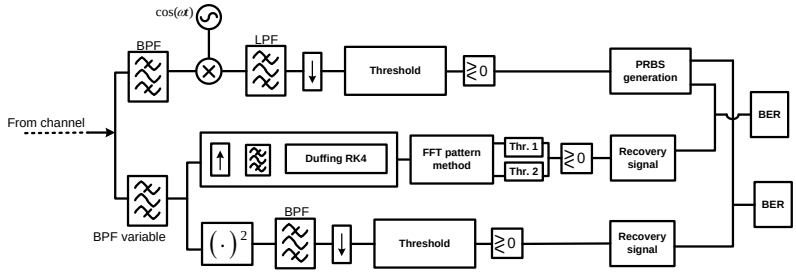


Figura E.6: Schematic of the demodulation implemented script. The set-up is composed by three branches. The first simulates a coherent detection receiver. The second branch is the chaotic Duffing receiving system. The third branch in the envelope detection receiver.

Later on, signal goes through three different paths and from them we obtain the curves Eb/No vs. BER. Set-up can be graphically seen in Figure E.6.

1. Ideal coherent detection branch is composed by a perfect matched BPF and connected to a block which demodulates signal. After all the process, recovered signal will be the most accurate output possible to obtain. Thus, we will take its first 15 bits and thanks to that, we will be able to regenerate the transmitted PRBS signal. That will be used further to compare PRBS signal with recovery signals from the two methods of demodulation described just below.

2. Duffing branch is the Duffing set-up used previously in numerous simulations in sections D.5 and D.6. In the entrance of its branch is defined a tuneable BPF, it will be used to compare its response for different width values. Several test will be carried out with and without the reference signal.

3. Envelope detection branch is similar to the set-up implemented in section D.4. It will be the easiest method to compute. Its output will be also studied for scenarios where BPF is not always perfectly matched.

E.2.3 Experimental results of the set-up

The most interesting data is the one where bit rate is $1Gbps$ and carrier frequency about $13.8GHz$. It is because the Nss for this case is 80. In the last versions of our code this value could be taken as acceptable with any upsampling block. But even if we increase the Nss, less extra samples were needed. That is why it will be the most convenient measurement from the laboratory.

Another point of interest resides in the comparison between the OSNR measured from the laboratory and the Eb/No used in the previous chapter for simulations. The first alludes to the optical domain and is the ratio between the signal power and the noise power in a given bandwidth. For our case a reference bandwidth of 0.1 nm is used. The second stage is to measure the signal to noise ratio for a digital communication system. It is measured at the input of the receiver and it is used as the basic reference of how strong the signal is.

In order to merge the two different measures in a same plot we have to

find an equivalence between them. We opt for an empirical procedure to convert OSNR into its equivalent Eb/No. From the laboratory signal we calculate its power, it will be the power of signal plus noise. Then, we search for one transmitted zero in the input and calculate also its power. This zero should be surrounded by zeros in order to avoid ISI as much as possible. If noise and signal are uncorrelated (Gaussian noise), from the previous two values it is possible to extract the signal power and subsequently apply the quotient of signal power and noise power.

Table E.1 shows this relation. Lower values of Eb/No than expected trough [42] are obtained. It is due to factors as optical noise added through the communication, the effect of linewidth and the incoherence between lasers which affect our incoming signal signal.

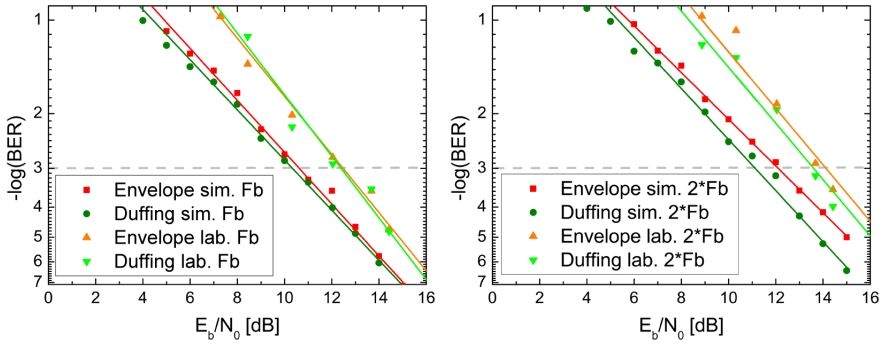
$OSNR[dB]$	$Eb/N_0[dB]$
18	21,55
17	19,41
16	17,94
15	16,32
14	14,42
13	13,68
12	12,04
11	10,32
10	8,87
9	7,52
8	6,21

Tabla E.1: Equivalence between Eb/No and OSNR.

Future plots will not cover such a large range of Eb/No values, opposite to that we could see in the simulation chapter. It is due to the difficulty in assuring the validity of PRBS signal around $8dB$ of OSNR. Close to this value, the theoretical probability of error is around 2%. Not far from there we start to find some errors in the generated PRBS signal. The reason is one error in the 15 taken bits to define this sequence. Thus, the smallest value of OSNR that will be study is $9dB$.

We are going to focus on the case where the chaotic Duffing receiving system has improved its behavior in comparison with the incoherent method. The system utilized for demodulating was the modification of the conventional Duffing oscillator where the reference signal is not needed.

So, we are going to compare between chaotic Duffing receiving system and envelope detection receiver. First, it is illustrated in Figure E.7(a) the result comparative for the case of a matched filter. The BPF is centered in the carrier frequency, its FWHM is the same as the signal bandwidth. This equivalent bandwidth will be $1GHz$. It can be seen how similar these curves are for envelope and Duffing detection, where the performance in terms of Eb/No is compared. There is a negligible difference between both method for the experimental results. The simulation results were previously plotted in Figure D.25(b) and we can observe a penalty about $1.5dB$ between data from the lab and simulated ones.



(a) Ideal reception filter, $1GHz$. (b) Reception filter width 2*Fb, $2GHz$.

Figura E.7: Comparison between simulated and experimental results for envelope detection receiver and chaotic Duffing receiving system. Communication parameters are $1Gbps$ of bit rate, signal modulated at $13.8GHz$ and sampling frequency at $80GHz$.

In the next scenario, shown in Figure E.7(b), BPF is set to $2GHz$, two times the signal bandwidth. We observe from the figure a constant performance improvement in terms of Eb/No for all the range of the signal between envelope detection and the Duffing system. There is an improvement of around $0.5dB$ for the case of the chaotic Duffing receiving system.

Penalty is defined as the difference in performance between experimental and simulated results. For a comparison with a BPF of $2GHz$, this difference is about $1.7dB$. All those references above are referred to the grey dashed line which is equivalent to $10^{-3}dB$ of BER.

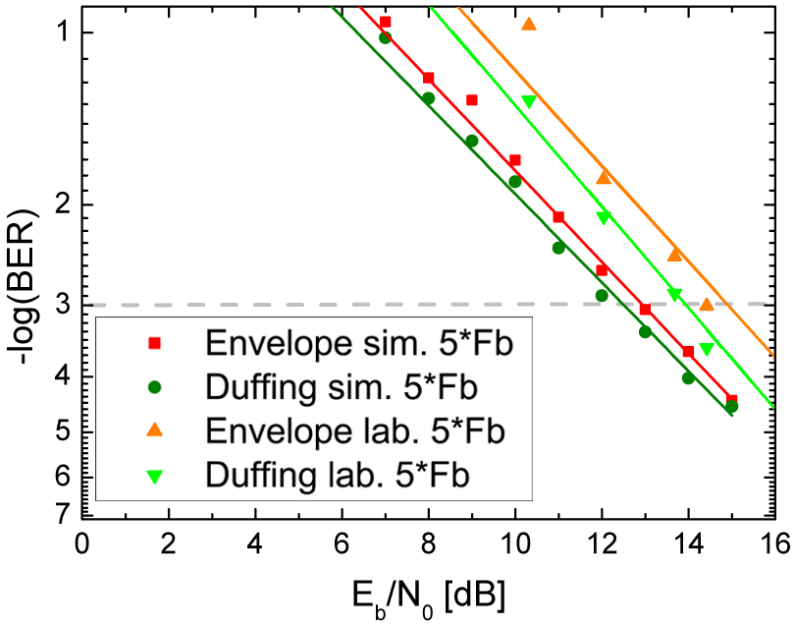


Figura E.8: Comparison between simulated and experimental results for envelope detection and Duffing methods where the FWHM is five times the signal bandwidth, $5GHz$. Communication parameters are $1Gbps$ of bit rate, signal modulated at $13.8GHz$ and sampling frequency at $80GHz$.

Last scenario studied is the case when the FWHM of the BPF is five times the signal bandwidth. The current equivalent bandwidth will be $5GHz$. We have not defined any BPF wider because this width was the limit where we could achieve $10^{-3}dB$ of BER for a reasonable Eb/No. In this scenario the improvement in performance is around $1dB$ between the envelope detection and the Duffing system in terms of Eb/No. This is the best result achieved and the improvement is higher than expected from the simulation results. Even so, penalty between experimental and simulated results is about $1.8dB$.

Penalty between simulated and experimental environments is due to many factors. We have carried out simulations taking into account gaussian noise mainly but in optical environments there are other factors to evaluate, which could make our signal get worse. One of them is the frequency uncertainty between lasers free running. Another factor that could get our signal worse is the frequency variation due to the beating of the phase noise of the lasers.

To sum up, we have defined and implemented a modification of the conventional chaotic Duffing receiving system. The performance of this system has been compared in terms of Eb/No with envelope detector in the same conditions as possible, turning out into a better behavior for some scenarios. This improvement is accomplished when the FWHM of our BPF in reception becomes wider than the signal bandwidth. The maximum difference in performance in terms of Eb/No has been achieved in the case of a FWHM of the BPF five times the signal bandwidth, equivalent bandwidth of $5GHz$, being the improvement of about $1dB$.

Apéndice F

Integral Implementation

F.1 Duffing implemented with Euler Forward method

```
omega=2*pi*f;
K1=omega^2;
K2=b*omega;
K3=(0.5/omega);

%var ADD
out_add=zeros(1,length(t));

%var INTEG
salida=zeros(1,length(t));
salida_2=zeros(1,length(t));
aux4=zeros(1,length(t));
h=t(2)-t(1);

for j=2:length(t)

    salida(j)=salida(j-1)+h*out_add(j-1);
    salida_2(j)=salida_2(j-1)+h*salida(j-1);

    aux4(j)=salida_2(j)-salida_2(j)^3;

    out_add(j)=-K2*salida(j)+K1*driver(j)+K1*signal(j)+K1*aux4(j
        );

end
```

F.2 Duffing implemented with Trapezoidal rule method

```
signal_in = yfilt_upsample;
max_signal = max(signal_in);
signal=signal_in./max_signal';

phase_duff = 0*2*pi;
diff_freq = 0.0;
```

```
7  driver = cos(2*pi*F_RF*(1-diff_freq)*t_up+phase_duff);
8
9  % var2
10 omega=2*pi*F_RF;
11
12 k=0.3; %default value k = 0.3;
13 r=0.3; %default value k = 0.6;
14
15 number_points = length(yfilt_upsample);
16
17 w2b=0;w2c=0;
18 xd=-1;x=-1;
19 h=1/Fs_up;
20 salida=zeros(1,number_points);
21 salida_2=zeros(1,number_points);
22 tv=zeros(1,number_points);
23
24 for i=1:length(signal)-1
25
26     xdd=-k*w*xd+w^2*(x-x^3+r*driver(i)+20*signal(i)); % out sin
       integrar
27
28     w1b=xdd+w2b;
29     w2b=xdd+w1b;
30
31     xd=w1b/(2*Fs_up); % out tras una integral
32
33     w1c=xd+w2c;
34     w2c=xd+w1c;
35
36     x=w1c/(2*Fs_up); % out tras las 2 integrales
37
38     salida(1,i)=xd;
39     salida_2(1,i)=x;
40     tv(1,i)=t_up(i);
41
42 end
```

F.3 Duffing implemented with Fourth-Order Runge-Kutta method

```
1  phase_duff = 0*pi;
2  diff_freq = 1;
3  ymod_duff = cos(2*pi*F_RF*diff_freq*t_up+phase_duff);
4
5  driver = ymod_duff; signal_in = yfilt_upsample;
6  max_signal = max(signal_in);
7  signal=signal_in./max_signal';
8
9  % var2
```

```
omega=2*pi*F_RF;
K1=omega^2;
K2=1*omega;
K3=(1/omega);

%var ADD
out_add=zeros(1,length(t_up));

%var INTEG
salida=zeros(1,length(t_up));
salida_2=zeros(1,length(t_up));
aux4=zeros(1,length(t_up));
h=1/Fs_up;

% var3
gamma = 0.5; % DAMPING COEFFICIENT -> changeless
fr = 0;%-1/max(ynoisy); % fd = 0.8274;
A = 4;

for i=1:length(signal)-1

    x1 = salida_2(i);
    y1 = salida(i);
    f1 = -gamma*K2*y1+K1*(x1-x1^3+fr*driver(i)+A*signal(i));

    x2 = salida_2(i)+y1*(h/2);
    y2 = salida(i)+f1*(h/2);
    f2 = -gamma*K2*y2+K1*(x2-x2^3+fr*driver(i)+A*signal(i));

    x3 = salida_2(i)+y2*(h/2);
    y3 = salida(i)+f2*(h/2);
    f3 = -gamma*K2*y3+K1*(x3-x3^3+fr*driver(i+1)+A*signal(i+1));

    x4 = salida_2(i)+y3*h;
    y4 = salida(i)+f3*h;
    f4 = -gamma*K2*y4+K1*(x4-x4^3+fr*driver(i+1)+A*signal(i+1));

    salida_2(i+1) = salida_2(i)+((h/6)*(y1+2*y2+2*y3+y4));
    salida(i+1) = salida(i)+((h/6)*(f1+2*f2+2*f3+f4));

end
```

Apéndice G

Matlab simulations for the three basic modulations

G.1 Code for BASK signal detection

```
clear all
close all
clc

k=0.3; %default value k = 0.3;
r=0.6; %default value k = 0.6;

D=0.1;
w=10;
w2b=0;w2c=0;
xd=-1;x=-1;
tfinal=50;
fs=1000;
h=1/fs;
number_points=fs*tfinal;
salida=zeros(1,number_points);
salida_2=zeros(1,number_points);
tv=zeros(1,number_points);
bits = round(rand(1,tfinal/10));
mod = kron(bits,ones(1,fs*10));
noise_total = 0;

t = linspace(0,tfinal-h,number_points);
phase = 1*pi;
signal = mod.*cos(w*t+phase);
driver = cos(w*t);
noise = 2*randn(1,number_points);

for i=1:number_points
   t=(i-1)*h;
     signal = mod(i)*cos(w*t(i));
     driver = cos(w*t(i));
     noise(i) = A;
    noise_total = noise_total+abs(noise(i)).^2;
    xdd=-k*w*xd+w^2*(x-x^3+r*driver(i)+noise(i)+signal(i)); %
        out sin integrar
    w1b=xdd+w2b;
    w2b=xdd+w1b;
    xd=w1b/(2*fs); % out tras una integral
```

```
39      w1c=xd+w2c;
40      w2c=xd+w1c;
41      x=w1c/(2*fs); % out tras las 2 integrales
42      salida(1,i)=xd;
43      salida_2(1,i)=x;
44      tv(1,i)=t(i);
45  end
46
47  pot_mod = sum(abs(mod).^2)/length(mod);
48  pot_noise = noise_total/number_points;
49  SNR = 10*log10(pot_mod/pot_noise)
```

G.2 Code for BPSK signal detection

```
1   clear all
2   close all
3   clc
4
5   k=0.3; %default value k = 0.3;
6   r=0.6; %default value k = 0.6;
7
8
9   w=10;
10  w_neg = w;
11  w2b=0;w2c=0;
12  xd=-1;x=-1;
13  tfinal=500;
14  fs=1000;
15  h=1/fs;
16  number_points=fs*tfinal;
17  salida=zeros(1,number_points);
18  salida_2=zeros(1,number_points);
19  tv=zeros(1,number_points);
20
21  bits = round(rand(1,tfinal/10));
22  bits_neg = not(bits);
23  mod = kron(bits,ones(1,fs*10));
24  mod_neg = kron(bits_neg,ones(1,fs*10));
25  noise_total = 0;
26
27  t = linspace(0,tfinal-h,number_points);
28  phase = 1*pi;
29  signal_f1 = mod.*cos(w*t+phase);
30  signal_f2 = mod_neg.*cos(w_neg*t+phase+pi);
31  signal = signal_f1 + signal_f2;
32  driver = cos(w*t);
33  noise = 2*randn(1,number_points);
34
35  for i=1:number_points
36
```

```
37      noise_total = noise_total+abs(noise(i)).^2;
38      xdd=-k*w*xd+w^2*(x-x^3+r*driver(i)+noise(i)+signal(i)); %
            out sin integrar
39      w1b=xdd+w2b;
40      w2b=xdd+w1b;
41      xd=w1b/(2*fs); % out tras una integral
42      w1c=xd+w2c;
43      w2c=xd+w1c;
44      x=w1c/(2*fs); % out tras las 2 integrales
45      salida(1,i)=xd;
46      salida_2(1,i)=x;
47      tv(1,i)=t(i);
48
49  end;
```

G.3 Code for BFSK signal detection

```
1   clear all
2   close all
3   clc
4
5   k=0.3; %default value k = 0.3;
6   r=0.6; %default value k = 0.6;
7
8   w=10;
9   w_neg = 14;
10  w2b=0;w2c=0;
11  xd=-1;x=-1;
12  tfinal=500;
13  fs=1000;
14  h=1/fs;
15  number_points=fs*tfinal;
16  salida=zeros(1,number_points);
17  salida_2=zeros(1,number_points);
18  tv=zeros(1,number_points);
19
20  bits = round(rand(1,tfinal/10));
21  bits_neg = not(bits);
22  mod = kron(bits,ones(1,fs*10));
23  mod_neg = kron(bits_neg,ones(1,fs*10));
24  noise_total = 0;
25
26  t = linspace(0,tfinal-h,number_points);
27  phase = 1*pi;
28  signal_f1 = mod.*cos(w*t+phase);
29  signal_f2 = mod_neg.*cos(w_neg*t+phase);
30  signal = signal_f1 + signal_f2;
31  driver = cos(w*t);
32  noise = randn(1,number_points);
33
```

```
34  for i=1:number_points
35      t=(i-1)*h;
36      signal = mod(i)*cos(w*t(i));
37      driver = cos(w*t(i));
38      noise(i) = A;
39      noise_total = noise_total+abs(noise(i)).^2;
40      xdd=-k*w*xd+w^2*(x-x^3+r*driver(i)+noise(i)+signal(i)); %
            out sin integrar
41      w1b=xdd+w2b;
42      w2b=xdd+w1b;
43      xd=w1b/(2*fs); % out tras una integral
44      w1c=xd+w2c;
45      w2c=xd+w1c;
46      x=w1c/(2*fs); % out tras las 2 integrales
47      salida(1,i)=xd;
48      salida_2(1,i)=x;
49      tv(1,i)=t(i);
50  end
```

Apéndice H

Matlab simulations for the performance evaluation for ASK

H.1 Code for transmitter and noisy channel

```
clear all;
close all;
clc;

%Bit and smapling Rates
Fs =80e9;       %Sampling frequency
Ts=1/Fs;

Fb = 80e6;
Tb=1/Fb;

F_RF= 600e6;        %Modulation frequency

Nss=Fs/Fb;
num_symb = 1e3;     %Number symbols
ts=num_symb/Fb;     %Length of simulation

%Time and Frequency axis
t=linspace(0,(ts-1/Fs),num_symb*Nss)'; %temporal axe
fplot=linspace(-Fs/2,Fs/2,Nss*num_symb)'; % for plotting (
    fftshift signal)

% Signal generation
PRBS= load('autentico_PRBS15_2^15.txt');
PRBS=PRBS';
rep_PRBS=ceil(num_symb/(length(PRBS)));
bits1=repmat(PRBS,[rep_PRBS 1]);
bits1 = bits1(1:num_symb);
clear PRBS;

% Upsample
y_up = upsample(bits1,Nss);
y_up = y_up(1:length(t));

% Raised cosine
delay = 3;
```

```
36 [num,den] = rcosine(1,Nss,'fir/sqrt',1,delay);
37 yinf2 = filter(num,den,y_up);
38 yinf2 = yinf2./max(max(yinf2));
39 yinf2 = yinf2(3*Nss:end);
40
41
42 % Modulation
43 y_cos = cos(2*pi*F_RF.*t(1:end-3*Nss+1));
44 ymod = yinf2'.*y_cos;
45
46 % Noise awg
47 SNR = 0;
48 pot_ymod = sum(abs(ymod).^2)/length(ymod);
49 pot_ymod_db = 10*log10(pot_ymod);
50 noise_power_db = pot_ymod_db - SNR;
51 noise_power = 10^(noise_power_db*0.1);
52 noise = randn(length(ymod),1)*sqrt(noise_power);
53 ynoisy = ymod+noise;
54 Eb_No = 10*log10(Fs/(2*Fb))+SNR
55 Eb_No_pp1 = 10*log10(1/(2*noise_power))
56 % ynoisy = awgn(ymod,SNR,'measured');
57
58 % Quantization Noise
59 Ei = ynoisy;
60 norm=max([Ei]);
61 Ei2= (Ei./norm);
62 pn = {'mode', 'roundmode', 'overflowmode', 'format'};
63 pv = {'fixed', 'ceil', 'saturate', [6 5]}; %Quantification 8
       bits A/D
64 q = quantizer(pn, pv);
65 EiQ = num2int(q,Ei2)./(2^7);
66 ynoisy_quant = EiQ;
67
68
69 file = 'channel.mat';
70 ymod_tx = ymod;
71 % save(file, 'ynoisy', 'SNR', 'Nss_tx', 'bits1', 'num_symb', '
       ymod_tx', 'yinf2', 'y_cos');
72 save(file, 'Fs', 'Fb', 'F_RF', 'SNR', 'num_symb', 'bits1', '
       yinf2', 'y_cos', 'ymod_tx', 'ynoisy');
```

H.2 Code for receiver

```
1 clear all;
2 close all;
3 clc;
4
5 file1 = 'channel.mat';
6 load(file1, 'Fs', 'Fb', 'F_RF', 'SNR', 'num_symb', 'bits1', '
      yinf2', 'y_cos', 'ymod_tx', 'ynoisy');
```

```

Ts = 1/Fs;
Tb = 1/Fb;
ts = num_symb/Fb;
Nss = Fs/Fb;

%Time and Frequency axis
t = linspace(0,(ts-1/Fs),num_symb*Nss)'; %temporal axe
fplot = linspace(-Fs/2,Fs/2,Nss*num_symb)'; % for plotting (
    fftshift signal)

% Band pass filter reception
Wo_rx= 0.92;
[B_rx,A_rx] = besself(1,Wo_rx);
yfilter_rx = filter(B_rx,A_rx,ynoisy);
ynorm_rx = yfilter_rx./max(max(yfilter_rx));
yout_rx=ynorm_rx-mean(ynorm_rx);

% %butterworth filter
Wn = [F_RF-Fb F_RF+Fb]*(2/Fs);
[b_duff,a_duff] = butter(4,Wn);
ynoisy2 = filter(b_duff,a_duff,ynoisy);
ynoisy2 = ynoisy2./max(max(ynoisy2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 1 -> duffing
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Nss_tx = Nss;

% Upsample (interpolation)
Nss_duff = Fs/Fb;
Fs_up = Nss_duff*Fs/(Nss_tx);
ynoisy_up = upsample(ynoisy,Nss_duff/Nss_tx);
ts_up = length(ynoisy_up)/(Nss_duff*Fb);
t_up = linspace(0,(ts_up-1/Fs_up),length(ynoisy_up))';


% Band pass filter (interpolation)
Wn_upsample = [F_RF-(Fb*1) F_RF+(Fb*1)]*(2/Fs_up);
[b_upsample,a_upsample] = butter(5,Wn_upsample);
yfilt_upsample = filter(b_upsample,a_upsample,ynoisy_up);
yfilt_upsample = yfilt_upsample./max(yfilt_upsample);

phase_duff = 0.5*pi;
diff_freq = 1;
ymod_duff = cos(2*pi*F_RF*diff_freq*t_up+phase_duff);

% var1
% c = 0.81; % c=0.77; %0.82673
% A = 1; % A=0.2; %0.2

driver = ymod_duff; signal_in = ynoisy;
% driver_in = ymod_duff; signal_in = yfilt_upsample;
```

```
% max_driver = max(driver_in);
% driver=driver_in./max_driver';
max_signal = max(signal_in);
signal=signal_in./max_signal';
% driver = driver_in;
% signal = signal_in;
% H = A*signal+c*driver(1:length(signal));
% H = H./max(H);

% var2
omega=2*pi*F_RF;
K1=omega^2;
K2=1*omega;
K3=(1/omega);

%var ADD
out_add=zeros(1,length(t_up));

%var INTEG
salida=zeros(1,length(t_up));
salida_2=zeros(1,length(t_up));
aux4=zeros(1,length(t_up));
h=1/Fs_up;

% var3
gamma = 0.5; % DAMPING COEFFICIENT -> changeless
fr = 0.77%-1/max(ynoisy); % fd = 0.8274;
A = 1;
% omega_R = 1;
% R = 4*cosh(pi*omega_R*0.5)/(3*sqrt(2)*pi*omega_R);

for i=1:length(signal)-1

    x1 = salida_2(i);
    y1 = salida(i);
    f1 = -gamma*K2*y1+K1*(x1-x1^3+fr*driver(i)+A*signal(i));

    x2 = salida_2(i)+y1*(h/2);
    y2 = salida(i)+f1*(h/2);
    f2 = -gamma*K2*y2+K1*(x2-x2^3+fr*driver(i)+A*signal(i));

    x3 = salida_2(i)+y2*(h/2);
    y3 = salida(i)+f2*(h/2);
    f3 = -gamma*K2*y3+K1*(x3-x3^3+fr*driver(i+1)+A*signal(i+1));

    x4 = salida_2(i)+y3*h;
    y4 = salida(i)+f3*h;
    f4 = -gamma*K2*y4+K1*(x4-x4^3+fr*driver(i+1)+A*signal(i+1));

    salida_2(i+1) = salida_2(i)+((h/6)*(y1+2*y2+2*y3+y4));
    salida(i+1) = salida(i)+((h/6)*(f1+2*f2+2*f3+f4));

end
```

H.3 Code for envelope detection

```
yout_env = (salida_2).^2;
Wn = (Fb*0.7)*(2/Fs);
[B,A] = butter(5,Wn,'low');
yout_meth1 = filter(B,A,yout_env);
yout_meth1 = yout_meth1./max(yout_meth1);

Y_variance=zeros(1,Nss);
for i=1:Nss
    Y_variance(i)=sum(abs(yout_meth1(i:Nss:end).^2));
end
[value index] = max(Y_variance);

ydownsamp_meth1 = yout_meth1(index:Nss:end);
ydownsamp_meth1 = ydownsamp_meth1./max(ydownsamp_meth1);

[a_duff,b_duff] = hist(ydownsamp_meth1(2:end),100);
pos_ceros = max(a_duff);
tt = find(b_duff<=mean(aux_duff),1,'last');
pos_ones_rel = max(a_duff(tt:end));
[val,pos_tr] = min(a_duff(pos_ceros:pos_ones_rel+tt));
threshold_duff = b_duff(pos_tr+pos_ceros);

aux1_meth1 = find(ydownsamp_meth1 > threshold_meth1);
aux2_meth1 = find(ydownsamp_meth1 <= threshold_meth1);
u_recovered_meth1(aux1_meth1) = 1;
u_recovered_meth1(aux2_meth1) = 0;

[error_number_meth1,error_rate_meth1]=symerr(bits1(1:length(
    u_recovered_meth1)-1),u_recovered_meth1(2:end));
BER_log_meth1 = -log10(error_rate_meth1);
```

H.4 Code for phase variance

```
yout_phase_var = (salida_2).*ymod_duff'+1i.*(salida_2).*(
    ymod_duff*exp(1i*pi/2))';
yout_angle = angle(yout_phase_var);
diff_phase = abs(yout_angle-circshift(yout_angle',-1)');
diff_phase = diff_phase(1:end-1);

for i2=1:length(salida_2)/Nss
    yout_meth2(i2) = sum(diff_phase((i2-1)*Nss+11:i2*Nss-10));
end;

[a_duff,b_duff] = hist(yout_meth2,50);
pos_ceros = max(a_duff);
tt = find(b_duff<=mean(aux_duff),1,'last');
pos_ones_rel = max(a_duff(tt:end));
[val,pos_tr] = min(a_duff(pos_ceros:pos_ones_rel+tt));
```

```
threshold_duff = b_duff(pos_tr+pos_ceros);

aux1_meth2 = find(yout_meth2 > threshold_meth2);
aux2_meth2 = find(yout_meth2 <= threshold_meth2);
u_recovered_meth2(aux1_meth2) = 1;
u_recovered_meth2(aux2_meth2) = 0;

[error_number_meth2,error_rate_meth2]=symerr(bits1(1:length(
    u_recovered_meth2)),u_recovered_meth2(1:end));
BER_log_meth2 = -log10(error_rate_meth2);
error_number_meth2
```

H.5 Code for mean method

```
for i3=1:length(salida_2)/Nss
    yout_meth3(i3) = mean(salida_2((i3-1)*Nss+101:i3*Nss-100));
end;
yout_meth3 = abs(yout_meth3);

[a_duff,b_duff] = hist(yout_meth3,50);
pos_ceros = max(a_duff);
tt = find(b_duff<=mean(aux_duff),1,'last');
pos_ones_rel = max(a_duff(tt:end));
[val,pos_tr] = min(a_duff(pos_ceros:pos_ones_rel+tt));
threshold_duff = b_duff(pos_tr+pos_ceros);

aux1_meth3 = find(yout_meth3 > threshold_meth3);
aux2_meth3 = find(yout_meth3 <= threshold_meth3);
u_recovered_meth3(aux1_meth3) = 0;
u_recovered_meth3(aux2_meth3) = 1;

[error_number_meth3,error_rate_meth3]=symerr(bits1(1:length(
    u_recovered_meth3)),u_recovered_meth3(1:end));
BER_log_meth3 = -log10(error_rate_meth3);
error_number_meth3
```

H.6 Code for variance method

```
%  first implementation
L_var = Nss;
for i4 = 1:(length(salida_2)-L_var+1)
    variance(i4) = var(salida_2(i4:i4+L_var-1));
end
yout_meth4 = abs(variance);

% second implementation
```

```
length_var = 1000;
variance = zeros(1,length(salida_2)/(length_var));
for i4=1:round(length(salida_2)/length_var)
    variance(i4) = var(salida_2((i4-1)*length_var+51:i4*
        length_var-50));
end;

for i5=1:length(variance)/10
    yout_meth4(i5) = mean(variance((i5-1)*10+1:i5*10));
end;

for i=1:0.01*Nss
     Y_variance(i)=sum(abs(variance(i:Nss:end).^2));
end
[value index] = max(Y_variance);
index = 5

ydownsamp_meth4 = variance(index:Nss/length_var:end);
ydownsamp_meth4 = ydownsamp_meth4./max(ydownsamp_meth4);

[a_duff,b_duff] = hist(variance,50);
pos_ceros = max(a_duff);
tt = find(b_duff<=mean(aux_duff),1,'last');
pos_ones_rel = max(a_duff(tt:end));
[val,pos_tr] = min(a_duff(pos_ceros:pos_ones_rel+tt));
threshold_duff = b_duff(pos_tr+pos_ceros);


aux1_meth4 = find(variance > threshold_meth4);
aux2_meth4 = find(variance <= threshold_meth4);
u_recovered_meth4(aux1_meth4) = 0;
u_recovered_meth4(aux2_meth4) = 1;
u_recovered_meth4 = u_recovered_meth4(2:end);

[error_number_meth4,error_rate_meth4]=symerr(bits1(2:length(
    u_recovered_meth4)+1),u_recovered_meth4(1:end));
BER_log_meth4 = -log10(error_rate_meth4);
error_number_meth4
```

H.7 Code for FFT pattern method

```
fBit = 1;
numBits = 20;
B = reshape(salida_2((fBit-1)*Nss_duff+1:((fBit+numBits-1)*
    Nss_duff)),Nss_duff,numBits);
FFT_B = fft(B)./length(B);
fplot_B = linspace(-Fs_up,Fs_up,Nss_duff);
close all;
figure
for var_bits=1:numBits
```

```
subplot(4,round(numBits/4),var_bits)
plot(20*log10(fftshift(abs(FFT_B(:,var_bits))))+30);
% plot(xcorr(20*log10(fftshift(abs(FFT_B(:,var_bits))))+30));
x=(20*log10(fftshift(abs(FFT_B(:,var_bits))))+30);
max_fft_plot(var_bits) = x(467);
submax_fft(var_bits) = max(x(470:480));
xlabel(num2str(x(467)))
ylabel(num2str(submax_fft(var_bits)))
% legend(num2str(bits1(fBit+var_bits-1)));
end
end;

% NO PLOT
B_calc = reshape(salida_2,Nss_duff,length(salida_2)/Nss_duff);
FFT_B_calc = fft(B_calc)./size(B_calc,1);
for fft_var=1:length(salida_2)/Nss_duff
x_calc=(20*log10(fftshift(abs(FFT_B_calc(:,fft_var))))+30);
max_fft_plot_calc(fft_var) = x_calc(467,:);
submax_fft_plot_calc(fft_var) = max(x_calc(470:480));
end;

% decision = [max_fft_plot_calc; submax_fft_plot_calc; 2*ones(1,
    length(salida_2)/Nss_duff); bits1(1:length(max_fft_plot_calc
    ))];
decision = [max_fft_plot_calc; submax_fft_plot_calc; 2*ones(1,
    length(salida_2)/Nss_duff)];
threshold_max_fft = 25; threshold_submax_fft = 13;

for var_decision=1:length(salida_2)/Nss_duff
if ((decision(1,var_decision) > threshold_max_fft) & (decision
    (2,var_decision) < threshold_submax_fft))
    decision(3,var_decision) = 1;
elseif ((decision(1,var_decision) < threshold_max_fft) & (
    decision(2,var_decision) > threshold_submax_fft))
    decision(3,var_decision) = 0;
elseif (1.2*decision(1,var_decision) > threshold_max_fft) &
    (0.8*decision(2,var_decision) < threshold_submax_fft)
    decision(3,var_decision) = 1;
else  decision(3,var_decision) = 0;
end;
end
```

H.8 Code for ASK envelope demodulation compared with theoretical values

```
clear all;
close all;
clc;

%Bit and smapling Rates
```

```
6  Fs = 80e9;
7  Ts=1/Fs;
8
9  Fb = 1e9;
10 Tb=1/Fb;
11
12 F_RF= 13.8e9;         %Modulation frequency
13
14 Nss=Fs/Fb;
15 num_symb = 5e4;      %Number symbols
16 ts=num_symb/Fb;      %Length of simulation
17
18 %Time and Frequency axis
19 t = linspace(0,(ts-1/Fs),num_symb*Nss)'; %temporal axe
20 fplot = linspace(-Fs/2,Fs/2,Nss*num_symb)'; % for plotting (
       fftshift signal)
21
22 % Signal generation
23 PRBS = round(rand(1,num_symb));
24 rep_PRBS = ceil(num_symb/(length(PRBS)));
25 bits1 = repmat(PRBS,[rep_PRBS 1]);
26 bits1 = bits1(1:num_symb);
27 clear PRBS;
28
29 yinf2 = kron(bits1,ones(1,Nss));
30 yinf2 = yinf2(1:length(t));
31
32 % Modulation
33 y_cos = cos(2*pi*F_RF.*t);
34 ymod = yinf2'.*y_cos;
35 ymod = ymod';
36
37 values_SNR = -5:15;
38 for var_SNR = 1:length(values_SNR)
39
40 % Noise
41 SNR_real = values_SNR(var_SNR)+4.2;
42 SNR_extra = 10*log10(Fs/(2*Fb));
43 SNR = SNR_real - SNR_extra;
44 ebn0 = 10^(SNR/10);
45 eb = sum(ymod.^2)/(length(ymod));
46 n0 = eb/ebn0;
47 pn = n0;
48 n = sqrt(pn)*randn(1,length(ymod));
49 ynoisy = ymod+n;
50
51
52 %Butterworth filter
53 Wn = [F_RF-Fb*5 F_RF+5*Fb]*(2/Fs);
54 [b_duff,a_duff] = butter(5,Wn);
55 ynoisy2 = filter(b_duff,a_duff,ynoisy);
56 ynoisy_filt = ynoisy2./max(ynoisy2);
57 ynoisy_filt = [ynoisy_filt(41:end) zeros(1,40)];
58
```

```
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % Envelope detection
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

yout_env = (ynoisy_filt).^2;
Wn = (Fb*1)*(2/Fs);
[B,A] = butter(5,Wn,'low');
yout_meth1 = filter(B,A,yout_env);
yout_meth1 = yout_meth1./max(yout_meth1);

Y_variance=zeros(1,Nss);
for i=1:Nss
    Y_variance(i)=sum(abs(yout_meth1(i:Nss:end).^2));
end
[value index] = max(Y_variance);

ydownsamp_meth1 = yout_meth1(index:Nss:end);
ydownsamp_meth1 = ydownsamp_meth1./max(ydownsamp_meth1);

[a_duff,b_duff] = hist(ydownsamp_meth1,40);
[value pos_ceros] = max(a_duff);
tt = find(b_duff<=mean(ydownsamp_meth1),1,'last');
[value pos_ones_rel] = max(a_duff(tt:end));
[val,pos_tr] = min(a_duff(pos_ceros:pos_ones_rel+tt));
threshold_env = b_duff(pos_tr+pos_ceros);

aux1_meth1 = find(ydownsamp_meth1 > threshold_env);
aux2_meth1 = find(ydownsamp_meth1 <= threshold_env);
u_recovered_meth1(aux1_meth1) = 1;
u_recovered_meth1(aux2_meth1) = 0;

[error_number_env(var_SNR),error_rate_env(var_SNR)]=symerr(bits1(1:end),u_recovered_meth1(1:end));
% [error_number_env(var_SNR),error_rate_env(var_SNR)]=symerr(bits1(1:end-1),u_recovered_meth1(2:end));
BER_log_env = -log10(error_rate_env);
find(bits1(1:end-1) == ~(u_recovered_meth1(2:end)));
%error_number_env

end;


BER = error_number_env / num_symb;
semilogy(values_SNR,BER,'*');
hold on;
BER_nocoher_id = berawgn(values_SNR, 'fsk', 2, 'noncoherent');
semilogy(values_SNR, BER_nocoher_id,'ro');
grid on;
xlabel('E_b/N_0 (dB)');
ylabel('BER');
legend('Simulated values','Theoretical values');
```

H.9 Code for ASK coherent demodulation compared with theoretical values

```
clear all;
close all;
clc;

%Bit and smapling Rates
Fs = 80e9;
Ts=1/Fs;

Fb = 1e9;
Tb=1/Fb;

F_RF= 13.8e9;        %Modulation frequency

Nss=Fs/Fb;
num_symb = 5e4;     %Number symbols
ts=num_symb/Fb;     %Length of simulation

%Time and Frequency axis
t = linspace(0,(ts-1/Fs),num_symb*Nss)'; %temporal axe
fplot = linspace(-Fs/2,Fs/2,Nss*num_symb)'; % for plotting (
    fftshift signal)

% Signal generation
PRBS = round(rand(1,num_symb));
rep_PRBS = ceil(num_symb/(length(PRBS)));
bits1 = repmat(PRBS,[rep_PRBS 1]);
bits1 = bits1(1:num_symb);
clear PRBS;

yinf2 = kron(bits1,ones(1,Nss));
yinf2 = yinf2(1:length(t));

% Modulation
y_cos = cos(2*pi*F_RF.*t);
ymod = yinf2'.*y_cos;
ymod = ymod';

values_SNR = -5:15;

for var_SNR = 1:length(values_SNR)

% Noise
SNR_real = values_SNR(var_SNR)+4.5;
SNR_extra = 10*log10(Fs/(2*Fb));
SNR = SNR_real - SNR_extra;
ebn0 = 10^(SNR/10);
eb = sum(ymod.^2)/(length(ymod));
n0 = eb/ebn0;
pn = n0;
n = sqrt(pn)*randn(1,length(ymod));
```

```
ynoisy = ymod+n;


% Butterworth filter <- MODIFICAR!!
Wn = [F_RF-8*Fb F_RF+8*Fb]*(2/Fs);
[b_duff,a_duff] = butter(5,Wn);
ynoisy2 = filter(b_duff,a_duff,ynoisy);
ynoisy_filt = ynoisy2./max(ynoisy2);
ynoisy_filt = [ynoisy_filt(41:end) zeros(1,40)];

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % 2-> Theoretical demodulation
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
yout_th = ynoisy_filt.*y_cos';

Wn = (Fb*1)*(2/Fs);
[B,A] = butter(5,Wn,'low');
yout_th_rc = filter(B,A,yout_th);
yout_th_rc = yout_th_rc./max(yout_th_rc);

Y_variance=zeros(1,Nss);
for i=1:Nss
    Y_variance(i)=sum(abs(yout_th_rc(i:Nss:end).^2));
end
[value index] = max(Y_variance);

ydownsamp_th = yout_th_rc(index:Nss:end);
ydownsamp_th = ydownsamp_th./max(ydownsamp_th);

hist_samples = 40;
[a_th,b_th] = hist(ydownsamp_th,hist_samples);
[value pos_ceros] = max(a_th);
if pos_ceros < hist_samples/2
    tt = find(b_th<=mean(ydownsamp_th),1,'last');
    [value pos_ones_rel] = max(a_th(tt:end));
    [val,pos_tr] = min(a_th(pos_ceros:pos_ones_rel+tt));
    threshold_th = b_th(pos_tr+pos_ceros);
    else pos_ones = pos_ceros;
    tt = find(b_th>=mean(ydownsamp_th),1,'first');
    [value pos_ceros] = max(a_th(1:tt));
    [val,pos_tr] = min(a_th(pos_ceros:pos_ones));
    threshold_th = b_th(pos_ceros+pos_tr);
end;




% threshold_th = 0.4;

aux1_th = find(ydownsamp_th > threshold_th);
aux2_th = find(ydownsamp_th <= threshold_th);
```

```
101 u_recovered_th(aux1_th) = 1;
102 u_recovered_th(aux2_th) = 0;
103 u_recovered_th = u_recovered_th;
104
105 % [error_number_th(var_SNR),error_rate_th(var_SNR)]=symerr(bits1
        (1:length(u_recovered_th)-1),u_recovered_th(2:end));
106 [error_number_th(var_SNR),error_rate_th(var_SNR)]=symerr(bits1
        (1:length(u_recovered_th)),u_recovered_th(1:end));
107 BER_log_th = -log10(error_rate_th);
108
109 end;
110
111
112 BER = error_number_th / num_symb;
113 semilogy(values_SNR,BER,'*');
114 hold on;
115 BER_coher_id = berawgn(values_SNR, 'fsk', 2, 'coherent');
116 semilogy(values_SNR, BER_coher_id,'ro');
117 grid on;
118 xlabel('E_b/N_0 (dB)');
119 ylabel('BER');
```

Apéndice I

Glosario de Acrónimos

AM Amplitude Modulation

ASE Amplified Spontaneous Emission

ASK Amplitude-Shift Keying

AWG Additive White Gaussian

BER Bit Error Rate

BPF Filtro Paso Banda - *Band-Pass Filter*

DC Direct Current

DSO Digital Storage Oscilloscope

DFB Distributed Feedback Laser

Eb/No Energy per Bit to Noise Power Spectral Density Ratio

ECL External Cavity Laser

EDFA Erbium-Doped Fiber Amplifier

FEC Forward Error Correction

FFT Transformada Rápida de Fourier - *Fast Fourier Transform*

FM Frequency Modulation

FPGA Field-programmable Gate Array

FSK Frequency-Shift Keying

FWHM Full Width at Half Maximum

ISI Interferencia Intersimbólica - *Intersymbol Interference*

LO Oscilador Local - *Local Oscillator*

LPF Filtro Paso Bajo - *Low-Pass Filter*

MZM Modulador Mach-Zehnder - *Mach-Zehnder Modulator*

Nss Número de muestras por símbolo - *Number of samples per symbol*

ODE Ecuación Diferencial Ordinaria - *Ordinary Differential Equation*

OFDM Orthogonal Frequency-Division Multiplexing

OOK On-Off Keying

OSA Analizados de Espectros Ópticos - *Optical Spectrum Analyzer*

OSC Oscilador - *Oscillator*

OSNR Relación Señal a Ruido óptica - *Optical Signal-to-Noise Ratio*

PD Fotodiodo - *Photodiode*

PLL Phase-Locked Loop

PM Power Meter

PPG Pulse Pattern Generator

PRBS Pseudo-Random Binary Sequence

PSD Densidad Espectral de Potencia - *Power Spectral Density*

PSK Phase-Shift Keying

QAM Quadrature Amplitude Modulation

RF Radiofrecuencia - *Radio Frequency*

RoF Radio sobre Fibra - *Radio-over-Fiber*

RK4 Cuarto Orden Runge-Kutta - *Fourth-Order Runge-Kutta*

SDIC Sensibilidad Dependiente de las Condiciones Iniciales - *Sensitive Dependence in Initial Conditions*

SNR Relación Señal a Ruido - *Signal-to-Noise Ratio*

STFT Transformada de Fourier de Tiempo Reducido - *Short-Time Fourier Transform*

VOA Atenuador Óptico Variable - *Variable Optical Attenuator*

VSA Vector Signal Analizer

WT Wavelet Transform

Bibliografía

[1] M. Li-xin, "Weak Signal Detection Based on Duffing Oscillator," *2008 International Conference on Information Management, Innovation Management and Industrial Engineering*, pp. 430–433, Dec. 2008.

[2] A. A. Gokhale, *Introduction to Telecommunications*. pp. 46-50, 2004. Thomson Delmar Learning.

[3] G. Wang, D. Chen, J. Lin, and X. Chen, "The application of chaotic oscillators to weak signal detection," *Industrial Electronics, IEEE Transactions on*, vol. 46, pp. 440–444, Apr. 1999.

[4] T. Liao and S. Tsai, "Adaptive synchronization of chaotic systems and its application to secure communications," *Chaos, Solitons & Fractals*, vol. 11, no. 9, pp. 1387 – 1396, 2000.

[5] J. Wang, J. Zhou and B. Peng, "Weak signal detection method based on duffing oscillator," *Kybernetes*, vol. 38, no. 10, pp. 1662–1668, 2009.

[6] D. Liu, H. Ren, L. Song and H. Li, "Study of weak signal detection based on second FFT and chaotic oscillator," *Nature and Science*, vol. 3 No. 2, pp. 59–64, 2005.

[7] Z. Peng and F. Chu, "Application of the wavelet transform in machine condition monitoring and fault diagnostics: a review with bibliography," *Mechanical Systems and Signal Processing*, vol. 18, pp. 199–221, 2004.

[8] Y. Fan and G. Zheng, "Research of high-resolution vibration signal detection technique and application to mechanical fault diagnosis," *Mechanical Systems and Signal Processing*, vol. 21, pp. 678–687, 2007.

[9] R. Brown, L. Chua and B. Popp, "Is sensitive dependence on initial conditions nature's sensory device?," *International Journal of Bifurcation and Chaos (IJBC) in Applied Sciences and Engineering*, vol. 2, pp. 193–199, 1992.

[10] L.E. Larson, J.M. Liu and L.S. Tsimring, *Digital Communications Using Chaos and Nonlinear Dynamics.* pp. 15-24, Springer, 2006.

[11] S. Wiggins, *Introduction to applied nonlinear dynamical systems and chaos.* pp. 77-82 & 717-719, Springer, 2003.

[12] H. Jin and K. Wang, "Carrier Detection Method of Binary-Phase-Shift-Keyed and Direct-Sequence-Spread-Spectrum Signals Based on Duffing Oscillator," *ITS Telecommunications Proceedings, 6th International Conference*, no. 3, pp. 5–8, 2006.

[13] H. Yiran, Y. Chengqun, L. Li, and B. Hongbo, "A Method of 2FSK Signal Detection Using Duffing Oscillator," *2008 ISECS International Colloquium on Computing, Communication, Control, and Management*, pp. 510–513, Aug. 2008.

[14] J. Felix and J.M. Balthazar, "A note on a control strategy for duffing oscillator type by using a kind of non-ideal excitation and saturation phenomenon,"

[15] Q. Gao, T. Zhang, Y. Huang and L. Yu, "Simulation studies on chaos demodulation weak bpsk signal based on simulink tools," *Journal of Computer Applications*, vol. 19, pp. 3211–3214, 2009.

[16] E. Jackson, *Perspectives of Nonlinear Dynamics.* Cambridge University Press, pp. 343-348, 1990.

[17] G. Wang, "Estimation of amplitude and phase of a weak signal by using the property of sensitive dependence on initial conditions of a nonlinear oscillator," *Signal Processing*, vol. 82, pp. 103–115, Jan. 2002.

[18] J. Guckenheimer and P.J. Holmes, "Nonlinear oscillations, dynamical systems and chaos," *Springer*, 1983.

[19] C. Holmes and P. Holmes, "Second order averaging and bifurcations to subharmonics in duffing's equation," *Journal of Sound and Vibration*, vol. 78, no. 2, pp. 161–174, 1981.

[20] M. Cai and J. Yang, "Bifurcation of periodic orbits and chaos in duffing equation," *Acta Mathematicae Applicatae Sinica, English Series*, vol. 22, no. 3, pp. 495–508, 2006.

[21] M. R. Roussel, "Hamiltonian systems," *University of Lethbridge*, pp. 1-9, 2004.

[22] E. Simiu and M. Frey, "Melnikov function and homoclinic chaos induced by weak perturbations," *Phys. Rev. E*, vol. 48, pp. 3190–3192, Oct 1993.

[23] K. Knopp, *Theory of Functions, Parts I and II.* Dover publications, pp. 32-44, 1996.

[24] Wang, J.X. and C. Hou, "A Method of Weak Signal Detection Based on Duffing Oscillator," *2010 International Conference on e-Education, e-Business, e-Management and e-Learning*, pp. 387–390, Jan. 2010.

[25] J. Li and Y. Shen, "Oscillators Array," *Science And Technology*, no. 1, pp. 1–4, 2009.

[26] D.L. Birx, "Chaotic oscillators and CMFFNS for signal detection in noise environments," *IEEE International Joint Conference on Neural Networks*, pp. 881–888, 1992.

[27] J. Han and P. Zheng, "A Chaotic Duffing Receiving System Based on OOK Digital Modulation," *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–3, Oct. 2008.

[28] Z. Xiangyang, L. Chongxin, L. Junhua and L. Weiping, "A new method for frequency detection based on chaos," *Journal of Data Acquisition & Processing*, vol. 17, pp. 183–186, 2002.

[29] Y. Xu and C. Yang, "The Study of Weak Signal Detection Based on Chaotic Oscillator," *Communications, Circuits and Systems, 2007. ICCCAS 2007. International Conference*, pp. 693–696.

[30] O. Hanna, "New explicit and implicit methods for the integration of ordinary differential equations," *Computers & Chemical Engineering*, vol. 12, no. 11, pp. 1083 – 1086, 1988.

[31] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in FORTRAN: The Art of Scientific Computing.* 2nd ed. Cambridge, England: Cambridge University Press, 1992.

[32] Courant-Friedrichs-Lewy-Condition, "http://mathworld.wolfram.com/EulerForwardMethod.html".

[33] Trapezoid Rule, "http://pathfinder.scar.utoronto.ca/dyer/csca57/bookp/node41.html."

[34] Q. I. Rahman and G. Schmeisser, "Characterization of the speed of convergence of the trapezoidal rule," *Numerische Mathematik*, vol. 57, no. 1, pp. 123–138, 1990.

[35] G. Wang and S. He, "A quantitative study on detection and estimation of weak signals by using chaotic duffing oscillators," *IEEE Trans. Circuits Syst. I*, vol. 50, no. 7, pp. 945–953, 2003.

[36] L. Wei-bo and G. Wei, "Communication system using fsk modulation under low voltage," *Journal of Shao Xing University*, vol. 8, pp. 48–51, 2005.

[37] A. Valdovinos, "Transmission theory." Lecture notes in the course Data Transmission, University of Zaragoza 2007.

[38] B. Widrow and I. Kollár, *Quantization Noise: Roundoff Error in Digital Computation, Signal Processing, Control, and Communications*. Cambridge, UK: Cambridge University Press, 2008.

[39] J. G. Proakis and D. G. Manolakis, *Digital signal processing ; principles, algorithms, and applications*. Upper Saddle River, NJ: Pearson, 2007.

[40] L. Tian-liang, "Frequency Estimation for Weak Signals Based on Chaos Theory," *2008 International Seminar on Future BioMedical Information Engineering*, pp. 361–364, Dec. 2008.

[41] T. Wong and T. Lok, *Theory of Digital Communications*. Available online.

[42] Y. Cai and A. Pilipetskii, "Channel capacity of fiberoptic communication systems with amplified spontaneous emission noise," *Conf Opt Fiber Commun Tech Dig Ser*, vol. 4, pp. 655–657, 2005.