Análisis e implementación de un ecualizador usando GNU radio companion

Andrés Felipe Amaya Pedraza
Angie Daniela Sierra Moyano
Sergio Andrés Ávila Peláez

Universidad Industrial de Santander

Bucaramanga, Santander, Colombia

April 27, 2018

Como proyecto de la asignatura Comunicaciones II se plantea un aporte para la sección 18.4 del libro del docente analizando los conceptos relacionados con la ecualización de canal e implementándolos en un bloque de GNU radio companion usando Python.

1 ¿Qué es ecualización?

Para empezar a hablar de ecualización debemos remontarnos al esquema básico de las comunicaciones (Figura 1) en el que se tienen tres elementos principales que son: Transmisor, canal y receptor,el primero de ellos es el encargado de enviar el mensaje que irá por el canal o medio para llegar a un receptor o destino.

Figure 1: Esquema básico de las comunicaciones

En este proceso el mensaje viaja siguiendo una línea de flujo horizontal desde el transmisor al receptor,donde se esperaría que la señal enviada sea muy similar o en el mejor de los casos igual a la recibida evitando pérdidas de información entre los dos elementos.
Sin embargo,en la vida real cuando la señal pasa por el canal se ve afectada por distintos sucesos que se conocen como fenómenos de canal y son causados por el medio de transmisión. Algunos de estos pueden ser corregidos usando un ecualizador, el cual, por medio de un algoritmo de aprendizaje, encuentra una respuesta al impulso tal que al convolucionarla con la señal que ha pasado por el canal, permite obtener la señal antes de su paso por el mismo. Es decir, viendo al canal como un sistema, el ecualizador funciona como la función que da solución a la convolución inversa entre el mensaje y el canal, permitiendo eliminar los efectos del mismo.

Finalmente y para intentar expresar más puntualmente una descripción lo que significa un ecualizador en un sistema de comunicaciones podemos decir que este es un elemento que permite revertir el efecto del canal en la señal mensaje, obteniendo por métodos adaptativos una función que describe el comportamiento del

canal y posteriormente desarrola una convolución inversa entre dicha señal y la recibida de tal manera que se pueda obtener el mensaje que fue transmitido en su totalidad o una aproximación más excata del mismo.

2 ¿Cómo funciona un ecualizador?

Una de las principales tareas del receptor es liberarse del efecto del canal para poder obtener la señal transmitida de la forma más limpia posible. Una solución es agregar un ecualizador en la etapa de canal para implementar en la parte receptora un sistema con respuesta inversa a la respuesta del canal permitiendo que de alguna forma el fenómeno se contrarreste.

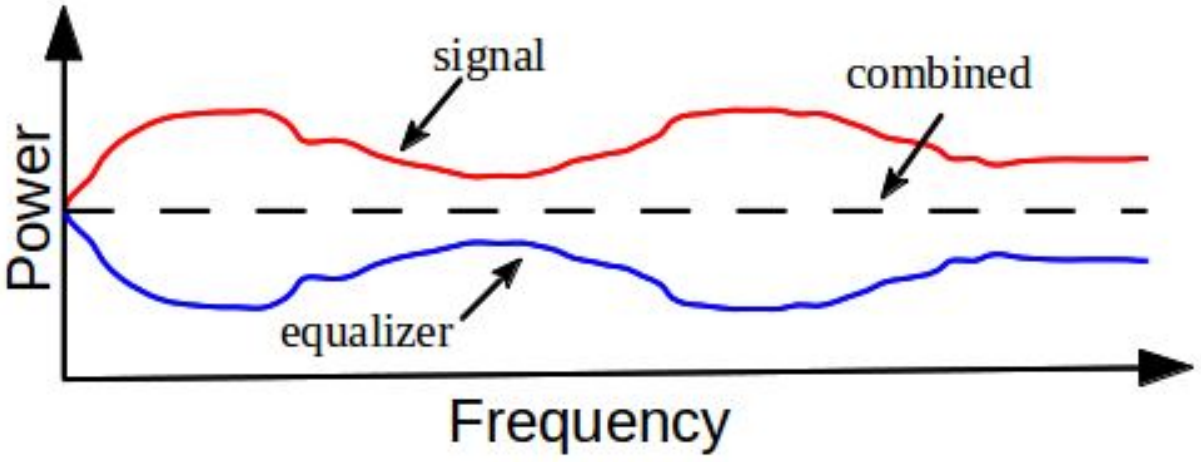

Figure 2: Funcionamiento de un ecualizador. (Imagen tomada del libro.)

El principio de funcionamiento del ecualizador se basa en obtener un vector Wk llamado pesos que se adapta de forma que elimine el efecto producido por el canal, este procedimiento se lleva a cabo de la siguiente forma:

A partir de una señal de referencia conocida, se realiza una convolución entre la señal de referencia (una vez ha pasado por el canal) y el vector de pesos wk. A la señal producida por esta convolución se le llama salida, y se define entonces al error como la diferencia entre la salida y la señal de referencia (sin pasar por el canal), esto significa que se está midiendo de manera cuantitativa que tanto se parecen la salida del ecualizador y la señal mensaje antes de su paso por el canal, en otras palabras, se mide que tan "bien" realiza su trabajo el ecualizador. Esta señal de error es el argumento para una función de optimización, la cual se múltiplica por un factor conocido como velocidad de aprendizaje, y este resultado se suma a los pesos. En otras palabras, teniendo en cuenta el error, se modifican los valores del vector wk de forma que la señal error tienda a hacerse lo más pequeña posible.

Un ecualizador se compone de 3 elementos principales: Filtro digital, algoritmo adaptativo y un generador de entrenamiento, los parametros de estos varían dependiendo del tipo de ecualizador implementado.

La idea principal es durante los primeros ciclos hacer un "entrenamiento" hasta obtener los pesos que permitan reproducir una señal inversa al modelo del canal para de esta forma recuperar la señal transmitida. Esto de puede ver como un sistema de control en lazo cerrado en el cual se utiliza la realimentación con el objetivo de minizimar el error reajustando iterativamente los parámetros del sistema hasta obtener la respuesta deseada. LA velocidad de aprendizaje en el entrenamiento del ecualizador guarda un parecido fuerte con el parámetro beta en un sistema de lazo cerrado, puesto que implica el efecto o sensibilidad del cambio ante el error.

3 ¿Para qué se utiliza un ecualizador?

El paso por el canal de una señal puede implicar pérdida parcial o total de la información del mensaje de la señal. Este es un grave problema pues en el mundo real el canal siempre está presente, y sus efectos pueden llegar a convertir nuestro mensaje en algo totalmente indistinguible del ruido, es por esto que la idea de un ecualizador que permita "eliminar" el efecto del paso por el canal resulta tan atractiva.

A pesar de los múltiples efectos que produce el canal en la señal, y de la complejidad matemática que puede acarrear el tratamiento individual de los mismos debido a las correlaciones que pueden darse entre estos efectos, el ecualizador a implementar permite eliminar estos efectos en conjunto sin tener en cuenta las complicaciones mencionadas. Este aspecto es de suma importancia en el por qué utilizarlo, puesto que es posible implementar una corrección individual para cada fenómeno del canal por separado. Sin embargo, esto implica no solo múltiples bloques de procesamiento, si no una alta complejidad matemática en los mismos, lo cual podría dificultar futuras modificaciones a los bloques.

4 Implementación:

Para la creación del bloque en GNU radio companion es necesario seguir los pasos que se mencionarán a continuación desde la consola:

Creación del módulo:

$ gr_modtool newmod Ecualizador

Ingresamos a la carpeta en la que fue creado el módulo:

$ cd gr-Ecualizador

Creamos el bloque:

$gr_modtool add -N Ecualizador -l python –t sync

Posteriormente se piden los parámetros a asignar en el bloque y esos son: taps, gain y train.

Entramos a la carpeta de python donde fue creado el módulo y creamos un archivo llamado CodigoEqualizador.py, el cual se compone de:

Definición de la clase:

```
import numpy as np
from gnuradio import gr
```

Figure 3: Librerías requeridas para la creación del ecualizador en GNU radio desde python.

Creamos la clase asigando el parametro gr.sync_blink:

```
class blk(gr.sync_block):
```

Figure 4: Creación de la clase BLK

Es necesario recordar que este parametro hace referencia a un tipo de bloque que envia de forma sincrona entradas y salidas en una relacion 1:1

Inicializamos y asignamos las variables o parámetros que pondremos en el bloque en la función __init__:

```
def __init__(self,taps=5,gain=0.5,train=20):  # only default arguments here
    gr.sync_block.__init__(
        self,
        name='Ecualizador LMS Final',
        in_sig=[np.complex64,np.complex64],
        out_sig=[np.complex64]
    )
    #self.w=np.full((taps,1),0,dtype=np.complex64)
    self.w=np.zeros((taps,1),dtype=np.complex64)
    self.alpha=gain
    self.N=taps
    self.train=train
    self.count=0
```

Figure 5: Inicialización

En la función work se hace el entrenamiento para obtener el vector de pesos, hasta que train=1 de la forma:

```
  def work(self, input_items, output_items,):

      x=input_items[0]
      d=input_items[1]
      if(self.count<=self.train):
              print 'Entrada entrenamiento'
              for i in range(self.N,len(x)):
                      xt=x[(i-self.N):i]
                      y=self.w.T.dot(xt)
                      e=d[i-1]-self.w.T.dot(xt)
                      e2=abs(e)
                      print e2
                      w1 = self.w.T + self.alpha*(e*(xt.conjugate()))
                      self.w=w1.T
                      #output_items[0][i]=y[0]
      else:
              for i in range(self.N,len(x)):
                      print 'Entrada convolucion'
                      xt=x[(i-self.N):i]
                      y=self.w.T.dot(xt)
                      #e=d[i-1]-y
                      #self.w = self.w + self.alpha*(e*(xt.conjugate()))
                      output_items[0][i]=y[0]
      print self.count
      self.count=self.count+1
      return len(output_items[0])
```

Figure 6: Work

Anexos. Se adjunta un archivo que contiene el diagrama de bloques en GNU RADIO y la creación del bloque para el ecualizador en Python.