Nrsc-am-cochannel-simulator

Detailed documentation for the “graveyard.m” simulation software

Version 2.6 – April 17, 2022

Author: David L. Hershberger (on Github: w9gr)

Introduction

GNU Octave code to simulate AM cochannel reception, with and without carrier synchronization. Produces audio and graphics files.

This software is used to subjectively evaluate the effects of carrier synchronization on cochannel interference. It runs under GNU Octave, which is a free open source matrix math tool. Versions are available for Linux, macOS, BSD, and Microsoft Windows.

GNU Octave is available here: https://www.gnu.org/software/octave/

GNU Octave scripts are interpreted, not compiled. **So you must install GNU Octave to run the source code in this repo.**

The software reads from a number of flac audio files to produce a desired signal with a number of interferers. The number of interferers can be from 1 to 71. The interferers may be set to randomly selected amplitudes or their amplitudes may be specified to simulate reception at a specific location of a specific station. Each interferer fades in both amplitude and phase.

The audio files are all 60 seconds in length. The desired signal is a talk show, chosen because the speaker pauses between phrases and words, making it possible to hear the interferers.

The flac audio file format is lossless. So it is just as accurate as a WAV file, but file size is smaller than WAV. Most audio players (such as VLC) can play the flac format.

The software includes a typical narrowband envelope detector receiver model with fast AGC.

The output files are generated in both mono and stereo formats. In the stereo files, one channel has synchronized carriers and the other channel does not, making it possible to hear both simultaneously.

There are also files where the desired signal is mostly canceled by subtracting it, leaving just the interferers. The cancellation is not perfect because envelope detection is nonlinear.

The software also generates graphics, showing the close-in RF spectra, the interferer (fading) amplitudes, the interferer (fading) phases, and the receiver AGC for both synchronous and nonsynchronous operation.

Instant Gratification

Although everything you need to do your own co-channel simulations is here, you may not need to do so. Sample output files have already been generated for 3 and 64 interferers, at a signal to interference ratio (SIR) of 10 dB, a SNR of 20 dB, and 75 microsecond emphasis enabled. All you have to do is download and play some audio files. The output files which are here include:

cochannel03.flac

cochannel03qrm.flac

cochannel03sync.flac

cochannel03nonsync.flac

cochannel03syncqrm.flac

cochannel03nonsyncqrm.flac

There are also seven graphics files produced:

cochannel03specsync.png

cochannel03specnonsync.png

cochannel03agc.png

cochannel03ampqrm.png

cochannel03ampqrmsum.png

cochannel03phaseqrmnonsync.png

cochannel03phaseqrmsync.png

The files above are for three interferers at 10 dB SIR. There are also simulation files for 64 interferers at 10 dB SIR. The file names are the same as above except that the "03" in each file name is replaced with "64." See the "files" section below for an explanation of what each of the example files contains.

If you want a different number of interferers, or a different SIR, then you will have to run the simulation yourself.

Requirements

You will need a computer running GNU/Linux, BSD, Windows, or macOS. To run the simulation you will need to install the version of GNU Octave for your operating system. It may be downloaded here:

https://www.gnu.org/software/octave/download

For many Linux systems, pkgs.org can direct you to the proper site for Octave downloads for your distribution.

If your download does not include the "packages," note that you will need the ltfat and signal packages. Those are installed after Octave has been installed, by typing "pkg install [package-name]" at the Octave command prompt. Regardless of which OS you use, the packages are downloaded here:

https://octave.sourceforge.io/packages.php

Theory and Method

The simulation is done at baseband. The RF carrier frequency is zero. Each signal begins as a real signal consisting of the baseband audio plus 1, where the 1 represents the carrier. Each real signal is spectrally symmetric about zero frequency.

If an interfering signal is not synchronous, it is multiplied by a complex sinusoid. This creates an imaginary component and positive and negative frequency components are no longer symmetric.

Phase shifts are produced by multiplying a signal by a complex constant. Fading of amplitude and phase is produced by multiplying a signal by a slowly varying complex value.

The signal to be received consists of a desired signal plus a group of undesired interfering signals, which may be fading in amplitude and phase, and which may be slightly off-frequency.

Reception is done by a 2.5 kHz lowpass filter followed by an envelope detector. The envelope detector forms the square root of the sum of the squares of the real and imaginary components of the combined signals. Fast AGC/AVC is also applied, as it would be in a typical real AM broadcast receiver. The use of AGC will demonstrate AGC pumping in the presence of co-channel beats. The subsonic beats themselves are removed by a highpass filter, also typical of real AM receivers.

Files

There are a number of files which are required to run the simulator. These files include GNU Octave scripts, command and configuration files, and data files. Output files include graphics in PNG format, and audio files in flac format.

The script files are:

graveyard.m – This is the main program. This is script which you will run to perform a simulation.

addsig.m – This is a function called by graveyard.m. Each time it is called, it adds an interfering signal to the total signal.

makeparams.m – This function produces random values for carrier frequency error, interferer amplitude, and a set of fading functions. This script does not normally need to be executed, because a set of random values and fading functions have already been generated. It is completed in case a different set of random parameters is desired, perhaps with different distributions or statistics. This function produces a file called “parameters.txt” which includes the frequency errors and amplitudes, and 71 flac files that contain complex fading functions. The fading functions are complex, which means that both amplitude and phase of each interferer vary with time. The fading files are named “fadeXX.flac” where the XX is a two digit number varying from 01 to 71, which associates the file with one of the 71 possible interfering signals.

The command and configuration files are:

config.txt – This file is used to select the operating mode of graveyard.m, and sets some basic parameters. It selects whether or not NIF mode is invoked (see below), it identifies which audio files are to be used for the interferers, and it sets the average signal to interference ratio (desired to undesired ratio) in decibels. It selects either flat audio files or 75 microsecond pre-emphasized audio files. It sets the signal to noise ratio (SNR) in decibels. If SNR>90 then no Gaussian noise is added. For the NIF mode, it selects the percent skywave amplitude. This is typically 10% or 50%. It expresses the time when the fading amplitude exceeds the specified value. The comments in the config.txt file explain the meaning of each parameter in more detail.

nif.txt – This file is used to set amplitudes of interferers in the “NIF” mode. In NIF mode, interferers are set to specific average amplitude levels. In the non-NIF mode, the interferer amplitudes are set randomly. Values in nif.txt may be entered as linear or in decibels. Positive values, including zero, are assumed to be linear values. So if you enter 0.1, the interferer will be set to -20 dB. Negative values are assumed to be in decibels. You may “mix and match” linear and dB values.

There are example config.txt and nif.txt files included. Both files contain comments, to help set the values correctly. Comment lines for GNU Octave data files begin with the “#” character. Any line beginning with # will be ignored. The comments may be deleted in any files the user creates.

The data files are:

audioXX.flac – These are audio files used for the desired and undesired signals. The desired signal is always taken from audio01.flac. The undesired signals are taken from the files audio02.flac through audio72.flac. These are 16 bit integer monophonic audio files with the peaks set accurately to 0.9 times full scale (i.e. 29491). The graveyard.m file will produce 100% envelope modulation when the flac file audio is 90% of full scale.

fadeXX.flac – These are stereo audio files containing the complex fading functions for the interferers. The left channel contains the real part and the right channel contains the imaginary part. These signals are very slowly varying, producing both RF phase fading and amplitude fading. The fading function is multiplied by the interfering signal to simulate skywave propagation.

qrn.flac – This is a stereo audio file containing the complex Gaussian noise signal which may be injected into the signal if the SNR value is set to a value less than 90 dB.

Output files:

There are six audio files produced by this script. They begin with the string "cochannelXX" where XX is the number of interferers. So you would have "cochannel03" or "cochannel64" for 3 or 64 interferers, respectively. If NIF mode is invoked, the file names will start with "nifXX" instead.

The files are:

cochannelXX.flac - stereo file, synchronous case on left channel, nonsynchronous case on the right channel

cochannelXXqrm.flac - stereo file, with just the interference signal. The envelope signal is canceled, but since envelope detection is nonlinear, the cancellation is not perfect, so the envelope detector nonlinear distortion will appear in this file. Synchronous on left, nonsynchronous on right.

cochannelXXsync.flac - monophonic file, synchronous case

cochannelXXnonsync.flac - monophonic file, nonsynchronous case

cochannelXXsyncqrm.flac - monophonic file, synchronous case, interference only

cochannelXXnonsyncqrm.flac - monophonic file, nonsynchronous case, interference only

There are also seven graphics files produced:

cochannelXXspecsync.png - close in spectrum of synchronous spectrum

cochannelXXspecnonsync.png -0 close in spectrum of nonsynchronous spectrum

cochannelXXagc.png - shows receiver AGC signal for both synchronous and nonsynchronous cases.

cochannelXXampqrm.png - shows amplitude versus time for interferers (both synchronous and nonsynchronous)

cochannelXXampqrmsum.png - shows amplitude versus time for sum of all interferers (both synchronous and nonsynchronous)

cochannelXXphaseqrmnonsync.png - shows phase versus time for nonsynchronous interferers.

cochannelXXphaseqrmsync.png - shows phase versus time for synchronous interferers.

Summary

To run a simulation with random interferers, edit the file config.txt. Choose the SIR you want and the number of interferers you want, which will be determined by nfile1 and nfile2 in the config.txt file. You may also set your SNR value, which will inject Gaussian noise into the signal. Then at the Octave prompt, type “graveyard.” All of the required files must be in the working directory. Octave will then run, and any previously generated output files of the same names will be overwritten.

To simulate with specified interferer amplitudes (NIF mode) do the same except set NIF to 1 in config.txt, and then specify the interferer amplitudes in the nif.txt file. The number of interferers will be set by the number of values you provide in the nif.txt file. Be sure you do not overrun the number of available interferers (71) as there is no error checking for this condition.

Supporting Organizations

The National Radio Systems Committee (NRSC) gratefully acknowledges the work of the author, David L. Hershberger. The National Radio Systems Committee (NRSC, https://nrscstandards.org) is the AM/FM technical standards body for North America. NRSC is jointly sponsored by the National Association of Broadcasters (NAB) and the Consumer Technology Association (CTA). Its purpose is to study and make recommendations for technical standards that relate to radio broadcasting and the reception of radio broadcast signals.