Datasets of Indoor Wireless Channel Measurements – v1.0

Adriano Pastore, Armin Ghani

June 2, 2021

This dataset was generated as part of the experimental activities of the ARISTIDES project (`http://www.cttc.es/project/aristides/`), which is funded by the Spanish Ministry of Economy and Competitiveness under grant RTI2018-099722-B-I00 (ARISTIDES).

1 Purpose

This is a very simple dataset, consisting of raw IQ measurements of a wireless indoor channel. It is intended primarily for researchers without access to software-defined radio equipment, and should be helpful for understanding and investigating fundamental properties of real wireless channels (e.g., for channel modeling).

2 Overview

The datasets contain several measurements of a wireless channel between a transmitting node and a receiving node in a static indoor lab environment.

- Both the sender and transmitter are equipped with a single vertical omnidirectional antenna.
- Two types of transmit signals are considered:
  - random bit sequences modulated with quadrature phase shift keying (QPSK)
  - pseudorandom Gaussian samples
- We diversify the measurement conditions by varying:
  1. the positions of sender and receiver
  2. the modulation frequencies
- At the receiver, we record raw in-phase/quadrature (IQ) baseband samples
- The purpose of this dataset is to provide wireless channel measurements that reflect hardware imperfections such as
  - non-linear distortions of transmitter power amplification
  - non-linearities of the channel
  - a mismatch of oscillator frequencies
  - quantization noise

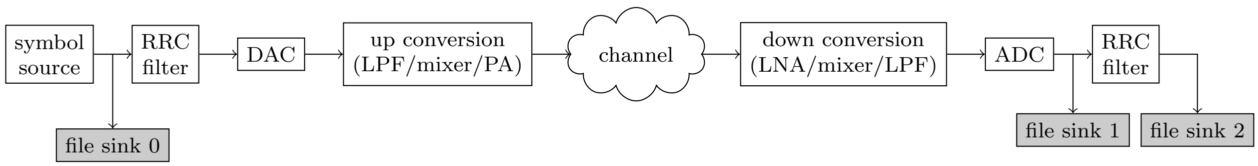


Figure 1: System block diagram

3 Setup

The simplified block diagram of the setup used for our channel measurements is depicted in Figure 1. The radio frontends are implemented by software-defined radio modules (Ettus Research USRP X310). Both nodes employ a single vertical antenna (Ettus Research VERT2450 dual band antenna). The parameters used for the measurements are summarized in the following two tables.

Parameter	Math. symbol	Value	Relationship
Bitrate at TX	f_{bits}	1 Mbps	
Modulation order (QPSK)	m	4	
Symbol rate at TX	f_{symb}	0.5 MS/s	
Roll-off factor	α	0.2	
Bandwidth	f_{BW}	0.6 MHz	$(1+\alpha)f_{\text{symb}}$
RRC interpolation factor	β_{int}	2	
RRC decimation factor	β_{dec}	4	
Number of taps of RRC filter		256	
Sample rate at TX (max. 200 MS/s)	$f_{\text{sample,TX}}$	1 MS/s	$\beta_{\text{int}} f_{\text{symb}}$
Sample rate at RX ($\geq f_{\text{sample,TX}}$)	$f_{\text{sample,RX}}$	2 MS/s	
Power amplifier (PA) gain at TX		0 dB	
Low noise amplifier (LNA) gain at RX		0 dB	

Table 1: Common parameters (for all measurements)

Signal-to-noise ratios	0 dB / 10 dB / 20 dB (approximate values)
Modulations	QPSK / pseudorandom Gaussian
Center frequencies	433.92 MHz / 708 MHz / 2450 MHz

Table 2: Variable parameters

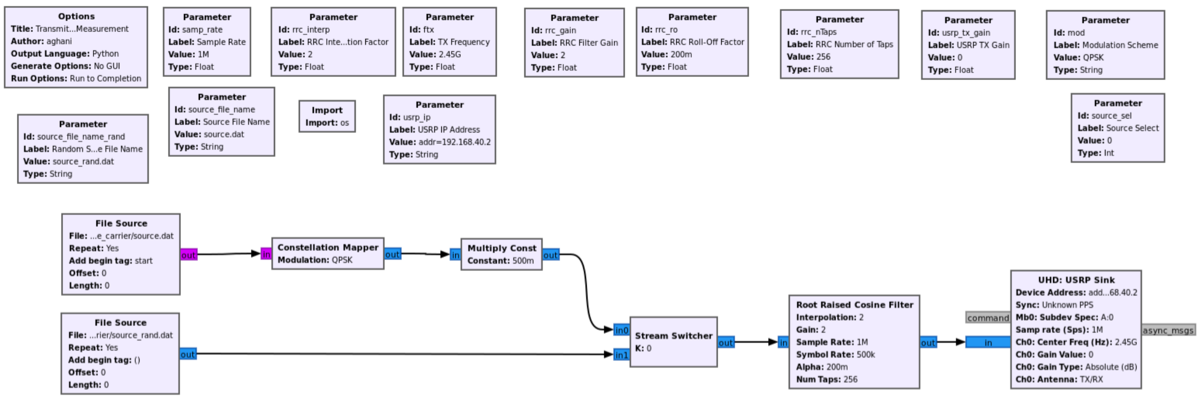
Concerning the chosen frequency bands:

- 433.92 MHz is the center of an ISM band (433.05–434.79 MHz, Region 1)
- 708 MHz is the center of the 5G New Radio band n12 (699 MHz–716 MHz)
- 2450 MHz is an IEEE 802.11a/b/g/n frequency band for WiFi

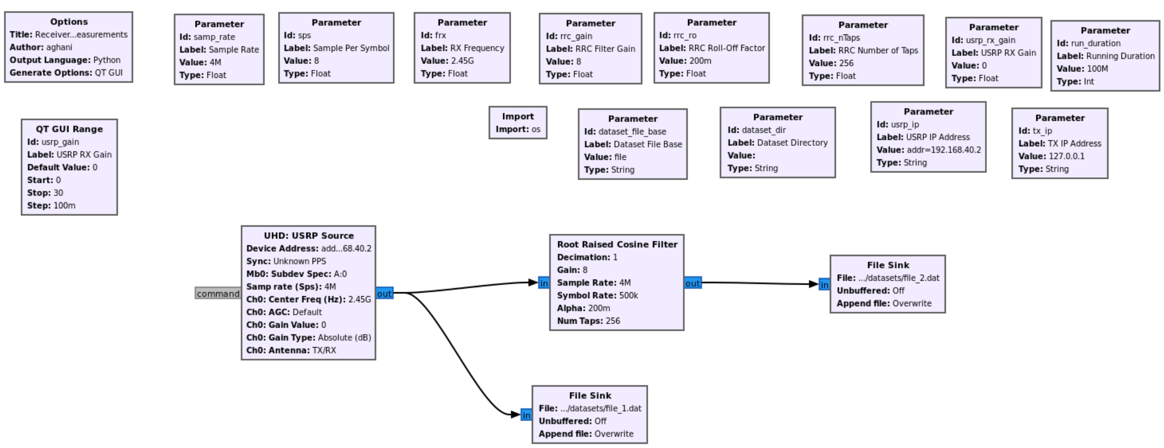
4 Implementation in GNU Radio

The block diagram of transmitter and receiver were generated with GNU Radio Companion, and are displayed in Figure 2 on the next page.

(a) Transmitter (b) Receiver (c) Transmitter and receiver close by



(a) Transmitter



(b) Receiver

Figure 2: GNU Radio block diagrams of transmitter and receiver processing chains

5 File structure and how to import

5.1 Source

The source sequences are stored in the files `source_QPSK.dat` and `source_Gaussian.dat`.

The former contains a sequence of 1 million random bits that are Gray code mapped to 500,000 complex QPSK symbols, which were transmitted repeatedly (in a loop) during each channel measurement.

The latter contains an overlong sequence of complex Gaussian samples with zero mean and a variance of 1/4 per real/imaginary component, hence a complex variance of 1/2.[1]

For each measurement, samples were recorded for approximately 10 seconds.

For each parameter configuration, the exact measurement parameters are stored in a file `config.txt` and the dataset consists of two files: `file 1.dat` and `file 2.dat`.

Dataset files are saved in raw binary format. The samples are interlaced in the following way: the symbols on odd positions belong to the in-phase (I) signal, the symbols on even positions belong to the quadrature (Q) signal.

The datasets are organized in a collection of files, each compressed in `tar.gz` format and have self-explanatory names.

The files `file 1.dat`, `file 2.dat` and `source_Gaussian.dat` can be imported in MATLAB and Python with the following commands.[2]

5.2 Python

```
# display N number of samples stored in dataset file
import numpy as np
# File directory
file_name = "f433_snr10_gaussian/file_1.dat"
# Number of samples to be read
N = 10000000

fd = open(file_name, 'rb')
# Complex data
data_complex = np.fromfile(fd, dtype=np.complex64, count=N)
```

Listing 1: Importing data in Python

5.3 Matlab

```
% display N number of samples stored in dataset file
% File directory
file_name = "datasets/pos1_f433_gaussian/file_1.dat";
% Number of samples to be read
N = 10000000;

fd = fopen(file_name, 'r');
data = fread(fd, N*2, 'float32');
% Complex data
data_complex = data(1:2:end) + 1i*data(2:2:end);
```

Listing 2: Importing data in Matlab

Important note: During approximately one second at the beginning of each measurement, the transmitter is silent. This is intentional, as it serves to let transient effects fade out, and allows to estimate the background noise (and therefore the signal-to-noise ratio) post-measurement.

6 How to cite and acknowledge

Please cite this dataset as indicated on the Zenodo website. If you use this dataset, we appreciate if you drop us an e-mail at {`apastore,aghani`}`@cttc.es`. We thank you for any comments and feedback.

[1] This value is chosen to mitigate (but not entirely remove) the effect of saturation for the Gaussian case, for which the symbol amplitude that is fed to the DAC, is potentially unbounded

[2] For importing the file `source_QPSK.dat`, which is in binary format, replace the complex/float data type by `uint8`